

# calibre User Manual

Release 8.7.0

**Kovid Goyal** 

July 18, 2025

# CONTENTS

1	The Graphical User Interface	3
2	Adding your favorite news website	33
3	The E-book viewer	53
4	E-book conversion	61
5	Editing e-books	79
6	The calibre Content server	111
7	Comparing e-books	119
8	Editing e-book metadata	123
9	Frequently Asked Questions	127
10	Tutorials	151
11	The calibre:// URL scheme	249
12	Customizing calibre	253
13	Command Line Interface	301
14	Setting up a calibre development environment	357
15	Digital Rights Management (DRM)	387
16	Glossary	389
Py	thon Module Index	391
Inc	lex	393

calibre is an e-book library manager. It can view, convert and catalog e-books in most of the major e-book formats. It can also talk to many e-book reader devices. It can go out to the Internet and fetch metadata for your books. It can download newspapers and convert them into e-books for convenient reading. It is cross platform, running on Linux, Windows and macOS.

You've just started calibre. What do you do now? Before calibre can do anything with your e-books, it first has to know about them. Drag and drop a few e-book files into calibre, or click the "Add books" button and browse for the e-books you want to work with. Once you've added the books, they will show up in the main view looking something like this:

110	The Trouble With Physics	Lee Smolin	18 Mar 2011	0.9	*****
111	The Wise Man's Fear	Patrick Rothfuss	08 Mar 2011	1.4	****
112	The Heroes	Joe Abercrombie	08 Mar 2011	1.2	***

Once you've admired the list of books you just added to your heart's content, you'll probably want to read one. In order to do that you'll have to convert the book to a format your reader understands. When first running calibre, the *Welcome wizard* starts and will set up calibre for your reader device. Conversion is a breeze. Just select the book you want to convert then click the "Convert books" button. Ignore all the options for now and click "OK". The little icon in the bottom right corner will start spinning. Once it's finished spinning, your converted book is ready. Click the "View" button to read the book.

If you want to read the book on your reader, connect the reader to the computer, wait till calibre detects it (10-20 seconds) and then click the "Send to device" button. Once the icon stops spinning again, disconnect your reader and read away! If you didn't convert the book in the previous step, calibre will auto convert it to the format your reader device understands.

To get started with more advanced usage, you should read about *The Graphical User Interface* (page 3). For even more power and versatility, learn the *Command Line Interface* (page 301). You will find the list of *Frequently Asked Questions* (page 127) useful as well.

If you have more questions, or want to discuss calibre with other users or ask for help with specific things, there are forums and other help resources available<sup>1</sup>.

#### Sections

<sup>&</sup>lt;sup>1</sup> https://calibre-ebook.com/help

# THE GRAPHICAL USER INTERFACE

The Graphical User Interface (*GUI*) provides access to all library management and e-book format conversion features. The basic workflow for using calibre is to first add books to the library from your hard disk. calibre will automatically try to read metadata from the books and add them to its internal database. Once they are in the database, you can perform various *Actions* (page 4) on them that include conversion from one format to another, transfer to the reading device, viewing on your computer, and editing metadata. The latter includes modifying the cover, description, and tags among other details. Note that calibre creates copies of the files you add to it. Your original files are left untouched.

The interface is divided into various sections:

- Actions (page 4)
- Preferences (page 10)
- Catalogs (page 11)
- Search & sort (page 11)
- The search interface (page 12)
- Saving searches (page 17)
- Searching the full text of all books (page 18)
- Virtual libraries (page 18)
- Temporarily marking books (page 18)
- Guessing metadata from file names (page 18)
- Book details (page 20)
- Tag browser (page 22)
- Cover grid (page 25)
- Cover browser (page 26)
- Adding notes for authors, series, etc. (page 27)
- Quickview (page 28)
- Jobs (page 29)
- Keyboard shortcuts (page 29)

# 1.1 Actions



The actions toolbar provides convenient shortcuts to commonly used actions. If you right-click the buttons, you can perform variations on the default action. Please note that the actions toolbar will look slightly different depending on whether you have an e-book reader attached to your computer.

- Add books (page 4)
- *Edit metadata* (page 5)
- Convert books (page 6)
- View (page 6)
- Send to device (page 6)
- Fetch news (page 7)
- *Library* (page 7)
- *Device* (page 8)
- Save to disk (page 8)
- Connect/share (page 9)
- Remove books (page 10)

### 1.1.1 Add books



The Add books action has seven variations accessed by doing a right-click on the button.

- 1. Add books from a single folder: Opens a file chooser dialog and allows you to specify which books in a folder should be added. This action is *context sensitive*, i.e. it depends on which *catalog* (page 11) you have selected. If you have selected the *Library*, books will be added to the library. If you have selected the e-book reader device, the books will be uploaded to the device, and so on.
- 2. Add books from folders and sub-folders: Allows you to choose a folder. The folder and all its sub-folders are scanned recursively, and any e-books found are added to the library. You can choose whether to have calibre add all files present in a single folder to a single book record or multiple book records. calibre assumes that each folder contains a single book. All e-book files in a folder are assumed to be the same book in different formats. This action is the inverse of the *Save to disk* (page 8) action, i.e. you can *Save to disk*, delete the books and re-add them in single book per folder mode, with no lost information except for the date (this assumes you have not changed any of the setting for the Save to disk action).
- 3. Add multiple books from archive (ZIP/RAR): Allows you to add multiple e-books that are stored inside the selected ZIP or RAR files. It is a convenient shortcut that avoids having to first unzip the archive and then add the books via one of the above two options.

- 4. Add empty book (Book Entry with no formats): Allows you to create a blank book record. This can be used to then manually fill out the information about a book that you may not have yet in your collection.
- 5. Add from ISBN: Allows you to add one or more books by entering their ISBNs.
- 6. Add files to selected book records: Allows you to add or update the files associated with an existing book in your library.
- 7. Add data files to selected book records: Allows you to add any number of extra files that will be stored in a data sub-directory in the book directory. See *Adding extra data files to a book* (page 125) for details.
- 8. Add an empty file to selected book records: Allows you to add an empty file of the specified format to the selected book records.

The *Add books* action can read metadata from a wide variety of e-book formats. In addition, it tries to guess metadata from the filename. See the *Guessing metadata from file names* (page 18) section, to learn how to configure this.

To add an additional format for an existing book you can do any of three things:

- 1. Drag and drop the file onto the Book details panel on the right side of the main window
- 2. Right click the Add books button and choose Add files to selected books.
- 3. Click the *Add books* button in the top right area of the *Edit metadata* dialog, accessed by the *Edit metadata* (page 5) action.

### 1.1.2 Edit metadata



The Edit metadata action has four variations which can be accessed by doing a right-click on the

button.

- 1. Edit metadata individually: Allows you to edit the metadata of books one-by-one with the option of fetching metadata, including covers, from the Internet. It also allows you to add or remove particular e-book formats from a book.
- 2. Edit metadata in bulk: Allows you to edit common metadata fields for large numbers of books simultaneously. It operates on all the books you have selected in the *Library view* (page 11).
- 3. **Download metadata and covers**: Downloads metadata and covers (if available) for the books that are selected in the book list.
- 4. **Merge book records**: Gives you the capability of merging the metadata and formats of two or more book records. You can choose to either delete or keep the records that were not clicked first.
- 5. Manage data files: Manage the extra data files associated with the selected books.

For more details, see Editing e-book metadata (page 123).

### 1.1.3 Convert books



E-books can be converted from a number of formats into whatever format your e-book reader prefers. Many e-books available for purchase will be protected by *Digital Rights Management* (page 387) (*DRM*) technology. calibre will not convert these e-books. It is easy to remove the DRM from many formats, but as this may be illegal, you will have to find tools to liberate your books yourself and then use calibre to convert them.

For most people, conversion should be a simple one-click affair. If you want to learn more about the conversion process, see *E-book conversion* (page 61).

The Convert books action has three variations, accessed by doing a right-click on the button.

- 1. **Convert individually**: Allows you to specify conversion options to customize the conversion of each selected e-book.
- 2. Bulk convert: Allows you to specify options only once to convert a number of e-books in bulk.
- 3. Create a catalog of the books in your calibre library: Allows you to generate a complete listing of the books in your library, including all metadata, in several formats such as XML, CSV, BiBTeX, EPUB and MOBI. The catalog will contain all the books currently showing in the library view. This allows you to use the search features to limit the books to be catalogued. In addition, if you select multiple books using the mouse, only those books will be added to the catalog. If you generate the catalog in an e-book format such as EPUB, MOBI or AZW3, the next time you connect your e-book reader the catalog will be automatically sent to the device. For more information on how catalogs work, read the *Creating AZW3 EPUB MOBI catalogs* (page 241).

### 1.1.4 View



The *View* action displays the book in an e-book viewer program. calibre has a built-in viewer for many e-book formats. For other formats it uses the default operating system application. You can configure which formats should open with the internal viewer via *Preferences*  $\rightarrow$  *Interface*  $\rightarrow$  *Behavior*. If a book has more than one format, you can view a particular format by doing a right-click on the button.

### 1.1.5 Send to device



The Send to device action has eight variations, accessed by doing a right-click on the button.

- 1. Send to main memory: The selected books are transferred to the main memory of the e-book reader.
- 2. Send to card (A): The selected books are transferred to the storage card (A) on the e-book reader.
- 3. Send to card (B): The selected books are transferred to the storage card (B) on the e-book reader.

- 4. Send specific format to: The selected books are transferred to the selected storage location on the device, in the format that you specify.
- 5. Eject device: Detaches the device from calibre.
- 6. Set default send to device action: Allows you to specify which of the options, 1 through 5 above or 7 below, will be the default action when you click the main button.
- 7. Send and delete from library: The selected books are transferred to the selected storage location on the device and then **deleted** from the Library.
- 8. Fetch Annotations (experimental): Transfers annotations you may have made on an e-book on your device to the comments metadata of the book in the calibre library.

You can control the file name and folder structure of files sent to the device by setting up a template in *Preferences*  $\rightarrow$  *Import/export*  $\rightarrow$  *Sending books to devices*. Also see *The calibre template language* (page 158).

### 1.1.6 Fetch news



The *Fetch news* action downloads news from various websites and converts it into an e-book that can be read on your e-book reader. Normally, the newly created e-book is added to your e-book library, but if an e-book reader is connected at the time the download finishes, the news is also uploaded to the reader automatically.

The *Fetch news* action uses simple recipes (10-15 lines of code) for each news site. To learn how to create recipes for your own news sources, see *Adding your favorite news website* (page 33).

The Fetch news action has three variations, accessed by doing a right-click on the button.

- 1. Schedule news download: Allows you to schedule the download of your selected news sources from a list of hundreds available. Scheduling can be set individually for each news source you select and the scheduling is flexible allowing you to select specific days of the week or a frequency of days between downloads.
- 2. Add a custom news source: Allows you to create a simple recipe for downloading news from a custom news site that you wish to access. Creating the recipe can be as simple as specifying an RSS news feed URL, or you can be more prescriptive by creating Python-based code for the task. For more information, see *Adding your favorite news website* (page 33).
- 3. **Download all scheduled news sources**: Causes calibre to immediately begin downloading all news sources that you have scheduled.

### 1.1.7 Library



The *Library* action allows you to create, switch between, rename or remove a Library. calibre allows you to create as many libraries as you wish. You could, for instance, create a fiction library, a non-fiction library, a foreign language library, a project library, or any structure that suits your needs. Libraries are the highest organizational structure within calibre. Each library has its own set of books, tags, categories and base storage location.

1. Switch/create library...: Allows you to; a) connect to a pre-existing calibre library at another location, b) create an empty library at a new location or, c) move the current library to a newly specified location.

- 2. Quick switch: Allows you to switch between libraries that have been registered or created within calibre.
- 3. Rename library: Allows you to rename a Library.
- 4. Pick a random book: Chooses a random book in the library for you
- 5. Remove library: Allows you to unregister a library from calibre.
- 6. Export/import all calibre data: Allows you to either export calibre data for migration to a new computer or import previously exported data.
- 7. **Received** 7. **\* Constant** 7, 8 etc... give you immediate switch access between multiple libraries that you have created or attached to. This list contains only the 5 most frequently used libraries. For the complete list, use the Quick Switch menu.
- 8. Library maintenance: Allows you to check the current library for data consistency issues and restore the current library's database from backups.

#### Note

Metadata about your e-books, e.g. title, author, and tags, is stored in a single file in your calibre library folder called metadata.db. If this file gets corrupted (a very rare event), you can lose the metadata. Fortunately, calibre automatically backs up the metadata for every individual book in the book's folder as an OPF file. By using the Restore database action under Library Maintenance described above, you can have calibre rebuild the metadata.db file from the individual OPF files for you.

You can copy or move books between different libraries (once you have more than one library setup) by right clicking on the book and selecting the action *Copy to library*.

### 1.1.8 Device



The *Device* action allows you to view the books in the main memory or storage cards of your device, or to eject the device (detach it from calibre). This icon shows up automatically on the main calibre toolbar when you connect a supported device. You can click on it to see the books on your device. You can also drag and drop books from your calibre library onto the icon to transfer them to your device. Conversely, you can drag and drop books from your device onto the library icon on the toolbar to transfer books from your device to the calibre library.

### 1.1.9 Save to disk



The Save to disk action has five variations, accessed by doing a right-click on the button.

1. Save to disk: Saves the selected books to disk organized in folders. The folder structure looks like:

```
Author_(sort)
Title
Book Files
```

You can control the file name and folder structure of files saved to disk by setting up a template in *Preferences*  $\rightarrow$  *Import/export*  $\rightarrow$  *Saving books to disk*. Also see *The calibre template language* (page 158).

2. Save to disk in a single folder: Saves the selected books to disk in a single folder.

For 1. and 2., all available formats, as well as metadata, are stored to disk for each selected book. Metadata is stored in an OPF file. Saved books can be re-imported to the library without any loss of information by using the *Add books* (page 4) action.

- 3. Save only \*<your preferred>\* format to disk: Saves the selected books to disk in the folder structure as shown in (1.) but only in your preferred e-book format. You can set your preferred format in *Preferences* → *Interface* → *Behaviour* → *Preferred output format*
- 4. Save only \*<your preferred>\* format to disk in a single folder: Saves the selected books to disk in a single folder but only in your preferred e-book format. You can set your preferred format in Preferences → Interface → Behaviour → Preferred output format
- 5. Save single format to disk...: Saves the selected books to disk in the folder structure as shown in (1.) but only in the format you select from the popup list.

### 1.1.10 Connect/share



It also allows you to set up your calibre library for access via a web browser or email.

The Connect/share action has four variations, accessed by doing a right-click on the button.

- 1. **Connect to folder**: Allows you to connect to any folder on your computer as though it were a device and use all the facilities calibre has for devices with that folder. Useful if your device cannot be supported by calibre but is available as a USB disk.
- 2. Start Content server: Starts calibre's built-in web server. When started, your calibre library will be accessible via a web browser from the Internet (if you choose). You can configure how the web server is accessed by setting preferences at *Preferences*  $\rightarrow$  *Sharing*  $\rightarrow$  *Sharing over the net*
- 3. Setup email based sharing of books: Allows sharing of books and news feeds by email. After setting up email addresses for this option, calibre will send news updates and book updates to the entered email addresses. You can configure how calibre sends email by setting preferences at *Preferences → Sharing → Sharing books by email*. Once you have set up one or more email addresses, this menu entry will be replaced by menu entries to send books to the configured email addresses.

### 1.1.11 Remove books



The *Remove books* action **deletes books permanently**, so use it with care. It is *context sensitive*, i.e. it depends on which *catalog* (page 11) you have selected. If you have selected the *Library*, books will be removed from the library. If you have selected the e-book reader device, books will be removed from the device. To remove only a particular format for a given book use the *Edit metadata* (page 5) action. Remove books also has five variations which can be accessed by doing a right-click on the button.

- 1. Remove selected books: Allows you to permanently remove all books that are selected in the book list.
- 2. **Remove files of a specific format from selected books...**: Allows you to **permanently** remove e-book files of a specified format from books that are selected in the book list.
- 3. **Remove all formats from selected books, except...**: Allows you to **permanently** remove e-book files of any format except a specified format from books that are selected in the book list.
- 4. **Remove all formats from selected books**: Allows you to **permanently** remove all e-book files from books that are selected in the book list. Only the metadata will remain.
- 5. **Remove covers from selected books**: Allows you to **permanently** remove cover image files from books that are selected in the book list.
- 6. **Remove matching books from device**: Allows you to remove e-book files from a connected device that match the books that are selected in the book list.
- 7. Restore recently deleted: Allows you to undo the removal of books or formats.

#### Note

Note that when you use *Remove books* to delete books from your calibre library, the book record is deleted, but the books are temporarily stored, for a few days, in a trash folder. You can undo the delete by right clicking the *Remove books* button and choosing to *Restore recently deleted* books.

# **1.2 Preferences**



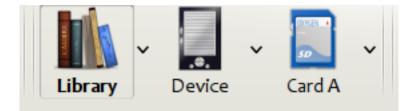
The *Preferences* action allows you to change the way various aspects of calibre work. It has four ad by doing a right dick on the button

variations, accessed by doing a right-click on the button.

- 1. **Preferences**: Allows you to change the way various aspects of calibre work. Clicking the button also performs this action.
- 2. Run Welcome wizard: Allows you to start the Welcome wizard which appeared the first time you started calibre.
- 3. Get plugins to enhance calibre: Opens a new window that shows plugins for calibre. These plugins are developed by third parties to extend calibre's functionality.

4. **Restart in debug mode**: Allows you to enable a debugging mode that can assist the calibre developers in solving problems you encounter with the program. For most users this should remain disabled unless instructed by a developer to enable it.

# 1.3 Catalogs



A catalog is a collection of books. calibre can manage two types of different catalogs:

- 1. Library: This is a collection of books stored in your calibre library on your computer.
- 2. **Device**: This is a collection of books stored in your e-book reader. It will be available when you connect the reader to your computer.

Many operations, such as adding books, deleting, viewing, etc., are context sensitive. So, for example, if you click the *View* button when you have the **Device** catalog selected, calibre will open the files on the device to view. If you have the **Library** catalog selected, files in your calibre library will be opened instead.

	Title	Author(s)	Size (MB)	Date	Rating 🗸	Publisher	Tags	Series
	The Complete Works of William Shakespeare	William Shakespeare	2.4	02 Jan 2007	****	manybooks.net		
	Stalky and Co.	Rudyard Kipling	0.2	19 Jan 2007	****	manybooks.net		
3	The Comedies of William Shakespeare	William Shakespeare	2.1	15 Mar 2007	****			
4	The Histories of William Shakespeare	William Shakespeare	1.5	15 Mar 2007	****		england, historical fiction	
5	The Tragedies of William Shakespeare	William Shakespeare	1.6	15 Mar 2007	****			
6	War and Peace	Leo Tolstoy	3.1	22 Aug 2007	****	gutenberg.org	classic	
7	Anna Karenina	Leo Tolstoy	1.9	22 Aug 2007	****	gutenberg.org	classic	
в	Guns, germs, and steel: the fates of human societies	Jared Diamond	0.4	29 Nov 2007	****	New York : W.W. Norton, c1997.		
9	A Game of Thrones	George R. R. Martin	1.3	23 Jan 2007	****		fantasy	
10	A Clash of Kings	George R. R. Martin	1.4	25 Jan 2007	****		fantasy	
11	A Storm of Swords	George R. R. Martin	1.9	27 Jan 2007	****		fantasy	
12	A Feast for Crows	George R. R. Martin	1.7	29 Jan 2007	****		fantasy	Song of Ice and Fire [4]
12	Banawraakar	locqueline Corev	0.0	00 May 2007	يد يد يد يد		fontony	The Sundaring [1]

# 1.4 Search & sort

The Search & Sort section allows you to perform several powerful actions on your book collections.

- You can sort them by title, author, date, rating, etc. by clicking on the column titles. You can also sub-sort, i.e. sort on multiple columns. For example, if you click on the title column and then the author column, the book will be sorted by author and then all the entries for the same author will be sorted by title.
- You can search for a particular book or set of books using the Search bar. More on that below.
- You can quickly and conveniently edit metadata by selecting the entry you want changed in the list and pressing the E key.

- You can perform Actions (page 4) on sets of books. To select multiple books you can either:
  - Keep the Ctrl key pressed and click on the books you want selected.
  - Keep the Shift key pressed and click on the starting and ending book of a range of books you want selected.
- You can configure which fields you want displayed by using the Preferences (page 10) dialog.
- To perform complex multiple column based sub-sorting add the *Sort by* tool to a toolbar via *Preferences* → *Toolbars* & *menus*.

### 1.5 The search interface

You can search all book metadata by entering search terms in the Search bar. For example:

Asimov Foundation format:lrf

This will match all books in your library that have Asimov and Foundation in their metadata and are available in the LRF format. Some more examples:

```
author:Asimov and not series:Foundation
title:"The Ring" or "This book is about a ring"
format:epub publisher:feedbooks.com
```

#### Search kinds

There are four search kinds: *contains, equality, regular expression* (see regular expressions<sup>2</sup>), and *character variant*. You choose the search kind with a prefix character.

#### 'Contains' searches

Searches with no prefix character are *contains* and are by default case insensitive. An item matches if the search string appears anywhere in the indicated metadata. You can make *contains* searches case sensitive by checking the option *Case sensitive searching* in *Preferences / Searching*. If the search option *Unaccented characters match accented characters and punctuation is ignored* is checked then a character will match all its variants (e.g., *e* matches é, è, ê, and  $\ddot{e}$ ) and all punctuation and whitespace are ignored. For example, if the *Unaccented characters match* ... option is checked then given the two book titles:

- 1. Big, Bothéred, and Bad
- 2. Big Bummer

then these searches find:

- title:"er" matches both ('e' matches both 'é' and 'e').
- title: "g " matches both because spaces are ignored.
- title: "g, " matches both because the comma is ignored.
- title: "gb" matches both because ', ' is ignored in book 1 and spaces are ignored in book 2.
- title: "g b" matches both because comma and space are ignored.
- title: "db" matches #1 because the space in 'and Bad' is ignored.
- title:", " matches both (it actually matches all books) because commas are ignored.

If the Unaccented characters match ... option is not checked then character variants, punctuation, and spacing are all significant.

<sup>&</sup>lt;sup>2</sup> https://en.wikipedia.org/wiki/Regular\_expression

You can set only one of the search options *Case sensitive searching* and *Unaccented characters match accented characters* and punctuation is ignored.

#### 'Equality' searches

Equality searches are indicated by prefixing the search string with an equals sign (=). For example, the query tag: "=science" will match *science*, but not *science fiction* or *hard science*. Character variants are significant:  $\acute{e}$  doesn't match e.

Two variants of equality searches are used for hierarchical items (e.g., A.B.C): hierarchical prefix searches and hierarchical component searches. The first, indicated by a single period after the equals (=.) matches the initial parts of a hierarchical item. The second, indicated by two periods after the equals (=..) matches an internal name in the hierarchical item. Examples, using the tag History.Military.WWII as the value:

- tags: "=. History" : True. History is a prefix of the tag.
- tags: "=. History. Military" : True. History. Military is a prefix of the tag.
- tags: "=.History.Military.WWII" : True. History.Military.WWII is a prefix of the tag, albeit an improper one.
- tags: "=.Military" : False. Military is not a prefix of the tag.
- tags: "=. WWII" : False. WWII is not a prefix of the tag.
- tags: "=...History" : True. The hierarchy contains the value History.
- tags: "=...Military" : True. The hierarchy contains the value Military.
- tags: "=...WWII" : True. The hierarchy contains the value WWII.
- tags: "=...Military.WWII": False. The .. search looks for single values.

#### 'Regular expression' searches

Regular expression searches are indicated by prefixing the search string with a tilde (~). Any Python-compatible regular expression<sup>3</sup> can be used. Backslashes used to escape special characters in regular expressions must be doubled because single backslashes will be removed during query parsing. For example, to match a literal parenthesis you must enter or alternatively use *super-quotes* (see below). Regular expression searches are 'contains' searches unless the expression is anchored. Character variants are significant: ~e doesn't match é.

#### 'Character variant' searches

Character variant searches are indicated by prefixing the search string with a caret (^). This search is similar to the *contains* search (above) except that:

- letter case is always ignored.
- character variants always match each other.
- punctuation and whitespace are always significant.

The search options *Unaccented characters match accented characters and punctuation is ignored* and *Case sensitive searching* are ignored. They have no effect on this search's behavior.

The following compares this search to a contains search assuming the *Unaccented characters match...* option is checked (see above) given the same two book titles:

- 1. Big, Bothéred, and Bad
- 2. Big Bummer

then these character variant searches find:

• title: "^er" matches both ('e' matches both 'é' and 'e')

<sup>&</sup>lt;sup>3</sup> https://docs.python.org/library/re.html

- title:"^g" matches both
- title: "^g " matches #2 because the space is significant
- title: "^g, " matches #1 because the comma is significant
- title: "^gb" matches nothing because space and comma are significant
- title: "^g b" matches #2 because the comma is significant
- title: "^db" matches nothing
- title: "^, " matches #1 (instead of all books) because the comma is significant

#### Search Expression Syntax

A search expression is a sequence of search terms optionally separated by the operators and or. If two search terms occur without a separating operator, and is assumed. The and operator has priority over the or operator; for example the expression a or b and c is the same as a or (b and c). You can use parenthesis to change the priority; for example (a or b) and c to make the or evaluate before the and. You can use the operator not to negate (invert) the result of evaluating a search expression. Examples:

- not tag: foo finds all books that don't contain the tag foo
- not (author:Asimov or author:Weber) finds all books not written by either Asimov or Weber.

The above examples show examples of *search terms*. A basic *search term* is a sequence of characters not including spaces, quotes ("), backslashes (\), or parentheses (()). It can be optionally preceded by a column name specifier: the *lookup name* of a column followed by a colon (:), for example author:Asimov. If a search term must contain a space then the entire term must be enclosed in quotes, as in title:"The Ring". If the search term must contain quotes then they must be *escaped* with backslashes. For example, to search for a series named *The "Ball" and The "Chain"*, use:

series:"The \"Ball\" and The \"Chain\"

If you need an actual backslash, something that happens frequently in *regular expression* searches, use two of them  $(\backslash \rangle)$ .

It is sometimes hard to get all the escapes right so the result is what you want, especially in *regular expression* and *template* searches. In these cases use the *super-quote*: """sequence of characters""". Super-quoted characters are used unchanged: no escape processing is done.

#### More information

To search for a string that begins with an equals, tilde, or caret; prefix the string with a backslash.

Enclose search strings with quotes (") if the string contains parenthesis or spaces. For example, to find books with the tag Science Fiction you must search for tag: "=science fiction". If you search for tag:=science fiction you will find all books with the tag science and the word fiction in any metadata.

You can build advanced search queries easily using the Advanced search dialog accessed by clicking the button



on the left of the search box.

Available fields for searching are: tag, title, author, publisher, series, series\_index, rating, cover, comments, format, identifiers, date, pubdate, search, size, vl and custom columns. If a device is plugged in, the ondevice field becomes available, when searching the calibre library view. To find the search name (actually called the *lookup name*) for a custom column, hover your mouse over the column header in the library view.

#### Dates

The syntax for searching for dates is:

```
pubdate:>2000-1 Will find all books published after Jan, 2000
date:<=2000-1-3 Will find all books added to calibre before 3 Jan, 2000
pubdate:=2009 Will find all books published in 2009
```

If the date is ambiguous then the current locale is used for date comparison. For example, in an mm/dd/yyyy locale 2/1/2009 is interpreted as 1 Feb 2009. In a dd/mm/yyyy locale it is interpreted as 2 Jan 2009. Some special date strings are available. The string today translates to today's date, whatever it is. The strings yesterday and thismonth (or the translated equivalent in the current language) also work. In addition, the string daysago (also translated) can be used to compare to a date some number of days ago. For example:

```
date:>10daysago
date:<=45daysago</pre>
```

To avoid potential problems with translated strings when using a non-English version of calibre, the strings \_today, \_yesterday, \_thismonth, and \_daysago are always available. They are not translated.

#### Searching dates and numeric values with relational comparisons

Dates and numeric fields support the relational operators = (equals), > (greater than), >= (greater than or equal to), < (less than), <= (less than or equal to), and != (not equal to). Rating fields are considered to be numeric. For example, the search rating:>=3 will find all books rated 3 or higher.

You can search for books that have a format of a certain size like this:

- size:>1.1M will find books with a format larger than 1.1MB
- size: <=1K will find books with a format smaller than or equal to 1KB

You can search for the number of items in multiple-valued fields such as tags using the character # then using the same syntax as numeric fields. For example, to find all books with more than 4 tags use tags: #>4. To find all books with exactly 10 tags use tags: #=10.

#### Series indices

Series indices are searchable. For the standard series, the search name is series\_index. For custom series columns, use the column search name followed by \_index. For example, to search the indices for a custom series column named #my\_series, you would use the search name #my\_series\_index. Series indices are numbers, so you can use the relational operators described above.

#### Saved searches

The special field search is used for *saved searches* (page 17). If you save a search with the name "My spouse's books" you can enter search: "My spouse's books" in the Search bar to reuse the saved search. More about saving searches below.

#### Virtual libraries

The special field v1 is used to search for books in a Virtual library. For example, v1:Read will find all the books in the *Read* Virtual library. The search v1:Read and v1:"Science Fiction" will find all the books that are in both the *Read* and *Science Fiction* virtual libraries. The value following v1: must be the name of a Virtual library. If the Virtual library name contains spaces then surround it with quotes.

#### Whether a field has a value

You can search for the absence or presence of a value for a field using "true" and "false". For example:

- cover:false finds all books without a cover
- series:true finds all books that are in a series
- series:false finds all books that are not in a series
- comments:false finds all books with an empty comment

• formats:false finds all books with no book files (empty records)

#### Yes/no custom columns

Searching Yes/no custom columns for false, empty, or blank will find all books with undefined values in the column. Searching for true will find all books that do not have undefined values in the column. Searching for yes or checked will find all books with Yes in the column. Searching for no or unchecked will find all books with No in the column. Note that the words yes, no, blank, empty, checked and unchecked are translated; you can use either the current language's equivalent word or the English word. The words true and false and the special values \_yes, \_no, and \_empty are not translated.

#### Identifiers

Identifiers (e.g., ISBN, DOI, LCCN, etc.) use an extended syntax. An identifier has the form type:value, as in isbn:123456789. The extended syntax permits you to specify independently the type and value to search for. Both the type and the value parts of the query can use any of the *search kinds* (page 12). Examples:

- identifiers:true will find books with any identifier.
- identifiers:false will find books with no identifier.
- identifiers:123 will search for books with any type having a value containing 123.
- identifiers:=123456789 will search for books with any type having a value equal to 123456789.
- identifiers:=isbn: and identifiers:isbn:true will find books with a type equal to ISBN having any value
- identifiers:=isbn:false will find books with no type equal to ISBN.
- identifiers:=isbn:123 will find books with a type equal to ISBN having a value containing 123.
- identifiers:=isbn:=123456789 will find books with a type equal to ISBN having a value equal to 123456789.
- identifiers:::1 will find books with a type containing an *i* having a value containing a *1*.

#### Categories visible in the Tag browser

The search  $in\_tag\_browser:true$  finds all books that are in categories (tags, authors, etc.) currently shown in the *Tag* browser. This is useful if you set the two preferences  $Preferences \rightarrow Look & feel \rightarrow Tag$  browser  $\rightarrow$  Hide empty categories and *Find* shows all items that match. With those two preferences set, doing a *Find* in the *Tag* browser shows only categories containing items matched by the *Find*. Then, the search  $in\_tag\_browser:true$  additionally finds books with these categories / items. You can easily run this search by pressing the key Ctrl+Alt+Shift+F or clicking the configure button in the *Tag* browser and choosing the *Show only books that have visible categories* entry.

#### Search using templates

You can search using a template in *The calibre template language* (page 158) instead of a metadata field. To do so you enter a template, a search type, and the value to search for. The syntax is:

template: (the template) #@#: (search type) : (the value)

The template is any valid calibre template language template. The search type must be one of t (text search), d (date search), n (numeric search), or b (set/not set (boolean)). The value is whatever you want, and can use the *search kinds* (page 12) described above for the various search types. You must quote the entire search string if there are spaces anywhere in it.

Examples:

• template:"program: connected\_device\_name('main')#@#:t:kindle" - is true when the kindle device is connected.

- template:"program: select(formats\_sizes(), 'EPUB')#@#:n:>1000000" finds books with EPUB files larger than 1 MB.
- template:"program: select(formats\_modtimes('iso'), 'EPUB')#@#:d:>10daysago" finds books with EPUB files newer than 10 days ago.
- template:"""program: book\_count('tags:^"' & \$series & '"', 0) != 0#@#:n:1""" finds all books containing the series name in the tags. This example uses super-quoting because the template uses both single quotes (') and double quotes (") when constructing the search expression.

You can build template search queries easily using the Advanced search dialog accessed by clicking the button



You can test templates on specific books using the calibre *Template tester*, which can be added to the toolbars or menus via *Preferences*  $\rightarrow$  *Toolbars* & *menus*. It can also be assigned a keyboard shortcut via *Preferences*  $\rightarrow$  *Shortcuts*.

Advanced search dialog

Q Advanced search	? ×
What kind of match to use: Contains: the word or phrase matches anywhere in the metadata field   Advanced search Title/author/series   Date search Template search   Find entries that have   All these words:   This exact phrase:   One or more of these words:   But don't show entries that have Any of these unwanted words:	You can also perform other kinds of advanced searches, for example checking for books that have no covers, combining multiple search expression using Boolean operators and so on. See <u>The search interface</u> for more information.
Clear	✓ OK X Cancel

Fig. 1: Advanced search dialog

### 1.6 Saving searches

calibre allows you to save a frequently used search under a special name and then reuse that search with a single click. To do this, create your search either by typing it in the Search bar or using the Tag browser. Then type the name you would like to give to the search in the Saved Searches box next to the Search bar. Click the plus icon next to the saved searches box to save the search.

Now you can access your saved search in the Tag browser under *Saved searches*. A single click will allow you to reuse any arbitrarily complex search easily, without needing to re-create it.

# 1.7 Searching the full text of all books



calibre can *optionally* index the full

text of books in the library to allow for instant searching of words inside any book. To use this functionality click the FT button at the left edge of the search bar.

Then, enable indexing for the current library. Once indexing is complete you can search all the text in the full library. When you add new books to the library, they will be automatically indexed in the background. This search allows you to quickly find all books that contain a word or combination of words. You can even search for words that occur near other words, as shown in the examples in the search popup window. Note that this search tool will find only one occurrence of the search query in a particular book, not list every occurrence, for that it is best to search inside the book using the calibre *E-book viewer*.

You can re-index an individual book by right clicking on the *Book details panel* in calibre and choosing *Re-index this book* for full text searching.

### **1.8 Virtual libraries**

A *Virtual library* is a way to pretend that your calibre library has only a few books instead of its full collection. This is an excellent way to partition your large collection of books into smaller, manageable chunks. To learn how to create and use Virtual libraries, see the tutorial: *Virtual libraries* (page 245).

# 1.9 Temporarily marking books

You can temporarily mark arbitrary sets of books. Marked books will have a pin on them and can be found with the search marked:true. To mark a book press Ctrl+M or go to *Preferences* → *Toolbars* & *menus* and add the *Mark books* button to the main toolbar.

You can mark books with a specific text label by right clicking the *Mark books* button and choosing *Mark books with text label*. Books marked with text labels can later be found using the search marked: "=the-text-you-entered".

### 1.10 Guessing metadata from file names

Normally, calibre reads metadata from inside the book file. However, it can be configured to read metadata from the file name instead, via *Preferences*  $\rightarrow$  *Import/export*  $\rightarrow$  *Adding books*  $\rightarrow$  *Read metadata from file contents*.

You can also control how metadata is read from the filename using regular expressions (see *All about using regular expressions in calibre* (page 212)). In the *Adding books* section of the configuration dialog, you can specify a regular expression that calibre will use to try and guess metadata from the names of e-book files that you add to the library. The default regular expression is:

```
title - author
```

that is, it assumes that all characters up to the first – are the title of the book and subsequent characters are the author of the book. For example, the filename:

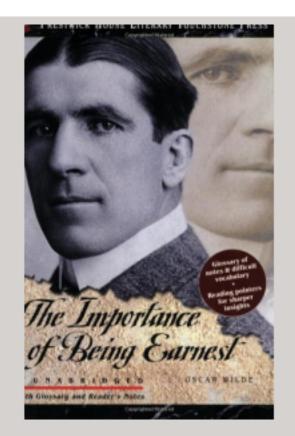
```
Foundation and Earth - Isaac Asimov.txt
```

will be interpreted to have the title: Foundation and Earth and author: Isaac Asimov

### 🖓 Tip

If the filename does not contain the hyphen, the above regular expression will fail.

# 1.11 Book details



Authors:	Oscar Wilde
Formats:	EPUB
lds:	9781580495806
Tags:	lit 101 homework
Path:	Click to open

### SUMMARY:

This Prestwick House Literary Touchstone Edition includes a glossary and reader's notes to help the modern reader appreciate Wilde's wry wit and elaborate plot twists.Oscar Wilde's madcap farce about mistaken identities, secret engagements, and lovers? entanglements still delights readers

The Book details display shows the cover and all the metadata for the currently selected book. It can be hidden via the

*Layout* button in the lower right corner of the main calibre window. The author names shown in the Book details panel are click-able, they will by default take you to the Wikipedia page for the author. This can be customized by right clicking on the author name and selecting Manage this author.

Similarly, if you download metadata for the book, the Book details panel will automatically show you links pointing to the web pages for the book on Amazon, WorldCat, etc. from where the metadata was downloaded.

You can right click on individual e-book formats in the Book details panel to delete them, compare them to their original versions, save them to disk, open them with an external program, etc.

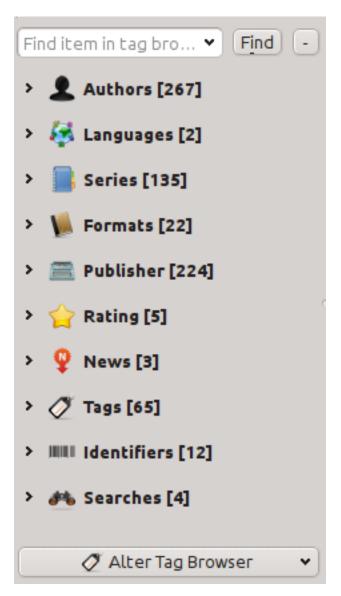
You can change the cover of the book by simply drag and dropping an image onto the Book details panel. If you wish to edit the cover image in an external program, simply right click on it and choose *Open with*.

You can also add e-book files to the current book by drag and dropping the files onto the Book details panel.

Double clicking the Book details panel will open it up in a separate popup window.

Finally, you can customize exactly what information is displayed in the Book details panel via *Preferences*  $\rightarrow$  *Interface*  $\rightarrow$  *Look* & *feel*  $\rightarrow$  *Book details*.

# 1.12 Tag browser



The Tag browser allows you to easily browse your collection by Author/Tags/Series/etc. If you click on any item in the Tag browser, for example the author name Isaac Asimov, then the list of books to the right is restricted to showing books by that author. You can click on category names as well. For example, clicking on "Series" will show you all books in any series.

The first click on an item will restrict the list of books to those that contain or match the item. Continuing the above example, clicking on Isaac Asimov will show books by that author. Clicking again on the item will change what is shown, depending on whether the item has children (see sub-categories and hierarchical items below). Continuing the Isaac Asimov example, clicking again on Isaac Asimov will restrict the list of books to those not by Isaac Asimov. A third click will remove the restriction, showing all books. If you hold down the Ctrl or Shift keys and click on multiple items, then restrictions based on multiple items are created. For example you could hold Ctrl and click on the tags History and Europe for finding books on European history. The Tag browser works by constructing search expressions that are automatically entered into the Search bar. Looking at what the Tag browser generates is a good way to learn how to construct basic search expressions.

Items in the Tag browser have their icons partially colored. The amount of color depends on the average rating of the

books in that category. So for example if the books by Isaac Asimov have an average of four stars, the icon for Isaac Asimov in the Tag browser will be 4/5th colored. You can hover your mouse over the icon to see the average rating.

The outer-level items in the *Tag browser*, such as Authors and Series, are called categories. You can create your own categories, called *User categories*, which are useful for organizing items. For example, you can use the *User categories editor* (click the *Configure* button at the lower-left of the *Tag browser* and choose *Manage authors, tags, etc*  $\rightarrow$  *User categories*) to create a User category called Favorite Authors, then put the items for your favorites into the category. User categories can have sub-categories. For example, the User category Favorites.Authors is a sub-category of Favorites. You might also have Favorites.Series, in which case there will be two sub-categories under Favorites. Sub-category name; or by using the *User categories editor* by entering names like the Favorites example above.

# You can search User categories in the same way as built-in categories, by clicking on them. There are four different searches cycled through by clicking:

- 1. "everything matching an item in the category" indicated by a single green plus sign.
- 2. "everything matching an item in the category or its sub-categories" indicated by two green plus signs.
- 3. "everything not matching an item in the category" shown by a single red minus sign.
- 4. "everything not matching an item in the category or its sub-categories" shown by two red minus signs.

It is also possible to create hierarchies inside some of the text categories such as tags, series, and custom columns. These hierarchies show with the small triangle, permitting the sub-items to be hidden. To use hierarchies of items in a category, you must first go to *Preferences*  $\rightarrow$  *Interface*  $\rightarrow$  *Look* & *feel* and enter the category name(s) into the "Categories with hierarchical items" field. Once this is done, items in that category that contain periods will be shown using the small triangle. For example, assume you create a custom column called "Genre" and indicate that it contains hierarchical items. Once done, items such as Mystery.Thriller and Mystery.English will display as Mystery with the small triangle next to it. Clicking on the triangle will show Thriller and English as sub-items. See *Managing subgroups of books, for example "genre"* (page 151) for more information.

Hierarchical items (items with children) use the same four 'click-on' searches as User categories. Items that do not have children use two of the searches: "everything matching" and "everything not matching".

You can drag and drop items in the Tag browser onto User categories to add them to that category. If the source is a User category, holding the Shift key while dragging will move the item to the new category. You can also drag and drop books from the book list onto items in the Tag browser; dropping a book on an item causes that item to be automatically applied to the dropped books. For example, dragging a book onto Isaac Asimov will set the author of that book to Isaac Asimov. Dropping it onto the tag History will add the tag History to the book's tags.

You can easily find any item in the Tag browser by clicking the search button at the lower-right corner. In addition, you can right click on any item and choose one of several operations. Some examples are to hide it, rename it, or open a "Manage x" dialog that allows you to manage items of that kind. For example, the *Manage authors* dialog allows you to rename authors and control how their names are sorted.

You can control how items are sorted in the Tag browser via the *Configure* button at the lower-left of the Tag browser. You can choose to sort by name, average rating or popularity (popularity is the number of books with an item in your library; for example, the popularity of Isaac Asimov is the number of books in your library by Isaac Asimov).

You can use your own icons for categories and values in categories. To change the icon for a category, right-click on the category (the outer-level item) and choose *Change (category name) category icon*. A dialog will open where you can pick an image to be used as the icon. To restore the icon to its default choose *Restore (category name) default icon*.

To choose icons for values in categories, right-click on a value then choose *Manage icon for (value name)*. You will see a list of choices:

- *Choose an icon for this value but not its children.* A dialog will open where you choose an icon for the value. Children of that value will not inherit that icon.
- *Choose an icon for this value and its children.* A dialog will open where you choose an icon for the value. Any children that don't have their own specified icon will inherit this icon.

- Use the existing icon for this value but not its children. This option is offered if the value already has an icon that is inherited by the value's children. Selecting it will make the icon apply to the value but not its children.
- Use the existing icon for this value and its children. This option is offered if the value already has an icon that is not inherited by the value's children. Selecting it will make the icon apply to the value and its children.
- Use the default icon for this value. This option is offered if the item has an icon. It removes the icon from the value and any children inheriting the icon. The default icon is what is specified below.
- *Reset all value icons to the default icon*. This option removes all item value icons for the category. It does not remove a template if one exists. There is no undo.
- *Use/edit a template to choose the default value icon*. This option permits you to provide a calibre template that returns the name of an icon file to be used as a default icon. The template can use two variables:
  - category: the lookup name of the category, for example authors, series, #mycolumn.
  - value: the value of the item within the category.
  - count: the number of books with this value. If the value is part of a hierarchy then the count includes the children.
  - avg\_rating: the average rating for books with this value. If the value is part of a hierarchy then the average includes the children.

Book metadata such as title is not available. Template database functions such as *book\_count* (page 184) and *book\_values* (page 185) will work, but the performance might not be acceptable. The following template functions will work in the GUI but won't work in the content server: *connected\_device\_name* (page 192), *connected\_device\_uuid* (page 192), *current\_virtual\_library\_name* (page 193), *is\_marked* (page 194), and *virtual\_libraries* (page 195).

In the GUI, Python templates have full access to the calibre database. In the content server, Python templates have access to new API (see API documentation for the database interface<sup>4</sup>) but not the old API (LibraryDatabase).

For example, this template specifies that any value in the clicked-on category beginning with *History* will have an icon named flower.png:

```
program:
    if substr($value, 0, 7) == 'History' then 'flower.png' fi
```

If a template returns the empty string ('') then the category icon will be used. If the template returns a file name that doesn't exist then no icon is displayed.

• Use the category icon as the default. This option specifies that the icon used for the category should be used for any value that doesn't otherwise have an icon. Selecting this option removes any template icon specification.

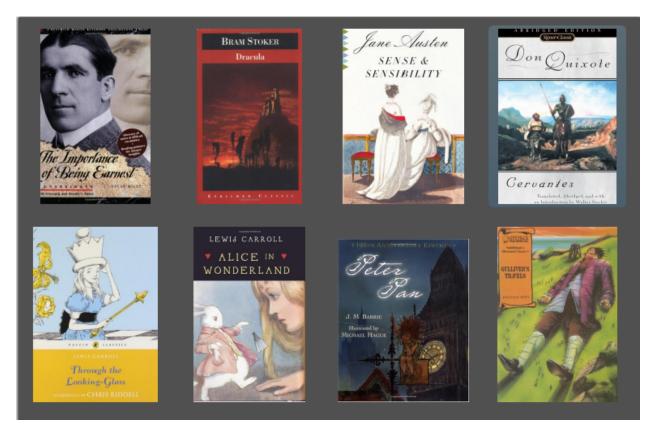
The icon is chosen using the following hierarchy:

- 1. The icon specified for the value, if it exists.
- 2. The icon specified for a parent node found by walking up the tree, if one exists.
- 3. The icon from a template, if a template exists and it returns a non-empty string.
- 4. The default category icon, which always exists.

Icons are per-user, not per-library, stored in the calibre configuration folder. Icons for item values are stored in the tb\_icons subfolder. Icons used by templates are in the template\_icons subfolder of tb\_icons.

<sup>&</sup>lt;sup>4</sup> https://manual.calibre-ebook.com/db\_api.html

# 1.13 Cover grid



You can have calibre display a grid of book covers instead of a list of books, if you prefer to browse your collection by covers instead. The *Cover grid* is activated by clicking the *Layout* button in the bottom right corner of the main calibre window. You can customize the cover sizes and the background of the *Cover grid* via *Preferences*  $\rightarrow$  *Interface*  $\rightarrow$  *Look* & *feel*  $\rightarrow$  *Cover grid*. You can even have calibre display any specified field under the covers, such as title or authors or rating or a custom column of your own devising.

# 1.14 Cover browser



In addition to the *Cover grid* described above, you can also have calibre display covers in the single row. This is activated via the *Layout* button in the lower right corner of the main window. In *Preferences*  $\rightarrow$  *Interface*  $\rightarrow$  *Look* & *feel*  $\rightarrow$  *Cover browser* you can change the number of covers displayed, and even have the *Cover browser* display itself in a separate popup window.

# 1.15 Adding notes for authors, series, etc.

# William Shakespeare Ζ



William Shakespeare (baptised 26 April 1564) was an English poet and playwright, widely regarded as the greatest writer in the English language and the world's pre-eminent dramatist. He is often called England's national poet and the "Bard of Avon" (or simply "The Bard"). His surviving works consist of 38 plays, 154 sonnets, two long narrative poems, and several other poems. His plays have been translated into every major living language, and are performed more often than those of any other playwright.

Shakespeare was born and raised in Stratford-upon-Avon. Scholars believe that he died on his fifty-second birthday, coinciding with St George's Day.

At the age of 18 he married Anne Hathaway, who bore him three children: Susanna, and twins Hamnet and Judith. Between 1585 and 1592 he began a successful career in London as an actor, writer, and part owner of the playing company the Lord Chamberlain's Men, later known as the King's Men.



You can add notes for an author/series/tag/publisher/etc. to your calibre library. To do so right click on the author name in the *Tag browser* on the left or the *Book details* panel on the right and choose *Create note* or *Edit note*.

A simple popup window will allow you to enter your notes using basic formatting and supporting links and images. Once a note for an author is created, it can be viewed easily from the *Book details* panel by clicking the little pencil icon next to the author name.

You can search through all the notes in your library using the *Browse notes* tool by pressing Ctrl+Shift+N or adding it to the toolbar via *Preferences*  $\rightarrow$  *Toolbars* & *menus*.

# 1.16 Quickview

Sometimes you want to select a book and quickly get a list of books with the same value in some category (authors, tags, publisher, series, etc.) as the currently selected book, but without changing the current view of the library. You can do this with Quickview. Quickview opens either a second window or a panel in the book list showing the list of books matching the value of interest. For example, assume you want to see a list of all the books with the one or more of the authors of the currently-selected book. Click in the author cell you are interested in and press the 'Q' key or click the *Quickview* icon in the *Layout* section of the calibre window. A window or panel will open with all the authors for that book on the left, and all the books by the selected author on the right.

#### Some example Quickview usages: quickly seeing what other books:

- have some tag(s) applied to the currently selected book,
- are in the same series as the current book
- have the same values in a custom column as the current book
- are written by one of the same authors of the current book
- share values in a custom column

There are two choices for where the Quickview information appears:

- 1. It can open "undocked": on top of the calibre window and will stay open until you explicitly close it.
- 2. It can open "docked": as a panel in the book list section of the calibre main window.

You can move the window from docked to undocked as desired using the "Dock/Undock" button.

The Quickview panel can be left open permanently, in which case it follows movements on the book list. For example, if you click in the calibre library view on a category column (tags, series, publisher, authors, etc.) for a book, the Quickview window contents will change to show you in the left-hand side panel the values in that category for the selected book (e.g., the tags for that book). The first item in that list will be selected, and Quickview will show you on the right-hand side panel all the books in your library that use that value. Click on an different value in the left-hand panel to see the books with that different value.

Double-click on a book in the Quickview window to select that book in the library view. This will also change the items display in the QuickView window (the left-hand panel) to show the items in the newly-selected book.

Shift- or Ctrl- double-click on a book in the Quickview window to open the edit metadata dialog on that book in the calibre window. The edited book will be Quickview'ed when you close the edit metadata dialog.

You can see if a column can be Quickview'ed by hovering your mouse over the column heading and looking at the tooltip for that heading. You can also know by right-clicking on the column heading to see of the "Quickview" option is shown in the menu, in which case choosing that Quickview option is equivalent to pressing 'Q' in the current cell.

Options (in *Preferences*  $\rightarrow$  *Look* & *feel*  $\rightarrow$  *Quickview*):

- Respect (or not) the current Virtual library. If checked then Quickview shows only books in the current Virtual library. Default: respect Virtual libraries
- Change the Quickview window contents when the column is changed on the book list using the cursor keys. Default: don't follow changes made with cursor keys
- Change the column being "quickview'ed" when a cell in the Quickview window is double-clicked. Otherwise the book is changed but the column being examined is not. Default: change the column
- Change the column being "quickview'ed" to the current column when the return key is pressed in the Quickview panel. Otherwise the book is changed but the column being examined is not. Default: change the column
- Choose which columns are shown in the Quickview window/panel.

# 1.17 Jobs



The Jobs panel shows the number of currently running jobs. Jobs are tasks that run in a separate process. They include converting e-books and talking to your reader device. You can click on the jobs panel to access the list of jobs. Once a job has completed you can see a detailed log from that job by double-clicking it in the list. This is useful to debug jobs that may not have completed successfully.

# **1.18 Keyboard shortcuts**

calibre has several keyboard shortcuts to save you time and mouse movement. These shortcuts are active in the book list view (when you're not editing the details of a particular book), and most of them affect the title you have selected. The calibre E-book viewer *has its own shortcuts* (page 57) which can be customised in the viewer *Preferences*.

#### Note

Note: The calibre keyboard shortcuts do not require a modifier key (Command, Option, Control, etc.), unless specifically noted. You only need to press the letter key, e.g. E to edit.

Key- board short- cut	Action
F2 (En- ter for ma-	Edit the metadata of the currently selected field in the book list.
COS)	
A	Add books
	Add formats to the selected books
С	Convert selected books
D	Send to device
Del	Remove selected books
E	Edit metadata of selected books
G	Get books
I	Show Book details
K	Edit Table of Contents
М	Merge selected records
Alt+M	Merge selected records, keeping originals
0	Open containing folder
P	Polish books
S	Save to disk
Т	Edit book
V	View
Shift+V	View last read book

Table 1: Keyboard shortcuts for the main calibre program

continues on next page

	Table 1 – continued from previous page
Key- board short- cut	Action
Alt+V/	View specific format
Cmd+V	
for	
macOS	
	Toggle jobs list
	Toggle Cover browser
	Toggle Book details panel
	Toggle Tag browser
	Toggle Cover grid
Alt+A	Show books by the same author as the current book
Alt+T	Show books with the same tags as current book
Alt+P	Show books by the same publisher as current book
	Show books in the same series as current book
/,	Focus the Search bar
Ctrl+F	
	Open the Advanced search dialog
	Toggle the Search bar
Esc	Clear the current search
	Focus the book list
	Clear the Virtual library
	Clear the additional restriction
	Create a temporary Virtual library based on the current search
	Select the next Virtual library tab
	Select the previous Virtual library tab
NorF3	Find the next book that matches the current search (only works if search highlighting is turned on in search
e1   C   11	preferences)
	Find the previous book that matches the current search (only works if search highlighting is turned on in
or	search preferences)
Shift+F	Download metadata and covers
Ctrl+D	
	Restart calibre
	Restart calibre in debug mode
	Add empty books to calibre
Ctrl+M	Toggle mark/unmarked status on selected books
Ctrl+/	Open the popup to search the full text of all books in the library
or	
Ctrl+Al	Onen the Origh Winner reason for viewing hereby in related conice/terrelate
Q	Open the Quick View popup for viewing books in related series/tags/etc.
	Focus the opened Quick View panel
	Perform a search in the Quick View panel
F5	Re-apply the current sort
Ctrl+Q	Quit calibre
X	Toggle auto scroll of the book list
	Restrict the displayed books to only those books that are in a category currently displayed in the <i>Tag browser</i>
В	Browse annotations (highlights and bookmarks) made in the calibre E-book viewer for all books in the
01 .1.1.01	library Provide notes appropriated with outbors/pariss/tags/sta
Ctrl+Sh	Browse notes associated with authors/series/tags/etc.
	continues on next page

Table 1 - continued from previo	us page
---------------------------------	---------

continues on next page

Key- board short- cut	Action
Alt+Shi	Toggle the layout between wide and narrow views

### Table 1 - continued from previous page

## ADDING YOUR FAVORITE NEWS WEBSITE

calibre has a powerful, flexible and easy-to-use framework for downloading news from the Internet and converting it into an e-book. The following will show you, by means of examples, how to get news from various websites.

To gain an understanding of how to use the framework, follow the examples in the order listed below:

- *Completely automatic fetching* (page 33)
  - The calibre blog (page 33)
  - bbc.co.uk (page 35)
- Customizing the fetch process (page 35)
  - Using the print version of bbc.co.uk (page 35)
  - Replacing article styles (page 36)
  - Slicing and dicing (page 37)
  - Real life example (page 37)
- Tips for developing new recipes (page 40)
- Further reading (page 41)
- API documentation (page 41)

## 2.1 Completely automatic fetching

If your news source is simple enough, calibre may well be able to fetch it completely automatically, all you need to do is provide the URL. calibre gathers all the information needed to download a news source into a *recipe*. In order to tell calibre about a news source, you have to create a *recipe* for it. Let's see some examples:

### 2.1.1 The calibre blog

The calibre blog is a blog of posts that describe many useful calibre features in a simple and accessible way for new calibre users. In order to download this blog into an e-book, we rely on the *RSS* feed of the blog:

http://blog.calibre-ebook.com/feeds/posts/default

I got the RSS URL by looking under "Subscribe to" at the bottom of the blog page and choosing  $Posts \rightarrow Atom$ . To make calibre download the feeds and convert them into an e-book, you should right click the *Fetch news* button and then the *Add a custom news source* menu item and then the *New Recipe* button. A dialog similar to that shown below should open up.

Recipe <u>t</u> itle:	My N	ews Source	
<u>O</u> ldest article:	7 day	(s)	
Max. number of article	s per feed: 100	-	
Feeds in recipe			
Add feed to recipe			
Feed title:			
Feed URL:			
_			

First enter Calibre Blog into the *Recipe title* field. This will be the title of the e-book that will be created from the articles in the above feeds.

The next two fields (*Oldest article* and *Max. number of articles*) allow you some control over how many articles should be downloaded from each feed, and they are pretty self explanatory.

To add the feeds to the recipe, enter the feed title and the feed URL and click the *Add feed* button. Once you have added the feed, simply click the *Save* button and you're done! Close the dialog.

To test your new *recipe*, click the *Fetch news* button and in the *Custom news sources* sub-menu click *calibre Blog*. After a couple of minutes, the newly downloaded e-book of blog posts will appear in the main library view (if you have your reader connected, it will be put onto the reader instead of into the library). Select it and hit the *View* button to read!

The reason this worked so well, with so little effort is that the blog provides *full-content RSS* feeds, i.e., the article content is embedded in the feed itself. For most news sources that provide news in this fashion, with *full-content* feeds, you don't need any more effort to convert them to e-books. Now we will look at a news source that does not provide full content feeds. In such feeds, the full article is a webpage and the feed only contains a link to the webpage with a short summary of the article.

### 2.1.2 bbc.co.uk

Let's try the following two feeds from *The BBC*:

- 1. News Front Page: https://newsrss.bbc.co.uk/rss/newsonline\_world\_edition/front\_page/rss.xml
- 2. Science/Nature: https://newsrss.bbc.co.uk/rss/newsonline\_world\_edition/science/nature/rss.xml

Follow the procedure outlined in *The calibre blog* (page 33) above to create a recipe for *The BBC* (using the feeds above). Looking at the downloaded e-book, we see that calibre has done a creditable job of extracting only the content you care about from each article's webpage. However, the extraction process is not perfect. Sometimes it leaves in undesirable content like menus and navigation aids or it removes content that should have been left alone, like article headings. In order, to have perfect content extraction, we will need to customize the fetch process, as described in the next section.

## 2.2 Customizing the fetch process

When you want to perfect the download process, or download content from a particularly complex website, you can avail yourself of all the power and flexibility of the *recipe* framework. In order to do that, in the *Add custom news sources* dialog, simply click the *Switch to Advanced mode* button.

The easiest and often most productive customization is to use the print version of the online articles. The print version typically has much less cruft and translates much more smoothly to an e-book. Let's try to use the print version of the articles from *The BBC*.

### 2.2.1 Using the print version of bbc.co.uk

The first step is to look at the e-book we downloaded previously from *bbc.co.uk* (page 35). At the end of each article, in the e-book is a little blurb telling you where the article was downloaded from. Copy and paste that URL into a browser. Now on the article webpage look for a link that points to the "Printable version". Click it to see the print version of the article. It looks much neater! Now compare the two URLs. For me they were:

#### Article URL

https://news.bbc.co.uk/2/hi/science/nature/7312016.stm

#### Print version URL

https://newsvote.bbc.co.uk/mpapps/pagetools/print/news.bbc.co.uk/2/hi/science/nature/7312016. stm

So it looks like to get the print version, we need to prefix every article URL with:

newsvote.bbc.co.uk/mpapps/pagetools/print/

Now in the *Advanced mode* of the Custom news sources dialog, you should see something like (remember to select *The BBC* recipe before switching to advanced mode):

– Recipe source code (python) -

```
class AdvancedUserRecipe1206418393(BasicNewsRecipe):
    title = u'The BBC'
    oldest_article = 7
    max_articles_per_feed = 100
    feeds = [(u'News Front Page', u'http://newsrss.bbc.co.uk/rss/newsonlir
```

You can see that the fields from the *Basic mode* have been translated to Python code in a straightforward manner. We need to add instructions to this recipe to use the print version of the articles. All that's needed is to add the following two lines:

```
def print_version(self, url):
    return url.replace('https://', 'https://newsvote.bbc.co.uk/mpapps/pagetools/print/
    -')
```

This is Python, so indentation is important. After you've added the lines, it should look like:

```
Recipe source code (python)

class AdvancedUserRecipe1206418393(BasicNewsRecipe):
    title = u'The BBC'
    oldest_article = 7
    max_articles_per_feed = 100
    feeds = [(u'News Front Page', u'http://newsrss.bbc.co.uk/rss/newsonlir
    def print_version(self, url):
        [return url.replace('http://', 'http://newsvote.bbc.co.uk/mpapps/pagetools/p
```

In the above, def print\_version (self, url) defines a *method* that is called by calibre for every article. url is the URL of the original article. What print\_version does is take that url and replace it with the new URL that points to the print version of the article. To learn about Python<sup>5</sup> see the tutorial<sup>6</sup>.

Now, click the *Add/update recipe* button and your changes will be saved. Re-download the e-book. You should have a much improved e-book. One of the problems with the new version is that the fonts on the print version webpage are too small. This is automatically fixed when converting to an e-book, but even after the fixing process, the font size of the menus and navigation bar become too large relative to the article text. To fix this, we will do some more customization, in the next section.

### 2.2.2 Replacing article styles

In the previous section, we saw that the font size for articles from the print version of *The BBC* was too small. In most websites, *The BBC* included, this font size is set by means of *CSS* stylesheets. We can disable the fetching of such stylesheets by adding the line:

no\_stylesheets = True

The recipe now looks like:

```
<sup>5</sup> https://www.python.org
```

```
<sup>6</sup> https://docs.python.org/tutorial/
```

```
Recipe source code (python)
```

The new version looks pretty good. If you're a perfectionist, you'll want to read the next section, which deals with actually modifying the downloaded content.

### 2.2.3 Slicing and dicing

calibre contains very powerful and flexible abilities when it comes to manipulating downloaded content. To show off a couple of these, let's look at our old friend the *The BBC* (page 36) recipe again. Looking at the source code (*HTML*) of a couple of articles (print version), we see that they have a footer that contains no useful information, contained in

```
<div class="footer">
...
</div>
```

This can be removed by adding:

```
remove_tags = [dict(name='div', attrs={'class':'footer'})]
```

to the recipe. Finally, lets replace some of the CSS that we disabled earlier, with our own CSS that is suitable for conversion to an e-book:

```
extra_css = '.headline {font-size: x-large;} \n .fact { padding-top: 10pt }'
```

With these additions, our recipe has become "production quality".

This *recipe* explores only the tip of the iceberg when it comes to the power of calibre. To explore more of the abilities of calibre we'll examine a more complex real life example in the next section.

### 2.2.4 Real life example

A reasonably complex real life example that exposes more of the *API* of BasicNewsRecipe is the *recipe* for *The New York Times* 

```
import string, re
from calibre import strftime
from calibre.web.feeds.recipes import BasicNewsRecipe
from calibre.ebooks.BeautifulSoup import BeautifulSoup
class NYTimes(BasicNewsRecipe):
    title = 'The New York Times'
```

(continues on next page)

(continued from previous page)

```
author = 'Kovid Goyal'
   description = 'Daily news from the New York Times'
   timefmt = ' [%a, %d %b, %Y]'
   needs_subscription = True
   remove_tags_before = dict(id='article')
   remove_tags_after = dict(id='article')
   remove_tags = [dict(attrs={'class':['articleTools', 'post-tools', 'side_tool',

→ 'nextArticleLink clearfix']}),
               dict(id=['footer', 'toolsRight', 'articleInline', 'navigation',
dict(name=['script', 'noscript', 'style'])]
   encoding = 'cp1252'
   no_stylesheets = True
   extra_css = 'h1 {font: sans-serif large;}\n.byline {font:monospace;}'
   def get browser(self):
       br = BasicNewsRecipe.get_browser(self)
       if self.username is not None and self.password is not None:
          br.open('https://www.nytimes.com/auth/login')
          br.select_form(name='login')
          br['USERID']
                        = self.username
          br['PASSWORD'] = self.password
          br.submit()
       return br
   def parse_index(self):
       soup = self.index_to_soup('https://www.nytimes.com/pages/todayspaper/index.
→html')
       def feed_title(div):
           return ''.join(div.findAll(text=True, recursive=False)).strip()
       articles = {}
       key = None
       ans = []
       for div in soup.findAll(True,
            attrs={'class':['section-headline', 'story', 'story headline']}):
            if ''.join(div['class']) == 'section-headline':
                key = string.capwords(feed_title(div))
                articles[key] = []
                ans.append(key)
            elif ''.join(div['class']) in ['story', 'story headline']:
                a = div.find('a', href=True)
                if not a:
                   continue
               url = re.sub(r'\?.*', '', a['href'])
               url += '?pagewanted=all'
                title = self.tag_to_string(a, use_alt=True).strip()
                description = ''
                pubdate = strftime('%a, %d %b')
```

(continues on next page)

(continued from previous page)

```
summary = div.find(True, attrs={'class':'summary'})
                if summary:
                    description = self.tag_to_string(summary, use_alt=False)
                feed = key if key is not None else 'Uncategorized'
                if feed not in articles:
                    articles[feed] = []
                if not 'podcasts' in url:
                    articles[feed].append(
                               dict(title=title, url=url, date=pubdate,
                                    description=description,
                                    content=''))
       ans = self.sort_index_by(ans, {'The Front Page':-1, 'Dining In, Dining Out':1,
→ 'Obituaries':2})
       ans = [(key, articles[key]) for key in ans if key in articles]
       return ans
   def preprocess_html(self, soup):
       refresh = soup.find('meta', {'http-equiv':'refresh'})
       if refresh is None:
           return soup
       content = refresh.get('content').partition('=')[2]
       raw = self.browser.open('https://www.nytimes.com'+content).read()
       return BeautifulSoup(raw.decode('cp1252', 'replace'))
```

We see several new features in this *recipe*. First, we have:

timefmt = ' [%a, %d %b, %Y]'

This sets the displayed time on the front page of the created e-book to be in the format, Day, Day\_Number Month, Year. See timefmt (page 51).

Then we see a group of directives to cleanup the downloaded HTML:

```
remove_tags_before = dict(name='h1')
remove_tags_after = dict(id='footer')
remove_tags = ...
```

These remove everything before the first <h1> tag and everything after the first tag whose id is footer. See remove\_tags (page 49), remove\_tags\_before (page 50), remove\_tags\_after (page 50).

The next interesting feature is:

```
needs_subscription = True
...
def get_browser(self):
...
```

needs\_subscription = True tells calibre that this recipe needs a username and password in order to access the content. This causes, calibre to ask for a username and password whenever you try to use this recipe. The code in *calibre.web.feeds.news.BasicNewsRecipe.get\_browser()* (page 42) actually does the login into the NYT website. Once logged in, calibre will use the same, logged in, browser instance to fetch all content. See mechanize<sup>7</sup> to understand the code in get\_browser.

<sup>&</sup>lt;sup>7</sup> https://mechanize.readthedocs.io/en/latest/

The next new feature is the *calibre.web.feeds.news.BasicNewsRecipe.parse\_index()* (page 44) method. Its job is to go to https://www.nytimes.com/pages/todayspaper/index.html and fetch the list of articles that appear in *todays* paper. While more complex than simply using *RSS*, the recipe creates an e-book that corresponds very closely to the days paper. parse\_index makes heavy use of BeautifulSoup<sup>8</sup> to parse the daily paper webpage. You can also use other, more modern parsers if you dislike BeautifulSoup. calibre comes with lxml<sup>9</sup> and html5lib<sup>10</sup>, which are the recommended parsers. To use them, replace the call to index\_to\_soup() with the following:

```
raw = self.index_to_soup(url, raw=True)
# For html5lib
import html5lib
root = html5lib.parse(raw, namespaceHTMLElements=False, treebuilder='lxml')
# For the lxml html 4 parser
from lxml import html
root = html.fromstring(raw)
```

The final new feature is the *calibre.web.feeds.news.BasicNewsRecipe.preprocess\_html()* (page 45) method. It can be used to perform arbitrary transformations on every downloaded HTML page. Here it is used to bypass the ads that the nytimes shows you before each article.

### 2.3 Tips for developing new recipes

The best way to develop new recipes is to use the command line interface. Create the recipe using your favorite Python editor and save it to a file say myrecipe.recipe. The *.recipe* extension is required. You can download content using this recipe with the command:

ebook-convert myrecipe.recipe .epub --test -vv --debug-pipeline debug

The command **ebook-convert** will download all the webpages and save them to the EPUB file myrecipe. epub. The -vv option makes ebook-convert spit out a lot of information about what it is doing. The *ebook-convert-recipe-input* --test (page 335) option makes it download only a couple of articles from at most two feeds. In addition, ebook-convert will put the downloaded HTML into the debug/input folder, where debug is the folder you specified in the *ebook-convert* --debug-pipeline (page 331) option.

Once the download is complete, you can look at the downloaded *HTML* by opening the file debug/input/index.html in a browser. Once you're satisfied that the download and preprocessing is happening correctly, you can generate e-books in different formats as shown below:

```
ebook-convert myrecipe.recipe myrecipe.epub
ebook-convert myrecipe.recipe myrecipe.mobi
...
```

If you're satisfied with your recipe, and you feel there is enough demand to justify its inclusion into the set of built-in recipes, post your recipe in the calibre recipes forum<sup>11</sup> to share it with other calibre users.

#### Note

On macOS, the command line tools are inside the calibre bundle, for example, if you installed calibre in / Applications the command line tools are in /Applications/calibre.app/Contents/MacOS/.

<sup>&</sup>lt;sup>8</sup> https://www.crummy.com/software/BeautifulSoup/bs4/doc/

<sup>9</sup> https://lxml.de/

<sup>&</sup>lt;sup>10</sup> https://github.com/html5lib/html5lib-python

<sup>&</sup>lt;sup>11</sup> https://www.mobileread.com/forums/forumdisplay.php?f=228

```
ebook-convert (page 322)
```

The command line interface for all e-book conversion.

## 2.4 Further reading

To learn more about writing advanced recipes using some of the facilities, available in BasicNewsRecipe you should consult the following sources:

```
API documentation (page 41)
```

Documentation of the BasicNewsRecipe class and all its important methods and fields.

```
BasicNewsRecipe<sup>12</sup>
```

The source code of BasicNewsRecipe

```
Built-in recipes<sup>13</sup>
```

The source code for the built-in recipes that come with calibre

```
The calibre recipes forum<sup>14</sup>
```

Lots of knowledgeable calibre recipe writers hang out here.

## 2.5 API documentation

### 2.5.1 API documentation for recipes

The API for writing recipes is defined by the *BasicNewsRecipe* (page 41)

class calibre.web.feeds.news.BasicNewsRecipe(options, log, progress\_reporter)

Base class that contains logic needed in all recipes. By overriding progressively more of the functionality in this class, you can make progressively more customized/powerful recipes. For a tutorial introduction to creating recipes, see *Adding your favorite news website* (page 33).

```
classmethod adeify_images(soup)
```

If your recipe when converted to EPUB has problems with images when viewed in Adobe Digital Editions, call this method from within *postprocess\_html()* (page 45).

```
classmethod image_url_processor(baseurl, url)
```

Perform some processing on image urls (perhaps removing size restrictions for dynamically generated images, etc.) and return the processed URL. Return None or an empty string to skip fetching the image.

#### classmethod print\_version(url)

Take a *url* pointing to the webpage with article content and return the *URL* pointing to the print version of the article. By default does nothing. For example:

```
def print_version(self, url):
    return url + '?&pagewanted=print'
```

classmethod tag\_to\_string(tag, use\_alt=True, normalize\_whitespace=True)

Convenience method to take a BeautifulSoup<sup>15</sup> Tag and extract the text from it recursively, including any CDATA sections and alt tag attributes. Return a possibly empty Unicode string.

<sup>&</sup>lt;sup>12</sup> https://github.com/kovidgoyal/calibre/blob/master/src/calibre/web/feeds/news.py

<sup>&</sup>lt;sup>13</sup> https://github.com/kovidgoyal/calibre/tree/master/recipes

<sup>&</sup>lt;sup>14</sup> https://www.mobileread.com/forums/forumdisplay.php?f=228

use\_alt: If True try to use the alt attribute for tags that don't have any textual content

tag: BeautifulSoup<sup>16</sup> Tag

#### abort\_article(msg=None)

Call this method inside any of the preprocess methods to abort the download for the current article. Useful to skip articles that contain inappropriate content, such as pure video articles.

#### abort\_recipe\_processing(msg)

Causes the recipe download system to abort the download of this recipe, displaying a simple feedback message to the user.

#### add\_toc\_thumbnail(article, src)

Call this from populate\_article\_metadata with the src attribute of an <img> tag from the article that is appropriate for use as the thumbnail representing the article in the Table of Contents. Whether the thumbnail is actually used is device dependent (currently only used by the Kindles). Note that the referenced image must be one that was successfully downloaded, otherwise it will be ignored.

#### canonicalize\_internal\_url (url, is\_link=True)

Return a set of canonical representations of url. The default implementation uses just the server hostname and path of the URL, ignoring any query parameters, fragments, etc. The canonical representations must be unique across all URLs for this news source. If they are not, then internal links may be resolved incorrectly.

#### Parameters

**is\_link** – Is True if the URL is coming from an internal link in an HTML file. False if the URL is the URL used to download an article.

#### cleanup()

Called after all articles have been download. Use it to do any cleanup like logging out of subscription sites, etc.

#### clone\_browser(br)

Clone the browser br. Cloned browsers are used for multi-threaded downloads, since mechanize is not thread safe. The default cloning routines should capture most browser customization, but if you do something exotic in your recipe, you should override this method in your recipe and clone manually.

Cloned browser instances use the same, thread-safe CookieJar by default, unless you have customized cookie handling.

#### default\_cover(cover\_file)

Create a generic cover for recipes that don't have a cover

#### download()

Download and pre-process all articles from the feeds in this recipe. This method should be called only once on a particular Recipe instance. Calling it more than once will lead to undefined behavior. :return: Path to index.html

#### extract\_readable\_article(html, url)

Extracts main article content from 'html', cleans up and returns as a (article\_html, extracted\_title) tuple. Based on the original readability algorithm by Arc90.

#### get\_article\_url(article)

Override in a subclass to customize extraction of the *URL* that points to the content for each article. Return the article URL. It is called with *article*, an object representing a parsed article from a feed. See feedparser<sup>17</sup>. By default it looks for the original link (for feeds syndicated via a service like FeedBurner or Pheedo) and if found, returns that or else returns article.link<sup>18</sup>.

#### get\_browser(\*args, \*\*kwargs)

Return a browser instance used to fetch documents from the web. By default it returns a mechanize<sup>19</sup> browser instance that supports cookies, ignores robots.txt, handles refreshes and has a random common user agent.

To customize the browser override this method in your sub-class as:

```
def get_browser(self, *a, **kw):
   br = super().get_browser(*a, **kw)
    # Add some headers
   br.addheaders += [
        ('My-Header', 'one'),
        ('My-Header2', 'two'),
    ]
    # Set some cookies
   br.set_cookie('name', 'value')
   br.set_cookie('name2', 'value2', domain='.mydomain.com')
    # Make a POST request with some data
   br.open('https://someurl.com', {'username': 'def', 'password': 'pwd'}).
\rightarrow read()
    # Do a login via a simple web form (only supported with mechanize_
\leftrightarrow browsers)
   if self.username is not None and self.password is not None:
        br.open('https://www.nytimes.com/auth/login')
        br.select_form(name='login')
        br['USERID'] = self.username
        br['PASSWORD'] = self.password
        br.submit()
    return br
```

#### get\_cover\_url()

Return a *URL* to the cover image for this issue or *None*. By default it returns the value of the member *self.cover\_url* which is normally *None*. If you want your recipe to download a cover for the e-book override this method in your subclass, or set the member variable *self.cover\_url* before this method is called.

#### get\_extra\_css()

By default returns *self.extra\_css*. Override if you want to programmatically generate the extra\_css.

#### get\_feeds()

Return a list of *RSS* feeds to fetch for this profile. Each element of the list must be a 2-element tuple of the form (title, url). If title is None or an empty string, the title from the feed is used. This method is useful if your recipe needs to do some processing to figure out the list of feeds to download. If so, override in your subclass.

#### get\_masthead\_title()

Override in subclass to use something other than the recipe title

#### get\_masthead\_url()

Return a *URL* to the masthead image for this issue or *None*. By default it returns the value of the member *self.masthead\_url* which is normally *None*. If you want your recipe to download a masthead for the e-book override this method in your subclass, or set the member variable *self.masthead\_url* before this method is called. Masthead images are used in Kindle MOBI files.

#### get\_obfuscated\_article(url)

If you set *articles\_are\_obfuscated* this method is called with every article URL. It should return the path to a file on the filesystem that contains the article HTML. That file is processed by the recursive HTML fetching engine, so it can contain links to pages/images on the web. Alternately, you can return a dictionary of the form:

{'data': <HTML data>, 'url': <the resolved URL of the article>}. This avoids needing to create temporary files. The *url* key in the dictionary is useful if the effective URL of the article is different from the URL passed into this method, for example, because of redirects. It can be omitted if the URL is unchanged.

This method is typically useful for sites that try to make it difficult to access article content automatically.

#### get\_url\_specific\_delay(url)

Return the delay in seconds before downloading this URL. If you want to programmatically determine the delay for the specified URL, override this method in your subclass, returning self.delay by default for URLs you do not want to affect.

#### Returns

A floating point number, the delay in seconds.

index\_to\_soup(url\_or\_raw, raw=False, as\_tree=False, save\_raw=None)

Convenience method that takes an URL to the index page and returns a BeautifulSoup<sup>20</sup> of it.

url\_or\_raw: Either a URL or the downloaded index page as a string

#### is\_link\_wanted(url, tag)

Return True if the link should be followed or False otherwise. By default, raises NotImplementedError which causes the downloader to ignore it.

#### **Parameters**

- url The URL to be followed
- tag The tag from which the URL was derived

#### ${\tt parse\_feeds}\ (\ )$

Create a list of articles from the list of feeds returned by *BasicNewsRecipe.get\_feeds()* (page 43). Return a list of Feed objects.

#### parse\_index()

This method should be implemented in recipes that parse a website instead of feeds to generate a list of articles. Typical uses are for news sources that have a "Print Edition" webpage that lists all the articles in the current print edition. If this function is implemented, it will be used in preference to *BasicNewsRecipe*. *parse\_feeds()* (page 44).

It must return a list. Each element of the list must be a 2-element tuple of the form ('feed title', list of articles).

Each list of articles must contain dictionaries of the form:

For an example, see the recipe for downloading *The Atlantic*. In addition, you can add 'author' for the author of the article.

If you want to abort processing for some reason and have calibre show the user a simple message instead of an error, call *abort\_recipe\_processing()* (page 42).

#### populate\_article\_metadata(article, soup, first)

Called when each HTML page belonging to article is downloaded. Intended to be used to get article metadata like author/summary/etc. from the parsed HTML (soup).

#### **Parameters**

- **article** A object of class calibre.web.feeds.Article. If you change the summary, remember to also change the text\_summary
- soup Parsed HTML belonging to this article
- first True iff the parsed HTML is the first page of the article.

#### postprocess\_book (oeb, opts, log)

Run any needed post processing on the parsed downloaded e-book.

#### **Parameters**

- oeb An OEBBook object
- opts Conversion options

#### postprocess\_html (soup, first\_fetch)

This method is called with the source of each downloaded *HTML* file, after it is parsed for links and images. It can be used to do arbitrarily powerful post-processing on the *HTML*. It should return *soup* after processing it.

#### **Parameters**

- soup A BeautifulSoup<sup>21</sup> instance containing the downloaded *HTML*.
- first\_fetch True if this is the first page of an article.

#### preprocess\_html(soup)

This method is called with the source of each downloaded *HTML* file, before it is parsed for links and images. It is called after the cleanup as specified by remove\_tags etc. It can be used to do arbitrarily powerful pre-processing on the *HTML*. It should return *soup* after processing it.

soup: A BeautifulSoup<sup>22</sup> instance containing the downloaded HTML.

#### preprocess\_image (img\_data, image\_url)

Perform some processing on downloaded image data. This is called on the raw data before any resizing is done. Must return the processed raw data. Return None to skip the image.

#### preprocess\_raw\_html (raw\_html, url)

This method is called with the source of each downloaded *HTML* file, before it is parsed into an object tree. raw\_html is a unicode string representing the raw HTML downloaded from the web. url is the URL from which the HTML was downloaded.

Note that this method acts *before* preprocess\_regexps.

This method must return the processed raw\_html as a unicode object.

#### publication\_date()

Use this method to set the date when this issue was published. Defaults to the moment of download. Must return a datetime.datetime object.

#### skip\_ad\_pages (soup)

This method is called with the source of each downloaded *HTML* file, before any of the cleanup attributes like remove\_tags, keep\_only\_tags are applied. Note that preprocess\_regexps will have already been applied. It is meant to allow the recipe to skip ad pages. If the soup represents an ad page, return the HTML of the real page. Otherwise return None.

soup: A BeautifulSoup<sup>23</sup> instance containing the downloaded HTML.

sort\_index\_by (index, weights)

Convenience method to sort the titles in *index* according to *weights. index* is sorted in place. Returns *index*.

index: A list of titles.

*weights*: A dictionary that maps weights to titles. If any titles in index are not in weights, they are assumed to have a weight of 0.

#### articles\_are\_obfuscated = False

Set to True and implement *get\_obfuscated\_article()* (page 43) to handle websites that try to make it difficult to scrape content.

#### auto\_cleanup = False

Automatically extract all the text from downloaded article pages. Uses the algorithms from the readability project. Setting this to True, means that you do not have to worry about cleaning up the downloaded HTML manually (though manual cleanup will always be superior).

#### auto\_cleanup\_keep = None

Specify elements that the auto cleanup algorithm should never remove. The syntax is a XPath expression. For example:

#### browser\_type = 'mechanize'

The simulated browser engine to use when downloading from servers. The default is to use the Python mechanize browser engine, which supports logging in. However, if you don't need logging in, consider changing this to either 'webengine' which uses an actual Chromium browser to do the network requests or 'qt' which uses the Qt Networking backend. Both 'webengine' and 'qt' support HTTP/2, which mechanize does not and are thus harder to fingerprint for bot protection services.

#### center\_navbar = True

If True the navigation bar is center aligned, otherwise it is left aligned

#### compress\_news\_images = False

Set this to False to ignore all scaling and compression parameters and pass images through unmodified. If True and the other compression parameters are left at their default values, images will be scaled to fit in the screen dimensions set by the output profile and compressed to size at most (w \* h)/16 where w x h are the scaled image dimensions.

#### compress\_news\_images\_auto\_size = 16

The factor used when auto compressing JPEG images. If set to None, auto compression is disabled. Otherwise, the images will be reduced in size to  $(w * h)/compress_news_images_auto_size$  bytes if possible by reducing the quality level, where w x h are the image dimensions in pixels. The minimum JPEG quality will be 5/100 so it is possible this constraint will not be met. This parameter can be overridden by the parameter compress\_news\_images\_max\_size which provides a fixed maximum size for images. Note that if you enable scale\_news\_images\_to\_device then the image will first be scaled and then its quality lowered until its size is less than (w \* h)/factor where w and h are now the *scaled* image dimensions. In other words, this compression happens after scaling.

#### compress\_news\_images\_max\_size = None

Set JPEG quality so images do not exceed the size given (in KBytes). If set, this parameter overrides auto compression via compress\_news\_images\_auto\_size. The minimum JPEG quality will be 5/100 so it is possible this constraint will not be met.

```
conversion_options = {}
```

Recipe specific options to control the conversion of the downloaded content into an e-book. These will override any user or plugin specified values, so only use if absolutely necessary. For example:

```
conversion_options = {
   'base_font_size' : 16,
   'linearize_tables' : True,
}
```

#### cover\_margins = (0, 0, '#ffffff')

By default, the cover image returned by get\_cover\_url() will be used as the cover for the periodical. Overriding this in your recipe instructs calibre to render the downloaded cover into a frame whose width and height are expressed as a percentage of the downloaded cover. cover\_margins = (10, 15, `#ffffff') pads the cover with a white margin 10px on the left and right, 15px on the top and bottom. Color names are defined here<sup>24</sup>. Note that for some reason, white does not always work in Windows. Use #ffffff instead

#### delay = 0

The default delay between consecutive downloads in seconds. The argument may be a floating point number to indicate a more precise time. See *get\_url\_specific\_delay()* (page 44) to implement per URL delays.

#### description = ''

A couple of lines that describe the content this recipe downloads. This will be used primarily in a GUI that presents a list of recipes.

#### encoding = None

Specify an override encoding for sites that have an incorrect charset specification. The most common being specifying latin1 and using cp1252. If None, try to detect the encoding. If it is a callable, the callable is called with two arguments: The recipe object and the source to be decoded. It must return the decoded source.

#### extra\_css = None

Specify any extra *CSS* that should be added to downloaded *HTML* files. It will be inserted into *<style>* tags, just before the closing *</head>* tag thereby overriding all *CSS* except that which is declared using the style attribute on individual *HTML* tags. Note that if you want to programmatically generate the extra\_css override the *get\_extra\_css()* (page 43) method instead. For example:

```
extra_css = '.heading { font: serif x-large }'
```

#### feeds = None

List of feeds to download. Can be either [url1, url2, ...] or [('title1', url1), ('title2', url2),...]

#### filter\_regexps = []

List of regular expressions that determines which links to ignore. If empty it is ignored. Used only if is\_link\_wanted is not implemented. For example:

filter\_regexps = [r'ads\.doubleclick\.net']

will remove all URLs that have ads.doubleclick.net in them.

Only one of *BasicNewsRecipe.match\_regexps* (page 48) or *BasicNewsRecipe.filter\_regexps* (page 47) should be defined.

#### handle\_gzip = True

Set to False if you do not want to use gzipped transfers with the mechanize browser. Note that some old servers flake out with gzip.

#### ignore\_duplicate\_articles = None

Ignore duplicates of articles that are present in more than one section. A duplicate article is an article that has the same title and/or URL. To ignore articles with the same title, set this to:

ignore\_duplicate\_articles = {'title'}

To use URLs instead, set it to:

```
ignore_duplicate_articles = {'url'}
```

To match on title or URL, set it to:

ignore\_duplicate\_articles = {'title', 'url'}

#### keep\_only\_tags = []

Keep only the specified tags and their children. For the format for specifying a tag see *BasicNewsRecipe*. *remove\_tags* (page 49). If this list is not empty, then the *<body>* tag will be emptied and re-filled with the tags that match the entries in this list. For example:

keep\_only\_tags = [dict(id=['content', 'heading'])]

will keep only tags that have an *id* attribute of "content" or "heading".

#### language = 'und'

The language that the news is in. Must be an ISO-639 code either two or three characters long

#### masthead\_url = None

By default, calibre will use a default image for the masthead (Kindle only). Override this in your recipe to provide a URL to use as a masthead.

#### match\_regexps = []

List of regular expressions that determines which links to follow. If empty, it is ignored. Used only if is\_link\_wanted is not implemented. For example:

```
match_regexps = [r'page=[0-9]+']
```

will match all URLs that have *page=some number* in them.

Only one of *BasicNewsRecipe.match\_regexps* (page 48) or *BasicNewsRecipe.filter\_regexps* (page 47) should be defined.

#### max\_articles\_per\_feed = 100

Maximum number of articles to download from each feed. This is primarily useful for feeds that don't have article dates. For most feeds, you should use *BasicNewsRecipe.oldest\_article* (page 48)

#### needs\_subscription = False

If True the GUI will ask the user for a username and password to use while downloading. If set to "optional" the use of a username and password becomes optional

#### no\_stylesheets = False

Convenient flag to disable loading of stylesheets for websites that have overly complex stylesheets unsuitable for conversion to e-book formats. If True stylesheets are not downloaded and processed

#### oldest\_article = 7.0

Oldest article to download from this news source. In days.

#### preprocess\_regexps = []

List of *regexp* substitution rules to run on the downloaded *HTML*. Each element of the list should be a two element tuple. The first element of the tuple should be a compiled regular expression and the second a callable that takes a single match object and returns a string to replace the match. For example:

```
preprocess_regexps = [
  (re.compile(r'<!--Article ends here-->.*</body>', re.DOTALL|re.IGNORECASE),
  lambda match: '</body>'),
]
```

will remove everything from <!-Article ends here-> to </body>.

#### publication\_type = 'unknown'

Publication type Set to newspaper, magazine or blog. If set to None, no publication type metadata will be written to the opf file.

#### recipe\_disabled = None

Set to a non empty string to disable this recipe. The string will be used as the disabled message

#### recipe\_specific\_options = None

Specify options specific to this recipe. These will be available for the user to customize in the Advanced tab of the Fetch News dialog or at the ebook-convert command line. The options are specified as a dictionary mapping option name to metadata about the option. For example:

When the recipe is run, self.recipe\_specific\_options will be a dict mapping option name to the option value specified by the user. When the option is unspecified by the user, it will have the value specified by 'default'. If no default is specified, the option will not be in the dict at all, when unspecified by the user.

#### recursions = 0

Number of levels of links to follow on article webpages

```
remove_attributes = []
```

List of attributes to remove from all tags. For example:

```
remove_attributes = ['style', 'font']
```

#### remove\_empty\_feeds = False

If True empty feeds are removed from the output. This option has no effect if parse\_index is overridden in the sub class. It is meant only for recipes that return a list of feeds using *feeds* or *get\_feeds()* (page 43). It is also used if you use the ignore\_duplicate\_articles option.

#### remove\_javascript = True

Convenient flag to strip all JavaScript tags from the downloaded HTML

#### remove\_tags = []

{

List of tags to be removed. Specified tags are removed from downloaded HTML. A tag is specified as a dictionary of the form:

```
name : 'tag name', #e.g. 'div'
attrs : a dictionary, #e.g. {'class': 'advertisement'}
```

All keys are optional. For a full explanation of the search criteria, see Beautiful Soup<sup>25</sup> A common example:

```
remove_tags = [dict(name='div', class_='advert')]
```

This will remove all *<div class="advert">* tags and all their children from the downloaded *HTML*.

#### remove\_tags\_after = None

Remove all tags that occur after the specified tag. For the format for specifying a tag see *BasicNewsRecipe*. remove\_tags (page 49). For example:

remove\_tags\_after = [dict(id='content')]

will remove all tags after the first element with *id="content"*.

#### remove\_tags\_before = None

Remove all tags that occur before the specified tag. For the format for specifying a tag see *BasicNewsRecipe.remove\_tags* (page 49). For example:

remove\_tags\_before = dict(id='content')

will remove all tags before the first element with *id="content"*.

#### requires\_version = (0, 6, 0)

Minimum calibre version needed to use this recipe

#### resolve\_internal\_links = False

If set to True then links in downloaded articles that point to other downloaded articles are changed to point to the downloaded copy of the article rather than its original web URL. If you set this to True, you might also need to implement *canonicalize\_internal\_url()* (page 42) to work with the URL scheme of your particular website.

#### reverse\_article\_order = False

Reverse the order of articles in each feed

#### scale\_news\_images = None

Maximum dimensions (w,h) to scale images to. If scale\_news\_images\_to\_device is True this is set to the device screen dimensions set by the output profile unless there is no profile set, in which case it is left at whatever value it has been assigned (default None).

```
scale_news_images_to_device = True
```

Rescale images to fit in the device screen dimensions set by the output profile. Ignored if no output profile is set.

#### simultaneous\_downloads = 5

Number of simultaneous downloads. Set to 1 if the server is picky. Automatically reduced to 1 if BasicNewsRecipe.delay (page 47) > 0

#### summary\_length = 500

Max number of characters in the short description

template\_css = '\n .article\_date {\n color: gray; font-family: monospace;\n }\n\n .article\_description {\n text-indent: 0pt;\n }\n\n a.article {\n font-weight: bold; text-align:left;\n }\n\n a.feed {\n font-weight: bold;\n }\n\n .calibre\_navbar {\n font-family:monospace;\n }\n '

The CSS that is used to style the templates, i.e., the navigation bars and the Tables of Contents. Rather than overriding this variable, you should use *extra\_css* in your recipe to customize look and feel.

#### timefmt = ' [%a, %d %b %Y]'

The format string for the date shown on the first page. By default: Day\_Name, Day\_Number Month\_Name Year

#### timeout = 120.0

Timeout for fetching files from server in seconds

#### title = 'Unknown News Source'

The title to use for the e-book

#### use\_embedded\_content = None

Normally we try to guess if a feed has full articles embedded in it based on the length of the embedded content. If *None*, then the default guessing is used. If *True* then the we always assume the feeds has embedded content and if *False* we always assume the feed does not have embedded content.

<sup>&</sup>lt;sup>15</sup> https://www.crummy.com/software/BeautifulSoup/bs4/doc/

<sup>&</sup>lt;sup>16</sup> https://www.crummy.com/software/BeautifulSoup/bs4/doc/

<sup>&</sup>lt;sup>17</sup> https://pythonhosted.org/feedparser/

<sup>&</sup>lt;sup>18</sup> https://pythonhosted.org/feedparser/reference-entry-link.html

<sup>&</sup>lt;sup>19</sup> https://mechanize.readthedocs.io/en/latest/

<sup>&</sup>lt;sup>20</sup> https://www.crummy.com/software/BeautifulSoup/bs4/doc

<sup>&</sup>lt;sup>21</sup> https://www.crummy.com/software/BeautifulSoup/bs4/doc/

<sup>&</sup>lt;sup>22</sup> https://www.crummy.com/software/BeautifulSoup/bs4/doc/

<sup>23</sup> https://www.crummy.com/software/BeautifulSoup/bs4/doc/

<sup>&</sup>lt;sup>24</sup> https://www.imagemagick.org/script/color.php

<sup>&</sup>lt;sup>25</sup> https://www.crummy.com/software/BeautifulSoup/bs4/doc/#searching-the-tree

### CHAPTER

### THREE

## THE E-BOOK VIEWER

calibre includes a built-in E-book viewer that can view all the major e-book formats. The E-book viewer is highly customizable and has many advanced features.

- Starting the E-book viewer (page 53)
- Navigating around an e-book (page 53)
- Highlighting text (page 54)
- Read aloud (page 55)
- Searching the text (page 55)
- Following links using only the keyboard (page 56)
- Customizing the look and feel of your reading experience (page 56)
- Dictionary lookup (page 56)
- Copying text and images (page 56)
- Zooming in on images (page 56)
- Syncing with a paper edition of the current book (page 56)
- Keyboard shortcuts (page 57)
- Non re-flowable content (page 59)
- Designing your book to work well with the calibre E-book viewer (page 59)

## 3.1 Starting the E-book viewer

You can view any of the books in your calibre library by selecting the book and pressing the *View* button. This will open up the book in the E-book viewer. You can also launch the E-book viewer by itself from the Start menu in Windows. On macOS, you can pin it to the dock and launch it from there. On Linux you can use its launcher in the desktop menus or run the command **ebook-viewer**.

# 3.2 Navigating around an e-book

You can "turn pages" in a book by either:

• Clicking in the left or right margin or the page with the mouse

- Pressing the spacebar, page up, page down or arrow keys
- On a touchscreen tapping on the text or swiping left and right

You can access the viewer controls by either:

- Right clicking on the text
- Pressing the Esc or Menu keys
- On a touchscreen by tapping the top 1/3rd of the screen

The viewer has two modes, "paged" and "flow". In paged mode the book content is presented as pages, similar to a paper book. In flow mode the text is presented continuously, like in a web browser. You can switch between them using the viewer *Preferences* under *Page layout* or by pressing the Ctrl+M key.

### 3.2.1 Bookmarks

When you are in the middle of a book and close the E-book viewer, it will remember where you stopped reading and return there the next time you open the book. You can also set bookmarks in the book by using the *Bookmarks* button in the E-book viewer controls or pressing Ctrl+B. When viewing EPUB format books, these bookmarks are actually saved in the EPUB file itself. You can add bookmarks, then send the file to a friend. When they open the file, they will be able to see your bookmarks. You can turn off this behavior in the *Miscellaneous* section of the viewer preferences.

### 3.2.2 Table of Contents

If the book you are reading defines a Table of Contents, you can access it by pressing the *Table of Contents* button. This will bring up a list of sections in the book. You can click on any of them to jump to that portion of the book.

### 3.2.3 Navigating by location

E-books, unlike paper books, have no concept of pages. You can refer to precise locations in e-books using the Go to  $\rightarrow$  Location functionality in the viewer controls.

You can use this location information to unambiguously refer to parts of the books when discussing it with friends or referring to it in other works. You can enter these locations under  $Go to \rightarrow Location$  in the viewer controls.

There is a URL you can copy to the clipboard and paste into other programs or documents. Clicking on this URL will open the book in the calibre E-book viewer at the current location.

If you click on links inside the e-book to take you to different parts of the book, such as an endnote, you can use the *Back* and *Forward* buttons in the top left corner of the viewer controls. These buttons behave just like those in a web browser.

### 3.2.4 Reference mode

calibre also has a very handy *Reference mode*. You can turn it on by clicking the *Reference mode* button in the viewer controls. Once you do this, every paragraph will have a unique number displayed at the start, made up of the section and paragraph numbers.

You can use this number to unambiguously refer to parts of the books when discussing it with friends or referring to it in other works. You can enter these numbers in the *Go to function* to navigate to a particular reference location.

## 3.3 Highlighting text

When you select text in the viewer, a little popup bar appears next to the selection. You can click the highlight button in that bar to create a highlight. You can add notes and change the color of the highlight. On a touch screen, long tap a word to select it and show the popup bar. Once in highlight mode you can change what text is selected, using touch screen friendly selection handles. Drag the handles to the top or bottom margins to scroll while selecting. You can also Shift+click or right click to extend the selection, particularly useful for multi-page selections.

You can use the *Highlights* button in the viewer controls to show a separate panel with a list of all highlights in the book, sorted by chapter.

You can browse *all highlights* in your entire calibre library by right clicking the *View* button and choosing *Browse annotations*.

Finally, if you use the calibre Content server's in browser viewer, you can have the viewer sync its annotations with the browser viewer by going to *Preferences*  $\rightarrow$  *Miscellaneous* in the viewer preferences and entering the username of the Content server viewer to sync with. Use the special value \* to sync with anonymous users.

### 3.4 Read aloud

The viewer can read book text aloud. To use it you can simply click the *Read aloud* button in the viewer controls to start reading book text aloud. The word or sentence being currently read is highlighted. Speech is synthesized from the text using either the Piper<sup>26</sup> neural text-to-speech engine or your operating system services for text-to-speech. You can change the backend and the voice being used by clicking the gear icon in the bar that is displayed while *Read aloud* is active.

You can also read aloud highlighted passages by adding the *Read aloud* button to the selection bar in the viewer preferences under *Selection behavior*.

#### \rm 1 Note

Support for text-to-speech in browsers is very incomplete and bug-ridden so how well *Read aloud* will work in the in-browser viewer is dependent on how well the underlying browser supports text-to-speech.

## 3.5 Searching the text

The viewer has very powerful search capabilities. Press the Ctrl+F key or access the viewer controls and click search. The simplest form of searching is to just search for whatever text you enter in the text box. The different forms of searching are chosen by the search mode box below the search input. Available modes are:

- 1. *Contains* The simplest default mode. The text entered in the search box is searched for anywhere. All punctuation, accents and spaces are ignored. For example, the search: Pena will match all of the following: penal, pen a, pen.a and Peña. If you select the *Case sensitive* box then accents, spaces and punctuation are no longer ignored.
- 2. Whole words Searches for whole words. So for example, the search pena will match the word Peña but not the word Penal. As with *Contains* searches above, accents and punctuation are ignored unless the *Case sensitive* box is checked.
- 3. *Nearby words* Searches for whole words that are near each other. So for example, the search calibre cool will match places where the words calibre and cool occur within sixty characters of each other. To change the number of characters add the new number to the end of the list of words. For instance, calibre cool awesome 120 will match places where the three words occur within 120 characters of each other. Note that punctuation and accents are *not* ignored for these searches.
- 4. *Regex* Interprets the search text as a *regular expression*. To learn more about using regular expressions, see *the tutorial* (page 212).

<sup>&</sup>lt;sup>26</sup> https://github.com/rhasspy/piper

## 3.6 Following links using only the keyboard

The E-book viewer has a *Hints mode* that allows you to click links in the text without using the mouse. Press the Alt+F key and all links in the current screen will be highlighted with a number or letter over them. Press the letter on your keyboard to click the link. Pressing the Esc key will abort the *Hints mode* without selecting any link.

If more than thirty five links are on-screen then some of them will have multiple letters, in which case type the first and second, or the first and press Enter to activate. You can also use the Backspace key to undo a mistake in typing.

## 3.7 Customizing the look and feel of your reading experience

You can change font sizes on the fly by using *Font size* in the viewer controls or Ctrl++ or Ctrl+- or holding the Ctrl key and using the mouse wheel.

Colors can be changed in the Colors section of the viewer preferences.

You can change the number of pages displayed on the screen as well as page margins in *Page layout* in the viewer preferences.

You can display custom headers and footers such as time left to read, current chapter title, book position, etc. via the *Headers and footers* section of the viewer preferences.

More advanced customization can be achieved by the *Styles* settings. Here you can specify a background image to display under the text and also a stylesheet you can set that will be applied to every book. Using it you can do things like change paragraph styles, text justification, etc. For examples of custom stylesheets used by calibre's users, see the forums<sup>27</sup>.

## 3.8 Dictionary lookup

You can look up the meaning of words in the current book by double clicking or long tapping the word you want to lookup and then clicking the lookup button that looks like a library.

## 3.9 Copying text and images

You can select text and images by dragging the content with your mouse and then right clicking and selecting *Copy* to copy to the clipboard. The copied material can be pasted into another application as plain text and images.

### 3.10 Zooming in on images

You can zoom in to show an image at full size in a separate window by either double clicking or long tapping on it. You can also right click on it and choose *View image*.

## 3.11 Syncing with a paper edition of the current book

Some e-books, that have corresponding print editions, include metadata that marks the start of each paper page. For such e-books, the viewer allows you to jump to a particular paper edition page via the *Go to* button in the viewer controls. You can also optionally display the paper page corresponding to the current location in the book's headers or footers via the viewer settings, by adding *Pages from paper edition* to either the header or the footer.

<sup>&</sup>lt;sup>27</sup> https://www.mobileread.com/forums/showthread.php?t=51500

# 3.12 Keyboard shortcuts

The viewer has extensive keyboard shortcuts, like the rest of calibre. They can be customised in the viewer *Preferences*. The default shortcuts are listed below:

17.	
Key-	Action
board	
short-	
cut	
Home,	Scroll to the start of the current file in a multi file book
Ctrl+Ar	
Ctrl+Ar	
	Scroll to the beginning of the book
Ctrl+En	Scroll to the end of the book
End,	Scroll to the end of the current file in a multi file book
Ctrl <b>+</b> Ar	
Ctrl <b>+</b> Ar	
Ar-	Scroll backwards, smoothly in flow mode and by screen fulls in paged mode
rowUp	San 11 farmanda and a farman da and ha san an faile in march and a
Ar-	Scroll forwards, smoothly in flow mode and by screen fulls in paged mode
row-	
Down	Scroll leftwards by a little in flow mode and by a page in paged mode
Ar- rowLeft	Scron lertwards by a nulle in now mode and by a page in paged mode
Ar-	Scroll rightwards by a little in flow mode and by a page in paged mode
rowRigh	Seron rightwards by a little in now mode and by a page in paged mode
-	Scroll backwards by screen-fulls
- ) 1 /	
Shift+S	
PageDow	Scroll forwards by screen-fulls
Space-	
bar	
	Scroll to the previous section
	Scroll to the next section
Alt+Arr	
Alt+Arr	
	Toggle Table of Contents
	Read aloud
Alt+P	Change settings quickly by creating and switching to <i>profiles</i>
Alt+f	Follow links with the keyboard
Ctrl+C	Copy to clipboard Copy current location to clipboard
Alt+C	Copy current location to chipboard Copy current location as calibre:// URL to clipboard
/,	Start search
/, Ctrl+f,	
Ctrifr, Cmd <b>+</b> f	
F3,	Find next
Enter	
2	

continues on next page

	Table 1 – continued from previous page	
Key- board short- cut	Action	
Shift <b>+</b> F	Find previous	
Shift <b>+</b> E		
	Increase font size	
Meta+Pl	December for the last	
	Decrease font size	
Meta+Mi	Destage default fant size	
	Restore default font size	
Ctrl+]	Increase number of pages per screen	
	Decrease number of pages per screen	
	Make number of pages per screen automatic	
F11, Ctrl+Sh	Toggle full screen	
	Toggle between Paged mode and Flow mode for text layout	
	Toggle the scrollbar	
	Toggle the Reference mode	
Ctrl <b>+</b> B	Show/hide bookmarks	
	New bookmark	
Ctrl+N,	Show the book metadata	
Ctrl+E		
Ctrl <b>+</b> Al	Reload book	
Ctrl+Al		
	Alter the current selection forward by a word	
	Alter the current selection backwards by a word	
	Alter the current selection forward by a character	
	Alter the current selection backwards by a character	
	Alter the current selection forward by a line	
	Extend the current selection to the start of the line	
	Extend the current selection to the end of the line	
	Alter the current selection backwards by a line	
	Alter the current selection forward by a paragraph	
Ctrl <b>+</b> Sh	Alter the current selection backwards by a paragraph	
Esc,	Show the E-book viewer controls	
MenuKey		
Ctrl+Cc	Show E-book viewer preferences	
Ctrl+Es		
Meta <b>+</b> Es		
Meta+Cc	continues on next page	

Table 1 – continued from previou	us page
----------------------------------	---------

continues on next page

Key- board short- cut	Action
Ctrl+G,	Go to a specified book location or position
;,:	
Ctrl+Sp	Toggle auto-scroll
Alt+Arr	Auto scroll faster
Alt+Arr	Auto scroll slower
Ctrl+I	Show/hide Inspector
Ctrl+L	Show/hide the word lookup panel
Ctrl <b>+</b> Q	Quit
(Cmd+Q	
on ma-	
COS)	
Ctrl+P	Print book to PDF
	Toggle the toolbar
Ctrl+H	Toggle the highlights panel
Ctrl+D	Edit this book

#### Table 1 - continued from previous page

### 3.13 Non re-flowable content

Some books have very wide content that cannot be broken up at page boundaries. For example tables or tags. In such cases, you should switch the viewer to *flow mode* by pressing Ctrl+M to read this content. Alternately, you can also add the following CSS to the *Styles* section of the viewer preferences to force the viewer to break up lines of text in tags:

code, pre { white-space: pre-wrap }

## 3.14 Designing your book to work well with the calibre E-book viewer

The calibre E-book viewer will set the is-calibre-viewer class on the root element. So you can write CSS rules that apply only for it. Additionally, the viewer will set the following classes on the body element:

```
body.calibre-viewer-dark-colors
Set when using a dark color scheme
```

```
body.calibre-viewer-light-colors
Set when using a light color scheme
```

```
body.calibre-viewer-paginated
Set when in paged mode
```

body.calibre-viewer-scrolling Set when in flow (non-paginated) mode

```
body.calibre-footnote-container
Set when displaying a popup footnote
```

Finally, you can use the calibre color scheme colors via CSS variables<sup>28</sup>. The calibre E-book viewer defines the following variables: --calibre-viewer-background-color, --calibre-viewer-foreground-color and optionally --calibre-viewer-link-color in color themes that define a link color.

 $<sup>^{28}\</sup> https://developer.mozilla.org/en-US/docs/Web/CSS/Using\_CSS\_custom\_properties$ 

## **E-BOOK CONVERSION**

calibre has a conversion system that is designed to be very easy to use. Normally, you just add a book to calibre, click convert and calibre will try hard to generate output that is as close as possible to the input. However, calibre accepts a very large number of input formats, not all of which are as suitable as others for conversion to e-books. In the case of such input formats, or if you just want greater control over the conversion system, calibre has a lot of options to fine tune the conversion process. Note however that calibre's conversion system is not a substitute for a full blown e-book editor. To edit e-books, I recommend first converting them to EPUB or AZW3 using calibre and then using the *Edit book* feature to get them into perfect shape. You can then use the edited e-book as input for conversion into other formats in calibre.

This document will refer mainly to the conversion settings as found in the conversion dialog, pictured below. All these settings are also available via command line interface to conversion, documented at *ebook-convert* (page 322). In calibre, you can obtain help on any individual setting by holding your mouse over it, a tooltip will appear describing the setting.

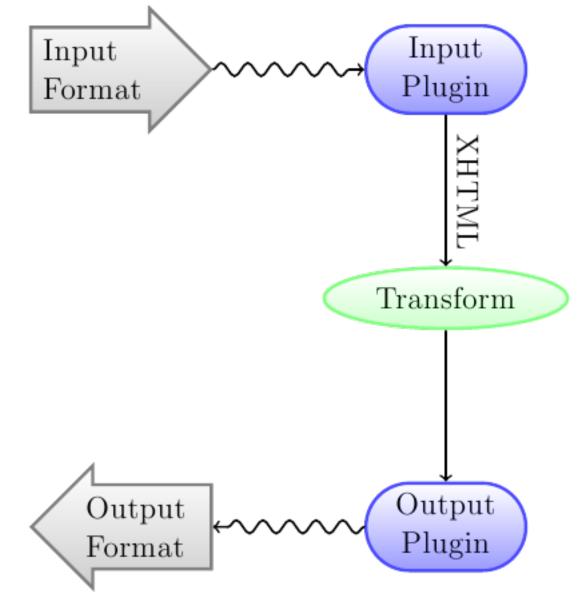
캳 💿 Convert The Cathedral a	nd the Bazaar: Musings on Linux and Open Source by an Acci	dental Revolut	ionary	$\bigcirc \bigcirc \bigcirc$	×
Input format: FB2 🗸			<u>O</u> utp	ut format: EPU	в 🕶
	Book Cover				
Metadata	"The must important book actor to concerning to have with implements of the foregoing program for a second	<u>T</u> itle:	1 Open Source by an Accide	ental Revolution	ary
	THE CATHEDRA	<u>A</u> uthor(s):	Eric S. Raymond		~
Look & Feel	& THE BAZAAR	_	Raymond, Eric S.		
	MUSINGS ON LINUX AND OPEN SOURCE By an accidental revolutionary	<u>P</u> ublisher:	O'Reilly		~
2		Tags:			
Page Setup		<u>S</u> eries:	Book 1.00		•
			500K 1.00		~
Structure Detection	ERIC S. RAYMOND				
	WITH A FOREWORD BY BOB YOUNG, CHAIRMAN & CED OF RED HAT, INC.		Comments		~
Table of			ce provides the competitive	-	
Contents	Change <u>c</u> over image:	Report, 56	et Age. According to the Au percent of IT managers into	erviewed at	
		of open so	o companies are already us urce software in their infras	structure and	
FB2 Input			percent will install it in the n		÷
	Use cover from <u>s</u> ource file				
EPUB Output	List of known series. You can add new series.				
<b>—</b> ·					
2					
Debug					
	Restore Defaults		<b>~</b> c	OK 🕜 Can	cel

### Contents

- *Introduction* (page 63)
- *Look & feel* (page 64)
- Page setup (page 67)
- *Heuristic processing* (page 67)
- Search & replace (page 69)
- *Structure detection* (page 69)
- Table of Contents (page 70)
- Using images as chapter titles when converting HTML input documents (page 72)
- Using tag attributes to supply the text for entries in the Table of Contents (page 72)
- How options are set/saved for conversion (page 72)
- Format specific tips (page 73)

## 4.1 Introduction

The first thing to understand about the conversion system is that it is designed as a pipeline. Schematically, it looks like this:

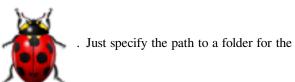


The input format is first converted to XHTML by the appropriate *Input plugin*. This HTML is then *transformed*. In the last step, the processed XHTML is converted to the specified output format by the appropriate *Output plugin*. The results of the conversion can vary greatly, based on the input format. Some formats convert much better than others. A list of the best source formats for conversion is available *here* (page 128).

The transforms that act on the XHTML output are where all the work happens. There are various transforms, for example, to insert book metadata as a page at the start of the book, to detect chapter headings and automatically create a Table of Contents, to proportionally adjust font sizes, et cetera. It is important to remember that all the transforms act on the

XHTML output by the *Input plugin*, not on the input file itself. So, for example, if you ask calibre to convert an RTF file to EPUB, it will first be converted to XHTML internally, the various transforms will be applied to the XHTML and then the *Output plugin* will create the EPUB file, automatically generating all metadata, Table of Contents, et cetera.

You can see this process in action by using the debug option



debug output. During conversion, calibre will place the XHTML generated by the various stages of the conversion pipeline in different sub-folders. The four sub-folders are:

Folder	Description
input	This contains the HTML output by the Input plugin. Use this to debug the Input plugin.
parsed	The result of pre-processing and converting to XHTML the output from the Input plugin. Use to debug structure detection.
struc- ture	Post structure detection, but before CSS flattening and font size conversion. Use to debug font size conversion and CSS transforms.
pro- cessed	Just before the e-book is passed to the Output plugin. Use to debug the Output plugin.

Table 1:	Stages of	the conversion	pipeline
----------	-----------	----------------	----------

If you want to edit the input document a little before having calibre convert it, the best thing to do is edit the files in the input sub-folder, then zip it up, and use the ZIP file as the input format for subsequent conversions. To do this use the *Edit meta information* dialog to add the ZIP file as a format for the book and then, in the top left corner of the conversion dialog, select ZIP as the input format.

This document will deal mainly with the various transforms that operate on the intermediate XHTML and how to control them. At the end are some tips specific to each input/output format.

## 4.2 Look & feel

Contents	
• Fonts (page 65)	
• <i>Text</i> (page 65)	
• <i>Layout</i> (page 66)	
• <i>Styling</i> (page 66)	
• Transform styles (page 66)	
• Transform HTML (page 67)	

This group of options controls various aspects of the look and feel of the converted e-book.

### 4.2.1 Fonts

One of the nicest features of the e-reading experience is the ability to easily adjust font sizes to suit individual needs and lighting conditions. calibre has sophisticated algorithms to ensure that all the books it outputs have a consistent font sizes, no matter what font sizes are specified in the input document.

The base font size of a document is the most common font size in that document, i.e., the size of the bulk of text in that document. When you specify a *Base font size*, calibre automatically rescales all font sizes in the document proportionately, so that the most common font size becomes the specified base font size and other font sizes are rescaled appropriately. By choosing a larger base font size, you can make the fonts in the document larger and vice versa. When you set the base font size, for best results, you should also set the font size key.

Normally, calibre will automatically choose a base font size appropriate to the output profile you have chosen (see *Page setup* (page 67)). However, you can override this here in case the default is not suitable for you.

The *Font size key* option lets you control how non-base font sizes are rescaled. The font rescaling algorithm works using a font size key, which is simply a comma-separated list of font sizes. The font size key tells calibre how many "steps" bigger or smaller a given font size should be compared to the base font size. The idea is that there should be a limited number of font sizes in a document. For example, one size for the body text, a couple of sizes for different levels of headings and a couple of sizes for super/sub scripts and footnotes. The font size key allows calibre to compartmentalize the font sizes in the input documents into separate "bins" corresponding to the different logical font sizes.

Let's illustrate with an example. Suppose the source document we are converting was produced by someone with excellent eyesight and has a base font size of 8pt. That means the bulk of the text in the document is sized at 8pts, while headings are somewhat larger (say 10 and 12pt) and footnotes somewhat smaller at 6pt. Now if we use the following settings:

```
Base font size : 12pt
Font size key : 7, 8, 10, 12, 14, 16, 18, 20
```

The output document will have a base font size of 12pt, headings of 14 and 16pt and footnotes of 8pt. Now suppose we want to make the largest heading size stand out more and make the footnotes a little larger as well. To achieve this, the font key should be changed to:

New font size key : 7, 9, 12, 14, 18, 20, 22

The largest headings will now become 18pt, while the footnotes will become 9pt. You can play with these settings to try and figure out what would be optimum for you by using the font rescaling wizard, which can be accessed by clicking the little button next to the *Font size key* setting.

All the font size rescaling in the conversion can also be disabled here, if you would like to preserve the font sizes in the input document.

A related setting is *Line height*. Line height controls the vertical height of lines. By default, (a line height of 0), no manipulation of line heights is performed. If you specify a non-default value, line heights will be set in all locations that don't specify their own line heights. However, this is something of a blunt weapon and should be used sparingly. If you want to adjust the line heights for some section of the input, it's better to use the *Extra CSS* (page 66).

In this section you can also tell calibre to embed any referenced fonts into the book. This will allow the fonts to work on reader devices even if they are not available on the device.

### 4.2.2 Text

Text can be either justified or not. Justified text has extra spaces between words to give a smooth right margin. Some people prefer justified text, others do not. Normally, calibre will preserve the justification in the original document. If you want to override it you can use the *Text justification* option in this section.

You can also tell calibre to *Smarten punctuation* which will replace plain quotes, dashes and ellipses with their typographically correct alternatives. Note that this algorithm is not perfect so it is worth reviewing the results. The reverse, namely, *Unsmarted punctuation* is also available.

Finally, there is *Input character encoding*. Older documents sometimes don't specify their character encoding. When converted, this can result in non-English characters or special characters like smart quotes being corrupted. calibre tries to auto-detect the character encoding of the source document, but it does not always succeed. You can force it to assume a particular character encoding by using this setting. *cp1252* is a common encoding for documents produced using Windows software. You should also read *How do I convert my file containing non-English characters, or smart quotes?* (page 128) for more on encoding issues.

### 4.2.3 Layout

Normally, paragraphs in XHTML are rendered with a blank line between them and no leading text indent. calibre has a couple of options to control this. *Remove spacing between paragraphs* forcefully ensure that all paragraphs have no inter paragraph spacing. It also sets the text indent to 1.5em (can be changed) to mark the start of every paragraph. *Insert blank line* does the opposite, guaranteeing that there is exactly one blank line between each pair of paragraphs. Both these options are very comprehensive, removing spacing, or inserting it for *all* paragraphs (technically and <div> tags). This is so that you can just set the option and be sure that it performs as advertised, irrespective of how messy the input file is. The one exception is when the input file uses hard line breaks to implement inter-paragraph spacing.

If you want to remove the spacing between all paragraphs, except a select few, don't use these options. Instead add the following CSS code to *Extra CSS* (page 66):

```
p, div { margin: 0pt; border: 0pt; text-indent: 1.5em }
.spacious { margin-bottom: 1em; text-indent: 0pt; }
```

Then, in your source document, mark the paragraphs that need spacing with *class="spacious"*. If your input document is not in HTML, use the Debug option, described in the Introduction to get HTML (use the input sub-folder).

Another useful options is *Linearize tables*. Some badly designed documents use tables to control the layout of text on the page. When converted these documents often have text that runs off the page and other artifacts. This option will extract the content from the tables and present it in a linear fashion. Note that this option linearizes *all* tables, so only use it if you are sure the input document does not use tables for legitimate purposes, like presenting tabular information.

### 4.2.4 Styling

The *Extra CSS* option allows you to specify arbitrary CSS that will be applied to all HTML files in the input. This CSS is applied with very high priority and so should override most CSS present in the **input document** itself. You can use this setting to fine tune the presentation/layout of your document. For example, if you want all paragraphs of class *endnote* to be right aligned, just add:

.endnote { text-align: right }

or if you want to change the indentation of all paragraphs:

p { text-indent: 5mm; }

*Extra CSS* is a very powerful option, but you do need an understanding of how CSS works to use it to its full potential. You can use the debug pipeline option described above to see what CSS is present in your input document.

A simpler option is to use *Filter style information*. This allows you to remove all CSS properties of the specified types from the document. For example, you can use it to remove all colors or fonts.

### 4.2.5 Transform styles

This is the most powerful styling related facility. You can use it to define rules that change styles based on various conditions. For example you can use it to change all green colors to blue, or remove all bold styling from the text or color all headings a certain color, etc.

### 4.2.6 Transform HTML

Similar to transform styles, but allows you to make changes to the HTML content of the book. You can replace one tag with another, add classes or other attributes to tags based on their content, etc.

## 4.3 Page setup

The *Page setup* options are for controlling screen layout, like margins and screen sizes. There are options to setup page margins, which will be used by the output plugin, if the selected output format supports page margins. In addition, you should choose an Input profile and an output profile. Both sets of profiles basically deal with how to interpret measurements in the input/output documents, screen sizes and default font rescaling keys.

If you know that the file you are converting was intended to be used on a particular device/software platform, choose the corresponding input profile, otherwise just choose the default input profile. If you know the files you are producing are meant for a particular device type, choose the corresponding output profile. Otherwise, choose one of the Generic output profiles. If you are converting to MOBI or AZW3 then you will almost always want to choose one of the Kindle output profiles. Otherwise, your best bet for modern E-book reading devices is to choose the *Generic e-ink HD* output profile.

The output profile also controls the screen size. This will cause, for example, images to be auto-resized to be fit to the screen in some output formats. So choose a profile of a device that has a screen size similar to your device.

## 4.4 Heuristic processing

Heuristic processing provides a variety of functions which can be used to try and detect and correct common problems in poorly formatted input documents. Use these functions if your input document suffers from poor formatting. Because these functions rely on common patterns, be aware that in some cases an option may lead to worse results, so use with care. As an example, several of these options will remove all non-breaking-space entities, or may include false positive matches relating to the function.

#### Enable heuristic processing

This option activates calibre's *Heuristic processing* stage of the conversion pipeline. This must be enabled in order for various sub-functions to be applied

#### Unwrap lines

Enabling this option will cause calibre to attempt to detect and correct hard line breaks that exist within a document using punctuation clues and line length. calibre will first attempt to detect whether hard line breaks exist, if they do not appear to exist calibre will not attempt to unwrap lines. The line-unwrap factor can be reduced if you want to 'force' calibre to unwrap lines.

#### Line-unwrap factor

This option controls the algorithm calibre uses to remove hard line breaks. For example, if the value of this option is 0.4, that means calibre will remove hard line breaks from the end of lines whose lengths are less than the length of 40% of all lines in the document. If your document only has a few line breaks which need correction, then this value should be reduced to somewhere between 0.1 and 0.2.

#### Detect and markup unformatted chapter headings and sub headings

If your document does not have chapter headings and titles formatted differently from the rest of the text, calibre can use this option to attempt to detect them and surround them with heading tags. <h2> tags are used for chapter headings; <h3> tags are used for any titles that are detected.

This function will not create a TOC, but in many cases it will cause calibre's default chapter detection settings to correctly detect chapters and build a TOC. Adjust the XPath under Structure detection if a TOC is not automatically created. If there are no other headings used in the document then setting "//h:h2" under Structure detection would be the easiest way to create a TOC for the document.

The inserted headings are not formatted, to apply formatting use the *Extra CSS* option under the Look and Feel conversion settings. For example, to center heading tags, use the following:

h2, h3 { text-align: center }

#### Renumber sequences of <h1> or <h2> tags

Some publishers format chapter headings using multiple <h1> or <h2> tags sequentially. calibre's default conversion settings will cause such titles to be split into two pieces. This option will re-number the heading tags to prevent splitting.

#### Delete blank lines between paragraphs

This option will cause calibre to analyze blank lines included within the document. If every paragraph is interleaved with a blank line, then calibre will remove all those blank paragraphs. Sequences of multiple blank lines will be considered scene breaks and retained as a single paragraph. This option differs from the *Remove paragraph spacing* option under *Look and Feel* in that it actually modifies the HTML content, while the other option modifies the document styles. This option can also remove paragraphs which were inserted using calibre's *Insert blank line* option.

#### Ensure scene breaks are consistently formatted

With this option calibre will attempt to detect common scene-break markers and ensure that they are center aligned. 'Soft' scene break markers, i.e. scene breaks only defined by extra white space, are styled to ensure that they will not be displayed in conjunction with page breaks.

#### Replace scene breaks

If this option is configured then calibre will replace scene break markers it finds with the replacement text specified by the user. Please note that some ornamental characters may not be supported across all reading devices.

In general you should avoid using HTML tags, calibre will discard any tags and use pre-defined markup. <hr /> tags, i.e. horizontal rules, and <img> tags are exceptions. Horizontal rules can optionally be specified with styles, if you choose to add your own style be sure to include the 'width' setting, otherwise the style information will be discarded. Image tags can used, but calibre does not provide the ability to add the image during conversion, this must be done after the fact using the 'Edit book' feature.

# Example image tag (place the image within an 'Images' folder inside the EPUB after conversion):

<img style="width:10%" src="../Images/scenebreak.png" />

#### Example horizontal rule with styles:

<hr style="width:20%;padding-top: 1px;border-top: 2px ridge black;border-bottom: 2px groove black;"/>

#### Remove unnecessary hyphens

calibre will analyze all hyphenated content in the document when this option is enabled. The document itself is used as a dictionary for analysis. This allows calibre to accurately remove hyphens for any words in the document in any language, along with made-up and obscure scientific words. The primary drawback is words appearing only a single time in the document will not be changed. Analysis happens in two passes, the first pass analyzes line endings. Lines are only unwrapped if the word exists with or without a hyphen in the document. The second pass analyzes all hyphenated words throughout the document, hyphens are removed if the word exists elsewhere in the document without a match.

#### Italicize common words and patterns

When enabled, calibre will look for common words and patterns that denote italics and italicize them. Examples are common text conventions such as ~word~ or phrases that should generally be italicized, e.g. latin phrases like 'etc.' or 'et cetera'.

#### Replace entity indents with CSS indents

Some documents use a convention of defining text indents using non-breaking space entities. When this option is enabled calibre will attempt to detect this sort of formatting and convert them to a 3% text indent using CSS.

# 4.5 Search & replace

These options are useful primarily for conversion of PDF documents or OCR conversions, though they can also be used to fix many document specific problems. As an example, some conversions can leaves behind page headers and footers in the text. These options use regular expressions to try and detect headers, footers, or other arbitrary text and remove or replace them. Remember that they operate on the intermediate XHTML produced by the conversion pipeline. There is a wizard to help you customize the regular expressions for your document. Click the magic wand beside the expression box, and click the 'Test' button after composing your search expression. Successful matches will be highlighted in Yellow.

The search works by using a Python regular expression. All matched text is simply removed from the document or replaced using the replacement pattern. The replacement pattern is optional, if left blank then text matching the search pattern will be deleted from the document. You can learn more about regular expressions and their syntax at *All about using regular expressions in calibre* (page 212).

# 4.6 Structure detection

Structure detection involves calibre trying its best to detect structural elements in the input document, when they are not properly specified. For example, chapters, page breaks, headers, footers, etc. As you can imagine, this process varies widely from book to book. Fortunately, calibre has very powerful options to control this. With power comes complexity, but if once you take the time to learn the complexity, you will find it well worth the effort.

# 4.6.1 Chapters and page breaks

calibre has two sets of options for *chapter detection* and *inserting page breaks*. This can sometimes be slightly confusing, as by default, calibre will insert page breaks before detected chapters as well as the locations detected by the page breaks option. The reason for this is that there are often location where page breaks should be inserted that are not chapter boundaries. Also, detected chapters can be optionally inserted into the auto generated Table of Contents.

calibre uses *XPath*, a powerful language to allow the user to specify chapter boundaries/page breaks. XPath can seem a little daunting to use at first, fortunately, there is a *XPath tutorial* (page 156) in the User Manual. Remember that Structure detection operates on the intermediate XHTML produced by the conversion pipeline. Use the debug option described in the *Introduction* (page 63) to figure out the appropriate settings for your book. There is also a button for a XPath wizard to help with the generation of simple XPath expressions.

By default, calibre uses the following expression for detecting chapters:

This expression is rather complex, because it tries to handle a number of common cases simultaneously. What it means is that calibre will assume chapters start at either  $\langle h1 \rangle$  or  $\langle h2 \rangle$  tags that have any of the words (*chapter*, *book*, *section or part*) in them or that have the *class="chapter"* attribute.

A related option is *Chapter mark*, which allows you to control what calibre does when it detects a chapter. By default, it will insert a page break before the chapter. You can have it insert a ruled line instead of, or in addition to the page break. You can also have it do nothing.

The default setting for detecting page breaks is:

//\*[name()='h1' or name()='h2']

which means that calibre will insert page breaks before every  $\langle h1 \rangle$  and  $\langle h2 \rangle$  tag by default.

#### Note

The default expressions may change depending on the input format you are converting.

### 4.6.2 Miscellaneous

There are a few more options in this section.

#### Insert metadata as page at start of book

One of the great things about calibre is that it allows you to maintain very complete metadata about all of your books, for example, a rating, tags, comments, etc. This option will create a single page with all this metadata and insert it into the converted e-book, typically just after the cover. Think of it as a way to create your own customised book jacket.

#### Remove first image

Sometimes, the source document you are converting includes the cover as part of the book, instead of as a separate cover. If you also specify a cover in calibre, then the converted book will have two covers. This option will simply remove the first image from the source document, thereby ensuring that the converted book has only one cover, the one specified in calibre.

# 4.7 Table of Contents

When the input document has a Table of Contents in its metadata, calibre will just use that. However, a number of older formats either do not support a metadata based Table of Contents, or individual documents do not have one. In these cases, the options in this section can help you automatically generate a Table of Contents in the converted e-book, based on the actual content in the input document.

#### Note

Using these options can be a little challenging to get exactly right. If you prefer creating/editing the Table of Contents by hand, convert to the EPUB or AZW3 formats and select the checkbox at the bottom of the Table of Contents section of the conversion dialog that says *Manually fine-tune the Table of Contents after conversion*. This will launch the ToC Editor tool after the conversion. It allows you to create entries in the Table of Contents by simply clicking the place in the book where you want the entry to point. You can also use the ToC Editor by itself, without doing a conversion. Go to *Preferences*  $\rightarrow$  *Interface*  $\rightarrow$  *Toolbars* and add the *ToC Editor* to the main toolbar. Then just select the book you want to edit and click the *ToC Editor* button.

The first option is *Force use of auto-generated Table of Contents*. By checking this option you can have calibre override any Table of Contents found in the metadata of the input document with the auto generated one.

The default way that the creation of the auto generated Table of Contents works is that, calibre will first try to add any detected chapters to the generated table of contents. You can learn how to customize the detection of chapters in the *Structure detection* (page 69) section above. If you do not want to include detected chapters in the generated table of contents, check the *Do not add detected chapters* option.

If less than the *Chapter threshold* number of chapters were detected, calibre will then add any hyperlinks it finds in the input document to the Table of Contents. This often works well: many input documents include a hyperlinked Table of Contents right at the start. The *Number of links* option can be used to control this behavior. If set to zero, no links are added. If set to a number greater than zero, at most that number of links is added.

calibre will automatically filter duplicates from the generated Table of Contents. However, if there are some additional undesirable entries, you can filter them using the *TOC Filter* option. This is a regular expression that will match the title

of entries in the generated table of contents. Whenever a match is found, it will be removed. For example, to remove all entries titles "Next" or "Previous" use:

Next|Previous

The *Level 1,2,3 TOC* options allow you to create a sophisticated multi-level Table of Contents. They are XPath expressions that match tags in the intermediate XHTML produced by the conversion pipeline. See the *Introduction* (page 63) for how to get access to this XHTML. Also read the *XPath tutorial* (page 156), to learn how to construct XPath expressions. Next to each option is a button that launches a wizard to help with the creation of basic XPath expressions. The following simple example illustrates how to use these options.

Suppose you have an input document that results in XHTML that look like this:

```
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <title>Sample document</title>
    </head>
    <body>
        <h1>Chapter 1</h1>
        . . .
        <h2>Section 1.1</h2>
        . . .
        <h2>Section 1.2</h2>
        . . .
        <h1>Chapter 2</h1>
        . . .
        <h2>Section 2.1</h2>
        . . .
    </body>
</html>
```

Then, we set the options as:

Level 1 TOC : //h:h1 Level 2 TOC : //h:h2

This will result in an automatically generated two level Table of Contents that looks like:

```
Chapter 1
Section 1.1
Section 1.2
Chapter 2
Section 2.1
```

#### 🛕 Warning

Not all output formats support a multi level Table of Contents. You should first try with EPUB output. If that works, then try your format of choice.

# 4.8 Using images as chapter titles when converting HTML input documents

Suppose you want to use an image as your chapter title, but still want calibre to be able to automatically generate a Table of Contents for you from the chapter titles. Use the following HTML markup to achieve this:

```
<html>
<body>
<h2>Chapter 1</h2>
chapter 1 text...
<h2 title="Chapter 2"><img src="chapter2.jpg" /></h2>
chapter 2 text...
</body>
</html>
```

Set the *Level 1 TOC* setting to //h:h2. Then, for chapter two, calibre will take the title from the value of the title attribute on the <h2> tag, since the tag has no text.

# 4.9 Using tag attributes to supply the text for entries in the Table of Contents

If you have particularly long chapter titles and want shortened versions in the Table of Contents, you can use the title attribute to achieve this, for example:

Set the *Level 1 TOC* setting to //h:h2/@title. Then calibre will take the title from the value of the title attribute on the <h2> tags, instead of using the text inside the tag. Note the trailing /@title on the XPath expression, you can use this form to tell calibre to get the text from any attribute you like.

# 4.10 How options are set/saved for conversion

There are two places where conversion options can be set in calibre. The first is in Preferences->Conversion. These settings are the defaults for the conversion options. Whenever you try to convert a new book, the settings set here will be used by default.

You can also change settings in the conversion dialog for each book conversion. When you convert a book, calibre remembers the settings you used for that book, so that if you convert it again, the saved settings for the individual book will take precedence over the defaults set in *Preferences*. You can restore the individual settings to defaults by using the *Restore defaults* button in the individual book conversion dialog. You can remove the saved settings for a group of books by selecting all the books and then clicking the *Edit metadata* button to bring up the bulk metadata edit dialog, near the bottom of the dialog is an option to remove stored conversion settings.

When you bulk convert a set of books, settings are taken in the following order (last one wins):

• From the defaults set in Preferences->Conversion

- From the saved conversion settings for each book being converted (if any). This can be turned off by the option in the top left corner of the Bulk conversion dialog.
- From the settings set in the Bulk conversion dialog

Note that the final settings for each book in a Bulk conversion will be saved and re-used if the book is converted again. Since the highest priority in Bulk Conversion is given to the settings in the Bulk conversion dialog, these will override any book specific settings. So you should only bulk convert books together that need similar settings. The exceptions are metadata and input format specific settings. Since the Bulk conversion dialog does not have settings for these two categories, they will be taken from book specific settings (if any) or the defaults.

#### Note

You can see the actual settings used during any conversion by clicking the rotating icon in the lower right corner and then double clicking the individual conversion job. This will bring up a conversion log that will contain the actual settings used, near the top.

# 4.11 Format specific tips

Here you will find tips specific to the conversion of particular formats. Options specific to particular format, whether input or output are available in the conversion dialog under their own section, for example *TXT input* or *EPUB output*.

# 4.11.1 Convert Microsoft Word documents

calibre can automatically convert . doex files created by Microsoft Word 2007 and newer. Just add the file to calibre and click convert.

#### Note

There is a demo .docx file<sup>29</sup> that demonstrates the capabilities of the calibre conversion engine. Just download it and convert it to EPUB or AZW3 to see what calibre can do.

calibre will automatically generate a Table of Contents based on headings if you mark your headings with the Heading 1, Heading 2, etc. styles in Microsoft Word. Open the output e-book in the calibre E-book viewer and click the *Table of Contents* button to view the generated Table of Contents.

#### Older .doc files

For older .doc files, you can save the document as HTML with Microsoft Word and then convert the resulting HTML file with calibre. When saving as HTML, be sure to use the "Save as Web Page, Filtered" option as this will produce clean HTML that will convert well. Note that Word produces really messy HTML, converting it can take a long time, so be patient. If you have a newer version of Word available, you can directly save it as .docx as well.

Another alternative is to use the free LibreOffice. Open your .doc file in LibreOffice and save it as .docx, which can be directly converted in calibre.

# 4.11.2 Convert TXT documents

TXT documents have no well defined way to specify formatting like bold, italics, etc, or document structure like paragraphs, headings, sections and so on, but there are a variety of conventions commonly used. By default calibre attempts automatic detection of the correct formatting and markup based on those conventions.

<sup>&</sup>lt;sup>29</sup> https://calibre-ebook.com/downloads/demos/demo.docx

TXT input supports a number of options to differentiate how paragraphs are detected.

#### Paragraph style: Auto

Analyzes the text file and attempts to automatically determine how paragraphs are defined. This option will generally work fine, if you achieve undesirable results try one of the manual options.

#### Paragraph style: Block

Assumes one or more blank lines are a paragraph boundary:

```
This is the first.
This is the
second paragraph.
```

#### Paragraph style: Single

Assumes that every line is a paragraph:

```
This is the first.
This is the second.
This is the third.
```

#### Paragraph style: Print

Assumes that every paragraph starts with an indent (either a tab or 2+ spaces). Paragraphs end when the next line that starts with an indent is reached:

```
This is the
first.
This is the second.
This is the
third.
```

#### Paragraph style: Unformatted

Assumes that the document has no formatting, but does use hard line breaks. Punctuation and median line length are used to attempt to re-create paragraphs.

#### Formatting style: Auto

Attempts to detect the type of formatting markup being used. If no markup is used then heuristic formatting will be applied.

#### Formatting style: Heuristic

Analyzes the document for common chapter headings, scene breaks, and italicized words and applies the appropriate HTML markup during conversion.

#### Formatting style: Markdown

calibre also supports running TXT input though a transformation preprocessor known as Markdown. Markdown allows for basic formatting to be added to TXT documents, such as bold, italics, section headings, tables, lists, a Table of Contents, etc. Marking chapter headings with a leading # and setting the chapter XPath detection expression to "//h:h1" is the easiest way to have a proper table of contents generated from a TXT document. You can learn more about the Markdown syntax at daringfireball<sup>30</sup>.

#### Formatting style: None

Applies no special formatting to the text, the document is converted to HTML with no other changes.

<sup>30</sup> https://daringfireball.net/projects/markdown/syntax

# 4.11.3 Convert PDF documents

PDF documents are one of the worst formats to convert from. They are a fixed page size and text placement format. Meaning, it is very difficult to determine where one paragraph ends and another begins. calibre will try to unwrap paragraphs using a configurable, *Line un-wrapping factor*. This is a scale used to determine the length at which a line should be unwrapped. Valid values are a decimal between 0 and 1. The default is 0.45, just under the median line length. Lower this value to include more text in the unwrapping. Increase to include less. You can adjust this value in the conversion settings under *PDF Input*.

Also, they often have headers and footers as part of the document that will become included with the text. Use the *Search and replace* panel to remove headers and footers to mitigate this issue. If the headers and footers are not removed from the text it can throw off the paragraph unwrapping. To learn how to use the header and footer removal options, read *All about using regular expressions in calibre* (page 212).

Some limitations of PDF input are:

- Complex, multi-column, and image based documents are not supported.
- Extraction of vector images and tables from within the document is also not supported.
- Some PDFs use special glyphs to represent ll or ff or fi, etc. Conversion of these may or may not work depending on just how they are represented internally in the PDF.
- Links and Tables of Contents are not supported
- PDFs that use embedded non-Unicode fonts to represent non-English characters will result in garbled output for those characters
- Some PDFs are made up of photographs of the page with OCRed text behind them. In such cases calibre uses the OCRed text, which can be very different from what you see when you view the PDF file
- PDFs that are used to display complex text, like right to left languages and math typesetting will not convert correctly

To re-iterate **PDF** is a really, really bad format to use as input. If you absolutely must use PDF, then be prepared for an output ranging anywhere from decent to unusable, depending on the input PDF.

# 4.11.4 Comic book collections

A comic book collection is a .cbc file. A .cbc file is a ZIP file that contains other CBZ/CBR files. In addition the .cbc file must contain a simple text file called comics.txt, encoded in UTF-8. The comics.txt file must contain a list of the comics files inside the .cbc file, in the form filename:title, as shown below:

```
one.cbz:Chapter One
two.cbz:Chapter Two
three.cbz:Chapter Three
```

#### The .cbc file will then contain:

```
comics.txt
one.cbz
two.cbz
three.cbz
```

calibre will automatically convert this .cbc file into a e-book with a Table of Contents pointing to each entry in comics.txt.

# 4.11.5 EPUB advanced formatting demo

Various advanced formatting for EPUB files is demonstrated in this demo file<sup>31</sup>. The file was created from hand coded HTML using calibre and is meant to be used as a template for your own EPUB creation efforts.

The source HTML it was created from is available demo.zip<sup>32</sup>. The settings used to create the EPUB from the ZIP file are:

```
ebook-convert demo.zip .epub -vv --authors "Kovid Goyal" --language en --level1-toc '/
→/*[@class="title"]' --disable-font-rescaling --page-breaks-before / --no-default-
→epub-cover
```

Note that because this file explores the potential of EPUB, most of the advanced formatting is not going to work on readers less capable than calibre's built-in EPUB viewer.

# 4.11.6 Convert ODT documents

calibre can directly convert ODT (OpenDocument Text) files. You should use styles to format your document and minimize the use of direct formatting. When inserting images into your document you need to anchor them to the paragraph, images anchored to a page will all end up in the front of the conversion.

To enable automatic detection of chapters, you need to mark them with the built-in styles called *Heading 1*, *Heading 2*, ..., *Heading 6* (*Heading 1* equates to the HTML tag <h1>, *Heading 2* to <h2>, etc). When you convert in calibre you can enter which style you used into the *Detect chapters at* box. Example:

- If you mark Chapters with style Heading 2, you have to set the 'Detect chapters at' box to //h:h2
- For a nested TOC with Sections marked with *Heading 2* and the Chapters marked with *Heading 3* you need to enter //h:h2|//h:h3. On the Convert TOC page set the *Level 1 TOC* box to //h:h2 and the *Level 2 TOC* box to //h:h3.

Well-known document properties (Title, Keywords, Description, Creator) are recognized and calibre will use the first image (not to small, and with good aspect-ratio) as the cover image.

There is also an advanced property conversion mode, which is activated by setting the custom property <code>opf.metadata</code> ('Yes or No' type) to Yes in your ODT document (File->Properties->Custom Properties). If this property is detected by calibre, the following custom properties are recognized (<code>opf.authors</code> overrides document creator):

opf.titlesort
opf.authors
opf.authorsort
opf.publisher
opf.pubdate
opf.isbn
opf.language
opf.series
opf.series

In addition to this, you can specify the picture to use as the cover by naming it opf.cover (right click, Picture->Options->Name) in the ODT. If no picture with this name is found, the 'smart' method is used. As the cover detection might result in double covers in certain output formats, the process will remove the paragraph (only if the only content is the cover!) from the document. But this works only with the named picture!

To disable cover detection you can set the custom property opf.nocover ('Yes or No' type) to Yes in advanced mode.

<sup>&</sup>lt;sup>31</sup> https://calibre-ebook.com/downloads/demos/demo.epub

<sup>&</sup>lt;sup>32</sup> https://calibre-ebook.com/downloads/demos/demo.zip

# 4.11.7 Converting to PDF

The first, most important, setting to decide on when converting to PDF is the page size. By default, calibre uses a page size of "U.S. Letter". You can change this to another standard page size or a completely custom size in the *PDF Output* section of the conversion dialog. If you are generating a PDF to be used on a specific device, you can turn on the option to use the page size from the *output profile* instead. So if your output profile is set to Kindle, calibre will create a PDF with page size suitable for viewing on the small Kindle screen.

#### **Headers and Footers**

You can insert arbitrary headers and footers on each page of the PDF by specifying header and footer templates. Templates are just snippets of HTML code that get rendered in the header and footer locations. For example, to display page numbers centered at the bottom of every page, in green, use the following footer template:

<footer><div style="margin: auto; color: green">\_PAGENUM\_</div></footer>

calibre will automatically replace \_PAGENUM\_ with the current page number. You can even put different content on even and odd pages, for example the following header template will show the title on odd pages and the author on even pages:

calibre will automatically replace \_TITLE\_ and \_AUTHOR\_ with the title and author of the document being converted. Setting justify-content to flex-end will cause the text to be right aligned.

You can also display text at the left and right edges and change the font size, as demonstrated with this header template:

This will display the title at the left and the author at the right, in a font size smaller than the main text.

You can also use the current section in templates, as shown below:

<header><div>\_SECTION\_</div></header>

\_SECTION\_ is replaced by whatever the name of the current section is. These names are taken from the metadata Table of Contents in the document (the PDF Outline). If the document has no table of contents then it will be replaced by empty text. If a single PDF page has multiple sections, the first section on the page will be used. Similarly, there is a variable named \_TOP\_LEVEL\_SECTION\_ that can be used to get the name of the current top-level section.

You can even use JavaScript inside the header and footer templates, for example, the following template will cause page numbers to start at 4 instead of 1:

In addition there are some more variables you can use in the headers and footers, documented below:

• \_TOTAL\_PAGES\_ - total number of pages in the PDF file, useful for implementing a progress counter, for example.

- \_TOP\_LEVEL\_SECTION\_PAGES\_ total number of pages in the current top level section
- \_TOP\_LEVEL\_SECTION\_PAGENUM\_ the page number of the current page within the current top level section
- \_WIDTH\_PIXELS\_ the width of the header/footer area in pixels
- \_HEIGHT\_PIXELS\_ the height of the header/footer area in pixels

#### Note

When adding headers and footers make sure you set the page top and bottom margins to large enough values, under the *PDF Output* section of the conversion dialog.

#### **Printable Table of Contents**

You can also insert a printable Table of Contents at the end of the PDF that lists the page numbers for every section. This is very useful if you intend to print out the PDF to paper. If you wish to use the PDF on an electronic device, then the PDF Outline provides this functionality and is generated by default.

You can customize the look of the generated Table of contents by using the Extra CSS conversion setting under the Look & feel part of the conversion dialog. The default CSS used is listed below, simply copy it and make whatever changes you like.

```
.calibre-pdf-toc table { width: 100%% }
.calibre-pdf-toc table tr td:last-of-type { text-align: right }
.calibre-pdf-toc .level-0 {
   font-size: larger;
}
.calibre-pdf-toc .level-1 td:first-of-type { padding-left: 1.4em }
.calibre-pdf-toc .level-2 td:first-of-type { padding-left: 2.8em }
```

#### Custom page margins for individual HTML files

If you are converting an EPUB or AZW3 file with multiple individual HTML files inside it and you want to change the page margins for a particular HTML file you can add the following style block to the HTML file using the calibre E-book editor:

```
<style>
@page {
    margin-left: 10pt;
    margin-right: 10pt;
    margin-top: 10pt;
    margin-bottom: 10pt;
}
</style>
```

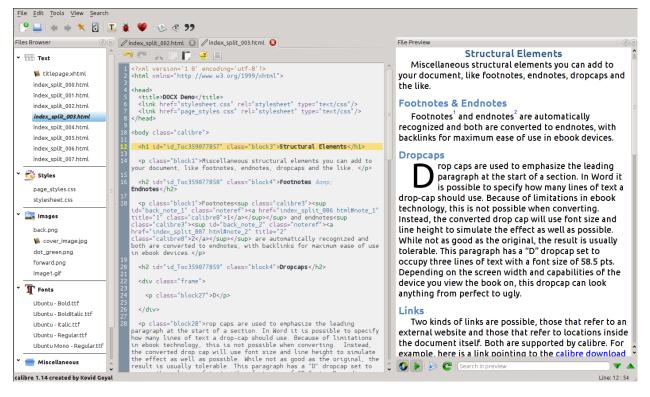
Then, in the PDF output section of the conversion dialog, turn on the option to *Use page margins from the document being converted*. Now all pages generated from this HTML file will have 10pt margins.

### CHAPTER

# **EDITING E-BOOKS**

calibre has an integrated e-book editor that can be used to edit books in the EPUB, KEPUB (Kobo) and AZW3 (Kindle) formats. The editor shows you the HTML and CSS that is used internally inside the book files, with a live preview that updates as you make changes. It also contains various automated tools to perform common cleanup and fixing tasks.

You can use this editor by right clicking on any book in calibre and selecting Edit book.



#### Contents

- Basic workflow (page 81)
- The File browser (page 83)
  - Renaming files (page 84)
  - Merging files (page 84)
  - Changing text file order (page 84)

- Marking the cover (page 85)
- Deleting files (page 85)
- Exporting files (page 85)
- Adding new images/fonts/etc. or creating new blank files (page 85)
- Replacing files (page 85)
- Linking stylesheets to HTML files efficiently (page 85)
- Search & replace (page 85)
  - Saved searches (page 86)
  - Function mode (page 86)
  - Search ignoring HTML tags (page 86)
- Automated tools (page 86)
  - Editing the Table of Contents (page 86)
  - Checking the book (page 88)
  - Adding a cover (page 89)
  - *Embedding referenced fonts* (page 89)
  - *Subsetting embedded fonts* (page 89)
  - Smartening punctuation (page 89)
  - Transforming CSS properties (page 89)
  - Removing unused CSS rules (page 89)
  - Fixing HTML (page 90)
  - Beautifying files (page 90)
  - Inserting an inline Table of Contents (page 90)
  - Setting Semantics (page 90)
  - Filtering style information (page 90)
  - Upgrading the book's internals (page 91)
- Checkpoints (page 91)
- The Live preview panel (page 93)
  - *Splitting HTML files* (page 94)
- The Live CSS panel (page 95)
- Miscellaneous tools (page 96)
  - The Table of Contents view (page 96)
  - Checking the spelling of words in the book (page 96)
  - Inserting special characters (page 97)
  - The code inspector view (page 98)
  - Checking external links (page 98)

- *Downloading external resources* (page 98)
- Arranging files into folders by type (page 98)
- Importing files in other e-book formats as EPUB (page 98)
- The Reports tool (page 108)
- Special features in the code editor (page 109)
  - Syntax highlighting (page 109)
  - Context sensitive help (page 109)
  - Auto-complete (page 110)
  - Snippets (page 110)

# 5.1 Basic workflow

#### Note

A video tour of the calibre E-book editor is available here<sup>33</sup>.

When you first open a book with the Edit book tool, you will be presented with a list of files on the left. These are the individual HTML files, stylesheets, images, etc. that make up the content of the book. Simply double click on a file to start editing it. Note that if you want to do anything more sophisticated than making a few small tweaks, you will need to know HTML Tutorial<sup>34</sup> and CSS Tutorial<sup>35</sup>.

As you make changes to the HTML or CSS in the editor, the changes will be previewed, live, in the preview panel to the right. When you are happy with how the changes you have made look, click the *Save* button or use  $File \rightarrow Save$  to save your changes into the e-book.

One useful feature is *Checkpoints*. Before you embark on some ambitious set of edits, you can create a checkpoint. The checkpoint will preserve the current state of your book, then if in the future you decide you don't like the changes you have made to you can go back to the state when you created the checkpoint. To create a checkpoint, use *Edit*  $\rightarrow$  *Create checkpoint*. Checkpoints will also be automatically created for you whenever you run any automated tool like global search and replace. The checkpointing functionality is in addition to the normal undo/redo mechanism when editing individual files. Checkpoints are needed for when changes are spread over multiple files in the book.

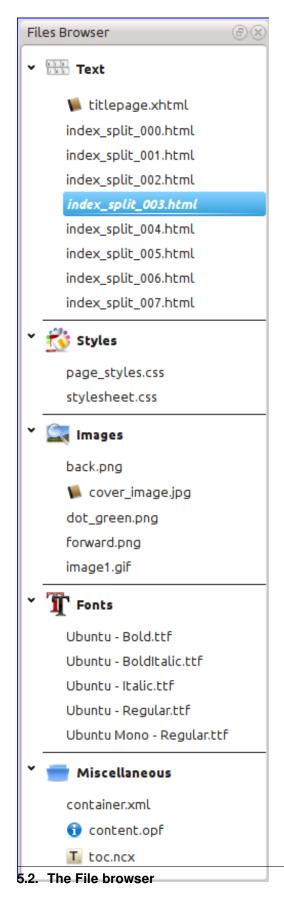
That is the basic work flow for editing books – Open a file, make changes, preview and save. The rest of this manual will discuss the various tools and features present to allow you to perform specific tasks efficiently.

<sup>33</sup> https://calibre-ebook.com/demo#tutorials

<sup>&</sup>lt;sup>34</sup> https://www.w3schools.com/Html/default.asp

<sup>&</sup>lt;sup>35</sup> https://www.w3schools.com/css/default.asp

# 5.2 The File browser



The *File browser* gives you an overview of the various files inside the book you are editing. The files are arranged by category, with text (HTML) files at the top, followed by stylesheet (CSS) files, images and so on. Simply double click on a file to start editing it. Editing is supported for HTML, CSS and image files. The order of text files is the same order that they would be displayed in, if you were reading the book. All other files are arranged alphabetically.

By hovering your mouse over an entry, you can see its size, and also, at the bottom of the screen, the full path to the file inside the book. Note that files inside e-books are compressed, so the size of the final book is not the sum of the individual file sizes.

Many files have special meaning, in the book. These will typically have an icon next to their names, indicating the special meaning. For example, in the picture to the left, you can see that the files *cover\_image.jpg* and *titlepage.xhtml* have the icon of a cover next to them, this indicates they are the book cover image and titlepage. Similarly, the *content.opf* file has a metadata icon next to it, indicating the book metadata is present in it and the *toc.ncx* file has a T icon next to it, indicating it is the Table of Contents.

You can perform many actions on individual files, by right clicking them.

# 5.2.1 Renaming files

You can rename an individual file by right clicking it and selecting *Rename*. Renaming a file automatically updates all links and references to it throughout the book. So all you have to do is provide the new name, calibre will take care of the rest.

You can also bulk rename many files at once. This is useful if you want the files to have some simple name pattern. For example you might want to rename all the HTML files to have names Chapter-1.html, Chapter-2.html and so on. Select the files you want bulk renamed by holding down the Shift or Ctrl key and clicking the files. Then right click and select *Bulk rename*. Enter a prefix and what number you would like the automatic numbering to start at, click OK and you are done. The bulk rename dialog also lets you rename files by the order they appear in the book instead of the order you selected them in, useful, for instance to rename all images by the order they appear.

Finally, you can bulk change the file extension for all selected files. Select multiple files, as above, and right click and choose *Change the file extension for the selected files*.

# 5.2.2 Merging files

Sometimes, you may want to merge two HTML files or two CSS files together. It can sometimes be useful to have everything in a single file. Be wary, though, putting a lot of content into a single file will cause performance problems when viewing the book in a typical e-book reader.

To merge multiple files together, select them by holding the Ctrl key and clicking on them (make sure you only select files of one type, either all HTML files or all CSS files and so on). Then right click and select merge. That's all, calibre will merge the files, automatically taking care of migrating all links and references to the merged files. Note that merging files can sometimes cause text styling to change, since the individual files could have used different stylesheets.

You can also select text files and then drag and drop the text files onto another text file to merge the dropped text files into the target text file.

# 5.2.3 Changing text file order

You can re-arrange the order in which text (HTML) files are opened when reading the book by simply dragging and dropping them in the *File browser* or clicking on the file to move and then pressing the Ctrl+Shift modifiers with the Up, Down, Home or End keys. For the technically inclined, this is called re-ordering the book spine.

Note that you have to drop the items *between* other items, not on top of them, this can be a little fiddly until you get used to it. Dropping on top of another file will cause the files to be merged.

# 5.2.4 Marking the cover

E-books typically have a cover image. This image is indicated in the *File browser* by the icon of a brown book next to the image name. If you want to designate some other image as the cover, you can do so by right clicking on the file and choosing *Mark as cover*.

In addition, EPUB files has the concept of a *titlepage*. A title page is a HTML file that acts as the title page/cover for the book. You can mark an HTML file as the titlepage when editing EPUBs by right-clicking. Be careful that the file you mark contains only the cover information. If it contains other content, such as the first chapter, then that content will be lost if the user ever converts the EPUB file in calibre to another format. This is because when converting, calibre assumes that the marked title page contains only the cover and no other content.

# 5.2.5 Deleting files

You can delete files by either right clicking on them or by selecting them and pressing the Delete key. Deleting a file removes all references to the file from the OPF file, saving you that chore. However, references in other places are not removed, you can use the Check Book tool to easily find and remove/replace them.

# 5.2.6 Exporting files

You can export a file from inside the book to somewhere else on your computer. This is useful if you want to work on the file in isolation, with specialised tools. To do this, simply right click on the file and choose *Export*.

Once you are done working on the exported file, you can re-import it into the book, by right clicking on the file again and choosing *Replace with file...* which will allow you to replace the file in the book with the previously exported file.

You can also copy files between multiple editor instances. Select the files you want to copy in the *File browser*, then right click and choose, *Copy selected files to another editor instance*. Then, in the other editor instance, right click in the *File browser* and choose *Paste file from other editor instance*.

# 5.2.7 Adding new images/fonts/etc. or creating new blank files

You can add a new image, font, stylesheet, etc. from your computer into the book by clicking  $File \rightarrow New$  file. This lets you either import a file by clicking the *Import resource file* button or create a new blank HTML file or stylesheet by simply entering the file name into the box for the new file.

You can also import multiple files into the book at once using File->Import files into book.

# 5.2.8 Replacing files

You can easily replace existing files in the book, by right clicking on the file and choosing replace. This will automatically update all links and references, in case the replacement file has a different name than the file being replaced.

# 5.2.9 Linking stylesheets to HTML files efficiently

As a convenience, you can select multiple HTML files in the File browser, right click and choose Link stylesheets to have calibre automatically insert the <link> tags for those stylesheets into all the selected HTML files.

# 5.3 Search & replace

Edit book has a very powerful search and replace interface that allows you to search and replace text in the current file, across all files and even in a marked region of the current file. You can search using a normal search or using regular expressions. To learn how to use regular expressions for advanced searching, see *All about using regular expressions in calibre* (page 212).

0	<u>F</u> ind:		Find	Replace and Find
	<u>R</u> eplace:		<u>R</u> eplace	Replace <u>a</u> ll
	<u>M</u> ode:	Regex • Current file • Down • Case sensitive	✓ Wrap □ [	<u>o</u> t all

Start the search and replace via the Search  $\rightarrow$  Find/replace menu entry (you must be editing an HTML or CSS file).

Type the text you want to find into the Find box and its replacement into the Replace box. You can the click the appropriate buttons to Find the next match, replace the current match and replace all matches.

Using the drop downs at the bottom of the box, you can have the search operate over the current file, all text files, all style files or all files. You can also choose the search mode to be a normal (string) search or a regular expression search.

You can count all the matches for a search expression via *Search*  $\rightarrow$  *Count all*. The count will run over whatever files/regions you have selected in the dropdown box.

You can also go to a specific line in the currently open editor via Search  $\rightarrow$  Go to line.

#### 1 Note

Remember, to harness the full power of search and replace, you will need to use regular expressions. See *All about using regular expressions in calibre* (page 212).

### 5.3.1 Saved searches

You can save frequently used search/replace expressions (including function mode expressions) and reuse them multiple times. To save a search simply right click in the Find box and select *Save current search*.

You can bring up the saved searches via *Search*  $\rightarrow$  *Saved searches*. This will present you with a list of search and replace expressions that you can apply. You can even select multiple entries in the list by holding down the Ctrl key while clicking so as to run multiple search and replace expressions in a single operation.

### 5.3.2 Function mode

Function mode allows you to write arbitrarily powerful Python functions that are run on every Find/replace. You can do pretty much any text manipulation you like in function mode. For more information, see *Function mode for Search & replace in the Editor* (page 98).

### 5.3.3 Search ignoring HTML tags

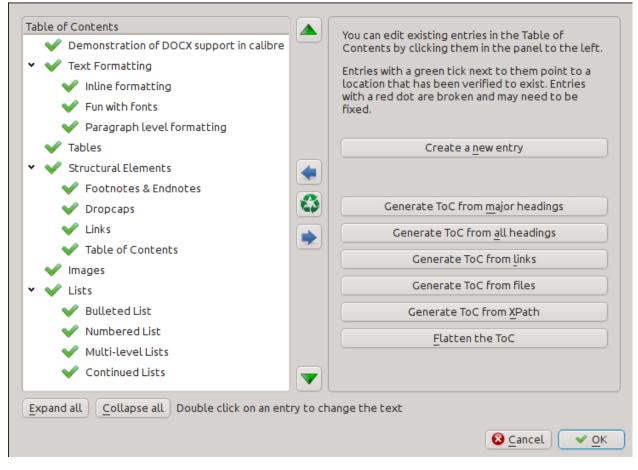
There is also a dedicated tool for searching for text, ignoring any HTML tags in between. For example, if the book has the HTML Empahisis on a <i>word</i>, you can search for on a word and it will be found even though there is an <i> tag in the middle. Use this tool via the *Search*  $\rightarrow$  *Search ignoring HTML markup* menu item.

# 5.4 Automated tools

Edit book has various tools to help with common tasks. These are accessed via the Tools menu.

### 5.4.1 Editing the Table of Contents

There is a dedicated tool to ease editing of the Table of Contents. Launch it with  $Tools \rightarrow Table of Contents \rightarrow Edit Table of Contents$ .



The Edit Table of Contents tool shows you the current Table of Contents (if any) on the left. Simply double click on any entry to change its text. You can also re-arrange entries by drag and drop or by using the buttons to the right.

For books that do not have a pre-existing Table of Contents, the tool gives you various options to auto-generate a Table of Contents from the text. You can generate from the headings in the document, from links, from individual files and so on.

You can edit individual entries by clicking on them and then clicking the *Change the location this entry points to* button. This will open up a mini-preview of the book, simply move the mouse cursor over the book view panel, and click where you want the entry to point to. A thick green line will show you the location. Click OK once you are happy with the location.

titlepage.xhtml index_split_000.html index_split_001.html index_split_002.html index_split_003.html	Footnotes & Endnotes Footnotes <sup>1</sup> and endnotes <sup>2</sup> are automatically recognized and both are	Here you can choose a destination fo the Table of Contents' entry to point to. First choose a file from the book in the left-most panel. The file will
ndex_split_004.html ndex_split_005.html ndex_split_006.html ndex_split_007.html	converted to endnotes, with backlinks for maximum ease of use in ebook devices. Dropcaps pop caps are used to emphasize the leading paragraph at the start of a section. In Word it is possible to specify how many lines of text a drop- cap should use. Because of limitations in ebook technology, this is not possible	open in the central panel. Then choose a location inside the file To do so, simply click on the place in the central panel that you want to use as the destination. As you move the mouse around the central panel, a thick green line appears, indicating the precise location that will be selected when you click. Name of the ToC entry:
	when converting. Instead, the converted drop cap will use font size and line height to simulate the effect as well as possible. While not as good as the original, the result is usually tolerable. This paragraph has a "D" dropcap set to occupy three lines of text with a font size of 58.5 pts. Depending on the screen width and capabilities of the device you view the Search for text Find next A Find prev	Currently selected destination: File: index_split_003.html Location: A <h2> tag inside the file [Approximately 7% from the top]</h2>

# 5.4.2 Checking the book

The *Check book* tool searches your book for problems that could prevent it working as intended on actual reader devices. Activate it via  $Tools \rightarrow Check \ book$ .

Check Book	æ 🔅
🔺 The file META-INF/calibre_bookmarks.txt is not listed in the manifest [META-INF/calibre]	Warning [2 / 2]
🔺 The file images/00001.jpg is not referenced [images/00001.jpg]	
	images/00001.jpg
	This file is included in the book but not referred to by any document in the spine. This means that the file will not be viewable on most ebook readers. You should probably remove this file from the book or add a link to it somewhere.
	Try to correct all fixable errors

Any problems found are reported in a nice, easy to use list. Clicking any entry in the list shows you some help about that error as well as giving you the option to auto-fix that error, if the error can be fixed automatically. You can also double click the error to open the location of the error in an editor, so you can fix it yourself.

Some of the checks performed are:

- Malformed HTML markup. Any HTML markup that does not parse as well-formed XML is reported. Correcting it
  will ensure that your markup works as intended in all contexts. calibre can also auto-fix these errors, but auto-fixing
  can sometimes have unexpected effects, so use with care. As always, a checkpoint is created before auto-fixing so
  you can easily revert all changes. Auto-fixing works by parsing the markup using the HTML5 algorithm, which is
  highly fault tolerant and then converting to well formed XML.
- Malformed or unknown CSS styles. Any CSS that is not valid or that has properties not defined in the CSS 2.1

standard (plus a few from CSS 3) are reported. CSS is checked in all stylesheets, inline style attributes and <style> tags in HTML files.

- Broken links. Links that point to files inside the book that are missing are reported.
- Unreferenced files. Files in the book that are not referenced by any other file or are not in the spine are reported.
- Various common problems in OPF files such as duplicate spine or manifest items, broken idrefs or meta cover tags, missing required sections and so on.
- Various compatibility checks for known problems that can cause the book to malfunction on reader devices.

### 5.4.3 Adding a cover

You can easily add a cover to the book via *Tools*  $\rightarrow$  *Add cover*. This allows you to either choose an existing image in the book as the cover or import a new image into the book and make it the cover. When editing EPUB files, the HTML wrapper for the cover is automatically generated. If an existing cover in the book is found, it is replaced. The tool also automatically takes care of correctly marking the cover files as covers in the OPF.

### 5.4.4 Embedding referenced fonts

Accessed via  $Tools \rightarrow Embed$  reference fonts, this tool finds all fonts referenced in the book and if they are not already embedded, searches your computer for them and embeds them into the book, if found. Please make sure that you have the necessary copyrights for embedding commercially licensed fonts, before doing this.

### 5.4.5 Subsetting embedded fonts

Accessed via  $Tools \rightarrow Subset$  embedded fonts, this tool reduces all the fonts in the book to only contain glyphs for the text actually present in the book. This commonly reduces the size of the font files by ~ 50%. However, be aware that once the fonts are subset, if you add new text whose characters are not previously present in the subset font, the font will not work for the new text. So do this only as the last step in your workflow.

### 5.4.6 Smartening punctuation

Convert plain text dashes, ellipsis, quotes, multiple hyphens, etc. into their typographically correct equivalents. Note that the algorithm can sometimes generate incorrect results, especially when single quotes at the start of contractions are involved. Accessed via *Tools*  $\rightarrow$  *Smarten punctuation*.

### 5.4.7 Transforming CSS properties

Create rules to transform the styling of the book. For example, create a rule to convert all red text to green or to double the font size of all text in the book or make text of a certain font family italic, etc.

Creating the rules is simple, the rules follow a natural language format, that looks like:

- If the property color is red change it to green
- If the property *font-size* is *any value multiply* the value by 2

Accessed via  $Tools \rightarrow Transform \ styles$ .

### 5.4.8 Removing unused CSS rules

Remove all unused CSS rules from stylesheets and <style> tags. Some books created from production templates can have a large number of extra CSS rules that don't match any actual content. These extra rules can slow down readers that need to process them all. Accessed via *Tools*  $\rightarrow$  *Remove unused CSS*.

# 5.4.9 Fixing HTML

This tool simply converts HTML that cannot be parsed as XML into well-formed XML. It is very common in e-books to have non-well-formed XML, so this tool simply automates the process of fixing such HTML. The tool works by parsing the HTML using the HTML5 algorithm (the algorithm used in all modern browsers) and then converting the result into XML. Be aware that auto-fixing can sometimes have counter-intuitive results. If you prefer, you can use the Check Book tool discussed above to find and manually correct problems in the HTML. Accessed via  $Tools \rightarrow Fix HTML$ .

# 5.4.10 Beautifying files

This tool is used to auto-format all HTML and CSS files so that they "look pretty". The code is auto-indented so that it lines up nicely, blank lines are inserted where appropriate and so on. Note that beautifying also auto-fixes broken HTML/CSS. Therefore, if you don't want any auto-fixing to be performed, first use the Check Book tool to correct all problems and only then run beautify. Accessed via *Tools*  $\rightarrow$  *Beautify all files*.

### 1 Note

In HTML any text can have significant whitespace, via the CSS white-space directive. Therefore, beautification could potentially change the rendering of the HTML. To avoid this as far as possible, the beautify algorithm only beautifies block level tags that contain other block level tags. So, for example, text inside a  $\langle p \rangle$  tag will not have its whitespace changed. But a  $\langle body \rangle$  tag that contains only other  $\langle p \rangle$  and  $\langle div \rangle$  tags will be beautified. This can sometimes mean that a particular file will not be affected by beautify as it has no suitable block level tags. In such cases you can try different beautification tools, that are less careful, for example: HTML Tidy<sup>36</sup>.

# 5.4.11 Inserting an inline Table of Contents

Normally in e-books, the Table of Contents is separate from the main text and is typically accessed via a special Table of Contents button/menu in the e-book reading device. You can also have calibre automatically generate an *inline* Table of Contents that becomes part of the text of the book. It is generated based on the currently defined Table of Contents.

If you use this tool multiple times, each invocation will cause the previously created inline Table of Contents to be replaced. The tool can be accessed via  $Tools \rightarrow Table of Contents \rightarrow Insert inline Table of Contents$ .

# 5.4.12 Setting Semantics

This tool is used to set *semantics* in EPUB files. Semantics are simply, links in the OPF file that identify certain locations in the book as having special meaning. You can use them to identify the foreword, dedication, cover, table of contents, etc. Simply choose the type of semantic information you want to specify and then select the location in the book the link should point to. This tool can be accessed via *Tools*  $\rightarrow$  *Set semantics*.

# 5.4.13 Filtering style information

This tool can be used to easily remove specified CSS style properties from the entire book. You can tell it what properties you want removed, for example, color, background-color, line-height and it will remove them from everywhere they occur — stylesheets, <style> tags and inline style attributes. After removing the style information, a summary of all the changes made is displayed so you can see exactly what was changed. The tool can be accessed via *Tools*  $\rightarrow$  *Filter style information*.

<sup>36</sup> https://infohound.net/tidy/

# 5.4.14 Upgrading the book's internals

This tool can be used to upgrade the book's internals, if possible. For instance it will upgrade EPUB 2 books to EPUB 3 books. The tool can be accessed via *Upgrade book internals*.

# 5.5 Checkpoints

*Checkpoints* are a way to mark the current state of the book as "special". You can then go on to do whatever changes you want to the book and if you don't like the results, return to the checkpointed state. Checkpoints are automatically created every time you run any of the automated tools described in the previous section.

You can create a checkpoint via  $Edit \rightarrow Create checkpoint$ . And go back to a previous checkpoint with  $Edit \rightarrow Revert to \dots$ 

The check pointing functionality is in addition to the normal Undo/redo mechanism when editing individual files. Checkpoints are needed for when changes are spread over multiple files in the book or when you wish to be able to revert a large group of related changes as a whole.

You can see a list of available checkpoints via  $View \rightarrow Checkpoints$ . You can compare the current state of the book to a specified checkpoint using the *Comparing e-books* (page 119) tool – by selecting the checkpoint of interest and clicking the *Compare* button. The *Revert to* button restores the book to the selected checkpoint, undoing all changes since that checkpoint was created.

(F) (X

# 5.6 The Live preview panel

# File Preview

# Inline formatting

Here, we demonstrate various types of inline text formatting and the use of embedded fonts.

Here is some **bold**, *italic*, *bold-italic*, <u>underlined</u> and <del>struck out</del> text. Then, we have a super<sup>script</sup> and a sub<sub>script</sub>. Now we see some red, green and blue text. Some text with a <u>yellow highlight</u>. Some text in a box. Some text in <u>inverse video</u>.

A paragraph with styled text: *subtle emphasis* followed by **strong text** and *intense emphasis*. This paragraph uses document wide styles for styling rather than inline text properties as demonstrated in the previous paragraph calibre can handle both with equal ease.

# Fun with fonts

This document has embedded the Ubuntu font family. The body text is in the Ubuntu typeface, here is some text in the Ubuntu Mono typeface, notice how every letter has the same width, even i and m. Every embedded font will automatically be embedded in the output ebook during conversion.

The *File preview* gives you an overview of the various files inside The live preview panel shows you the changes you are making live (with a second or two of delay). As you edit HTML or CSS files, the preview panel is updated automatically to reflect your changes. As you move the cursor around in the editor, the preview panel will track its location, showing you the corresponding location in the book. Clicking in the preview panel, will cause the cursor in the editor to be positioned over the element you clicked. If you click a link pointing to another file in the book, that file will be opened in the edit and the preview panel, automatically.

You can turn off the automatic syncing of position and live preview of changes – by buttons under the preview panel. The live update of the preview panel only happens when you are not actively typing in the editor, so as not to be distracting or slow you down, waiting for the preview to render.

The preview panel shows you how the text will look when viewed. However, the preview panel is not a substitute for actually testing your book an actual reader device. It is both more, and less capable than an actual reader. It will tolerate errors and sloppy markup much better than most reader devices. It will also not show you page margins, page breaks and embedded fonts that use font name aliasing. Use the preview panel while you are working on the book, but once you are done, review it in an actual reader device or software emulator.

#### Note

The preview panel does not support embedded fonts if the name of the font inside the font file does not match the name in the CSS @font-face rule. You can use the Check Book tool to quickly find and fix any such problem fonts.

# 5.6.1 Splitting HTML files

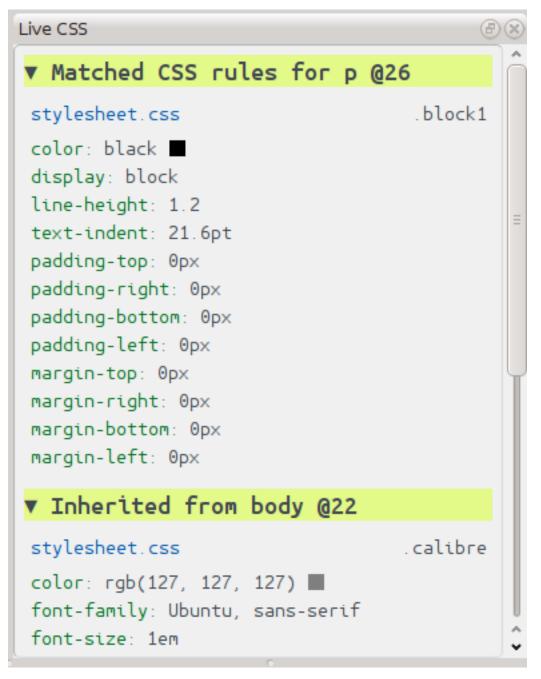
One, perhaps non-obvious, use of the preview panel is to split long HTML files. While viewing the file you want to split,

click the *Split mode* button under the preview panel . Then simply move your mouse to the place where you want to split the file and click. A thick green line will show you exactly where the split will happen as you move your mouse. Once you have found the location you want, simply click and the split will be performed.

Splitting the file will automatically update all links and references that pointed into the bottom half of the file and will open the newly split file in an editor.

You can also split a single HTML file at multiple locations automatically, by right clicking inside the file in the editor and choosing *Split at multiple locations*. This will allow you to easily split a large file at all heading tags or all tags having a certain class and so on.

# 5.7 The Live CSS panel



The *Live CSS* panel shows you all the style rules that apply to the tag you are currently editing. The name of tag, along with its line number in the editor are displayed, followed by a list of matching style rules.

It is a great way to quickly see which style rules apply to any tag. The view also has clickable links (in blue), which take you directly to the location where the style was defined, in case you wish to make any changes to the style rules. Style rules that apply directly to the tag, as well as rules that are inherited from parent tags are shown.

The panel also shows you what the finally calculated styles for the tag are. Properties in the list that are superseded by higher priority rules are shown with a line through them.

You can enable the Live CSS panel via  $View \rightarrow Live$  CSS.

# 5.8 Miscellaneous tools

There are a few more tools that can be useful while you edit the book.

# 5.8.1 The Table of Contents view

The Table of Contents view shows you the current table of contents in the book. Double clicking on any entry opens the place that entry points to in an editor. You can right click to edit the Table of Contents, refresh the view or expand/collapse all items. Access this view via *View*  $\rightarrow$  *Table of Contents*.

# 5.8.2 Checking the spelling of words in the book

Word	<ul> <li>Count</li> </ul>	Language	Ignore inline
DOCX	16	English	
dropcap	2	English	Add to <u>d</u> ictionary:
Dropcaps	3	English	Default
dropcaps	2	English	C characterization
ebook	11	English	Show <u>n</u> ext occurrence
ebook.com	1	English	
ebooks	3	English	
EPUB	2	English	
etc	1	English	
Goyal	2	English	<u>Change selected word to:</u>
дгау	1	English	online
hyperlinks	1	English	online incline
i	1	English	in-line in line mainline
i.e	1	English	inlier unlined
inline	2	English	newline
Inline	5	English	on-line
Kovid	2	English	

You can run a spelling checker via  $Tools \rightarrow Check spelling$ .

Words are shown with the number of times they occur in the book and the language the word belongs to. Language

information is taken from the books metadata and from lang attributes in the HTML files. This allows the spell checker to work well even with books that contain text in multiple languages. For example, in the following HTML extract, the word color will be checked using American English and the word colour using British English:

<div lang="en\_US">color <span lang="en\_GB">colour</span></div>

#### \rm 1 Note

You can double click a word to highlight the next occurrence of that word in the editor. This is useful if you wish to manually edit the word, or see what context it is in.

To change a word, simply double click one of the suggested alternative spellings on the right, or type in your own corrected spelling and click the *Change selected word to* button. This will replace all occurrences of the word in the book. You can also right click on a word in the main word list to change the word conveniently from the right click menu.

You can have the spelling checker ignore a word for the current session by clicking the *Ignore* button. You can also add a word to the user dictionary by clicking the *Add to dictionary* button. The spelling checker supports multiple user dictionaries, so you can select the dictionary you want the word added to.

You can also have the spelling checker display all the words in your book, not just the incorrectly spelled ones. This is useful to see what words are most common in your book and to run a simple search and replace on individual words.

#### 1 Note

If you make any changes to the book by editing files while the spell check tool is open, you should click the *Refresh* button in the Spell check tool. If you do not do this and continue to use the Spell check tool, you could lose the changes you have made in the editor.

#### 1 Note

To exclude an individual file from being spell checked when running the spell check tool, you can use the *Exclude files* button or add the following comment just under the opening tag in the file:

<!-- calibre-no-spell-check -->

#### Adding new dictionaries

The spelling checker comes with builtin dictionaries for the English and Spanish languages. You can install your own dictionaries via *Preferences*  $\rightarrow$  *Editor*  $\rightarrow$  *Manage spelling dictionaries*. The spell checker can use dictionaries from the LibreOffice program (in the .oxt format). You can download these dictionaries from The LibreOffice Extensions repository<sup>37</sup>.

### 5.8.3 Inserting special characters

You can insert characters that are difficult to type by using the  $Edit \rightarrow Insert special character$  tool. This shows you all Unicode characters, simply click on the character you want to type. If you hold Ctrl while clicking, the window will close itself after inserting the selected character. This tool can be used to insert special characters into the main text or into any other area of the user interface, such as the Search and replace tool.

Because there are a lot of characters, you can define your own *Favorite* characters, that will be shown first. Simply right click on a character to mark it as favorite. You can also right click on a character in favorites to remove it from favorites.

<sup>&</sup>lt;sup>37</sup> https://extensions.libreoffice.org/?Tags%5B%5D=50

Finally, you can re-arrange the order of characters in favorites by clicking the *Re-arrange favorites* button and then drag and dropping the characters in favorites around.

You can also directly type in special characters using the keyboard. To do this, you type the Unicode code for the character (in hexadecimal) and then press the Alt+x key which will convert the previously typed code into the corresponding character. For example, to type  $\ddot{y}$  you would type ff and then Alt+x. To type a non-breaking space you would use a0 and then Alt+x, to type the horizontal ellipsis you would use 2026 and Alt+x and so on.

Finally, you can type in special characters by using HTML named entities. For example, typing will be replaced by a non breaking space when you type the semi-colon. The replacement happens only when typing the semi-colon.

# 5.8.4 The code inspector view

This view shows you the HTML coding and CSS that applies to the current element of interest. You open it by right clicking a location in the preview panel and choosing *Inspect*. It allows you to see the HTML coding for that element and more importantly, the CSS styles that apply to it. You can even dynamically edit the styles and see what effect your changes have instantly. Note that editing the styles does not actually make changes to the book contents, it only allows for quick experimentation. The ability to live edit inside the Inspector is under development.

# 5.8.5 Checking external links

You can use this tool to check all links in your book that point to external websites. The tool will try to visit every externally linked website, and if the visit fails, it will report all broken links in a convenient format for you to fix.

### 5.8.6 Downloading external resources

You can use this tool to automatically download any images/stylesheets/etc. in the book that are not bundled with the book (i.e. they have URLs pointing to a location on the internet). The tool will find all such resources and automatically download them, add them to the book and replace all references to them to use the downloaded files.

# 5.8.7 Arranging files into folders by type

Often when editing EPUB files that you get from somewhere, you will find that the files inside the EPUB are arranged haphazardly, in different sub-folders. This tool allows you to automatically move all files into sub-folders based on their types. Access it via *Tools*  $\rightarrow$  *Arrange into folders*. Note that this tool only changes how the files are arranged inside the EPUB, it does not change how they are displayed in the File browser.

# 5.8.8 Importing files in other e-book formats as EPUB

The editor includes the ability to import files in some other e-book formats directly as a new EPUB, without going through a full conversion. This is particularly useful to directly create EPUB files from your own hand-edited HTML files. You can do this via *File*  $\rightarrow$  *Import an HTML or DOCX file as a new book*.

### Function mode for Search & replace in the Editor

The *Search & replace* tool in the editor support a *function mode*. In this mode, you can combine regular expressions (see *All about using regular expressions in calibre* (page 212)) with arbitrarily powerful Python functions to do all sorts of advanced text processing.

In the standard *regexp* mode for search and replace, you specify both a regular expression to search for as well as a template that is used to replace all found matches. In function mode, instead of using a fixed template, you specify an arbitrary function, in the Python programming language<sup>38</sup>. This allows you to do lots of things that are not possible with simple templates.

<sup>&</sup>lt;sup>38</sup> https://docs.python.org

Techniques for using function mode and the syntax will be described by means of examples, showing you how to create functions to perform progressively more complex tasks.

<u>F</u> ind:	▼ <u>Find</u> Replace and Find
F <u>u</u> nction:	Create/ <u>e</u> dit Remo <u>v</u> e Replace Replace <u>a</u> ll
<u>M</u> ode:	Regex-Function ▼ Current file ▼ Down ▼ □ Case sensitive ✔ Wrap □ Dot all

#### Automatically fixing the case of headings in the document

Here, we will leverage one of the builtin functions in the editor to automatically change the case of all text inside heading tags to title case:

```
Find expression: <([Hh][1-6])[^>]*>.+?</\1>
```

For the function, simply choose the *Title-case text (ignore tags)* builtin function. The will change titles that look like: <h1>some titLE</h1> to <h1>Some Title</h1>. It will work even if there are other HTML tags inside the heading tags.

#### Your first custom function - smartening hyphens

The real power of function mode comes from being able to create your own functions to process text in arbitrary ways. The Smarten Punctuation tool in the editor leaves individual hyphens alone, so you can use the this function to replace them with em-dashes.

To create a new function, simply click the *Create/edit* button to create a new function and copy the Python code from below.

Every *Search & replace* custom function must have a unique name and consist of a Python function named replace, that accepts all the arguments shown above. For the moment, we won't worry about all the different arguments to replace() function. Just focus on the match argument. It represents a match when running a search and replace. Its full documentation in available here<sup>39</sup>. match.group() simply returns all the matched text and all we do is replace hyphens in that text with em-dashes, first replacing double hyphens and then single hyphens.

Use this function with the find regular expression:

 $> [ ^ < > ] + <$ 

And it will replace all hyphens with em-dashes, but only in actual text and not inside HTML tag definitions.

#### The power of function mode - using a spelling dictionary to fix mis-hyphenated words

Often, e-books created from scans of printed books contain mis-hyphenated words – words that were split at the end of the line on the printed page. We will write a simple function to automatically find and fix such words.

```
import regex
from calibre import replace_entities
from calibre import prepare_string_for_xml
```

(continues on next page)

<sup>&</sup>lt;sup>39</sup> https://docs.python.org/library/re.html#match-objects

(continued from previous page)

```
def replace(match, number, file_name, metadata, dictionaries, data, functions, *args,_

→**kwargs):

    def replace_word(wmatch):

        # Try to remove the hyphen and replace the words if the resulting

        # hyphen free word is recognized by the dictionary

        without_hyphen = wmatch.group(1) + wmatch.group(2)

        if dictionaries.recognized(without_hyphen):

            return without_hyphen

        return wmatch.group()

        # Search for words split by a hyphen

        text = replace_entities(match.group()[1:-1])  # Handle HTML entities like &

        corrected = regex.sub(r'(\w+)\s*-\s*(\w+)', replace_word, text, flags=regex.

        •VERSION1 | regex.UNICODE)

        return '>$s<' % prepare_string_for_xml(corrected)  # Put back required entities
    }
}
</pre>
```

Use this function with the same find expression as before, namely:

 $> [ ^ < > ] + <$ 

And it will magically fix all mis-hyphenated words in the text of the book. The main trick is to use one of the useful extra arguments to the replace function, dictionaries. This refers to the dictionaries the editor itself uses to spell check text in the book. What this function does is look for words separated by a hyphen, remove the hyphen and check if the dictionary recognizes the composite word, if it does, the original words are replaced by the hyphen free composite word.

Note that one limitation of this technique is it will only work for mono-lingual books, because, by default, dictionaries.recognized() uses the main language of the book.

#### Auto numbering sections

Now we will see something a little different. Suppose your HTML file has many sections, each with a heading in an <h2> tag that looks like <h2>Some text</h2>. You can create a custom function that will automatically number these headings with consecutive section numbers, so that they look like <h2>1. Some text</h2>.

```
def replace(match, number, file_name, metadata, dictionaries, data, functions, *args,_
→**kwargs):
    section_number = '%d. ' % number
    return match.group(1) + section_number + match.group(2)
# Ensure that when running over multiple files, the files are processed
# in the order in which they appear in the book
replace.file_order = 'spine'
```

Use it with the find expression:

(?s) (<h2[^<>]\*>) (.+?</h2>)

Place the cursor at the top of the file and click Replace all.

This function uses another of the useful extra arguments to replace (): the number argument. When doing a *Replace All* number is automatically incremented for every successive match.

Another new feature is the use of replace.file\_order – setting that to 'spine' means that if this search is run on multiple HTML files, the files are processed in the order in which they appear in the book. See *Choose file order when running on multiple HTML files* (page 104) for details.

#### Auto create a Table of Contents

Finally, lets try something a little more ambitious. Suppose your book has headings in h1 and h2 tags that look like <h1 id="someid">Some Text</h1>. We will auto-generate an HTML Table of Contents based on these headings. Create the custom function below:

```
from calibre import replace_entities
from calibre.ebooks.oeb.polish.toc import TOC, toc_to_html
from calibre.gui2.tweak_book import current_container
from calibre.ebooks.oeb.base import xml2str
def replace (match, number, file_name, metadata, dictionaries, data, functions, *args,_
\rightarrow **kwargs):
    if match is None:
        # All matches found, output the resulting Table of Contents.
        # The argument metadata is the metadata of the book being edited
        if 'toc' in data:
            toc = data['toc']
            root = TOC()
            for (file_name, tag_name, anchor, text) in toc:
                parent = root.children[-1] if tag_name == 'h2' and root.children else.
→root.
                parent.add(text, file_name, anchor)
           toc = toc_to_html(root, current_container(), 'toc.html', 'Table of_
⇔Contents for ' + metadata.title, metadata.language)
           print(xml2str(toc))
        else:
            print ('No headings to build ToC from found')
    else:
        # Add an entry corresponding to this match to the Table of Contents
        if 'toc' not in data:
            # The entries are stored in the data object, which will persist
            # for all invocations of this function during a 'Replace All' operation
            data['toc'] = []
        tag_name, anchor, text = match.group(1), replace_entities(match.group(2)),__

→replace_entities(match.group(3))

        data['toc'].append((file_name, tag_name, anchor, text))
        return match.group() # We don't want to make any actual changes, so return.
⇔the original matched text
# Ensure that we are called once after the last match is found so we can
# output the ToC
replace.call_after_last_match = True
# Ensure that when running over multiple files, this function is called,
# the files are processed in the order in which they appear in the book
replace.file_order = 'spine'
```

And use it with the find expression:

#### <(h[12]) [^<>] \* id=['"]([^'"]+)['"][^<>] \*>([^<>]+)

Run the search on *All text files* and at the end of the search, a window will popup with "Debug output from your function" which will have the HTML Table of Contents, ready to be pasted into toc.html.

The function above is heavily commented, so it should be easy to follow. The key new feature is the use of another useful extra argument to the replace() function, the data object. The data object is a Python *dictionary* that persists between all successive invocations of replace() during a single *Replace All* operation.

Another new feature is the use of call\_after\_last\_match - setting that to True on the replace() function means that the editor will call replace() one extra time after all matches have been found. For this extra call, the match object will be None.

This was just a demonstration to show you the power of function mode, if you really needed to generate a Table of Contents from headings in your book, you would be better off using the dedicated Table of Contents tool in *Tools*  $\rightarrow$  *Table of Contents*.

#### The API for the function mode

All function mode functions must be Python functions named replace, with the following signature:

When a find/replace is run, for every match that is found, the replace() function will be called, it must return the replacement string for that match. If no replacements are to be done, it should return match.group() which is the original string. The various arguments to the replace() function are documented below.

#### The match argument

The match argument represents the currently found match. It is a Python Match  $object^{40}$ . Its most useful method is group () which can be used to get the matched text corresponding to individual capture groups in the search regular expression.

#### The number argument

The number argument is the number of the current match. When you run *Replace All*, every successive match will cause replace () to be called with an increasing number. The first match has number 1.

#### The file\_name argument

This is the filename of the file in which the current match was found. When searching inside marked text, the file\_name is empty. The file\_name is in canonical form, a path relative to the root of the book, using / as the path separator.

#### The metadata argument

This represents the metadata of the current book, such as title, authors, language, etc. It is an object of class *calibre*. *ebooks.metadata.book.base.Metadata*(page 208). Useful attributes include, title, authors (a list of authors) and language (the language code).

<sup>&</sup>lt;sup>40</sup> https://docs.python.org/library/re.html#match-objects

#### The dictionaries argument

This represents the collection of dictionaries used for spell checking the current book. Its most useful method is dictionaries.recognized(word) which will return True if the passed in word is recognized by the dictionary for the current book's language.

#### The data argument

This a simple Python dictionary. When you run *Replace all*, every successive match will cause replace() to be called with the same dictionary as data. You can thus use it to store arbitrary data between invocations of replace() during a *Replace all* operation.

#### The functions argument

The functions argument gives you access to all other user defined functions. This is useful for code re-use. You can define utility functions in one place and re-use them in all your other functions. For example, suppose you create a function name My Function like this:

```
def utility():
    # do something

def replace(match, number, file_name, metadata, dictionaries, data, functions, *args,...
    ↔**kwargs):
    ....
```

Then, in another function, you can access the utility() function like this:

You can also use the functions object to store persistent data, that can be re-used by other functions. For example, you could have one function that when run with *Replace All* collects some data and another function that uses it when it is run afterwards. Consider the following two functions:

#### **Debugging your functions**

You can debug the functions you create by using the standard print() function from Python. The output of print will be displayed in a popup window after the Find/replace has completed. You saw an example of using print() to output an entire table of contents above.

#### Choose file order when running on multiple HTML files

When you run a *Replace all* on multiple HTML files, the order in which the files are processes depends on what files you have open for editing. You can force the search to process files in the order in which the appear by setting the file\_order attribute on your function, like this:

file\_order accepts two values, spine and spine-reverse which cause the search to process multiple files in the order they appear in the book, either forwards or backwards, respectively.

#### Having your function called an extra time after the last match is found

Sometimes, as in the auto generate table of contents example above, it is useful to have your function called an extra time after the last match is found. You can do this by setting the call\_after\_last\_match attribute on your function, like this:

#### Appending the output from the function to marked text

When running search and replace on marked text, it is sometimes useful to append so text to the end of the marked text. You can do that by setting the append\_final\_output\_to\_marked attribute on your function (note that you also need to set call\_after\_last\_match), like this:

#### Suppressing the result dialog when performing searches on marked text

You can also suppress the result dialog (which can slow down the repeated application of a search/replace on many blocks of text) by setting the suppress\_result\_dialog attribute on your function, like this:

### More examples

More useful examples, contributed by calibre users, can be found in the calibre E-book editor forum<sup>41</sup>.

#### **Snippets**

The calibre E-book editor supports *snippets*. A snippet is a piece of text that is either re-used often or contains a lot of redundant text. The editor allows you to insert a snippet with only a few key strokes. For example, suppose you often find yourself inserting link tags when editing HTML files, then you can simply type <a in the editor and press Control+J. The editor will expand it to:

```
<a href="filename"></a>
```

Not only that, the word filename will be selected, with the cursor placed over it, so that you can easily type in the real filename, using the editor's nifty *Auto-complete* (page 110) feature. And once you are done typing the filename, press Control+J again and the cursor will jump to the position in between the <a> tags so you can easily type in the text for the link.

The snippets system in the editor is very sophisticated, there are a few built-in snippets and you can create your own to suit your editing style.

The following discussion of the built-in snippets should help illustrate the power of the snippets system.

#### 1 Note

You can also use snippets in the text entry fields in the *Search & replace* panel, however, placeholders (using Con-trol+J to jump around) will not work.

### The built-in snippets

The built-in snippets are described below. Note that you can override them by creating your own snippets with the same trigger text.

### Inserting filler text [Lorem]

The first built-in snippet, and the simplest is used to insert filler text into a document. The filler text is taken from De finibus bonorum et malorum<sup>42</sup> a philosophical work by Cicero (translated to English). To use it simply type Lorem in an HTML file and press Control+J. It will be replaced by a couple of paragraphs of filler.

The definition of this snippet is very simple, the trigger text is defined as Lorem and the template is defined simply as the literal text to be inserted. You can easily customize it to use your favorite form of filler text.

<sup>&</sup>lt;sup>41</sup> https://www.mobileread.com/forums/showthread.php?t=237181

<sup>&</sup>lt;sup>42</sup> https://en.wikipedia.org/wiki/De\_finibus\_bonorum\_et\_malorum

### Inserting a self-closing HTML tag [<>]

Now let's look at a simple example of the powerful concept of *placeholders*. Say you want to insert the self-closing tag <hr/> . Just type <>, and press Control+J, the editor will expand the snippet to:

 $< \mid / >$ 

Here, the | symbol represents the current cursor position. You can then type hr and press Control+J to move the cursor to after the end of the tag. This snippet is defined as:

```
Trigger: <>
Template: <$1/>$2
```

Placeholders are simply the dollar (\$) sign followed by a number. When the snippet is expanded by pressing Control+J the cursor is positioned at the first placeholder (the placeholder with the lowest number). When you press Control+J again the cursor jumps to the next placeholder (the placeholder with the next higher number).

### Inserting an HTML link tag [<a]

HTML link tags all share a common structure. They have an href attribute and some text between the opening and closing tags. A snippet to make typing them more efficient will introduce us to some more features of placeholders. To use this snippet, simply type <a and press Control+J. The editor will expand this to:

<a href="filename|"></a>

Not only that, the word filename will be selected, with the cursor placed over it, so that you can easily type in the real filename, using the editor's nifty *Auto-complete* (page 110) feature. And once you are done typing the filename, press Control+J again and the cursor will jump to the position in between the <a> tags so you can easily type in the text for the link. After you are done typing the text, press Control+J again to jump to the point after the closing tag. This snippet is defined as:

Trigger: <a
Template: <a href="\${1:filename}">\${2\*}</a>\$3

There are a couple of new features here. First the \$1 placeholder has become more complex. It now includes some *default text* (the word filename). If a placeholder contains default text, the default text is substituted for the placeholder when the snippet is expanded. Also when you jump to a placeholder with default text using Control+J, the default text is selected. In this way, you can use default text to act as a reminder to you to fill in important parts of the template. You can specify default text for a placeholder by using the syntax:  $\{<\text{number}\}: default text\}$ .

The other new feature is that the second placeholder has an asterisk after it  $(\{2^*\})$ . This means that any text that was selected before expanding the template is substituted for the placeholder. To see this in action, select some text in the editor, press Control+J, type <a and press Control+J again, the template will be expanded to:

<a href="filename">whatever text you selected</a>

### Inserting a HTML image tag [<i]

This is very similar to inserting an HTML link, as we saw above. It allows you to quickly input an <img src="filename" alt="description" /> tag and jump between the src and alt attributes:

```
Trigger: <i
Template: <img src="${1:filename}" alt="${2*:description}" />$3
```

### Insert an arbitrary HTML tag [<<]

This allows you to insert an arbitrary full HTML tag (or wrap previously selected text in the tag). To use it, simply type << and press Control+J. The editor will expand it to:

< | >< / >

Type the tag name, for example: span and press Control+J, that will result in:

<span>|</span>

You will note that the closing tag has been automatically filled with span. This is achieved with yet another feature of placeholders, *mirroring*. Mirroring simply means that if you specify the sample placeholder more than once in a template, the second and all later positions will be automatically filled in with whatever you type in the first position, when you press Control+J. The definition for this snippet is:

```
Trigger: <<
Template: <$1>${2*}</$1>$3
```

As you can see, the first placeholder (\$1) has been specified twice, the second time in the closing tag, which will simply copy whatever you type in the opening tag.

### Inserting an arbitrary HTML tag with a class attribute [<c]

This is very similar to the insert arbitrary tag example above, except that it assumes that you want to specify a class for the tag:

```
Trigger: <c
Template: <$1 class="${2:classname}">${3*}</$1>$4
```

This will allow you to first type the tag name, press Control+J, type the class name, press Control+J type the contents of the tag and press Control+J one last time to jump out of the tag. The closing tag will be auto-filled.

### Creating your own snippets

Snippets really shine because you can create your own to suit your editing style. To create your own snippets go to  $Edit \rightarrow Preferences \rightarrow Editor \ settings \rightarrow Manage \ snippets$  in the editor. This will pop-up an easy to use dialog to help you create your own snippets. Simply click the *Add snippet* button and you will see a dialog that looks like:

	a snippet th snippets, see the User Manual				
	The name of this snippet				
Tri <u>g</u> ger:	The text used to trigger this snippet				
<u>T</u> emplate:					
<u>F</u> ile types:	✓ All css html javascript text xml				
T <u>e</u> st:					
	✓OK SCancel				

First give your snippet a name, something descriptive, to help identify the snippet in the future. Then specify the *trigger*. A trigger is simply the text that you have to type in the editor before pressing Control+J in order to expand the snippet.

Then specify the snippet template. You should start with one of the examples above and modify it to suit your needs. Finally, specify which file types you want the snippet to be active for. This way you can have multiple snippets with the same trigger text that work differently in different file types.

The next step is to test your newly created snippet. Use the *Test* box at the bottom. Type in the trigger text and press Control+J to expand the snippet and jump between placeholders.

# 5.8.9 The Reports tool

The editor includes a nice *Reports* tool (via *Tools*  $\rightarrow$  *Reports*) that shows summaries of the files, images, links, words, characters and styles used in the book. Every line in the report is hot-linked. Double clicking a line jumps to the place in the book where that item is used or defined (as appropriate). For example, in the *Links* view, you can double click entries the *Source* column to jump to where the link is defined and entries in the *Target* column to jump to where the link points.

Files Words	Filter	Files Words	Filte	r			
mages	▼ [ 4] .td_31	Images		Image	Times used	Size (KB)	Resolution
<del>Style Rules</del> Characters		Style Rules Characters	1	cover_image.jpg	1	64.25	1200×1600
	<pre>&gt; [ 4] .td_13 &gt; [ 4] .td_13 &gt; [ 4] .td_5 &gt; [ 4] .td_5 &gt; [ 4] .block_2 &gt; [ 5] .block_30 &gt; [ 5] .text_17 &gt; [ 5] .td_29</pre>		2	back.png	1	4.58	128×128
	<pre>&gt; [ 5].td[41 &gt; [ 5].td[43 &gt; [ 5].block_11 &gt; [ 5].block_11 &gt; [ 5].td[2 </pre>		з	forward.png	1	4.52	128×128
	<pre>&gt; [ 5] .block_21 &gt; [ 5] .block_21 &gt; [ 5] .td_3 &gt; [ 6] .block_8 </pre>		4	dot_green.png	2	1.49	32 x 32
	▶ [ 6] .td_38 ▶ [ 6] .td_35 ▶ [ 6] .block_3		5	image1.gif	1	0.30	12×12
	Sort by:  Counts Name Ascending  147 rules, 0 unused CRefresh					fresh Sav	e 😵 Clos

# 5.9 Special features in the code editor

The calibre HTML editor is very powerful. It has many features that make editing of HTML (and CSS) easier.

# 5.9.1 Syntax highlighting

The HTML editor has very sophisticated syntax highlighting. Features include:

- The text inside bold, italic and heading tags is made bold/italic
- As you move your cursor through the HTML, the matching HTML tags are highlighted, and you can jump to the opening or closing tag with the keyboard shortcuts Ctrl+{ and Ctrl+}. Similarly, you can select the contents of a tag with Ctrl+Alt+T or Ctrl+Shift+T.
- · Invalid HTML is highlighted with a red underline
- Spelling errors in the text inside HTML tags and attributes such as title are highlighted. The spell checking is language aware, based on the value of the lang attribute of the current tag and the overall book language.
- CSS embedded inside <style> tags is highlighted
- Special characters that can be hard to distinguish such as non-breaking spaces, different types of hyphens, etc. are highlighted.
- Links to other files in <a> tags, <img> and <link> tags all have the filenames highlighted. If the filename they point to does not exist, the filename is marked with a red underline.

# 5.9.2 Context sensitive help

You can right click on an HTML tag name or a CSS property name to get help for that tag or property.

You can also hold down the Ctrl key and click on any filename inside a link tag to open that file in the editor automatically. Similarly, Ctrl clicking a class name will take you to the first style rule that matches the tag and class.

Right clicking a class name in an HTML file will allow you to rename the class, changing all occurrences of the class throughout the book and all its stylesheets.

# 5.9.3 Auto-complete

When editing an e-book, one of the most tedious tasks is creating links to other files inside the book, or to CSS stylesheets, or images. You have to figure out the correct filename and relative path to the file. The editor has auto-complete to make that easier.

As you type a filename, the editor automatically pops up suggestions. Simply use the Tab key to select the correct file name. The editor even offers suggestions for links pointing to an anchor inside another HTML file. After you type the # character, the editor will show you a list of all anchors in the target file, with a small snippet of text to help you choose the right anchor.

Note that unlike most other completion systems, the editor's completion system uses subsequence matching. This means that you can type just two or three letters from anywhere in the filename to complete the filename. For example, say you want the filename .../images/arrow1.png, you can simply type ia1 and press Tab to complete the filename. When searching for matches, the completion system prioritizes letters that are at the start of a word, or immediately after a path separator. Once you get used to this system, you will find it saves you a lot of time and effort.

# 5.9.4 Snippets

The calibre E-book editor supports *snippets*. A snippet is a piece of text that is either re-used often or contains a lot of redundant text. The editor allows you to insert a snippet with only a few key strokes. The snippets are very powerful, with many features, such as placeholders you can jump between, automatic mirroring of repeated text and so on. For more information, see *Snippets* (page 105).

# THE CALIBRE CONTENT SERVER

The calibre *Content server* allows you to access your calibre libraries and read books directly in a browser on your favorite mobile phone or tablet device. As a result, you do not need to install any dedicated book reading/management apps on your phone. Just use the browser. The server downloads and stores the book you are reading in an off-line cache so that you can read it even when there is no internet connection.

#### Contents

- Accessing the Content server from other devices (page 112)
  - Accessing the server from devices on your home network (page 112)
  - Accessing the server from anywhere on the internet (page 113)
- The server interface (page 113)
  - The book list (page 113)
  - The book viewer (page 114)
- Browser support (page 114)
- Enabling offline support (page 114)
- Managing user accounts from the command-line only (page 114)
- Integrating the calibre Content server into other servers (page 115)
  - Using a full virtual host (page 115)
  - Using a URL prefix (page 115)
- Creating a service for the calibre server on a modern Linux system (page 116)

To start the server, click the *Connect/share* button and choose *Start Content server*. You might get a message from your computer's firewall or anti-virus program asking if it is OK to allow access to calibre.exe. Click the Allow or OK button. Then open a browser (preferably Chrome or Firefox) in your computer and type in the following address:

#### http://127.0.0.1:8080

This will open a page in the browser showing you your calibre libraries, click on any one and browse the books in it. Click on a book, and it will show you all the metadata about the book, along with buttons to *Read* and *Download* the book. Click the *Read* button to start reading the book.



The address used above http://127.0.0.1:8080 will only work on the computer that is running calibre. To access the server from other computers/phones/tablets/etc. you will need to do a little more work, as described in the next section.

# 6.1 Accessing the Content server from other devices

There are two types of remote device access that you will typically need. The first, simpler kind is from within your home network. If you are running calibre on a computer on your home network and you have also connected your other devices to the same home network, then you should be easily able to access the server on those devices.

### 6.1.1 Accessing the server from devices on your home network

After starting the server in calibre as described above, click the *Connect/share* button again. Instead of the *Start Content server* action, you should see a *Stop Content server* action instead. To the right of this action will be listed an IP address and port number. These look like a bunch of numbers separated by periods. For example:

Stop Content server [192.168.1.5, port 8080]

These numbers tell you what address to use to connect to the server in your devices. Following the example above, the address becomes:

http://192.168.1.5:8080

The first part of the address is always http:// the next part is the IP address, which is the numbers before the comma and finally we have the port number which must be added to the IP address with a colon (:). If you are lucky, that should be all you need and you will be looking at the calibre libraries on your device. If not, read on.

### Trouble-shooting the home network connection

If you are unable to access the server from your device, try the following steps:

- 1. Check that the server is running by opening the address http://127.0.0.1:8080 in a browser running on the same computer as the server.
- 2. Check that your firewall/anti-virus is allowing connections to your computer on the port 8080 and to the calibre program. The easiest way to eliminate the firewall/anti-virus as the source of problems is to temporarily turn them both off and then try connecting. You should first disconnect from the internet, before turning off the firewall, to keep your computer safe.
- 3. Check that your device and computer are on the same network. This means they should both be connected to the same wireless router. In particular neither should be using a cellular or ISP provided direct-WiFi connection.
- 4. If you have non-standard networking setup, it might be that the IP address shown on the *Connect/share* menu is incorrect. In such a case you will have to figure out what the correct IP address to use is, yourself. Unfortunately, given the infinite diversity of network configurations possible, it is not possible to give you a roadmap for doing so.
- 5. If you have setup a username and password, first try it without that to see if it is causing issues. Some e-ink devices have browsers that do not handle authentication. You can sometimes workaround this by including the username and password in the URL, for example: http://username:password@192.168.1.2:8080.
- 6. If you are stuck, you can always ask for help in the calibre user forums<sup>43</sup>.

<sup>43</sup> https://www.mobileread.com/forums/forumdisplay.php?f=166

# 6.1.2 Accessing the server from anywhere on the internet

### 🛕 Warning

Before doing this you should turn on username/password protection in the server, otherwise anyone in the world will be able to access your books. Go to *Preferences*  $\rightarrow$  *Sharing*  $\rightarrow$  *Sharing over the net* and enable the option to *Require username and password to access the content server*.

While the particular details on setting up internet access vary depending on the network configuration and type of computer you are using, the basic schema is as follows.

- 1. Find out the external IP address of the computer you are going to run the server on. You can do that by visiting the site What is my IP address<sup>44</sup> in a browser running on the computer.
- 2. If the computer is behind a router, enable port forwarding on the router to forward the port 8080 (or whatever port you choose to run the calibre Content server on) to the computer.
- 3. Make sure the calibre server is allowed through any firewalls/anti-virus programs on your computer.
- 4. Now you should be able to access the server on any internet-connected device using the IP address you found in the first step. For example, if the IP address you found was 123.123.123.123.123 and the port you are using for the calibre server is 8080, the address to use on your device becomes: http://123.123.123.123.123.8080.
- 5. Optionally, use a service like no-ip<sup>45</sup> to setup an easy to remember address to use instead of the IP address you found in the first step.

#### Note

For maximum security, you should also enable HTTPS on the Content server. You can either do so directly in the server by providing the path to the HTTPS certificate to use in the advanced configuration options for the server, or you can setup a reverse proxy as described below, to use an existing HTTPS setup.

# 6.2 The server interface

The server interface is a simplified version of the main calibre interface, optimised for use with touch screens. The home screen shows you books you are currently reading as well as allowing to choose a calibre library you want to browse. The server in calibre gives you access to all your libraries, not just a single one, as before.

### 6.2.1 The book list

The server book list is a simple grid of covers. Tap on a cover to see the detailed metadata for a book, or to read the book. If you prefer a more detailed list, you can change the default view by clicking the three vertical dots in the top right corner.

Sorting and searching of the book list should be familiar to calibre users. They can be accessed by clicking their icons in the top right area. They both work exactly the same as in the main calibre program. The search page even allows you to construct search queries by clicking on authors/tags/etc., just as you can using the Tag browser in the main program.

A much loved feature of the main program, *Virtual libraries* is present in the server interface as well. Click the three vertical dots in the top right corner to choose a Virtual library.

<sup>44</sup> https://www.whatismyip.com/

<sup>&</sup>lt;sup>45</sup> https://www.noip.com/free

### 6.2.2 The book viewer

You can read any book in your calibre library by simply tapping on it and then tapping the *Read* button. The book viewer is very simple to operate. You can both tap and swipe to turn pages. Swiping up/down skips between chapters. Tapping the top quarter of the screen gets you the detailed controls and viewer preferences.

If you leave the Content server running, you can even open the same book on multiple devices and it will remember your last read position. If it does not you can force a sync by tapping in the top quarter and choosing *Sync*.

# 6.3 Browser support

The new calibre server makes lots of use of advanced HTML 5 and CSS 3 features. As such it requires an up-to-date browser to use. It has been tested on Android Chrome and iOS Safari as well as Chrome and Firefox on the desktop.

The server is careful to use functionality that has either been already standardised or is on the standards track. As such if it does not currently work with your favorite browser, it probably will once that browser has caught up.

If you are using a particularly old or limited browser or you don't like to run JavaScript, you can use the *mobile* view, by simply adding /mobile to the server address.

#### Note

On iOS, Apple allows only a single browser engine, so Firefox, Chrome and Safari are all actually the same browser under the hood. The new server interface requires iOS 10.3.2 or newer. On Android, the server has been tested with Chrome version 58 and newer.

# 6.4 Enabling offline support

Browser makers have been trying to force people to use SSL by disabling advanced features in their browsers for plain HTTP connections. One such casualty is ApplicationCache, which was what was used in calibre for offline support. As a result now-a-days sadly, offline mode works only as long as you keep the browser tab open. In addition, in Firefox on Android, you will need to type about:config and create a preference called browser.tabs.useCache and set it to true.

# 6.5 Managing user accounts from the command-line only

The calibre program has a nice section in *Preferences* to allow you to manage user accounts for the server. However, if you want to run the standalone server and cannot run the main calibre program on the same computer/user account, you can also manage users using just the command-line.

You can manage user accounts using the --manage-users option to the standalone calibre-server program. Suppose you want to store the user database in the folder /srv/calibre, then you create it by running:

calibre-server --userdb /srv/calibre/users.sqlite --manage-users

Just follow the prompts to create user accounts, set their permission, etc. Once you are done, you can run the server as:

```
calibre-server --userdb /srv/calibre/users.sqlite --enable-auth
```

It will use the user accounts you created in the previous step.

# 6.6 Integrating the calibre Content server into other servers

Here, we will show you how to integrate the calibre Content server into another server. The most common reason for this is to make use of SSL or to serve the calibre library as part of a larger site. The basic technique is to run the calibre server and setup a reverse proxy to it from the main server.

A reverse proxy is when your normal server accepts incoming requests and passes them onto the calibre server. It then reads the response from the calibre server and forwards it to the client. This means that you can simply run the calibre server as normal without trying to integrate it closely with your main server.

# 6.6.1 Using a full virtual host

The simplest configuration is to dedicate a full virtual host to the calibre server. In this case, run the calibre server as:

```
calibre-server
```

Now setup the virtual host in your main server, for example, for nginx:

```
http {
    client_max_body_size 64M; # needed to upload large books
}
server {
    listen [::]:80;
    server_name myserver.example.com;
    location / {
        proxy_pass http://127.0.0.1:8080;
    }
}
```

Or, for Apache:

```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_http_module modules/mod_proxy_http.so
<VirtualHost *:80>
   ServerName myserver.example.com
   AllowEncodedSlashes On
   ProxyPreserveHost On
   ProxyPass "/" "http://localhost:8080/"
</VirtualHost>
```

# 6.6.2 Using a URL prefix

If you do not want to dedicate a full virtual host to calibre, you can have it use a URL prefix. Start the calibre server as:

calibre-server --url-prefix /calibre --port 8080

The key parameter here is --url-prefix /calibre. This causes the Content server to serve all URLs prefixed by / calibre. To see this in action, visit http://localhost:8080/calibre in your browser. You should see the normal Content server website, but now it will run under /calibre.

With nginx, the required configuration is:

```
http {
    client_max_body_size 64M; # needed to upload large books
}
proxy_set_header X-Forwarded-For $remote_addr;
location /calibre/ {
    proxy_buffering off;
    proxy_pass http://127.0.0.1:8080$request_uri;
}
location /calibre {
    # we need a trailing slash for the Application Cache to work
    rewrite /calibre /calibre/ permanent;
}
```

For Apache, first enable the proxy modules in Apache, by adding the following to httpd.conf:

```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_http_module modules/mod_proxy_http.so
```

The exact technique for enabling the proxy modules will vary depending on your Apache installation. Once you have the proxy modules enabled, add the following rules to httpd.conf (or if you are using virtual hosts to the conf file for the virtual host in question):

```
AllowEncodedSlashes On
RewriteEngine on
RewriteRule ^/calibre/(.*) http://127.0.0.1:8080/calibre/$1 [proxy]
RedirectMatch permanent ^/calibre$ /calibre/
```

That's all, you will now be able to access the calibre Content server under the /calibre URL in your main server. The above rules pass all requests under /calibre to the calibre server running on port 8080 and thanks to the --url-prefix option above, the calibre server handles them transparently.

### Note

When using a reverse proxy, you should tell the calibre Content server to only listen on localhost, by using --listen-on 127.0.0.1. That way, the server will only listen for connections coming from the same computer, i.e. from the reverse proxy.

#### Note

If you have setup SSL for your main server, you should tell the calibre server to use basic authentication instead of digest authentication, as it is faster. To do so, pass the --auth-mode=basic option to calibre-server.

# 6.7 Creating a service for the calibre server on a modern Linux system

You can easily create a service to run calibre at boot on a modern (systemd<sup>46</sup>) based Linux system. Just create the file /etc/systemd/system/calibre-server.service with the contents shown below:

<sup>&</sup>lt;sup>46</sup> https://www.freedesktop.org/wiki/Software/systemd/

```
[Unit]
Description=calibre Content server
After=network.target
[Service]
Type=simple
User=mylinuxuser
Group=mylinuxgroup
ExecStart=/opt/calibre/calibre-server "/path/to/calibre library folder"
[Install]
```

Change mylinuxuser and mylinuxgroup to whatever user and group you want the server to run as. This should be the same user and group that own the files in the calibre library folder. Note that it is generally not a good idea to run the server as root. Also change the path to the calibre library folder to suit your system. You can add multiple libraries if needed. See the help for the calibre-server command.

Now run:

sudo systemctl start calibre-server

to start the server. Check its status with:

WantedBy=multi-user.target

sudo systemctl status calibre-server

To make it start at boot, run:

```
sudo systemctl enable calibre-server
```

#### **1** Note

The calibre server *does not* need a running X server, but it does need the X libraries installed as some components it uses link against them.

### Note

The calibre server also supports systemd socket activation, so you can use that, if needed, as well.

CHAPTER

# SEVEN

# **COMPARING E-BOOKS**

calibre includes an integrated e-book comparison tool that can be used to see what has changed inside an e-book after editing or converting it. It can compare books in the EPUB and AZW3 formats.

To use it, either open the e-book in the tool for *Editing e-books* (page 79) and then click *File*  $\rightarrow$  *Compare to other book* or use the *Book details* (page 20) panel. If you do a conversion from EPUB to EPUB, the original EPUB file will be saved as ORIGINAL\_EPUB. Simply right click on the ORIGINAL\_EPUB entry in the Book details panel and choose *Compare to EPUB format*.

The comparison tool that opens will look like the screenshot below. It shows you the differences in text, styles and images in the chosen books.

images/forward.png	images/forward.png					
Size: 4.5 KB Resolution: 128x128	Size: 64.3 KB Resolution: 128×128					
index_split_000.html	index_split_000.html					
8 9 <body class="calibre"> 10</body>	<pre></pre>					
<pre>11 Demonstration         of DOCX support in calibre</pre>	<pre>Demonstration 11 of the ebook comparison tool in calibre </pre>					
<pre>12     This document demonstrates the     ability of the calibre DOCX Input plugin to convert the     various typographic features in a Microsoft Word (2007     and newer) document. Convert this document to a modern     ebook format, such as AZW3 for Kindles or EPUB for     other ebook readers, to see it in action. 14 15    There is support for images,     tables, lists, footnotes, endnotes, links, dropcaps and     various types of text and paragraph level     formatting. 16 26 27 28 29 20 20 20 20 20 20 20 20 20 20 20 20 20</pre>	<pre>sp class="block1"&gt;This document demonstrates the ability of the calibre ebook comaprison tool to show changes made to the text, styles and images in a book. <pre>styles="block1"&gt;There is support for images, tables, lists, footnotes, endnotes, links, dropcaps and various types of text and paragraph level formatting. <pre>16 stylesheet.css .block1 {     color: black;     display: block;     text-indent: 21.6pt;     padding: 0;     margin: 0;     } .block2 {     color: green;     display: block;     text-align: center;     font-family: serif;     font-size: 0.75en;     line-height: 1.15;     color: 27</pre></pre></pre>					
Previous change     Next change     Search for text	Next <u>match</u> A Previous match C Left panel Right panel S Options					
Before Editing for screenshot <> Current state	Close <u>Pevert changes</u>					

# 7.1 Understanding the comparison view

As can be seen in the screenshot above, the comparison view shows the differences between the two books side by side. Only the differences, with a few lines of context around them are shown. This makes it easy to see at a glance only what was changed inside a large document like a book.

Added text is shown with a green background, removed text with a red background and changed text with a blue background.

The line numbers of all changed text are show at the sides, making it easy to go to a particular change in the editor. When you open the comparison tool from within the editor, you can also double click on a line in the right panel to go to that line in the editor automatically.

One useful technique when comparing books is to tell the comparison tool to beautify the text and style files before calculating differences. This can often result in cleaner and easier to follow differences. To do this, click the *Options* button in the bottom right and choose *Beautify files before comparing*. Note that beautifying can sometimes have undesired

effects, as it can cause invalid markup to be altered to make it valid. You can also change the number of lines of context shown around differences via the *Options* button.

You can search for any text in the differences via the Search bar at the bottom. You will need to specify which panel to search, the *Left* or the *Right*.

# 7.2 Launching the comparison tool

The comparison tool is most useful when you have two versions of the same book and you want to see what is different between them. To that end, there are several ways to launch the tool.

# 7.2.1 Comparing two e-book files

Open the first file in the *Editing e-books* (page 79) tool. Now click *File*  $\rightarrow$  *Compare to another book* and choose the second file (it must be in the same format as the first). The comparison view will open with the file being edited on the right and the second file on the left.

# 7.2.2 Comparing the ORIGINAL\_FMT to FMT

When you do a conversion in calibre from a FMT to itself, the original file is saved as ORIGINAL\_FMT. You can see what was changed by the conversion, by right clicking on the ORIGINAL\_FMT entry in the *Book details* (page 20) panel in the main calibre window and selecting *Compare to FMT*. The comparison view will open with ORIGINAL\_FMT on the left and FMT on the right.

# 7.2.3 Comparing a checkpoint to the current state of the book while editing

The *Editing e-books* (page 79) tool has a very useful feature, called *Checkpoints* (page 91). This allows you to save the current state of the book as a named *checkpoint*, to which you can revert if you do not like the changes you have made since creating the checkpoint. Checkpoints are also created automatically when you perform various automated actions in the editor. You can see the list of checkpoints by going to *View*  $\rightarrow$  *Checkpoints* and then use the *Compare* button to compare the book at the selected checkpoint with the current state. The comparison tool will show the checkpoint on the left and the current state on the right.

### CHAPTER

# EIGHT

# EDITING E-BOOK METADATA

### Contents

- Editing the metadata of one book at a time (page 123)
  - Downloading metadata (page 124)
  - Managing book formats (page 124)
  - All about covers (page 124)
- Editing the metadata of many books at a time (page 124)
  - Search and replace (page 124)
  - Bulk downloading of metadata (page 125)
- Adding extra data files to a book (page 125)

E-books come in all shapes and sizes and more often than not, their metadata (things like title/author/series/publisher) is incomplete or incorrect. The simplest way to change metadata in calibre is to simply double click on an entry and type in the correct replacement. For more sophisticated, "power editing" use the edit metadata tools discussed below.

# 8.1 Editing the metadata of one book at a time

Click the book you want to edit and then click the *Edit metadata* button or press the E key. A dialog opens that allows you to edit all aspects of the metadata. It has various features to make editing faster and more efficient. A list of the commonly used tips:

- You can click the button in between title and authors to swap them automatically.
- You can click the button next to author sort to have calibre automatically fill it in using the sort values stored with each author. Use the *Manage authors* dialog to see and change the authors' sort values. This dialog can be opened by clicking and holding the button next to author sort.
- You can click the button next to tags to use the *Tag editor* to manage the tags associated with the book.
- The "Ids" box can be used to enter an ISBN (and many other types of id), it will have a red background if you enter an invalid ISBN. It will be green for valid ISBNs.
- The author sort box will be red if the author sort value differs from what calibre thinks it should be.

# 8.1.1 Downloading metadata

The nicest feature of the edit metadata dialog is its ability to automatically fill in many metadata fields by getting metadata from various websites. Currently, calibre uses Google Books and Amazon. The metadata download can fill in Title, author, series, tags, rating, description and ISBN for you.

To use the download, fill in the title and author fields and click the *Fetch metadata* button. calibre will present you with a list of books that most closely match the title and author. If you fill in the ISBN field first, it will be used in preference to the title and author. If no matches are found, try making your search a little less specific by including only some key words in the title and only the author last name.

# 8.1.2 Managing book formats

In calibre, a single book entry can have many different *formats* associated with it. For example you may have obtained the Complete Works of Shakespeare in EPUB format and later converted it to MOBI to read on your Kindle. calibre automatically manages multiple formats for you. In the *Available formats* section of the Edit metadata dialog, you can manage these formats. You can add a new format, delete an existing format and also ask calibre to set the metadata and cover for the book entry from the metadata in one of the formats.

# 8.1.3 All about covers

You can ask calibre to download book covers for you, provided the book has a known ISBN. Alternatively you can specify a file on your computer to use as the cover. calibre can even generate a default cover with basic metadata on it for you. You can drag and drop images onto the cover to change it and also right click to copy/paste cover images.

In addition, there is a button to automatically trim borders from the cover, in case your cover image has an ugly border.

# 8.2 Editing the metadata of many books at a time

First select the books you want to edit by holding Ctrl or Shift and clicking on them. If you select more than one book, clicking the *Edit metadata* button will cause the *Bulk* metadata edit dialog to open. Using this dialog, you can quickly set the author/publisher/rating/tags/series etc of a bunch of books to the same value. This is particularly useful if you have just imported a number of books that have some metadata in common. This dialog is very powerful, for example, it has a *Search and replace* tab that you can use to perform bulk operations on metadata and even copy metadata from one column to another.

The normal edit metadata dialog also has *Next* and *Previous* buttons that you can use to edit the metadata of several books one after the other.

# 8.2.1 Search and replace

The *Edit metadata for many books* dialog allows you to perform arbitrarily powerful search and replace operations on the selected books. By default it uses a simple text search and replace, but it also support *regular expressions*. For more on regular expressions, see *All about using regular expressions in calibre* (page 212).

As noted above, there are two search and replace modes: character match and regular expression. Character match will look in the *Search field* you choose for the characters you type in the *search for* box and replace those characters with what you type in the *replace with* box. Each occurrence of the search characters in the field will be replaced. For example, assume the field being searched contains *a bad cat*. If you search for *a* to be replaced with *HELLO*, then the result will be *HELLO bHELLOd cHELLOt*.

If the field you are searching on is a *multiple* field like tags, then each tag is treated separately. For example, if your tags contain *Horror, Scary*, the search expression *r*, will not match anything because the expression will first be applied to *Horror* and then to *Scary*.

If you want the search to ignore upper/lowercase differences, uncheck the Case sensitive box.

You can have calibre change the case of the result (information after the replace has happened) by choosing one of the functions from the *Apply function after replace* box. The operations available are:

- Lower case change all the characters in the field to lower case
- Upper case change all the characters in the field to upper case
- Title case capitalize each word in the result.

The *Your test* box is provided for you to enter text to check that search/replace is doing what you want. In the majority of cases the book test boxes will be sufficient, but it is possible that there is a case you want to check that isn't shown in these boxes. Enter that case into *Your test*.

Regular expression mode has some differences from character mode, beyond (of course) using regular expressions. The first is that functions are applied to the parts of the string matched by the search string, not the entire field. The second is that functions apply to the replacement string, not to the entire field.

The third and most important is that the replace string can make reference to parts of the search string by using backreferences. A backreference is  $\n$  where n is an integer that refers to the n'th parenthesized group in the search expression. For example, given the same example as above, *a bad cat*, a search expression *a* (...) (...), and a replace expression *a*  $\2 \I$ , the result will be *a cat bad*. Please see the *All about using regular expressions in calibre* (page 212) for more information on backreferences.

One useful pattern: assume you want to change the case of an entire field. The easiest way to do this is to use character mode, but lets further assume you want to use regular expression mode. The search expression should be  $(^{.*}\$)$ , the replace expression should be  $\backslash I$ , and the desired case change function should be selected.

Finally, in regular expression mode you can copy values from one field to another. Simply make the source and destination field different. The copy can replace the destination field, prepend to the field (add to the front), or append to the field (add at the end). The 'use comma' checkbox tells calibre to (or not to) add a comma between the text and the destination field in prepend and append modes. If the destination is multiple (e.g., tags), then you cannot uncheck this box.

Search and replace is done after all the other metadata changes in the other tabs are applied. This can lead to some confusion, because the test boxes will show the information before the other changes, but the operation will be applied after the other changes. If you have any doubts about what is going to happen, do not mix search/replace with other changes.

### 8.2.2 Bulk downloading of metadata

If you want to download the metadata for multiple books at once, right-click the *Edit metadata* button and select *Download metadata*. You can choose to download only metadata, only covers, or both.

# 8.3 Adding extra data files to a book

calibre can store any number of extra data files associated to a book. These can be alternate covers, supplementary material, etc. They cannot be viewed directly or used as conversion sources. Nor are they indexed by the Full text search engine in calibre. To view/add/delete them select the book and right click the *Edit metadata* button and choose *Manage data files*. This will pop-up a window where you can perform operations on these files. Alternately, you can right click the *Add books* button and choose *Add data files to selected book records* to more quickly add data files.

### CHAPTER

# NINE

# FREQUENTLY ASKED QUESTIONS

### Contents

- *E-book format conversion* (page 127)
- Device integration (page 130)
- *Library management* (page 137)
- Miscellaneous (page 141)

# 9.1 E-book format conversion

### Contents

- What formats does calibre support conversion to/from? (page 127)
- What are the best source formats to convert? (page 128)
- *I converted a PDF file, but the result has various problems?* (page 128)
- How do I convert my file containing non-English characters, or smart quotes? (page 128)
- What's the deal with Table of Contents in MOBI files? (page 128)
- How do I convert a collection of HTML files in a specific order? (page 129)
- The EPUB I produced with calibre is not valid? (page 130)
- How do I use some of the advanced features of the conversion tools? (page 130)

### 9.1.1 What formats does calibre support conversion to/from?

calibre supports the conversion of many input formats to many output formats. It can convert every input format in the following list, to every output format.

*Input Formats:* AZW, AZW3, AZW4, CBZ, CBR, CB7, CBC, CHM, DJVU, DOCX, EPUB, FB2, FBZ, HTML, HTMLZ, KEPUB, LIT, LRF, MOBI, ODT, PDF, PRC, PDB, PML, RB, RTF, SNB, TCR, TXT, TXTZ

*Output Formats:* AZW3, EPUB, DOCX, FB2, HTMLZ, KEPUB, OEB, LIT, LRF, MOBI, PDB, PMLZ, RB, PDF, RTF, SNB, TCR, TXT, TXTZ, ZIP

### 1 Note

PRC is a generic format, calibre supports PRC files with TextRead and MOBIBook headers. PDB is also a generic format. calibre supports eReader, Plucker (input only), PML and zTxt PDB files. DJVU support is only for converting DJVU files that contain embedded text. These are typically generated by OCR software. MOBI books can be of two types Mobi6 and KF8. calibre fully supports both. MOBI files often have .azw or .azw3 file extensions. DOCX files from Microsoft Word 2007 and newer are supported.

# 9.1.2 What are the best source formats to convert?

In order of decreasing preference: LIT, MOBI, AZW, EPUB, KEPUB, AZW3, FB2, FBZ, DOCX, HTML, PRC, ODT, RTF, PDB, TXT, PDF

### 9.1.3 I converted a PDF file, but the result has various problems?

PDF is a terrible format to convert from. For a list of the various issues you will encounter when converting PDF, see: *Convert PDF documents* (page 75).

### 9.1.4 How do I convert my file containing non-English characters, or smart quotes?

#### There are two aspects to this problem:

- Knowing the encoding of the source file: calibre tries to guess what character encoding your source files use, but often, this is impossible, so you need to tell it what encoding to use. This can be done in the GUI via the *Input character encoding* field in the *Look & feel → Text* section of the conversion dialog. The command-line tools have an ebook-convert-txt-input --input-encoding (page 336) option.
- 2. When adding HTML files to calibre, you may need to tell calibre what encoding the files are in. To do this go to *Preferences → Advanced → Plugins → File type* and customize the *HTML to ZIP* plugin, telling it what encoding your HTML files are in. Now when you add HTML files to calibre they will be correctly processed. HTML files from different sources often have different encodings, so you may have to change this setting repeatedly. A common encoding for many files from the web is cp1252 and I would suggest you try that first. Note that when converting HTML files, leave the input encoding setting mentioned above blank. This is because the *HTML to ZIP* plugin automatically converts the HTML files to a standard encoding (UTF-8).

### 9.1.5 What's the deal with Table of Contents in MOBI files?

The first thing to realize is that most e-books have two tables of contents. One is the traditional Table of Contents, like the ToC you find in paper books. This Table of Contents is part of the main document flow and can be styled however you like. This ToC is called the *content ToC*.

Then there is the *metadata ToC*. A metadata ToC is a ToC that is not part of the book text and is typically accessed by some special button on a reader. For example, in the calibre E-book viewer, you use the Show Table of Contents button to see this ToC. This ToC cannot be styled by the book creator. How it is represented is up to the viewer program.

In the MOBI format, the situation is a little confused. This is because the MOBI format, alone amongst mainstream e-book formats, *does not* have decent support for a metadata ToC. A MOBI book simulates the presence of a metadata ToC by putting an *extra* content ToC at the end of the book. When you click Go to Table of Contents on your Kindle, it is to this extra content ToC that the Kindle takes you.

Now it might well seem to you that the MOBI book has two identical ToCs. Remember that one is semantically a content ToC and the other is a metadata ToC, even though both might have exactly the same entries and look the same. One can be accessed directly from the Kindle's menus, the other cannot.

When converting to MOBI, calibre detects the *metadata ToC* in the input document and generates an end-of-file ToC in the output MOBI file. You can turn this off by an option in the MOBI Output settings. You can also tell calibre whether

to put it at the start or the end of the book via an option in the MOBI Output settings. Remember this ToC is semantically a *metadata ToC*, in any format other than MOBI it *cannot not be part of the text*. The fact that it is part of the text in MOBI is an accident caused by the limitations of MOBI. If you want a ToC at a particular location in your document text, create one by hand. So we strongly recommend that you leave the default as it is, i.e. with the metadata ToC at the end of the book. Also note that if you disable the generation of the end-of-file ToC the resulting MOBI file may not function correctly on a Kindle, since the Kindle's use the metadata ToC for many things, including the Page Flip feature.

If you have a hand edited ToC in the input document, you can use the ToC detection options in calibre to automatically generate the metadata ToC from it. See the conversion section of the User Manual for more details on how to use these options.

Finally, I encourage you to ditch the content ToC and only have a metadata ToC in your e-books. Metadata ToCs will give the people reading your e-books a much superior navigation experience (except on the Kindle, where they are essentially the same as a content ToC).

#### \rm 1 Note

The newer AZW3 format has proper support for a metadata ToC. However, the Kindle firmware tends to malfunction if you disable the generation of the end-of-file inline ToC. So it is recommended that you leave the generated ToC alone. If you create an AZW3 file with a metadata ToC and no end-of-file generated ToC, some features on the Kindle will not work, such as the Page Flip feature.

### 9.1.6 How do I convert a collection of HTML files in a specific order?

In order to convert a collection of HTML files in a specific order, you have to create a table of contents file. That is, another HTML file that contains links to all the other files in the desired order. Such a file looks like:

<html>

```
<body>
<h1>Table of Contents</h1>

<a href="file1.html">First File</a><br/>
<a href="file2.html">Second File</a><br/>
</body>
</html>
```

Then, just add this HTML file to the GUI and use the *Convert* button to create your e-book. You can use the option in the Table of Contents section in the conversion dialog to control how the Table of Contents is generated.

### Note

By default, when adding HTML files, calibre follows links in the files in *depth first* order. This means that if file A.html links to B.html and C.html and D.html, but B.html also links to D.html, then the files will be in the order A.html, B.html, D.html, C.html. If instead you want the order to be A.html, B.html, C.html, D.html then you must tell calibre to add your files in *breadth first* order. Do this by going to *Preferences*  $\rightarrow$  *Advanced*  $\rightarrow$  *Plugins*  $\rightarrow$  *File type* and customizing the *HTML to ZIP* plugin.

# 9.1.7 The EPUB I produced with calibre is not valid?

calibre does not guarantee that an EPUB produced by it is valid. The only guarantee it makes is that if you feed it valid XHTML 1.1 + CSS 2.1 it will output a valid EPUB. calibre tries hard to ensure that EPUBs it produces actually work as intended on a wide variety of devices, a goal that is incompatible with producing valid EPUBs, and one that is far more important to the vast majority of its users. If you need a tool that always produces valid EPUBs, calibre is not for you. This means, that if you want to send a calibre produced EPUB to an online store that uses an EPUB validity checker, you have to make sure that the EPUB is valid yourself, calibre will not do it for you – in other words you must feed calibre valid XHTML + CSS as the input documents.

# 9.1.8 How do I use some of the advanced features of the conversion tools?

You can get help on any individual feature of the converters by mousing over it in the GUI or running <code>ebook-convert dummy.html .epub -h</code> at a terminal. A good place to start is to look at the following demo file that demonstrates some of the advanced features html-demo.zip<sup>47</sup>.

# 9.2 Device integration

### Contents

- What devices does calibre support? (page 130)
- How can I help get my device supported in calibre? (page 131)
- *My device is not being detected by calibre?* (page 131)
- My device is non-standard or unusual. What can I do to connect to it? (page 131)
- How do I use calibre with my iPad/iPhone/iPod touch? (page 132)
- How do I use calibre with my Android phone/tablet or Kindle Fire? (page 132)
- Can I access my calibre books using the web browser in my Kindle or other reading device? (page 133)
- I cannot send emails using calibre? (page 134)
- My device is getting mounted read-only in Linux, so calibre cannot connect to it? (page 134)
- Why does calibre not support collections on the Kindle or shelves on the Nook? (page 135)
- I am getting an error when I try to use calibre with my Kobo Touch/Glo/etc.? (page 135)
- Covers for books I send to my e-ink Kindle show up momentarily and then are replaced by a generic cover? (page 135)
- Covers for books sent to my Kindle ColorSoft and newer do not show up in the book list? (page 136)
- The covers for my MOBI files have stopped showing up in Kindle for PC/Kindle for Android/iPad etc. (page 136)
- I transferred some books to my Kindle using calibre and they did not show up? (page 136)

### 9.2.1 What devices does calibre support?

calibre can directly connect to all the major (and most of the minor) e-book reading devices, smartphones, tablets, etc. In addition, using the *Connect to folder* function you can use it with any e-book reader that exports itself as a USB disk. Finally, you can connect wirelessly to any device that has a web browser using the calibre Content server.

<sup>&</sup>lt;sup>47</sup> https://calibre-ebook.com/downloads/html-demo.zip

# 9.2.2 How can I help get my device supported in calibre?

If your device appears as a USB disk to the operating system, adding support for it to calibre is very easy. We just need some information from you:

- Complete list of e-book formats that your device supports.
- Is there a special folder on the device in which all e-book files should be placed? Also does the device detect files placed in sub-folders?
- We also need information about your device that calibre will collect automatically. First, if your device supports SD cards, insert them. Then connect your device to the computer. In calibre go to *Preferences* → *Miscellaneous* and click the "Debug device detection" button. This will create some debug output. Copy it to a file and repeat the process, this time with your device disconnected from your computer.
- Send both the above outputs to us with the other information and we will write a device driver for your device.

Once you send us the output for a particular operating system, support for the device in that operating system will appear in the next release of calibre. To send us the output, open a bug report and attach the output to it. See how to report bugs<sup>48</sup>.

# 9.2.3 My device is not being detected by calibre?

Follow these steps to find the problem:

- Make sure that you are connecting only a single device to your computer at a time. Do not have another calibre supported device like an iPhone/iPad etc. at the same time.
- If you are connecting an Apple iDevice (iPad, iPod Touch, iPhone), Apple no longer allows third party software to connect to their devices using a USB cable. Instead use a wireless connection, via the calibre Content server.
- If you are connecting a 2024 Kindle or newer or an Android device, and are on macOS or Linux, read the note under *Using a USB cable* (page 132).
- On macOS if you get permission errors when connecting a device to calibre, you can fix that by looking under *System Preferences > Security and Privacy > Privacy > Files and Folders.*
- Make sure you are running the latest version of calibre (currently 8.7.0). The latest version can always be downloaded from the calibre website<sup>49</sup>. You can tell what version of calibre you are currently running by looking at the bottom line of the main calibre window.
- Ensure your operating system is seeing the device. That is, the device should show up in Windows Explorer (in Windows) or Finder (in macOS).
- In calibre, go to *Preferences*  $\rightarrow$  *Ignored Devices* and check that your device is not being ignored
- If all the above steps fail, go to *Preferences*  $\rightarrow$  *Miscellaneous* and click *Debug device detection* with your device attached and post the output as a ticket on the calibre bug tracker<sup>50</sup>.

# 9.2.4 My device is non-standard or unusual. What can I do to connect to it?

In addition to the *Connect to folder* function found under the *Connect/share* button, calibre provides a User defined device plugin that can be used to connect to any USB device that shows up as a disk drive in your operating system. Note: on Windows, the device must have a drive letter for calibre to use it. See the device plugin Preferences -> Plugins -> Device plugins -> User defined and Preferences -> Miscellaneous -> Get information to setup the user defined device for more information. Note that if you are using the user defined plugin for a device normally detected by a builtin calibre plugin, you must disable the builtin plugin first, so that your user defined plugin is used instead.

<sup>&</sup>lt;sup>48</sup> https://calibre-ebook.com/bugs

<sup>&</sup>lt;sup>49</sup> https://calibre-ebook.com/download

<sup>&</sup>lt;sup>50</sup> https://bugs.launchpad.net/calibre

### 9.2.5 How do I use calibre with my iPad/iPhone/iPod touch?

An easy way to browse your calibre collection from your Apple device is by using *The calibre Content server* (page 111), which makes your collection available over the net. First perform the following steps in calibre

- Set the Preferred Output Format in calibre to EPUB (The output format can be set under *Preferences* → *Interface* → *Behavior*)
- Set the output profile to iPad (this will work for iPhone/iPods as well), under *Preferences* → *Conversion* → *Common* options → *Page setup*
- Convert the books you want to read on your iDevice to EPUB format by selecting them and clicking the *Convert* button.
- Turn on the Content server by clicking the *Connect/share* button and leave calibre running. You can also tell calibre to automatically start the Content server via *Preferences* → *Sharing* → *Sharing* over the net.

The Content server allows you to read books directly in Safari itself. In addition, there are many apps for your iDevice that can connect to the calibre Content server. Examples include: Marvin, Mapleread and iBooks itself.

#### Using the Content server

Start the Safari browser and type in the IP address and port of the computer running the calibre server, like this:

http://192.168.1.2:8080/

Replace 192.168.1.2 with the local IP address of the computer running calibre. See *The calibre Content server* (page 111) for details on running the server and finding out the right IP address to use.

You will see a list of books in Safari, tap on any book and you will be given the option to either download it, or read it in the browser itself. If you choose to download it, Safari will ask you if you want to open it with iBooks.

Many reading apps support browsing the calibre library directly via its OPDS support. In such apps you can go to the online catalog screen and add the IP address of the calibre server to browse and download books from your calibre library within the app.

### 9.2.6 How do I use calibre with my Android phone/tablet or Kindle Fire?

There are two ways that you can connect your Android device to calibre. Using a USB cable – or wirelessly, over the air. The first step to using an Android device is installing an e-book reading application on it. There are many free and paid e-book reading applications for Android: Some examples (in no particular order): FBReader<sup>51</sup>, Moon+<sup>52</sup>, Mantano<sup>53</sup>, Aldiko<sup>54</sup>, Kindle<sup>55</sup>.

### Using a USB cable

Simply plug your device into the computer with a USB cable. calibre should automatically detect the device and then you can transfer books to it by clicking the *Send to device* button. Note that on macOS and Linux only a single program can connect to an Android device at a time, so make sure the device is not opened in the OS File manager, or the Android File Transfer utility, etc.

<sup>&</sup>lt;sup>51</sup> https://play.google.com/store/apps/details?id=org.geometerplus.zlibrary.ui.android&hl=en

<sup>&</sup>lt;sup>52</sup> https://play.google.com/store/apps/details?id=com.flyersoft.moonreader&hl=en

<sup>53</sup> https://play.google.com/store/apps/details?id=com.mantano.reader.android.lite&hl=en

<sup>&</sup>lt;sup>54</sup> https://play.google.com/store/apps/details?id=com.aldiko.android&hl=en

<sup>&</sup>lt;sup>55</sup> https://play.google.com/store/apps/details?id=com.amazon.kindle&feature=related\_apps

#### 1 Note

With newer Android devices, you might have to jump through a few hoops to get the connection working, as Google really does not want you to be independent of its cloud. First, unlock the screen before plugging in the USB cable. When you plugin in the USB cable you will get a popup notification. Make sure it says some thing like "Transferring Media files" or "MTP (Media Transfer mode)". If it does not, tap the notification, and change the mode to Media Transfer (MTP). You may need to restart calibre at this point in order for your device to be recognized. Finally, you might get a popup on the device every time calibre or the operating system actually tries to connect to it, asking for permission, tap OK.

### 1 Note

With the Kindle Fire 8 or newer there is an icon that shows up when the USB cable is plugged in, showing that the device is charging. Tap that and switch the device to data transfer mode, and then start calibre, it should then be detected.

### Over the air

calibre has a builtin web server, the *calibre Content server* (page 111). It makes your calibre collection available over the net. You can browse it on your device using a simple browser or a dedicated application. First perform the following steps in calibre:

- Set the *Preferred Output Format* in calibre to EPUB for normal Android devices or MOBI for Kindles (The output format can be set under *Preferences* → *Interface* → *Behavior*)
- Convert the books you want to read on your device to EPUB/MOBI format by selecting them and clicking the *Convert* button.
- Turn on the *Content server* in calibre's preferences and leave calibre running.

Now on your Android device, open the browser and browse to

### http://192.168.1.2:8080/

Replace 192.168.1.2 with the local IP address of the computer running calibre. See *The calibre Content server* (page 111) for details on running the server and finding out the right IP address to use.

You can now browse your book collection and download books from calibre to your device to open with whatever e-book reading software you have on your Android device.

Many reading apps support browsing the calibre library directly via its OPDS support. In such apps you can go to the online catalog screen and add the IP address of the calibre server to browse and download books from your calibre library within the app.

# 9.2.7 Can I access my calibre books using the web browser in my Kindle or other reading device?

calibre has a *Content server* that exports the books in calibre as a web page. See *The calibre Content server* (page 111) for details.

Some devices, like the Kindle (1/2/DX), do not allow you to access port 8080 (the default port on which the content server runs). In that case, change the port in the calibre Preferences to 80. (On some operating systems, you may not be able to run the server on a port number less than 1024 because of security settings. In this case the simplest solution is to adjust your router to forward requests on port 80 to port 8080).

Also some devices do not have browsers advanced enough to run the app-like interface used by the Content server. For such devices, you can simply add /mobile to the server URL to get a simplified, non-JavaScript interface.

### 9.2.8 I cannot send emails using calibre?

Because of the large amount of spam in email, sending email can be tricky, as different mail servers use different strategies to block email. The most common problem is if you are sending email directly (without a mail relay) in calibre. Many servers (for example, Amazon) block email that does not come from a well known relay. The most robust way to setup email sending in calibre is to do the following:

- Create a free GMX account at GMX<sup>56</sup>.
- Go to *Preferences* → *Sharing* → *Sharing books by email* in calibre and click the *Use GMX* button and fill in the information asked for.
- Log into your GMX account on the website and enable SMTP sending (*Settings->POP3 & IMAP->Send and receive emails via external program*)
- calibre will then be able to use GMX to send the mail.
- If you are sending to your Kindle, remember to update the email preferences on your Amazon Kindle page to allow email sent from your GMX email address. Also note that Amazon does not allow email delivery of AZW3 and new style (KF8) MOBI files. Finally, Amazon recently started sending confirmation emails that you have to click on back to your GMX account before the book is actually delivered.

Even after doing this, you may have problems. One common source of problems is that some poorly designed antivirus programs block calibre from opening a connection to send email. Try adding an exclusion for calibre in your antivirus program.

### Note

Microsoft/GMX can disable your account if you use it to send large amounts of email. So, when using these services to send mail calibre automatically restricts itself to sending one book every five minutes. If you don't mind risking your account being blocked you can reduce this wait interval by going to *Preferences*  $\rightarrow$  *Advanced*  $\rightarrow$  *Tweaks* in calibre.

### Note

Google recently deliberately broke their email sending protocol (SMTP) support in an attempt to force everyone to use their web interface so they can show you more ads. They are trying to claim that SMTP is insecure, that is incorrect and simply an excuse. Use some other email provider instead.

#### \rm 1 Note

If you are concerned about giving calibre access to your email account, simply create a new free email account with GMX or Outlook and use it only for calibre.

# 9.2.9 My device is getting mounted read-only in Linux, so calibre cannot connect to it?

Linux kernels mount devices read-only when their filesystems have errors. You can repair the filesystem with:

<sup>56</sup> https://www.gmx.com

```
sudo fsck.vfat -y /dev/sdc
```

Replace /dev/sdc with the path to the device node of your device. You can find the device node of your device, which will always be under /dev by examining the output of:

mount

# 9.2.10 Why does calibre not support collections on the Kindle or shelves on the Nook?

Neither the Kindle nor the Nook provide any way to manipulate collections over a USB connection. If you really care about using collections, I would urge you to sell your Kindle/Nook and get a Kobo. Only Kobo seems to understand that life is too short to be entering collections one by one on an e-ink screen [?]

Note that in the case of the Kindle, there is a way to manipulate collections via USB, but it requires that the Kindle be rebooted *every time* it is disconnected from the computer, for the changes to the collections to be recognized. As such, it is unlikely that any calibre developers will ever feel motivated enough to support it. There is however, a calibre plugin that allows you to create collections on your Kindle from the calibre metadata. It is available from here<sup>57</sup>.

#### Note

Amazon have removed the ability to manipulate collections completely in their newer models, like the Kindle Touch and Kindle Fire, making even the above plugin useless, unless you root your Kindle and install custom firmware.

### 9.2.11 I am getting an error when I try to use calibre with my Kobo Touch/Glo/etc.?

The Kobo has very buggy firmware. Connecting to it has been known to fail at random. Certain combinations of motherboard, USB ports/cables/hubs can exacerbate this tendency to fail. If you are getting an error when connecting to your touch with calibre try the following, each of which has solved the problem for *some* calibre users.

- · Connect the Kobo directly to your computer, not via USB Hub
- Try a different USB cable and a different USB port on your computer
- Log out of the Kobo and log in again, this causes it to rebuild the database, fixing corrupted database errors.
- Try upgrading the firmware on your Kobo Touch to the latest
- Try resetting the Kobo (sometimes this cures the problem for a little while, but then it re-appears, in which case you have to reset again and again)
- Try only putting one or two books onto the Kobo at a time and do not keep large collections on the Kobo

# 9.2.12 Covers for books I send to my e-ink Kindle show up momentarily and then are replaced by a generic cover?

This happens because of an Amazon bug. They try to download a cover for the book from their servers and when that fails, they replace the existing cover that calibre created with a generic cover. For details see this forum thread<sup>58</sup>. As of version 4.17, calibre has a workaround, where if you connect the Kindle to calibre after the covers have been destroyed by Amazon, calibre will restore them automatically. So in order to see the covers on your Kindle, you have to:

- 1) Send the book to the Kindle with calibre
- 2) Disconnect the Kindle and wait for Amazon to destroy the cover

<sup>&</sup>lt;sup>57</sup> https://www.mobileread.com/forums/showthread.php?t=244202

<sup>&</sup>lt;sup>58</sup> https://www.mobileread.com/forums/showthread.php?t=329945

### 3) Reconnect the Kindle to calibre

Note that this workaround only works for books sent with calibre 4.17 or later. Alternately, simply keep your Kindle in airplane mode, you don't really want Amazon knowing every book you read anyway. I encourage you to contact Amazon customer support and complain loudly about this bug. Maybe Amazon will listen.

### 1 Note

If the workaround is not working for you make sure you Kindle firmware is at least version 5.12.5, released in April 2020.

# 9.2.13 Covers for books sent to my Kindle ColorSoft and newer do not show up in the book list?

Amazon deliberately broke this functionality in their ColorSoft and newer devices in order to discourage you from reading non Amazon books on their devices. See this forum thread<sup>59</sup> for details. The only known workaround is to send the books as "Personal documents" to the Kindle which will fix the covers not showing up but break other features such as Whispersync and the books will show up under "Personal documents" not "Books" on the device. To enable this in calibre go to *Preferences*  $\rightarrow$  *Output options*  $\rightarrow$  *MOBI output* and enable the check box that says *Enable sharing of book content*. This will cause all future books sent to the Kindle by calibre to be marked as personal documents.

# 9.2.14 The covers for my MOBI files have stopped showing up in Kindle for PC/Kindle for Android/iPad etc.

This is caused by a bug in the Amazon software. You can work around it by going to *Preferences*  $\rightarrow$  *Conversion*  $\rightarrow$  *Output Options*  $\rightarrow$  *MOBI output* and setting the *Enable sharing of book content* option. If you are reconverting a previously converted book, you will also have to enable the option in the conversion dialog for that individual book (as per book conversion settings are saved and take precedence).

Note that doing this will mean that the generated MOBI will show up under personal documents instead of Books on the Kindle Fire and Amazon whispersync will not work, but the covers will. It's your choice which functionality is more important to you. I encourage you to contact Amazon and ask them to fix this bug.

The bug in Amazon's software is that when you put a MOBI file on a Kindle, unless the file is marked as a Personal document, Amazon assumes you bought the book from it and tries to download the cover thumbnail for it from its servers. When the download fails, it refuses to fallback to the cover defined in the MOBI file. This is likely deliberate on Amazon's part to try to force authors to sell only through them. In other words, the Kindle only displays covers for books marked as Personal Documents or books bought directly from Amazon.

If you send a MOBI file to an e-ink Kindle with calibre using a USB connection, calibre works around this Amazon bug by uploading a cover thumbnail itself. However, that workaround is only possible when using a USB connection and sending with calibre. Note that if you send using email, Amazon will automatically mark the MOBI file as a Personal Document and the cover will work, but the book will show up in Personal Documents.

# 9.2.15 I transferred some books to my Kindle using calibre and they did not show up?

Books sent to the Kindle only show up on the Kindle after they have been *indexed* by the Kindle. This can take some time. If the book still does not show up after some time, then it is likely that the Kindle indexer crashed. Sometimes a particular book can cause the indexer to crash. Unfortunately, Amazon has not provided any way to deduce which book is causing a crash on the Kindle. Your only recourse is to either reset the Kindle, or delete all files from its memory using Windows Explorer (or whatever file manager you use) and then send the books to it again, one by one, until you discover

<sup>&</sup>lt;sup>59</sup> https://www.mobileread.com/forums/showthread.php?t=364350

the problem book. Once you have found the problem book, delete it off the Kindle and do a MOBI to MOBI or MOBI to AZW3 conversion in calibre and then send it back. This will most likely take care of the problem.

# 9.3 Library management

### Contents

- Where are the book files stored? (page 137)
- How does calibre manage author names and sorting? (page 137)
- Why doesn't calibre let me store books in my own folder structure? (page 139)
- Why doesn't calibre have a column for foo? (page 139)
- Can I have a column showing the formats or the ISBN? (page 139)
- How do I move my calibre data from one computer to another? (page 140)
- The list of books in calibre is blank! (page 140)
- I am getting errors with my calibre library on a networked drive/NAS? (page 141)

### 9.3.1 Where are the book files stored?

When you first run calibre, it will ask you for a folder in which to store your books. Whenever you add a book to calibre, it will copy the book into that folder. Books in the folder are nicely arranged into sub-folders by Author and Title. Note that the contents of this folder are automatically managed by calibre, **do not** add any files/folders manually to this folder, as they may be automatically deleted. If you want to add a file associated to a particular book, use the top right area of *Edit metadata* dialog to do so. Then, calibre will automatically put that file into the correct folder and move it around when the title/author changes.

Metadata about the books is stored in the file metadata.db at the top level of the library folder. This file is a sqlite database. When backing up your library make sure you copy the entire folder and all its sub-folders.

The library folder and all its contents make up what is called a calibre library. You can have multiple such libraries. To manage the libraries, click the calibre icon on the toolbar. You can create new libraries, remove/rename existing ones and switch between libraries easily.

You can copy or move books between different libraries (once you have more than one library setup) by right clicking on a book and selecting the *Copy to library* action.

### 9.3.2 How does calibre manage author names and sorting?

Author names are complex, especially across cultures, see this note<sup>60</sup> for some of the complexities. calibre has a very flexible strategy for managing author names. The first thing to understand is that books and authors are separate entities in calibre. A book can have more than one author, and an author can have more than one book. You can manage the authors of a book by the edit metadata dialog. You can manage individual authors by right clicking on the author in the Tag browser on the left of the main calibre window and selecting *Manage authors*. Using this dialog you can change the name of an author and also how that name is sorted. This will automatically change the name of the author in all the books of that author. When a book has multiple authors, separate their names using the & character.

Now coming to author name sorting:

<sup>&</sup>lt;sup>60</sup> https://www.w3.org/International/questions/qa-personal-names.en.php?changelang=en

- When a new author is added to calibre (this happens whenever a book by a new author is added), calibre automatically computes a sort string for both the book and the author.
- Authors in the Tag browser are sorted by the sort value for the **authors**. Remember that this is different from the Author sort field for a book.
- By default, this sort algorithm assumes that the author name is in First name Last name format and generates a Last name, First name sort value.
- You can change this algorithm by going to *Preferences* → *Advanced* → *Tweaks* and setting the *au*-*thor\_sort\_copy\_method* tweak.
- You can force calibre to recalculate the author sort values for every author by right clicking on any author and selecting *Manage authors*, then pushing the *Recalculate all author sort values* button. Do this after you have set the author\_sort\_copy\_method tweak to what you want.
- You can force calibre to recalculate the author sort values for all books by using the bulk metadata edit dialog (select all books and click edit metadata, check the *Automatically set author sort* checkbox, then press OK).
- When recalculating the author sort values for books, calibre uses the author sort values for each individual author. Therefore, ensure that the individual author sort values are correct before recalculating the books' author sort values.
- You can control whether the Tag browser display authors using their names or their sort values by setting the *categories\_use\_field\_for\_author\_name* tweak in *Preferences* → *Advanced* → *Tweaks*

Note that you can set an individual author's sort value to whatever you want using *Manage authors*. This is useful when dealing with names that calibre will not get right, such as complex multi-part names like Miguel de Cervantes Saavedra or when dealing with Asian names like Sun Tzu.

With all this flexibility, it is possible to have calibre manage your author names however you like. For example, one common request is to have calibre display author names LN, FN. To do this, and if the note below does not apply to you, then:

- Set the author\_sort\_copy\_method tweak to copy as described above.
- Restart calibre. Do not change any book metadata before doing the remaining steps.
- Change all author names to LN, FN using the Manage authors dialog.
- After you have changed all the authors, press the Recalculate all author sort values button.
- Press OK, at which point calibre will change the authors in all your books. This can take a while.

#### 1 Note

When changing from FN LN to LN, FN, it is often the case that the values in author\_sort are already in LN, FN format. If this is your case, then do the following:

- Set the author\_sort\_copy\_method tweak to copy as described above.
- Restart calibre. Do not change any book metadata before doing the remaining steps.
- Open the Manage authors dialog. Press the copy all author sort values to author button.
- Check through the authors to be sure you are happy. You can still press Cancel to abandon the changes. Once you press OK, there is no undo.
- Press OK, at which point calibre will change the authors in all your books. This can take a while.

### 9.3.3 Why doesn't calibre let me store books in my own folder structure?

The whole point of calibre's library management features is that they provide a search and sort based interface for locating books that is *much* more efficient than any possible folder scheme you could come up with for your collection. Indeed, once you become comfortable using calibre's interface to find, sort and browse your collection, you won't ever feel the need to hunt through the files on your disk to find a book again. By managing books in its own folder structure of Author -> Title -> Book files, calibre is able to achieve a high level of reliability and standardization. To illustrate why a search/tagging based interface is superior to folders, consider the following. Suppose your book collection is nicely sorted into folders with the following scheme:

Genre -> Author -> Series -> ReadStatus

Now this makes it very easy to find for example all science fiction books by Isaac Asimov in the Foundation series. But suppose you want to find all unread science fiction books. There's no easy way to do this with this folder scheme, you would instead need a folder scheme that looks like:

ReadStatus -> Genre -> Author -> Series

In calibre, you would instead use tags to mark genre and read status and then just use a simple search query like tag:scifi and not tag:read. calibre even has a nice graphical interface, so you don't need to learn its search language instead you can just click on tags to include or exclude them from the search.

To those of you that claim that you need access to the filesystem, so that you can have access to your books over the network, calibre has an excellent Content server that gives you access to your calibre library over the net.

If you are worried that someday calibre will cease to be developed, leaving all your books marooned in its folder structure, explore the powerful *Save to disk* feature in calibre that lets you export all your files into a folder structure of arbitrary complexity based on their metadata.

Finally, the reason there are numbers at the end of every title folder, is for *robustness*. That number is the id number of the book record in the calibre database. The presence of the number allows you to have multiple records with the same title and author names. It is also part of what allows calibre to magically regenerate the database with all metadata if the database file gets corrupted. Given that calibre's mission is to get you to stop storing metadata in filenames and stop using the filesystem to find things, the increased robustness afforded by the id numbers is well worth the uglier folder names.

If you are still not convinced, then I'm afraid calibre is not for you. Look elsewhere for your book cataloguing needs. Just so we're clear, **this is not going to change**. Kindly do not contact us in an attempt to get us to change this.

### 9.3.4 Why doesn't calibre have a column for foo?

calibre is designed to have columns for the most frequently and widely used fields. In addition, you can add any columns you like. Columns can be added via *Preferences*  $\rightarrow$  *Interface*  $\rightarrow$  *Add your own columns*. Watch the tutorial UI Power tips<sup>61</sup> to learn how to create your own columns, or read this blog post<sup>62</sup>.

You can also create "virtual columns" that contain combinations of the metadata from other columns. In the add column dialog use the *Quick create* links to easily create columns to show the book ISBN or formats. You can use the powerful calibre template language to do much more with columns. For more details, see *The calibre template language* (page 158).

# 9.3.5 Can I have a column showing the formats or the ISBN?

Yes, you can. Follow the instructions in the answer above for adding custom columns.

<sup>&</sup>lt;sup>61</sup> https://calibre-ebook.com/demo#tutorials

<sup>62</sup> https://blog.calibre-ebook.com/calibre-custom-columns/

# 9.3.6 How do I move my calibre data from one computer to another?

You can export all calibre data (books, settings and plugins) and then import it on another computer. First let's see how to export the data:

- Right click the calibre icon in the main calibre toolbar and select *Export/import all calibre data*. Note that if there is currently a device connected, this menu option will not be available so, disconnect any connected devices. Then click the button labelled *Export all your calibre data*. You will see a list of all your calibre libraries. Click OK and choose an empty folder somewhere on your computer. The exported data will be saved in this folder. Simply copy this folder to your new computer and follow the instructions below to import the data.
- Install calibre on your new computer and run through the *Welcome wizard*, it does not matter what you do there, as you will be importing your old settings in the next step. You will now have an empty calibre, with just the *Getting Started* guide in your library. Once again, right click the calibre button and choose *Export/import all calibre data*. Then click the button labelled *Import previously exported data*. Select the folder with the exported data that you copied over earlier. You will now have a list of libraries you can import. Go through the list one by one, and select the new location for each library (a location is just an empty folder somewhere on your computer). Click OK. After the import completes, calibre will restart, with all your old libraries, settings and calibre plugins.

### **1** Note

This import/export functionality is only available from calibre version 2.47 onwards. If you have an older version of calibre, or if you encounter problems with the import/export, you can just copy over your calibre library folder manually, as described in the next paragraph.

Simply copy the calibre library folder from the old to the new computer. You can find out what the library folder is by clicking the calibre icon in the toolbar. Choose the *Switch/create calibre library* action and you will see the path to the current calibre library.

Now on the new computer, start calibre for the first time. It will run the *Welcome wizard* asking you for the location of the calibre library. Point it to the previously copied folder. If the computer you are transferring to already has a calibre installation, then the *Welcome wizard* won't run. In that case, right-click the calibre icon in the toolbar and point it to the newly copied folder. You will now have two calibre libraries on your computer and you can switch between them by clicking the calibre icon on the toolbar. Transferring your library in this manner preserves all your metadata, tags, custom columns, etc.

# 9.3.7 The list of books in calibre is blank!

In order to understand why that happened, you have to understand what a calibre library is. At the most basic level, a calibre library is just a folder. Whenever you add a book to calibre, that book's files are copied into this folder (arranged into sub folders by author and title). Inside the calibre library folder, at the top level, you will see a file called metadata.db. This file is where calibre stores the metadata like title/author/rating/tags etc. for *every* book in your calibre library. The list of books that calibre displays is created by reading the contents of this metadata.db file.

There can be two reasons why calibre is showing a empty list of books:

- Your calibre library folder changed its location. This can happen if it was on an external disk and the drive letter for that disk changed. Or if you accidentally moved the folder. In this case, calibre cannot find its library and so starts up with an empty library instead. To remedy this, do a right-click on the calibre icon in the calibre toolbar and select Switch/create library. Click the little blue icon to select the new location of your calibre library and click OK. If you don't know the new location search your computer for the file metadata.db.
- Your metadata.db file was deleted/corrupted. In this case, you can ask calibre to rebuild the metadata.db from its backups. Right click the calibre icon in the calibre toolbar and select Library maintenance->Restore database. calibre will automatically rebuild metadata.db.

## 9.3.8 I am getting errors with my calibre library on a networked drive/NAS?

#### Do not put your calibre library on a networked drive.

A filesystem is a complex beast. Most network filesystems lack various filesystem features that calibre uses. Some don't support file locking, some don't support hardlinking, some are just flaky. Additionally, calibre is a single user application, if you accidentally run two copies of calibre on the same networked library, bad things will happen. Finally, different OSes impose different limitations on filesystems, so if you share your networked drive across OSes, once again, bad things *will happen*.

Consider using the calibre Content server to make your books available on other computers. Run calibre on a single computer and access it via the Content server or a Remote Desktop solution.

If you must share the actual library, use a file syncing tool like DropBox or rsync instead of a networked drive. If you are using a file-syncing tool it is **essential** that you make sure that both calibre and the file syncing tool do not try to access the calibre library at the same time. In other words, **do not** run the file syncing tool and calibre at the same time.

Even with these tools there is danger of data corruption/loss, so only do this if you are willing to live with that risk. In particular, be aware that **Google Drive** is incompatible with calibre, if you put your calibre library in Google Drive, **you will suffer data loss**. See this thread<sup>63</sup> for details.

## 9.4 Miscellaneous

#### Contents

- Amazon is stopping email delivery of MOBI files? (page 142)
- I want calibre to download news from my favorite news website. (page 142)
- Why the name calibre? (page 142)
- Why does calibre show only some of my fonts on macOS? (page 143)
- *calibre is not starting on Windows?* (page 143)
- calibre freezes/crashes occasionally? (page 143)
- The calibre E-book viewer and Edit book tools do not work on Windows? (page 144)
- Using the viewer or doing any conversions results in a permission denied error on Windows (page 144)
- *calibre is not starting/crashing on macOS?* (page 145)
- I get only a black or white screen when running the calibre E-book viewer? (page 145)
- I downloaded the installer, but it is not working? (page 145)
- My antivirus program claims calibre is a virus/trojan? (page 146)
- How do I backup calibre? (page 146)
- How do I use purchased EPUB books with calibre (or what do I do with .acsm files)? (page 146)
- *I am getting a "Permission Denied" error?* (page 146)
- Can I have the comment metadata show up on my reader? (page 147)
- *How do I get calibre to use my HTTP proxy?* (page 147)
- *I want some feature added to calibre. What can I do?* (page 147)

<sup>63</sup> https://www.mobileread.com/forums/showthread.php?t=205581

- Why doesn't calibre have an automatic update? (page 147)
- *How is calibre licensed?* (page 148)
- How do I run calibre from my USB stick? (page 148)
- How do I run parts of calibre like news download and the Content server on my own Linux server? (page 148)

## 9.4.1 Amazon is stopping email delivery of MOBI files?

Amazon have announced<sup>64</sup> that they will stop accepting MOBI files emailed to <code>@kindle.com</code> email addresses. You can instruct calibre to send EPUB instead of MOBI by going to *Preferences*  $\rightarrow$  *Sharing books by email* and then removing MOBI from the list of formats to send to your <code>@kindle.com</code> email address and adding EPUB instead.

Note however, that Amazon's EPUB intake is very flawed, they will reject a number of EPUB files that work everywhere else. In such cases you can try the following trick:

- 1. Convert the EPUB file to MOBI
- 2. Then convert the MOBI file back to EPUB and send the resulting EPUB file

This will remove all advanced formatting, embedded fonts, etc., but greatly increase the chances of Amazon accepting the EPUB.

#### **1** Note

If you were previously using email delivery of periodicals downloaded by calibre, you will be better off sending those by USB cable or downloading them from the calibre Content server via the Kindle's built-in browser. However, if you want to continue using email delivery you can try changing the output format in Preferences->Behavior to EPUB, then calibre will download the news in EPUB format. Whether Amazon will accept the EPUB or not is a whole other question.

## 9.4.2 I want calibre to download news from my favorite news website.

If you are reasonably proficient with computers, you can teach calibre to download news from any website of your choosing. To learn how to do this see *Adding your favorite news website* (page 33).

Otherwise, you can request a particular news site by posting in the calibre Recipes forum<sup>65</sup>.

## 9.4.3 Why the name calibre?

#### Take your pick:

- Converter And LIBRary for E-books
- A high calibre product
- A tribute to the SONY Librie which was the first e-ink based e-book reader
- My wife chose it ;-)

calibre is pronounced as cal-i-ber *not* ca-li-bre. If you're wondering, calibre is the British/commonwealth spelling for caliber. Being Indian, that's the natural spelling for me.

<sup>&</sup>lt;sup>64</sup> https://blog.the-ebook-reader.com/2022/05/03/amazon-dropping-mobi-support-on-send-to-kindle-apps/

<sup>65</sup> https://www.mobileread.com/forums/forumdisplay.php?f=228

## 9.4.4 Why does calibre show only some of my fonts on macOS?

calibre embeds fonts in e-book files it creates. E-book files support embedding only TrueType and OpenType (.ttf and .otf) fonts. Most fonts on macOS systems are in .dfont format, thus they cannot be embedded. calibre shows only TrueType and OpenType fonts found on your system. You can obtain many such fonts on the web. Simply download the .ttf/.otf files and add them to the Library/Fonts folder in your home folder.

## 9.4.5 calibre is not starting on Windows?

There can be several causes for this:

- If you get no errors but the calibre window does not appear, it has probably just appeared off screen. You can gather all windows onto the current screen using one of the techniques described here<sup>66</sup>.
- Some software has been known to interfere with calibre, try rebooting in Safe mode and see if it works. A known culprit is the Sunshine<sup>67</sup> screen sharing software.
- If you get an error about calibre not being able to open a file because it is in use by another program, do the following:
  - Uninstall calibre
  - Reboot your computer
  - Re-install calibre. But do not start calibre from the installation wizard.
  - Temporarily disable your antivirus program (disconnect from the Internet before doing so, to be safe)
  - Look inside the folder you chose for your calibre library. If you see a file named metadata.db, delete it.
  - Start calibre
  - From now on you should be able to start calibre normally.
- If you get an error about a Python function terminating unexpectedly after upgrading calibre, first uninstall calibre, then delete the folders (if they exists) C:\Program Files\Calibre and C:\Program Files\Calibre2. Now re-install and you should be fine.
- If you get an error in the *Welcome wizard* on an initial run of calibre, try choosing a folder like C: \library as the calibre library (calibre sometimes has trouble with library locations if the path contains non-English characters, or only numbers, etc.)
- Try running it as administrator (Right click on the icon and select Run as administrator)

If it still won't launch, start a command prompt (press the Windows key and R; then type cmd.exe in the Run dialog that appears). At the command prompt type the following command and press Enter:

calibre-debug -g

Post any output you see in a help message on the Forum<sup>68</sup>.

## 9.4.6 calibre freezes/crashes occasionally?

There are several possible things I know of, that can cause this:

You recently connected an external monitor or TV to your computer. In this case, whenever calibre opens a new
window like the edit metadata window or the conversion dialog, it appears on the second monitor where you don't
notice it and so you think calibre has frozen. Disconnect your second monitor and restart calibre.

<sup>66</sup> https://www.wikihow.com/Bring-an-Off-Screen-Window-Back-on-Windows

<sup>&</sup>lt;sup>67</sup> https://github.com/LizardByte/Sunshine

<sup>&</sup>lt;sup>68</sup> https://www.mobileread.com/forums/forumdisplay.php?f=166

- The following programs have been reported to cause crashes in calibre: If you are running any of these, close them before starting calibre, or uninstall them: *RoboForm, Logitech SetPoint Settings, Constant Guard Protection by Xfinity, Spybot, Killer Network Manager, Nahimic UI Interface, Acronis True Image.*
- You are using a Wacom branded USB mouse/tablet. There is an incompatibility between Wacom drivers and the graphics toolkit calibre uses. Try using a non-Wacom mouse.
- On some 64 bit versions of Windows there are security software/settings that prevent 64-bit calibre from working properly. If you are using the 64-bit version of calibre try switching to the 32-bit version.
- If the crash happens when you are trying to copy text from the calibre E-book viewer, it is most likely caused by some clipboard monitoring/managing application you have running. Turn it off and you should be fine.
- If the crashes happen specifically when you are using a file dialog, like clicking on the *Add books* button or the *Save to Disk* button, then you have some software that has installed broken Shell extensions on your computer. Known culprits include: *SpiderOak*, *odrive sync* and *Dell Backup and Recovery* and *NetDrive*. If you have one of these, uninstall them and you will be fine. You can also use the NirSoft Shell Extension Viewer<sup>69</sup> to see what shell extensions are installed on your system and disable them individually, if you don't want to uninstall the full program. Remember to use "Restart Explorer" or reboot your computer after disabling the shell extensions.

If none of the above apply to you, then there is some other program on your computer that is interfering with calibre. First reboot your computer in safe mode, to have as few running programs as possible, and see if the crashes still happen. If they do not, then you know it is some program causing the problem. The most likely such culprit is a program that modifies other programs' behavior, such as an antivirus, a device driver, something like RoboForm (an automatic form filling app) or an assistive technology like Voice Control or a Screen Reader.

The only way to find the culprit is to eliminate the programs one by one and see which one is causing the issue. Basically, stop a program, run calibre, check for crashes. If they still happen, stop another program and repeat.

## 9.4.7 The calibre E-book viewer and Edit book tools do not work on Windows?

These two programs use hardware acceleration as they embed a version of the Chrome browser to render HTML. If they do not work it will be because of incompatibility with your system's GPU (graphics) drivers. Try updating these first, and reboot. If that does not fix it, you can set the QTWEBENGINE\_CHROMIUM\_FLAGS environment variable to the value --disable-gpu to turn off hardware acceleration. See this page<sup>70</sup> for details.

# 9.4.8 Using the viewer or doing any conversions results in a permission denied error on Windows

Something on your computer is preventing calibre from accessing its own temporary files. Most likely the permissions on your Temp folder are incorrect. Go to the folder file:*C:\Users\USERNAME\AppData\Local* in Windows Explorer and then right click on the file:*Temp* folder, select *Properties* and go to the *Security* tab. Make sure that your user account has full control for this folder.

Some users have reported that running the following command in an Administrator Command Prompt fixed their permissions. To get an Administrator Command Prompt search for cmd.exe in the start menu, then right click on the command prompt entry and select *Run as administrator*. At the command prompt type the following command and press Enter:

icacls "%appdata%\..\Local\Temp" /reset /T

Alternately, you can run calibre as Administrator, but doing so will cause some functionality, such as drag and drop to not work.

Finally, some users have reported that disabling UAC fixes the problem.

<sup>&</sup>lt;sup>69</sup> https://www.nirsoft.net/utils/shexview.html

<sup>&</sup>lt;sup>70</sup> https://doc.qt.io/qt-6/qtwebengine-debugging.html

## 9.4.9 calibre is not starting/crashing on macOS?

One common cause of failures on macOS is the use of accessibility technologies that are incompatible with the graphics toolkit calibre uses. Try turning off VoiceOver if you have it on. Also go to System Preferences->System->Universal Access and turn off the setting for enabling access for assistive devices in all the tabs. Another cause can be some third party apps that modify system behavior, such as Smart Scroll and Default Folder X.

You can obtain debug output about why calibre is not starting by running *Console.app*. Debug output will be printed to it. If the debug output contains a line that looks like:

Qt: internal: -108: Error ATSUMeasureTextImage text/qfontengine\_mac.mm

then the problem is probably a corrupted font cache. You can clear the cache by following these instructions<sup>71</sup>. If that doesn't solve it, look for a corrupted font file on your system, in ~/Library/Fonts or the like. An easy way to check for corrupted fonts in macOS is to start the "Font Book" application, select all fonts and then in the File menu, choose "Validate fonts".

### 9.4.10 I get only a black or white screen when running the calibre E-book viewer?

This will be because of an incompatibility between Qt WebEngine, which the viewer uses to render and the GPU drivers on your system. First try upgrading the GPU drivers. If that does not help, you can try turning off hardware acceleration in Qt WebEngine by setting the environment variable QTWEBENGINE\_CHROMIUM\_FLAGS to the value --disable-gpu. See *Environment variables* (page 285) for how to change environment variables.

### 9.4.11 I downloaded the installer, but it is not working?

Downloading from the Internet can sometimes result in a corrupted download. If the calibre installer you downloaded is not opening, try downloading it again. If re-downloading it does not work, download it from an alternate location<sup>72</sup>. If the installer still doesn't work, then something on your computer is preventing it from running.

- Try temporarily disabling your antivirus program (Microsoft Security Essentials, or Kaspersky or Norton or McAfee or whatever). This is most likely the culprit if the upgrade process is hanging in the middle.
- Similarly, if the installer is failing/rolling back and you have Microsoft PowerToys running, quit it.
- If you have installed to a non-standard location, try running the installer as Administrator
- Try rebooting your computer and running a registry cleaner like Wise registry cleaner<sup>73</sup>.
- Try a clean install. That is, uninstall calibre, delete C:\Program Files\Calibre2 (or wherever you previously chose to install calibre). Then re-install calibre. Note that uninstalling does not touch your books or settings.
- Try downloading the installer with an alternate browser. For example if you are using Microsoft Edge, try using Firefox or Chrome instead.
- If you get an error about a missing DLL on Windows, then most likely, the permissions on your temporary folder are incorrect. Go to the folder C:\Users\USERNAME\AppData\Local in Windows Explorer and then right click on the Temp folder and select *Properties* and go to the *Security* tab. Make sure that your user account has full control for this folder.

If you still cannot get the installer to work and you are on Windows, you can use the calibre portable install<sup>74</sup>, which does not need an installer (it is just a ZIP file).

<sup>&</sup>lt;sup>71</sup> https://typefaceapp.com/help/troubleshooting/macos-font-cache

<sup>72</sup> https://github.com/kovidgoyal/calibre/releases/latest

<sup>&</sup>lt;sup>73</sup> https://www.wisecleaner.com

<sup>&</sup>lt;sup>74</sup> https://calibre-ebook.com/download\_portable

## 9.4.12 My antivirus program claims calibre is a virus/trojan?

The first thing to check is that you are downloading calibre from the official website<sup>75</sup>. Make sure you are clicking the download links on the left, not the advertisements on the right. calibre is a very popular program and unscrupulous people try to setup websites offering it for download to fool the unwary.

If you have the official download and your antivirus program is still claiming calibre is a virus, then, your antivirus program is wrong. Antivirus programs use heuristics, patterns of code that "look suspicious" to detect viruses. It's rather like racial profiling. calibre is a completely open source product. You can actually browse the source code yourself (or hire someone to do it for you) to verify that it is not a virus. Please report the false identification to whatever company you buy your antivirus software from. If the antivirus program is preventing you from downloading/installing calibre, disable it temporarily, install calibre and then re-enable it.

## 9.4.13 How do I backup calibre?

The most important thing to backup is the calibre library folder, that contains all your books and metadata. This is the folder you chose for your calibre library when you ran calibre for the first time. You can get the path to the library folder by clicking the calibre icon on the main toolbar. You must backup this complete folder with all its files and sub-folders.

You can switch calibre to using a backed up library folder by simply clicking the calibre icon on the toolbar and choosing your backup library folder. A backed up library folder backs up your custom columns and saved searches as well as all your books and metadata.

If you want to backup the calibre configuration/plugins, you have to backup the config folder. You can find this config folder via *Preferences*  $\rightarrow$  *Miscellaneous*. Note that restoring configuration folders is not officially supported, but should work in most cases. Just copy the contents of the backup folder into the current configuration folder to restore.

# 9.4.14 How do I use purchased EPUB books with calibre (or what do I do with .acsm files)?

Most purchased EPUB books have *DRM* (page 387). This prevents calibre from opening them. You can still use calibre to store and transfer them to your e-book reader. First, you must authorize your reader on a Windows machine with Adobe Digital Editions. Once this is done, EPUB books transferred with calibre will work fine on your reader. When you purchase an epub book from a website, you will get an ".acsm" file. This file should be opened with Adobe Digital Editions, which will then download the actual ".epub" e-book. The e-book file will be stored in the folder "My Digital Editions", from where you can add it to calibre.

## 9.4.15 I am getting a "Permission Denied" error?

A permission denied error can occur because of many possible reasons, none of them having anything to do with calibre.

- You can get permission denied errors if you are using an SD card with write protect enabled.
- On macOS if you get permission errors when connecting a device to calibre, you can fix that by looking under *System Preferences > Security and Privacy > Privacy > Files and Folders.*
- If you, or some program you used changed the file permissions of the files in question to read only.
- If there is a filesystem error on the device which caused your operating system to mount the filesystem in read only mode or mark a particular file as read only pending recovery.
- If the files have their owner set to a user other than you.
- If your file is open in another program.
- If the file resides on a device, you may have reached the limit of a maximum of 256 files in the root of the device. In this case you need to reformat the device/sd card referred to in the error message with a FAT32 filesystem, or delete some files from the SD card/device memory.

<sup>75</sup> https://calibre-ebook.com/download

You will need to fix the underlying cause of the permissions error before resuming to use calibre. Read the error message carefully, see what file it points to and fix the permissions on that file or its containing folders.

## 9.4.16 Can I have the comment metadata show up on my reader?

Most readers do not support this. You should complain to the manufacturer about it and hopefully if enough people complain, things will change. In the meantime, you can insert the metadata, including comments into a "Jacket page" at the start of the e-book, by using the option to "Insert metadata as page at start of book" during conversion. The option is found in the *Structure detection* section of the conversion settings. Note that for this to have effect you have to *convert* the book. If your book is already in a format that does not need conversion, you can convert from that format to the same format.

Another alternative is to create a catalog in e-book form containing a listing of all the books in your calibre library, with their metadata. Click-and-hold the *Convert* button to access the catalog creation tool. And before you ask, no you cannot have the catalog "link directly to" books on your reader.

## 9.4.17 How do I get calibre to use my HTTP proxy?

By default, calibre uses whatever proxy settings are set in your OS. Sometimes these are incorrect, for example, on Windows if you don't use Microsoft Edge then the proxy settings may not be up to date. You can tell calibre to use a particular proxy server by setting the http\_proxy and https\_proxy environment variables. The format of the variable is: http://username:password@servername you should ask your network administrator to give you the correct value for this variable. Note that calibre only supports HTTP proxies not SOCKS proxies. You can see the current proxies used by calibre in Preferences->Miscellaneous.

## 9.4.18 I want some feature added to calibre. What can I do?

#### You have two choices:

- 1. Create a patch by hacking on calibre and send it to me for review and inclusion. See Development<sup>76</sup>.
- 2. Open a bug requesting the feature<sup>77</sup>. Remember that while you may think your feature request is extremely important/essential, calibre developers might not agree. Fortunately, calibre is open source, which means you always have the option of implementing your feature yourself, or hiring someone to do it for you. Furthermore, calibre has a comprehensive plugin architecture, so you might be able to develop your feature as a plugin, see *Writing your own plugins to extend calibre's functionality* (page 222).

## 9.4.19 Why doesn't calibre have an automatic update?

For many reasons:

- *There is no need to update every week.* If you are happy with how calibre works turn off the update notification and be on your merry way. Check back to see if you want to update once a year or so. There is a check box to turn off the update notification, on the update notification itself.
- calibre downloads currently use about 150TB of bandwidth a month<sup>78</sup>. Implementing automatic updates would greatly increase that and end up costing thousands of dollars a month, which someone has to pay.
- If I implement a dialog that downloads the update and launches it, instead of going to the website as it does now, that would save the most ardent calibre updater, *at most five clicks a week*. There are far higher priority things to do in calibre development.
- If you really, really hate downloading calibre every week but still want to be up to the latest, I encourage you to run from source, which makes updating trivial. Instructions are *available here* (page 357).

<sup>76</sup> https://calibre-ebook.com/get-involved

<sup>77</sup> https://calibre-ebook.com/bugs

<sup>&</sup>lt;sup>78</sup> https://calibre-ebook.com/dynamic/downloads

- There are third party automatic updaters for calibre made by calibre users in the calibre forum<sup>79</sup>.
- Additionally, some third-party updaters such as Norton or Avast may update software behind the user's back. If you find calibre has updated unexpectedly, check for the presence of one.

## 9.4.20 How is calibre licensed?

calibre is licensed under the GNU General Public License v3 (an open source license). This means that you are free to redistribute calibre as long as you make the source code available. So if you want to put calibre on a CD with your product, you must also put the calibre source code on the CD. The source code is available for download<sup>80</sup>. You are free to use the results of conversions from calibre however you want. You cannot use either code or libraries from calibre in your software without making your software open source. For details, see The GNU GPL v3<sup>81</sup>.

## 9.4.21 How do I run calibre from my USB stick?

A portable version of calibre is available here<sup>82</sup>.

# 9.4.22 How do I run parts of calibre like news download and the Content server on my own Linux server?

First, you must install calibre onto your Linux server. If your server is using a modern Linux distribution, you should have no problems installing calibre onto it.

#### Note

calibre needs GLIBC >= 2.31 and libstdc++ >= 6.0.28. If you have an older server, you will either need to compile these from source, or use calibre 3.48 which requires GLIBC >= 2.17 or 2.85.1 which requires GLIBC >= 2.13 or calibre 1.48 which requires only GLIBC >= 2.10. In addition, although the calibre command line utilities do not need a running X server, some of them do require the X server libraries to be installed on your system. This is because of Qt, which is used for various image processing tasks, and links against these libraries. If you get an ImportError about some Qt modules, you are likely missing some X libraries. Typical candidates are: libxcb-cursor0, libxcb-xinerama0, libeg11, libopeng10.

You can run the calibre server via the command:

/opt/calibre/calibre-server /path/to/the/library/you/want/to/share

You can download news and convert it into an e-book with the command:

/opt/calibre/ebook-convert "Title of news source.recipe" outputfile.epub

If you want to generate MOBI, use outputfile.mobi instead and use --output-profile kindle.

You can email downloaded news with the command:

/opt/calibre/calibre-smtp

I leave figuring out the exact command line as an exercise for the reader.

Finally, you can add downloaded news to the calibre library with:

<sup>79</sup> https://www.mobileread.com/forums/forumdisplay.php?f=238

<sup>&</sup>lt;sup>80</sup> https://download.calibre-ebook.com

<sup>&</sup>lt;sup>81</sup> https://www.gnu.org/licenses/gpl.html

<sup>82</sup> https://calibre-ebook.com/download\_portable

/opt/calibre/calibredb add --with-library /path/to/library outfile.epub

Remember to read the *Command Line Interface* (page 301) section of the calibre User Manual to learn more about these, and other commands.

## CHAPTER

# TUTORIALS

Here you will find tutorials to get you started using calibre's more advanced features, such as XPath and templates.

# 10.1 Managing subgroups of books, for example "genre"

Some people wish to organize the books in their library into subgroups, similar to subfolders. The most commonly provided reason is to create genre hierarchies, but there are many others. One user asked for a way to organize textbooks by subject and course number. Another wanted to keep track of gifts by subject and recipient. This tutorial will use the genre example for the rest of this post.

Before going on, please note that we are not talking about folders on the hard disk. Subgroups are not file folders. Books will not be copied anywhere. calibre's library file structure is not affected. Instead, we are presenting a way to organize and display subgroups of books within a calibre library.

- *Setup* (page 153)
- Searching (page 155)
- *Restrictions* (page 155)
- Useful template functions (page 156)

The commonly-provided requirements for subgroups such as genres are:

- A subgroup (e.g., a genre) must contain (point to) books, not categories of books. This is what distinguishes subgroups from calibre user categories.
- A book can be in multiple subgroups (genres). This distinguishes subgroups from physical file folders.
- Subgroups (genres) must form a hierarchy; subgroups can contain subgroups.

Tags give you the first two. If you tag a book with the genre then you can use the Tag browser (or search) to find the books with that genre, giving you the first. Many books can have the same tag(s), giving you the second. The problem is that tags don't satisfy the third requirement. They don't provide a hierarchy.



**17** C The calibre hierarchy feature gives you the third – the ability to see the genres in a 'tree' and the ability to easily search for books in genre or sub-genre. For example, assume that your genre structure is similar to the following:

Genre				
. History				
Japanese				
Military				
Roman				
. Mysteries				
English				
Vampire				
. Science Fiction				
Alternate History				
Military				
Space Opera				
. Thrillers				
Crime				
Horror				
etc.				

By using the hierarchy feature, you can see these genres in the Tag browser in tree form, as shown in the screen image. In this example the outermost level (Genre) is a custom column that contains the genres. Genres containing sub-genres appear with a small triangle next to them. Clicking on that triangle will open the item and show the sub-genres, as you can see with History and Science Fiction.

Clicking on a genre can search for all books with that genre or children of that genre. For example, clicking on Science Fiction can give all three of the child genres, Alternate History, Military, and Space Opera. Clicking on Alternate History will give books in that genre, ignoring those in Military and Space Opera. Of course, a book can have multiple genres. If a book has both Space Opera and Military genres, then you will see that book if you click on either genre. Searching is discussed in more detail below.

Another thing you can see from the image is that the genre Military appears twice, once under History and once under Science Fiction. Because the genres are in a hierarchy, these are two separate genres. A book can be in one, the other,

or (doubtfully in this case) both. For example, the books in Winston Churchill's "The Second World War" could be in "History.Military". David Weber's Honor Harrington books could be in "Science Fiction.Military", and for that matter also in "Science Fiction.Space Opera."

Once a genre exists, that is at least one book has that genre, you can easily apply it to other books by dragging the books from the library view onto the genre you want the books to have. You can also apply genres in the metadata editors; more on this below.

## 10.1.1 Setup

By now, your question might be "How was all of this setup?" There are three steps: 1) create the custom column, 2) tell calibre that the new column is to be treated as a hierarchy, and 3) add genres.

You create the custom column in the usual way, using Preferences -> Add your own columns. This example uses "#genre" as the lookup name and "Genre" as the column heading. It is important that the column type is set to *Comma-separated text, like tags, shown in the Tag browser*.

Create a custo	m column				
Quick create: ISBN, Formats, Modified Date, Yes/No, Tags, Series, Rating					
Lookup name	#genre				
Column heading	Genre				
Column type	Comma separated text, like tags, shown in the tag browser 💌				
	OK Cancel				

Then after restarting calibre, you must tell calibre that the column is to be treated as a hierarchy. Go to *Preferences*  $\rightarrow$  *Look* & *feel*  $\rightarrow$  *Tag browser*  $\rightarrow$  *Hierarchy and searching* and choose the new Genre column as having hierarchical items.

At the point there are no genres in the column. We are left with the last step: how to apply a genre to a book. A genre does not exist in calibre until it appears on at least one book. To learn how to apply a genre for the first time, we must go into some detail about what a genre looks like in the metadata for a book.

A hierarchy of 'things' is built by creating an item consisting of phrases separated by periods. Continuing the genre example, these items would "History.Military", "Mysteries.Vampire", "Science Fiction.Space Opera", etc. Thus to create a new genre, you pick a book that should have that genre, edit its metadata, and enter the new genre into the column you created. Continuing our example, if you want to assign a new genre "Comics" with a sub-genre "Superheroes" to a book, you would 'edit metadata' for that (comic) book, choose the Custom metadata tab, and then enter "Comics.Superheroes" as shown in the following (ignore the other custom columns):

Basic met	adata Custom metadata	
énum:		
enum2:		
Genre:	Comics.Superheroes	_
mybool:	✓ Yes	
mydate:	Undefined	

After doing the above, you see in the Tag browser:



From here on, to apply this new genre to a book (a comic book, presumably), you can either drag the book onto the genre, or add it to the book using edit metadata in exactly the same way as done above.

#### Note

Hierarchical display only works if the Tag browser is set to sort items by name. This is the default and can be checked by clicking the *Configure* button at the bottom of the Tag browser.

## 10.1.2 Searching

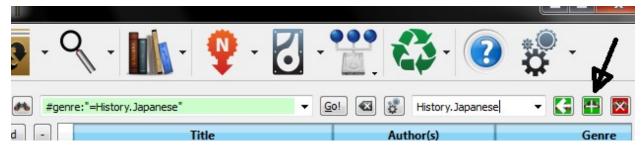


The easiest way to search for genres is using the Tag browser, clicking on the genre you wish to see. Clicking on a genre with children will show you books with that genre and all child genres. However, this might bring up a question. Just because a genre has children doesn't mean that it isn't a genre in its own right. For example, a book can have the genre "History" but not "History.Military". How do you search for books with only "History"?

The Tag browser search mechanism knows if an item has children. If it does, clicking on the item cycles through 5 searches instead of the normal three. The first is the normal green plus, which shows you books with that genre only (e.g., History). The second is a doubled plus (shown above), which shows you books with that genre and all sub-genres (e.g., History and History.Military). The third is the normal red minus, which shows you books without that exact genre. The fourth is a doubled minus, which shows you books without that genre or sub-genres. The fifth is back to the beginning, no mark, meaning no search.

## **10.1.3 Restrictions**

If you search for a genre then create a saved search for it, you can use the 'restrict to' box to create a Virtual library of books with that genre. This is useful if you want to do other searches within the genre or to manage/update metadata for books in the genre. Continuing our example, you can create a Saved search named 'History.Japanese' by first clicking on the genre Japanese in the Tag browser to get a search into the search field, entering History.Japanese into the saved search field, then pushing the "Save search" button (the green box with the white plus, on the right-hand side).



After creating the saved search, you can use it as a restriction.



## 10.1.4 Useful template functions

You might want to use the genre information in a template, such as with save to disk or send to device. The question might then be "How do I get the outermost genre name or names?" A calibre template function, subitems, is provided to make doing this easier.

For example, assume you want to add the outermost genre level to the save-to-disk template to make genre folders, as in "History/The Gathering Storm - Churchill, Winston". To do this, you must extract the first level of the hierarchy and add it to the front along with a slash to indicate that it should make a folder. The template below accomplishes this:

```
{#genre:subitems(0,1)|//}{title} - {authors}
```

See The template language (page 158) for more information about templates and the subitems () function.

## 10.2 XPath tutorial

In this tutorial, you will be given a gentle introduction to XPath<sup>83</sup>, a query language that can be used to select arbitrary parts of HTML<sup>84</sup> documents in calibre. XPath is a widely used standard, and googling it will yield a ton of information. This tutorial, however, focuses on using XPath for e-book related tasks like finding chapter headings in an unstructured HTML document.

#### Contents

- Selecting by tag name (page 157)
- *Selecting by attributes* (page 157)
- Selecting by tag content (page 157)
- Sample e-book (page 158)
- XPath built-in functions (page 158)

<sup>83</sup> https://en.wikipedia.org/wiki/XPath

<sup>84</sup> https://en.wikipedia.org/wiki/HTML

### 10.2.1 Selecting by tag name

The simplest form of selection is to select tags by name. For example, suppose you want to select all the <h2> tags in a document. The XPath query for this is simply:

//h:h2 (Selects all <h2> tags)

The prefix // means *search at any level of the document*. Now suppose you want to search for <span> tags that are inside <a> tags. That can be achieved with:

//h:a/h:span (Selects <span> tags inside <a> tags)

If you want to search for tags at a particular level in the document, change the prefix:

/h:body/h:div/h:p (Selects tags that are children of <div> tags that are children of the <body> tag)

This will match only A very short e-book to demonstrate the use of XPath. in the Sample e-book (page 158) but not any of the other tags. The h: prefix in the above examples is needed to match XHTML tags. This is because internally, calibre represents all content as XHTML. In XHTML tags have a namespace, and h: is the namespace prefix for HTML tags.

Now suppose you want to select both <h1> and <h2> tags. To do that, we need a XPath construct called *predicate*. A *predicate* is simply a test that is used to select tags. Tests can be arbitrarily powerful and as this tutorial progresses, you will see more powerful examples. A predicate is created by enclosing the test expression in square brackets:

//\*[name()='h1' or name()='h2']

There are several new features in this XPath expression. The first is the use of the wildcard \*. It means *match any tag.* Now look at the test expression name()='h1' or name()='h2'. *name()* is an example of a *built-in function*. It simply evaluates to the name of the tag. So by using it, we can select tags whose names are either h1 or h2. Note that the *name()* function ignores namespaces so that there is no need for the h: prefix. XPath has several useful built-in functions. A few more will be introduced in this tutorial.

#### 10.2.2 Selecting by attributes

To select tags based on their attributes, the use of predicates is required:

```
//*[@style] (Select all tags that have a style attribute)
//*[@class="chapter"] (Select all tags that have class="chapter")
//h:h1[@class="bookTitle"] (Select all h1 tags that have class="bookTitle")
```

Here, the @ operator refers to the attributes of the tag. You can use some of the *XPath built-in functions* (page 158) to perform more sophisticated matching on attribute values.

#### 10.2.3 Selecting by tag content

Using XPath, you can even select tags based on the text they contain. The best way to do this is to use the power of *regular expressions* via the built-in function *re:test()*:

Here the . operator refers to the contents of the tag, just as the @ operator referred to its attributes.

## 10.2.4 Sample e-book

```
<html>
   <head>
       <title>A very short e-book</title>
       <meta name="charset" value="utf-8" />
   </head>
   <body>
       <h1 class="bookTitle">A very short e-book</h1>
       Written by Kovid Goyal
       <div class="introduction">
          A very short e-book to demonstrate the use of XPath.
       </div>
       <h2 class="chapter">Chapter One</h2>
       This is a truly fascinating chapter.
       <h2 class="chapter">Chapter Two</h2>
       A worthy continuation of a fine tradition.
   </body>
</html>
```

## 10.2.5 XPath built-in functions

#### name()

The name of the current tag.

#### contains()

contains (s1, s2) returns true if s1 contains s2.

#### re:test()

re:test(src, pattern, flags) returns *true* if the string *src* matches the regular expression *pattern*. A particularly useful flag is i, it makes matching case insensitive. A good primer on the syntax for regular expressions can be found at regexp syntax<sup>85</sup>

## 10.3 The calibre template language

The calibre template language is a calibre-specific language used throughout calibre for tasks such as specifying file paths, formatting values, and computing the value for user-specified columns. Examples:

- Specify the folder structure and file names when saving files from the calibre library to the disk or e-book reader.
- Define rules for adding icons and colors to the calibre book list.
- Define virtual columns that contain data from other columns.
- Advanced library searching.
- Advanced metadata search and replace.

The language is built around the notion of a *template*, which specifies which book metadata to use, computations on that metadata, and how it is to be formatted.

<sup>85</sup> https://docs.python.org/library/re.html

### 10.3.1 Basic templates

A basic template consists one or more template expressions. A template expression consists of text and names in curly brackets ({}) that is replaced by the corresponding metadata from the book being processed. For example, the default template in calibre used for saving books to device has 4 template expressions:

{author\_sort}/{title} - {authors}

For the book "The Foundation" by "Isaac Asimov" the will become:

Asimov, Isaac/The Foundation/The Foundation - Isaac Asimov

The slashes are not template expressions because they are in between in {}. Such text is left where it appears. For example, if the template is:

{author\_sort} Some Important Text {title}/{title} - {authors}

then for "The Foundation" the template produces:

Asimov, Isaac Some Important Text The Foundation/The Foundation - Isaac Asimov

A template expression can access all the metadata available in calibre, including custom columns (columns you create yourself), by using a column's lookup name. To find the lookup name for a *column* (sometimes called *fields*), hover your mouse over the column header in calibre's book list. Lookup names for custom columns always begin with #. For series type columns there is an additional field named #lookup name\_index that is the series index for that book in the series. For example, if you have a custom series column named #myseries, there will also be a column named #myseries\_index.

In addition to the standard column based fields, you also can use:

- {formats} A list of formats available in the calibre library for a book
- {identifiers:select(isbn)} The ISBN of the book

If the metadata for the field for a given book is not defined then the field in the template is replaced by the empty string (''). For example, consider the following template:

{author\_sort}/{series}/{title} {series\_index}

If Asimov's book "Second Foundation" is in the series "Foundation" then the template produces:

Asimov, Isaac/Foundation/Second Foundation 3

If a series has not been entered for the book then the template produces:

Asimov, Isaac/Second Foundation

The template processor automatically removes multiple slashes and leading or trailing spaces.

### 10.3.2 Advanced formatting

In addition to metadata substitution, templates can conditionally include additional text and control how substituted data is formatted.

#### Conditionally including text

Sometimes you want text to appear in the output only if a field is not empty. A common case is series and series\_index where you want either nothing or the two values separated by a hyphen. calibre handles this case using a special template expression syntax.

For example and using the above Foundation example, assume you want the template to produce *Foundation - 3 - Second Foundation*. This template produces that output:

{series} - {series\_index} - {title}

However, if a book has no series the template will produce - - *the title*, which is probably not what you want. Generally, people want the result be the title without the extraneous hyphens. You can accomplish this using the following template syntax:

{field:|prefix\_text|suffix\_text}

This template expression says that if field has the value XXXX then the result will be *prefix\_textXXXXsuffix\_text*. If field is empty (has no value) then the result will be the empty string (nothing) because the prefix and suffix are ignored. The prefix and suffix can contain blanks.

#### Do not use subtemplates (`{ ... }`) or functions (see below) in the prefix or the suffix.

Using this syntax, we can solve the above no-series problem with the template:

{series]{series\_index:| - | - }{title}

The hyphens will be included only if the book has a series index, which it has only if it has a series. Continuing the Foundation example again, the template will produce *Foundation - 1 - Second Foundation*.

Notes:

- You must include the colon after the lookup name if you are using a prefix or a suffix.
- You must either use either no or both | characters. Using one, as in {field: | }, is not allowed.
- It is OK to provide no text for either the prefix or the suffix, such as in {series: || }. The template {ti-tle: ||} is the same as {title}.

#### Formatting

Suppose you want the series\_index to be formatted as three digits with leading zeros. This does the trick:

{series\_index:0>3s} - Three digits with leading zeros

For trailing zeros, use:

{series\_index:0<3s} - Three digits with trailing zeros

If you use series indices with fractional values, e.g., 1.1, you might want the decimal points to line up. For example, you might want the indices 1 and 2.5 to appear as 01.00 and 02.50 so that they will sort correctly on a device that does lexical sorting. To do this, use:

{series\_index:0>5.2f} - Five characters consisting of two digits with leading zeros, a decimal point, then 2 digits after the decimal point.

If you want only the first two letters of the data, use:

{author\_sort:.2} - Only the first two letters of the author sort name

Much of the calibre template language formatting comes from Python. For more details on the syntax of these advanced formatting operations see the Python documentation<sup>86</sup>.

#### 10.3.3 Using templates to define custom columns

Templates can be used to display information that isn't in calibre metadata, or to display metadata differently from calibre's normal format. For example, you might want to show the ISBN, a field that calibre does not display. You can accomplish this creating a custom column with the type *Column built from other columns* (hereafter called *composite columns*) and providing a template to generate the displayed text. The column will display the result of evaluating the template. For

<sup>86</sup> https://docs.python.org/3/library/string.html#formatstrings

example, to display the ISBN, create the column and enter {identifiers:select(isbn)} in the template box. To display a column containing the values of two series custom columns, separated by a comma, use {#series1:||, }{#series2}.

Composite columns can use any template option, including formatting.

Note: You cannot edit the data displayed in a composite column. Instead you edit the source columns. If you edit a composite column, for example by double-clicking it, calibre will open the template for editing, not the underlying data.

## 10.3.4 Templates and plugboards

Plugboards are used for changing the metadata written into books during send-to-device and save-to-disk operations. A plugboard permits you to specify a template to provide the data to write into the book's metadata. You can use plugboards to modify the following fields: authors, author\_sort, language, publisher, tags, title, title\_sort. This feature helps people who want to use different metadata in books on devices to solve sorting or display issues.

When you create a plugboard, you specify the format and device for which the plugboard is to be used. A special device is provided, save\_to\_disk, that is used when saving formats (as opposed to sending them to a device). Once you have chosen the format and device, you choose the metadata fields to change, providing templates to supply the new values. These templates are *connected* to their destination fields, hence the name *plugboards*. You can of course use composite columns in these templates.

Plugboards are quite flexible and can be written in Single Function Mode, Template Program Mode, General Program Mode, or Python Template mode.

When a plugboard might apply (Content server, save to disk, or send to device), calibre searches the defined plugboards to choose the correct one for the given format and device. For example, to find the appropriate plugboard for an EPUB book being sent to an ANDROID device, calibre searches the plugboards using the following search order:

- a plugboard with an exact match on format and device, e.g., EPUB and ANDROID
- a plugboard with an exact match on format and the special any device choice, e.g., EPUB and any device
- a plugboard with the special any format choice and an exact match on device, e.g., any format and ANDROID
- a plugboard with any format and any device

The tags and authors fields have special treatment, because both of these fields can hold more than one item. A book can have many tags and many authors. When you specify that one of these two fields is to be changed, the template's result is examined to see if more than one item is there. For tags, the result is cut apart wherever calibre finds a comma. For example, if the template produces the value Thriller, Horror, then the result will be two tags, Thriller and Horror. There is no way to put a comma in the middle of a tag.

The same thing happens for authors, but using a different character for the cut, a & (ampersand) instead of a comma. For example, if the template produces the value Blogs, Joe&Posts, Susan, then the book will end up with two authors, Blogs, Joe and Posts, Susan. If the template produces the value Blogs, Joe;Posts, Susan, then the book will have one author with a rather strange name.

Plugboards affect the metadata written into the book when it is saved to disk or written to the device. Plugboards do not affect the metadata used by save to disk and send to device to create the file names. Instead, file names are constructed using the templates entered on the appropriate preferences window.

## 10.3.5 Using functions in templates - Single Function Mode

Suppose you want to display the value of a field in upper case when that field is normally in title case. You can do this using *template functions*. For example, to display the title in upper case use the uppercase function, as in {ti-tle:uppercase()}. To display it in title case, use {title:titlecase()}.

Functions go into the format part of the template, after the : and before the first | or the closing  $\}$  if no prefix/suffix is used. If you have both a format and a function reference, the function comes after a second :. Functions return the value of the column specified in the template, suitably modified.

The syntax for using functions is one of:

```
{lookup_name:function(arguments)}
{lookup_name:format:function(arguments)}
{lookup_name:function(arguments)|prefix|suffix}
{lookup_name:format:function(arguments)|prefix|suffix}
```

Function names must always be followed by opening and closing parentheses. Some functions require extra values (arguments), and these go inside the parentheses. Arguments are separated by commas. Literal commas (commas as text, not argument separators) must be preceded by a backslash ( $\setminus$ ). The last (or only) argument cannot contain a textual closing parenthesis.

Functions are evaluated before format specifications and the prefix/suffix. See further down for an example of using both a format and a function.

**Important**: If you have programming experience, please note that the syntax in *Single Function Mode* is not what you expect. Strings are not quoted and spaces are significant. All arguments are considered to be constants; there are no expressions.

**Do not use subtemplates** (`{ ... }`) **as function arguments.** Instead, use *Template Program Mode* (page 169) and *General Program Mode* (page 164).

Notes on calling functions in Single Function Mode:

- When functions are used in Single Function Mode, the first parameter, value, is automatically replaced by the content of the field specified in the template. For example, when the template {title:capitalize()} is processed, the content of the title field is passed as the parameter value to the capitalize function.
- In the function documentation, the notation [something] \* means that something can be repeated zero or more times. The notation [something] + means that the something is repeated one or more times (must exist at least one time).
- Some functions use regular expressions. In the template language regular expression matching is case-insensitive.

Functions are documented in *Template function reference* (page 178). The documentation tells you what arguments the functions require and what the functions do. For example, here is the documentation of the *ifempty* (page 204) function.

• ifempty(value, text\_if\_empty) - if the value is not empty then return that value, otherwise return text\_if\_empty.

You see that the function requires two arguments, value and text\_if\_empty. However, because we are using Single Function Mode, we omit the value argument, passing only text\_if\_empty. For example, this template:

{tags:ifempty(No tags on this book)}

shows the tags for a book, if any. If it has no tags then it show No tags on this book.

The following functions are usable in Single Function Mode because their first parameter is value.

- *capitalize* (page 184) (value) returns the value with the first letter in upper case and the rest lower case.
- ceiling (page 182) (value) returns the smallest integer greater than or equal to value.
- *cmp* (page 202) (value, y, lt, eq, gt) compares value and y after converting both to numbers.
- contains (page 203) (value, pattern, text\_if\_match, text\_if\_not\_match) checks if the value is matched by the regular expression pattern
- date\_arithmetic (page 187) (value, calc\_spec, fmt) Calculate a new date from value using calc\_spec.
- *encode\_for\_url* (page 206) (value, use\_plus) returns the value encoded for use in a URL as specified by use\_plus. The value is first URL-encoded. Next, if use\_plus is 0 then spaces are replaced by '+' (plus) signs. If it is 1 then spaces are replaced by %20.

- floor (page 183) (value) returns the largest integer less than or equal to value.
- *format\_date* (page 188) (value, format\_string) format the value, which must be a date string, using the format\_string, returning a string.
- *format\_duration* (page 190) (value, template, [largest\_unit]) format the value, a number of seconds, into a string showing weeks, days, hours, minutes, and seconds. If the value is a float then it is rounded to the nearest integer.
- *format\_number* (page 191)(value, template) interprets the value as a number and formats that number using a Python formatting template such as {0:5.2f} or {0:,d} or \${0:5,.2f}.
- fractional\_part (page 183) (value) returns the part of the value after the decimal point.
- *human\_readable* (page 191) (value) expects the value to be a number and returns a string representing that number in KB, MB, GB, etc.
- *ifempty* (page 204) (value, text\_if\_empty) if the value is not empty then return that value, otherwise return text\_if\_empty.
- *language\_strings* (page 194) (value, localize) return the language names for the language codes (see here for names and codes<sup>87</sup>) passed in value.
- *list\_contains* (page 196)(value, separator, [ pattern, found\_val, ]\* not\_found\_val) interpret the value as a list of items separated by separator, checking the pattern against each item in the list.
- *list\_count* (page 197)(value, separator) interprets the value as a list of items separated by separator and returns the number of items in the list.
- *list\_count\_matching* (page 197) (value, pattern, separator) interprets value as a list of items separated by separator, returning the number of items in the list that match the regular expression pattern.
- *list\_item* (page 196)(value, index, separator) interpret the value as a list of items separated by separator, returning the 'index'th item.
- *list\_sort* (page 199) (value, direction, separator) return value sorted using a case-insensitive lexical sort.
- *lookup* (page 196)(value, [ pattern, key, ]\* else\_key) The patterns will be checked against the value in order
- lowercase (page 184) (value) returns the value in lower case.
- mod (page 183) (value, y) returns the floor of the remainder of value / y.
- rating\_to\_stars (page 191) (value, use\_half\_stars) Returns the value as string of star (\*) characters.
- re (page 204) (value, pattern, replacement) return the value after applying the regular expression.
- *re\_group* (page 204) (value, pattern [, template\_for\_group]\*) return a string made by applying the regular expression pattern to value and replacing each matched instance
- round (page 183) (value) returns the nearest integer to value.
- *select* (page 197)(value, key) interpret the value as a comma-separated list of items with each item having the form id:id\_value (the calibre identifier format).
- *shorten* (page 204)(value, left\_chars, middle\_text, right\_chars) Return a shortened version of the value
- *str\_in\_list* (page 197) (value, separator, [ string, found\_val, ] + not\_found\_val) interpret the value as a list of items separated by separator then compare string against each value in the list.

<sup>87</sup> https://www.loc.gov/standards/iso639-2/php/code\_list.php

- *subitems* (page 200) (value, start\_index, end\_index) This function breaks apart lists of tag-like hier-archical items such as genres.
- *sublist* (page 201) (value, start\_index, end\_index, separator) interpret the value as a list of items separated by separator, returning a new list made from the items from start\_index to end\_index.
- substr (page 205) (value, start, end) returns the start'th through the end'th characters of value
- *swap\_around\_articles* (page 205) (value, separator) returns the value with articles moved to the end, separated by a semicolon.
- swap\_around\_comma (page 205) (value) given a value of the form B, A, return A B.
- *switch* (page 196)(value, [patternN, valueN,]+ else\_value) for each patternN, valueN pair, checks if the value matches the regular expression patternN
- *test* (page 205) (value, text\_if\_not\_empty, text\_if\_empty) return text\_if\_not\_empty if the value is not empty, otherwise return text\_if\_empty.
- *titlecase* (page 184) (value) returns the value in title case.
- *transliterate* (page 205)(value) Return a string in a latin alphabet formed by approximating the sound of the words in value.
- uppercase (page 184) (value) returns the value in upper case.

#### Using functions and formatting in the same template

Suppose you have an integer custom column #myint that you want displayed with leading zeros, as in 003. One way to do this is to use a format of 0>3s. However, by default if a number (integer or float) equals zero then the value is displayed as the empty string so zero values will produce the empty string, not 000. If you want to see 000 values then you use both the format string and the ifempty function to change the empty value back to a zero. The template would be:

```
{ #myint:0>3s:ifempty(0) }
```

Note that you can use the prefix and suffix as well. If you want the number to appear as [003] or [000], then use the template:

{#myint:0>3s:ifempty(0) | [ | ] }

#### 10.3.6 General Program Mode

*General Program Mode* (*GPM*) replaces *template expressions* with a program written in the *template language*. The syntax of the language is defined by the following grammar:

```
program
                                                               ::= 'program:' expression_list
expression_list ::= top_expression [ ';' top_expression ]*
top_expression ::= or_expression
or_expression := and_expression [ '||' and_expression ]*
and_expression ::= not_expression [ '&&' not_expression ]*
not_expression :== [ '!' not_expression ]* | concatenate_expr
concatenate_expr::= compare_expr [ '&' compare_expr ]*
compare_expr := add_sub_expr [ compare_op add_sub_expr ]
compare_op
                                                               ::= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  ^{!}= \  \  ^{!}= \  ^{!}= \  \  ^{!}= \  \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}= \  ^{!}=
                                                                                'in' | 'inlist' | 'inlist field' |
                                                                                  '==\#' | '!=#' | '>=#' | '>#' | '<=#' | '<#'
                                                                ::= times_div_expr [ add_sub_op times_div_expr ]*
add_sub_expr
                                                               ::= '+' | '-'
add_sub_op
```

(continues on next page)

(continued from previous page)

```
times_div_expr := unary_op_expr [ times_div_op unary_op_expr ]*
times_div_op ::= '*' | '/'
unary_op_expr := [ add_sub_op unary_op_expr ]* | expression
expression := identifier | constant | function | assignment | field_reference |
                   if_expr | for_expr | break_expr | continue_expr |
                   '(' expression_list ')' | function_def
field_reference ::= '$' [ '$' ] [ '#' ] identifier
identifier := id_start [ id_rest ]*
              := letter | underscore
id_start
              ::= id_start | digit
id_rest
constant
              ::= " string " | ' string ' | number
function := identifier '(' expression_list [ ',' expression_list ]* ')'
function_def := 'def' identifier '(' top_expression [ ',' top_expression ]* ')' ':
\hookrightarrow '
                   expression_list 'fed'
assignment
              ::= identifier '=' top_expression
if_expr
               ::= 'if' condition 'then' expression_list
                   [ elif_expr ] [ 'else' expression_list ] 'fi'
condition
              := top_expression
              ::= 'elif' condition 'then' expression_list elif_expr | ''
elif_expr
for_expr
              := for_list | for_range
              ::= 'for' identifier 'in' list_expr
for_list
                  [ 'separator' separator_expr ] ':' expression_list 'rof'
for_range
             ::= 'for' identifier 'in' range_expr ':' expression_list 'rof'
range_expr
              ::= 'range' '(' [ start_expr ',' ] stop_expr
                  [ ',' step_expr [ ',' limit_expr ] ] ')'
list_expr
              := top_expression
break_expr
             ::= 'break'
continue_expr := 'continue'
separator_expr := top_expression
start_expr
             ::= top_expression
stop_expr
               := top_expression
              := top_expression
step_expr
limit_expr
              := top_expression
```

#### Notes:

- a top\_expression always has a value. The value of an expression\_list is the value of the last top\_expression in the list. For example, the value of the expression list 1;2; 'foobar'; 3 is 3.
- In a logical context, any non-empty value is True
- In a logical context, the empty value is False
- Strings and numbers can be used interchangeably. For example, 10 and '10' are the same thing.
- · Comments are lines starting with a '#' character, possibly preceded by blanks or tabs.

#### **Operator precedence**

The operator precedence (order of evaluation) from highest (evaluated first) to lowest (evaluated last) is:

- Function calls, constants, parenthesized expressions, statement expressions, assignment expressions, field references.
- Unary plus (+) and minus (-). These operators evaluate right to left.

These and all the other arithmetic operators return integers if the expression results in a fractional part equal to zero. For example, if an expression returns 3.0 it is changed to 3.

- Multiply (\*) and divide (/). These operators are associative and evaluate left to right. Use parentheses if you want to change the order of evaluation.
- Add (+) and subtract (-). These operators are associative and evaluate left to right.
- Numeric and string comparisons. These operators return '1' if the comparison succeeds, otherwise the empty string (''). Comparisons are not associative: a < b < c is a syntax error.
- String concatenation (&). The & operator returns a string formed by concatenating the left-hand and right-hand expressions. Example: 'aaa' & 'bbb' returns 'aaabbb'. The operator is associative and evaluates left to right.
- Unary logical not (!). This operator returns '1' if the expression is False (evaluates to the empty string), otherwise ''.
- Logical and (&&). This operator returns '1' if both the left-hand and right-hand expressions are True, or the empty string '' if either is False. It is associative, evaluates left to right, and does short-circuiting<sup>88</sup>.
- Logical or (||). This operator returns '1' if either the left-hand or right-hand expression is True, or '' if both are False. It is associative, evaluates left to right, and does short-circuiting<sup>89</sup>. It is an *inclusive or*, returning '1' if both the left- and right-hand expressions are True.

#### **Field references**

A field\_reference evaluates to the value of the metadata field named by lookup name that follows the \$ or \$. Using \$ is equivalent to using the *field* (page 193) function. Using \$ is equivalent to using the *raw\_field* (page 195) function. Examples:

```
* $authors ==> field('authors')
* $#genre ==> field('#genre')
```

```
* $$pubdate ==> raw_field('pubdate')
```

```
* $$#my_int ==> raw_field('#my_int')
```

#### If expressions

If expressions first evaluate the condition. If the condition is True (a non-empty value) then the expression\_list in the then clause is evaluated. If it is False then if present the expression\_list in the elif or else clause is evaluated. The elif and else parts are optional. The words if, then, elif, else, and fi are reserved; you cannot use them as identifier names. You can put newlines and white space wherever they make sense. The condition is a top\_expression not an expression\_list; semicolons are not allowed. The expression\_lists are semicolonseparated sequences of top\_expressions. An if expression returns the result of the last top\_expression in the evaluated expression\_list, or the empty string if no expression list was evaluated.

Examples:

```
* program: if field('series') then 'yes' else 'no' fi
* program:
    if field('series') then
        a = 'yes';
        b = 'no'
    else
        a = 'no';
        b = 'yes'
    fi;
    strcat(a, '-', b)
```

88 https://chortle.ccsu.edu/java5/Notes/chap40/ch40\_2.html

89 https://chortle.ccsu.edu/java5/Notes/chap40/ch40\_2.html

Nested if example:

```
program:
if field('series') then
    if check_yes_no(field('#mybool'), '', '', '1') then
        'yes'
    else
        'no'
    fi
else
        'no series'
fi
```

As said above, an if produces a value. This means that all the following are equivalent:

```
* program: if field('series') then 'foo' else 'bar' fi
* program: if field('series') then a = 'foo' else a = 'bar' fi; a
* program: a = if field('series') then 'foo' else 'bar' fi; a
```

For example, this program returns the value of the series column if the book has a series, otherwise the value of the title column:

program: field(if field('series') then 'series' else 'title' fi)

#### For expressions

The for expression iterates over a list of values, processing them one at a time. The <code>list\_expression</code> must evaluate either to a metadata field <code>lookup</code> name e.g., tags or #genre, or to a list of values. The *range* (page 200) generates a list of numbers. If the result is a valid <code>lookup</code> name then the field's value is fetched and the separator specified for that field type is used. If the result isn't a valid <code>lookup</code> name then it is assumed to be a list of values. The list is assumed to be separated by commas unless the optional keyword <code>separator</code> is supplied, in which case the list values must be separated by the result of evaluating the <code>separator\_expr</code>. A separator cannot be used if the list is generated by <code>range()</code>. Each value in the list is assigned to the specified variable then the expression\_list is evaluated. You can use <code>break</code> to jump out of the loop, and <code>continue</code> to jump to the beginning of the loop for the next iteration.

Example: This template removes the first hierarchical name for each value in Genre (#genre), constructing a list with the new names:

```
program:
    new_tags = '';
    for i in '#genre':
        j = re(i, '^.*?\.(.*)$', '\1');
        new_tags = list_union(new_tags, j, ',')
        rof;
        new_tags
```

If the original Genre is *History.Military, Science Fiction.Alternate History, ReadMe* then the template returns *Military, Alternate History, ReadMe*. You could use this template in calibre's *Edit metadata in bulk*  $\rightarrow$  *Search & replace* with *Search for* set to template to strip off the first level of the hierarchy and assign the resulting value to Genre.

Note: the last line in the template, new\_tags, isn't strictly necessary in this case because for returns the value of the last top\_expression in the expression list. The value of an assignment is the value of its expression, so the value of the for statement is what was assigned to new\_tags.

#### **Function definition**

If you have repeated code in a template then you can put that code into a local function. The def keyword starts the

definition. It is followed by the function name, the argument list, then the code in the function. The function definition ends with the fed keyword.

Arguments are positional. When a function is called the supplied arguments are matched left to right against the defined parameters, with the value of the argument assigned to the parameter. It is an error to provide more arguments than defined parameters. Parameters can have default values, such as a = 25. If an argument is not supplied for that parameter then the default value is used, otherwise the parameter is set to the empty string.

The return statement can be used in a local function.

A function must be defined before it can be used.

Example: This template computes an approximate duration in years, months, and days from a number of days. The function to\_plural() formats the computed values. Note that the example also uses the & operator:

```
program:
    days = 2112;
    years = floor(days/360);
    months = floor(mod(days, 360)/30);
    days = days - ((years*360) + (months * 30));
    def to_plural(v, str):
        if v == 0 then return '' fi;
        return v & ' ' & (if v == 1 then str else str & 's' fi) & ' '
    fed;
    to_plural(years, 'year') & to_plural(months, 'month') & to_plural(days,'day')
```

#### **Relational operators**

Relational operators return '1' if the comparison is true, otherwise the empty string ('').

There are two forms of relational operators: string comparisons and numeric comparisons.

String comparisons do case-insensitive string comparison using lexical order. The supported string comparison operators are ==, !=, <, <=, >, >=, in, inlist, and inlist\_field. For the in, inlist, and inlist\_field operators, the result of the left hand expression is interpreted as a regular expression pattern. They are true if the value of left-hand regular expression matches the value of the right hand expression. The regular expressions are case-insensitive.

The inlist operator is true if the left hand regular expression matches any one of the items in the right hand list where the items in the list are separated by commas. The inlist\_field operator is true if the left hand regular expression matches any of the items in the field (column) named by the right hand expression, using the separator defined for the field. NB: the inlist\_field operator requires the right hand expression to evaluate to a field name, while the inlist operator requires the right hand expression to evaluate to a string containing a comma-separated list. Because of this difference, inlist\_field is substantially faster than inlist because no string conversions or list constructions are done.

The numeric comparison operators are ==#, !=#, <#, <#, >#, >=#. The left and right expressions must evaluate to numeric values with two exceptions: both the string value "None" (undefined field) and the empty string evaluate to the value zero.

Examples:

- program: field('series') == 'foo' returns '1' if the book's series is foo, otherwise ''.
- program: 'f.o' in field('series') returns '1' if the book's series matches the regular expression f.o (e.g., *foo*, *Off Onyx*, etc.), otherwise ''.
- program: 'science' inlist \$#genre returns '1' if any of the values retrieved from the book's genres match the regular expression science, e.g., *Science, History of Science, Science Fiction* etc., otherwise ''.

- program: '^science\$' inlist \$#genre returns '1' if any of the book's genres exactly match the regular expression ^science\$, e.g., Science, otherwise ''. The genres History of Science and Science Fiction don't match.
- program: 'asimov' inlist \$authors returns '1' if any author matches the regular expression asimov, e.g., *Asimov, Isaac* or *Isaac Asimov*, otherwise ''.
- program: 'asimov' inlist\_field 'authors' returns '1' if any author matches the regular expression asimov, e.g., *Asimov, Isaac* or *Isaac Asimov*, otherwise ''.
- program: 'asimov\$' inlist\_field 'authors' returns '1' if any author matches the regular expression asimov\$, e.g., *Isaac Asimov*, otherwise ''. It doesn't match *Asimov*, *Isaac* because of the \$ anchor in the regular expression.
- program: if field('series') != 'foo' then 'bar' else 'mumble' fi returns 'bar' if the book's series is not *foo*. Otherwise it returns 'mumble'.
- program: if field('series') == 'foo' || field('series') == '1632' then 'yes' else 'no' fi returns 'yes' if series is either *foo* or *1632*, otherwise 'no'.
- program: if '^(foo|1632)\$' in field('series') then 'yes' else 'no' fi returns 'yes' if series is either *foo* or *1632*, otherwise 'no'.
- program: if 11 > 2 then 'yes' else 'no' fi returns 'no' because the > operator does a lexical comparison.
- program: if 11 ># 2 then 'yes' else 'no' fireturns 'yes' because the ># operator does a numeric comparison.

#### **Functions in General Program Mode**

See Template function reference (page 178) for the list of functions built into the template language.

Notes:

- As opposed to *Single Function Mode* (page 161), in General Program Mode you must specify the first parameter value.
- All parameters are expression\_lists (see the grammar above).

#### 10.3.7 More complex programs in template expressions - Template Program Mode

*Template Program Mode (TPM)* is a blend of *General Program Mode* (page 164) and *Single Function Mode* (page 161). *TPM* differs from Single Function Mode in that it permits writing template expressions that refer to other metadata fields, use nested functions, modify variables, and do arithmetic. It differs from *General Program Mode* in that the template is contained between { and } characters and doesn't begin with the word program:. The program portion of the template is a General Program Mode expression list.

Example: assume you want a template to show the series for a book if it has one, otherwise show the value of a custom field #genre. You cannot do this in the *Single Function Mode* (page 161) because you cannot make reference to another metadata field within a template expression. In *TPM* you can, as the following expression demonstrates:

```
{series:'ifempty($, $#genre)'}
```

The example shows several things:

• *TPM* is used if the expression begins with : ' and ends with ' }. Anything else is assumed to be in *Single Function Mode* (page 161).

If the template contains a prefix and suffix, the expression ends with '| where the | is the delimiter for the prefix. Example:

```
{series:'ifempty($, $#genre)'|prefix | suffix}
```

- Functions must be given all their arguments. For example, the standard built-in functions must be given the initial parameter value.
- The variable \$ is usable as the value argument and stands for the value of the field named in the template, series in this case.
- white space is ignored and can be used anywhere within the expression.
- constant strings are enclosed in matching quotes, either ' or ".

In *TPM*, using { and } characters in string literals can lead to errors or unexpected results because they confuse the template processor. It tries to treat them as template expression boundaries, not characters. In some but not all cases you can replace a { with [ [ and a } with ]]. Advice: if your program contains { and } characters then you should use *General Program Mode*.

## 10.3.8 Python Template Mode

Python Template Mode (PTM) lets you write templates using native Python and the calibre API<sup>90</sup>. The database API will be of most use; further discussion is beyond the scope of this manual. PTM templates are faster and can do more complicated operations but you must know how to write code in Python using the calibre API.

#### A PTM template begins with:

```
python:
def evaluate(book, context):
    # book is a calibre metadata object
    # context is an instance of calibre.utils.formatter.PythonTemplateContext,
    # which currently contains the following attributes:
    # db: a calibre legacy database object.
    # globals: the template global variable dictionary.
    # arguments: is a list of arguments if the template is called by a GPM template,_
    +otherwise None.
    # funcs: used to call Built-in/User functions and Stored GPM/Python templates.
    # Example: context.funcs.list_re_group()
    # your Python code goes here
    return 'a string'
```

You can add the above text to your template using the context menu, usually accessed with a right click. The comments are not significant and can be removed. You must use python indenting.

The context object supports str(context) that returns a string of the context's contents, and context.attributes that returns a list of the attribute names in the context.

The context.funcs attribute allows calling Built-in and User template functions, and Stored GPM/Python templates, so that you can execute them directly in your code. The functions are retrieved using their names. If the name conflicts with a Python keyword, add an underscore to the end of the name. Examples:

```
context.funcs.list_re_group()
context.funcs.assert_()
```

Here is an example of a PTM template that produces a list of all the authors for a series. The list is stored in a *Column built* from other columns, behaves like tags. It shows in *Book details* and has the on separate lines checked (in *Preferences*  $\rightarrow$  *Look* & feel  $\rightarrow$  Book details). That option requires the list to be comma-separated. To satisfy that requirement the template

<sup>90</sup> https://manual.calibre-ebook.com/develop.html#api-documentation-for-various-parts-of-calibre

converts commas in author names to semicolons then builds a comma-separated list of authors. The authors are then sorted, which is why the template uses author\_sort.

```
python:
def evaluate(book, context):
   if book.series is None:
       return ''
   db = context.db.new_api
   ans = set()
    # Get the list of books in the series
   ids = db.search(f'series:"={book.series}"', '')
   if ids:
        # Get all the author_sort values for the books in the series
       author_sorts = (v for v in db.all_field_for('author_sort', ids).values())
        # Add the names to the result set, removing duplicates
       for aus in author_sorts:
           ans.update(v.strip() for v in aus.split('&'))
    # Make a sorted comma-separated string from the result set
   return ', '.join(v.replace(',', ';') for v in sorted(ans))
```

The output in Book details looks like this:

	•	
~	Authors:	John Public
•	Series:	3.02 of Great Series
l.	lds:	1111
r	Publisher:	Baen
	Date:	13 Jan 2011
	Published:	2011
	Modified:	2022-10-10T16:39:34.066740+01:00
	Languages:	English
	Series Authors	s: B; A
		C; B
		Flintstone; Wilma
Ŀ		Machiavelli; Niccolò
		Nietzsche; Friedrich
9		Papper; A. Persone B. C.
		Public; John

#### 10.3.9 Templates and URLs

You can use templates to construct URLs. Two cases are described here:

- Custom column Book details search URLs
- The calibre URL scheme

#### Custom column book details search URLs

When you create a custom column you can provide a URL to be used in *Book details* using a template. For example, if you have a custom column for *Translators* you can define a URL to take you to a site for translators. Book details search URLs can be provided for *Text, Enumerated, Series*, and *Column built from other column* column types.

When an item with a *search template* is clicked in *Book details* the template is evaluated. It is provided the normal book metadata. It is also provided three additional fields:

- item\_value: the value of the clicked item.
- item\_value\_quoted: the value of clicked item, URL-encoded. Special characters are escaped to make them valid in URLs and spaces are replaced by '+' (plus) signs.
- item\_value\_no\_plus: the value of clicked item, URL-encoded. Special characters are escaped to make them valid in URLs and spaces are replaced by the %20, not plus.

There are several ways to construct the URL. The following use Wikipedia as an example.

The simplest is a basic template:

https://en.wikipedia.org/w/index.php?search={item\_value\_encoded}

In some cases you might want to do more processing. There are four template functions you can use, depending on the complexity of the processing.

- *make\_url* (page 206) (path, [query\_name, query\_value]+) this function is the easiest way to construct a query URL. It uses a path, the web site and page you want to query, and query\_name, query\_value pairs from which the query is built. In general, the query\_value must be URL-encoded. With this function it is always encoded and spaces are always replaced with '+' signs.
- *make\_url\_extended* (page 206) (...) this function is similar to *make\_url()* (page 206) but gives you more control over the URL components. The components of a URL are

#### scheme:://authority/path?query string.

See Uniform Resource Locator<sup>91</sup> on Wikipedia for more detail.

The function has two variants:

```
make_url_extended(scheme, authority, path, [query_name, query_value]+)
```

and

make\_url\_extended(scheme, authority, path, query\_string)

- *query\_string* (page 207)([query\_name, query\_value, how\_to\_encode]+) returns a URL query string constructed from the query\_name, query\_value, how\_to\_encode triads. A query string is a series of items where each item looks like query\_name=query\_value where query\_value is URL-encoded as instructed. The query items are separated by '&' (ampersand) characters.
- *encode\_for\_url* (page 206) (value, use\_plus) returns the value encoded for use in a URL as specified by use\_plus. The value is first URL-encoded. Next, if use\_plus is 0 then spaces are replaced by '+' (plus) signs. If it is 1 then spaces are replaced by %20.

For example, assume you have a custom column *Translators* (#translators) where the names are *Last name*, *First name*. You might need to convert the name to *First name Last name* when creating the URL. You can use the *make\_url* (page 206) function to do this:

<sup>91</sup> https://en.wikipedia.org/wiki/URL

If we assume that the translator's name is *Boy-Żeleński*, *Tadeusz* then the above template produces the link:

https://en.wikipedia.org/w/index.php?search=Tadeusz+Boy-%C5%BBele%C5%84ski

Note that the person's first name is now first, the space is now a plus, and that the non-English characters in the last name are URL-encoded.

The functions *make\_url\_extended* (page 206), *query\_string* (page 207), and *encode\_for\_url* (page 206) might be useful depending upon any additional processing complexity.

#### The calibre URL scheme

Calibre supports several different URLs to navigate your calibre libraries. This section shows how to use templates to construct some of the URLs. See *The calibre:// URL scheme* (page 249) for details on the URLs available.

• Switch to a specific library. The syntax of this URL is:

calibre://switch-library/Library\_Name

Library\_Name must be replaced with the name of the calibre library you wish to open. The library name is shown in the title bar of the window. It is a simple name, not the file path to the library. You must spell it as shown in the title bar, including letter case. The character \_ (underscore) stands for the current library. If the name contains any spaces or special characters then it must be hex encoded using the *to\_hex* (page 208) function, as in the following example:

program: strcat('calibre://switch-library\_hex\_-', to\_hex(current\_library\_name()))

The template generates the URL:

calibre://switch-library/\_hex\_-4c6962726172792e746573745f736d616c6c

You can replace the current\_library\_name() function with the actual name of the library, as in:

program: strcat('calibre://switch-library/\_hex\_-', to\_hex('Library.test\_small'))

• Links to show books. These links select a book in the calibre library. The syntax for this URL is:

calibre://show-book/Library\_Name/book\_id

The book id is the numeric calibre id for the book, available to templates as \$id. As above, the library name might need to be hex encoded. Here is an example:

It produces the URL:

calibre://show-book/\_hex\_-4c6962726172792e746573745f736d616c6c/1353

• Searching for books. These links search for books in the specified calibre library. The syntax for this URL is:

```
calibre://search/Library_Name?q=query
calibre://search/Library_Name?eq=hex_encoded_query
```

where *query* is any valid calibre search expression. You must hex encode any query containing spaces or special characters, which generally means all of them. For example, the calibre search expression for searching for a hierarchical tag beginning with 'AA' is tags: "=. AA". This template constructs a search URL for that expression:

The resulting URL is:

Here is an example of the same URL built using the :ref:ff\_make\_url\_extended function instead of *strcat* (page 204):

```
'eq', to_hex('tags:"=.AA"'))
```

• Open a book details window on a book in some library. The syntax for this URL is:

calibre://book-details/Library\_Name/book\_id

An example template is:

which produces the URL:

```
calibre://book-details/_hex_-4c6962726172792e746573745f736d616c6c/1353
```

• Open the notes associated with an author/series/etc. The syntax of the URL is:

calibre://book-details/Library\_Name/Field\_Name/id\_Item\_Id
calibre://book-details/Library\_Name/Field\_Name/hex\_Hex\_Encoded\_Item\_Name

Field\_Name is the lookup name of the field. If the field is a custom column then replace the # character with an underscore (\_). Item\_Id is the internal numeric ID of the value in the field. There isn't a template function that returns the Item\_Id, so templates will normally use the second form, Hex\_Encoded\_Item\_Name. Here is a sample template that opens the note for the person Boy-Żeleński, Tadeusz in the field #authtest:

which produces the URL:

```
calibre://show-note/_hex_-4c6962726172792e746573745f736d616c6c/_authtest/hex_

\leftrightarrow426f792dc5bb656c65c584736b692c205461646575737a
```

### 10.3.10 Stored templates

Both *General Program Mode* (page 164) and *Python Template Mode* (page 170) support saving templates and calling those templates from another template, much like calling stored functions. You save templates using *Preferences*  $\rightarrow$  *Advanced*  $\rightarrow$  *Template functions*. More information is provided in that dialog. You call a template the same

way you call a function, passing positional arguments if desired. An argument can be any expression. Examples of calling a template, assuming the stored template is named foo:

- $f \circ \circ ()$  call the template passing no arguments.
- foo(a, b) call the template passing the values of the two variables a and b.
- foo(if field('series') then field('series\_index') else 0 fi) if the book has a series then pass the series\_index, otherwise pass the value 0.

In GPM you retrieve the arguments passed in the call to the stored template using the arguments function. It both declares and initializes local variables, effectively parameters. The variables are positional; they get the value of the parameter given in the call in the same position. If the corresponding parameter is not provided in the call then arguments assigns that variable the provided default value. If there is no default value then the variable is set to the empty string. For example, the following arguments function declares 2 variables, key, alternate:

```
arguments(key, alternate='series')
```

Examples, again assuming the stored template is named foo:

- foo('#myseries') argument key is assigned the value 'myseries' and the argument alternate is assigned the default value 'series'.
- foo('series', '#genre') the variable key is assigned the value 'series' and the variable alternate is assigned the value '#genre'.
- foo() the variable key is assigned the empty string and the variable alternate is assigned the value 'series'.

In PTM the arguments are passed in the arguments parameter, which is a list of strings. There isn't any way to specify default values. You must check the length of the arguments list to be sure that the number of arguments is what you expect.

An easy way to test stored templates is using the Template tester dialog. For ease of access give it a keyboard shortcut in *Preferences*  $\rightarrow$  *Advanced*  $\rightarrow$  *Keyboard shortcuts*  $\rightarrow$  *Template tester*. Giving the Stored templates dialog a shortcut will help switching more rapidly between the tester and editing the stored template's source code.

#### 10.3.11 Providing additional information to templates

A developer can choose to pass additional information to the template processor, such as application-specific book metadata or information about what the processor is being asked to do. A template can access this information and use it during the evaluation.

#### Developer: how to pass additional information

The additional information is a Python dictionary containing pairs variable\_name: variable\_value where the values must be strings. The template can access the dictionary, creating template local variables named variable\_name containing the value variable\_value. The user cannot change the name so it is best to use names that won't collide with other template local variables, for example by prefixing the name with an underscore.

This dictionary is passed to the template processor (the formatter) using the named parameter global\_vars=your\_dict. The full method signature is:

#### Template writer: how to access the additional information

You access the additional information (the globals dictionary) in a template using the template function:

globals(id[=expression] [, id[=expression]]\*)

where id is any legal variable name. This function checks whether the additional information provided by the developer contains the name. If it does then the function assigns the provided value to a template local variable with that name. If the name is not in the additional information and if an expression is provided, the expression is evaluated and the result is assigned to the local variable. If neither a value nor an expression is provided, the function assigns the empty string ('') to the local variable.

A template can set a value in the globals dictionary using the template function:

set\_globals(id[=expression] [, id[=expression]]\*)

This function sets the globals dictionary key:value pair id:value where value is the value of the template local variable id. If that local variable doesn't exist then value is set to the result of evaluating expression.

#### 10.3.12 Notes on the difference between modes

The three program modes, *Single Function Mode* (page 161) (SFM), *Template Program Mode* (page 169) (*TPM*), and *General Program Mode* (page 164) (*GPM*), work differently. SFM is intended to be 'simple' so it hides a lot of programming language bits.

Differences:

- In SFM the value of the column is always passed as an 'invisible' first argument to a function included in the template.
- SFM doesn't support the difference between variables and strings; all values are strings.
- The following SFM template returns either the series name or the string "no series":

{series:ifempty(no series)}

The equivalent template in *TPM* is

{series:'ifempty(\$, 'no series')'}

The equivalent template in *GPM* is:

```
program: ifempty(field('series'), 'no series')
```

The first argument to ifempty is the value of the field series. The second argument is the string no series. In SFM the first argument, the value of the field, is automatically passed (the invisible argument).

- Several template functions, for example booksize() and current\_library\_name(), take no arguments. Because of the 'invisible argument' you cannot use these functions in SFM.
- Nested functions, where a function calls another function to compute an argument, cannot be used in SFM. For example this template, intended to return the first 5 characters of the series value uppercased, won't work in SFM:

{series:uppercase(substr(0,5))}

• TPM and GPM support nested functions. The above template in TPM would be:

```
{series:'uppercase(substr($, 0,5))'}
```

In GPM it would be:

program: uppercase(substr(field('series'), 0,5))

• As noted in the above *Template Program Mode* (page 169) section, using { and } characters in *TPM* string literals can lead to errors or unexpected results because they confuse the template processor. It tries to treat them as template boundaries, not characters. In some but not all cases you can replace a { with [ [ and a } with ]]. Generally, if your program contains { and } characters then you should use *General Program Mode*.

# 10.3.13 User-defined Python template functions

You can add your own Python functions to the template processor. Such functions can be used in any of the three template programming modes. The functions are added by going to *Preferences*  $\rightarrow$  *Advanced*  $\rightarrow$  *Template functions*. Instructions are shown in that dialog. Note that you can use *Python Templates* for a similar purpose. As calling user-defined functions is faster than calling a Python template, user-defined functions might be more efficient depending on the complexity of what the function or template does.

# 10.3.14 Special notes for using templates in different contexts

In the GUI (Columns made from other columns and Template searches):

- GPM templates work as before.
- Python templates have full access to the calibre database.

In icon rules:

• icon rule templates have no book data so field-based functions such as *format\_date\_field* (page 189), *list\_count\_field* (page 197), and *check\_yes\_no* (page 203) won't work.

In the Content server:

- Templates have access to the new API but not the old API (LibraryDatabase).
- Because of the above, the following formatter functions are not guaranteed to work in GPM templates (composite columns, icon rules, etc) and should be avoided if you use the content server:
  - connected\_device\_name (page 192)
  - connected\_device\_uuid (page 192)
  - current\_virtual\_library\_name (page 193)
  - is\_marked (page 194)
  - virtual\_libraries (page 195)

# 10.3.15 Special notes for save/send templates

Special processing is applied when a template is used in a *Save to disk* or *Send to device* template. The values of the fields are cleaned, replacing characters that are special to file systems with underscores, including slashes. This means that field text cannot be used to create folders. However, slashes are not changed in prefix or suffix strings, so slashes in these strings will cause folders to be created. Because of this, you can create variable-depth folder structure.

For example, assume we want the folder structure *series/series\_index - title*, with the caveat that if series does not exist, then the title should be in the top folder. The template to do this is:

{series:||/}{series\_index:|| - }{title}

The slash and the hyphen appear only if series is not empty.

The lookup function lets us do even fancier processing. For example, assume that if a book has a series, then we want the folder structure *series/series index - title.fmt*. If the book does not have a series then we want the folder structure

genre/author\_sort/title.fmt. If the book has no genre then we want to use 'Unknown'. We want two completely different paths, depending on the value of series.

To accomplish this, we:

- 1. Create a composite field (give it lookup name #aa) containing {series}/{series\_index} {title}. If the series is not empty, then this template will produce *series/series\_index title*.
- 2. Create a composite field (give it lookup name #bb) containing {#genre:ifempty(Unknown)}/ {author\_sort}/{title}. This template produces genre/author\_sort/title, where an empty genre is replaced with Unknown.
- 3. Set the save template to {series:lookup(., #aa, #bb)}. This template chooses composite field #aa if series is not empty and composite field #bb if series is empty. We therefore have two completely different save paths, depending on whether or not *series* is empty.

# 10.3.16 Tips

- Use the Template Tester to test templates. Add the tester to the context menu for books in the library and/or give it a keyboard shortcut.
- Templates can use other templates by referencing composite columns built with the desired template. Alternatively, you can use Stored Templates.
- In a plugboard, you can set a field to empty (or whatever is equivalent to empty) by using the special template {}. This template will always evaluate to an empty string.
- The technique described above to show numbers even if they have a zero value works with the standard field series\_index.

# 10.3.17 Template function reference

# Reference for all built-in template language functions

Here, we document all the built-in functions available in the calibre template language. Every function is implemented as a class in python and you can click the source links to see the source code, in case the documentation is insufficient. The functions are arranged in logical groups by type.

- *Arithmetic* (page 182)
  - *add* (page 182)
  - ceiling (page 182)
  - divide (page 182)
  - *floor* (page 183)
  - *fractional\_part* (page 183)
  - mod (page 183)
  - multiply (page 183)
  - round (page 183)
  - subtract (page 183)
- Boolean (page 183)
  - and (page 183)

- *not* (page 183)
- or (page 184)
- Case changes (page 184)
  - capitalize (page 184)
  - *lowercase* (page 184)
  - titlecase (page 184)
  - uppercase (page 184)
- Database functions (page 184)
  - book\_count (page 184)
  - book\_values (page 185)
  - extra\_file\_modtime (page 185)
  - extra\_file\_names (page 185)
  - extra\_file\_size (page 186)
  - get\_link (page 186)
  - *get\_note* (page 186)
  - has\_extra\_files (page 187)
  - *has\_note* (page 187)
- Date functions (page 187)
  - date\_arithmetic (page 187)
  - days\_between (page 188)
  - today (page 188)
- Formatting values (page 188)
  - finish\_formatting (page 188)
  - format\_date (page 188)
  - *format\_date\_field* (page 189)
  - *format\_duration* (page 190)
  - *format\_number* (page 191)
  - *human\_readable* (page 191)
  - *rating\_to\_stars* (page 191)
- Get values from metadata (page 191)
  - annotation\_count (page 191)
  - *approximate\_formats* (page 191)
  - author\_links (page 191)
  - author\_sorts (page 192)
  - *booksize* (page 192)

- connected\_device\_name (page 192)
- connected\_device\_uuid (page 192)
- current\_library\_name (page 192)
- *current\_library\_path* (page 192)
- current\_virtual\_library\_name (page 193)
- *field* (page 193)
- *formats\_modtimes* (page 193)
- formats\_path\_segments (page 193)
- formats\_paths (page 194)
- formats\_sizes (page 194)
- *has\_cover* (page 194)
- is\_marked (page 194)
- *language\_codes* (page 194)
- *language\_strings* (page 194)
- ondevice (page 195)
- raw\_field (page 195)
- *raw\_list* (page 195)
- series\_sort (page 195)
- user\_categories (page 195)
- virtual\_libraries (page 195)
- Iterate over values (page 195)
  - *first\_non\_empty* (page 195)
  - lookup (page 196)
  - switch (page 196)
  - switch\_if (page 196)
- *List lookup* (page 196)
  - *identifier\_in\_list* (page 196)
  - *list\_contains* (page 196)
  - *list\_item* (page 196)
  - select (page 197)
  - *str\_in\_list* (page 197)
- List manipulation (page 197)
  - list\_count (page 197)
  - *list\_count\_field* (page 197)
  - *list\_count\_matching* (page 197)

- list\_difference (page 198)
- list\_equals (page 198)
- *list\_intersection* (page 198)
- *list\_join* (page 198)
- *list\_re* (page 199)
- *list\_re\_group* (page 199)
- *list\_remove\_duplicates* (page 199)
- *list\_sort* (page 199)
- *list\_split* (page 199)
- list\_union (page 199)
- range (page 200)
- subitems (page 200)
- sublist (page 201)
- *Other* (page 201)
  - arguments (page 201)
  - assign (page 201)
  - globals (page 201)
  - is\_dark\_mode (page 201)
  - print (page 202)
  - set\_globals (page 202)
- Recursion (page 202)
  - eval (page 202)
  - template (page 202)
- Relational (page 202)
  - *cmp* (page 202)
  - first\_matching\_cmp (page 202)
  - strcmp (page 203)
  - strcmpcase (page 203)
- String manipulation (page 203)
  - character (page 203)
  - check\_yes\_no (page 203)
  - contains (page 203)
  - *field\_exists* (page 204)
  - *ifempty* (page 204)
  - *re* (page 204)

- re\_group (page 204)
- shorten (page 204)
- strcat (page 204)
- strcat\_max (page 205)
- strlen (page 205)
- substr (page 205)
- *swap\_around\_articles* (page 205)
- swap\_around\_comma (page 205)
- test (page 205)
- transliterate (page 205)
- URL functions (page 206)
  - encode\_for\_url (page 206)
  - make\_url (page 206)
  - make\_url\_extended (page 206)
  - query\_string (page 207)
  - to\_hex (page 208)
  - urls\_from\_identifiers (page 208)
- API of the Metadata objects (page 208)

# Arithmetic

# add

class calibre.utils.formatter\_functions.BuiltinAdd

add  $(x [, y]^*)$  – returns the sum of its arguments. Throws an exception if an argument is not a number. In most cases you can use the + operator instead of this function.

# ceiling

class calibre.utils.formatter\_functions.BuiltinCeiling

ceiling(value) - returns the smallest integer greater than or equal to value. Throws an exception if value is not a number.

#### divide

class calibre.utils.formatter\_functions.BuiltinDivide

divide (x, y) – returns x / y. Throws an exception if either x or y are not numbers. This function can usually be replaced by the / operator.

# floor

class calibre.utils.formatter\_functions.BuiltinFloor

floor (value) - returns the largest integer less than or equal to value. Throws an exception if value is not a number.

# fractional\_part

class calibre.utils.formatter\_functions.BuiltinFractionalPart

fractional\_part(value) - returns the part of the value after the decimal point. For example, fractional\_part(3.14) returns 0.14. Throws an exception if value is not a number.

### mod

```
class calibre.utils.formatter_functions.BuiltinMod
```

mod(value, y) - returns the floor of the remainder of value / y. Throws an exception if either value or y is not a number.

#### multiply

class calibre.utils.formatter\_functions.BuiltinMultiply

multiply  $(x [, y]^*)$  – returns the product of its arguments. Throws an exception if any argument is not a number. This function can usually be replaced by the \* operator.

#### round

class calibre.utils.formatter\_functions.BuiltinRound

round (value) - returns the nearest integer to value. Throws an exception if value is not a number.

# subtract

class calibre.utils.formatter\_functions.BuiltinSubtract

subtract (x, y) – returns x – y. Throws an exception if either x or y are not numbers. This function can usually be replaced by the – operator.

#### **Boolean**

### and

class calibre.utils.formatter\_functions.BuiltinAnd

and (value [, value]\*) – returns the string '1' if all values are not empty, otherwise returns the empty string. You can have as many values as you want. In most cases you can use the && operator instead of this function. One reason not to replace and () with && is when short-circuiting can change the results because of side effects. For example, and (a='', b=5) will always do both assignments, where the && operator won't do the second.

#### not

class calibre.utils.formatter\_functions.BuiltinNot

not (value) – returns the string '1' if the value is empty, otherwise returns the empty string. This function can usually be replaced with the unary not (!) operator.

#### or

class calibre.utils.formatter\_functions.BuiltinOr

or  $(value [, value]^*)$  – returns the string '1' if any value is not empty, otherwise returns the empty string. You can have as many values as you want. This function can usually be replaced by the || operator. A reason it cannot be replaced is if short-circuiting will change the results because of side effects.

# **Case changes**

# capitalize

class calibre.utils.formatter\_functions.BuiltinCapitalize

capitalize (value) – returns the value with the first letter in upper case and the rest lower case.

### lowercase

class calibre.utils.formatter\_functions.BuiltinLowercase

lowercase (value) - returns the value in lower case.

#### titlecase

class calibre.utils.formatter\_functions.BuiltinTitlecase

titlecase (value) - returns the value in title case.

#### uppercase

class calibre.utils.formatter\_functions.BuiltinUppercase

uppercase (value) - returns the value in upper case.

### **Database functions**

#### book\_count

class calibre.utils.formatter\_functions.BuiltinBookCount

book\_count (query, use\_vl) - returns the count of books found by searching for query. If use\_vl is 0 (zero) then virtual libraries are ignored. This function and its companion book\_values() are particularly useful in template searches, supporting searches that combine information from many books such as looking for series with only one book. It cannot be used in composite columns unless the tweak allow\_template\_database\_functions\_in\_composites is set to True. It can be used only in the GUI.

For example this template search uses this function and its companion to find all series with only one book:

• Define a stored template (using *Preferences* → *Advanced* → *Template functions*) named series\_only\_one\_book (the name is arbitrary). The template is:

```
program:
  vals = globals(vals='');
  if !vals then
     all_series = book_values('series', 'series:true', ',', 0);
     for series in all_series:
```

(continues on next page)

(continued from previous page)

```
if book_count('series:="' & series & '"', 0) == 1 then
        vals = list_join(',', vals, ',', series, ',')
        fi
        rof;
        set_globals(vals)
fi;
str_in_list(vals, ',', $series, 1, '')
```

The first time the template runs (the first book checked) it stores the results of the database lookups in a global template variable named vals. These results are used to check subsequent books without redoing the lookups.

• Use the stored template in a template search:

template:"program: series\_only\_one\_book()#@#:n:1"

Using a stored template instead of putting the template into the search eliminates problems caused by the requirement to escape quotes in search expressions.

This function can be used only in the GUI and the content server.

#### book\_values

class calibre.utils.formatter\_functions.BuiltinBookValues

book\_values(column, query, sep, use\_vl) - returns a list of the unique values contained in the column column (a lookup name), separated by sep, in the books found by searching for query. If use\_vl is 0 (zero) then virtual libraries are ignored. This function and its companion book\_count() are particularly useful in template searches, supporting searches that combine information from many books such as looking for series with only one book. It cannot be used in composite columns unless the tweak allow\_template\_database\_functions\_in\_composites is set to True. This function can be used only in the GUI and the content server.

# extra\_file\_modtime

class calibre.utils.formatter\_functions.BuiltinExtraFileModtime

extra\_file\_modtime (file\_name, format\_string) - returns the modification time of the extra file file\_name in the book's data/ folder if it exists, otherwise -1. The modtime is formatted according to format\_string (see format\_date() (page 188) for details). If format\_string is the empty string, returns the modtime as the floating point number of seconds since the epoch. See also the functions has\_extra\_file() (page 187), extra\_file\_names() (page 185) and extra\_file\_size() (page 186). The epoch is OS dependent. This function can be used only in the GUI and the content server.

#### extra\_file\_names

#### class calibre.utils.formatter\_functions.BuiltinExtraFileNames

extra\_file\_names(sep [, pattern]) - returns a sep-separated list of extra files in the book's data/ folder. If the optional parameter pattern, a regular expression, is supplied then the list is filtered to files that match pattern. The pattern match is case insensitive. See also the functions has\_extra\_files() (page 187), extra\_file\_modtime() (page 185) and extra\_file\_size() (page 186). This function can be used only in the GUI and the content server.

# extra\_file\_size

class calibre.utils.formatter\_functions.BuiltinExtraFileSize

extra\_file\_size(file\_name) - returns the size in bytes of the extra file file\_name in the book's data/ folder if it exists, otherwise -1. See also the functions *has\_extra\_files()* (page 187), *extra\_file\_names()* (page 185) and *extra\_file\_modtime()* (page 185). This function can be used only in the GUI and the content server.

# get\_link

class calibre.utils.formatter\_functions.BuiltinGetLink

get\_link(field\_name, field\_value) - fetch the link for field field\_name with value field\_value. If there
is no attached link, return the empty string. Examples:

• The following returns the link attached to the tag Fiction:

```
get_link('tags', 'Fiction')
```

• This template makes a list of the links for all the tags associated with a book in the form value:link, ...:

```
program:
    ans = '';
    for t in $tags:
        l = get_link('tags', t);
        if l then
            ans = list_join(', ', ans, ',', t & ':' & get_link('tags', t), ',')
        fi
        rof;
    ans
```

This function works only in the GUI and the content server.

#### get\_note

class calibre.utils.formatter\_functions.BuiltinGetNote

get\_note(field\_name, field\_value, plain\_text) - fetch the note for field field\_name with value field\_value. If plain\_text is empty, return the note's HTML including images. If plain\_text is 1 (or '1'), return the note's plain text. If the note doesn't exist, return the empty string in both cases. Example:

• Return the HTML of the note attached to the tag Fiction:

```
program:
    get_note('tags', 'Fiction', '')
```

• Return the plain text of the note attached to the author Isaac Asimov:

```
program:
    get_note('authors', 'Isaac Asimov', 1)
```

This function works only in the GUI and the content server.

# has\_extra\_files

```
class calibre.utils.formatter_functions.BuiltinHasExtraFiles
```

has\_extra\_files([pattern]) - returns the count of extra files, otherwise " (the empty string). If the optional parameter pattern (a regular expression) is supplied then the list is filtered to files that match pattern before the files are counted. The pattern match is case insensitive. See also the functions *extra\_file\_names()* (page 185), *extra\_file\_size()* (page 186) and *extra\_file\_modtime()* (page 185). This function can be used only in the GUI and the content server.

# has\_note

#### class calibre.utils.formatter\_functions.BuiltinHasNote

has\_note(field\_name, field\_value). Check if a field has a note. This function has two variants:

• if field\_value is not '' (the empty string) return '1' if the value field\_value in the field\_name has a note, otherwise ''.

```
Example: has_note('tags', 'Fiction') returns '1' if the tag fiction has an attached note, otherwise ''.
```

• If field\_value is '' then return a list of values in field\_name that have a note. If no item in the field has a note, return ''. This variant is useful for showing column icons if any value in the field has a note, rather than a specific value.

Example: has\_note('authors', '') returns a list of authors that have notes, or '' if no author has a note.

You can test if all the values in field\_name have a note by comparing the list length of this function's return value against the list length of the values in field\_name. Example:

list\_count(has\_note('authors', ''), '&') ==# list\_count\_field('authors')

This function works only in the GUI and the content server.

# **Date functions**

# date\_arithmetic

class calibre.utils.formatter\_functions.BuiltinDateArithmetic

date\_arithmetic(value, calc\_spec, fmt) - Calculate a new date from value using calc\_spec. Return the new date formatted according to optional fmt: if not supplied then the result will be in ISO format. The calc\_spec is a string formed by concatenating pairs of vW (valueWhat) where v is a possibly-negative number and W is one of the following letters:

- s: add v seconds to date
- m: add v minutes to date
- h: add v hours to date
- d: add v days to date
- w: add v weeks to date
- y: add v years to date, where a year is 365 days.

Example: '1s3d-1m' will add 1 second, add 3 days, and subtract 1 minute from date.

# days\_between

class calibre.utils.formatter\_functions.BuiltinDaysBetween

days\_between (date1, date2) - return the number of days between date1 and date2. The number is positive if date1 is greater than date2, otherwise negative. If either date1 or date2 are not dates, the function returns the empty string.

# today

class calibre.utils.formatter\_functions.BuiltinToday

today () - return a date+time string for today (now). This value is designed for use in format\_date or days\_between, but can be manipulated like any other string. The date is in  $ISO^{92}$  date/time format.

# **Formatting values**

# finish\_formatting

class calibre.utils.formatter\_functions.BuiltinFinishFormatting

finish\_formatting(value, format, prefix, suffix) – apply the format, prefix, and suffix to the value in the same way as done in a template like {series\_index:05.2f| - |-}. This function is provided to ease conversion of complex single-function- or template-program-mode templates to *GPM* Templates. For example, the following program produces the same output as the above template:

program: finish\_formatting(field("series\_index"), "05.2f", " - ", " - ")

Another example: for the template:

```
{series:re(([^\s]) [^\s]+(\s|$), \1)}{series_index:0>2s| - | - }{title}
```

use:

```
program:
    strcat(
        re(field('series'), '([^\s])[^\s]+(\s|$)', '\1'),
        finish_formatting(field('series_index'), '0>2s', ' - ', ' - '),
        field('title')
)
```

# format\_date

class calibre.utils.formatter\_functions.BuiltinFormatDate

format\_date(value, format\_string) - format the value, which must be a date string, using the format\_string, returning a string. It is best if the date is in ISO format as using other date formats often causes errors because the actual date value cannot be unambiguously determined. Note that the format\_date\_field() function is both faster and more reliable.

The formatting codes are:

- d : the day as number without a leading zero (1 to 31)
- dd : the day as number with a leading zero (01 to 31)

<sup>&</sup>lt;sup>92</sup> https://en.wikipedia.org/wiki/ISO\_8601

- ddd : the abbreviated localized day name (e.g. "Mon" to "Sun")
- dddd : the long localized day name (e.g. "Monday" to "Sunday")
- M : the month as number without a leading zero (1 to 12)
- MM : the month as number with a leading zero (01 to 12)
- MMM : the abbreviated localized month name (e.g. "Jan" to "Dec")
- MMMM : the long localized month name (e.g. "January" to "December")
- yy : the year as two digit number (00 to 99)
- yyyy : the year as four digit number.
- h : the hours without a leading 0 (0 to 11 or 0 to 23, depending on am/pm)
- hh : the hours with a leading 0 (00 to 11 or 00 to 23, depending on am/pm)
- m : the minutes without a leading 0 (0 to 59)
- mm : the minutes with a leading 0 (00 to 59)
- s : the seconds without a leading 0 (0 to 59)
- ss : the seconds with a leading 0 (00 to 59)
- ap : use a 12-hour clock instead of a 24-hour clock, with 'ap' replaced by the lowercase localized string for am or pm
- AP : use a 12-hour clock instead of a 24-hour clock, with 'AP' replaced by the uppercase localized string for AM or PM
- aP : use a 12-hour clock instead of a 24-hour clock, with 'aP' replaced by the localized string for AM or PM
- Ap : use a 12-hour clock instead of a 24-hour clock, with 'Ap' replaced by the localized string for AM or PM
- iso : the date with time and timezone. Must be the only format present
- to\_number : convert the date & time into a floating point number (a *timestamp*)
- from\_number : convert a floating point number (a *timestamp*) into an ISO-formatted date. If you want a different date format then add the desired formatting string after from\_number and a colon (:). Example:

format\_date(val, 'from\_number:MMM dd yyyy')

You might get unexpected results if the date you are formatting contains localized month names, which can happen if you changed the date format to contain MMMM. Using format\_date\_field() avoids this problem.

# format\_date\_field

class calibre.utils.formatter\_functions.BuiltinFormatDateField

format\_date\_field(field\_name, format\_string) – format the value in the field field\_name, which must be the lookup name of a date field, either standard or custom. See *format\_date()* (page 188) for the formatting codes. This function is much faster than format\_date() and should be used when you are formatting the value in a field (column). It is also more reliable because it works directly on the underlying date. It can't be used for computed dates or dates in string variables. Examples:

```
format_date_field('pubdate', 'yyyy.MM.dd')
format_date_field('#date_read', 'MMM dd, yyyy')
```

# format\_duration

class calibre.utils.formatter\_functions.BuiltinFormatDuration

format\_duration(value, template, [largest\_unit]) - format the value, a number of seconds, into a string showing weeks, days, hours, minutes, and seconds. If the value is a float then it is rounded to the nearest integer. You choose how to format the value using a template consisting of value selectors surrounded by [ and ] characters. The selectors are:

- [w]: weeks
- [d]: days
- [h]: hours
- [m]: minutes
- [s]: seconds

You can put arbitrary text between selectors.

The following examples use a duration of 2 days (172,800 seconds) 1 hour (3,600 seconds) and 20 seconds, which totals to 176,420 seconds.

- format\_duration(176420, '[d][h][m][s]') will return the value 2d 1h 0m 20s.
- format\_duration(176420, '[h][m][s]') will return the value 49h Om 20s.
- format\_duration(176420, 'Your reading time is [d][h][m][s]') returns the value Your reading time is 49h 0m 20s.
- format\_duration(176420, '[w][d][h][m][s]') will return the value 2d 1h 0m 20s. Note that the zero weeks value is not returned.

If you want to see zero values for items such as weeks in the above example, use an uppercase selector. For example, the following uses 'W' to show zero weeks:

format\_duration(176420, '[W][d][h][m][s]') returns 0w 2d 1h 0m 20s.

By default the text following a value is the selector followed by a space. You can change that to whatever text you want. The format for a selector with your text is the selector followed by a colon followed by text segments separated by '|' characters. You must include any space characters you want in the output.

You can provide from one to three text segments.

- If you provide one segment, as in [w: weeks ] then that segment is used for all values.
- If you provide two segments, as in [w: weeks | week ] then the first segment is used for 0 and more than 1. The second segment is used for 1.
- If you provide three segments, as in [w: weeks | week | weeks ] then the first segment is used for 0, the second segment is used for 1, and the third segment is used for more than 1.

The second form is equivalent to the third form in many languages.

For example, the selector:

- [w: weeks | week | weeks ] produces '0 weeks ', '1 week ', or '2 weeks '.
- [w: weeks | week ] produces '0 weeks ', '1 week ', or '2 weeks '.
- [w: weeks ] produces 0 weeks ', 1 weeks ', or 2 weeks '.

The optional largest\_unit parameter specifies the largest of weeks, days, hours, minutes, and seconds that will be produced by the template. It must be one of the value selectors. This can be useful to truncate a value.

format\_duration(176420, '[h][m][s]', 'd') will return the value 1h 0m 20s instead of 49h 0m 20s.

# format\_number

class calibre.utils.formatter\_functions.BuiltinFormatNumber

format\_number (value, template) – interprets the value as a number and formats that number using a Python formatting template such as  $\{0:5.2f\}$  or  $\{0:,d\}$  or  $\{0:5,.2f\}$ . The formatting template must begin with  $\{0:$  and end with  $\}$  as in the above examples. Exception: you can leave off the leading " $\{0:$ " and trailing " $\}$ " if the format template contains only a format. See the Template Language<sup>93</sup> and the Python<sup>94</sup> documentation for more examples. Returns the empty string if formatting fails.

# human\_readable

class calibre.utils.formatter\_functions.BuiltinHumanReadable

human\_readable(value) - expects the value to be a number and returns a string representing that number in KB, MB, GB, etc.

## rating\_to\_stars

class calibre.utils.formatter\_functions.BuiltinRatingToStars

rating\_to\_stars (value, use\_half\_stars) – Returns the value as string of star ( $\star$ ) characters. The value must be a number between 0 and 5. Set use\_half\_stars to 1 if you want half star characters for fractional numbers available with custom ratings columns.

# Get values from metadata

#### annotation\_count

class calibre.utils.formatter\_functions.BuiltinAnnotationCount

annotation\_count () - return the total number of annotations of all types attached to the current book. This function works only in the GUI and the content server.

### approximate\_formats

class calibre.utils.formatter\_functions.BuiltinApproximateFormats

approximate\_formats() - return a comma-separated list of formats associated with the book. Because the list comes from calibre's database instead of the file system, there is no guarantee that the list is correct, although it probably is. Note that resulting format names are always uppercase, as in EPUB. The approximate\_formats() function is much faster than the formats\_... functions.

This function works only in the GUI. If you want to use these values in save-to-disk or send-to-device templates then you must make a custom "Column built from other columns", use the function in that column's template, and use that column's value in your save/send templates.

### author\_links

class calibre.utils.formatter\_functions.BuiltinAuthorLinks

author\_links (val\_separator, pair\_separator) - returns a string containing a list of authors and those authors' link values in the form: author1 val\_separator author1\_link pair\_separator author2 val\_separator author2\_link etc.

<sup>&</sup>lt;sup>93</sup> https://manual.calibre-ebook.com/template\_lang.html

<sup>94</sup> https://docs.python.org/3/library/string.html#formatstrings

An author is separated from its link value by the val\_separator string with no added spaces. Assuming the val\_separator is a colon, author:link value pairs are separated by the pair\_separator string argument with no added spaces. It is up to you to choose separators that do not occur in author names or links. An author is included even if the author link is empty.

# author\_sorts

class calibre.utils.formatter\_functions.BuiltinAuthorSorts

author\_sorts (val\_separator) - returns a string containing a list of author's sort values for the authors of the book. The sort is the one in the author metadata information, which can be different from the author\_sort in books. The returned list has the form author sort 1 val\_separator author sort 2 etc. with no added spaces. The author sort values in this list are in the same order as the authors of the book. If you want spaces around val\_separator then include them in the val\_separator string.

# booksize

class calibre.utils.formatter\_functions.BuiltinBooksize

booksize() - returns the value of the calibre size field. Returns " if the book has no formats.

This function works only in the GUI. If you want to use this value in save-to-disk or send-to-device templates then you must make a custom "Column built from other columns", use the function in that column's template, and use that column's value in your save/send templates

#### connected\_device\_name

class calibre.utils.formatter\_functions.BuiltinConnectedDeviceName

connected\_device\_name (storage\_location\_key) - if a device is connected then return the device name, otherwise return the empty string. Each storage location on a device has its own device name. The storage\_location\_key names are 'main', 'carda' and 'cardb'. This function works only in the GUI.

# connected\_device\_uuid

class calibre.utils.formatter\_functions.BuiltinConnectedDeviceUUID

connected\_device\_uuid(storage\_location\_key) - if a device is connected then return the device uuid (unique id), otherwise return the empty string. Each storage location on a device has a different uuid. The storage\_location\_key location names are 'main', 'carda' and 'cardb'. This function works only in the GUI.

# current\_library\_name

class calibre.utils.formatter\_functions.BuiltinCurrentLibraryName

current\_library\_name() - return the last name on the path to the current calibre library.

# current\_library\_path

class calibre.utils.formatter\_functions.BuiltinCurrentLibraryPath

 $current\_library\_path()$  – return the full path to the current calibre library.

# current\_virtual\_library\_name

class calibre.utils.formatter\_functions.BuiltinCurrentVirtualLibraryName

current\_virtual\_library\_name() - return the name of the current virtual library if there is one, otherwise the empty string. Library name case is preserved. Example:

program: current\_virtual\_library\_name()

This function works only in the GUI.

### field

class calibre.utils.formatter\_functions.BuiltinField

field (lookup\_name) - returns the value of the metadata field with lookup name lookup\_name. The \$ prefix can be used instead of the function, as in \$tags.

#### formats\_modtimes

class calibre.utils.formatter\_functions.BuiltinFormatsModtimes

formats\_modtimes (date\_format\_string) - return a comma-separated list of colon-separated items FMT:DATE representing modification times for the formats of a book. The date\_format\_string parameter specifies how the date is to be formatted. See the *format\_date()* (page 188) function for details. You can use the *select()* (page 197) function to get the modification time for a specific format. Note that format names are always uppercase, as in EPUB.

# formats\_path\_segments

class calibre.utils.formatter\_functions.BuiltinFormatsPathSegments

formats\_path\_segments(with\_author, with\_title, with\_format, with\_ext, sep) - return parts of the path to a book format in the calibre library separated by sep. The parameter sep should usually be a slash ('/'). One use is to be sure that paths generated in Save to disk and Send to device templates are shortened consistently. Another is to be sure the paths on the device match the paths in the calibre library.

A book path consists of 3 segments: the author, the title including the calibre database id in parentheses, and the format (author - title). Calibre can shorten any of the three because of file name length limitations. You choose which segments to include by passing 1 for that segment. If you don't want a segment then pass 0 or the empty string for that segment. For example, the following returns just the format name without the extension:

formats\_path\_segments(0, 0, 1, 0, '/')

Because there is only one segment the separator is ignored.

If there are multiple formats (multiple extensions) then one of the extensions will be picked at random. If you care about which extension is used then get the path without the extension then add the desired extension to it.

Examples: Assume there is a book in the calibre library with an epub format by Joe Blogs with title 'Help'. It would have the path

Joe Blogs/Help - (calibre\_id)/Help - Joe Blogs.epub

The following shows what is returned for various parameters:

- formats\_path\_segments(0, 0, 1, 0, '/') returns Help Joe Blogs
- formats\_path\_segments(0, 0, 1, 1, '/') returns Help Joe Blogs.epub

- formats\_path\_segments(1, 0, 1, 1, '/') returns Joe Blogs/Help Joe Blogs.epub
- formats\_path\_segments(1, 0, 1, 0, '/') returns Joe Blogs/Help Joe Blogs
- formats\_path\_segments(0, 1, 0, 0, '/') returns Help (calibre\_id)

# formats\_paths

class calibre.utils.formatter\_functions.BuiltinFormatsPaths

formats\_paths ([separator]) - return a separator-separated list of colon-separated items FMT: PATH giving the full path to the formats of a book. The separator argument is optional. If not supplied then the separator is ', ' (comma space). If the separator is a comma then you can use the select() function to get the path for a specific format. Note that format names are always uppercase, as in EPUB.

# formats\_sizes

class calibre.utils.formatter\_functions.BuiltinFormatsSizes

formats\_sizes() - return a comma-separated list of colon-separated FMT:SIZE items giving the sizes of the formats of a book in bytes. You can use the select() function to get the size for a specific format. Note that format names are always uppercase, as in EPUB.

#### has cover

class calibre.utils.formatter\_functions.BuiltinHasCover

has\_cover() - return 'Yes' if the book has a cover, otherwise the empty string.

#### is marked

class calibre.utils.formatter\_functions.BuiltinIsMarked

is\_marked() - check whether the book is *marked* in calibre. If it is then return the value of the mark, either 'true' (lower case) or a comma-separated list of named marks. Returns '' (the empty string) if the book is not marked. This function works only in the GUI.

# language\_codes

class calibre.utils.formatter\_functions.BuiltinLanguageCodes

language\_codes (lang\_strings) - return the language codes<sup>95</sup> for the language names passed in lang\_strings. The strings must be in the language of the current locale. lang\_strings is a comma-separated list.

### language\_strings

#### class calibre.utils.formatter\_functions.BuiltinLanguageStrings

language\_strings (value, localize) - return the language names for the language codes (see here for names and codes<sup>96</sup>) passed in value. Example: {languages:language\_strings()}. If localize is zero, return the strings in English. If localize is not zero, return the strings in the language of the current locale. lang\_codes is a comma-separated list.

<sup>95</sup> https://www.loc.gov/standards/iso639-2/php/code\_list.php

<sup>&</sup>lt;sup>96</sup> https://www.loc.gov/standards/iso639-2/php/code\_list.php

### ondevice

class calibre.utils.formatter\_functions.BuiltinOndevice

ondevice () – return the string 'Yes' if ondevice is set, otherwise return the empty string. This function works only in the GUI. If you want to use this value in save-to-disk or send-to-device templates then you must make a custom "Column built from other columns", use the function in that column's template, and use that column's value in your save/send templates.

## raw\_field

class calibre.utils.formatter\_functions.BuiltinRawField

raw\_field(lookup\_name [, optional\_default]) - returns the metadata field named by lookup\_name without applying any formatting. It evaluates and returns the optional second argument optional\_default if the field's value is undefined (None). The \$\$ prefix can be used instead of the function, as in \$\$pubdate.

## raw\_list

class calibre.utils.formatter\_functions.BuiltinRawList

raw\_list(lookup\_name, separator) - returns the metadata list named by lookup\_name without applying any formatting or sorting, with the items separated by separator.

## series\_sort

class calibre.utils.formatter\_functions.BuiltinSeriesSort

series\_sort() - returns the series sort value.

#### user\_categories

class calibre.utils.formatter\_functions.BuiltinUserCategories

user\_categories() – return a comma-separated list of the user categories that contain this book. This function works only in the GUI. If you want to use these values in save-to-disk or send-to-device templates then you must make a custom *Column built from other columns*, use the function in that column's template, and use that column's value in your save/send templates

# virtual\_libraries

class calibre.utils.formatter\_functions.BuiltinVirtualLibraries

virtual\_libraries() – return a comma-separated list of Virtual libraries that contain this book. This function works only in the GUI. If you want to use these values in save-to-disk or send-to-device templates then you must make a custom "Column built from other columns", use the function in that column's template, and use that column's value in your save/send templates.

#### Iterate over values

# first\_non\_empty

class calibre.utils.formatter\_functions.BuiltinFirstNonEmpty

first\_non\_empty (value [, value]\*) - returns the first value that is not empty. If all values are empty, then the empty string is returned. You can have as many values as you want.

# lookup

class calibre.utils.formatter\_functions.BuiltinLookup

lookup(value, [pattern, key, ]\* else\_key) – The patterns will be checked against the value in order. If a pattern matches then the value of the field named by key is returned. If no pattern matches then the value of the field named by else\_key is returned. See also the *switch()* (page 196) function.

# switch

class calibre.utils.formatter\_functions.BuiltinSwitch

switch(value, [patternN, valueN,]+ else\_value) - for each patternN, valueN pair, checks if the value matches the regular expression patternN and if so returns the associated valueN. If no patternN matches, then else\_value is returned. You can have as many patternN, valueN pairs as you wish. The first match is returned.

# switch if

class calibre.utils.formatter\_functions.BuiltinSwitchIf

switch\_if([test\_expression, value\_expression,]+ else\_expression) - for each test\_expression, value\_expression pair, checks if test\_expression is True (non-empty) and if so returns the result of value\_expression. If no test\_expression is True then the result of else\_expression is returned. You can have as many test\_expression, value\_expression pairs as you want.

# List lookup

# identifier\_in\_list

class calibre.utils.formatter\_functions.BuiltinIdentifierInList

identifier\_in\_list(val, id\_name [, found\_val, not\_found\_val]) - treat val as a list of identifiers separated by commas. An identifier has the format id\_name:value. The id\_name parameter is the id\_name text to search for, either id\_name or id\_name:regexp. The first case matches if there is any identifier matching that id\_name. The second case matches if id\_name matches an identifier and the regexp matches the identifier's value. If found\_val and not\_found\_val are provided then if there is a match then return found\_val, otherwise return not\_found\_val. If found\_val and not\_found\_val are not provided then if there is a match then return the identifier:value pair, otherwise the empty string ('').

# list\_contains

class calibre.utils.formatter\_functions.BuiltinInList

list\_contains (value, separator, [ pattern, found\_val, ]\* not\_found\_val) - interpret the value as a list of items separated by separator, checking the pattern against each item in the list. If the pattern matches an item then return found\_val, otherwise return not\_found\_val. The pair pattern and found\_value can be repeated as many times as desired, permitting returning different values depending on the item's value. The patterns are checked in order, and the first match is returned.

Aliases: in\_list(), list\_contains()

# list\_item

#### class calibre.utils.formatter\_functions.BuiltinListitem

list\_item (value, index, separator) – interpret the value as a list of items separated by separator, returning the 'index'th item. The first item is number zero. The last item has the index -1 as in list\_item(-1, separator). If the item is not in the list, then the empty string is returned. The separator has the same meaning as in the count function, usually comma but is ampersand for author-like lists.

# select

class calibre.utils.formatter\_functions.BuiltinSelect

select (value, key) – interpret the value as a comma-separated list of items with each item having the form id:id\_value (the calibre identifier format). The function finds the first pair with the id equal to key and returns the corresponding id\_value. If no id matches then the function returns the empty string.

## str\_in\_list

class calibre.utils.formatter\_functions.BuiltinStrInList

str\_in\_list(value, separator, [ string, found\_val, ]+ not\_found\_val) - interpret the value as a list of items separated by separator then compare string against each value in the list. The string is not a regular expression. If string is equal to any item (ignoring case) then return the corresponding found\_val. If string contains separators then it is also treated as a list and each subvalue is checked. The string and found\_value pairs can be repeated as many times as desired, permitting returning different values depending on string's value. If none of the strings match then not\_found\_value is returned. The strings are checked in order. The first match is returned.

## List manipulation

#### list\_count

class calibre.utils.formatter\_functions.BuiltinCount

list\_count (value, separator) – interprets the value as a list of items separated by separator and returns the number of items in the list. Most lists use a comma as the separator, but authors uses an ampersand (&).

Examples: {tags:list\_count(,)}, {authors:list\_count(&)}.

Aliases: count(), list\_count()

# list\_count\_field

class calibre.utils.formatter\_functions.BuiltinFieldListCount

list\_count\_field(lookup\_name) - returns the count of items in the field with the lookup\_name lookup\_name.
The field must be multi-valued such as authors or tags, otherwise the function raises an error. This function is much
faster than list\_count() because it operates directly on calibre data without converting it to a string first. Example:
list\_count\_field('tags').

#### list\_count\_matching

class calibre.utils.formatter\_functions.BuiltinListCountMatching

list\_count\_matching(value, pattern, separator) - interprets value as a list of items separated by separator, returning the number of items in the list that match the regular expression pattern.

Aliases: list\_count\_matching(), count\_matching()

# list\_difference

class calibre.utils.formatter\_functions.BuiltinListDifference

list\_difference(list1, list2, separator) - return a list made by removing from list1 any item found in list2 using a case-insensitive comparison. The items in list1 and list2 are separated by separator, as are the items in the returned list.

# list\_equals

class calibre.utils.formatter\_functions.BuiltinListEquals

list\_equals(list1, sep1, list2, sep2, yes\_val, no\_val) – return yes\_val if list1 and list2 contain the same items, otherwise return no\_val. The items are determined by splitting each list using the appropriate separator character (sep1 or sep2). The order of items in the lists is not relevant. The comparison is case-insensitive.

# list\_intersection

class calibre.utils.formatter\_functions.BuiltinListIntersection

list\_intersection(list1, list2, separator) - return a list made by removing from list1 any item not found in list2, using a case-insensitive comparison. The items in list1 and list2 are separated by separator, as are the items in the returned list.

# list\_join

class calibre.utils.formatter\_functions.BuiltinListJoin

list\_join(with\_separator, list1, separator1 [, list2, separator2]\*) - return a list made by joining the items in the source lists (list1 etc) using with\_separator between the items in the result list. Items in each source list[123...] are separated by the associated separator[123...]. A list can contain zero values. It can be a field like publisher that is single-valued, effectively a one-item list. Duplicates are removed using a caseinsensitive comparison. Items are returned in the order they appear in the source lists. If items on lists differ only in letter case then the last is used. All separators can be more than one character.

Example:

```
program:
    list_join('#@#', $authors, '&', $tags, ',')
```

You can use <code>list\_join</code> on the results of previous calls to <code>list\_join</code> as follows:

```
program:
    a = list_join('#@#', $authors, '&', $tags, ',');
    b = list_join('#@#', a, '#@#', $#genre, ',', $#people, '&', 'some value', ',')
```

You can use expressions to generate a list. For example, assume you want items for authors and #genre, but with the genre changed to the word "Genre: " followed by the first letter of the genre, i.e. the genre "Fiction" becomes "Genre: F". The following will do that:

# list\_re

class calibre.utils.formatter\_functions.BuiltinListRe

list\_re(src\_list, separator, include\_re, opt\_replace) - Construct a list by first separating src\_list into items using the separator character. For each item in the list, check if it matches include\_re. If it does then add it to the list to be returned. If opt\_replace is not the empty string then apply the replacement before adding the item to the returned list.

## list\_re\_group

class calibre.utils.formatter\_functions.BuiltinListReGroup

list\_re\_group(src\_list, separator, include\_re, search\_re [,template\_for\_group]\*) - Like list\_re except replacements are not optional. It uses re\_group(item, search\_re, template ...) when doing the replacements.

## list\_remove\_duplicates

class calibre.utils.formatter\_functions.BuiltinListRemoveDuplicates

list\_remove\_duplicates(list, separator) - return a list made by removing duplicate items in list. If items differ only in case then the last is returned. The items in list are separated by separator, as are the items in the returned list.

# list\_sort

class calibre.utils.formatter\_functions.BuiltinListSort

list\_sort (value, direction, separator) - return value sorted using a case-insensitive lexical sort. If direction is zero (number or character), value is sorted ascending, otherwise descending. The list items are separated by separator, as are the items in the returned list.

#### list split

class calibre.utils.formatter\_functions.BuiltinListSplit

 $list\_split(list\_val, sep, id\_prefix) - splits list\_val into separate values using sep, then assigns the values to local variables named id\_prefix_N where N is the position of the value in the list. The first item has position 0 (zero). The function returns the last element in the list.$ 

Example:

```
list_split('one:two:foo', ':', 'var')
```

is equivalent to:

```
var_0 = 'one'
    var_1 = 'two'
    var_2 = 'foo'
```

# list\_union

class calibre.utils.formatter\_functions.BuiltinListUnion

list\_union(list1, list2, separator) - return a list made by merging the items in list1 and list2, removing
duplicate items using a case-insensitive comparison. If items differ in case, the one in list1 is used. The items in list1
and list2 are separated by separator, as are the items in the returned list. Aliases: merge\_lists(), list\_union()

# range

class calibre.utils.formatter\_functions.BuiltinRange

range (start, stop, step, limit) - returns a list of numbers generated by looping over the range specified by the parameters start, stop, and step, with a maximum length of limit. The first value produced is 'start'. Subsequent values  $next_v = current_v + step$ . The loop continues while  $next_v < stop$  assuming step is positive, otherwise while  $next_v > stop$ . An empty list is produced if start fails the test: start >= stop if step is positive. The limit sets the maximum length of the list and has a default of 1000. The parameters start, step, and limit are optional. Calling range() with one argument specifies stop. Two arguments specify start and stop. Three arguments specify start, stop, and step. Four arguments specify start, stop, step and limit.

Examples:

```
range(5) -> '0, 1, 2, 3, 4'
range(0, 5) -> '0, 1, 2, 3, 4'
range(-1, 5) -> '-1, 0, 1, 2, 3, 4'
range(1, 5) -> '1, 2, 3, 4'
range(1, 5, 2) -> '1, 3'
range(1, 5, 2, 5) -> '1, 3'
range(1, 5, 2, 1) -> error(limit exceeded)
```

# subitems

#### class calibre.utils.formatter\_functions.BuiltinSubitems

subitems (value, start\_index, end\_index) – This function breaks apart lists of tag-like hierarchical items such as genres. It interprets the value as a comma- separated list of tag-like items, where each item is a period-separated list. It returns a new list made by extracting from each item the components from start\_index to end\_index, then merging the results back together. Duplicates are removed. The first subitem in a period-separated list has an index of zero. If an index is negative then it counts from the end of the list. As a special case, an end\_index of zero is assumed to be the length of the list.

Examples:

- Assuming a #genre column containing "A.B.C":
  - {#genre:subitems(0,1)} returns "A"
  - {#genre:subitems(0,2)} returns "A.B"
  - {#genre:subitems(1,0)} returns "B.C"
- Assuming a #genre column containing "A.B.C, D.E":
  - {#genre:subitems(0,1)} returns "A, D"
  - {#genre:subitems(0,2)} returns "A.B, D.E"

# sublist

class calibre.utils.formatter\_functions.BuiltinSublist

sublist (value, start\_index, end\_index, separator) - interpret the value as a list of items separated by separator, returning a new list made from the items from start\_index to end\_index. The first item is number zero. If an index is negative, then it counts from the end of the list. As a special case, an end\_index of zero is assumed to be the length of the list.

Examples assuming that the tags column (which is comma-separated) contains "A, B, C":

- {tags:sublist(0,1,\,) } returns "A"
- {tags:sublist(-1,0,\,)} returns "C"
- {tags:sublist(0,-1,\,) } returns "A, B"

## Other

## arguments

class calibre.utils.formatter\_functions.BuiltinArguments

arguments(id[=expression] [, id[=expression]]\*) - Used in a stored template to retrieve the arguments passed in the call. It both declares and initializes local variables with the supplied names, the id``s, making them effectively parameters. The variables are positional; they get the value of the argument given in the call in the same position. If the corresponding argument is not provided in the call then ``arguments() assigns that variable the provided default value. If there is no default value then the variable is set to the empty string.

# assign

class calibre.utils.formatter\_functions.BuiltinAssign

assign (id, value) - assigns value to id, then returns value. id must be an identifier, not an expression. In most cases you can use the = operator instead of this function.

#### globals

class calibre.utils.formatter\_functions.BuiltinGlobals

globals (id[=expression] [, id[=expression]]\*) – Retrieves "global variables" that can be passed into the formatter. The name id is the name of the global variable. It both declares and initializes local variables with the names of the global variables passed in (the id parameters. If the corresponding variable is not provided in the globals then it assigns that variable the provided default value. If there is no default value then the variable is set to the empty string.)

# is\_dark\_mode

class calibre.utils.formatter\_functions.BuiltinIsDarkMode

is\_dark\_mode() - returns '1' if calibre is running in dark mode, '' (the empty string) otherwise. This function can be used in advanced color and icon rules to choose different colors/icons according to the mode. Example:

if is\_dark\_mode() then 'dark.png' else 'light.png' fi

# print

class calibre.utils.formatter\_functions.BuiltinPrint

print (a [, b]\*) - prints the arguments to standard output. Unless you start calibre from the command line (calibre-debug -g), the output will go into a black hole. The print function always returns its first argument.

# set\_globals

class calibre.utils.formatter\_functions.BuiltinSetGlobals

set\_globals(id[=expression] [, id[=expression]]\*) - Sets globalvariables that can be passed into the formatter. The globals are given the name of the id passed in. The value of the id is used unless an expression is provided.

# Recursion

# eval

class calibre.utils.formatter\_functions.BuiltinEval

eval(string) – evaluates the string as a program, passing the local variables. This permits using the template processor to construct complex results from local variables. In Template Program Mode<sup>97</sup>, because the { and } characters are interpreted before the template is evaluated you must use [[ for the { character and ]] for the } character. They are converted automatically. Note also that prefixes and suffixes (the |prefix|suffix syntax) cannot be used in the argument to this function when using Template Program Mode.

# template

class calibre.utils.formatter\_functions.BuiltinTemplate

template(x) - evaluates x as a template. The evaluation is done in its own context, meaning that variables are not shared between the caller and the template evaluation. If not using General Program Mode, because the { and } characters are special, you must use [[ for the { character and ]] for the } character; they are converted automatically. For example, template(\'[[title\_sort]]\') will evaluate the template {title\_sort} and return its value. Note also that prefixes and suffixes (the |prefix|suffix syntax) cannot be used in the argument to this function when using template program mode.

# Relational

# cmp

class calibre.utils.formatter\_functions.BuiltinCmp

cmp (value, y, lt, eq, gt) – compares value and y after converting both to numbers. Returns lt if value <# y, eq if value ==# y, otherwise gt. This function can usually be replaced with one of the numeric compare operators (==#, <#, >#, etc).

# first\_matching\_cmp

 ${\tt class} \ {\tt calibre.utils.formatter\_functions.BuiltinFirstMatchingCmp}$ 

first\_matching\_cmp(val, [ cmp, result, ]\* else\_result) - compares val < cmp in sequence, returning
the associated result for the first comparison that succeeds. Returns else\_result if no comparison succeeds.</pre>

 $<sup>^{97}\</sup> https://manual.calibre-ebook.com/template_lang.html#more-complex-programs-in-template-expressions-template-program-mode$ 

Example:

```
i = 10;
first_matching_cmp(i,5,"small",10,"middle",15,"large","giant")
```

returns "large". The same example with a first value of 16 returns "giant".

#### strcmp

class calibre.utils.formatter\_functions.BuiltinStrcmp

strcmp(x, y, lt, eq, gt) – does a case-insensitive lexical comparison of x and y. Returns lt if x < y, eq if x == y, otherwise gt. This function can often be replaced by one of the lexical comparison operators (==, >, <, etc.)

#### strcmpcase

class calibre.utils.formatter\_functions.BuiltinStrcmpcase

strcmpcase(x, y, lt, eq, gt) - does a case-sensitive lexical comparison of x and y. Returns lt if x < y, eq if x == y, otherwise gt.

Note: This is NOT the default behavior used by calibre, for example, in the lexical comparison operators (==, >, <, etc.). This function could cause unexpected results, preferably use strcmp() whenever possible.

### String manipulation

#### character

class calibre.utils.formatter\_functions.BuiltinCharacter

character(character\_name) - returns the character named by character\_name. For example, character('newline') returns a newline character ('\n'). The supported character names are newline, return, tab, and backslash. This function is used to put these characters into the output of templates.

### check\_yes\_no

class calibre.utils.formatter\_functions.BuiltinCheckYesNo

check\_yes\_no(field\_name, is\_undefined, is\_false, is\_true) - checks if the value of the yes/no field named by the lookup name field\_name is one of the values specified by the parameters, returning 'Yes' if a match is found otherwise returning the empty string. Set the parameter is\_undefined, is\_false, or is\_true to 1 (the number) to check that condition, otherwise set it to 0.

Example: check\_yes\_no("#bool", 1, 0, 1) returns 'Yes' if the yes/no field #bool is either True or undefined (neither True nor False).

More than one of is\_undefined, is\_false, or is\_true can be set to 1.

## contains

class calibre.utils.formatter\_functions.BuiltinContains

contains(value, pattern, text\_if\_match, text\_if\_not\_match) - checks if the value is matched by the regular expression pattern. Returns text\_if\_match if the pattern matches the value, otherwise returns text\_if\_not\_match.

# field\_exists

class calibre.utils.formatter\_functions.BuiltinFieldExists

field\_exists(lookup\_name) - checks if a field (column) with the lookup name lookup\_name exists, returning '1' if so and the empty string if not.

# ifempty

class calibre.utils.formatter\_functions.BuiltinIfempty

ifempty(value, text\_if\_empty) - if the value is not empty then return that value, otherwise return text\_if\_empty.

## re

class calibre.utils.formatter\_functions.BuiltinRe

re(value, pattern, replacement) - return the value after applying the regular expression. All instances of pattern in the value are replaced with replacement. The template language uses case insensitive Python regular expressions<sup>98</sup>.

# re\_group

class calibre.utils.formatter\_functions.BuiltinReGroup

re\_group(value, pattern [, template\_for\_group]\*) - return a string made by applying the regular expression pattern to value and replacing each matched instance with the value returned by the corresponding template. In Template Program Mode<sup>99</sup>, like for the template and the eval functions, you use [[ for { and ]] for }.

The following example looks for a series with more than one word and uppercases the first word:

```
program: re_group(field('series'), "(\S* )(.*)", "{$:uppercase()}", "{$}")'}
```

# shorten

class calibre.utils.formatter\_functions.BuiltinShorten

shorten(value, left\_chars, middle\_text, right\_chars) - Return a shortened version of the value, consisting of left\_chars characters from the beginning of the value, followed by middle\_text, followed by right\_chars characters from the end of the value. left\_chars and right\_chars must be non-negative integers.

Example: assume you want to display the title with a length of at most 15 characters in length. One template that does this is {title:shorten(9, -, 5)}. For a book with the title *Ancient English Laws in the Times of Ivanhoe* the result will be *Ancient E-anhoe*: the first 9 characters of the title, a -, then the last 5 characters. If the value's length is less than left chars + right chars + the length of middle text then the value will be returned unchanged. For example, the title *TheDome* would not be changed.

# strcat

class calibre.utils.formatter\_functions.BuiltinStrcat

strcat  $(a [, b]^*)$  – returns a string formed by concatenating all the arguments. Can take any number of arguments. In most cases you can use the & operator instead of this function.

<sup>&</sup>lt;sup>98</sup> https://docs.python.org/3/library/re.html

<sup>99</sup> https://manual.calibre-ebook.com/template\_lang.html#more-complex-programs-in-template-expressions-template-program-mode

# strcat\_max

class calibre.utils.formatter\_functions.BuiltinStrcatMax

strcat\_max(max, string1 [, prefix2, string2]\*) - Returns a string formed by concatenating the arguments. The returned value is initialized to string1. Strings made from prefix, string pairs are added to the end of the value as long as the resulting string length is less than max. Prefixes can be empty. Returns string1 even if string1 is longer than max. You can pass as many prefix, string pairs as you wish.

### strlen

class calibre.utils.formatter\_functions.BuiltinStrlen

strlen (value) - Returns the length of the string value.

#### substr

class calibre.utils.formatter\_functions.BuiltinSubstr

substr(value, start, end) – returns the start'th through the end'th characters of value. The first character in value is the zero'th character. If end is negative then it indicates that many characters counting from the right. If end is zero, then it indicates the last character. For example, substr('12345', 1, 0) returns '2345', and substr('12345', 1, -1) returns '234'.

## swap\_around\_articles

class calibre.utils.formatter\_functions.BuiltinSwapAroundArticles

swap\_around\_articles(value, separator) - returns the value with articles moved to the end, separated by a semicolon. The value can be a list, in which case each item in the list is processed. If the value is a list then you must provide the separator. If no separator is provided or the separator is the empty string then the value is treated as being a single value, not a list. The *articles* are those used by calibre to generate the title\_sort.

#### swap\_around\_comma

class calibre.utils.formatter\_functions.BuiltinSwapAroundComma

swap\_around\_comma (value) - given a value of the form B, A, return A B. This is most useful for converting names in LN, FN format to FN LN. If there is no comma in the value then the function returns the value unchanged.

### test

class calibre.utils.formatter\_functions.BuiltinTest

test(value, text\_if\_not\_empty, text\_if\_empty) - return text\_if\_not\_empty if the value is not empty,
otherwise return text\_if\_empty.

# transliterate

class calibre.utils.formatter\_functions.BuiltinTransliterate

transliterate (value) – Return a string in a latin alphabet formed by approximating the sound of the words in value. For example, if value is Фёдор Миха́йлович Достоевский this function returns Fiodor Mikhailovich Dostoievskii.

# **URL** functions

# encode\_for\_url

class calibre.utils.formatter\_functions.BuiltinEncodeForURL

encode\_for\_url(value, use\_plus) - returns the value encoded for use in a URL as specified by use\_plus. The value is first URL-encoded. Next, if use\_plus is 0 then spaces are replaced by '+' (plus) signs. If it is 1 then spaces are replaced by %20.

If you do not want the value to be encoding but to have spaces replaced then use the re() (page 204) function, as in re(\$series, '', '%20')

See also the functions *make\_url()* (page 206), *make\_url\_extended()* (page 206) and *query\_string()* (page 207).

# make\_url

class calibre.utils.formatter\_functions.BuiltinMakeUrl

make\_url(path, [query\_name, query\_value]+) - this function is the easiest way to construct a query URL. It uses a path, the web site and page you want to query, and query\_name, query\_value pairs from which the query is built. In general, the query\_value must be URL-encoded. With this function it is always encoded and spaces are always replaced with '+' signs.

At least one query\_name, query\_value pair must be provided.

Example: constructing a Wikipedia search URL for the author Niccolò Machiavelli:

make\_url('https://en.wikipedia.org/w/index.php', 'search', 'Niccolò Machiavelli')

returns

https://en.wikipedia.org/w/index.php?search=Niccol%C3%B2+Machiavelli

If you are writing a custom column book details URL template then use <code>\$item\_name</code> or <code>field('item\_name')</code> to obtain the value of the field that was clicked on. Example: if *Niccolò Machiavelli* was clicked then you can construct the URL using:

make\_url('https://en.wikipedia.org/w/index.php', 'search', \$item\_name)

See also the functions *make\_url\_extended()* (page 206), *query\_string()* (page 207) and *encode\_for\_url()* (page 206).

# make\_url\_extended

class calibre.utils.formatter\_functions.BuiltinMakeUrlExtended

 $make\_url\_extended(...)$  – this function is similar to  $make\_url()$  (page 206) but gives you more control over the URL components. The components of a URL are

# scheme:://authority/path?query string.

See Uniform Resource Locator<sup>100</sup> on Wikipedia for more detail.

The function has two variants:

make\_url\_extended(scheme, authority, path, [query\_name, query\_value]+)

#### and

100 https://en.wikipedia.org/wiki/URL

make\_url\_extended(scheme, authority, path, query\_string)

This function returns a URL constructed from the scheme, authority, path, and either the query\_string or a query string constructed from the query argument pairs. The authority can be empty, which is the case for calibre scheme URLs. You must supply either a query\_string or at least one query\_name, query\_value pair. If you supply query\_string and it is empty then the resulting URL will not have a query string section.

Example 1: constructing a Wikipedia search URL for the author Niccolò Machiavelli:

returns

https://en.wikipedia.org/w/index.php?search=Niccol%C3%B2+Machiavelli

See the *query\_string()* (page 207) function for an example using make\_url\_extended() with a query\_string.

If you are writing a custom column book details URL template then use <code>\$item\_name</code> or <code>field('item\_name')</code> to obtain the value of the field that was clicked on. Example: if *Niccolò Machiavelli* was clicked on then you can construct the URL using :

make\_url\_extended('https', 'en.wikipedia.org', '/w/index.php', 'search', \$item\_name')

See also the functions *make\_url()* (page 206), *query\_string()* (page 207) and *encode\_for\_url()* (page 206).

# query\_string

class calibre.utils.formatter\_functions.BuiltinQueryString

query\_string([query\_name, query\_value, how\_to\_encode]+) - returns a URL query string constructed from the query\_name, query\_value, how\_to\_encode triads. A query string is a series of items where each item looks like query\_name=query\_value where query\_value is URL-encoded as instructed. The query items are separated by '&' (ampersand) characters.

If how\_to\_encode is 0 then query\_value is encoded and spaces are replaced with '+' (plus) signs. If how\_to\_encode is 1 then query\_value is encoded with spaces replaced by %20. If how\_to\_encode is 2 then query\_value is returned unchanged; no encoding is done and spaces are not replaced. If you want query\_value not to be encoded but spaces to be replaced then use the re() (page 204) function, as in re(\$series, '', '%20')

You use this function if you need specific control over how the parts of the query string are constructed. You could then use the resultingquery string in *make\_url\_extended()* (page 206), as in

giving you

https://your\_host/your\_path?encoded=Hendrik+B%C3%A4%C3%9Fler&unencoded=Hendrik Bäßler

You must have at least one query\_name, query\_value, how\_to\_encode triad, but can have as many as you wish.

The returned value is a URL query string with all the specified items, for example: name1=val1[&nameN=valN]\*. Note that the '?' *path | query string* separator is not included in the returned result.

If you are writing a custom column book details URL template then use <code>%item\_name</code> or <code>field('item\_name')</code> to obtain the unencoded value of the field that was clicked. You also have <code>item\_value\_quoted</code> where the value is already

encoded with plus signs replacing spaces, and item\_value\_no\_plus where the value is already encoded with %20 replacing spaces.

See also the functions make\_url() (page 206), make\_url\_extended() (page 206) and encode\_for\_url() (page 206).

# to\_hex

class calibre.utils.formatter\_functions.BuiltinToHex

to\_hex (val) - returns the string val encoded into hex. This is useful when constructing calibre URLs.

#### urls\_from\_identifiers

class calibre.utils.formatter\_functions.BuiltinUrlsFromIdentifiers

urls\_from\_identifiers (identifiers, sort\_results) - given a comma-separated list of identifiers, where an identifier is a colon-separated pair of values (id\_name:id\_value), returns a comma-separated list of HTML URLs generated from the identifiers. The list not sorted if sort\_results is 0 (character or number), otherwise it is sorted alphabetically by the identifier name. The URLs are generated in the same way as the built-in identifiers column when shown in Book Details.

# API of the Metadata objects

The python implementation of the template functions is passed in a Metadata object. Knowing it's API is useful if you want to define your own template functions.

A class representing all the metadata for a book. The various standard metadata fields are available as attributes of this object. You can also stick arbitrary attributes onto this object.

Metadata from custom columns should be accessed via the get() method, passing in the lookup name for the column, for example: "#mytags".

Use the *is\_null()* (page 208) method to test if a field is null.

This object also has functions to format fields into strings.

The list of standard metadata fields grows with time is in STANDARD\_METADATA\_FIELDS (page 209).

Please keep the method based API of this class to a minimum. Every method becomes a reserved field name.

is\_null(field)

Return True if the value of field is null in this object. 'null' means it is unknown or evaluates to False. So a title of \_('Unknown') is null or a language of 'und' is null.

Be careful with numeric fields since this will return True for zero as well as None.

Also returns True if the field does not exist.

deepcopy (class\_generator=<function Metadata.<lambda>>)

Do not use this method unless you know what you are doing, if you want to create a simple clone of this object, use deepcopy\_metadata() instead. Class\_generator must be a function that returns an instance of Metadata or a subclass of it.

get\_identifiers()

Return a copy of the identifiers dictionary. The dict is small, and the penalty for using a reference where a copy is needed is large. Also, we don't want any manipulations of the returned dict to show up in the book.

# set\_identifiers(identifiers)

Set all identifiers. Note that if you previously set ISBN, calling this method will delete it.

### set\_identifier(typ, val)

If val is empty, deletes identifier of type typ

# standard\_field\_keys()

return a list of all possible keys, even if this book doesn't have them

# custom\_field\_keys()

return a list of the custom fields in this book

### all\_field\_keys()

All field keys known by this instance, even if their value is None

### metadata\_for\_field(key)

return metadata describing a standard or custom field.

# all\_non\_none\_fields()

Return a dictionary containing all non-None metadata fields, including the custom ones.

## get\_standard\_metadata(field, make\_copy)

return field metadata from the field if it is there. Otherwise return None. field is the key name, not the label. Return a copy if requested, just in case the user wants to change values in the dict.

# get\_all\_standard\_metadata(make\_copy)

return a dict containing all the standard field metadata associated with the book.

# get\_all\_user\_metadata(make\_copy)

return a dict containing all the custom field metadata associated with the book.

# get\_user\_metadata(field, make\_copy)

return field metadata from the object if it is there. Otherwise return None. field is the key name, not the label. Return a copy if requested, just in case the user wants to change values in the dict.

# set\_all\_user\_metadata(metadata)

store custom field metadata into the object. Field is the key name not the label

### set\_user\_metadata (field, metadata)

store custom field metadata for one column into the object. Field is the key name not the label

# remove\_stale\_user\_metadata(other\_mi)

Remove user metadata keys (custom column keys) if they don't exist in 'other\_mi', which must be a metadata object

#### template\_to\_attribute (other, ops)

Takes a list [(src,dest), (src,dest)], evaluates the template in the context of other, then copies the result to self[dest]. This is on a best-efforts basis. Some assignments can make no sense.

#### smart\_update(other, replace\_metadata=False)

Merge the information in *other* into self. In case of conflicts, the information in *other* takes precedence, unless the information in *other* is NULL.

# format\_field(key, series\_with\_index=True)

Returns the tuple (display\_name, formatted\_value)

# to\_html()

A HTML representation of this object.

 $\texttt{calibre.ebooks.metadata.book.base.STANDARD\_METADATA\_FIELDS}$ 

The set of standard metadata fields.

```
. . .
All fields must have a NULL value represented as None for simple types,
an empty list/dictionary for complex types and (None, None) for cover_data
. . .
SOCIAL_METADATA_FIELDS = frozenset((
   'tags', # Ordered list
'rating', # A floating point number between 0 and 10
'comments', # A simple HTML enabled string
   'series',
                       # A simple string
   'series_index',  # A floating point number
    # Of the form { scheme1:value1, scheme2:value2}
    # For example: {'isbn':'123456789', 'doi':'xxxx', ... }
   'identifiers',
))
. . .
The list of names that convert to identifiers when in get and set.
. . .
TOP_LEVEL_IDENTIFIERS = frozenset((
   'isbn',
))
PUBLICATION_METADATA_FIELDS = frozenset((
   'title', # title must never be None. Should be _('Unknown')
    # Pseudo field that can be set, but if not set is auto generated
    # from title and languages
    'title_sort',
                       # Ordered list. Must never be None, can be [_('Unknown')]
    'authors',
    'author_sort_map', # Map of sort strings for each author
    # Pseudo field that can be set, but if not set is auto generated
    # from authors and languages
    'author_sort',
    'book_producer',
                       # Dates and times must be timezone aware
    'timestamp',
   'pubdate',
    'last_modified',
    'rights',
    # So far only known publication type is periodical:calibre
    # If None, means book
    'publication_type',
    'uuid',
                         # A UUID usually of type 4
    'languages',
                      # ordered list of languages in this publication
    'publisher',
                       # Simple string, no special semantics
    # Absolute path to image file encoded in filesystem_encoding
    'cover',
    # Of the form (format, data) where format is, e.g. 'jpeg', 'png', 'gif'...
    'cover_data',
```

(continues on next page)

```
(continued from previous page)
```

```
# Either thumbnail data, or an object with the attribute
    # image_path which is the path to an image file, encoded
    # in filesystem_encoding
   'thumbnail',
))
BOOK_STRUCTURE_FIELDS = frozenset((
   # These are used by code, Null values are None.
    'toc', 'spine', 'guide', 'manifest',
))
USER_METADATA_FIELDS = frozenset((
   # A dict of dicts similar to field_metadata. Each field description dict
    # also contains a value field with the key #value#.
   'user_metadata',
))
DEVICE_METADATA_FIELDS = frozenset((
   'device_collections', # Ordered list of strings
                           # Unicode, / separated
    'lpath',
   'size',
                           # In bytes
    'mime'.
                           # Mimetype of the book file being represented
))
CALIBRE_METADATA_FIELDS = frozenset((
    'application_id', # An application id, currently set to the db_id.
    'db_id',
                      # the calibre primary key of the item.
                  # list of formats (extensions) for this book
    'formats',
    # a dict of user category names, where the value is a list of item names
    # from the book that are in that category
   'user_categories',
    # a dict of items to associated hyperlink
    'link_maps',
))
ALL_METADATA_FIELDS =
                          SOCIAL_METADATA_FIELDS.union(
                           PUBLICATION_METADATA_FIELDS).union(
                          BOOK_STRUCTURE_FIELDS).union(
                          USER_METADATA_FIELDS).union(
                           DEVICE_METADATA_FIELDS).union(
                           CALIBRE_METADATA_FIELDS)
# All fields except custom fields
STANDARD METADATA FIELDS = SOCIAL METADATA FIELDS.union(
                          PUBLICATION_METADATA_FIELDS).union(
                          BOOK_STRUCTURE_FIELDS).union(
                           DEVICE_METADATA_FIELDS).union(
                           CALIBRE_METADATA_FIELDS)
# Metadata fields that smart update must do special processing to copy.
SC_FIELDS_NOT_COPIED = frozenset(('title', 'title_sort', 'authors',
                                      'author_sort', 'author_sort_map',
```

(continues on next page)

(continued from previous page)

```
'cover_data', 'tags', 'languages',
                                     'identifiers'))
# Metadata fields that smart update should copy only if the source is not None
SC_FIELDS_COPY_NOT_NULL = frozenset(('device_collections', 'lpath', 'size', 'comments
# Metadata fields that smart update should copy without special handling
                       SOCIAL_METADATA_FIELDS.union(
SC_COPYABLE_FIELDS =
                          PUBLICATION_METADATA_FIELDS).union(
                          BOOK_STRUCTURE_FIELDS).union(
                          DEVICE_METADATA_FIELDS).union(
                          CALIBRE_METADATA_FIELDS) - \
                          SC_FIELDS_NOT_COPIED.union(
                          SC_FIELDS_COPY_NOT_NULL)
SERIALIZABLE_FIELDS =
                          SOCIAL_METADATA_FIELDS.union(
                          USER_METADATA_FIELDS).union(
                          PUBLICATION_METADATA_FIELDS).union(
                          CALIBRE_METADATA_FIELDS).union(
                          DEVICE_METADATA_FIELDS) - \
                          frozenset(('device_collections', 'formats',
                              'cover_data'))
# these are rebuilt when needed
```

# 10.4 All about using regular expressions in calibre

Regular expressions are features used in many places in calibre to perform sophisticated manipulation of e-book content and metadata. This tutorial is a gentle introduction to getting you started with using regular expressions in calibre.

```
Contents
First, a word of warning and a word of courage (page 213)
Where in calibre can you use regular expressions? (page 213)
What on earth is a regular expression? (page 213)
Care to explain? (page 213)
That doesn't sound too bad. What's next? (page 214)
Hey, neat! This is starting to make sense! (page 214)
Well, these special characters are very neat and all, but what if I wanted to match a dot or a question mark? (page 214)
So, what are the most useful sets? (page 215)
But if I had a few varying strings I wanted to match, things get complicated? (page 215)
You missed... (page 216)
In the beginning, you said there was a way to make a regular expression case insensitive? (page 216)
```

- I think I'm beginning to understand these regular expressions now... how do I use them in calibre? (page 216)
  - Conversions (page 216)
  - Adding books (page 217)
  - Bulk editing metadata (page 217)
- *Quick reference* (page 217)
- *Credits* (page 222)

# 10.4.1 First, a word of warning and a word of courage

This is, inevitably, going to be somewhat technical- after all, regular expressions are a technical tool for doing technical stuff. I'm going to have to use some jargon and concepts that may seem complicated or convoluted. I'm going to try to explain those concepts as clearly as I can, but really can't do without using them at all. That being said, don't be discouraged by any jargon, as I've tried to explain everything new. And while regular expressions themselves may seem like an arcane, black magic (or, to be more prosaic, a random string of mumbo-jumbo letters and signs), I promise that they are not all that complicated. Even those who understand regular expressions really well have trouble reading the more complex ones, but writing them isn't as difficult- you construct the expression step by step. So, take a step and follow me into the rabbit hole.

# 10.4.2 Where in calibre can you use regular expressions?

There are a few places calibre uses regular expressions. There's the *Search & replace* in conversion options, metadata detection from filenames in the import settings and Search & replace when editing the metadata of books in bulk. The calibre book editor can also use regular expressions in its *Search and replace* feature. Finally, you can use regular expressions when searching the calibre book list and when searching inside the calibre E-book viewer.

## 10.4.3 What on earth is a regular expression?

A regular expression is a way to describe sets of strings. A single regular expression can *match* a number of different strings. This is what makes regular expression so powerful – they are a concise way of describing a potentially large number of variations.

## Note

I'm using string here in the sense it is used in programming languages: a string of one or more characters, characters including actual characters, numbers, punctuation and so-called whitespace (linebreaks, tabulators etc.). Please note that generally, uppercase and lowercase characters are not considered the same, thus "a" being a different character from "A" and so forth. In calibre, regular expressions are case insensitive in the Search bar, but not in the conversion options. There's a way to make every regular expression case insensitive, but we'll discuss that later. It gets complicated because regular expressions allow for variations in the strings it matches, so one expression can match multiple strings, which is why people bother using them at all. More on that in a bit.

# 10.4.4 Care to explain?

Well, that's why we're here. First, this is the most important concept in regular expressions: A string by itself is a regular expression that matches itself. That is to say, if I wanted to match the string "Hello, World!" using a regular expression, the regular expression to use would be Hello, World!. And yes, it really is that simple. You'll notice, though, that this only matches the exact string "Hello, World!", not e.g. "Hello, world!" or "hello, world!" or any other such variation.

# 10.4.5 That doesn't sound too bad. What's next?

Next is the beginning of the really good stuff. Remember where I said that regular expressions can match multiple strings? This is where it gets a little more complicated. Say, as a somewhat more practical exercise, the e-book you wanted to convert had a nasty footer counting the pages, like "Page 5 of 423". Obviously the page number would rise from 1 to 423, thus you'd have to match 423 different strings, right? Wrong, actually: regular expressions allow you to define sets of characters that are matched: To define a set, you put all the characters you want to be in the set into square brackets. So, for example, the set [abc] would match either the character "a", "b" or "c". *Sets will always only match one of the characters in the set*. They "understand" character ranges, that is, if you wanted to match all the lower case characters, you'd use the set [a-z] for lower- and uppercase characters you'd use [a-zA-Z] and so on. Got the idea? So, obviously, using the expression Page [0-9] of 423 you'd be able to match the first 9 pages, thus reducing the expressions needed to three: The second expression Page [0-9] of 423 would match all two-digit page numbers, and I'm sure you can guess what the third expression would look like. Yes, go ahead. Write it down.

# 10.4.6 Hey, neat! This is starting to make sense!

I was hoping you'd say that. But brace yourself, now it gets even better! We just saw that using sets, we could match one of several characters at once. But you can even repeat a character or set, reducing the number of expressions needed to handle the above page number example to one. Yes, ONE! Excited? You should be! It works like this: Some so-called special characters, "+", "?" and "\*", *repeat the single element preceding them*. (Element means either a single character, a character set, an escape sequence or a group (we'll learn about those last two later)- in short, any single entity in a regular expression). These characters are called wildcards or quantifiers. To be more precise, "?" matches *0 or 1* of the preceding element, "\*" matches *0 or more* of the preceding element and "+" matches *1 or more* of the preceding element. A few examples: The expression a? would match either "" (which is the empty string, not strictly useful in this case) or "a", the expression a\* would match "", "a", "aa" or any number of a's in a row, and, finally, the expression a+ would match "a", "aa" or any number of a's in a row (Note: it wouldn't match the empty string]). Same deal for sets: The expression [0-9]+ would match *every integer number there is*! I know what you're thinking, and you're right: If you use that in the above case of matching page numbers, wouldn't that be the single one expression to match all the page numbers? Yes, the expression Page [0-9]+ of 423 would match every page number in that book!

#### **1** Note

A note on these quantifiers: They generally try to match as much text as possible, so be careful when using them. This is called "greedy behaviour"- I'm sure you get why. It gets problematic when you, say, try to match a tag. Consider, for example, the string "<p class="calibre2">Title here" and let's say you'd want to match the opening tag (the part between the first pair of angle brackets, a little more on tags later). You'd think that the expression <p.\*> would match that tag, but actually, it matches the whole string! (The character "." is another special character. It matches anything *except* linebreaks, so, basically, the expression .\* would match any single line you can think of). Instead, try using <p.\*?> which makes the quantifier "\*" non-greedy. That expression would only match the first opening tag, as intended. There's actually another way to accomplish this: The expression  $<p[^>] = .*? > will match that same opening tag- you'll see why after the next section. Just note that there quite frequently is more than one way to write a regular expression.$ 

# 10.4.7 Well, these special characters are very neat and all, but what if I wanted to match a dot or a question mark?

You can of course do that: Just put a backslash in front of any special character and it is interpreted as the literal character, without any special meaning. This pair of a backslash followed by a single character is called an escape sequence, and the act of putting a backslash in front of a special character is called escaping that character. An escape sequence is interpreted as a single element. There are of course escape sequences that do more than just escaping special characters, for example "\t" means a tabulator. We'll get to some of the escape sequences later. Oh, and by the way, concerning those special characters: Consider any character we discuss in this introduction as having some function to be special and thus needing to be escaped if you want the literal character.

# 10.4.8 So, what are the most useful sets?

Knew you'd ask. Some useful sets are [0-9] matching a single number, [a-z] matching a single lowercase letter, [a-zA-z] matching a single letter and [a-zA-z0-9] matching a single letter or number. You can also use an escape sequence as shorthand:

\d

is equivalent to [0-9]

\w

is equivalent to [a-zA-Z0-9\_]

\s

is equivalent to any whitespace

## Note

"Whitespace" is a term for anything that won't be printed. These characters include space, tabulator, line feed, form feed, carriage return, non-breaking spaces, etc.

## 1 Note

The upper and lower case sets may match both upper and lowercase if the setting to make searches case insensitive is enabled. Such settings are found, for instance in Preferences->Searching in calibre itself and on the Search panel in the calibre *E-book viewer* as well as the calibre *Edit book* tool.

As a last note on sets, you can also define a set as any character *but* those in the set. You do that by including the character "^" as the *very first character in the set*. Thus, [^a] would match any character excluding "a". That's called complementing the set. Those escape sequence shorthands we saw earlier can also be complemented: "\D" means any non-number character, thus being equivalent to [^0-9]. The other shorthands can be complemented by, you guessed it, using the respective uppercase letter instead of the lowercase one. So, going back to the example  $p[^>] *>$  from the previous section, now you can see that the character set it's using tries to match any character except for a closing angle bracket.

# 10.4.9 But if I had a few varying strings I wanted to match, things get complicated?

Fear not, life still is good and easy. Consider this example: The book you're converting has "Title" written on every odd page and "Author" written on every even page. Looks great in print, right? But in e-books, it's annoying. You can group whole expressions in normal parentheses, and the character "|" will let you match *either* the expression to its right *or* the one to its left. Combine those and you're done. Too fast for you? Okay, first off, we group the expressions for odd and even pages, thus getting (Title) (Author) as our two needed expressions. Now we make things simpler by using the vertical bar ("|" is called the vertical bar character): If you use the expression (Title|Author) you'll either get a match for "Title" (on the odd pages) or you'd match "Author" (on the even pages). Well, wasn't that easy?

You can, of course, use the vertical bar without using grouping parentheses, as well. Remember when I said that quantifiers repeat the element preceding them? Well, the vertical bar works a little differently: The expression "TitlelAuthor" will also match either the string "Title" or the string "Author", just as the above example using grouping. *The vertical bar selects between the entire expression preceding and following it.* So, if you wanted to match the strings "Calibre" and "calibre" and wanted to select only between the upper- and lowercase "c", you'd have to use the expression (c|C) alibre, where the grouping ensures that only the "c" will be selected. If you were to use c|Calibre, you'd get a match on the string "Calibre", which isn't what we wanted. In short: If in doubt, use grouping together with the vertical bar.

# 10.4.10 You missed...

... wait just a minute, there's one last, really neat thing you can do with groups. If you have a group that you previously matched, you can use references to that group later in the expression: Groups are numbered starting with 1, and you reference them by escaping the number of the group you want to reference, thus, the fifth group would be referenced as 5. So, if you searched for  $([^]+) \\ 1$  in the string "Test Test", you'd match the whole string!

# 10.4.11 In the beginning, you said there was a way to make a regular expression case insensitive?

Yes, I did, thanks for paying attention and reminding me. You can tell calibre how you want certain things handled by using something called flags. You include flags in your expression by using the special construct (?flags go here) where, obviously, you'd replace "flags go here" with the specific flags you want. For ignoring case, the flag is i, thus you include (?i) in your expression. Thus, (?i)test would match "Test", "tEst", "TEst" and any case variation you could think of.

Another useful flag lets the dot match any character at all, *including* the newline, the flag s. If you want to use multiple flags in an expression, just put them in the same statement: (?is) would ignore case and make the dot match all. It doesn't matter which flag you state first, (?si) would be equivalent to the above.

# 10.4.12 I think I'm beginning to understand these regular expressions now... how do I use them in calibre?

## Conversions

Let's begin with the conversion settings, which is really neat. In the *Search & replace* part, you can input a regexp (short for regular expression) that describes the string that will be replaced during the conversion. The neat part is the wizard. Click on the wizard staff and you get a preview of what calibre "sees" during the conversion process. Scroll down to the string you want to remove, select and copy it, paste it into the regexp field on top of the window. If there are variable parts, like page numbers or so, use sets and quantifiers to cover those, and while you're at it, remember to escape special characters, if there are some. Hit the button labeled *Test* and calibre highlights the parts it would replace were you to use the regexp. Once you're satisfied, hit OK and convert. Be careful if your conversion source has tags like this example:

```
Maybe, but the cops feel like you do, Anita. What's one more dead vampire?
New laws don't change that. 
 <b class="calibre2">Generated by ABC Amber LIT Conv
<a href="http://www.processtext.com/abclit.html" class="calibre3">erter,
http://www.processtext.com/abclit.html" class="calibre3">erter,
http://www.processtext.com/abclit.html</a></b>
 It had only been two years since Addison v. Clark.
The court case gave us a revised version of what life was
```

(shamelessly ripped out of this thread<sup>101</sup>). You'd have to remove some of the tags as well. In this example, I'd recommend beginning with the tag <b class="calibre2">, now you have to end with the corresponding closing tag (opening tags are <tag>, closing tags are </tag>), which is simply the next </b> in this case. (Refer to a good HTML manual or ask in the forum if you are unclear on this point). The opening tag can be described using <b.\*?>, the closing tag using </b>, thus we could remove everything between those tags using <b.\*?>.\*?</b>. But using this expression would be a bad idea, because it removes everything enclosed by <b> tags (which, by the way, render the enclosed text in bold print), and it's a fair bet that we'll remove portions of the book in this way. Instead, include the beginning of the enclosed string as well, making the regular expression <br/> <br/> \*?>\s\*Generated\s+by\s+ABC\s+Amber\s+LIT.\*?</br/> The \s with quantifiers are included here instead of explicitly using the spaces as seen in the string to catch any variations of the string that might occur. Remember to check what calibre will remove to make sure you don't remove any portions you want to keep if you test a new expression. If you only check one occurrence, you might miss a mismatch somewhere else in the text. Also note that should you accidentally remove more or fewer tags than you actually wanted to, calibre tries to repair the damaged code after doing the removal.

<sup>&</sup>lt;sup>101</sup> https://www.mobileread.com/forums/showthread.php?t=75594"

## Adding books

Another thing you can use regular expressions for is to extract metadata from filenames. You can find this feature in the "Adding books" part of the settings. There's a special feature here: You can use field names for metadata fields, for example (?P<title>) would indicate that calibre uses this part of the string as book title. The allowed field names are listed in the windows, together with another nice test field. An example: Say you want to import a whole bunch of files named like Classical Texts: The Divine Comedy by Dante Alighieri.mobi. (Obviously, this is already in your library, since we all love classical italian poetry) or Science Fiction epics: The Foundation Trilogy by Isaac Asimov.epub. This is obviously a naming scheme that calibre won't extract any meaningful data out of - its standard expression for extracting metadata is (?P<title>.+) - (?P<author>[^]+). A regular expression that works here would be [a-zA-Z]+: (?P<title>.+) by (?P<author>.+). Please note that, inside the group for the metadata field, you need to use expressions to describe what the field actually matches. And also note that, when using the test field calibre provides, you need to add the file extension to your testing filename, otherwise you won't get any matches at all, despite using a working expression.

## **Bulk editing metadata**

The last part is regular expression *Search and replace* in metadata fields. You can access this by selecting multiple books in the library and using bulk metadata edit. Be very careful when using this last feature, as it can do **Very Bad Things** to your library! Doublecheck that your expressions do what you want them to using the test fields, and only mark the books you really want to change! In the regular expression search mode, you can search in one field, replace the text with something and even write the result into another field. A practical example: Say your library contained the books of Frank Herbert's Dune series, named after the fashion Dune 1 – Dune, Dune 2 – Dune Messiah and so on. Now you want to get Dune into the series field. You can do that by searching for (.\*?) \d+ – .\* in the title field and replacing it with \1 in the series field. See what I did there? That's a reference to the first group you're replacing the series field with. Now that you have the series all set, you only need to do another search for .\*? – in the title field and replace it with "" (an empty string), again in the title field, and your metadata is all neat and tidy. Isn't that great? By the way, instead of replacing the entire field, you can also append or prepend to the field, so, if you *wanted* the book title to be prepended with series info, you could do that as well. As you by now have undoubtedly noticed, there's a checkbox labeled *Case sensitive*, so you won't have to use flags to select behaviour here.

Well, that just about concludes the very short introduction to regular expressions. Hopefully I'll have shown you enough to at least get you started and to enable you to continue learning by yourself- a good starting point would be the Python documentation for regexps<sup>102</sup>.

One last word of warning, though: Regexps are powerful, but also really easy to get wrong. calibre provides really great testing possibilities to see if your expressions behave as you expect them to. Use them. Try not to shoot yourself in the foot. (God, I love that expression...). But should you, despite the warning, injure your foot (or any other body parts), try to learn from it.

# **10.4.13 Quick reference**

#### Quick reference for regexp syntax

This checklist summarizes the most commonly used/hard to remember parts of the regexp engine available in most parts of calibre.

## Contents

- Character classes (page 218)
- Shorthand character classes (page 219)

<sup>&</sup>lt;sup>102</sup> https://docs.python.org/library/re.html

- *The quantifiers* (page 219)
- *Greed* (page 219)
- *Alternation* (page 219)
- *Exclusion* (page 219)
- Anchors (page 220)
- Groups (page 220)
- Lookarounds (page 220)
- *Recursion* (page 221)
- Special characters (page 221)
- *Meta-characters* (page 221)
- *Modes* (page 222)

## **Character classes**

Character classes are useful to represent different groups of characters, succinctly.

Examples:

Represen- tation	Class
[a-z]	Lowercase letters. Does not include characters with accent mark and ligatures
[a-z0-9]	Lowercase letters from a to z or numbers from 0 to 9
[A-Za-z-]	Uppercase or lowercase letters, or a dash. To include the dash in a class, you must put it at the beginning or at the end so as not to confuse it with the hyphen that specifies a range of characters
[^0-9]	Any character except a digit. The caret (^) placed at the beginning of the class excludes the characters of the class (complemented class)
[[a-z][a	The lowercase consonants. A class can be included in a class. The characters exclude what follows them
[\w[\ d_]]	All letters (including foreign accented characters). Abbreviated classes can be used inside a class

#### Example:

 $<[\,^{<>}]\,+>$  to select an HTML tag

# Shorthand character classes

Representa- tion	Class
∖d	A digit (same as [0-9])
\D	Any non-numeric character (same as [^0-9])
\w	An alphanumeric character ([a-zA-Z0-9]) including characters with accent mark and ligatures
$\setminus W$	Any "non-word" character
\s	Space, non-breaking space, tab, return line
\S	Any "non-whitespace" character
	Any character except newline. Use the "dot all" checkbox or the (?s) regexp modifier to include the newline character.

### The quantifiers

Quantifier	Number of occurrences of the expression preceding the quantifier
?	0 or 1 occurrence of the expression. Same as {0,1}
+	1 or more occurrences of the expression. Same as {1, }
*	0, 1 or more occurrences of the expression. Same as {0, }
{n}	Exactly n occurrences of the expression
{min,max}	Number of occurrences between the minimum and maximum values included
{min,}	Number of occurrences between the minimum value included and the infinite
{,max}	Number of occurrences between 0 and the maximum value included

## Greed

By default, with quantifiers, the regular expression engine is greedy: it extends the selection as much as possible. This often causes surprises, at first. ? follows a quantifier to make it lazy. Avoid putting two in the same expression, the result can be unpredictable.

Beware of nesting quantifiers, for example, the pattern (a\*)\*, as it exponentially increases processing time.

#### Alternation

The | character in a regular expression is a logical OR. It means that either the preceding or the following expression can match.

#### **Exclusion**

#### Method 1

pattern\_to\_exclude(\*SKIP)(\*FAIL)|pattern\_to\_select

#### Example:

"Blabla"(\*SKIP)(\*FAIL)|Blabla

selects Blabla, in the strings Blabla or "Blabla or Blabla", but not in "Blabla".

#### Method 2

pattern\_to\_exclude\K|(pattern\_to\_select)

"Blabla"\K|(Blabla)

selects Blabla, in the strings Blabla or "Blabla or Blabla", but not in "Blabla".

#### Anchors

An anchor is a way to match a logical location in a string, rather than a character. The most useful anchors for text processing are:

\b

Designates a word boundary, i.e. a transition from space to non-space character. For example, you can use \bsurd to match the surd but not absurd.

^

Matches the start of a line (in multi-line mode, which is the default)

\$

Matches the end of a line (in multi-line mode, which is the default)

\K

Resets the start position of the selection to its position in the pattern. Some regexp engines (but not calibre) do not allow lookbehind of variable length, especially with quantifiers. When you can use  $\K$  with these engines, it also allows you to get rid of this limit by writing the equivalent of a positive lookbehind of variable length.

#### Groups

#### (expression)

Capturing group, which stores the selection and can be recalled later in the *search* or *replace* patterns with n, where n is the sequence number of the capturing group (starting at 1 in reading order)

#### (?:expression)

Group that does not capture the selection

#### (?>expression)

Atomic Group: As soon as the expression is satisfied, the regexp engine passes, and if the rest of the pattern fails, it will not backtrack to try other combinations with the expression. Atomic groups do not capture.

#### (?|expression)

Branch reset group: the branches of the alternations included in the expression share the same group numbers

#### (?<name>expression)

Group named "name". The selection can be recalled later in the *search* pattern by (?P=name) and in the *replace* by \g<name>. Two different groups can use the same name.

#### Lookarounds

Lookaround	Meaning
?=	Positive lookahead (to be placed after the selection)
?!	Negative lookahead (to be placed after the selection)
?<=	Positive lookbehind (to be placed before the selection)
? </th <th>Negative lookbehind (to be placed before the selection)</th>	Negative lookbehind (to be placed before the selection)

Lookaheads and lookbehinds do not consume characters, they are zero length and do not capture. They are atomic groups: as soon as the assertion is satisfied, the regexp engine passes, and if the rest of the pattern fails, it will not backtrack inside the lookaround to try other combinations.

When looking for multiple matches in a string, at the starting position of each match attempt, a lookbehind can inspect the characters before the current position. Therefore, on the string 123, the pattern  $(?<=\d)\d$  (a digit preceded by a digit)

should, in theory, select 2 and 3. On the other hand,  $\d\K\d$  can only select 2, because the starting position after the first selection is immediately before 3, and there are not enough digits for a second match. Similarly,  $\d\(\d)$  only captures 2. In calibre's regexp engine practice, the positive lookbehind behaves in the same way, and selects only 2, contrary to theory.

Groups can be placed inside lookarounds, but capture is rarely useful. Nevertheless, if it is useful, it will be necessary to be very careful in the use of a quantifier in a lookbehind: the greed associated with the absence of backtracking can give a surprising capture. For this reason, use  $\K$  rather than a positive lookbehind when you have a quantifier (or worse, several) in a capturing group of the positive lookbehind.

Example of negative lookahead:

(?![^<>{}]\*[>}])

Placed at the end of the pattern prevents to select within a tag or a style embedded in the file.

Whenever possible, it is always better to "anchor" the lookarounds, to reduce the number of steps necessary to obtain the result.

#### Recursion

Representation	Meaning
(?R)	Recursion of the entire pattern
(?1)	Recursion of the only pattern of the numbered capturing group, here group 1

Recursion is calling oneself. This is useful for balanced queries, such as quoted strings, which can contain embedded quoted strings. Thus, if during the processing of a string between double quotation marks, we encounter the beginning of a new string between double quotation marks, well we know how to do, and we call ourselves. Then we have a pattern like:

```
start-pattern(?>atomic sub-pattern(?R))*end-pattern
```

To select a string between double quotation marks without stopping on an embedded string:

```
"((?>[^""]+|(?R))*[^""]+)"
```

This template can also be used to modify pairs of tags that can be embedded, such as <div> tags.

#### **Special characters**

Representation	Character
\t	tabulation
∖n	line break
\x20	(breakable) space
\xa0	no-break space

#### **Meta-characters**

Meta-characters are those that have a special meaning for the regexp engine. Of these, twelve must be preceded by an escape character, the backslash ( $\)$ , to lose their special meaning and become a regular character again:

^ . [ ] \$ ( ) \* + ? |  $\setminus$ 

Seven other meta-characters do not need to be preceded by a backslash (but can be without any other consequence):

 $\{ \} ! < > = :$ 

Special characters lose their status if they are used inside a class (between brackets []). The closing bracket and the dash have a special status in a class. Outside the class, the dash is a simple literal, the closing bracket remains a meta-character.

The slash (/) and the number sign (or hash character) (#) are not meta-characters, they don't need to be escaped.

In some tools, like regex101.com with the Python engine, double quotes have the special status of separator, and must be escaped, or the options changed. This is not the case in the editor of calibre.

#### Modes

```
(?s)
```

Causes the dot (.) to match newline characters as well

(?m)

Makes the  $\uparrow$  and \$ anchors match the start and end of lines instead of the start and end of the entire string.

# 10.4.14 Credits

Thanks for helping with tips, corrections and such:

- Idolse
- kovidgoyal
- · chaley
- dwanthny
- kacir
- Starson17
- Orpheu

For more about regexps see The Python User Manual<sup>103</sup>. The actual regular expression library used by calibre is: regex<sup>104</sup> which supports several useful enhancements over the Python standard library one.

# 10.5 Writing your own plugins to extend calibre's functionality

calibre has a very modular design. Almost all functionality in calibre comes in the form of plugins. Plugins are used for conversion, for downloading news (though these are called recipes), for various components of the user interface, to connect to different devices, to process files when adding them to calibre and so on. You can get a complete list of all the built-in plugins in calibre by going to *Preferences*  $\rightarrow$  *Advanced*  $\rightarrow$  *Plugins*.

Here, we will teach you how to create your own plugins to add new features to calibre.

## Contents

- Anatomy of a calibre plugin (page 223)
- A User Interface plugin (page 224)

<sup>103</sup> https://docs.python.org/library/re.html

<sup>104</sup> https://bitbucket.org/mrabarnett/mrab-regex/src/hg/

- \_\_*init\_\_.py* (page 225)
- ui.py (page 226)
- main.py (page 228)
- Getting resources from the plugin ZIP file (page 230)
- Enabling user configuration of your plugin (page 231)
- Edit book plugins (page 233)
  - main.py (page 233)
- Adding translations to your plugin (page 236)
- The plugin API (page 237)
- Debugging plugins (page 237)
- More plugin examples (page 237)
- Sharing your plugins with others (page 237)

#### \rm 1 Note

This only applies to calibre releases  $\geq 0.8.60$ 

# 10.5.1 Anatomy of a calibre plugin

A calibre plugin is very simple, it's just a ZIP file that contains some Python code and any other resources like image files needed by the plugin. Without further ado, let's see a basic example.

Suppose you have an installation of calibre that you are using to self publish various e-documents in EPUB and MOBI formats. You would like all files generated by calibre to have their publisher set as "Hello world", here's how to do it. Create a file named \_\_init\_\_.py (this is a special name and must always be used for the main file of your plugin) and enter the following Python code into it:

```
from calibre.customize import FileTypePlugin
class HelloWorld(FileTypePlugin):
                      = 'Hello World Plugin' # Name of the plugin
   name
   description = 'Set the publisher to Hello World for all new conversions'
   supported_platforms = ['windows', 'osx', 'linux'] # Platforms this plugin will_
⇔run on
                      = 'Acme Inc.' # The author of this plugin
   author
                       = (1, 0, 0) # The version number of this plugin
   version
                      = {'epub', 'mobi'} # The file types that this plugin will be_
   file_types
→applied to
   on_postprocess
                     = True # Run this plugin after conversion is complete
   minimum_calibre_version = (0, 7, 53)
   def run(self, path_to_ebook):
```

```
from calibre.ebooks.metadata.meta import get_metadata, set_metadata
with open(path_to_ebook, 'r+b') as file:
    ext = os.path.splitext(path_to_ebook)[-1][1:].lower()
    mi = get_metadata(file, ext)
    mi.publisher = 'Hello World'
    set_metadata(file, mi, ext)
return path_to_ebook
```

That's all. To add this code to calibre as a plugin, simply run the following in the folder in which you created \_\_init\_\_. py:

calibre-customize -b .

#### 1 Note

On macOS, the command line tools are inside the calibre bundle, for example, if you installed calibre in / Applications the command line tools are in /Applications/calibre.app/Contents/MacOS/.

You can download the Hello World plugin from helloworld\_plugin.zip<sup>105</sup>.

Every time you use calibre to convert a book, the plugin's run () method will be called and the converted book will have its publisher set to "Hello World". This is a trivial plugin, lets move on to a more complex example that actually adds a component to the user interface.

## 10.5.2 A User Interface plugin

This plugin will be spread over a few files (to keep the code clean). It will show you how to get resources (images or data files) from the plugin ZIP file, allow users to configure your plugin, how to create elements in the calibre user interface and how to access and query the books database in calibre.

You can download this plugin from interface\_demo\_plugin.zip<sup>106</sup>

The first thing to note is that this ZIP file has a lot more files in it, explained below, pay particular attention to plugin-import-name-interface\_demo.txt.

#### plugin-import-name-interface\_demo.txt

An empty text file used to enable the multi-file plugin magic. This file must be present in all plugins that use more than one .py file. It should be empty and its filename must be of the form: plugin-import-name-\*\*some\_name\*\*.txt. The presence of this file allows you to import code from the .py files present inside the ZIP file, using a statement like:

```
from calibre_plugins.some_name.some_module import some_object
```

The prefix calibre\_plugins must always be present. some\_name comes from the filename of the empty text file. some\_module refers to some\_module.py file inside the ZIP file. Note that this importing is just as powerful as regular Python imports. You can create packages and subpackages of .py modules inside the ZIP file, just like you would normally (by defining \_\_init\_\_.py in each sub-folder), and everything should "just work".

The name you use for some\_name enters a global namespace shared by all plugins, so make it as unique as possible. But remember that it must be a valid Python identifier (only alphabets, numbers and the underscore).

<sup>105</sup> https://calibre-ebook.com/downloads/helloworld\_plugin.zip

<sup>&</sup>lt;sup>106</sup> https://calibre-ebook.com/downloads/interface\_demo\_plugin.zip

#### \_\_init\_\_.py

As before, the file that defines the plugin class

#### main.py

This file contains the actual code that does something useful

#### ui.py

This file defines the interface part of the plugin

#### images/icon.png

The icon for this plugin

#### about.txt

A text file with information about the plugin

#### translations

A folder containing .mo files with the translations of the user interface of your plugin into different languages. See below for details.

Now let's look at the code.

#### \_\_init\_\_.py

First, the obligatory \_\_init\_\_.py to define the plugin metadata:

```
from calibre.customize import InterfaceActionBase
class InterfacePluginDemo(InterfaceActionBase):
    111
   This class is a simple wrapper that provides information about the actual
   plugin class. The actual interface plugin class is called InterfacePlugin
   and is defined in the ui.py file, as specified in the actual_plugin field
   below.
   The reason for having two classes is that it allows the command line
   calibre utilities to run without needing to load the GUI libraries.
    1.1.1
   name
                       = 'Interface Plugin Demo'
   description = 'An advanced plugin demo'
   supported_platforms = ['windows', 'osx', 'linux']
   author = 'Kovid Goyal'
   version
                      = (1, 0, 0)
   minimum_calibre_version = (0, 7, 53)
   #: This field defines the GUI plugin class that contains all the code
    #: that actually does something. Its format is module_path:class_name
    #: The specified class must be defined in the specified module.
   actual_plugin = 'calibre_plugins.interface_demo.ui:InterfacePlugin'
   def is_customizable(self):
        . . .
       This method must return True to enable customization via
       Preferences->Plugins
        1.1.1
       return True
```

```
def config_widget(self):
    . . .
    Implement this method and :meth: `save_settings` in your plugin to
    use a custom configuration dialog.
    This method, if implemented, must return a QWidget. The widget can have
    an optional method validate() that takes no arguments and is called
    immediately after the user clicks OK. Changes are applied if and only
    if the method returns True.
    If for some reason you cannot perform the configuration at this time,
    return a tuple of two strings (message, details), these will be
    displayed as a warning dialog to the user and the process will be
    aborted.
    The base class implementation of this method raises NotImplementedError
    so by default no user configuration is possible.
    ...
    # It is important to put this import statement here rather than at the
    # top of the module as importing the config class will also cause the
    # GUI libraries to be loaded, which we do not want when using calibre
    # from the command line
    from calibre_plugins.interface_demo.config import ConfigWidget
    return ConfigWidget()
def save_settings(self, config_widget):
    . . .
    Save the settings specified by the user with config_widget.
    :param config_widget: The widget returned by :meth:`config_widget`.
    . . .
    config_widget.save_settings()
    # Apply the changes
    ac = self.actual_plugin_
    if ac is not None:
        ac.apply_settings()
```

The only noteworthy feature is the field actual\_plugin. Since calibre has both command line and GUI interfaces, GUI plugins like this one should not load any GUI libraries in \_\_init\_\_.py. The actual\_plugin field does this for you, by telling calibre that the actual plugin is to be found in another file inside your ZIP archive, which will only be loaded in a GUI context.

Remember that for this to work, you must have a plugin-import-name-some\_name.txt file in your plugin ZIP file, as discussed above.

Also there are a couple of methods for enabling user configuration of the plugin. These are discussed below.

#### ui.py

Now let's look at ui.py which defines the actual GUI plugin. The source code is heavily commented and should be self explanatory:

```
from calibre.qui2.actions import InterfaceAction
from calibre_plugins.interface_demo.main import DemoDialog
class InterfacePlugin(InterfaceAction):
   name = 'Interface Plugin Demo'
    # Declare the main action associated with this plugin
    # The keyboard shortcut can be None if you don't want to use a keyboard
    # shortcut. Remember that currently calibre has no central management for
    # keyboard shortcuts, so try to use an unusual/unused shortcut.
   action_spec = ('Interface Plugin Demo', None,
            'Run the Interface Plugin Demo', 'Ctrl+Shift+F1')
   def genesis(self):
        # This method is called once per plugin, do initial setup here
        # Set the icon for this interface action
        # The get_icons function is a builtin function defined for all your
        # plugin code. It loads icons from the plugin zip file. It returns
        # QIcon objects, if you want the actual data, use the analogous
        # get_resources builtin function.
        # Note that if you are loading more than one icon, for performance, you
        # should pass a list of names to get_icons. In this case, get_icons
        # will return a dictionary mapping names to QIcons. Names that
        # are not found in the zip file will result in null QIcons.
       icon = get_icons('images/icon.png', 'Interface Demo Plugin')
        # The qaction is automatically created from the action_spec defined
        # above
       self.qaction.setIcon(icon)
        self.qaction.triggered.connect(self.show_dialog)
    def show_dialog(self):
        # The base plugin object defined in __init__.py
       base_plugin_object = self.interface_action_base_plugin
        # Show the config dialog
        # The config dialog can also be shown from within
        # Preferences->Plugins, which is why the do_user_config
        # method is defined on the base plugin class
        do_user_config = base_plugin_object.do_user_config
        # self.gui is the main calibre GUI. It acts as the gateway to access
        # all the elements of the calibre user interface, it should also be the
        # parent of the dialog
        d = DemoDialog(self.gui, self.qaction.icon(), do_user_config)
        d.show()
    def apply_settings(self):
       from calibre plugins.interface demo.config import prefs
        # In an actual non trivial plugin, you would probably need to
                                                                          (continues on next page)
```

```
\ensuremath{\#} do something based on the settings in prefs prefs
```

#### main.py

The actual logic to implement the Interface Plugin Demo dialog.

```
from calibre_plugins.interface_demo.config import prefs
class DemoDialog(QDialog):
    def __init__(self, gui, icon, do_user_config):
        QDialog.__init__(self, gui)
        self.gui = gui
        self.do_user_config = do_user_config
        # The current database shown in the GUI
        # db is an instance of the class LibraryDatabase from db/legacy.py
        # This class has many, many methods that allow you to do a lot of
        # things. For most purposes you should use db.new_api, which has
        # a much nicer interface from db/cache.py
        self.db = gui.current_db
        self.l = QVBoxLayout()
        self.setLayout(self.l)
        self.label = QLabel(prefs['hello_world_msg'])
        self.l.addWidget(self.label)
        self.setWindowTitle('Interface Plugin Demo')
        self.setWindowIcon(icon)
        self.about_button = QPushButton('About', self)
        self.about_button.clicked.connect(self.about)
        self.l.addWidget(self.about_button)
        self.marked_button = QPushButton(
            'Show books with only one format in the calibre GUI', self)
        self.marked_button.clicked.connect(self.marked)
        self.l.addWidget(self.marked_button)
        self.view_button = QPushButton(
            'View the most recently added book', self)
        self.view_button.clicked.connect(self.view)
        self.l.addWidget(self.view_button)
        self.update_metadata_button = QPushButton(
            "Update metadata in a book's files", self)
        self.update_metadata_button.clicked.connect(self.update_metadata)
        self.l.addWidget(self.update_metadata_button)
                                                                          (continues on next page)
```

```
(continued from previous page)
```

```
self.conf_button = QPushButton(
                'Configure this plugin', self)
       self.conf_button.clicked.connect(self.config)
       self.l.addWidget(self.conf_button)
       self.resize(self.sizeHint())
   def about(self):
       # Get the about text from a file inside the plugin zip file
        # The get_resources function is a builtin function defined for all your
        # plugin code. It loads files from the plugin zip file. It returns
        # the bytes from the specified file.
        # Note that if you are loading more than one file, for performance, you
       # should pass a list of names to get_resources. In this case,
       # get_resources will return a dictionary mapping names to bytes. Names that
       # are not found in the zip file will not be in the returned dictionary.
       text = get_resources('about.txt')
       QMessageBox.about(self, 'About the Interface Plugin Demo',
               text.decode('utf-8'))
   def marked(self):
       ''' Show books with only one format '''
       db = self.db.new_api
       matched_ids = {book_id for book_id in db.all_book_ids() if len(db.
\rightarrow formats(book_id)) == 1}
       # Mark the records with the matching ids
        # new_api does not know anything about marked books, so we use the full
       # db object
       self.db.set_marked_ids(matched_ids)
       # Tell the GUI to search for all marked records
       self.gui.search.setEditText('marked:true')
       self.gui.search.do_search()
   def view(self):
        ''' View the most recently added book '''
       most_recent = most_recent_id = None
       db = self.db.new_api
       for book_id, timestamp in db.all_field_for('timestamp', db.all_book_ids()).
\rightarrow items():
           if most_recent is None or timestamp > most_recent:
               most_recent = timestamp
               most_recent_id = book_id
       if most_recent_id is not None:
            # Get a reference to the View plugin
           view_plugin = self.gui.iactions['View']
            # Ask the view plugin to launch the viewer for row_number
           view_plugin._view_calibre_books([most_recent_id])
```

```
def update_metadata(self):
    . . .
    Set the metadata in the files in the selected book's record to
    match the current metadata in the database.
    1 1 1
    from calibre.ebooks.metadata.meta import set_metadata
    from calibre.gui2 import error_dialog, info_dialog
    # Get currently selected books
    rows = self.qui.library_view.selectionModel().selectedRows()
    if not rows or len(rows) == 0:
        return error_dialog(self.gui, 'Cannot update metadata',
                         'No books selected', show=True)
    # Map the rows to book ids
    ids = list(map(self.gui.library_view.model().id, rows))
    db = self.db.new api
    for book_id in ids:
        # Get the current metadata for this book from the db
        mi = db.get_metadata(book_id, get_cover=True, cover_as_data=True)
        fmts = db.formats(book_id)
        if not fmts:
            continue
        for fmt in fmts:
            fmt = fmt.lower()
            # Get a python file object for the format. This will be either
            # an in memory file or a temporary on disk file
            ffile = db.format(book_id, fmt, as_file=True)
            ffile.seek(0)
            # Set metadata in the format
            set_metadata(ffile, mi, fmt)
            ffile.seek(0)
            # Now replace the file in the calibre library with the updated
            # file. We don't use add format with hooks as the hooks were
            # already run when the file was first added to calibre.
            db.add_format(book_id, fmt, ffile, run_hooks=False)
    info_dialog(self, 'Updated files',
            f'Updated the metadata in the files of {len(ids)} book(s)',
            show=True)
def config(self):
    self.do_user_config(parent=self)
    # Apply the changes
    self.label.setText(prefs['hello_world_msg'])
```

#### Getting resources from the plugin ZIP file

calibre's plugin loading system defines a couple of built-in functions that allow you to conveniently get files from the plugin ZIP file.

#### get\_resources(name\_or\_list\_of\_names)

This function should be called with a list of paths to files inside the ZIP file. For example to access the file icon.png in the folder images in the ZIP file, you would use: images/icon.png. Always use a

forward slash as the path separator, even on Windows. When you pass in a single name, the function will return the raw bytes of that file or None if the name was not found in the ZIP file. If you pass in more than one name then it returns a dictionary mapping the names to bytes. If a name is not found, it will not be present in the returned dictionary.

#### get\_icons(name\_or\_list\_of\_names, plugin\_name=")

A wrapper for get\_resources() that creates QIcon objects from the raw bytes returned by get\_resources. If a name is not found in the ZIP file the corresponding QIcon will be null. In order to support icon theme-ing, pass in the human friendly name of your plugin as plugin\_name. If the user is using an icon theme with icons for your plugin, they will be loaded preferentially.

#### Enabling user configuration of your plugin

To allow users to configure your plugin, you must define three methods in your base plugin class, **is\_customizable**, **config\_widget** and **save\_settings** as shown below:

```
def is_customizable(self):
    '''
    This method must return True to enable customization via
    Preferences->Plugins
    '''
    return True
```

def config\_widget(self):

. . .

```
Implement this method and :meth:`save_settings` in your plugin to
use a custom configuration dialog.
This method, if implemented, must return a QWidget. The widget can have
an optional method validate() that takes no arguments and is called
```

an optional method validate() that takes no arguments and is called immediately after the user clicks OK. Changes are applied if and only if the method returns True.

If for some reason you cannot perform the configuration at this time, return a tuple of two strings (message, details), these will be displayed as a warning dialog to the user and the process will be aborted.

The base class implementation of this method raises NotImplementedError so by default no user configuration is possible. ''' # It is important to put this import statement here rather than at the # top of the module as importing the config class will also cause the

# GUI libraries to be loaded, which we do not want when using calibre # from the command line from calibre plugins.interface demo.config import ConfigWidget

```
return ConfigWidget()
```

```
def save_settings(self, config_widget):
    '''
    Save the settings specified by the user with config_widget.
    :param config_widget: The widget returned by :meth:`config_widget`.
    '''
```

calibre has many different ways to store configuration data (a legacy of its long history). The recommended way is to use the **JSONConfig** class, which stores your configuration information in a .json file.

The code to manage configuration data in the demo plugin is in config.py:

```
from calibre.utils.config import JSONConfig
# This is where all preferences for this plugin will be stored
# Remember that this name (i.e. plugins/interface_demo) is also
# in a global namespace, so make it as unique as possible.
# You should always prefix your config file name with plugins/,
# so as to ensure you don't accidentally clobber a calibre config file
prefs = JSONConfig('plugins/interface_demo')
# Set defaults
prefs.defaults['hello_world_msg'] = 'Hello, World!'
class ConfigWidget(QWidget):
    def __init__(self):
       QWidget.___init___(self)
        self.l = QHBoxLayout()
        self.setLayout(self.l)
        self.label = QLabel('Hello world &message:')
        self.l.addWidget(self.label)
       self.msg = QLineEdit(self)
        self.msg.setText(prefs['hello_world_msg'])
        self.l.addWidget(self.msg)
        self.label.setBuddy(self.msg)
    def save_settings(self):
        prefs['hello_world_msg'] = self.msg.text()
```

The prefs object is now available throughout the plugin code by a simple:

from calibre\_plugins.interface\_demo.config import prefs

You can see the prefs object being used in main.py:

```
def config(self):
    self.do_user_config(parent=self)
    # Apply the changes
```

```
self.label.setText(prefs['hello_world_msg'])
```

# 10.5.3 Edit book plugins

Now let's change gears for a bit and look at creating a plugin to add tools to the calibre book editor. The plugin is available here: editor\_demo\_plugin.zip<sup>107</sup>.

The first step, as for all plugins is to create the import name empty txt file, as described *above* (page 224). We shall name the file plugin-import-name-editor\_plugin\_demo.txt.

Now we create the mandatory \_\_init\_\_.py file that contains metadata about the plugin – its name, author, version, etc.

```
class DemoPlugin(EditBookToolPlugin):
    name = 'Edit Book plugin demo'
    version = (1, 0, 0)
    author = 'Kovid Goyal'
    supported_platforms = ['windows', 'osx', 'linux']
    description = 'A demonstration of the plugin interface for the ebook editor'
    minimum_calibre_version = (1, 46, 0)
```

A single editor plugin can provide multiple tools each tool corresponds to a single button in the toolbar and entry in the *Plugins* menu in the editor. These can have sub-menus in case the tool has multiple related actions.

The tools must all be defined in the file main.py in your plugin. Every tool is a class that inherits from the *calibre*. gui2.tweak\_book.plugin.Tool (page 382) class. Let's look at main.py from the demo plugin, the source code is heavily commented and should be self-explanatory. Read the API documents of the *calibre.gui2.tweak\_book*. plugin.Tool (page 382) class for more details.

#### main.py

Here we will see the definition of a single tool that will multiply all font sizes in the book by a number provided by the user. This tool demonstrates various important concepts that you will need in developing your own plugins, so you should read the (heavily commented) source code carefully.

```
from css_parser.css import CSSRule
from qt.core import QAction, QInputDialog
from calibre import force_unicode
from calibre.ebooks.oeb.polish.container import OEB_DOCS, OEB_STYLES, serialize
from calibre.gui2 import error_dialog
# The base class that all tools must inherit from
from calibre.gui2.tweak_book.plugin import Tool
class DemoTool(Tool):
    #: Set this to a unique name it will be used as a key
```

<sup>&</sup>lt;sup>107</sup> https://calibre-ebook.com/downloads/editor\_demo\_plugin.zip

```
name = 'demo-tool'
   #: If True the user can choose to place this tool in the plugins toolbar
   allowed_in_toolbar = True
   #: If True the user can choose to place this tool in the plugins menu
   allowed_in_menu = True
   def create_action(self, for_toolbar=True):
       # Create an action, this will be added to the plugins toolbar and
       # the plugins menu
       ac = QAction(get_icons('images/icon.png'), 'Magnify fonts', self.gui) #_
→noqa: F821
       if not for_toolbar:
           # Register a keyboard shortcut for this toolbar action. We only
           # register it for the action created for the menu, not the toolbar,
           # to avoid a double trigger
           self.register_shortcut(ac, 'magnify-fonts-tool', default_keys=(
ac.triggered.connect(self.ask_user)
       return ac
   def ask_user(self):
       # Ask the user for a factor by which to multiply all font sizes
       factor, ok = QInputDialog.getDouble(
           self.gui, 'Enter a magnification factor', 'Allow font sizes in the book_
↔ will be multiplied by the specified factor',
           value=2, min=0.1, max=4
       )
       if ok:
           # Ensure any in progress editing the user is doing is present in the.
⇔container
           self.boss.commit_all_editors_to_container()
           try:
               self.magnify_fonts(factor)
           except Exception:
               # Something bad happened report the error to the user
               import traceback
               error_dialog(self.gui, _('Failed to magnify fonts'), _(
                   'Failed to magnify fonts, click "Show details" for more info'),
                   det_msg=traceback.format_exc(), show=True)
               # Revert to the saved restore point
               self.boss.revert_requested(self.boss.global_undo.previous_container)
           else:
               # Show the user what changes we have made, allowing her to
               # revert them if necessary
               self.boss.show_current_diff()
               # Update the editor UI to take into account all the changes we
               # have made
               self.boss.apply_container_update_to_gui()
   def magnify_fonts(self, factor):
```

```
# Magnify all font sizes defined in the book by the specified factor
        # First we create a restore point so that the user can undo all changes
        # we make.
       self.boss.add_savepoint('Before: Magnify fonts')
       container = self.current_container # The book being edited as a container.
→object
        # Iterate over all style declarations in the book, this means css
        # stylesheets, <style> tags and style="" attributes
       for name, media_type in container.mime_map.items():
            if media_type in OEB_STYLES:
                # A stylesheet. Parsed stylesheets are css_parser CSSStylesheet
                # objects.
                self.magnify_stylesheet(container.parsed(name), factor)
                container.dirty(name) # Tell the container that we have changed the.
→ stylesheet
            elif media_type in OEB_DOCS:
                # A HTML file. Parsed HTML files are lxml elements
                for style_tag in container.parsed(name).xpath('//*[local-name="style"]
\rightarrow '):
                    if style_tag.text and style_tag.get('type', None) in {None, 'text/
\leftrightarrow css'}:
                        # We have an inline CSS <style> tag, parse it into a
                        # stylesheet object
                        sheet = container.parse_css(style_tag.text)
                        self.magnify_stylesheet(sheet, factor)
                        style_tag.text = serialize(sheet, 'text/css', pretty_
→print=True)
                        container.dirty(name) # Tell the container that we have_
\hookrightarrow changed the stylesheet
                for elem in container.parsed(name).xpath('//*[@style]'):
                    # Process inline style attributes
                    block = container.parse_css(elem.get('style'), is_
→declaration=True)
                    self.magnify_declaration(block, factor)
                    elem.set('style', force_unicode(block.getCssText(separator=' '),
\leftrightarrow 'utf-8'))
   def magnify_stylesheet(self, sheet, factor):
       # Magnify all fonts in the specified stylesheet by the specified
        # factor.
       for rule in sheet.cssRules.rulesOfType(CSSRule.STYLE_RULE):
            self.magnify_declaration(rule.style, factor)
   def magnify_declaration(self, style, factor):
       # Magnify all fonts in the specified style declaration by the specified
        # factor
       val = style.getPropertyValue('font-size')
       if not val:
            return
```

```
# see if the font-size contains a number
num = re.search(r'[0-9.]+', val)
if num is not None:
    num = num.group()
    val = val.replace(num, f'{float(num)*factor:f}')
    style.setProperty('font-size', val)
# We should also be dealing with the font shorthand property and
# font sizes specified as non numbers, but those are left as exercises
# for the reader
```

Let's break down main.py. We see that it defines a single tool, named *Magnify fonts*. This tool will ask the user for a number and multiply all font sizes in the book by that number.

The first important thing is the tool name which you must set to some relatively unique string as it will be used as the key for this tool.

The next important entry point is the *calibre.gui2.tweak\_book.plugin.Tool.create\_action()* (page 383). This method creates the QAction objects that appear in the plugins toolbar and plugin menu. It also, optionally, assigns a keyboard shortcut that the user can customize. The triggered signal from the QAction is connected to the ask\_user() method that asks the user for the font size multiplier, and then runs the magnification code.

The magnification code is well commented and fairly simple. The main things to note are that you get a reference to the editor window as self.gui and the editor *Boss* as self.boss. The *Boss* is the object that controls the editor user interface. It has many useful methods, that are documented in the *calibre.gui2.tweak\_book.boss.Boss* (page 384) class.

Finally, there is self.current\_container which is a reference to the book being edited as a *calibre.ebooks*. *oeb.polish.container.Container* (page 375) object. This represents the book as a collection of its constituent HTML/CSS/image files and has convenience methods for doing many useful things. The container object and various useful utility functions that can be reused in your plugin code are documented in *API documentation for the e-book editing tools* (page 375).

# 10.5.4 Adding translations to your plugin

You can have all the user interface strings in your plugin translated and displayed in whatever language is set for the main calibre user interface.

The first step is to go through your plugin's source code and mark all user visible strings as translatable, by surrounding them in \_(). For example:

action\_spec = (\_('My plugin'), None, \_('My plugin is cool'), None)

Then use some program to generate .po files from your plugin source code. There should be one .po file for every language you want to translate into. For example: de.po for German, fr.po for French and so on. You can use the Poedit<sup>108</sup> program for this.

Send these .po files to your translators. Once you get them back, compile them into .mo files. You can again use Poedit for that, or just do:

```
calibre-debug -c "from calibre.translations.msgfmt import main; main()" filename.po
```

Put the .mo files into the translations folder in your plugin.

The last step is to simply call the function *load\_translations()* at the top of your plugin's .py files. For performance reasons you should only call this function in those .py files that actually have translatable strings. So in a typical User Interface

<sup>&</sup>lt;sup>108</sup> https://poedit.net/

plugin you would call it at the top of ui.py but not \_\_init\_\_.py.

You can test the translations of your plugins by changing the user interface language in calibre under *Preferences*  $\rightarrow$  *Interface*  $\rightarrow$  *Look* & *feel* or by running calibre with the CALIBRE\_OVERRIDE\_LANG environment variable set. For example:

```
CALIBRE_OVERRIDE_LANG=de
```

Replace de with the language code of the language you want to test.

For translations with plurals, use the ngettext () function instead of \_(). For example:

ngettext('Delete a book', 'Delete {} books', num\_books).format(num\_books)

## 10.5.5 The plugin API

As you may have noticed above, a plugin in calibre is a class. There are different classes for the different types of plugins in calibre. Details on each class, including the base class of all plugins can be found in *API documentation for plugins* (page 253).

Your plugin is almost certainly going to use code from calibre. To learn how to find various bits of functionality in the calibre code base, read the section on the calibre *Code layout* (page 358).

## 10.5.6 Debugging plugins

The first, most important step is to run calibre in debug mode. You can do this from the command line with:

calibre-debug -g

Or from within calibre by right-clicking the *Preferences* button or using the Ctrl+Shift+R keyboard shortcut.

When running from the command line, debug output will be printed to the console, when running from within calibre the output will go to a txt file.

You can insert print statements anywhere in your plugin code, they will be output in debug mode. Remember, this is Python, you really shouldn't need anything more than print statements to debug ;) I developed all of calibre using just this debugging technique.

You can quickly test changes to your plugin by using the following command line:

calibre-debug -s; calibre-customize -b /path/to/your/plugin/folder; calibre

This will shutdown a running calibre, wait for the shutdown to complete, then update your plugin in calibre and relaunch calibre.

### 10.5.7 More plugin examples

You can find a list of many sophisticated calibre plugins here<sup>109</sup>.

## 10.5.8 Sharing your plugins with others

If you would like to share the plugins you have created with other users of calibre, post your plugin in a new thread in the calibre plugins forum<sup>110</sup>.

<sup>109</sup> https://www.mobileread.com/forums/showthread.php?t=118764

<sup>&</sup>lt;sup>110</sup> https://www.mobileread.com/forums/forumdisplay.php?f=237

# 10.6 Typesetting mathematics in e-books

The calibre E-book viewer has the ability to display mathematics embedded in e-books (EPUB and HTML files). You can typeset the mathematics directly with TeX or MathML or AsciiMath. The calibre E-book viewer uses the excellent MathJax<sup>111</sup> library to do this. This is a brief tutorial on creating e-books with mathematics in them that work well with the calibre E-book viewer.

# 10.6.1 A simple HTML file with mathematics

You can write mathematics inline inside a simple HTML file and the calibre E-book viewer will render it into properly typeset mathematics. In the example below, we use TeX notation for mathematics. You will see that you can use normal TeX commands, with the small caveat that ampersands and less than and greater than signs have to be written as & amp; & lt; and > respectively.

The first step is to tell calibre that this will contains mathematics. You do this by adding the following snippet of code to the <head> section of the HTML file:

```
<script type="text/x-mathjax-config"></script>
```

That's it, now you can type mathematics just as you would in a .tex file. For example, here are Lorentz's equations:

```
<h2>The Lorenz Equations</h2>
\begin{align}
\dot{x} & amp; = \sigma(y-x) \\
\dot{y} & amp; = \rho x - y - xz \\
\dot{z} & amp; = -\beta z + xy
\end{align}
```

This snippet looks like the following screen shot in the calibre E-book viewer.

```
egin{aligned} \dot{x} &= \sigma(y-x) \ \dot{y} &= 
ho x - y - xz \ \dot{z} &= -eta z + xy \end{aligned}
```

Fig. 1	1:	The	Lorenz	Equations
--------	----	-----	--------	-----------

The complete HTML file, with more equations and inline mathematics is reproduced below. You can convert this HTML file to EPUB in calibre to end up with an e-book you can distribute easily to other people.

```
<!DOCTYPE html>
<html>
<!-- Copyright (c) 2012 Design Science, Inc. -->
<head>
<title>Math Test Page</title>
```

(continues on next page)

<sup>111</sup> https://www.mathjax.org

```
(continued from previous page)
<meta http-equiv="content-type" content="text/html; charset=UTF-8" />
<!-- This script tag is needed to make calibre's ebook-viewer recpgnize that this_
→file needs math typesetting -->
<script type="text/x-mathjax-config">
    // This line adds numbers to all equations automatically, unless explicitly_
\hookrightarrow suppressed.
    MathJax.tex = {tags: 'all'};
</script>
<style>
h1 {text-align:center}
h2 {
  font-weight: bold;
 background-color: #DDDDDD;
  padding: .2em .5em;
  margin-top: 1.5em;
 border-top: 3px solid #666666;
 border-bottom: 2px solid #999999;
}
</style>
</head>
<body>
<h1>Sample Equations</h1>
<h2>The Lorenz Equations</h2>
<p>
\begin{align}
\dot{x} & = \sigma(y-x) \label{lorenz}\\
\det\{y\} \& amp; = \ x - y - xz \
\det{z} \quad \text{amp;} = -\det z + xy
\end{align}
</p>
<h2>The Cauchy-Schwarz Inequality</h2>
<q>\[
\left( \sum_{k=1}^n a_k b_k \right)^{\left(\left( \sum_{k=1}^n a_k b_k \right)^{\left( \sum_{k=1}^n a_k b_k \right)} \right)
\left( \sum_{k=1}^n a_k^2 \right) \left( \sum_{k=1}^n b_k^2 \right)
\]
<h2>A Cross Product Formula</h2>
<p>\[
  \mathbb{V}_1 \times \mathbb{V}_2 =
   \begin{vmatrix}
    \mathbf{i} & \mathbf{j} & \mathbf{k} \\
    \frac{\partial X}{\partial u} & \frac{\partial Y}{\partial u} & amp; 0 \\
    \frac{\partial X}{\partial v} & amp; \frac{\partial Y}{\partial v} & amp; 0 \\
   \end{vmatrix}
```

```
\]</p>
<h2>The probability of getting (k) heads when flipping (n) coins is:</h2>
\[P(E) = {n \choose k} p^k (1-p)^{ n-k} \]
<h2>An Identity of Ramanujan</h2>
<q>\[
        \frac{1}{(\sqrt{\phi \sqrt{5}}-\phi) e^{\frac25 \pi}} =
               1+\frac{e^{-2\psi}}{1+\frac{e^{-4\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac{e^{-6\psi}}{1+\frac
                   {1+\frac{e^{-8\pi}} {1+\ldots} } }
\]</p>
<h2>A Rogers-Ramanujan Identity</h2>
<p>\[
     1 + \frac{q^2}{(1-q)} + \frac{q^6}{(1-q)} + \frac{q^6}{(1-q)} = 0
            \prod_{j=0}^{\infty}\frac{1}{(1-q^{5j+2})(1-q^{5j+3})},
               \quad\quad \text{for $|q|<1$}.
\]
<h2>Maxwell's Equations</h2>
<p>
\begin{align}
     \nabla \times \vec{\mathbf{B}} -\, \frac1c\, \frac{\partial\vec{\mathbf{E}}}{\
\nabla \cdot \vec{\mathbf{E}} & amp; = 4 \pi \rho \\
     \nabla \times \vec{\mathbf{E}}\, +\, \frac1c\, \frac{\partial\vec{\mathbf{B}}}{\
→partial t} & = \vec{\mathbf{0}} \\
      \nabla \cdot \vec{\mathbf{B}} & amp; = 0
\end{align}
<h2>Inline Mathematics</h2>
While display equations look good for a page of samples, the
ability to mix math and text in a paragraph is also important. This
expression (\left| \frac{3x-1}{4x}\right)^2) is an example of an inline equation. As
you see, equations can be used this way as well, without unduly
disturbing the spacing between lines.
<h2>References to equations</h2>
Here is a reference to the Lorenz Equations (\ref{lorenz}). Clicking on the_
 \rightarrowequation number will take you back to the equation.
</body>
</html>
```

## 1 Note

The calibre E-book viewer supports MathML as well as TeX, but you must include the <script type="text/x-mathjax-config"></script> line in your HTML file otherwise the MathML will not render.

# 10.6.2 More information

Since the calibre E-book viewer uses the MathJax library to render mathematics, the best place to find out more about mathematics in e-books and get help is the MathJax website<sup>112</sup>.

# 10.7 Creating AZW3 • EPUB • MOBI catalogs

calibre's Create catalog feature enables you to create a catalog of your library in a variety of formats. This help file describes cataloging options when generating a catalog in AZW3, EPUB and MOBI formats.

- Selecting books to catalog (page 241)
- *Included sections* (page 242)
- Prefixes (page 243)
- Excluded books (page 243)
- Excluded genres (page 244)
- Other options (page 244)
- *Custom catalog covers* (page 245)
- Additional help resources (page 245)

## 10.7.1 Selecting books to catalog

If you want *all* of your library cataloged, remove any search or filtering criteria in the main window. With a single book selected, all books in your library will be candidates for inclusion in the generated catalog. Individual books may be excluded by various criteria; see the *Excluded genres* (page 244) section below for more information.

If you want only some of your library cataloged, you have two options:

- Create a multiple selection of the books you want cataloged. With more than one book selected in calibre's main window, only the selected books will be cataloged.
- Use the Search field or the Tag browser to filter the displayed books. Only the displayed books will be cataloged.

To begin catalog generation, select the menu item *Convert books > Create a catalog of the books in your calibre library*. You may also add a *Create catalog* button to a toolbar in *Preferences > Interface > Toolbars & menus* for easier access to the Generate catalog dialog.

<sup>&</sup>lt;sup>112</sup> https://www.mathjax.org

Generate catalog	for 19 books		
Catalog options	E-book options		
Catalog <u>f</u> ormat:		EPUB	•
Catalog <u>t</u> itle (exis same title will be	sting catalog with the	My Books	

In *Catalog options*, select **AZW3**, **EPUB or MOBI** as the Catalog format. In the *Catalog title* field, provide a name that will be used for the generated catalog. If a catalog of the same name and format already exists, it will be replaced with the newly-generated catalog.

(	Send catalog to device automatically
	🔞 Help 🛛 🖌 Apply

Enabling Send catalog to device automatically will download the generated catalog to a connected device upon completion.

# 10.7.2 Included sections

✓ <u>T</u> itles	Series
✓ <u>R</u> ecently Added	✓ <u>D</u> escriptions
	<ul> <li>✓ <u>T</u>itles</li> <li>✓ <u>R</u>ecently Added</li> </ul>

Sections enabled by a checkmark will be included in the generated catalog:

- Authors all books, sorted by author, presented in a list format. Non-series books are listed before series books.
- Titles all books, sorted by title, presented in a list format.
- Series all books that are part of a series, sorted by series, presented in a list format.
- Genres individual genres presented in a list, sorted by Author and Series.
- *Recently Added* all books, sorted in reverse chronological order. List includes books added in the last 30 days, then a month-by-month listing of added books.
- *Descriptions* detailed description page for each book, including a cover thumbnail and comments. Sorted by author, with non-series books listed before series books.

# 10.7.3 Prefixes

_	Name	Pre	fix	Field	_	Value	
V	Read book	<ul> <li>Image: A second s</li></ul>	*	Last Read	۲	any date	•
V	Wishlist item	×	۲	Tags	¥	Wishlist	*
•	Library books	↔	٠	Available in Library	*	True	*

Prefix rules allow you to add a prefix to book listings when certain criteria are met. For example, you might want to mark books you've read with a checkmark, or books on your wishlist with an X.

The checkbox in the first column enables the rule. *Name* is a rule name that you provide. *Field* is either *Tags* or a custom column from your library. *Value* is the content of *Field* to match. When a prefix rule is satisfied, the book will be marked with the selected *Prefix*.

Three prefix rules have been specified in the example above:

- 1. *Read book* specifies that a book with any date in a custom column named *Last read* will be prefixed with a checkmark symbol.
- 2. Wishlist item specifies that any book with a Wishlist tag will be prefixed with an X symbol.
- 3. *Library* books specifies that any book with a value of True (or Yes) in a custom column *Available in Library* will be prefixed with a double arrow symbol.

The first matching rule supplies the prefix. Disabled or incomplete rules are ignored.

# 10.7.4 Excluded books

	Name	Field		Value	
•	Catalogs	Tags	•	Catalog	•
•	Archived Books	Status	•	Archived	•

Exclusion rules allow you to specify books that will not be cataloged.

The checkbox in the first column enables the rule. *Name* is a rule name that you provide. *Field* is either *Tags* or a custom column in your library. *Value* is the content of *Field* to match. When an exclusion rule is satisfied, the book will be excluded from the generated catalog.

Two exclusion rules have been specified in the example above:

- 1. The Catalogs rule specifies that any book with a Catalog tag will be excluded from the generated catalog.
- 2. The *Archived* Books rule specifies that any book with a value of *Archived* in the custom column *Status* will be excluded from the generated catalog.

All rules are evaluated for every book. Disabled or incomplete rules are ignored.

# 10.7.5 Excluded genres

xcluded genres		
Tags to <u>e</u> xclude (regex):	\[.+\] \+	0
Results of regex:	+, [Amazon Freebie], [Project Gutenberg], [Test]	

When the catalog is generated, tags in your database are used as genres. For example, you may use the tags Fiction and Nonfiction. These tags become genres in the generated catalog, with books listed under their respective genre lists based on their assigned tags. A book will be listed in every genre section for which it has a corresponding tag.

You may be using certain tags for other purposes, perhaps a + to indicate a read book, or a bracketed tag like [Amazon Freebie] to indicate a book's source. The *Excluded genres* regex allows you to specify tags that you don't want used as genres in the generated catalog. The default exclusion regex pattern \[.+\]\+ excludes any tags of the form [tag], as well as excluding +, the default tag for read books, from being used as genres in the generated catalog.

You can also use an exact tag name in a regex. For example, [Amazon Freebie] or [Project Gutenberg]. If you want to list multiple exact tags for exclusion, put a pipe (vertical bar) character between them: [Amazon Freebie] | [Project Gutenberg].

*Results of regex* shows you which tags will be excluded when the catalog is built, based on the tags in your database and the regex pattern you enter. The results are updated as you modify the regex pattern.

# 10.7.6 Other options

O Generate new o	over 💿 Use exi	sting cover	
Last Read	▼ <u>T</u> hu	mb width: 1.00	inch 🗘
Author Bio	• <u>B</u> efore	• After	✓ Include <u>S</u> eparator
	Generate new of Last Read	Last Read	Last Read   Thumb width: 1.00

*Catalog cover* specifies whether to generate a new cover or use an existing cover. It is possible to create a custom cover for your catalogs - see *Custom catalog covers* (page 245) for more information. If you have created a custom cover that you want to reuse, select *Use existing cover*. Otherwise, select *Generate new cover*.

*Extra Description note* specifies a custom column's contents to be inserted into the Description page, next to the cover thumbnail. For example, you might want to display the date you last read a book using a *Last Read* custom column. For advanced use of the Description note feature, see this post in the calibre forum<sup>113</sup>.

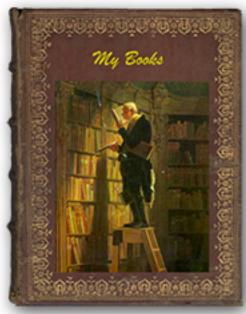
*Thumb width* specifies a width preference for cover thumbnails included with Descriptions pages. Thumbnails are cached to improve performance. To experiment with different widths, try generating a catalog with just a few books until you've determined your preferred width, then generate your full catalog. The first time a catalog is generated with a new thumbnail width, performance will be slower, but subsequent builds of the catalog will take advantage of the thumbnail cache.

*Merge with comments* specifies a custom column whose content will be non-destructively merged with the comments metadata during catalog generation. For example, you might have a custom column *Author bio* that you'd like to append to the comments metadata. You can choose to insert the custom column contents *before or after* the comments section,

<sup>&</sup>lt;sup>113</sup> https://www.mobileread.com/forums/showpost.php?p=1335767&postcount=395

and optionally separate the appended content with a horizontal rule separator. Eligible custom column types include text, comments, and composite.

# 10.7.7 Custom catalog covers



With the Generate Cover plugin<sup>114</sup> installed, you can create custom covers for your catalog. To install the plugin, go to *Preferences* > *Advanced* > *Plugins* > *Get new plugins*.

# 10.7.8 Additional help resources

For more information on calibre's Catalog feature, see the MobileRead forum sticky Creating Catalogs - Start here<sup>115</sup>, where you can find information on how to customize the catalog templates, and how to submit a bug report.

To ask questions or discuss calibre's Catalog feature with other users, visit the MobileRead forum Library Management<sup>116</sup>.

# **10.8 Virtual libraries**

In calibre, a Virtual library is a way to tell calibre to open only a subset of a normal library. For example, you might want to only work with books by a certain author, or books having only a certain tag. Using Virtual libraries is the preferred way of partitioning your large book collection into smaller sub collections. It is superior to splitting up your library into multiple smaller libraries as, when you want to search through your entire collection, you can simply go back to the full library. There is no way to search through multiple separate libraries simultaneously in calibre.

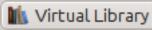
A Virtual library is different from a simple search. A search will only restrict the list of books shown in the book list. A Virtual library does that, and in addition it also restricts the entries shown in the *Tag browser* to the left. The Tag browser will only show tags, authors, series, publishers, etc. that come from the books in the Virtual library. A Virtual library thus behaves as though the actual library contains only the restricted set of books.

<sup>&</sup>lt;sup>114</sup> https://www.mobileread.com/forums/showthread.php?t=124219

<sup>&</sup>lt;sup>115</sup> https://www.mobileread.com/forums/showthread.php?t=118556

<sup>&</sup>lt;sup>116</sup> https://www.mobileread.com/forums/forumdisplay.php?f=236

# **10.8.1 Creating Virtual libraries**



To use a Virtual library click the *Virtual library* button located to the left of the Search bar and select the *Create Virtual library* option. As a first example, let's create a Virtual library that shows us only the books by a particular author. Click the *Authors* link as shown in the image below and choose the author you want to use and click OK.

Virtual library <u>n</u> ame: Search expression: Create a virtual library based on: Authors, bags, Publishers, Series, Saved Searches. Create a virtual library to assed on: Authors, bags, Publishers, Series, Saved Searches. Saved Searches. Virtual Libraries you can restrict calibre to only show you books that match a search. When a virtual library is in effect, calibre behaves as though the library contains only the matched books. The Tag Browser display only the tags/authors/series/etc. that belong to the matched books in the virtual library. This is a good way to partition your large library into smaller and easier to work with subsets. For example you can use a Virtual Library to only show you books with the Tag "Unread" or only books by "My Favorite Author" or only books in a particular series.	🗴 😒 📀 😳 Create vir	tual library 📀 🗽
V OK OK	Search expression:	Using virtual libraries you can restrict calibre to only show you books that match a search. When a virtual library is in effect, calibre behaves as though the library contains only the matched books. The Tag Browser display only the tags/authors/series/etc. that belong to the matched books and any searches you do will only search within the books in the virtual library. This is a good way to partition your large library into smaller and easier to work with subsets. For example you can use a Virtual Library to only show you books with the Tag "Unread" or only books by "My Favorite Author" or only books in a particular series.

The Create Virtual library dialog has been filled in for you. Click OK and you will see that a new Virtual library has been created, and automatically switched to, that displays only the books by the selected author. As far as calibre is concerned, it is as if your library contains only the books by the selected author.

You can switch back to the full library at any time by once again clicking the *Virtual library* and selecting the entry named *<None>*.

Virtual libraries are based on *searches*. You can use any search as the basis of a Virtual library. The Virtual library will contain only the books matched by that search. First, type in the search you want to use in the Search bar or build a search using the *Tag browser*. When you are happy with the returned results, click the *Virtual library* button, choose *Create library* and enter a name for the new Virtual library. The Virtual library will then be created based on the search you just typed in. Searches are very powerful, for examples of the kinds of things you can do with them, see *The search interface* (page 12).

#### **Examples of useful Virtual libraries**

- Books added to calibre in the last day:: date:>1daysago
- Books added to calibre in the last month:: date:>30daysago
- Books with a rating of 5 stars:: rating:5
- Books with a rating of at least 4 stars:: rating:>=4
- Books with no rating:: rating:false

- Periodicals downloaded by the Fetch News function in calibre:: tags:=News and author:=calibre
- Books with no tags:: tags:false
- Books with no covers:: cover:false

# 10.8.2 Working with Virtual libraries

You can edit a previously created Virtual library or remove it, by clicking the *Virtual library* and choosing the appropriate action.

You can tell calibre that you always want to apply a particular Virtual library when the current library is opened, by going to *Preferences*  $\rightarrow$  *Interface*  $\rightarrow$  *Behavior*.

You can quickly use the current search as a temporary Virtual library by clicking the *Virtual library* button and choosing the *\*current search* entry.

You can display all available Virtual libraries as tabs above the book list. This is particularly handy if you like switching between Virtual libraries very often. Click the *Virtual library* button and select *Show Virtual libraries as tabs*. You can re-arrange the tabs by drag and drop and close ones you do not want to see. Closed tabs can be restored by right-clicking on the tab bar.

# 10.8.3 Using Virtual libraries in searches

You can search for books that are in a Virtual library using the vl: prefix. For example, vl:Read will find all the books in the *Read* Virtual library. The search vl:Read and vl:"Science Fiction" will find all the books that are in both the *Read* and *Science Fiction* Virtual libraries.

The value following v1: must be the name of a Virtual library. If the Virtual library name contains spaces then surround it with quotes.

One use for a Virtual library search is in the Content server. In *Preferences*  $\rightarrow$  *Sharing over the net*  $\rightarrow$  *Require username and password* you can limit the calibre libraries visible to a user. For each visible library you can specify a search expression to further limit which books are seen. Use v1:"Virtual library name" to limit the books to those in a Virtual library.

# 10.8.4 Using additional restrictions

You can further restrict the books shown in a Virtual library by using *Additional restrictions*. An additional restriction is saved search you previously created that can be applied to the current Virtual library to further restrict the books shown in a Virtual library. For example, say you have a Virtual library for books tagged as *Historical Fiction* and a saved search that shows you unread books, you can click the *Virtual Library* button and choose the *Additional restriction* option to show only unread Historical Fiction books. To learn about saved searches, see *Saving searches* (page 17).

CHAPTER

## ELEVEN

## THE CALIBRE:// URL SCHEME

calibre registers itself as the handler program for calibre:// URLs. So you can use these to perform actions like opening books, searching for books, etc from other programs/documents or via the command line. For example, running the following at the command line:

calibre calibre://switch-library/Some\_Library

Will open calibre with the library named Some Library. Library names are the folder name of the library folder with spaces replaced by underscores. The special value \_ means the current library. The various types of URLs are documented below.

You can even place these links inside HTML files or Word documents or similar and the operating system will automatically run calibre to perform the specified action.

- Switch to a specific library (page 249)
- *Show a specific book in calibre* (page 250)
- Open a specific book in the E-book viewer at a specific position (page 250)
- Searching for books (page 250)
- Open a book details window on a book in some library (page 251)
- Open the notes associated with an author/series/etc. (page 251)
- Hex encoding of URL parameters (page 251)

## 11.1 Switch to a specific library

#### The URL syntax is:

calibre://switch-library/Library\_Name

Library names are the folder name of the library with spaces replaced by underscores. The special value \_ means the current library. You can also use *hex encoding* (page 251) for the library names, useful if the library names have special characters that would otherwise require URL encoding. Hex encoded library names look like:

\_hex\_-AD23F4BC

Where the part after the \_hex\_- prefix is the library name encoded as UTF-8 and every byte represented by two hexadecimal characters.

## 11.2 Show a specific book in calibre

The URL syntax is:

```
calibre://show-book/Library_Name/book_id
```

This will show the book with book\_id (a number) in calibre. The ids for books can be seen in the calibre interface by hovering over the *Click to open* link in the *Book details* panel, it is the number in brackets at the end of the path to the book folder.

You can copy a link to the current book displayed in calibre by right clicking the *Book details* panel and choosing *Copy link to book*.

If a search is active and the book is not matched by the search then the search is cleared.

If a Virtual library is selected, calibre will use it when showing the book. If the book isn't found in that virtual library then the virtual library is cleared.

If you want to switch to a particular Virtual library when showing the book, use:

replacing spaces in the Virtual library name by \$20. If the book isn't found in that virtual library then the virtual library is ignored.

## 11.3 Open a specific book in the E-book viewer at a specific position

The URL syntax is:

calibre://view-book/Library\_Name/book\_id/book\_format?open\_at=location

Here, book\_format is the format of the book, for example, EPUB or MOBI and the location is an optional location inside the book. The easiest way to get these links is to open a book in the viewer, then in the viewer controls select Go to  $\rightarrow$  Location and there such a link will be given that you can copy/paste elsewhere.

## 11.4 Searching for books

The URL syntax is:

```
calibre://search/Library_Name?q=query
calibre://search/Library_Name?eq=hex_encoded_query
```

Here query is any valid *search expression* (page 12). If the search expression is complicated, *encode it as a hex string* (page 251) and use eq instead. Leaving out the query will cause the current search to be cleared.

By default, if a Virtual library is selected, calibre will clear it before doing the search to ensure all books are found. If you want to preserve the Virtual library, use:

calibre://search/Library\_Name?q=query&virtual\_library=\_

If you want to switch to a particular Virtual library, use:

```
calibre://search/Library_Name?virtual_library=Library%20Name
or
calibre://search/Library_Name?encoded_virtual_library=hex_encoded_virtual_library_name
```

replacing spaces in the Virtual library name by %20.

If you perform a search in calibre and want to generate a link for it you can do so by right clicking the search bar and choosing *Copy search as URL*.

## 11.5 Open a book details window on a book in some library

The URL syntax is:

```
calibre://book-details/Library_Name/book_id
```

This opens a book details window on the specified book from the specified library without changing the current library or the selected book.

## 11.6 Open the notes associated with an author/series/etc.

The URL syntax is:

calibre://book-details/Library\_Name/Field\_Name/id\_Item\_Id

This opens a window showing the notes of the specified item. The easiest way to create such URLs is to show the notes you want in calibre and click the *Copy URL* button to copy the URL to the clipboard and paste it wherever you need.

Here Field\_Name is the name of the columns such as authors or tags. For user created columns, replace the leading # in the field name with an underscore, so #mytags becomes \_mytags.

In addition to specifying items by id using Item\_Id you can also specify them by name using either val\_Item\_Name or hex\_Hex\_Encoded\_Item\_Name. For example:

calibre://book-details/Library\_Name/authors/val\_John%20Doe

## 11.7 Hex encoding of URL parameters

Hex encoding of URL parameters is done by first encoding the parameter as UTF-8 bytes, and then replacing each byte by two hexadecimal characters representing the byte. For example, the string abc is the bytes  $0 \times 61$   $0 \times 62$  and  $0 \times 63$  in UTF-8, so the encoded version is the string: 616263.

### CHAPTER

TWELVE

## **CUSTOMIZING CALIBRE**

calibre has a highly modular design. Various parts of it can be customized. Here, you will learn:

- how to use environment variables and tweaks to customize calibre's behavior,
- · how to specify your own static resources like icons and templates to override the defaults
- how to use *plugins* to add functionality to calibre.
- how to share icon themes and plugins with other calibre users.
- to see how to create *recipes* to add new sources of online content to calibre visit the Section Adding your favorite news website (page 33).

### **1** Note

Note that although icon themes and plugins are indexed and downloadable via calibre's builtin updater, they are not part of calibre, and their canonical locations for support and source code are on the Mobileread forums<sup>117</sup> in their support threads.

- Environment variables (page 285)
- Tweaks (page 286)
- Overriding icons, templates, et cetera (page 298)
- Creating your own icon theme for calibre (page 299)
- Customizing calibre with plugins (page 299)

## 12.1 API documentation for plugins

Defines various abstract base classes that can be subclassed to create powerful plugins. The useful classes are:

- *Plugin* (page 254)
- *FileTypePlugin* (page 256)
- Metadata plugins (page 258)

<sup>&</sup>lt;sup>117</sup> https://www.mobileread.com/forums/forumdisplay.php?f=166

- Catalog plugins (page 259)
- Metadata download plugins (page 259)
- Conversion plugins (page 263)
- Device drivers (page 266)
- User interface actions (page 279)
- Preferences plugins (page 283)

## 12.1.1 Plugin

class calibre.customize.Plugin(plugin\_path)

A calibre plugin. Useful members include:

- self.installation\_type: Stores how the plugin was installed.
- self.plugin\_path: Stores path to the ZIP file that contains this plugin or None if it is a builtin plugin
- self.site\_customization: Stores a customization string entered by the user.

Methods that should be overridden in sub classes:

- initialize() (page 255)
- customization\_help() (page 255)

#### Useful methods:

- temporary\_file() (page 256)
- \_\_\_\_\_()
- load\_resources() (page 255)

#### supported\_platforms = []

List of platforms this plugin works on. For example: ['windows', 'osx', 'linux']

#### name = 'Trivial Plugin'

The name of this plugin. You must set it something other than Trivial Plugin for it to work.

#### version = (1, 0, 0)

The version of this plugin as a 3-tuple (major, minor, revision)

#### description = 'Does absolutely nothing'

A short string describing what this plugin does

#### author = 'Unknown'

The author of this plugin

#### priority = 1

When more than one plugin exists for a filetype, the plugins are run in order of decreasing priority. Plugins with higher priority will be run first. The highest possible priority is sys.maxsize. Default priority is 1.

#### minimum\_calibre\_version = (0, 4, 118)

The earliest version of calibre this plugin requires

#### installation\_type = None

The way this plugin is installed

#### can\_be\_disabled = True

If False, the user will not be able to disable this plugin. Use with care.

#### type = 'Base'

The type of this plugin. Used for categorizing plugins in the GUI

#### initialize()

Called once when calibre plugins are initialized. Plugins are re-initialized every time a new plugin is added. Also note that if the plugin is run in a worker process, such as for adding books, then the plugin will be initialized for every new worker process.

Perform any plugin specific initialization here, such as extracting resources from the plugin ZIP file. The path to the ZIP file is available as self.plugin\_path.

Note that self.site\_customization is not available at this point.

#### config\_widget()

Implement this method and *save\_settings()* (page 255) in your plugin to use a custom configuration dialog, rather then relying on the simple string based default customization.

This method, if implemented, must return a QWidget. The widget can have an optional method validate() that takes no arguments and is called immediately after the user clicks OK. Changes are applied if and only if the method returns True.

If for some reason you cannot perform the configuration at this time, return a tuple of two strings (message, details), these will be displayed as a warning dialog to the user and the process will be aborted.

#### save\_settings(config\_widget)

Save the settings specified by the user with config\_widget.

#### Parameters

config\_widget - The widget returned by config\_widget () (page 255).

#### do\_user\_config(parent=None)

This method shows a configuration dialog for this plugin. It returns True if the user clicks OK, False otherwise. The changes are automatically applied.

#### load\_resources(names)

If this plugin comes in a ZIP file (user added plugin), this method will allow you to load resources from the ZIP file.

For example to load an image:

#### Parameters

names - List of paths to resources in the ZIP file using / as separator

#### Returns

A dictionary of the form {name: file\_contents}. Any names that were not found in the ZIP file will not be present in the dictionary.

#### customization\_help(gui=False)

Return a string giving help on how to customize this plugin. By default raise a NotImplementedError, which indicates that the plugin does not require customization.

If you re-implement this method in your subclass, the user will be asked to enter a string as customization for this plugin. The customization string will be available as self.site\_customization.

Site customization could be anything, for example, the path to a needed binary on the user's computer.

#### **Parameters**

gui – If True return HTML help, otherwise return plain text help.

### temporary\_file (suffix)

Return a file-like object that is a temporary file on the file system. This file will remain available even after being closed and will only be removed on interpreter shutdown. Use the name member of the returned object to access the full path to the created temporary file.

#### Parameters

**suffix** – The suffix that the temporary file will have.

#### cli\_main(args)

This method is the main entry point for your plugins command line interface. It is called when the user does: calibre-debug -r "Plugin Name". Any arguments passed are present in the args variable.

## 12.1.2 FileTypePlugin

```
class calibre.customize.FileTypePlugin(plugin_path)
```

```
Bases: Plugin (page 254)
```

A plugin that is associated with a particular set of file types.

#### file\_types = {}

Set of file types for which this plugin should be run. Use '\*' for all file types. For example: {'lit', 'mobi', 'prc'}

#### on\_import = False

If True, this plugin is run when books are added to the database

#### on\_postimport = False

If True, this plugin is run after books are added to the database. In this case the postimport and postadd methods of the plugin are called.

#### on\_postconvert = False

If True, this plugin is run after a book is converted. In this case the postconvert method of the plugin is called.

```
on_postdelete = False
```

If True, this plugin is run after a book file is deleted from the database. In this case the postdelete method of the plugin is called.

```
on_preprocess = False
```

If True, this plugin is run just before a conversion

#### on\_postprocess = False

If True, this plugin is run after conversion on the final file produced by the conversion output plugin.

```
type = 'File type'
```

The type of this plugin. Used for categorizing plugins in the GUI

#### run (path\_to\_ebook)

Run the plugin. Must be implemented in subclasses. It should perform whatever modifications are required on the e-book and return the absolute path to the modified e-book. If no modifications are needed, it should return the path to the original e-book. If an error is encountered it should raise an Exception. The default implementation simply return the path to the original e-book. Note that the path to the original file (before any file type plugins are run, is available as self.original\_path\_to\_file).

The modified e-book file should be created with the temporary\_file() method.

#### Parameters

path\_to\_ebook - Absolute path to the e-book.

#### Returns

Absolute path to the modified e-book.

#### postimport(book\_id, book\_format, db)

Called post import, i.e., after the book file has been added to the database. Note that this is different from *postadd()* (page 257) which is called when the book record is created for the first time. This method is called whenever a new file is added to a book record. It is useful for modifying the book record based on the contents of the newly added file.

#### Parameters

- **book\_id** Database id of the added book.
- **book\_format** The file type of the book that was added.
- **db** Library database.

#### postconvert (book\_id, book\_format, db)

Called post conversion, i.e., after the conversion output book file has been added to the database. Note that it is run after a conversion only, not after a book is added. It is useful for modifying the book record based on the contents of the newly added file.

#### Parameters

- book\_id Database id of the added book.
- **book\_format** The file type of the book that was added.
- **db** Library database.

#### postdelete (book\_id, book\_format, db)

Called post deletion, i.e., after the book file has been deleted from the database. Note that it is not run when a book record is deleted, only when one or more formats from the book are deleted. It is useful for modifying the book record based on the format of the deleted file.

#### Parameters

- book\_id Database id of the added book.
- **book\_format** The file type of the book that was added.
- db Library database.

#### postadd(book\_id, fmt\_map, db)

Called post add, i.e. after a book has been added to the db. Note that this is different from *postimport()* (page 257), which is called after a single book file has been added to a book. postadd() is called only when an entire book record with possibly more than one book file has been created for the first time. This is useful if you wish to modify the book record in the database when the book is first added to calibre.

#### **Parameters**

- book\_id Database id of the added book.
- fmt\_map Map of file format to path from which the file format was added. Note that this
  might or might not point to an actual existing file, as sometimes files are added as streams.
  In which case it might be a dummy value or a non-existent path.
- db Library database

### 12.1.3 Metadata plugins

```
class calibre.customize.MetadataReaderPlugin(*args, **kwargs)
```

Bases: Plugin (page 254)

A plugin that implements reading metadata from a set of file types.

```
file_types = {}
```

Set of file types for which this plugin should be run. For example: set (['lit', 'mobi', 'prc'])

```
supported_platforms = ['windows', 'osx', 'linux']
```

List of platforms this plugin works on. For example: ['windows', 'osx', 'linux']

```
version = (8, 7, 0)
```

The version of this plugin as a 3-tuple (major, minor, revision)

```
author = 'Kovid Goyal'
```

The author of this plugin

```
type = 'Metadata reader'
```

The type of this plugin. Used for categorizing plugins in the GUI

#### get\_metadata(stream, type)

Return metadata for the file represented by stream (a file like object that supports reading). Raise an exception when there is an error with the input data.

#### Parameters

type – The type of file. Guaranteed to be one of the entries in *file\_types* (page 258).

Returns

A calibre.ebooks.metadata.book.Metadata object

```
class calibre.customize.MetadataWriterPlugin(*args, **kwargs)
```

```
Bases: Plugin (page 254)
```

A plugin that implements reading metadata from a set of file types.

```
file_types = {}
```

Set of file types for which this plugin should be run. For example: set (['lit', 'mobi', 'prc'])

```
supported_platforms = ['windows', 'osx', 'linux']
```

List of platforms this plugin works on. For example: ['windows', 'osx', 'linux']

```
version = (8, 7, 0)
```

The version of this plugin as a 3-tuple (major, minor, revision)

```
author = 'Kovid Goyal'
```

The author of this plugin

```
type = 'Metadata writer'
```

The type of this plugin. Used for categorizing plugins in the GUI

#### set\_metadata(stream, mi, type)

Set metadata for the file represented by stream (a file like object that supports reading). Raise an exception when there is an error with the input data.

#### Parameters

- type The type of file. Guaranteed to be one of the entries in *file\_types* (page 258).
- mi A calibre.ebooks.metadata.book.Metadata object

## 12.1.4 Catalog plugins

```
class calibre.customize.CatalogPlugin(plugin_path)
```

Bases: Plugin (page 254)

A plugin that implements a catalog generator.

#### file\_types = {}

Output file type for which this plugin should be run. For example: 'epub' or 'xml'

#### type = 'Catalog generator'

The type of this plugin. Used for categorizing plugins in the GUI

#### cli\_options = []

CLI parser options specific to this plugin, declared as namedtuple Option:

from collections import namedtuple Option = namedtuple('Option', 'option, default, dest, help') cli\_options = [Option('-catalog-title', default = 'My Catalog', dest = 'catalog\_title', help = (\_('Title of generated catalog. nDefault:') + " " + '%default' + ""))] cli\_options parsed in calibre.db.cli.cmd\_catalog:option\_parser()

#### initialize()

If plugin is not a built-in, copy the plugin's .ui and .py files from the ZIP file to \$TMPDIR. Tab will be dynamically generated and added to the Catalog Options dialog in calibre.gui2.dialogs.catalog.py:Catalog

run (path\_to\_output, opts, db, ids, notification=None)

Run the plugin. Must be implemented in subclasses. It should generate the catalog in the format specified in file\_types, returning the absolute path to the generated catalog file. If an error is encountered it should raise an Exception.

The generated catalog file should be created with the temporary\_file() method.

#### Parameters

- **path\_to\_output** Absolute path to the generated catalog file.
- opts A dictionary of keyword arguments
- db A LibraryDatabase2 object

## 12.1.5 Metadata download plugins

```
class calibre.ebooks.metadata.sources.base.Source(*args, **kwargs)
```

Bases: Plugin (page 254)

#### type = 'Metadata source'

The type of this plugin. Used for categorizing plugins in the GUI

```
author = 'Kovid Goyal'
```

The author of this plugin

#### supported\_platforms = ['windows', 'osx', 'linux']

List of platforms this plugin works on. For example: ['windows', 'osx', 'linux']

#### capabilities = frozenset({})

Set of capabilities supported by this plugin. Useful capabilities are: 'identify', 'cover'

#### touched\_fields = frozenset({})

List of metadata fields that can potentially be download by this plugin during the identify phase

#### has\_html\_comments = False

Set this to True if your plugin returns HTML formatted comments

#### supports\_gzip\_transfer\_encoding = False

Setting this to True means that the browser object will indicate that it supports gzip transfer encoding. This can speedup downloads but make sure that the source actually supports gzip transfer encoding correctly first

#### ignore\_ssl\_errors = False

Set this to True to ignore HTTPS certificate errors when connecting to this source.

#### cached\_cover\_url\_is\_reliable = True

Cached cover URLs can sometimes be unreliable (i.e. the download could fail or the returned image could be bogus). If that is often the case with this source, set to False

#### options = ()

A list of Option objects. They will be used to automatically construct the configuration widget for this plugin

#### config\_help\_message = None

A string that is displayed at the top of the config widget for this plugin

#### can\_get\_multiple\_covers = False

If True this source can return multiple covers for a given query

#### auto\_trim\_covers = False

If set to True covers downloaded by this plugin are automatically trimmed.

#### prefer\_results\_with\_isbn = True

If set to True, and this source returns multiple results for a query, some of which have ISBNs and some of which do not, the results without ISBNs will be ignored

#### is\_configured()

Return False if your plugin needs to be configured before it can be used. For example, it might need a username/password/API key.

#### customization\_help()

Return a string giving help on how to customize this plugin. By default raise a NotImplementedError, which indicates that the plugin does not require customization.

If you re-implement this method in your subclass, the user will be asked to enter a string as customization for this plugin. The customization string will be available as self.site\_customization.

Site customization could be anything, for example, the path to a needed binary on the user's computer.

#### Parameters

gui – If True return HTML help, otherwise return plain text help.

#### config\_widget()

Implement this method and *save\_settings()* (page 261) in your plugin to use a custom configuration dialog, rather then relying on the simple string based default customization.

This method, if implemented, must return a QWidget. The widget can have an optional method validate() that takes no arguments and is called immediately after the user clicks OK. Changes are applied if and only if the method returns True.

If for some reason you cannot perform the configuration at this time, return a tuple of two strings (message, details), these will be displayed as a warning dialog to the user and the process will be aborted.

```
save_settings (config_widget)
```

Save the settings specified by the user with config\_widget.

#### Parameters

config\_widget - The widget returned by config\_widget () (page 260).

#### get\_author\_tokens (authors, only\_first\_author=True)

Take a list of authors and return a list of tokens useful for an AND search query. This function tries to return tokens in first name middle names last name order, by assuming that if a comma is in the author name, the name is in lastname, other names form.

#### get\_title\_tokens (title, strip\_joiners=True, strip\_subtitle=False)

Take a title and return a list of tokens useful for an AND search query. Excludes connectives(optionally) and punctuation.

#### split\_jobs(jobs, num)

Split a list of jobs into at most num groups, as evenly as possible

#### test\_fields (mi)

Return the first field from self.touched\_fields that is null on the mi object

#### $\verb|clean_downloaded_metadata(mi)||$

Call this method in your plugin's identify method to normalize metadata before putting the Metadata object into result\_queue. You can of course, use a custom algorithm suited to your metadata source.

#### get\_book\_url(identifiers)

Return a 3-tuple or None. The 3-tuple is of the form: (identifier\_type, identifier\_value, URL). The URL is the URL for the book identified by identifiers at this source. identifier\_type, identifier\_value specify the identifier corresponding to the URL. This URL must be browsable to by a human using a browser. It is meant to provide a clickable link for the user to easily visit the books page at this source. If no URL is found, return None. This method must be quick, and consistent, so only implement it if it is possible to construct the URL from a known scheme given identifiers.

#### get\_book\_url\_name (idtype, idval, url)

Return a human readable name from the return value of get\_book\_url().

#### get\_book\_urls(identifiers)

Override this method if you would like to return multiple URLs for this book. Return a list of 3-tuples. By default this method simply calls *get\_book\_url()* (page 261).

#### get\_cached\_cover\_url (identifiers)

Return cached cover URL for the book identified by the identifiers dictionary or None if no such URL exists.

Note that this method must only return validated URLs, i.e. not URLS that could result in a generic cover image or a not found error.

id\_from\_url(url)

Parse a URL and return a tuple of the form: (identifier\_type, identifier\_value). If the URL does not match the pattern for the metadata source, return None.

identify\_results\_keygen(title=None, authors=None, identifiers={})

Return a function that is used to generate a key that can sort Metadata objects by their relevance given a search query (title, authors, identifiers).

These keys are used to sort the results of a call to *identify()* (page 262).

For details on the default algorithm see InternalMetadataCompareKeyGen (page 262). Re-implement this function in your plugin if the default algorithm is not suitable.

identify (log, result\_queue, abort, title=None, authors=None, identifiers={}, timeout=30)

Identify a book by its Title/Author/ISBN/etc.

If identifiers(s) are specified and no match is found and this metadata source does not store all related identifiers (for example, all ISBNs of a book), this method should retry with just the title and author (assuming they were specified).

If this metadata source also provides covers, the URL to the cover should be cached so that a subsequent call to the get covers API with the same ISBN/special identifier does not need to get the cover URL again. Use the caching API for this.

Every Metadata object put into result\_queue by this method must have a *source\_relevance* attribute that is an integer indicating the order in which the results were returned by the metadata source for this query. This integer will be used by <code>compare\_identify\_results()</code>. If the order is unimportant, set it to zero for every result.

Make sure that any cover/ISBN mapping information is cached before the Metadata object is put into result\_queue.

#### **Parameters**

- log A log object, use it to output debugging information/errors
- **result\_queue** A result Queue, results should be put into it. Each result is a Metadata object
- abort If abort.is\_set() returns True, abort further processing and return as soon as possible
- title The title of the book, can be None
- authors A list of authors of the book, can be None
- identifiers A dictionary of other identifiers, most commonly {'isbn':'1234...'}
- timeout Timeout in seconds, no network request should hang for longer than timeout.

#### Returns

None if no errors occurred, otherwise a unicode representation of the error suitable for showing to the user

#### 

Download a cover and put it into result\_queue. The parameters all have the same meaning as for *identify()* (page 262). Put (self, cover\_data) into result\_queue.

This method should use cached cover URLs for efficiency whenever possible. When cached data is not present, most plugins simply call identify and use its results.

If the parameter get\_best\_cover is True and this plugin can get multiple covers, it should only get the "best" one.

**class** calibre.ebooks.metadata.sources.base.**InternalMetadataCompareKeyGen** (*mi, source\_plugin, title, authors,* 

*identifiers*)

Generate a sort key for comparison of the relevance of Metadata objects, given a search query. This is used only to compare results from the same metadata source, not across different sources.

The sort key ensures that an ascending order sort is a sort by order of decreasing relevance.

The algorithm is:

- Prefer results that have at least one identifier the same as for the query
- · Prefer results with a cached cover URL
- · Prefer results with all available fields filled in
- · Prefer results with the same language as the current user interface language
- Prefer results that are an exact title match to the query
- Prefer results with longer comments (greater than 10% longer)
- Use the relevance of the result as reported by the metadata source's search engine

### 12.1.6 Conversion plugins

class calibre.customize.conversion.InputFormatPlugin(\*args)

Bases: Plugin (page 254)

InputFormatPlugins are responsible for converting a document into HTML+OPF+CSS+etc. The results of the conversion *must* be encoded in UTF-8. The main action happens in *convert()* (page 264).

#### type = 'Conversion input'

The type of this plugin. Used for categorizing plugins in the GUI

```
can_be_disabled = False
```

If False, the user will not be able to disable this plugin. Use with care.

```
supported_platforms = ['windows', 'osx', 'linux']
```

List of platforms this plugin works on. For example: ['windows', 'osx', 'linux']

#### file\_types = {}

Set of file types for which this plugin should be run For example: set (['azw', 'mobi', 'prc'])

#### is\_image\_collection = False

If True, this input plugin generates a collection of images, one per HTML file. This can be set dynamically, in the convert method if the input files can be both image collections and non-image collections. If you set this to True, you must implement the get\_images() method that returns a list of images.

```
core_usage = 1
```

Number of CPU cores used by this plugin. A value of -1 means that it uses all available cores

```
for_viewer = False
```

If set to True, the input plugin will perform special processing to make its output suitable for viewing

```
output_encoding = 'utf-8'
```

The encoding that this input plugin creates files in. A value of None means that the encoding is undefined and must be detected individually

#### common\_options = {<calibre.customize.conversion.OptionRecommendation object>}

Options shared by all Input format plugins. Do not override in sub-classes. Use *options* (page 263) instead. Every option must be an instance of OptionRecommendation.

#### options = {}

Options to customize the behavior of this plugin. Every option must be an instance of OptionRecommendation.

#### recommendations = {}

A set of 3-tuples of the form (option\_name, recommended\_value, recommendation\_level)

#### get\_images()

Return a list of absolute paths to the images, if this input plugin represents an image collection. The list of images is in the same order as the spine and the TOC.

### convert (stream, options, file\_ext, log, accelerators)

This method must be implemented in sub-classes. It must return the path to the created OPF file or an OEBBook instance. All output should be contained in the current folder. If this plugin creates files outside the current folder they must be deleted/marked for deletion before this method returns.

#### **Parameters**

- stream A file like object that contains the input file.
- **options** Options to customize the conversion process. Guaranteed to have attributes corresponding to all the options declared by this plugin. In addition, it will have a verbose attribute that takes integral values from zero upwards. Higher numbers mean be more verbose. Another useful attribute is input\_profile that is an instance of calibre.customize. profiles.InputProfile.
- file\_ext The extension (without the .) of the input file. It is guaranteed to be one of the *file\_types* supported by this plugin.
- $\log A$  calibre.utils.logging.Log object. All output should use this object.
- **accelerators** A dictionary of various information that the input plugin can get easily that would speed up the subsequent stages of the conversion.

#### postprocess\_book (oeb, opts, log)

Called to allow the input plugin to perform postprocessing after the book has been parsed.

#### specialize (oeb, opts, log, output\_fmt)

Called to allow the input plugin to specialize the parsed book for a particular output format. Called after postprocess\_book and before any transforms are performed on the parsed book.

#### gui\_configuration\_widget (parent, get\_option\_by\_name, get\_option\_help, db, book\_id=None)

Called to create the widget used for configuring this plugin in the calibre GUI. The widget must be an instance of the PluginWidget class. See the builtin input plugins for examples.

class calibre.customize.conversion.OutputFormatPlugin(\*args)

Bases: Plugin (page 254)

OutputFormatPlugins are responsible for converting an OEB document (OPF+HTML) into an output e-book.

The OEB document can be assumed to be encoded in UTF-8. The main action happens in convert () (page 265).

#### type = 'Conversion output'

The type of this plugin. Used for categorizing plugins in the GUI

#### can\_be\_disabled = False

If False, the user will not be able to disable this plugin. Use with care.

#### supported\_platforms = ['windows', 'osx', 'linux']

List of platforms this plugin works on. For example: ['windows', 'osx', 'linux']

#### file\_type = None

The file type (extension without leading period) that this plugin outputs

#### common\_options = {<calibre.customize.conversion.OptionRecommendation object>}

Options shared by all Input format plugins. Do not override in sub-classes. Use *options* (page 265) instead. Every option must be an instance of OptionRecommendation.

#### options = {}

Options to customize the behavior of this plugin. Every option must be an instance of OptionRecommendation.

#### recommendations = {}

A set of 3-tuples of the form (option\_name, recommended\_value, recommendation\_level)

#### property description

str(object=") -> str str(bytes\_or\_buffer[, encoding[, errors]]) -> str

Create a new string object from the given object. If encoding or errors is specified, then the object must expose a data buffer that will be decoded using the given encoding and error handler. Otherwise, returns the result of object.\_\_str\_\_() (if defined) or repr(object). encoding defaults to 'utf-8'. errors defaults to 'strict'.

#### convert (oeb\_book, output, input\_plugin, opts, log)

Render the contents of *oeb\_book* (which is an instance of calibre.ebooks.oeb.OEBBook) to the file specified by output.

#### **Parameters**

- **output** Either a file like object or a string. If it is a string it is the path to a folder that may or may not exist. The output plugin should write its output into that folder. If it is a file like object, the output plugin should write its output into the file.
- input\_plugin The input plugin that was used at the beginning of the conversion pipeline.
- opts Conversion options. Guaranteed to have attributes corresponding to the OptionRecommendations of this plugin.
- log The logger. Print debug/info messages etc. using this.

#### specialize\_options (log, opts, input\_fmt)

Can be used to change the values of conversion options, as used by the conversion pipeline.

#### specialize\_css\_for\_output (log, opts, item, stylizer)

Can be used to make changes to the CSS during the CSS flattening process.

#### Parameters

- item The item (HTML file) being processed
- stylizer A Stylizer object containing the flattened styles for item. You can get the style for any element by stylizer.style(element).

#### gui\_configuration\_widget (parent, get\_option\_by\_name, get\_option\_help, db, book\_id=None)

Called to create the widget used for configuring this plugin in the calibre GUI. The widget must be an instance of the PluginWidget class. See the builtin output plugins for examples.

## 12.1.7 Device drivers

The base class for all device drivers is *DevicePlugin* (page 266). However, if your device exposes itself as a USBMS drive to the operating system, you should use the USBMS class instead as it implements all the logic needed to support these kinds of devices.

class calibre.devices.interface.DevicePlugin(plugin\_path)

```
Bases: Plugin (page 254)
```

Defines the interface that should be implemented by backends that communicate with an e-book reader.

```
type = 'Device interface'
```

The type of this plugin. Used for categorizing plugins in the GUI

```
FORMATS = ['lrf', 'rtf', 'pdf', 'txt']
```

Ordered list of supported formats

```
VENDOR_ID = 0
```

VENDOR\_ID can be either an integer, a list of integers or a dictionary If it is a dictionary, it must be a dictionary of dictionaries, of the form:

```
integer_vendor_id : { product_id : [list of BCDs], ... },
...
}
```

#### $PRODUCT_ID = 0$

An integer or a list of integers

BCD = None

BCD can be either None to not distinguish between devices based on BCD, or it can be a list of the BCD numbers of all devices supported by this driver.

#### THUMBNAIL\_HEIGHT = 68

Height for thumbnails on the device

#### THUMBNAIL\_COMPRESSION\_QUALITY = 75

Compression quality for thumbnails. Set this closer to 100 to have better quality thumbnails with fewer compression artifacts. Of course, the thumbnails get larger as well.

#### WANTS\_UPDATED\_THUMBNAILS = False

Set this to True if the device supports updating cover thumbnails during sync\_booklists. Setting it to true will ask device.py to refresh the cover thumbnails during book matching

```
CAN_SET_METADATA = ['title', 'authors', 'collections']
```

Whether the metadata on books can be set via the GUI.

```
CAN_DO_DEVICE_DB_PLUGBOARD = False
```

Whether the device can handle device\_db metadata plugboards

```
path_sep = '/'
```

Path separator for paths to books on device

```
icon = 'reader.png'
```

Icon for this device

#### UserAnnotation

alias of Annotation

#### OPEN\_FEEDBACK\_MESSAGE = None

GUI displays this as a message if not None in the status bar. Useful if opening can take a long time

#### VIRTUAL\_BOOK\_EXTENSIONS = frozenset({})

Set of extensions that are "virtual books" on the device and therefore cannot be viewed/saved/added to library. For example: frozenset(['kobo'])

#### VIRTUAL\_BOOK\_EXTENSION\_MESSAGE = None

Message to display to user for virtual book extensions.

#### NUKE\_COMMENTS = None

Whether to nuke comments in the copy of the book sent to the device. If not None this should be short string that the comments will be replaced by.

#### MANAGES\_DEVICE\_PRESENCE = False

If True indicates that this driver completely manages device detection, ejecting and so forth. If you set this to True, you *must* implement the detect\_managed\_devices and debug\_managed\_device\_detection methods. A driver with this set to true is responsible for detection of devices, managing a blacklist of devices, a list of ejected devices and so forth. calibre will periodically call the detect\_managed\_devices() method and if it returns a detected device, calibre will call open(). open() will be called every time a device is returned even if previous calls to open() failed, therefore the driver must maintain its own blacklist of failed devices. Similarly, when ejecting, calibre will call eject() and then assuming the next call to detect\_managed\_devices() returns None, it will call post\_yank\_cleanup().

#### SLOW\_DRIVEINFO = False

If set the True, calibre will call the *get\_driveinfo()* (page 269) method after the books lists have been loaded to get the driveinfo.

#### ASK\_TO\_ALLOW\_CONNECT = False

If set to True, calibre will ask the user if they want to manage the device with calibre, the first time it is detected. If you set this to True you must implement get\_device\_uid() (page 272) and ignore\_connected\_device() (page 272) and get\_user\_blacklisted\_devices() (page 272) and set\_user\_blacklisted\_devices() (page 272)

#### user\_feedback\_after\_callback = None

Set this to a dictionary of the form {'title':title, 'msg':msg, 'det\_msg':detailed\_msg} to have calibre popup a message to the user after some callbacks are run (currently only upload\_books). Be careful to not spam the user with too many messages. This variable is checked after *every* callback, so only set it when you really need to.

#### classmethod get\_open\_popup\_message()

GUI displays this as a non-modal popup. Should be an instance of OpenPopupMessage

```
classmethod model_metadata() \rightarrow tuple[ModelMetadata, ...]
```

Metadata about all the actual device models this driver supports

#### is\_usb\_connected(devices\_on\_system, debug=False, only\_presence=False)

Return True, device\_info if a device handled by this plugin is currently connected.

#### Parameters

devices\_on\_system - List of devices currently connected

#### detect\_managed\_devices (devices\_on\_system, force\_refresh=False)

Called only if MANAGES\_DEVICE\_PRESENCE is True.

Scan for devices that this driver can handle. Should return a device object if a device is found. This object will be passed to the open() method as the connected\_device. If no device is found, return None. The returned object can be anything, calibre does not use it, it is only passed to open().

This method is called periodically by the GUI, so make sure it is not too resource intensive. Use a cache to avoid repeatedly scanning the system.

#### Parameters

- devices\_on\_system Set of USB devices found on the system.
- **force\_refresh** If True and the driver uses a cache to prevent repeated scanning, the cache must be flushed.

debug\_managed\_device\_detection (devices\_on\_system, output)

Called only if MANAGES\_DEVICE\_PRESENCE is True.

Should write information about the devices detected on the system to output, which is a file like object.

Should return True if a device was detected and successfully opened, otherwise False.

**reset** (key='-1', log\_packets=False, report\_progress=None, detected\_device=None)

#### **Parameters**

- **key** The key to unlock the device
- log\_packets If true the packet stream to/from the device is logged
- **report\_progress** Function that is called with a % progress (number between 0 and 100) for various tasks. If it is called with -1 that means that the task does not have any progress information
- **detected\_device** Device information from the device scanner

#### can\_handle\_windows (usbdevice, debug=False)

Optional method to perform further checks on a device to see if this driver is capable of handling it. If it is not it should return False. This method is only called after the vendor, product ids and the bcd have matched, so it can do some relatively time intensive checks. The default implementation returns True. This method is called only on Windows. See also *can\_handle()* (page 268).

Note that for devices based on USBMS this method by default delegates to *can\_handle()* (page 268). So you only need to override *can\_handle()* (page 268) in your subclass of USBMS.

#### **Parameters**

usbdevice - A usbdevice as returned by calibre.devices.winusb. scan\_usb\_devices()

#### can\_handle(device\_info, debug=False)

Unix version of can\_handle\_windows () (page 268).

#### **Parameters**

device\_info – Is a tuple of (vid, pid, bcd, manufacturer, product, serial number)

#### open (connected\_device, library\_uuid)

Perform any device specific initialization. Called after the device is detected but before any other functions that communicate with the device. For example: For devices that present themselves as USB Mass storage devices, this method would be responsible for mounting the device or if the device has been automounted, for finding out where it has been mounted. The method *calibre.devices.usbms.device.Device.open()* (page 276) has an implementation of this function that should serve as a good example for USB Mass storage devices.

This method can raise an OpenFeedback exception to display a message to the user.

#### Parameters

- connected\_device The device that we are trying to open. It is a tuple of (vendor id, product id, bcd, manufacturer name, product name, device serial number). However, some devices have no serial number and on Windows only the first three fields are present, the rest are None.
- **library\_uuid** The UUID of the current calibre library. Can be None if there is no library (for example when used from the command line).

#### eject()

Un-mount / eject the device from the OS. This does not check if there are pending GUI jobs that need to communicate with the device.

NOTE: That this method may not be called on the same thread as the rest of the device methods.

#### post\_yank\_cleanup()

Called if the user yanks the device without ejecting it first.

#### set\_progress\_reporter(report\_progress)

Set a function to report progress information.

#### Parameters

**report\_progress** – Function that is called with a % progress (number between 0 and 100) for various tasks. If it is called with -1 that means that the task does not have any progress information

#### get\_device\_information(end\_session=True)

Ask device for device information. See L{DeviceInfoQuery}.

#### Returns

(device name, device version, software version on device, MIME type) The tuple can optionally have a fifth element, which is a drive information dictionary. See usbms.driver for an example.

#### get\_driveinfo()

Return the driveinfo dictionary. Usually called from get\_device\_information(), but if loading the driveinfo is slow for this driver, then it should set SLOW\_DRIVEINFO. In this case, this method will be called by calibre after the book lists have been loaded. Note that it is not called on the device thread, so the driver should cache the drive info in the books() method and this function should return the cached data.

#### card\_prefix (end\_session=True)

Return a 2 element list of the prefix to paths on the cards. If no card is present None is set for the card's prefix. E.G. ('/place', '/place2') (None, 'place2') ('place', None) (None, None)

#### total\_space(end\_session=True)

#### Get total space available on the mountpoints:

- 1. Main memory
- 2. Memory Card A
- 3. Memory Card B

#### Returns

A 3 element list with total space in bytes of (1, 2, 3). If a particular device doesn't have any of these locations it should return 0.

free\_space (end\_session=True)

#### Get free space available on the mountpoints:

1. Main memory

- 2. Card A
- 3. Card B

#### Returns

A 3 element list with free space in bytes of (1, 2, 3). If a particular device doesn't have any of these locations it should return -1.

**books** (*oncard=None*, *end\_session=True*)

Return a list of e-books on the device.

#### Parameters

**oncard** – If 'carda' or 'cardb' return a list of e-books on the specific storage card, otherwise return list of e-books in main memory of device. If a card is specified and no books are on the card return empty list.

#### Returns

A BookList.

#### upload\_books (files, names, on\_card=None, end\_session=True, metadata=None)

Upload a list of books to the device. If a file already exists on the device, it should be replaced. This method should raise a FreeSpaceError if there is not enough free space on the device. The text of the FreeSpaceError must contain the word "card" if on\_card is not None otherwise it must contain the word "memory".

#### **Parameters**

- files A list of paths
- **names** A list of file names that the books should have once uploaded to the device. len(names) == len(files)
- metadata If not None, it is a list of Metadata objects. The idea is to use the metadata to determine where on the device to put the book. len(metadata) == len(files). Apart from the regular cover (path to cover), there may also be a thumbnail attribute, which should be used in preference. The thumbnail attribute is of the form (width, height, cover\_data as jpeg).

#### Returns

A list of 3-element tuples. The list is meant to be passed to add\_books\_to\_metadata() (page 270).

#### classmethod add\_books\_to\_metadata (locations, metadata, booklists)

Add locations to the booklists. This function must not communicate with the device.

#### Parameters

- locations Result of a call to L{upload\_books}
- metadata List of Metadata objects, same as for upload\_books() (page 270).
- booklists A tuple containing the result of calls to (books(oncard=None)(), books(oncard='carda')(), :meth`books(oncard='cardb')`).

#### delete\_books (paths, end\_session=True)

Delete books at paths on device.

#### classmethod remove\_books\_from\_metadata (paths, booklists)

Remove books from the metadata list. This function must not communicate with the device.

#### **Parameters**

• **paths** – paths to books on the device.

• **booklists** - A tuple containing the result of calls to (books(oncard=None)(), books(oncard='carda')(), :meth`books(oncard='cardb')`).

#### sync\_booklists (booklists, end\_session=True)

Update metadata on device.

#### Parameters

**booklists** - A tuple containing the result of calls to (books(oncard=None)(), books(oncard='carda')(), :meth`books(oncard='cardb')`).

#### get\_file (path, outfile, end\_session=True)

Read the file at path on the device and write it to outfile.

#### Parameters

outfile - file object like sys.stdout or the result of an open() (page 268) call.

#### classmethod config\_widget()

Should return a QWidget. The QWidget contains the settings for the device interface

#### classmethod save\_settings(settings\_widget)

Should save settings to disk. Takes the widget created in *config\_widget()* (page 271) and saves all settings to disk.

#### classmethod settings()

Should return an opts object. The opts object should have at least one attribute *format\_map* which is an ordered list of formats for the device.

#### set\_plugboards (plugboards, pb\_func)

provide the driver the current set of plugboards and a function to select a specific plugboard. This method is called immediately before add\_books and sync\_booklists.

#### pb\_func is a callable with the following signature::

def pb\_func(device\_name, format, plugboards)

You give it the current device name (either the class name or DEVICE\_PLUGBOARD\_NAME), the format you are interested in (a 'real' format or 'device\_db'), and the plugboards (you were given those by set\_plugboards, the same place you got this method).

#### Returns

None or a single plugboard instance.

#### set\_driveinfo\_name(location\_code, name)

Set the device name in the driveinfo file to 'name'. This setting will persist until the file is re-created or the name is changed again.

Non-disk devices should implement this method based on the location codes returned by the get\_device\_information() method.

#### prepare\_addable\_books (paths)

Given a list of paths, returns another list of paths. These paths point to addable versions of the books.

If there is an error preparing a book, then instead of a path, the position in the returned list for that book should be a three tuple: (original\_path, the exception instance, traceback)

#### $\texttt{startup}\left( \;\right)$

Called when calibre is starting the device. Do any initialization required. Note that multiple instances of the class can be instantiated, and thus \_\_init\_\_ can be called multiple times, but only one instance will have this method called. This method is called on the device thread, not the GUI thread.

#### shutdown()

Called when calibre is shutting down, either for good or in preparation to restart. Do any cleanup required. This method is called on the device thread, not the GUI thread.

#### get\_device\_uid()

Must return a unique id for the currently connected device (this is called immediately after a successful call to open()). You must implement this method if you set ASK\_TO\_ALLOW\_CONNECT = True

#### ignore\_connected\_device (uid)

Should ignore the device identified by uid (the result of a call to get\_device\_uid()) in the future. You must implement this method if you set ASK\_TO\_ALLOW\_CONNECT = True. Note that this function is called immediately after open(), so if open() caches some state, the driver should reset that state.

#### get\_user\_blacklisted\_devices()

Return map of device uid to friendly name for all devices that the user has asked to be ignored.

#### set\_user\_blacklisted\_devices(devices)

Set the list of device uids that should be ignored by this driver.

### specialize\_global\_preferences(device\_prefs)

Implement this method if your device wants to override a particular preference. You must ensure that all call sites that want a preference that can be overridden use device\_prefs['something'] instead of prefs['something']. Your method should call device\_prefs.set\_overrides(pref=val, pref=val, ...). Currently used for: metadata management (prefs['manage\_device\_metadata'])

#### set\_library\_info(library\_name, library\_uuid, field\_metadata)

Implement this method if you want information about the current calibre library. This method is called at startup and when the calibre library changes while connected.

#### is\_dynamically\_controllable()

Called by the device manager when starting plugins. If this method returns a string, then a) it supports the device manager's dynamic control interface, and b) that name is to be used when talking to the plugin.

This method can be called on the GUI thread. A driver that implements this method must be thread safe.

#### start\_plugin()

This method is called to start the plugin. The plugin should begin to accept device connections however it does that. If the plugin is already accepting connections, then do nothing.

This method can be called on the GUI thread. A driver that implements this method must be thread safe.

#### stop\_plugin()

This method is called to stop the plugin. The plugin should no longer accept connections, and should cleanup behind itself. It is likely that this method should call shutdown. If the plugin is already not accepting connections, then do nothing.

This method can be called on the GUI thread. A driver that implements this method must be thread safe.

#### get\_option (opt\_string, default=None)

Return the value of the option indicated by opt\_string. This method can be called when the plugin is not started. Return None if the option does not exist.

This method can be called on the GUI thread. A driver that implements this method must be thread safe.

### set\_option (opt\_string, opt\_value)

Set the value of the option indicated by opt\_string. This method can be called when the plugin is not started.

This method can be called on the GUI thread. A driver that implements this method must be thread safe.

#### is\_running()

Return True if the plugin is started, otherwise false

This method can be called on the GUI thread. A driver that implements this method must be thread safe.

synchronize\_with\_db(db, book\_id, book\_metadata, first\_call)

Called during book matching when a book on the device is matched with a book in calibre's db. The method is responsible for synchronizing data from the device to calibre's db (if needed).

The method must return a two-value tuple. The first value is a set of calibre book ids changed if calibre's database was changed or None if the database was not changed. If the first value is an empty set then the metadata for the book on the device is updated with calibre's metadata and given back to the device, but no GUI refresh of that book is done. This is useful when the calibre data is correct but must be sent to the device.

The second value is itself a 2-value tuple. The first value in the tuple specifies whether a book format should be sent to the device. The intent is to permit verifying that the book on the device is the same as the book in calibre. This value must be None if no book is to be sent, otherwise return the base file name on the device (a string like foobar.epub). Be sure to include the extension in the name. The device subsystem will construct a send\_books job for all books with not- None returned values. Note: other than to later retrieve the extension, the name is ignored in cases where the device uses a template to generate the file name, which most do. The second value in the returned tuple indicated whether the format is future-dated. Return True if it is, otherwise return False. calibre will display a dialog to the user listing all future dated books.

Extremely important: this method is called on the GUI thread. It must be threadsafe with respect to the device manager's thread.

book\_id: the calibre id for the book in the database. book\_metadata: the Metadata object for the book coming from the device. first\_call: True if this is the first call during a sync, False otherwise

```
class calibre.devices.interface.BookList (oncard, prefix, settings)
```

Bases: list

A list of books. Each Book object must have the fields

- 1. title
- 2. authors
- 3. size (file size of the book)
- 4. datetime (a UTC time tuple)
- 5. path (path on the device to the book)
- 6. thumbnail (can be None) thumbnail is either a str/bytes object with the image data or it should have an attribute image\_path that stores an absolute (platform native) path to the image
- 7. tags (a list of strings, can be empty).

#### supports\_collections()

Return True if the device supports collections for this book list.

add\_book (book, replace\_metadata)

Add the book to the booklist. Intent is to maintain any device-internal metadata. Return True if booklists must be sync'ed

#### remove\_book(book)

Remove a book from the booklist. Correct any device metadata at the same time

#### get\_collections (collection\_attributes)

Return a dictionary of collections created from collection\_attributes. Each entry in the dictionary is of the form collection name:[list of books]

The list of books is sorted by book title, except for collections created from series, in which case series\_index is used.

#### Parameters

collection\_attributes - A list of attributes of the Book object

#### **USB Mass Storage based devices**

The base class for such devices is *calibre.devices.usbms.driver.USBMS* (page 277). This class in turn inherits some of its functionality from its bases, documented below. A typical basic USBMS based driver looks like this:

```
from calibre.devices.usbms.driver import USBMS
class PDNOVEL (USBMS) :
    name = 'Pandigital Novel device interface'
   gui_name = 'PD Novel'
   description = _('Communicate with the Pandigital Novel')
    author = 'Kovid Goyal'
    supported_platforms = ['windows', 'linux', 'osx']
   FORMATS = ['epub', 'pdf']
   VENDOR_ID = [0 \times 18 d1]
    PRODUCT_ID = [0xb004]
   BCD
             = [0x224]
   THUMBNAIL_HEIGHT = 144
   EBOOK_DIR_MAIN = 'eBooks'
   SUPPORTS_SUB_DIRS = False
    def upload_cover(self, path, filename, metadata):
        coverdata = getattr(metadata, 'thumbnail', None)
        if coverdata and coverdata[2]:
            with open('%s.jpg' % os.path.join(path, filename), 'wb') as coverfile:
                coverfile.write(coverdata[2])
```

class calibre.devices.usbms.device.Device(plugin\_path)

Bases: DeviceConfig, DevicePlugin (page 266)

This class provides logic common to all drivers for devices that export themselves as USB Mass Storage devices. Provides implementations for mounting/ejecting of USBMS devices on all platforms.

#### $VENDOR_ID = 0$

VENDOR\_ID can be either an integer, a list of integers or a dictionary If it is a dictionary, it must be a dictionary of dictionaries, of the form:

```
{
  integer_vendor_id : { product_id : [list of BCDs], ... },
   ...
}
```

#### $PRODUCT_ID = 0$

An integer or a list of integers

#### BCD = None

BCD can be either None to not distinguish between devices based on BCD, or it can be a list of the BCD numbers of all devices supported by this driver.

#### WINDOWS\_MAIN\_MEM = None

String identifying the main memory of the device in the Windows PnP id strings This can be None, string, list of strings or compiled regex

#### WINDOWS\_CARD\_A\_MEM = None

String identifying the first card of the device in the Windows PnP id strings This can be None, string, list of strings or compiled regex

### WINDOWS\_CARD\_B\_MEM = None

String identifying the second card of the device in the Windows PnP id strings This can be None, string, list of strings or compiled regex

#### OSX\_MAIN\_MEM\_VOL\_PAT = None

Used by the new driver detection to disambiguate main memory from storage cards. Should be a regular expression that matches the main memory mount point assigned by macOS

#### MAX\_PATH\_LEN = 250

The maximum length of paths created on the device

#### NEWS\_IN\_FOLDER = True

Put news in its own folder

#### $\texttt{classmethod model_metadata()} \rightarrow tuple[ModelMetadata, ...]$

Metadata about all the actual device models this driver supports

**reset** (*key*='-1', *log\_packets*=False, *report\_progress*=None, *detected\_device*=None)

#### **Parameters**

- key The key to unlock the device
- **log\_packets** If true the packet stream to/from the device is logged
- **report\_progress** Function that is called with a % progress (number between 0 and 100) for various tasks. If it is called with -1 that means that the task does not have any progress information
- detected\_device Device information from the device scanner

#### set\_progress\_reporter(report\_progress)

Set a function to report progress information.

#### Parameters

**report\_progress** – Function that is called with a % progress (number between 0 and 100) for various tasks. If it is called with -1 that means that the task does not have any progress information

#### card\_prefix (end\_session=True)

Return a 2 element list of the prefix to paths on the cards. If no card is present None is set for the card's prefix. E.G. ('/place', '/place2') (None, 'place2') ('place', None) (None, None)

#### total\_space (end\_session=True)

#### Get total space available on the mountpoints:

- 1. Main memory
- 2. Memory Card A
- 3. Memory Card B

#### Returns

A 3 element list with total space in bytes of (1, 2, 3). If a particular device doesn't have any of these locations it should return 0.

#### free\_space (end\_session=True)

#### Get free space available on the mountpoints:

- 1. Main memory
- 2. Card A
- 3. Card B

#### Returns

A 3 element list with free space in bytes of (1, 2, 3). If a particular device doesn't have any of these locations it should return -1.

#### windows\_sort\_drives (drives)

Called to disambiguate main memory and storage card for devices that do not distinguish between them on the basis of *WINDOWS\_CARD\_NAME*. For example: The EB600

#### can\_handle\_windows (usbdevice, debug=False)

Optional method to perform further checks on a device to see if this driver is capable of handling it. If it is not it should return False. This method is only called after the vendor, product ids and the bcd have matched, so it can do some relatively time intensive checks. The default implementation returns True. This method is called only on Windows. See also can\_handle().

Note that for devices based on USBMS this method by default delegates to can\_handle(). So you only need to override can\_handle() in your subclass of USBMS.

#### Parameters

usbdevice - A usbdevice as returned by calibre.devices.winusb. scan\_usb\_devices()

open (connected\_device, library\_uuid)

Perform any device specific initialization. Called after the device is detected but before any other functions that communicate with the device. For example: For devices that present themselves as USB Mass storage devices, this method would be responsible for mounting the device or if the device has been automounted, for finding out where it has been mounted. The method *calibre.devices.usbms.device.Device. open()* (page 276) has an implementation of this function that should serve as a good example for USB Mass storage devices.

This method can raise an OpenFeedback exception to display a message to the user.

#### Parameters

• **connected\_device** – The device that we are trying to open. It is a tuple of (vendor id, product id, bcd, manufacturer name, product name, device serial number). However, some

devices have no serial number and on Windows only the first three fields are present, the rest are None.

• **library\_uuid** – The UUID of the current calibre library. Can be None if there is no library (for example when used from the command line).

eject()

Un-mount / eject the device from the OS. This does not check if there are pending GUI jobs that need to communicate with the device.

NOTE: That this method may not be called on the same thread as the rest of the device methods.

#### post\_yank\_cleanup()

Called if the user yanks the device without ejecting it first.

#### sanitize\_callback(path)

Callback to allow individual device drivers to override the path sanitization used by create\_upload\_path().

#### filename\_callback (default, mi)

Callback to allow drivers to change the default file name set by create\_upload\_path().

#### sanitize\_path\_components(components)

Perform any device specific sanitization on the path components for files to be uploaded to the device

#### get\_annotations (path\_map)

Resolve path\_map to annotation\_map of files found on the device

#### add\_annotation\_to\_library(db, db\_id, annotation)

Add an annotation to the calibre library

```
class calibre.devices.usbms.cli.CLI
```

class calibre.devices.usbms.driver.USBMS(plugin\_path)

Bases: CLI (page 277), Device (page 274)

The base class for all USBMS devices. Implements the logic for sending/getting/updating metadata/caching metadata/etc.

#### description = 'Communicate with an e-book reader.'

A short string describing what this plugin does

author = 'John Schember'

The author of this plugin

#### supported\_platforms = ['windows', 'osx', 'linux']

List of platforms this plugin works on. For example: ['windows', 'osx', 'linux']

```
booklist_class
```

alias of BookList

### book\_class

alias of Book

### FORMATS = []

Ordered list of supported formats

#### CAN\_SET\_METADATA = []

Whether the metadata on books can be set via the GUI.

#### get\_device\_information (end\_session=True)

Ask device for device information. See L{DeviceInfoQuery}.

#### Returns

(device name, device version, software version on device, MIME type) The tuple can optionally have a fifth element, which is a drive information dictionary. See usbms.driver for an example.

### set\_driveinfo\_name(location\_code, name)

Set the device name in the driveinfo file to 'name'. This setting will persist until the file is re-created or the name is changed again.

Non-disk devices should implement this method based on the location codes returned by the get\_device\_information() method.

#### books (oncard=None, end\_session=True)

Return a list of e-books on the device.

#### Parameters

**oncard** – If 'carda' or 'cardb' return a list of e-books on the specific storage card, otherwise return list of e-books in main memory of device. If a card is specified and no books are on the card return empty list.

### Returns

A BookList.

#### upload\_books (files, names, on\_card=None, end\_session=True, metadata=None)

Upload a list of books to the device. If a file already exists on the device, it should be replaced. This method should raise a FreeSpaceError if there is not enough free space on the device. The text of the FreeSpaceError must contain the word "card" if on\_card is not None otherwise it must contain the word "memory".

#### Parameters

- files A list of paths
- **names** A list of file names that the books should have once uploaded to the device. len(names) == len(files)
- metadata If not None, it is a list of Metadata objects. The idea is to use the metadata to determine where on the device to put the book. len(metadata) == len(files). Apart from the regular cover (path to cover), there may also be a thumbnail attribute, which should be used in preference. The thumbnail attribute is of the form (width, height, cover\_data as jpeg).

#### Returns

A list of 3-element tuples. The list is meant to be passed to add\_books\_to\_metadata() (page 278).

#### upload\_cover(path, filename, metadata, filepath)

Upload book cover to the device. Default implementation does nothing.

#### Parameters

- path The full path to the folder where the associated book is located.
- filename The name of the book file without the extension.
- metadata metadata belonging to the book. Use metadata.thumbnail for cover
- filepath The full path to the e-book file

add\_books\_to\_metadata (locations, metadata, booklists)

Add locations to the booklists. This function must not communicate with the device.

#### Parameters

- locations Result of a call to L{upload\_books}
- metadata List of Metadata objects, same as for upload\_books () (page 278).
- **booklists** A tuple containing the result of calls to (books(oncard=None)(), books(oncard='carda')(), :meth`books(oncard='cardb')`).

#### delete\_books (paths, end\_session=True)

Delete books at paths on device.

#### remove\_books\_from\_metadata(paths, booklists)

Remove books from the metadata list. This function must not communicate with the device.

#### Parameters

- **paths** paths to books on the device.
- **booklists** A tuple containing the result of calls to (books(oncard=None)(), books(oncard='carda')(), :meth`books(oncard='cardb')`).

#### sync\_booklists (booklists, end\_session=True)

Update metadata on device.

#### **Parameters**

**booklists** – A tuple containing the result of calls to (books(oncard=None)(), books(oncard='carda')(), :meth`books(oncard='cardb')`).

#### classmethod normalize\_path(path)

Return path with platform native path separators

## 12.1.8 User interface actions

If you are adding your own plugin in a ZIP file, you should subclass both InterfaceActionBase and InterfaceAction. The load\_actual\_plugin() method of your InterfaceActionBase subclass must return an instantiated object of your InterfaceBase subclass.

class calibre.gui2.actions.InterfaceAction(parent, site\_customization)

Bases: QObject

A plugin representing an "action" that can be taken in the graphical user interface. All the items in the toolbar and context menus are implemented by these plugins.

Note that this class is the base class for these plugins, however, to integrate the plugin with calibre's plugin system, you have to make a wrapper class that references the actual plugin. See the calibre.customize.builtins module for examples.

If two InterfaceAction (page 279) objects have the same name, the one with higher priority takes precedence.

Sub-classes should implement the genesis() (page 282), library\_changed() (page 282), location\_selected() (page 282), shutting\_down() (page 282), initialization\_complete() (page 282) and tag\_browser\_context\_action() (page 282) methods.

Once initialized, this plugin has access to the main calibre GUI via the gui member. You can access other plugins by name, for example:

self.gui.iactions['Save To Disk']

To access the actual plugin, use the interface\_action\_base\_plugin attribute, this attribute only becomes available after the plugin has been initialized. Useful if you want to use methods from the plugin class like do\_user\_config().

The QAction specified by action\_spec (page 280) is automatically create and made available as self.qaction.

#### name = 'Implement me'

The plugin name. If two plugins with the same name are present, the one with higher priority takes precedence.

#### priority = 1

The plugin priority. If two plugins with the same name are present, the one with higher priority takes precedence.

#### popup\_type = 1

The menu popup type for when this plugin is added to a toolbar

#### auto\_repeat = False

Whether this action should be auto repeated when its shortcut key is held down.

#### action\_spec = ('text', 'icon', None, None)

Of the form: (text, icon\_path, tooltip, keyboard shortcut). icon, tooltip and keyboard shortcut can be None. keyboard shortcut must be either a string, None or tuple of shortcuts. If None, a keyboard shortcut corresponding to the action is not registered. If you pass an empty tuple, then the shortcut is registered with no default key binding.

#### action\_shortcut\_name = None

If not None, used for the name displayed to the user when customizing the keyboard shortcuts for the above action spec instead of action\_spec[0]

#### action\_add\_menu = False

If True, a menu is automatically created and added to self.qaction

#### action\_menu\_clone\_qaction = False

If True, a clone of self.qaction is added to the menu of self.qaction If you want the text of this action to be different from that of self.qaction, set this variable to the new text

#### dont\_add\_to = frozenset({})

Set of locations to which this action must not be added. See all\_locations for a list of possible locations

#### dont\_remove\_from = frozenset({})

Set of locations from which this action must not be removed. See all\_locations for a list of possible locations

#### action\_type = 'global'

Type of action 'current' means acts on the current view 'global' means an action that does not act on the current view, but rather on calibre as a whole

#### accepts\_drops = False

If True, then this InterfaceAction will have the opportunity to interact with drag and drop events. See the methods, *accept\_enter\_event()* (page 280), :meth`:accept\_drag\_move\_event`, *drop\_event()* (page 281) for details.

#### accept\_enter\_event(event, mime\_data)

This method should return True iff this interface action is capable of handling the drag event. Do not call accept/ignore on the event, that will be taken care of by the calibre UI.

#### accept\_drag\_move\_event (event, mime\_data)

This method should return True iff this interface action is capable of handling the drag event. Do not call accept/ignore on the event, that will be taken care of by the calibre UI.

#### drop\_event (event, mime\_data)

This method should perform some useful action and return True iff this interface action is capable of handling the drop event. Do not call accept/ignore on the event, that will be taken care of by the calibre UI. You should not perform blocking/long operations in this function. Instead emit a signal or use QTimer.singleShot and return quickly. See the builtin actions for examples.

### create\_menu\_action (menu, unique\_name, text, icon=None, shortcut=None, description=None, triggered=None, shortcut\_name=None, persist\_shortcut=False)

Convenience method to easily add actions to a QMenu. Returns the created QAction. This action has one extra attribute calibre\_shortcut\_unique\_name which if not None refers to the unique name under which this action is registered with the keyboard manager.

#### Parameters

- menu The QMenu the newly created action will be added to
- unique\_name A unique name for this action, this must be globally unique, so make it as descriptive as possible. If in doubt, add an UUID to it.
- text The text of the action.
- icon Either a QIcon or a file name. The file name is passed to the QIcon.ic() builtin, so you do not need to pass the full path to the images folder.
- **shortcut** A string, a list of strings, None or False. If False, no keyboard shortcut is registered for this action. If None, a keyboard shortcut with no default keybinding is registered. String and list of strings register a shortcut with default keybinding as specified.
- description A description for this action. Used to set tooltips.
- triggered A callable which is connected to the triggered signal of the created action.
- **shortcut\_name** The text displayed to the user when customizing the keyboard shortcuts for this action. By default it is set to the value of text.
- **persist\_shortcut** Shortcuts for actions that don't always appear, or are library dependent, may disappear when other keyboard shortcuts are edited unless `persist\_shortcut` is set True.

#### load\_resources(names)

If this plugin comes in a ZIP file (user added plugin), this method will allow you to load resources from the ZIP file.

For example to load an image:

#### **Parameters**

names – List of paths to resources in the ZIP file using / as separator

#### Returns

A dictionary of the form {name : file\_contents}. Any names that were not found in the ZIP file will not be present in the dictionary.

#### genesis()

Setup this plugin. Only called once during initialization. self.gui is available. The action specified by *ac-tion\_spec* (page 280) is available as self.qaction.

#### location\_selected(loc)

Called whenever the book list being displayed in calibre changes. Currently values for loc are: library, main, card and cardb.

This method should enable/disable this action and its sub actions as appropriate for the location.

#### library\_about\_to\_change(*olddb*, *db*)

Called whenever the current library is changed.

#### Parameters

- olddb The LibraryDatabase corresponding to the previous library.
- db The LibraryDatabase corresponding to the new library.

#### $library_changed(db)$

Called whenever the current library is changed.

#### Parameters

db – The LibraryDatabase corresponding to the current library.

#### gui\_layout\_complete()

Called once per action when the layout of the main GUI is completed. If your action needs to make changes to the layout, they should be done here, rather than in *initialization\_complete()* (page 282).

#### initialization\_complete()

Called once per action when the initialization of the main GUI is completed.

#### tag\_browser\_context\_action(index)

Called when displaying the context menu in the Tag browser. index is the QModelIndex that points to the Tag browser item that was right clicked. Test it for validity with index.valid() and get the underlying TagTreeItem object with index.data(Qt.ItemDataRole.UserRole). Any action objects yielded by this method will be added to the context menu.

#### shutting\_down()

Called once per plugin when the main GUI is in the process of shutting down. Release any used resources, but try not to block the shutdown for long periods of time.

#### class calibre.customize.InterfaceActionBase(\*args, \*\*kwargs)

#### Bases: *Plugin* (page 254)

#### supported\_platforms = ['windows', 'osx', 'linux']

List of platforms this plugin works on. For example: ['windows', 'osx', 'linux']

#### author = 'Kovid Goyal'

The author of this plugin

#### type = 'User interface action'

The type of this plugin. Used for categorizing plugins in the GUI

#### can\_be\_disabled = False

If False, the user will not be able to disable this plugin. Use with care.

### load\_actual\_plugin(gui)

This method must return the actual interface action plugin object.

## **12.1.9 Preferences plugins**

```
class calibre.customize.PreferencesPlugin(plugin_path)
```

Bases: Plugin (page 254)

A plugin representing a widget displayed in the Preferences dialog.

This plugin has only one important method *create\_widget()* (page 283). The various fields of the plugin control how it is categorized in the UI.

```
supported_platforms = ['windows', 'osx', 'linux']
```

List of platforms this plugin works on. For example: ['windows', 'osx', 'linux']

author = 'Kovid Goyal'

The author of this plugin

#### type = 'Preferences'

The type of this plugin. Used for categorizing plugins in the GUI

#### can\_be\_disabled = False

If False, the user will not be able to disable this plugin. Use with care.

#### config\_widget = None

Import path to module that contains a class named ConfigWidget which implements the ConfigWidgetInterface. Used by create\_widget() (page 283).

#### category\_order = 100

Where in the list of categories the *category* (page 283) of this plugin should be.

#### name\_order = 100

Where in the list of names in a category, the gui\_name (page 283) of this plugin should be

#### category = None

The category this plugin should be in

#### gui\_category = None

The category name displayed to the user for this plugin

#### gui\_name = None

The name displayed to the user for this plugin

#### icon = None

The icon for this plugin, should be an absolute path

#### description = None

The description used for tooltips and the like

#### create\_widget(parent=None)

Create and return the actual Qt widget used for setting this group of preferences. The widget must implement the calibre.gui2.preferences.ConfigWidgetInterface (page 283).

The default implementation uses *config\_widget* (page 283) to instantiate the widget.

#### class calibre.gui2.preferences.ConfigWidgetInterface

This class defines the interface that all widgets displayed in the Preferences dialog must implement. See *Config-WidgetBase* (page 284) for a base class that implements this interface and defines various convenience methods as well.

#### changed\_signal = None

This signal must be emitted whenever the user changes a value in this widget

#### supports\_restoring\_to\_defaults = True

Set to True iff the restore\_to\_defaults() method is implemented.

# restore\_defaults\_desc = 'Restore settings to default values. You have to click Apply to actually save the default settings.'

The tooltip for the "Restore defaults" button

#### restart\_critical = False

If True the Preferences dialog will not allow the user to set any more preferences. Only has effect if *commit()* (page 284) returns True.

#### genesis (gui)

Called once before the widget is displayed, should perform any necessary setup.

#### Parameters

gui - The main calibre graphical user interface

#### initialize()

Should set all config values to their initial values (the values stored in the config files). A "return" statement is optional. Return False if the dialog is not to be shown.

#### restore\_defaults()

Should set all config values to their defaults.

#### commit()

Save any changed settings. Return True if the changes require a restart, False otherwise. Raise an Abort-Commit exception to indicate that an error occurred. You are responsible for giving the user feedback about what the error is and how to correct it.

#### refresh\_gui (gui)

Called once after this widget is committed. Responsible for causing the gui to reread any changed settings. Note that by default the GUI re-initializes various elements anyway, so most widgets won't need to use this method.

#### initial\_tab\_changed()

Called if the initially displayed tab is changed before the widget is shown, but after it is initialized.

#### class calibre.gui2.preferences.ConfigWidgetBase(parent=None)

Base class that contains code to easily add standard config widgets like checkboxes, combo boxes, text fields and so on. See the *register()* (page 285) method.

This class automatically handles change notification, resetting to default, translation between gui objects and config objects, etc. for registered settings.

If your config widget inherits from this class but includes setting that are not registered, you should override the *ConfigWidgetInterface* (page 283) methods and call the base class methods inside the overrides.

#### changed\_signal

This signal must be emitted whenever the user changes a value in this widget

#### supports\_restoring\_to\_defaults = True

Set to True iff the restore\_to\_defaults() method is implemented.

## restart\_critical = False

If True the Preferences dialog will not allow the user to set any more preferences. Only has effect if *commit()* (page 285) returns True.

Register a setting.

## **Parameters**

- name The setting name
- config\_obj The config object that reads/writes the setting
- gui\_name The name of the GUI object that presents an interface to change the setting. By default it is assumed to be 'opt\_' + name.
- choices If this setting is a multiple choice (combobox) based setting, the list of choices. The list is a list of two element tuples of the form: [(gui name, value), ...]
- **setting** The class responsible for managing this setting. The default class handles almost all cases, so this param is rarely used.

## initialize()

Should set all config values to their initial values (the values stored in the config files). A "return" statement is optional. Return False if the dialog is not to be shown.

## commit (\*args)

Save any changed settings. Return True if the changes require a restart, False otherwise. Raise an Abort-Commit exception to indicate that an error occurred. You are responsible for giving the user feedback about what the error is and how to correct it.

## restore\_defaults(\*args)

Should set all config values to their defaults.

# **12.2 Environment variables**

- CALIBRE\_CONFIG\_DIRECTORY sets the folder where configuration files are stored/read.
- CALIBRE\_TEMP\_DIR sets the temporary folder used by calibre
- CALIBRE\_CACHE\_DIRECTORY sets the folder calibre uses to cache persistent data between sessions
- CALIBRE\_OVERRIDE\_DATABASE\_PATH allows you to specify the full path to metadata.db. Using this variable you can have metadata.db be in a location other than the library folder. Useful if your library folder is on a networked drive that does not support file locking.
- CALIBRE\_DEVELOP\_FROM used to run from a calibre development environment. See Setting up a calibre development environment (page 357).
- CALIBRE\_OVERRIDE\_LANG used to force the language used by the interface (ISO 639 language code)
- CALIBRE\_TEST\_TRANSLATION used to test a translation .po file (should be the path to the .po file)
- CALIBRE\_NO\_NATIVE\_FILEDIALOGS causes calibre to not use native file dialogs for selecting files/folders.
- CALIBRE\_NO\_NATIVE\_MENUBAR causes calibre to not create a native (global) menu on Ubuntu Unity and similar Linux desktop environments. The menu is instead placed inside the window, as is traditional.
- CALIBRE\_USE\_SYSTEM\_THEME by default, on Linux, calibre uses its own builtin Qt style. This is to avoid crashes and hangs caused by incompatibilities between the version of Qt calibre is built against and the system Qt. The

downside is that calibre may not follow the system look and feel. If you set this environment variable on Linux, it will cause calibre to use the system theme – beware of crashes and hangs.

- CALIBRE\_SHOW\_DEPRECATION\_WARNINGS causes calibre to print deprecation warnings to stdout. Useful for calibre developers.
- CALIBRE\_NO\_DEFAULT\_PROGRAMS prevent calibre from automatically registering the filetypes it is capable of handling with Windows.
- CALIBRE\_USE\_SYSTEM\_CERTIFICATES make calibre use the system certificate store for SSL certificate verification instead of its own certificate store on Windows and macOS.
- CALIBRE\_NO\_ICONS\_IN\_MENUS Disable icons in menus
- QT\_QPA\_PLATFORM On Linux set this to wayland to force calibre to use Wayland and xcb to force use of X11.
- SYSFS\_PATH Use if sysfs is mounted somewhere other than /sys
- http\_proxy, https\_proxy used on Linux to specify an HTTP(S) proxy

See How to set environment variables in Windows<sup>118</sup>. If you are on macOS you can set environment variables by creating the ~/Library/Preferences/calibre/macos-env.txt and putting the environment variables one per line in it, for example:

```
CALIBRE_DEVELOP_FROM=$HOME/calibre-src/src
CALIBRE_NO_NATIVE_FILEDIALOGS=1
CALIBRE_CONFIG_DIRECTORY=~/.config/calibre
```

# 12.3 Tweaks

Tweaks are small changes that you can specify to control various aspects of calibre's behavior. You can change them by going to Preferences->Advanced->Tweaks. The default values for the tweaks are reproduced below

```
#!/usr/bin/env python
# vim:fileencoding=UTF-8:ts=4:sw=4:sta:et:sts=4:ai
# License: GPLv3 Copyright: 2010, Kovid Goyal <kovid at kovidgoyal.net>
\# Contains various tweaks that affect calibre behavior. Only edit this file if
# you know what you are doing. If you delete this file, it will be recreated from
# defaults.
#: Auto increment series index
# The algorithm used to assign a book added to an existing series a series number.
# New series numbers assigned using this tweak are always integer values, except
# if a constant non-integer is specified.
# Possible values are:
  next - First available integer larger than the largest existing number
#
  first_free - First available integer larger than 0
#
  next_free - First available integer larger than the smallest existing number
#
   last_free - First available integer smaller than the largest existing number...
→Return largest existing + 1 if no free number is found
  const - Assign the number 1 always
#
   no_change - Do not change the series index
#
   a number - Assign that number always. The number is not in quotes. Note that 0.0-
```

<sup>&</sup>lt;sup>118</sup> https://www.computerhope.com/issues/ch000549.htm

```
\rightarrow can be used here.
# Examples:
  series_index_auto_increment = 'next'
#
#
  series_index_auto_increment = 'next_free'
  series_index_auto_increment = 16.5
#
# Set the use_series_auto_increment_tweak_when_importing tweak to True to
# use the above values when importing/adding books. If this tweak is set to
# False (the default) then the series number will be set to 1 if it is not
# explicitly set during the import. If set to True, then the
# series index will be set according to the series_index_auto_increment setting.
# Note that the use_series_auto_increment_tweak_when_importing tweak is used
# only when a value is not provided during import. If the importing regular
# expression produces a value for series_index, or if you are reading metadata
# from books and the import plugin produces a value, then that value will
# be used irrespective of the setting of the tweak.
series_index_auto_increment = 'next'
use_series_auto_increment_tweak_when_importing = False
#: Add separator after completing an author name
# Set this if the completion separator should be appended to the end of the
# completed text to automatically begin a new completion operation for authors.
# It can be either True or False
authors_completer_append_separator = False
#: Author sort name algorithm
# The algorithm used to copy author to author_sort.
# Possible values are:
  invert: use "fn ln" -> "ln, fn"
# copy : copy author to author_sort without modification
# comma : use 'copy' if there is a ',' in the name, otherwise use 'invert'
  nocomma : "fn ln" -> "ln fn" (without the comma)
# When this tweak is changed, the author_sort values stored with each author
# must be recomputed by right-clicking on an author in the left-hand tags
# panel, selecting 'Manage authors', and pressing
# 'Recalculate all author sort values'.
# The author_name_suffixes are words that are ignored when they occur at the
# end of an author name. The case of the suffix is ignored and trailing
# periods are automatically handled.
#
# The same is true for author_name_prefixes.
#
# The author_name_copywords are a set of words which, if they occur in an
# author name, cause the automatically generated author sort string to be
# identical to the author's name. This means that the sort for a string like
# "Acme Inc." will be "Acme Inc." instead of "Inc., Acme".
# If author_use_surname_prefixes is enabled, any of the words in
# author_surname_prefixes will be treated as a prefix to the surname, if they
# occur before the surname. So for example, "John von Neumann" would be sorted
# as "von Neumann, John" and not "Neumann, John von".
```

```
(continued from previous page)
author_sort_copy_method = 'comma'
author_name_suffixes = ('Jr', 'Sr', 'Inc', 'Ph.D', 'Phd',
                        'MD', 'M.D', 'I', 'II', 'III', 'IV',
                        'Junior', 'Senior')
author_name_prefixes = ('Mr', 'Mrs', 'Ms', 'Dr', 'Prof')
author_name_copywords = (
    'Agency', 'Corporation', 'Company', 'Co.', 'Council',
    'Committee', 'Inc.', 'Institute', 'National', 'Society', 'Club', 'Team',
    'Software', 'Games', 'Entertainment', 'Media', 'Studios',
)
author_use_surname_prefixes = False
author_surname_prefixes = ('da', 'de', 'di', 'la', 'le', 'van', 'von')
#: Splitting multiple author names
# By default, calibre splits a string containing multiple author names on
# ampersands and the words "and" and "with". You can customize the splitting
# by changing the regular expression below. Strings are split on whatever the
# specified regular expression matches, in addition to ampersands.
# Default: r'(?i),?\s+(and|with)\s+'
authors_split_regex = r'(?i),?\s+(and|with)\s+'
#: Use author_sort and/or series_sort for names in Tag browser
# Set which author or series field is used to display as the item name in the
# Tag browser (the list of authors, series, publishers etc on the left hand
# side). The choices for author are 'author' and 'author_sort'. The choices for
# series are 'series' and 'series_sort'. This tweak affects only what is
# displayed under the category in the Tag browser and Content server. Please
# note that if you set this to ..._sort, it is possible to see duplicate names in
# the list because although it is guaranteed that author and series names are
# unique, there is no such guarantee for their sort values. Showing duplicates
# won't break anything but it could lead to some confusion. The tooltip for an
# item will show the item's name no matter how you set this tweak.
# Examples:
  categories_use_field_for_author_name = 'author'
#
#
  categories_use_field_for_author_name = 'author_sort'
   categories_use_field_for_series_name = 'series'
#
   categories_use_field_for_series_name = 'series_sort'
#
categories_use_field_for_author_name = 'author'
categories_use_field_for_series_name = 'series'
#: Control partitioning of Tag browser
# When partitioning the Tag browser, the format of the subcategory label is
# controlled by a template: categories_collapsed_name_template if sorting by
# name, categories_collapsed_rating_template if sorting by average rating, and
# categories_collapsed_popularity_template if sorting by popularity. There are
# two variables available to the template: first and last. The variable 'first'
\# is the initial item in the subcategory, and the variable 'last' is the final
# item in the subcategory. Both variables are 'objects'; they each have multiple
# values that are obtained by using a suffix. For example, first.name for an
# author category will be the name of the author. The sub-values available are:
#
  name: the printable name of the item
```

```
# count: the number of books that references this item
```

```
(continued from previous page)
  avg_rating: the average rating of all the books referencing this item
  sort: the sort value. For authors, this is the author_sort for that author
  category: the category (e.g., authors, series) that the item is in.
#
# Note that the "r'" in front of the { is necessary if there are backslashes
# (\ characters) in the template. It doesn't hurt anything to leave it there
# even if there aren't any backslashes.
categories_collapsed_name_template = r'{first.sort:shorten(4,,0)} - {last.
\leftrightarrow sort: shorten (4,,0) }'
categories_collapsed_rating_template = r'{first.avg_rating:4.2f:ifempty(0)} - {last.
→avg_rating:4.2f:ifempty(0) }'
categories_collapsed_popularity_template = r'{first.count:d} - {last.count:d}'
#: Specify columns to sort the booklist by on startup
# Provide a set of columns to be sorted on when calibre starts.
# The argument is None if saved sort history is to be used
# otherwise it is a list of column, order pairs. Column is the
# lookup/search name, found using the tooltip for the column
# Order is 0 for ascending, 1 for descending.
# For example, set it to [('authors',0),('title',0)] to sort by
# title within authors.
sort_columns_at_startup = None
#: Control how dates are displayed
# Format to be used for publication date and the timestamp (date).
  A string controlling how the publication date is displayed in the GUI
#
        the day as number without a leading zero (1 to 31)
#
  d
  dd
        the day as number with a leading zero (01 to 31)
  ddd
        the abbreviated localized day name (e.g. 'Mon' to 'Sun')
#
  dddd the long localized day name (e.g. 'Monday' to 'Sunday')
        the month as number without a leading zero (1-12)
# M
        the month as number with a leading zero (01-12)
# MM
  MMM the abbreviated localized month name (e.g. 'Jan' to 'Dec')
#
  MMMM the long localized month name (e.g. 'January' to 'December')
#
# VV
        the year as two digit number (00-99)
  yyyy the year as four digit number
#
         the hours without a leading 0 (0 to 11 or 0 to 23, depending on am/pm)
#
  h
  hh
         the hours with a leading 0 (00 to 11 or 00 to 23, depending on am/pm)
#
#
         the minutes without a leading 0 (0 to 59)
  т
         the minutes with a leading 0 (00 to 59)
# mm
         the seconds without a leading 0 (0 to 59)
#
  S
        the seconds with a leading 0 (00 to 59)
#
  SS
        use a 12-hour clock instead of a 24-hour clock, with "ap" replaced by the.
# ap
→lowercase localized string for am or pm
         use a 12-hour clock instead of a 24-hour clock, with "AP" replaced by the.
  AP
#
\leftrightarrowuppercase localized string for AM or PM
# aP
        use a 12-hour clock instead of a 24-hour clock, with "aP" replaced by the.
\hookrightarrow localized string for am or pm
        use a 12-hour clock instead of a 24-hour clock, with "Ap" replaced by the_
#
  Ap
⇔localized string for AM or PM
\# iso the date with time and timezone. Must be the only format present
  For example, given the date of 9 Jan 2010, the following formats show
  MMM yyyy ==> Jan 2010 yyyy ==> 2010 dd MMM yyyy ==> 09 Jan 2010
                                                                          (continues on next page)
```

```
(continued from previous page)
                           d/M/VV = > 9/1/10
  MM/yyyy = > 01/2010
                                               _{VV} ==> 10
# publication default if not set: MMM yyyy
# timestamp default if not set: dd MMM yyyy
# last_modified_display_format if not set: dd MMM yyyy
gui_pubdate_display_format = 'MMM yyyy'
gui_timestamp_display_format = 'dd MMM yyyy'
gui_last_modified_display_format = 'dd MMM yyyy'
#: Control sorting of titles and series in the library display
# Control title and series sorting in the library view. If set to
# 'library_order', the title sort field will be used instead of the title.
# Unless you have manually edited the title sort field, leading articles such as
# The and A will be ignored. If set to 'strictly_alphabetic', the titles will be
# sorted as-is (sort by title instead of title sort). For example, with
# library_order, The Client will sort under 'C'. With strictly_alphabetic, the
# book will sort under 'T'.
# This flag affects calibre's library display. It has no effect on devices. In
# addition, titles for books added before changing the flag will retain their
# order until the title is edited. Editing a title and hitting Enter
# without changing anything is sufficient to change the sort. Or you can use
# the 'Update title sort' action in the Bulk metadata edit dialog to update
# it for many books at once.
title_series_sorting = 'library_order'
#: Control formatting of title and series when used in templates
# Control how title and series names are formatted when saving to disk/sending
# to device. The behavior depends on the field being processed. If processing
# title, then if this tweak is set to 'library_order', the title will be
# replaced with title_sort. If it is set to 'strictly_alphabetic', then the
# title will not be changed. If processing series, then if set to
# 'library_order', articles such as 'The' and 'An' will be moved to the end. If
# set to 'strictly_alphabetic', the series will be sent without change.
# For example, if the tweak is set to library_order, "The Lord of the Rings"
# will become "Lord of the Rings, The". If the tweak is set to
# strictly_alphabetic, it would remain "The Lord of the Rings". Note that the
# formatter function raw_field will return the base value for title and
# series regardless of the setting of this tweak.
save_template_title_series_sorting = 'library_order'
#: Set the list of words considered to be "articles" for sort strings
# Set the list of words that are to be considered 'articles' when computing the
# title sort strings. The articles differ by language. By default, calibre uses
# a combination of articles from English and whatever language the calibre user
# interface is set to. In addition, in some contexts where the book language is
# available, the language of the book is used. You can change the list of
# articles for a given language or add a new language by editing
# per_language_title_sort_articles. To tell calibre to use a language other
# than the user interface language, set, default_language_for_title_sort. For
# example, to use German, set it to 'deu'. A value of None means the user
# interface language is used. The setting title_sort_articles is ignored
# (present only for legacy reasons).
```

```
per_language_title_sort_articles = {
        # English
        'eng': (r'A\s+', r'The\s+', r'An\s+'),
        # Esperanto
        'epo': (r'La\s+', r"L'", 'L´'),
        # Spanish
        'spa': (r'El\s+', r'La\s+', r'Lo\s+', r'Los\s+', r'Las\s+', r'Un\s+',
                r'Una\s+', r'Unos\s+', r'Unas\s+'),
        # French
        'fra': (r'Le\s+', r'La\s+', r"L'", r'L'', r'L'', r'Les\s+', r'Un\s+', r'Une\s+
\rightarrow',
                r'Des\s+', r'De\s+La\s+', r'De\s+', r"D'", r'D'', r'D''),
        # Polish
        'pol': (),
        # Italian
        'ita': (r'Lo\s+', r'Il\s+', r"L'", r'L´', r'La\s+', r'Gli\s+',
                r'I\s+', r'Le\s+', r'Uno\s+', r'Un\s+', r'Una\s+', "rUn'",
                r'Un'', r'Dei\s+', r'Degli\s+', r'Delle\s+', r'Del\s+',
                r'Della\s+', r'Dello\s+', r"Dell'", r'Dell''),
        # Portuguese
        'por': (r'A\s+', r'O\s+', r'Os\s+', r'As\s+', r'Um\s+', r'Uns\s+',
                r'Uma\s+', r'Umas\s+'),
        # Romanian
        'ron': (r'Un\s+', r'O\s+', r'Nişte\s+'),
        # German
        'deu': (r'Der\s+', r'Die\s+', r'Das\s+', r'Den\s+', r'Ein\s+',
                r'Eine\s+', r'Einen\s+', r'Dem\s+', r'Des\s+', r'Einem\s+',
                r'Eines\s+'),
        # Dutch
        'nld': (r'De\s+', r'Het\s+', r'Een\s+', r"'n\s+", r"'s\s+", r'Ene\s+',
                r'Ener\s+', r'Enes\s+', r'Den\s+', r'Der\s+', r'Des\s+',
                r"'t > +"),
        # Swedish
        'swe': (r'En\s+', r'Ett\s+', r'Det\s+', r'Den\s+', r'De\s+'),
        # Turkish
        'tur': (r'Bir\s+',),
        # Afrikaans
        'afr': (r"'n\s+", r'Die\s+'),
        # Greek
        'ell': (r'O\s+', r'I\s+', r'To\s+', r'Ta\s+', r'Tus\s+', r'Tis\s+',
                r"'Enas\s+", r"'Mia\s+", r"'Ena\s+", r"'Enan\s+"),
        # Hungarian
        'hun': (r'A \setminus s+', r'Az \setminus s+', r'Egy \setminus s+'),
}
default_language_for_title_sort = None
title_sort_articles = r'^ (A | The | An) \s+'
#: Specify a folder calibre should connect to at startup
# Specify a folder that calibre should connect to at startup using
# connect_to_folder. This must be a full path to the folder. If the folder does
# not exist when calibre starts, it is ignored.
# Example for Windows:
```

```
(continued from previous page)
     auto_connect_to_folder = 'C:/Users/someone/Desktop/testlib'
# Example for other operating systems:
     auto_connect_to_folder = '/home/dropbox/My Dropbox/someone/library'
#
auto_connect_to_folder = ''
#: Specify renaming rules for SONY collections
# Specify renaming rules for SONY collections. This tweak is only applicable if
# metadata management is set to automatic. Collections on SONYs are named
# depending upon whether the field is standard or custom. A collection derived
# from a standard field is named for the value in that field.
# For example, if the standard 'series' column contains the value 'Darkover', then the
# collection name is 'Darkover'. A collection derived from a custom field will
# have the name of the field added to the value. For example, if a custom series
# column named 'My Series' contains the name 'Darkover', then the collection
# will by default be named 'Darkover (My Series)'. For purposes of this
# documentation, 'Darkover' is called the value and 'My Series' is called the
# category. If two books have fields that generate the same collection name,
# then both books will be in that collection.
#
# This set of tweaks lets you specify for a standard or custom field how
# the collections are to be named. You can use it to add a description to a
# standard field, for example 'Foo (Tag)' instead of the 'Foo'. You can also use
# it to force multiple fields to end up in the same collection.
# For example, you could force the values in 'series', '#my_series_1', and
# '#my_series_2' to appear in collections named 'some_value (Series)', thereby
# merging all of the fields into one set of collections.
# There are two related tweaks. The first determines the category name to use
# for a metadata field. The second is a template, used to determines how the
# value and category are combined to create the collection name.
# The syntax of the first tweak, sony_collection_renaming_rules, is:
# {'field_lookup_name':'category_name_to_use', 'lookup_name':'name', ...}
# The second tweak, sony_collection_name_template, is a template. It uses the
# same template language as plugboards and save templates. This tweak controls
# how the value and category are combined together to make the collection name.
# The only two fields available are {category} and {value}. The {value} field is
# never empty. The {category} field can be empty. The default is to put the
# value first, then the category enclosed in parentheses, it isn't empty:
# '{value} {category: |(|)}'
# Examples: The first three examples assume that the second tweak
# has not been changed.
# 1) I want three series columns to be merged into one set of collections. The
# column lookup names are 'series', '#series_1' and '#series_2'. I want nothing
# in the parenthesis. The value to use in the tweak value would be:
#
    sony_collection_renaming_rules={'series':'', '#series_1':'', '#series_2':''}
# 2) I want the word '(Series)' to appear on collections made from series, and
```

```
(continues on next page)
```

```
# the word '(Tag)' to appear on collections made from tags. Use:
  sony_collection_renaming_rules={'series':'Series', 'tags':'Tag'}
#
#
# 3) I want 'series' and '#myseries' to be merged, and for the collection name
# to have '(Series)' appended. The renaming rule is:
  sony_collection_renaming_rules={'series':'Series', '#myseries':'Series'}
#
# 4) Same as example 2, but instead of having the category name in parentheses
# and appended to the value, I want it prepended and separated by a colon, such
# as in Series: Darkover. I must change the template used to format the category name
# The resulting two tweaks are:
    sony_collection_renaming_rules={'series':'Series', 'tags':'Tag'}
#
    sony_collection_name_template='{category:||: }{value}'
#
sony_collection_renaming_rules = {}
sony_collection_name_template = '{value}{category:| (|)}'
#: Specify how SONY collections are sorted
# Specify how SONY collections are sorted. This tweak is only applicable if
# metadata management is set to automatic. You can indicate which metadata is to
# be used to sort on a collection-by-collection basis. The format of the tweak
# is a list of metadata fields from which collections are made, followed by the
# name of the metadata field containing the sort value.
# Example: The following indicates that collections built from pubdate and tags
# are to be sorted by the value in the custom column '#mydate', that collections
# built from 'series' are to be sorted by 'series_index', and that all other
# collections are to be sorted by title. If a collection metadata field is not
# named, then if it is a series- based collection it is sorted by series order,
# otherwise it is sorted by title order.
# [(['pubdate', 'tags'],'#mydate'), (['series'],'series_index'), (['*'], 'title')]
# Note that the bracketing and parentheses are required. The syntax is
# [ ( [list of fields], sort field ) , ( [ list of fields ] , sort field ) ]
# Default: empty (no rules), so no collection attributes are named.
sony_collection_sorting_rules = []
#: Control how tags are applied when copying books to another library
# Set this to True to ensure that tags in 'Tags to add when adding
# a book' are added when copying books to another library
add_new_book_tags_when_importing_books = False
#: Set the maximum number of sort 'levels'
# Set the maximum number of sort 'levels' that calibre will use to resort the
# library after certain operations such as searches or device insertion. Each
# sort level adds a performance penalty. If the database is large (thousands of
# books) the penalty might be noticeable. If you are not concerned about multi-
# level sorts, and if you are seeing a slowdown, reduce the value of this tweak.
maximum_resort_levels = 5
#: Choose whether dates are sorted using visible fields
# Date values contain both a date and a time. When sorted, all the fields are
# used, regardless of what is displayed. Set this tweak to True to use only
# the fields that are being displayed.
```

```
sort_dates_using_visible_fields = False
#: Fuzz value for trimming covers
# The value used for the fuzz distance when trimming a cover.
# Colors within this distance are considered equal.
# The distance is in absolute intensity units.
cover_trim_fuzz_value = 10
#: Control behavior of the book list
# You can control the behavior of double clicks and pressing Enter on the books
# list. Choices: open_viewer, do_nothing, show_book_details,
# show_locked_book_details, edit_cell, edit_metadata. Selecting anything other
# than open_viewer, show_book_details, or show_locked_book_details has the side
# effect of disabling editing a field using a single click.
# Default: open_viewer.
# Example: doubleclick_on_library_view = 'do_nothing'
# You can also control whether the book list scrolls per item or
# per pixel. Default is per item.
doubleclick_on_library_view = 'open_viewer'
enter_key_behavior = 'do_nothing'
horizontal_scrolling_per_column = False
vertical_scrolling_per_row = False
#: Language to use when sorting
# Setting this tweak will force sorting to use the
# collating order for the specified language. This might be useful if you run
# calibre in English but want sorting to work in the language where you live.
# Set the tweak to the desired ISO 639-1 language code, in lower case.
# You can find the list of supported locales at
# https://en.wikipedia.org/wiki/List_of_ISO_639-1_codes
# Default: locale_for_sorting = '' -- use the language calibre displays in
# Example: locale_for_sorting = 'fr' -- sort using French rules.
# Example: locale_for_sorting = 'nb' -- sort using Norwegian rules.
locale_for_sorting = ''
#: The number of seconds to wait before sending emails
# The number of seconds to wait before sending emails when using a
# public email server like GMX/Hotmail/Gmail. Default is: 5 minutes
# Setting it to lower may cause the server's SPAM controls to kick in,
# making email sending fail. Changes will take effect only after a restart of
# calibre. You can also change the list of hosts that calibre considers
# to be public relays here. Any relay host ending with one of the suffixes
# in the list below will be considered a public email server.
public_smtp_relay_delay = 301
public_smtp_relay_host_suffixes = ['gmail.com', 'live.com', 'gmx.com', 'outlook.com']
#: The maximum width and height for covers saved in the calibre library
# All covers in the calibre library will be resized, preserving aspect ratio,
# to fit within this size. This is to prevent slowdowns caused by extremely
# large covers
maximum_cover_size = (1650, 2200)
```

```
#: Where to send downloaded news
# When automatically sending downloaded news to a connected device, calibre
# will by default send it to the main memory. By changing this tweak, you can
# control where it is sent. Valid values are "main", "carda", "cardb". Note
# that if there isn't enough free space available on the location you choose,
# the files will be sent to the location with the most free space.
send_news_to_device_location = 'main'
#: Unified toolbar on macOS
# If you enable this option and restart calibre, the toolbar will be 'unified'
# with the titlebar as is normal for macOS applications. However, doing this has
# various bugs, for instance the minimum width of the toolbar becomes twice
# what it should be and it causes other random bugs on some systems, so turn it
# on at your own risk!
unified_title_toolbar_on_osx = False
#: Save original file when converting/polishing from same format to same format
# When calibre does a conversion from the same format to the same format, for
# example, from EPUB to EPUB, the original file is saved, so that in case the
# conversion is poor, you can tweak the settings and run it again. By setting
# this to False you can prevent calibre from saving the original file.
# Similarly, by setting save_original_format_when_polishing to False you can
# prevent calibre from saving the original file when polishing.
save_original_format = True
save_original_format_when_polishing = True
#: Number of recently viewed books to show
# Right-clicking the "View" button shows a list of recently viewed books. Control
# how many should be shown, here.
gui_view_history_size = 15
#: Change the font size of the Book details panel in the interface
# Change the font size at which book details are rendered in the side panel and
# comments are rendered in the metadata edit dialog. Set it to a positive or
# negative number to increase or decrease the font size.
change_book_details_font_size_by = 0
#: What format to default to when using the "Unpack book" feature
# The "Unpack book" feature of calibre allows direct editing of a book format.
# If multiple formats are available, calibre will offer you a choice
# of formats, defaulting to your preferred output format if it is available.
# Set this tweak to a specific value of 'EPUB' or 'AZW3' to always default
# to that format rather than your output format preference.
# Set to a value of 'remember' to use whichever format you chose last time you
# used the "Unpack book" feature.
# Examples:
#
  default_tweak_format = None
                                      (Use output format)
#
   default_tweak_format = 'EPUB'
  default_tweak_format = 'remember'
default tweak format = None
#: Do not preselect a completion when editing authors/tags/series/etc.
```

```
# This means that you can make changes and press Enter and your changes will
\# not be overwritten by a matching completion. However, if you wish to use the
# completions you will now have to press Tab to select one before pressing
# Enter. Which technique you prefer will depend on the state of metadata in
# your library and your personal editing style.
# If preselect_first_completion is False and you want Tab to accept what you
# typed instead of the first completion then set tab_accepts_uncompleted_text
# to True. If you do this then to select from the completions you must press
# the Down or Up arrow keys. The tweak tab accepts uncompleted text is ignored
# if preselect_first_completion is True
preselect_first_completion = False
tab_accepts_uncompleted_text = False
#: Completion mode when editing authors/tags/series/etc.
# By default, when completing items, calibre will show you all the candidates
# that start with the text you have already typed. You can instead have it show
\# all candidates that contain the text you have already typed. To do this, set
# completion_mode to 'contains'. For example, if you type asi it will match both
# Asimov and Quasimodo, whereas the default behavior would match only Asimov.
completion_mode = 'prefix'
#: Sort the list of libraries alphabetically
# The list of libraries in the Copy to library and Quick switch menus are
# normally sorted by most used. However, if there are more than a certain
# number of such libraries, the sorting becomes alphabetic. You can set that
# number here. The default is ten libraries.
many_libraries = 10
#: Choose available output formats for conversion
# Restrict the list of available output formats in the conversion dialogs.
# For example, if you only want to convert to EPUB and AZW3, change this to
# restrict_output_formats = ['EPUB', 'AZW3']. The default value of None causes
# all available output formats to be present.
restrict_output_formats = None
#: Set the thumbnail image quality used by the Content server
# The quality of a thumbnail is largely controlled by the compression quality
# used when creating it. Set this to a larger number to improve the quality.
# Note that the thumbnails get much larger with larger compression quality
# numbers.
# The value can be between 50 and 99
content_server_thumbnail_compression_quality = 75
#: Image file types to treat as e-books when dropping onto the "Book details" panel
# Normally, if you drop any image file in a format known to calibre onto the
# "Book details" panel, it will be used to set the cover. If you want to store
# some image types as e-books instead, you can set this tweak.
# Examples:
# cover_drop_exclude = {'tiff', 'webp'}
cover_drop_exclude = ()
```

```
#: Exclude fields when copy/pasting metadata
# You can ask calibre to not paste some metadata fields when using the
# Edit metadata->Copy metadata/Paste metadata actions. For example,
# exclude_fields_on_paste = ['cover', 'timestamp', '#mycolumn']
# to prevent pasting of the cover, Date and custom column, mycolumn.
# You can also add a shortcut in Preferences->Shortcuts->Edit metadata
# to paste metadata ignoring this tweak.
exclude_fields_on_paste = []
#: Skip internet connected check
# Skip checking whether the internet is available before downloading news.
# Useful if for some reason your operating systems network checking
# facilities are not reliable (for example NetworkManager on Linux).
skip_network_check = False
#: Tab stop width in the template editor
# Sets the width of the tab stop in the template editor in "average characters".
# For example, a value of 1 results in a space with the width of one average.
\hookrightarrow character.
template_editor_tab_stop_width = 4
#: Value for undefined numbers when sorting
# Sets the value to use for undefined numbers when sorting.
\# For example, the value -10 sorts undefined numbers as if they were set to -10.
# Use 'maximum' for the largest possible number. Use 'minimum' for the smallest
# possible number. Quotes are optional if entering a number.
# Examples:
#
  value_for_undefined_numbers_when_sorting = -100
    value_for_undefined_numbers_when_sorting = '2'
#
  value_for_undefined_numbers_when_sorting = -0.01
#
  value_for_undefined_numbers_when_sorting = 'minimum'
#
  value_for_undefined_numbers_when_sorting = 'maximum'
value_for_undefined_numbers_when_sorting = 0
#: Allow template database functions in composite columns
# If True then the template database functions book_values() and book_count()
# can be used in composite custom columns. Note: setting this tweak to True and
# using these functions in composites can be very slow.
# Default: False
allow_template_database_functions_in_composites = False
#: Change the programs that are run when opening files/URLs
# By default, calibre passes URLs to the operating system to open using
# whatever default programs are configured there. Here you can override
# that by specifying the program to use, per URL type. For local files,
# the type is "file" and for web links it is "http*". For example:
# openers_by_scheme = { "http*": "firefox %u" } will make calibre run Firefox
# for https://whatever URLs. %u is replaced by the URL to be opened. The scheme
# takes a glob pattern allowing a single entry to match multiple URL types.
openers_by_scheme = {}
```

```
#: Set the first day of the week for calendar popups
# It must be one of the values Default, Sunday, Monday, Tuesday, Wednesday,
# Thursday, Friday, or Saturday, all in English, spelled exactly as shown.
calendar_start_day_of_week = 'Default'
#: East Asian language to use for transliteration
# Setting this tweak will make calibre use the specified language as the "base"
# language when transliterating East Asian languages to English. This might be
# useful if you run calibre in English but want text transliterated to
# Japanese instead of Chinese. The valid values are:
#
    'ja' for Japanese
#
    'kr' for Korean
    'vn' for Vietnamese
#
    'zh' for Chinese
# Any other value will use the language set in calibre preferences as the base
# language. A base language other than those in the above list causes transliteration
# with a base language of Chinese.
# Example: east_asian_base_language = 'ja'
east_asian_base_language = ''
```

# 12.4 Overriding icons, templates, et cetera

# Note

calibre has direct support for icon themes, there are several icon themes available for calibre, that you can use by going to *Preferences*  $\rightarrow$  *Interface*  $\rightarrow$  *Look* & *Feel*  $\rightarrow$  *Change icon theme*. It is preferable to use icon themes over overriding individual icons.

calibre allows you to override the static resources, like icons, JavaScript and templates for the metadata jacket, catalogs, etc. with customized versions that you like. All static resources are stored in the resources sub-folder of the calibre install location. On Windows, this is usually C:\Program Files\Calibre2\app\resources. On macOS, /Applications/calibre.app/Contents/Resources/resources/. On Linux, if you are using the binary installer from the calibre website it will be /opt/calibre/resources. These paths can change depending on where you choose to install calibre.

You should not change the files in this resources folder, as your changes will get overwritten the next time you update calibre. Instead, go to *Preferences*  $\rightarrow$  *Advanced*  $\rightarrow$  *Miscellaneous* and click *Open calibre configuration folder*. In this configuration folder, create a sub-folder called resources and place the files you want to override in it. Place the files in the appropriate sub folders, for example place images in resources/images, etc. calibre will automatically use your custom file in preference to the built-in one the next time it is started.

For example, if you wanted to change the icon for the *Remove books* action, you would first look in the built-in resources folder and see that the relevant file is resources/images/remove\_books.png. Assuming you have an alternate icon in PNG format called my\_remove\_books.png you would save it in the configuration folder as resources/images/remove\_books.png. All the icons used by the calibre user interface are in resources/images and its sub-folders. Placing an override file here will have even higher priority than a custom icon theme.

# 12.5 Creating your own icon theme for calibre

If you have created a beautiful set of icons and wish to share them with other calibre users via calibre's builtin icon theme support, you can easily package up your icons into a theme. To do so, go to *Preferences*  $\rightarrow$  *Miscellaneous*  $\rightarrow$  *Create icon theme*, select the folder where you have put your icons. Then fill up the theme metadata and click OK. This will result in a ZIP file containing the theme icons. You can upload that to the calibre forum at Mobileread<sup>119</sup> and then I will make your theme available via calibre's builtin icon theme system. By default, the icon theme you just created will also be installed as the current theme in calibre. If you are testing your theme, remember to remove the images from the resources/images folder so that the icons from the theme are used.

As of calibre 6, you can have custom icons for light and dark mode. Simply create two versions of the icon and name the files with the suffix -for-dark-theme and -for-light-theme. For example, modified-for-dark-theme.png and modified-for-light-theme.png. Then calibre will automatically use the appropriate icon based on the current theme.

# 12.6 Customizing calibre with plugins

calibre has a very modular design. Almost all functionality in calibre comes in the form of plugins. Plugins are used for conversion, for downloading news (though these are called recipes), for various components of the user interface, to connect to different devices, to process files when adding them to calibre and so on. You can get a complete list of all the built-in plugins in calibre by going to *Preferences*  $\rightarrow$  *Advanced*  $\rightarrow$  *Plugins*.

You can write your own plugins to customize and extend the behavior of calibre. The plugin architecture in calibre is very simple, see the tutorial *Writing your own plugins to extend calibre's functionality* (page 222).

Once you have written a plugin, you can upload that to the calibre plugins forum at Mobileread<sup>120</sup> and it will be made available via calibre's builtin plugin updater.

<sup>&</sup>lt;sup>119</sup> https://www.mobileread.com/forums/forumdisplay.php?f=166

<sup>&</sup>lt;sup>120</sup> https://www.mobileread.com/forums/forumdisplay.php?f=237

CHAPTER

# THIRTEEN

# **COMMAND LINE INTERFACE**

## kovid giskard ~/work/libprs500/src/libprs500/manual \$

# 1 Note

On macOS, the command line tools are inside the calibre bundle, for example, if you installed calibre in /Applications the command line tools are in /Applications/calibre.app/Contents/MacOS/. So, for example, to run ebook-convert you would use: /Applications/calibre.app/Contents/MacOS/ ebook-convert.

# **13.1 Documented commands**

# 13.1.1 calibre

calibre [options] [path\_to\_ebook or calibre url ...]

Launch the main **calibre** Graphical User Interface and optionally add the e-book at path\_to\_ebook to the database. You can also specify **calibre** URLs to perform various different actions, than just adding books. For example:

```
calibre://view-book/test_library/1842/epub
```

Will open the book with id 1842 in the EPUB format from the library "test\_library" in the **calibre** E-book viewer. Library names are the folder names of the libraries with spaces replaced by underscores. A full description of the various URL based actions is in the User Manual.

Whenever you pass arguments to **calibre** that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

## [options]

## --detach

Detach from the controlling terminal, if any (Linux only)

```
--help, -h
```

show this help message and exit

```
--ignore-plugins
```

Ignore custom plugins, useful if you installed a plugin that is preventing calibre from starting

```
--no-update-check
```

Do not check for updates

## --shutdown-running-calibre, -s

Cause a running calibre instance, if any, to be shutdown. Note that if there are running jobs, they will be silently aborted, so use with care.

## --start-in-tray

Start minimized to system tray.

## --verbose, -v

Ignored, do not use. Present only for legacy reasons

#### --version

show program's version number and exit

## --with-library

Use the library located at the specified path.

# 13.1.2 calibre-customize

calibre-customize options

Customize calibre by loading external plugins.

Whenever you pass arguments to **calibre-customize** that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

## [options]

#### --add-plugin, -a

Add a plugin by specifying the path to the ZIP file containing it.

## --build-plugin, -b

For plugin developers: Path to the folder where you are developing the plugin. This command will automatically zip up the plugin and update it in calibre.

## --customize-plugin

Customize plugin. Specify name of plugin and customization string separated by a comma. The customization string is the same as you would enter when customizing the plugin in the main calibre GUI.

## --disable-plugin

Disable the named plugin

## --enable-plugin

Enable the named plugin

## --help, -h

show this help message and exit

## --list-plugins, -l

List all installed plugins

#### --remove-plugin, -r

Remove a custom plugin by name. Has no effect on builtin plugins

## --version

show program 's version number and exit

# 13.1.3 calibre-debug

calibre-debug [options]

Various command line interfaces useful for debugging calibre. With no options, this command starts an embedded Python interpreter. You can also run the main calibre GUI, the calibre E-book viewer and the calibre editor in debug mode.

It also contains interfaces to various bits of calibre that do not have dedicated command line tools, such as font subsetting, the E-book diff tool and so on.

You can also use **calibre-debug** to run standalone scripts. To do that use it like this:

calibre-debug -e myscript.py -- --option1 --option2 file1 file2 ...

Everything after the -- is passed to the script. You can also use **calibre-debug** as a shebang in scripts, like this:

#!/usr/bin/env -S calibre-debug -e ---

Whenever you pass arguments to **calibre-debug** that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

## [options]

#### --add-simple-plugin

Add a simple plugin (i.e. a plugin that consists of only a .py file), by specifying the path to the py file containing the plugin code.

#### --command, -c

Run Python code.

## --debug-device-driver, -d

Debug device detection

### --default-programs

(Un)register calibre from Windows Default Programs. --*default-programs* (page 303) = (register)

#### --diff

Run the calibre diff tool. For example: calibre-debug --diff (page 303) file1 file2

## --edit-book

Launch the calibre "Edit book" tool in debug mode.

#### --exec-file, -e

Run the Python code in file.

```
--explode-book, -x
```

Explode the book into the specified folder. Usage: -x file.epub output\_dir Exports the book as a collection of HTML files and metadata, which you can edit using standard HTML editing tools. Works with EPUB, AZW3, HTMLZ and DOCX files.

#### --export-all-calibre-data

Export all calibre data (books/settings/plugins). Normally, you will be asked for the export folder and the libraries to export. You can also specify them as command line arguments to skip the questions. Use absolute paths for the export folder and libraries. The special keyword "all" can be used to export all libraries. Examples: calibre-debug --export-all-calibre-data (page 303) # for interactive use calibre-debug --export-all-calibre-data (page 303) /path/to/empty/export/folder /path/to/library/folder1 /path/to/library2 calibre-debug --export-all-calibre-data (page 303) /export/folder all # export all known libraries

## --gui, -g

Run the GUI with debugging enabled. Debug output is printed to stdout and stderr.

## --gui-debug

Run the GUI with a debug console, logging to the specified path. For internal use only, use the -g option to run the GUI in debug mode

## --help, -h

show this help message and exit

## --implode-book, -i

Implode a previously exploded book. Usage: -i output\_dir file.epub Imports the book from the files in output\_dir which must have been created by a previous call to --explode-book (page 303). Be sure to specify the same file type as was used when exploding.

## --import-calibre-data

Import previously exported calibre data

## --inspect-mobi, -m

Inspect the MOBI file(s) at the specified path(s)

## --kepubify

Convert the specified EPUB file to KEPUB without doing a full conversion. This is what the Kobo driver does when sending files to the device.

## --paths

Output the paths necessary to setup the calibre environment

## --run-plugin, -r

Run a plugin that provides a command line interface. For example: calibre-debug -r "Plugin name" -- file1 -- option1 Everything after the -- will be passed to the plugin as arguments.

## --run-test, -t

Run the named test(s). Use the special value "all" to run all tests. If the test name starts with a period it is assumed to be a module name. If the test name starts with @ it is assumed to be a category name.

## --run-without-debug

Don't run with the DEBUG flag set

## --shutdown-running-calibre, -s

Cause a running calibre instance, if any, to be shutdown. Note that if there are running jobs, they will be silently aborted, so use with care.

## --subset-font, -f

Subset the specified font. Use -- after this option to pass option to the font subsetting program.

## --test-build

Test binary modules in build

## --un-kepubify

Convert the specified KEPUB file to EPUB without doing a full conversion. This is what the Kobo driver does when importing files from the device.

#### --version

show program's version number and exit

#### --viewer, -w

Run the E-book viewer in debug mode

# 13.1.4 calibre-server

calibre-server [options] [path to library folder...]

Start the calibre Content server. The calibre Content server exposes your calibre libraries over the internet. You can specify the path to the library folders as arguments to **calibre-server**. If you do not specify any paths, all the libraries that the main calibre program knows about will be used.

Whenever you pass arguments to **calibre-server** that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

## [options]

#### --access-log

Path to the access log file. This log contains information about clients connecting to the server and making requests. By default no access logging is done.

#### --ajax-timeout

Time (in seconds) to wait for a response from the server when making queries.

## --auth-mode

Choose the type of authentication used. Set the HTTP authentication mode used by the server. Set to "basic" if you are putting this server behind an SSL proxy. Otherwise, leave it as "auto", which will use "basic" if SSL is configured otherwise it will use "digest".

#### --auto-reload

Automatically reload server when source code changes. Useful for development. You should also specify a small value for the shutdown timeout.

#### --ban-after

Number of login failures for ban. The number of login failures after which an IP address is banned

#### --ban-for

Ban IP addresses that have repeated login failures. Temporarily bans access for IP addresses that have repeated login failures for the specified number of minutes. Useful to prevent attempts at guessing passwords. If set to zero, no banning is done.

## --book-list-mode

Choose the default book list mode. Set the default book list mode that will be used for new users. Individual users can override the default in their own settings. The default is to use a cover grid.

## --compress-min-size

Minimum size for which responses use data compression (in bytes).

## --custom-list-template

Path to a JSON file containing a template for the custom book list mode. The easiest way to create such a template file is to go to Preferences-> Sharing over the net-> Book list template in calibre, create the template and export it.

## --daemonize

Run process in background as a daemon (Linux only).

#### --displayed-fields

Restrict displayed user-defined fields. Comma separated list of user-defined metadata fields that will be displayed by the Content server in the /opds and /mobile views. If you specify this option, any fields not in this list will not be displayed. For example: my\_rating,my\_tags

## --enable-allow-socket-preallocation, --disable-allow-socket-preallocation

Socket pre-allocation, for example, with systemd socket activation. By default, this option is enabled.

## --enable-auth, --disable-auth

Password based authentication to access the server. Normally, the server is unrestricted, allowing anyone to access it. You can restrict access to predefined users with this option. By default, this option is disabled.

## --enable-fallback-to-detected-interface, --disable-fallback-to-detected-interface

Fallback to auto-detected interface. If for some reason the server is unable to bind to the interface specified in the listen\_on option, then it will try to detect an interface that connects to the outside world and bind to that. By default, this option is enabled.

## --enable-local-write, --disable-local-write

Allow un-authenticated local connections to make changes. Normally, if you do not turn on authentication, the server operates in read-only mode, so as to not allow anonymous users to make changes to your calibre libraries. This option allows anybody connecting from the same computer as the server is running on to make changes. This is useful if you want to run the server without authentication but still use calibred to make changes to your calibre libraries. Note that turning on this option means any program running on the computer can make changes to your calibre libraries. By default, this option is disabled.

## --enable-log-not-found, --disable-log-not-found

Log HTTP 404 (Not Found) requests. Normally, the server logs all HTTP requests for resources that are not found. This can generate a lot of log spam, if your server is targeted by bots. Use this option to turn it off. By default, this option is enabled.

## --enable-use-bonjour, --disable-use-bonjour

Advertise OPDS feeds via BonJour. Advertise the OPDS feeds via the BonJour service, so that OPDS based reading apps can detect and connect to the server automatically. By default, this option is enabled.

## --enable-use-sendfile, --disable-use-sendfile

Zero copy file transfers for increased performance. This will use zero-copy in-kernel transfers when sending files over the network, increasing performance. However, it can cause corrupted file transfers on some broken filesystems. If you experience corrupted file transfers, turn it off. By default, this option is enabled.

## --help, -h

show this help message and exit

## --ignored-fields

Ignored user-defined metadata fields. Comma separated list of user-defined metadata fields that will not be displayed by the Content server in the /opds and /mobile views. For example: my\_rating,my\_tags

## --listen-on

The interface on which to listen for connections. The default is to listen on all available IPv6 and IPv4 interfaces. You can change this to, for example, "127.0.0.1" to only listen for IPv4 connections from the local machine, or to "0.0.0.0" to listen to all incoming IPv4 connections.

## --log

Path to log file for server log. This log contains server information and errors, not access logs. By default it is written to stdout.

#### --manage-users

Manage the database of users allowed to connect to this server. You can use it in automated mode by adding a –. See calibre-server *--manage-users* (page 306) –- help for details. See also the *--userdb* (page 308) option.

## --max-header-line-size

Max. size of single HTTP header (in KB).

#### --max-job-time

Maximum time for worker processes. Maximum amount of time worker processes are allowed to run (in minutes). Set to zero for no limit.

#### --max-jobs

Maximum number of worker processes. Worker processes are launched as needed and used for large jobs such as preparing a book for viewing, adding books, converting, etc. Normally, the max. number of such processes is based on the number of CPU cores. You can control it by this setting.

#### --max-log-size

Max. log file size (in MB). The maximum size of log files, generated by the server. When the log becomes larger than this size, it is automatically rotated. Set to zero to disable log rotation.

#### --max-opds-items

Maximum number of books in OPDS feeds. The maximum number of books that the server will return in a single OPDS acquisition feed.

#### --max-opds-ungrouped-items

Maximum number of ungrouped items in OPDS feeds. Group items in categories such as author/tags by first letter when there are more than this number of items. Set to zero to disable.

## --max-request-body-size

Max. allowed size for files uploaded to the server (in MB).

## --num-per-page

Number of books to show in a single page. The number of books to show in a single page in the browser.

#### --pidfile

Write process PID to the specified file

## --port

The port on which to listen for connections.

## --search-the-net-urls

Path to a JSON file containing URLs for the "Search the internet" feature. The easiest way to create such a file is to go to Preferences-> Sharing over the net->Search the internet in calibre, create the URLs and export them.

## --shutdown-timeout

Total time in seconds to wait for clean shutdown.

#### --ssl-certfile

Path to the SSL certificate file.

#### --ssl-keyfile

Path to the SSL private key file.

#### --timeout

Time (in seconds) after which an idle connection is closed.

### --trusted-ips

Allow un-authenticated connections from specific IP addresses to make changes. Normally, if you do not turn on authentication, the server operates in read-only mode, so as to not allow anonymous users to make changes to your calibre libraries. This option allows anybody connecting from the specified IP addresses to make changes. Must be a comma separated list of address or network specifications. This is useful if you want to run the server without authentication but still use calibred to make changes to your calibre libraries. Note that turning on this option means anyone connecting from the specified IP addresses can make changes to your calibre libraries.

## --url-prefix

A prefix to prepend to all URLs. Useful if you wish to run this server behind a reverse proxy. For example use, /calibre as the URL prefix.

#### --userdb

Path to the user database to use for authentication. The database is a SQLite file. To create it use *--manage-users* (page 306). You can read more about managing users at: https://manual.calibre-ebook.com/server.html# managing-user-accounts-from-the-command-line-only

#### --version

show program's version number and exit

### --worker-count

Number of worker threads used to process requests.

## 13.1.5 calibre-smtp

calibre-smtp [options] [from to text]

Send mail using the SMTP protocol. **calibre-smtp** has two modes of operation. In the compose mode you specify from to and text and these are used to build and send an email message. In the filter mode, **calibre-smtp** reads a complete email message from STDIN and sends it.

text is the body of the email message. If text is not specified, a complete email message is read from STDIN. from is the email address of the sender and to is the email address of the recipient. When a complete email is read from STDIN, from and to are only used in the SMTP negotiation, the message headers are not modified.

Whenever you pass arguments to **calibre-smtp** that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

## [options]

## --fork, -f

Fork and deliver message in background. If you use this option, you should also use *--outbox* (page 308) to handle delivery failures.

```
--help, -h
```

show this help message and exit

```
--localhost, -l
```

Host name of localhost. Used when connecting to SMTP server.

```
--outbox, -o
```

Path to maildir folder to store failed email messages in.

```
--timeout, -t
```

Timeout for connection

```
--verbose, -v
```

Be more verbose

```
--version
```

show program's version number and exit

## **COMPOSE MAIL**

Options to compose an email. Ignored if text is not specified

## --attachment, -a

File to attach to the email

## --subject, -s

Subject of the email

## **SMTP RELAY**

Options to use an SMTP relay server to send mail. calibre will try to send the email directly unless -relay is specified.

## --cafile

Path to a file of concatenated CA certificates in PEM format, used to verify the server certificate when using TLS. By default, the system CA certificates are used.

## --dont-verify-server-certificate

Do not verify the server certificate when connecting using TLS. This used to be the default behavior in calibre versions before 3.27. If you are using a relay with a self-signed or otherwise invalid certificate, you can use this option to restore the pre 3.27 behavior

## --encryption-method, -e

Encryption method to use when connecting to relay. Choices are TLS, SSL and NONE. Default is TLS. WARN-ING: Choosing NONE is highly insecure

## --password, -p

Password for relay

## --port

Port to connect to on relay server. Default is to use 465 if encryption method is SSL and 25 otherwise.

## --relay, -r

An SMTP relay server to use to send mail.

## --username, -u

Username for relay

# 13.1.6 calibredb

```
calibredb command [options] [arguments]
```

calibredb is the command line interface to the calibre database. It has several sub-commands, documented below.

**calibredb** can be used to manipulate either a calibre database specified by path or a calibre *Content server* running either on the local machine or over the internet. You can start a calibre *Content server* using either the **calibre-server** program or in the main calibre program click *Connect/share*  $\rightarrow$  *Start Content server*. Since **calibredb** can make changes to your calibre libraries, you must setup authentication on the server first. There are two ways to do that:

- If you plan to connect only to a server running on the same computer, you can simply use the --enable-local-write option of the Content server, to allow any program, including calibredb, running on the local computer to make changes to your calibre data. When running the server from the main calibre program, this option is in *Preferences* → *Sharing over the net* → *Advanced*.
- If you want to enable access over the internet, then you should setup user accounts on the server and use the --username (page 311) and --password (page 311) options to calibredb to give it access. You can setup user authentication for calibre-server by using the --enable-auth option and using --manage-users to

create the user accounts. If you are running the server from the main calibre program, use *Preferences*  $\rightarrow$  *Sharing over the net*  $\rightarrow$  *Require username/password*.

To connect to a running Content server, pass the URL of the server to the *--with-library* (page 310) option, see the documentation of that option for details and examples.

- Global Options (page 310)
- *list* (page 311)
- *add* (page 312)
  - Adding From Folders (page 313)
- remove (page 313)
- add\_format (page 313)
- *remove\_format* (page 314)
- show\_metadata (page 314)
- set\_metadata (page 314)
- export (page 314)
- catalog (page 316)
  - Epub Options (page 316)
- *saved\_searches* (page 317)
- *add\_custom\_column* (page 318)
- custom\_columns (page 318)
- remove\_custom\_column (page 318)
- set\_custom (page 318)
- restore\_database (page 319)
- check\_library (page 319)
- *list\_categories* (page 319)
- backup\_metadata (page 320)
- *clone* (page 320)
- embed\_metadata (page 320)
- search (page 321)
- *fts\_index* (page 321)
- *fts\_search* (page 322)

# **Global Options**

## --help, -h

show this help message and exit

## --library-path, --with-library

Path to the calibre library. Default is to use the path stored in the settings. You can also connect to a calibre Content

server to perform actions on remote libraries. To do so use a URL of the form: http://hostname:port/#library\_id for example, http://localhost:8080/#mylibrary. library\_id is the library id of the library you want to connect to on the Content server. You can use the special library\_id value of - to get a list of library ids available on the server. For details on how to setup access via a Content server, see https://manual.calibre-ebook.com/generated/ en/calibredb.html.

## --password

Password for connecting to a calibre Content server. To read the password from standard input, use the special value: <stdin>. To read the password from a file, use: <f:/path/to/file> (i.e. <f: followed by the full path to the file and a trailing >). The angle brackets in the above are required, remember to escape them or use quotes for your shell.

## --timeout

The timeout, in seconds, when connecting to a calibre library over the network. The default is two minutes.

## --username

Username for connecting to a calibre Content server

## --version

show program's version number and exit

## list

calibredb list [options]

List the books available in the calibre database.

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

## --ascending

Sort results in ascending order

## --fields, -f

The fields to display when listing books in the database. Should be a comma separated list of fields. Available fields: author\_sort, authors, comments, cover, formats, identifiers, isbn, languages, last\_modified, publate, publisher, rating, series, series\_index, size, tags, template, timestamp, title, uuid Default: title,authors. The special field "all" can be used to select all fields. In addition to the builtin fields above, custom fields are also available as \*field\_name, for example, for a custom field #rating, use the name: \*rating

## --for-machine

Generate output in JSON format, which is more suitable for machine parsing. Causes the line width and separator options to be ignored.

## --limit

The maximum number of results to display. Default: all

```
--line-width, -w
```

The maximum width of a single line in the output. Defaults to detecting screen size.

## --prefix

The prefix for all file paths. Default is the absolute path to the library folder.

### --search, -s

Filter the results by the search query. For the format of the search query, please see the search related documentation in the User Manual. Default is to do no filtering.

#### --separator

The string used to separate fields. Default is a space.

### --sort-by

The field by which to sort the results. You can specify multiple fields by separating them with commas. Available fields: author\_sort, authors, comments, cover, formats, identifiers, isbn, languages, last\_modified, pubdate, publisher, rating, series, series\_index, size, tags, template, timestamp, title, uuid Default: id

#### --template

The template to run if "template" is in the field list. Note that templates are ignored while connecting to a calibre server. Default: None

## --template\_file, -t

Path to a file containing the template to run if "template" is in the field list. Default: None

## --template\_heading

Heading for the template column. Default: template. This option is ignored if the option *--for-machine* (page 311) is set

## add

calibredb add [options] file1 file2 file3 ...

Add the specified files as books to the database. You can also specify folders, see the folder related options below.

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

## --authors, -a

Set the authors of the added book(s)

```
--automerge, -m
```

If books with similar titles and authors are found, merge the incoming formats (files) automatically into existing book records. A value of "ignore" means duplicate formats are discarded. A value of "overwrite" means duplicate formats in the library are overwritten with the newly added files. A value of "new\_record" means duplicate formats are placed into a new book record.

```
--cover, -c
```

Path to the cover to use for the added book

```
--duplicates, -d
```

Add books to database even if they already exist. Comparison is done based on book titles and authors. Note that the *--automerge* (page 312) option takes precedence.

```
--empty, -e
```

Add an empty book (a book with no formats)

```
--identifier, -I
```

Set the identifiers for this book, e.g. -I asin:XXX -I isbn:YYY

```
--isbn, -i
```

Set the ISBN of the added book(s)

```
--languages, -l
```

A comma separated list of languages (best to use ISO639 language codes, though some language names may also be recognized)

## --series, -s

Set the series of the added book(s)

## --series-index, -S

Set the series number of the added book(s)

## --tags, -T

Set the tags of the added book(s)

## --title, -t

Set the title of the added book(s)

## **Adding From Folders**

Options to control the adding of books from folders. By default only files that have extensions of known e-book file types are added.

--add

A filename (glob) pattern, files matching this pattern will be added when scanning folders for files, even if they are not of a known e-book file type. Can be specified multiple times for multiple patterns.

## --ignore

A filename (glob) pattern, files matching this pattern will be ignored when scanning folders for files. Can be specified multiple times for multiple patterns. For example: \*.pdf will ignore all PDF files

#### --one-book-per-directory, -1

Assume that each folder has only a single logical book and that all files in it are different e-book formats of that book

#### --recurse, -r

Process folders recursively

## remove

calibredb remove ids

Remove the books identified by ids from the database. ids should be a comma separated list of id numbers (you can get id numbers by using the search command). For example, 23,34,57-85 (when specifying a range, the last number in the range is not included).

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

#### --permanent

Do not use the Recycle Bin

## add format

calibredb add\_format [options] id ebook\_file

Add the e-book in ebook\_file to the available formats for the logical book identified by id. You can get id by using the search command. If the format already exists, it is replaced, unless the do not replace option is specified.

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

## --as-extra-data-file

Add the file as an extra data file to the book, not an ebook format

## --dont-replace

Do not replace the format if it already exists

## remove\_format

calibredb remove\_format [options] id fmt

Remove the format fmt from the logical book identified by id. You can get id by using the search command. fmt should be a file extension like LRF or TXT or EPUB. If the logical book does not have fmt available, do nothing.

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

## show\_metadata

calibredb show\_metadata [options] id

Show the metadata stored in the calibre database for the book identified by id. id is an id number from the search command.

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

#### --as-opf

Print metadata in OPF form (XML)

## set\_metadata

calibredb set\_metadata [options] book\_id [/path/to/metadata.opf]

Set the metadata stored in the calibre database for the book identified by book\_id from the OPF file metadata.opf. book\_id is a book id number from the search command. You can get a quick feel for the OPF format by using the –as-opf switch to the show\_metadata command. You can also set the metadata of individual fields with the –field option. If you use the –field option, there is no need to specify an OPF file.

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

```
--field, -f
```

The field to set. Format is field\_name:value, for example: *--field* (page 314) tags:tag1,tag2. Use *--list-fields* (page 314) to get a list of all field names. You can specify this option multiple times to set multiple fields. Note: For languages you must use the ISO639 language codes (e.g. en for English, fr for French and so on). For identifiers, the syntax is *--field* (page 314) identifiers:isbn:XXXX,doi:YYYYY. For boolean (yes/no) fields use true and false or yes and no.

--list-fields, -l

List the metadata field names that can be used with the *--field* (page 314) option

## export

calibredb export [options] ids

Export the books specified by ids (a comma separated list) to the filesystem. The **export** operation saves all formats of the book, its cover and metadata (in an OPF file). Any extra data files associated with the book are also saved. You can get id numbers from the search command.

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

--all

Export all books in database, ignoring the list of ids.

#### --dont-asciiize

Have calibre convert all non English characters into English equivalents for the file names. This is useful if saving to a legacy filesystem without full support for Unicode filenames. Specifying this switch will turn this behavior off.

#### --dont-save-cover

Normally, calibre will save the cover in a separate file along with the actual e-book files. Specifying this switch will turn this behavior off.

## --dont-save-extra-files

Save any data files associated with the book when saving the book Specifying this switch will turn this behavior off.

### --dont-update-metadata

Normally, calibre will update the metadata in the saved files from what is in the calibre library. Makes saving to disk slower. Specifying this switch will turn this behavior off.

## --dont-write-opf

Normally, calibre will write the metadata into a separate OPF file along with the actual e-book files. Specifying this switch will turn this behavior off.

## --formats

Comma separated list of formats to save for each book. By default all available formats are saved.

## --progress

Report progress

## --replace-whitespace

Replace whitespace with underscores.

## --single-dir

Export all books into a single folder

#### --template

The template to control the filename and folder structure of the saved files. Default is "{author\_sort}/{title}/{title} - {authors}" which will save books into a per-author subfolder with filenames containing title and author. Available controls are: {author\_sort, authors, id, isbn, languages, last\_modified, publate, publisher, rating, series, series\_index, tags, timestamp, title}

#### --timefmt

The format in which to display dates. %d - day, %b - month, %m - month number, %Y - year. Default is: %b, %Y

#### --to-dir

Export books to the specified folder. Default is .

#### --to-lowercase

Convert paths to lowercase.

## catalog

calibredb catalog /path/to/destination.(csv|epub|mobi|xml...) [options]

Export a **catalog** in format specified by path/to/destination extension. Options control how entries are displayed in the generated **catalog** output. Note that different **catalog** formats support different sets of options. To see the different options, specify the name of the output file and then the –help option.

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

## --ids, -i

Comma-separated list of database IDs to catalog. If declared, --search (page 316) is ignored. Default: all

## --search, -s

Filter the results by the search query. For the format of the search query, please see the search-related documentation in the User Manual. Default: no filtering

## --verbose, -v

Show detailed output information. Useful for debugging

## **Epub Options**

## --catalog-title

Title of generated catalog used as title in metadata. Default: 'My Books' Applies to: AZW3, EPUB, MOBI output formats

## --cross-reference-authors

Create cross-references in Authors section for books with multiple authors. Default: 'False' Applies to: AZW3, EPUB, MOBI output formats

## --debug-pipeline

Save the output from different stages of the conversion pipeline to the specified folder. Useful if you are unsure at which stage of the conversion process a bug is occurring. Default: 'None' Applies to: AZW3, EPUB, MOBI output formats

#### --exclude-genre

Regex describing tags to exclude as genres. Default: '[.+]|^+\$' excludes bracketed tags, e.g. '[Project Gutenberg]', and '+', the default tag for read books. Applies to: AZW3, EPUB, MOBI output formats

#### --exclusion-rules

Specifies the rules used to exclude books from the generated catalog. The model for an exclusion rule is either ('<rule name>','Tags','<comma-separated list of tags>') or ('<rule name>','<custom column>','<pattern>'). For example: (('Archived books','#status','Archived'),) will exclude a book with a value of 'Archived' in the custom column 'status'. When multiple rules are defined, all rules will be applied. Default: "(('Catalogs','Tags','Catalog'),)" Applies to: AZW3, EPUB, MOBI output formats

### --generate-authors

Include 'Authors' section in catalog. Default: 'False' Applies to: AZW3, EPUB, MOBI output formats

#### --generate-descriptions

Include 'Descriptions' section in catalog. Default: 'False' Applies to: AZW3, EPUB, MOBI output formats

#### --generate-genres

Include 'Genres' section in catalog. Default: 'False' Applies to: AZW3, EPUB, MOBI output formats

#### --generate-recently-added

Include 'Recently Added' section in catalog. Default: 'False' Applies to: AZW3, EPUB, MOBI output formats

#### --generate-series

Include 'Series' section in catalog. Default: 'False' Applies to: AZW3, EPUB, MOBI output formats

#### --generate-titles

Include 'Titles' section in catalog. Default: 'False' Applies to: AZW3, EPUB, MOBI output formats

#### --genre-source-field

Source field for 'Genres' section. Default: 'Tags' Applies to: AZW3, EPUB, MOBI output formats

#### --header-note-source-field

Custom field containing note text to insert in Description header. Default: ' ' Applies to: AZW3, EPUB, MOBI output formats

#### --merge-comments-rule

#<custom field>:[beforelafter]:[TruelFalse] specifying: <custom field> Custom field containing notes to merge with comments [beforelafter] Placement of notes with respect to comments [TruelFalse] - A horizontal rule is inserted between notes and comments Default: '::' Applies to: AZW3, EPUB, MOBI output formats

## --output-profile

Specifies the output profile. In some cases, an output profile is required to optimize the catalog for the device. For example, 'kindle' or 'kindle\_dx' creates a structured Table of Contents with Sections and Articles. Default: 'None' Applies to: AZW3, EPUB, MOBI output formats

#### --prefix-rules

Specifies the rules used to include prefixes indicating read books, wishlist items and other user-specified prefixes. The model for a prefix rule is ('<rule name>','<source field>','<pattern>','<prefix>'). When multiple rules are defined, the first matching rule will be used. Default: "(('Read books','tags','+',' $\checkmark$ '),('Wishlist item','tags','Wishlist',' $\times$ '))" Applies to: AZW3, EPUB, MOBI output formats

#### --preset

Use a named preset created with the GUI catalog builder. A preset specifies all settings for building a catalog. Default: 'None' Applies to: AZW3, EPUB, MOBI output formats

#### --thumb-width

Size hint (in inches) for book covers in catalog. Range: 1.0 - 2.0 Default: '1.0' Applies to: AZW3, EPUB, MOBI output formats

#### --use-existing-cover

Replace existing cover when generating the catalog. Default: 'False' Applies to: AZW3, EPUB, MOBI output formats

#### saved\_searches

calibredb saved\_searches [options] (list|add|remove)

Manage the saved searches stored in this database. If you try to add a query with a name that already exists, it will be replaced.

Syntax for adding:

calibredb **saved\_searches** add search\_name search\_expression

Syntax for removing:

calibredb saved\_searches remove search\_name

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

## add\_custom\_column

calibredb add\_custom\_column [options] label name datatype

Create a custom column. label is the machine friendly name of the column. Should not contain spaces or colons. name is the human friendly name of the column. datatype is one of: bool, comments, composite, datetime, enumeration, float, int, rating, series, text

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

## --display

A dictionary of options to customize how the data in this column will be interpreted. This is a JSON string. For enumeration columns, use *--display* (page 318)"{\"enum\_values\":[\"val1\", \"val2\"]}" There are many options that can go into the display variable. The options by column type are: composite: composite\_template, composite\_sort, make\_category,contains\_html, use\_decorations datetime: date\_format enumeration: enum\_values, enum\_colors, use\_decorations int, float: number\_format text: is\_names, use\_decorations The best way to find legal combinations is to create a custom column of the appropriate type in the GUI then look at the backup OPF for a book (ensure that a new OPF has been created since the column was added). You will see the JSON for the "display" for the new column in the OPF.

## --is-multiple

This column stores tag like data (i.e. multiple comma separated values). Only applies if datatype is text.

## custom\_columns

calibredb custom\_columns [options]

List available custom columns. Shows column labels and ids.

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

```
--details, -d
```

Show details for each column.

## remove\_custom\_column

calibredb remove\_custom\_column [options] label

Remove the custom column identified by label. You can see available columns with the custom\_columns command.

Whenever you pass arguments to calibred that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

--force, -f

Do not ask for confirmation

## set\_custom

calibredb set\_custom [options] column id value

Set the value of a custom column for the book identified by id. You can get a list of ids using the search command. You can get a list of custom column names using the custom\_columns command.

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

--append, -a

If the column stores multiple values, append the specified values to the existing ones, instead of replacing them.

## restore\_database

```
calibredb restore_database [options]
```

Restore this database from the metadata stored in OPF files in each folder of the calibre library. This is useful if your metadata.db file has been corrupted.

WARNING: This command completely regenerates your database. You will lose all saved searches, user categories, plugboards, stored per-book conversion settings, and custom recipes. Restored metadata will only be as accurate as what is found in the OPF files.

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

--really-do-it, -r

Really do the recovery. The command will not run unless this option is specified.

## check\_library

```
calibredb check_library [options]
```

Perform some checks on the filesystem representing a library. Reports are invalid\_titles, extra\_titles, invalid\_authors, extra\_authors, missing\_formats, extra\_formats, extra\_files, missing\_covers, extra\_covers, failed\_folders

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

--csv, -c

Output in CSV

--ignore\_extensions, -e

Comma-separated list of extensions to ignore. Default: all

```
--ignore_names, -n
```

Comma-separated list of names to ignore. Default: all

```
--report, -r
```

Comma-separated list of reports. Default: all

```
--vacuum-fts-db
```

Vacuum the full text search database. This can be very slow and memory intensive, depending on the size of the database.

## list\_categories

```
calibredb list_categories [options]
```

Produce a report of the category information in the database. The information is the equivalent of what is shown in the Tag browser.

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

```
--categories, -r
```

Comma-separated list of category lookup names. Default: all

```
--csv, -c
```

Output in CSV

```
--dialect
```

The type of CSV file to produce. Choices: excel, excel-tab, unix

```
--item_count, -i
```

Output only the number of items in a category instead of the counts per item within the category

## --width, -w

The maximum width of a single line in the output. Defaults to detecting screen size.

## backup\_metadata

calibredb backup\_metadata [options]

Backup the metadata stored in the database into individual OPF files in each books folder. This normally happens automatically, but you can run this command to force re-generation of the OPF files, with the –all option.

Note that there is normally no need to do this, as the OPF files are backed up automatically, every time metadata is changed.

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

--all

Normally, this command only operates on books that have out of date OPF files. This option makes it operate on all books.

## clone

```
calibredb clone path/to/new/library
```

Create a **clone** of the current library. This creates a new, empty library that has all the same custom columns, Virtual libraries and other settings as the current library.

The cloned library will contain no books. If you want to create a full duplicate, including all books, then simply use your filesystem tools to copy the library folder.

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

## embed\_metadata

```
calibredb embed_metadata [options] book_id
```

Update the metadata in the actual book files stored in the calibre library from the metadata in the calibre database. Normally, metadata is updated only when exporting files from calibre, this command is useful if you want the files to be updated in place. Note that different file formats support different amounts of metadata. You can use the special value

'all' for book\_id to update metadata in all books. You can also specify many book ids separated by spaces and id ranges separated by hyphens. For example: calibredb embed\_metadata 1 2 10-15 23

Whenever you pass arguments to calibred that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

```
--only-formats, -f
```

Only update metadata in files of the specified format. Specify it multiple times for multiple formats. By default, all formats are updated.

### search

```
calibredb search [options] search expression
```

Search the library for the specified **search** term, returning a comma separated list of book ids matching the **search** expression. The output format is useful to feed into other commands that accept a list of ids as input.

The **search** expression can be anything from calibre's powerful **search** query language, for example: calibredb **search** author:asimov 'title:"i robot"

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

# --limit, -l

The maximum number of results to return. Default is all results.

# fts\_index

calibredb fts\_index [options] enable/disable/status/reindex

Control the Full text search indexing process.

#### enable

Turns on FTS indexing for this library

#### disable

Turns off FTS indexing for this library

# status

Shows the current indexing status

# reindex

Can be used to re-index either particular books or the entire library. To re-index particular books specify the book ids as additional arguments after the reindex command. If no book ids are specified the entire library is re-indexed.

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

### --indexing-speed

The speed of indexing. Use fast for fast indexing using all your computers resources and slow for less resource intensive indexing. Note that the speed is reset to slow after every invocation.

#### --wait-for-completion

Wait till all books are indexed, showing indexing progress periodically

## fts\_search

calibredb fts\_search [options] search expression

Do a full text search on the entire library or a subset of it.

Whenever you pass arguments to calibredb that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

# --do-not-match-on-related-words

Only match on exact words not related words. So correction will not match correcting.

#### --include-snippets

Include snippets of the text surrounding each match. Note that this makes searching much slower.

#### --indexing-threshold

How much of the library must be indexed before searching is allowed, as a percentage. Defaults to 90

#### --match-end-marker

The marker used to indicate the end of a matched word inside a snippet

### --match-start-marker

The marker used to indicate the start of a matched word inside a snippet

### --output-format

The format to output the search results in. Either "text" for plain text or "json" for JSON output.

#### --restrict-to

Restrict the searched books, either using a search expression or ids. For example: ids:1,2,3 to restrict by ids or search:tag:foo to restrict to books having the tag foo.

# 13.1.7 ebook-convert

ebook-convert input\_file output\_file [options]

Convert an e-book from one format to another.

input\_file is the input and output\_file is the output. Both must be specified as the first two arguments to the command.

The output e-book format is guessed from the file extension of output\_file. output\_file can also be of the special format .EXT where EXT is the output file extension. In this case, the name of the output file is derived from the name of the input file. Note that the filenames must not start with a hyphen. Finally, if output\_file has no extension, then it is treated as a folder and an "open e-book" (OEB) consisting of HTML files is written to that folder. These files are the files that would normally have been passed to the output plugin.

After specifying the input and output file you can customize the conversion by specifying various options. The available options depend on the input and output file types. To get help on them specify the input and output file and then use the -h option.

For full documentation of the conversion system see *E-book conversion* (page 61)

Whenever you pass arguments to **ebook-convert** that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

The options and default values for the options change depending on both the input and output formats, so you should always check with:

ebook-convert myfile.input\_format myfile.output\_format -h

Below are the options that are common to all conversion, followed by the options specific to every input and output format.

- Look and Feel (page 325)
- Heuristic Processing (page 327)
- Search and Replace (page 328)
- *Structure Detection* (page 328)
- Table of Contents (page 329)
- Metadata (page 330)
- Debug (page 331)
- AZW4 Input Options (page 331)
- CHM Input Options (page 331)
- Comic Input Options (page 331)
- DJVU Input Options (page 332)
- DOCX Input Options (page 332)
- EPUB Input Options (page 333)
- FB2 Input Options (page 333)
- HTLZ Input Options (page 333)
- HTML Input Options (page 333)
- *LIT Input Options* (page 333)
- LRF Input Options (page 334)
- MOBI Input Options (page 334)
- ODT Input Options (page 334)
- PDB Input Options (page 334)
- PDF Input Options (page 334)
- PML Input Options (page 335)
- *RB Input Options* (page 335)
- RTF Input Options (page 335)
- Recipe Input Options (page 335)
- SNB Input Options (page 336)
- TCR Input Options (page 336)
- *TXT Input Options* (page 336)
- AZW3 Output Options (page 337)
- DOCX Output Options (page 337)
- EPUB Output Options (page 338)
- FB2 Output Options (page 339)

- HTML Output Options (page 339)
- HTMLZ Output Options (page 340)
- KEPUB Output Options (page 340)
- LIT Output Options (page 341)
- LRF Output Options (page 341)
- *MOBI Output Options* (page 342)
- OEB Output Options (page 343)
- PDB Output Options (page 343)
- PDF Output Options (page 343)
- PML Output Options (page 345)
- RB Output Options (page 345)
- RTF Output Options (page 345)
- *SNB Output Options* (page 345)
- TCR Output Options (page 346)
- *TXT Output Options* (page 346)
- TXTZ Output Options (page 347)

# --help, -h

show this help message and exit

### --input-profile

Specify the input profile. The input profile gives the conversion system information on how to interpret various information in the input document. For example resolution dependent lengths (i.e. lengths in pixels). Choices are: cybookg3, cybook\_opus, default, hanlinv3, hanlinv5, illiad, irexdr1000, irexdr800, kindle, msreader, mobipocket, nook, sony, sony300, sony900

# --list-recipes

List builtin recipe names. You can create an e-book from a builtin recipe like this: ebook-convert "Recipe Name.recipe" output.epub

# --output-profile

Specify the output profile. The output profile tells the conversion system how to optimize the created document for the specified device. In some cases, an output profile can be used to optimize the output for a particular device, but this is rarely necessary. Choices are:cybookg3, cybook\_opus, default, generic\_eink, generic\_eink\_hd, generic\_eink\_large, hanlinv3, hanlinv5, illiad, ipad, ipad3, irexdr1000, irexdr800, jetbook5, kindle, kindle\_dx, kindle\_fire, kindle\_oasis, kindle\_pw, kindle\_pw3, kindle\_scribe, kindle\_voyage, kobo, msreader, mobipocket, nook, nook\_color, nook\_hd\_plus, pocketbook\_inkpad3, pocketbook\_lux, pocketbook\_hd, pocketbook\_900, pocketbook\_pro\_912, galaxy, sony, sony300, sony900, sony-landscape, sonyt3, tablet

# --version

show program 's version number and exit

# Look and Feel

Options to control the look and feel of the output

### --asciiize

Transliterate Unicode characters to an ASCII representation. Use with care because this will replace Unicode characters with ASCII. For instance it will replace "Pelé" with "Pele". Also, note that in cases where there are multiple representations of a character (characters shared by Chinese and Japanese for instance) the representation based on the current calibre interface language will be used.

### --base-font-size

The base font size in pts. All font sizes in the produced book will be rescaled based on this size. By choosing a larger size you can make the fonts in the output bigger and vice versa. By default, when the value is zero, the base font size is chosen based on the output profile you chose.

# --change-justification

Change text justification. A value of "left" converts all justified text in the source to left aligned (i.e. unjustified) text. A value of "justify" converts all unjustified text to justified. A value of "original" (the default) does not change justification in the source file. Note that only some output formats support justification.

# --disable-font-rescaling

Disable all rescaling of font sizes.

# --embed-all-fonts

Embed every font that is referenced in the input document but not already embedded. This will search your system for the fonts, and if found, they will be embedded. Embedding will only work if the format you are converting to supports embedded fonts, such as EPUB, AZW3, DOCX or PDF. Please ensure that you have the proper license for embedding the fonts used in this document.

# --embed-font-family

Embed the specified font family into the book. This specifies the "base" font used for the book. If the input document specifies its own fonts, they may override this base font. You can use the filter style information option to remove fonts from the input document. Note that font embedding only works with some output formats, principally EPUB, AZW3 and DOCX.

### --expand-css

By default, calibre will use the shorthand form for various CSS properties such as margin, padding, border, etc. This option will cause it to use the full expanded form instead. Note that CSS is always expanded when generating EPUB files with the output profile set to one of the Nook profiles as the Nook cannot handle shorthand CSS.

### --extra-css

Either the path to a CSS stylesheet or raw CSS. This CSS will be appended to the style rules from the source file, so it can be used to override those rules.

# --filter-css

A comma separated list of CSS properties that will be removed from all CSS style rules. This is useful if the presence of some style information prevents it from being overridden on your device. For example: font-family,color,margin-left,margin-right

# --font-size-mapping

Mapping from CSS font names to font sizes in pts. An example setting is 12,12,14,16,18,20,22,24. These are the mappings for the sizes xx-small to xx-large, with the final size being for huge fonts. The font rescaling algorithm uses these sizes to intelligently rescale fonts. The default is to use a mapping based on the output profile you chose.

# --insert-blank-line

Insert a blank line between paragraphs. Will not work if the source file does not use paragraphs ( or <div> tags).

# --insert-blank-line-size

Set the height of the inserted blank lines (in em). The height of the lines between paragraphs will be twice the value set here.

### --keep-ligatures

Preserve ligatures present in the input document. A ligature is a combined character of a pair of characters like ff, fi, fl et cetera. Most readers do not have support for ligatures in their default fonts, so they are unlikely to render correctly. By default, calibre will turn a ligature into the corresponding pair of normal characters. Note that ligatures here mean only unicode ligatures not ligatures created via CSS or font styles. This option will preserve them instead.

# --line-height

The line height in pts. Controls spacing between consecutive lines of text. Only applies to elements that do not define their own line height. In most cases, the minimum line height option is more useful. By default no line height manipulation is performed.

### --linearize-tables

Some badly designed documents use tables to control the layout of text on the page. When converted these documents often have text that runs off the page and other artifacts. This option will extract the content from the tables and present it in a linear fashion.

#### --margin-bottom

Set the bottom margin in pts. Default is 5.0. Setting this to less than zero will cause no margin to be set (the margin setting in the original document will be preserved). Note: Page oriented formats such as PDF and DOCX have their own margin settings that take precedence.

#### --margin-left

Set the left margin in pts. Default is 5.0. Setting this to less than zero will cause no margin to be set (the margin setting in the original document will be preserved). Note: Page oriented formats such as PDF and DOCX have their own margin settings that take precedence.

#### --margin-right

Set the right margin in pts. Default is 5.0. Setting this to less than zero will cause no margin to be set (the margin setting in the original document will be preserved). Note: Page oriented formats such as PDF and DOCX have their own margin settings that take precedence.

### --margin-top

Set the top margin in pts. Default is 5.0. Setting this to less than zero will cause no margin to be set (the margin setting in the original document will be preserved). Note: Page oriented formats such as PDF and DOCX have their own margin settings that take precedence.

## --minimum-line-height

The minimum line height, as a percentage of the element's calculated font size. calibre will ensure that every element has a line height of at least this setting, irrespective of what the input document specifies. Set to zero to disable. Default is 120%. Use this setting in preference to the direct line height specification, unless you know what you are doing. For example, you can achieve "double spaced" text by setting this to 240.

#### --remove-paragraph-spacing

Remove spacing between paragraphs. Also sets an indent on paragraphs of 1.5em. Spacing removal will not work if the source file does not use paragraphs ( or <div> tags).

# --remove-paragraph-spacing-indent-size

When calibre removes blank lines between paragraphs, it automatically sets a paragraph indent, to ensure that paragraphs can be easily distinguished. This option controls the width of that indent (in em). If you set this value negative, then the indent specified in the input document is used, that is, calibre does not change the indentation.

#### --smarten-punctuation

Convert plain quotes, dashes and ellipsis to their typographically correct equivalents. For details, see https://daringfireball.net/projects/smartypants.

### --subset-embedded-fonts

Subset all embedded fonts. Every embedded font is reduced to contain only the glyphs used in this document. This decreases the size of the font files. Useful if you are embedding a particularly large font with lots of unused glyphs.

### --transform-css-rules

Path to a file containing rules to transform the CSS styles in this book. The easiest way to create such a file is to use the wizard for creating rules in the calibre GUI. Access it in the "Look & feel->Transform styles" section of the conversion dialog. Once you create the rules, you can use the "Export" button to save them to a file.

# --transform-html-rules

Path to a file containing rules to transform the HTML in this book. The easiest way to create such a file is to use the wizard for creating rules in the calibre GUI. Access it in the "Look & feel->Transform HTML" section of the conversion dialog. Once you create the rules, you can use the "Export" button to save them to a file.

# --unsmarten-punctuation

Convert fancy quotes, dashes and ellipsis to their plain equivalents.

# **Heuristic Processing**

Modify the document text and structure using common patterns. Disabled by default. Use –enable-heuristics to enable. Individual actions can be disabled with the –disable-\* options.

# --disable-dehyphenate

Analyze hyphenated words throughout the document. The document itself is used as a dictionary to determine whether hyphens should be retained or removed.

### --disable-delete-blank-paragraphs

Remove empty paragraphs from the document when they exist between every other paragraph

### --disable-fix-indents

Turn indentation created from multiple non-breaking space entities into CSS indents.

## --disable-format-scene-breaks

Left aligned scene break markers are center aligned. Replace soft scene breaks that use multiple blank lines with horizontal rules.

### --disable-italicize-common-cases

Look for common words and patterns that denote italics and italicize them.

# --disable-markup-chapter-headings

Detect unformatted chapter headings and sub headings. Change them to h2 and h3 tags. This setting will not create a TOC, but can be used in conjunction with structure detection to create one.

### --disable-renumber-headings

Looks for occurrences of sequential <h1> or <h2> tags. The tags are renumbered to prevent splitting in the middle of chapter headings.

# --disable-unwrap-lines

Unwrap lines using punctuation and other formatting clues.

# --enable-heuristics

Enable heuristic processing. This option must be set for any heuristic processing to take place.

# --html-unwrap-factor

Scale used to determine the length at which a line should be unwrapped. Valid values are a decimal between 0 and 1. The default is 0.4, just below the median line length. If only a few lines in the document require unwrapping this value should be reduced

### --replace-scene-breaks

Replace scene breaks with the specified text. By default, the text from the input document is used.

# Search and Replace

Modify the document text and structure using user defined patterns.

#### --search-replace

Path to a file containing search and replace regular expressions. The file must contain alternating lines of regular expression followed by replacement pattern (which can be an empty line). The regular expression must be in the Python regex syntax and the file must be UTF-8 encoded.

#### --sr1-replace

Replacement to replace the text found with sr1-search.

### --sr1-search

Search pattern (regular expression) to be replaced with sr1-replace.

### --sr2-replace

Replacement to replace the text found with sr2-search.

### --sr2-search

Search pattern (regular expression) to be replaced with sr2-replace.

#### --sr3-replace

Replacement to replace the text found with sr3-search.

#### --sr3-search

Search pattern (regular expression) to be replaced with sr3-replace.

# **Structure Detection**

Control auto-detection of document structure.

#### --add-alt-text-to-img

When an <img> tag has no alt attribute, check the associated image file for metadata that specifies alternate text, and use it to fill in the alt attribute. The alt attribute improves accessibility by providing text descriptions for users who cannot see or fully interpret visual content.

#### --chapter

An XPath expression to detect chapter titles. The default is to consider <h1> or <h2> tags that contain the words "chapter", "book", "section", "prologue", "epilogue" or "part" as chapter titles as well as any tags that have class="chapter". The expression used must evaluate to a list of elements. To disable chapter detection, use the expression "/". See the XPath Tutorial in the calibre User Manual for further help on using this feature.

### --chapter-mark

Specify how to mark detected chapters. A value of "pagebreak" will insert page breaks before chapters. A value of "rule" will insert a line before chapters. A value of "none" will disable chapter marking and a value of "both" will use both page breaks and lines to mark chapters.

### --disable-remove-fake-margins

Some documents specify page margins by specifying a left and right margin on each individual paragraph. calibre will try to detect and remove these margins. Sometimes, this can cause the removal of margins that should not have been removed. In this case you can disable the removal.

#### --insert-metadata

Insert the book metadata at the start of the book. This is useful if your e-book reader does not support displaying/searching metadata directly.

# --page-breaks-before

An XPath expression. Page breaks are inserted before the specified elements. To disable use the expression: /

### --prefer-metadata-cover

Use the cover detected from the source file in preference to the specified cover.

### --remove-first-image

Remove the first image from the input e-book. Useful if the input document has a cover image that is not identified as a cover. In this case, if you set a cover in calibre, the output document will end up with two cover images if you do not specify this option.

### --start-reading-at

An XPath expression to detect the location in the document at which to start reading. Some e-book reading programs (most prominently the Kindle) use this location as the position at which to open the book. See the XPath tutorial in the calibre User Manual for further help using this feature.

# Table of Contents

Control the automatic generation of a Table of Contents. By default, if the source file has a Table of Contents, it will be used in preference to the automatically generated one.

#### --duplicate-links-in-toc

When creating a TOC from links in the input document, allow duplicate entries, i.e. allow more than one entry with the same text, provided that they point to a different location.

# --level1-toc

XPath expression that specifies all tags that should be added to the Table of Contents at level one. If this is specified, it takes precedence over other forms of auto-detection. See the XPath Tutorial in the calibre User Manual for examples.

# --level2-toc

XPath expression that specifies all tags that should be added to the Table of Contents at level two. Each entry is added under the previous level one entry. See the XPath Tutorial in the calibre User Manual for examples.

# --level3-toc

XPath expression that specifies all tags that should be added to the Table of Contents at level three. Each entry is added under the previous level two entry. See the XPath Tutorial in the calibre User Manual for examples.

#### --max-toc-links

Maximum number of links to insert into the TOC. Set to 0 to disable. Default is: 50. Links are only added to the TOC if less than the threshold number of chapters were detected.

#### --no-chapters-in-toc

Don't add auto-detected chapters to the Table of Contents.

#### --toc-filter

Remove entries from the Table of Contents whose titles match the specified regular expression. Matching entries and all their children are removed.

## --toc-threshold

If fewer than this number of chapters is detected, then links are added to the Table of Contents. Default: 6

# --use-auto-toc

Normally, if the source file already has a Table of Contents, it is used in preference to the auto-generated one. With this option, the auto-generated one is always used.

# Metadata

Options to set metadata in the output

### --author-sort

String to be used when sorting by author.

### --authors

Set the authors. Multiple authors should be separated by ampersands.

#### --book-producer

Set the book producer.

#### --comments

Set the e-book description.

# --cover

Set the cover to the specified file or URL

#### --isbn

Set the ISBN of the book.

### --language

Set the language.

### --pubdate

Set the publication date (assumed to be in the local timezone, unless the timezone is explicitly specified)

# --publisher

Set the e-book publisher.

# --rating

Set the rating. Should be a number between 1 and 5.

# --read-metadata-from-opf, --from-opf, -m

Read metadata from the specified OPF file. Metadata read from this file will override any metadata in the source file.

# --series

Set the series this e-book belongs to.

# --series-index

Set the index of the book in this series.

# --tags

Set the tags for the book. Should be a comma separated list.

### --timestamp

Set the book timestamp (no longer used anywhere)

# --title

Set the title.

# --title-sort

The version of the title to be used for sorting.

# Debug

Options to help with debugging the conversion

# --debug-pipeline, -d

Save the output from different stages of the conversion pipeline to the specified folder. Useful if you are unsure at which stage of the conversion process a bug is occurring.

#### --verbose, -v

Level of verbosity. Specify multiple times for greater verbosity. Specifying it twice will result in full verbosity, once medium verbosity and zero times least verbosity.

# **AZW4 Input Options**

# --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

### **CHM Input Options**

# --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

# **Comic Input Options**

# --colors

Reduce the number of colors used in the image. This works only if you choose the PNG output format. It is useful to reduce file sizes. Set to zero to turn off. Maximum value is 256. It is off by default.

### --comic-image-size

Specify the image size as width x height pixels, for example: 123x321. Normally, an image size is automatically calculated from the output profile, this option overrides it.

#### --despeckle

Enable Despeckle. Reduces speckle noise. May greatly increase processing time.

### --disable-trim

Disable trimming of comic pages. For some comics, trimming might remove content as well as borders.

#### --dont-add-comic-pages-to-toc

When converting a CBC do not add links to each page to the TOC. Note this only applies if the TOC has more than one section

# --dont-grayscale

Do not convert the image to grayscale (black and white)

## --dont-normalize

Disable normalize (improve contrast) color range for pictures. Default: False

### --dont-sharpen

Disable sharpening.

# --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

# --keep-aspect-ratio

Maintain picture aspect ratio. Default is to fill the screen.

#### --landscape

Don't split landscape images into two portrait images

#### --no-process

Apply no processing to the image

#### --no-sort

Don't sort the files found in the comic alphabetically by name. Instead use the order they were added to the comic.

### --output-format

The format that images in the created e-book are converted to. You can experiment to see which format gives you optimal size and look on your device.

# --right2left

Used for right-to-left publications like manga. Causes landscape pages to be split into portrait pages from right to left.

## --wide

Keep aspect ratio and scale image using screen height as image width for viewing in landscape mode.

# **DJVU Input Options**

# --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

# **DOCX Input Options**

#### --docx-inline-subsup

Render superscripts and subscripts so that they do not affect the line height

# --docx-no-cover

Normally, if a large image is present at the start of the document that looks like a cover, it will be removed from the document and used as the cover for created e-book. This option turns off that behavior.

## --docx-no-pagebreaks-between-notes

Do not insert a page break after every endnote.

### --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

# **EPUB Input Options**

### --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

# **FB2 Input Options**

# --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

# --no-inline-fb2-toc

Do not insert a Table of Contents at the beginning of the book

# **HTLZ Input Options**

# --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

# **HTML Input Options**

# --allow-local-files-outside-root

Normally, resources linked to by the HTML file or its children will only be allowed if they are in a sub-folder of the original HTML file. This option allows including local files from any location on your computer. This can be a security risk if you are converting untrusted HTML and expecting to distribute the result of the conversion.

### --breadth-first

Traverse links in HTML files breadth first. Normally, they are traversed depth first.

### --dont-package

Normally this input plugin re-arranges all the input files into a standard folder hierarchy. Only use this option if you know what you are doing as it can result in various nasty side effects in the rest of the conversion pipeline.

# --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

### --max-levels

Maximum levels of recursion when following links in HTML files. Must be non-negative. 0 implies that no links in the root HTML file are followed. Default is 5.

# **LIT Input Options**

### --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

# **LRF Input Options**

# --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

# **MOBI Input Options**

# --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

# **ODT Input Options**

# --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

# **PDB Input Options**

### --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

# **PDF Input Options**

# --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

### --no-images

Do not extract images from the document

# --pdf-engine

The PDF engine to use, the "calibre" engine is recommended as it has automatic header and footer removal. Choices: calibre, pdftohtml

### --pdf-footer-regex

Regular expression to remove lines at the bottom of a page. This only looks at the last line of a page and works only with the calibre PDF engine.

# --pdf-footer-skip

Skip everything to the specified number of pixels at the bottom of a page. Negative numbers mean auto-detect and remove footers, zero means do not remove footers and positive numbers mean remove footers that appear below that many pixels from the bottom of the page. Works only with the calibre PDF engine.

# --pdf-header-regex

Regular expression to remove lines at the top of a page. This only looks at the first line of a page and works only with the calibre PDF engine.

# --pdf-header-skip

Skip everything to the specified number of pixels at the top of a page. Negative numbers mean auto-detect and remove headers, zero means do not remove headers and positive numbers mean remove headers that appear above that many pixels from the top of the page. Works only with the new PDF engine.

# --unwrap-factor

Scale used to determine the length at which a line should be unwrapped. Valid values are a decimal between 0 and 1. The default is 0.45, just below the median line length.

# **PML Input Options**

### --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

# **RB Input Options**

### --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

# **RTF Input Options**

#### --ignore-wmf

Ignore WMF images instead of replacing them with a placeholder image.

# --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

# **Recipe Input Options**

#### --dont-download-recipe

Do not download latest version of builtin recipes from the calibre server

# --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

### --lrf

Optimize fetching for subsequent conversion to LRF.

# --password

Password for sites that require a login to access content.

# --recipe-specific-option

Recipe specific options. Syntax is option\_name:value. For example: --recipe-specific-option (page 335) = date:2030-11-31. Can be specified multiple times to set different options. To see a list of all available options for a recipe, use --recipe-specific-option (page 335) = list.

#### --test

Useful for recipe development. Forces max\_articles\_per\_feed to 2 and downloads at most 2 feeds. You can change the number of feeds and articles by supplying optional arguments. For example: --test (page 335) 3 1 will download at most 3 feeds and only 1 article per feed.

### --username

Username for sites that require a login to access content.

# **SNB Input Options**

#### --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

# **TCR Input Options**

#### --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

# **TXT Input Options**

# --formatting-type

Formatting used within the document. \* auto: Automatically decide which formatting processor to use \* plain: No formatting \* heuristic: Use heuristics to determine chapter headings, italics, etc. \* textile: Use the Textile markup language \* markdown: Use the Markdown markup language To learn more about Markdown see https: //daringfireball.net/projects/markdown/

# --input-encoding

Specify the character encoding of the input document. If set this option will override any encoding declared by the document itself. Particularly useful for documents that do not declare an encoding or that have erroneous encoding declarations.

### --markdown-extensions

Enable extensions to Markdown syntax. Extensions are formatting that is not part of the standard Markdown format. The extensions enabled by default: footnotes, tables, toc. To learn more about Markdown extensions, see https://python-markdown.github.io/extensions/ This should be a comma separated list of extensions to enable: \* abbr: Abbreviations \* admonition: Support admonitions \* attr\_list: Add attribute to HTML tags \* codehilite: Add code highlighting via Pygments \* def\_list: Definition lists \* extra: Enables various common extensions \* fenced\_code: Alternative code block syntax \* footnotes: Footnotes \* legacy\_attrs: Use legacy element attributes \* legacy\_em: Use legacy underscore handling for connected words \* meta: Metadata in the document \* nl2br: Treat newlines as hard breaks \* sane\_lists: Do not allow mixing list types \* smarty: Use Markdown 's internal smartypants parser \* tables: Support tables \* toc: Generate a table of contents \* wikilinks: Wiki style links

### --paragraph-type

Paragraph structure to assume. The value of "off" is useful for formatted documents such as Markdown or Textile. Choices are: \* auto: Try to auto detect paragraph type \* block: Treat a blank line as a paragraph break \* single: Assume every line is a paragraph \* print: Assume every line starting with 2+ spaces or a tab starts a paragraph \* unformatted: Most lines have hard line breaks, few/no blank lines or indents \* off: Don't modify the paragraph structure

## --preserve-spaces

Normally extra spaces are condensed into a single space. With this option all spaces will be displayed.

### --txt-in-remove-indents

Normally extra space at the beginning of lines is retained. With this option they will be removed.

# **AZW3 Output Options**

#### --dont-compress

Disable compression of the file contents.

#### --extract-to

Extract the contents of the generated AZW3 file to the specified folder. The contents of the folder are first deleted, so be careful.

### --mobi-toc-at-start

When adding the Table of Contents to the book, add it at the start of the book instead of the end. Not recommended.

#### --no-inline-toc

Don't add Table of Contents to the book. Useful if the book has its own table of contents.

### --prefer-author-sort

When present, use author sort field as author.

### --pretty-print

If specified, the output plugin will try to create output that is as human readable as possible. May not have any effect for some output plugins.

#### --share-not-sync

Enable sharing of book content via Facebook etc. on the Kindle. WARNING: Using this feature means that the book will not auto sync its last read position on multiple devices. Complain to Amazon.

#### --toc-title

Title for any generated inline table of contents.

# **DOCX Output Options**

#### --docx-custom-page-size

Custom size of the document. Use the form width x height, for example: 123x321 to specify the width and height (in pts). This overrides any specified page-size.

# --docx-no-cover

Do not insert the book cover as an image at the start of the document. If you use this option, the book cover will be discarded.

# --docx-no-toc

Do not insert the table of contents as a page at the start of the document.

# --docx-page-margin-bottom

The size of the bottom page margin, in pts. Default is 72pt. Overrides the common bottom page margin setting, unless set to zero.

### --docx-page-margin-left

The size of the left page margin, in pts. Default is 72pt. Overrides the common left page margin setting.

# --docx-page-margin-right

The size of the right page margin, in pts. Default is 72pt. Overrides the common right page margin setting, unless set to zero.

#### --docx-page-margin-top

The size of the top page margin, in pts. Default is 72pt. Overrides the common top page margin setting, unless set to zero.

### --docx-page-size

The size of the page. Default is letter. Choices are ['a0', 'a1', 'a2', 'a3', 'a4', 'a5', 'a6', 'b0', 'b1', 'b2', 'b3', 'b4', 'b5', 'b6', 'legal', 'letter']

#### --extract-to

Extract the contents of the generated DOCX file to the specified folder. The contents of the folder are first deleted, so be careful.

### --preserve-cover-aspect-ratio

Preserve the aspect ratio of the cover image instead of stretching it out to cover the entire page.

# --pretty-print

If specified, the output plugin will try to create output that is as human readable as possible. May not have any effect for some output plugins.

# **EPUB** Output Options

### --dont-split-on-page-breaks

Turn off splitting at page breaks. Normally, input files are automatically split at every page break into two files. This gives an output e-book that can be parsed faster and with less resources. However, splitting is slow and if your source file contains a very large number of page breaks, you should turn off splitting on page breaks.

# --epub-flatten

This option is needed only if you intend to use the EPUB with FBReaderJ. It will flatten the file system inside the EPUB, putting all files into the top level.

#### --epub-inline-toc

Insert an inline Table of Contents that will appear as part of the main book content.

#### --epub-max-image-size

The maximum image size (width x height). A value of none means use the screen size from the output profile. A value of profile means no maximum size is specified. For example, a value of 100x200 will cause all images to be resized so that their width is no more than 100 pixels and their height is no more than 200 pixels. Note that this only affects the size of the actual image files themselves. Any given image may be rendered at a different size depending on the styling applied to it in the document.

## --epub-toc-at-end

Put the inserted inline Table of Contents at the end of the book instead of the start.

#### --epub-version

The version of the EPUB file to generate. EPUB 2 is the most widely compatible, only use EPUB 3 if you know you actually need it.

#### --extract-to

Extract the contents of the generated book to the specified folder. The contents of the folder are first deleted, so be careful.

### --flow-size

Split all HTML files larger than this size (in KB). This is necessary as most EPUB readers cannot handle large file sizes. The default of 260KB is the size required for Adobe Digital Editions. Set to 0 to disable size based splitting.

#### --no-default-epub-cover

Normally, if the input file has no cover and you don't specify one, a default cover is generated with the title, authors, etc. This option disables the generation of this cover.

#### --no-svg-cover

Do not use SVG for the book cover. Use this option if your EPUB is going to be used on a device that does not support SVG, like the iPhone or the JetBook Lite. Without this option, such devices will display the cover as a blank page.

# --preserve-cover-aspect-ratio

When using an SVG cover, this option will cause the cover to scale to cover the available screen area, but still preserve its aspect ratio (ratio of width to height). That means there may be white borders at the sides or top and bottom of the image, but the image will never be distorted. Without this option the image may be slightly distorted, but there will be no borders.

#### --pretty-print

If specified, the output plugin will try to create output that is as human readable as possible. May not have any effect for some output plugins.

#### --toc-title

Title for any generated inline table of contents.

# **FB2 Output Options**

### --fb2-genre

Genre for the book. Choices: sf\_history, sf\_action, sf\_epic, sf\_heroic, sf\_detective, sf\_cyberpunk, sf\_space, sf\_social, sf\_horror, sf\_humor, sf\_fantasy, sf, det\_classic, det\_police, det\_action, det\_irony, det\_history, det\_espionage, det\_crime, det\_political, det\_maniac, det\_hard, thriller, detective, prose\_classic, prose\_history, prose\_contemporary, prose\_counter, prose\_rus\_classic, prose\_su\_classics, love\_contemporary, love\_history, love\_detective, love\_short, love\_erotica, adv\_western, adv\_history, adv\_indian, adv\_maritime, adv\_geo, adv\_animal, adventure, child\_tale, child\_verse, child\_prose, child\_sf, child\_det, child\_adv, child\_education, children, poetry, dramaturgy, antique\_ant, antique\_european, antique\_russian, antique\_east, antique\_myths, antique, sci\_history, sci\_psychology, sci\_culture, sci\_religion, sci\_philosophy, sci\_politics, sci\_business, sci\_juris, sci\_linguistic, sci\_medicine, sci\_phys, sci\_math, sci\_chem, sci\_biology, sci\_tech, science, comp\_www, comp\_programming, comp\_hard, comp\_soft, comp\_db, comp\_osnet, computers, ref\_encyc, ref\_dict, ref\_ref, ref\_guide, reference, nonf\_biography, nonf\_publicism, nonf\_criticism, design, nonfiction, religion\_rel, religion\_esoterics, religion\_self, religion, humor\_anecdote, humor\_prose, humor\_verse, humor, home\_cooking, home\_pets, home\_crafts, home\_entertain, home\_health, home\_garden, home\_diy, home\_sport, home\_sex, home See: http://www.fictionbook.org/index.php/Eng:FictionBook\_2.1\_genres for a complete list with descriptions.

# --pretty-print

If specified, the output plugin will try to create output that is as human readable as possible. May not have any effect for some output plugins.

# --sectionize

Specify how sections are created: \* nothing: A single section \* files: Section per file \* toc: Section per entry in the ToC If ToC based generation fails, adjust the "Structure detection" and/or "Table of Contents" settings (turn on "Force use of auto-generated Table of Contents").

# **HTML Output Options**

### --extract-to

Extract the contents of the generated ZIP file to the specified folder. WARNING: The contents of the folder will be deleted.

### --pretty-print

If specified, the output plugin will try to create output that is as human readable as possible. May not have any effect for some output plugins.

#### --template-css

CSS file used for the output instead of the default file

# --template-html

Template used for the generation of the HTML contents of the book instead of the default file

### --template-html-index

Template used for generation of the HTML index file instead of the default file

# **HTMLZ Output Options**

### --htmlz-class-style

How to handle the CSS when using css-type = 'class'. Default is external. external: Use an external CSS file inline: Use a <style> tag in the HTML file

# --htmlz-css-type

Specify the handling of CSS. Default is class: Use CSS classes inline: Use the style attribute tag: Use HTML tags wherever possible

# --htmlz-title-filename

If set this option causes the file name of the HTML file inside the HTMLZ archive to be based on the book title.

# --pretty-print

If specified, the output plugin will try to create output that is as human readable as possible. May not have any effect for some output plugins.

# **KEPUB Output Options**

### --dont-split-on-page-breaks

Turn off splitting at page breaks. Normally, input files are automatically split at every page break into two files. This gives an output e-book that can be parsed faster and with less resources. However, splitting is slow and if your source file contains a very large number of page breaks, you should turn off splitting on page breaks.

#### --extract-to

Extract the contents of the generated book to the specified folder. The contents of the folder are first deleted, so be careful.

#### --flow-size

Split all HTML files larger than this size (in KB). This is necessary as some devices cannot handle large file sizes. Set to 0 to disable size based splitting.

# --kepub-affect-hyphenation

Modify how hyphenation is performed for this book. Note that hyphenation does not perform well for all languages, as it depends on the dictionaries present on the device, which are not always of the highest quality.

#### --kepub-disable-hyphenation

Override all hyphenation settings in book, forcefully disabling hyphenation completely.

#### --kepub-hyphenation-limit-lines

Maximum consecutive hyphenated lines.

## --kepub-hyphenation-min-chars

Minimum word length to hyphenate, in characters.

# --kepub-hyphenation-min-chars-after

Minimum characters after hyphens.

#### --kepub-hyphenation-min-chars-before

Minimum characters before hyphens.

#### --kepub-max-image-size

The maximum image size (width x height). A value of none means use the screen size from the output profile. A value of profile means no maximum size is specified. For example, a value of 100x200 will cause all images to be resized so that their width is no more than 100 pixels and their height is no more than 200 pixels. Note that this only affects the size of the actual image files themselves. Any given image may be rendered at a different size depending on the styling applied to it in the document.

# --kepub-prefer-justification

The KEPUB renderer on the Kobo has a bug when text justification is turned on. It will either not justify text properly or when highlighting there will be gaps between neighboring highlighted parts of text. By default, calibre generates KEPUB that avoid the highlighting gaps at the expense of worse text justification. This option reverses that tradeoff. Use this option if you use justification when reading on your Kobo device.

#### --pretty-print

If specified, the output plugin will try to create output that is as human readable as possible. May not have any effect for some output plugins.

# LIT Output Options

# --pretty-print

If specified, the output plugin will try to create output that is as human readable as possible. May not have any effect for some output plugins.

# **LRF Output Options**

# --enable-autorotation

Enable auto-rotation of images that are wider than the screen width.

### --header

Add a header to all the pages with title and author.

#### --header-format

Set the format of the header. %a is replaced by the author and %t by the title. Default is %t by %a

# --header-separation

Add extra spacing below the header. Default is 0 pt.

#### --minimum-indent

Minimum paragraph indent (the indent of the first line of a paragraph) in pts. Default: 0

#### --mono-family

The monospace family of fonts to embed

#### --pretty-print

If specified, the output plugin will try to create output that is as human readable as possible. May not have any effect for some output plugins.

## --render-tables-as-images

This option has no effect

# --sans-family

The sans-serif family of fonts to embed

### --serif-family

The serif family of fonts to embed

### --text-size-multiplier-for-rendered-tables

Multiply the size of text in rendered tables by this factor. Default is 1.0

### --wordspace

Set the space between words in pts. Default is 2.5

# **MOBI Output Options**

#### --dont-compress

Disable compression of the file contents.

#### --extract-to

Extract the contents of the generated MOBI file to the specified folder. The contents of the folder are first deleted, so be careful.

# --mobi-file-type

By default calibre generates MOBI files that contain the old MOBI 6 format. This format is compatible with all devices. However, by changing this setting, you can tell calibre to generate MOBI files that contain both MOBI 6 and the new KF8 format, or only the new KF8 format. KF8 has more features than MOBI 6, but only works with newer Kindles. Allowed values: old, both, new

#### --mobi-ignore-margins

Ignore margins in the input document. If False, then the MOBI output plugin will try to convert margins specified in the input document, otherwise it will ignore them.

# --mobi-keep-original-images

By default calibre converts all images to JPEG format in the output MOBI file. This is for maximum compatibility as some older MOBI viewers have problems with other image formats. This option tells calibre not to do this. Useful if your document contains lots of GIF/PNG images that become very large when converted to JPEG.

## --mobi-toc-at-start

When adding the Table of Contents to the book, add it at the start of the book instead of the end. Not recommended.

# --no-inline-toc

Don't add Table of Contents to the book. Useful if the book has its own table of contents.

### --personal-doc

Tag for MOBI files to be marked as personal documents. This option has no effect on the conversion. It is used only when sending MOBI files to a device. If the file being sent has the specified tag, it will be marked as a personal document when sent to the Kindle. Use the value \* to match all books.

# --prefer-author-sort

When present, use author sort field as author.

### --pretty-print

If specified, the output plugin will try to create output that is as human readable as possible. May not have any effect for some output plugins.

# --share-not-sync

Enable sharing of book content via Facebook etc. on the Kindle. WARNING: Using this feature means that the book will not auto sync its last read position on multiple devices. Complain to Amazon.

# --toc-title

Title for any generated inline table of contents.

# **OEB Output Options**

# --pretty-print

If specified, the output plugin will try to create output that is as human readable as possible. May not have any effect for some output plugins.

# **PDB Output Options**

# --format, -f

Format to use inside the PDB container. Choices are: doc, ereader, ztxt

# --inline-toc

Add Table of Contents to beginning of the book.

### --pdb-output-encoding

Specify the character encoding of the output document. The default is cp1252. Note: This option is not honored by all formats.

### --pretty-print

If specified, the output plugin will try to create output that is as human readable as possible. May not have any effect for some output plugins.

# **PDF Output Options**

#### --custom-size

Custom size of the document. Use the form width x height e.g. 123x321 to specify the width and height. This overrides any specified paper-size.

### --paper-size

The size of the paper. This size will be overridden when a non default output profile is used. Default is letter. Choices are a0, a1, a2, a3, a4, a5, a6, b0, b1, b2, b3, b4, b5, b6, legal, letter

### --pdf-add-toc

Add a Table of Contents at the end of the PDF that lists page numbers. Useful if you want to print out the PDF. If this PDF is intended for electronic use, use the PDF Outline instead.

# --pdf-default-font-size

The default font size (in pixels)

# --pdf-footer-template

An HTML template used to generate footers on every page. The strings \_PAGENUM\_, \_TITLE\_, \_AUTHOR\_ and \_SECTION\_ will be replaced by their current values.

# --pdf-header-template

An HTML template used to generate headers on every page. The strings \_PAGENUM\_, \_TITLE\_, \_AUTHOR\_ and \_SECTION\_ will be replaced by their current values.

# --pdf-hyphenate

Break long words at the end of lines. This can give the text at the right margin a more even appearance. Note that depending on the fonts used this option can break the copying of text from the PDF file.

## --pdf-mark-links

Surround all links with a red box, useful for debugging.

# --pdf-mono-family

The font family used to render monospace fonts. Will work only if the font is available system-wide.

#### --pdf-mono-font-size

The default font size for monospaced text (in pixels)

### --pdf-no-cover

Do not insert the book cover as an image at the start of the document. If you use this option, the book cover will be discarded.

#### --pdf-odd-even-offset

Shift the text horizontally by the specified offset (in pts). On odd numbered pages, it is shifted to the right and on even numbered pages to the left. Use negative numbers for the opposite effect. Note that this setting is ignored on pages where the margins are smaller than the specified offset. Shifting is done by setting the PDF CropBox, not all software respects the CropBox.

# --pdf-page-margin-bottom

The size of the bottom page margin, in pts. Default is 72pt. Overrides the common bottom page margin setting, unless set to zero.

## --pdf-page-margin-left

The size of the left page margin, in pts. Default is 72pt. Overrides the common left page margin setting.

# --pdf-page-margin-right

The size of the right page margin, in pts. Default is 72pt. Overrides the common right page margin setting, unless set to zero.

# --pdf-page-margin-top

The size of the top page margin, in pts. Default is 72pt. Overrides the common top page margin setting, unless set to zero.

#### --pdf-page-number-map

Adjust page numbers, as needed. Syntax is a JavaScript expression for the page number. For example, "if (n < 3) 0; else n - 3;", where n is current page number.

# --pdf-page-numbers

Add page numbers to the bottom of every page in the generated PDF file. If you specify a footer template, it will take precedence over this option.

### --pdf-sans-family

The font family used to render sans-serif fonts. Will work only if the font is available system-wide.

### --pdf-serif-family

The font family used to render serif fonts. Will work only if the font is available system-wide.

### --pdf-standard-font

The type of font family used to render font for which no font family is specified.

#### --pdf-use-document-margins

Use the page margins specified in the input document via @page CSS rules. This will cause the margins specified in the conversion settings to be ignored. If the document does not specify page margins, the conversion settings will be used as a fallback.

# --preserve-cover-aspect-ratio

Preserve the aspect ratio of the cover, instead of stretching it to fill the full first page of the generated PDF.

# --pretty-print

If specified, the output plugin will try to create output that is as human readable as possible. May not have any effect for some output plugins.

### --toc-title

Title for generated table of contents.

### --uncompressed-pdf

Generate an uncompressed PDF, useful for debugging.

### --unit, -u

The unit of measure for page sizes. Default is inch. Choices are millimeter, centimeter, point, inch, pica, didot, cicero, devicepixel Note: This does not override the unit for margins!

#### --use-profile-size

Instead of using the paper size specified in the PDF Output options, use a paper size corresponding to the current output profile. Useful if you want to generate a PDF for viewing on a specific device.

# **PML Output Options**

# --full-image-depth

Do not reduce the size or bit depth of images. Images have their size and depth reduced by default to accommodate applications that can not convert images on their own such as Dropbook.

# --inline-toc

Add Table of Contents to beginning of the book.

# --pml-output-encoding

Specify the character encoding of the output document. The default is cp1252.

#### --pretty-print

If specified, the output plugin will try to create output that is as human readable as possible. May not have any effect for some output plugins.

# **RB** Output Options

# --inline-toc

Add Table of Contents to beginning of the book.

# --pretty-print

If specified, the output plugin will try to create output that is as human readable as possible. May not have any effect for some output plugins.

# **RTF Output Options**

## --pretty-print

If specified, the output plugin will try to create output that is as human readable as possible. May not have any effect for some output plugins.

# **SNB Output Options**

#### --pretty-print

If specified, the output plugin will try to create output that is as human readable as possible. May not have any effect for some output plugins.

# --snb-dont-indent-first-line

Specify whether or not to insert two space characters to indent the first line of each paragraph.

### --snb-full-screen

Resize all the images for full screen mode.

### --snb-hide-chapter-name

Specify whether or not to hide the chapter title for each chapter. Useful for image-only output (eg. comics).

### --snb-insert-empty-line

Specify whether or not to insert an empty line between two paragraphs.

### --snb-max-line-length

The maximum number of characters per line. This splits on the first space before the specified value. If no space is found the line will be broken at the space after and will exceed the specified value. Also, there is a minimum of 25 characters. Use 0 to disable line splitting.

# --snb-output-encoding

Specify the character encoding of the output document. The default is utf-8.

# **TCR Output Options**

## --pretty-print

If specified, the output plugin will try to create output that is as human readable as possible. May not have any effect for some output plugins.

# --tcr-output-encoding

Specify the character encoding of the output document. The default is utf-8.

# **TXT Output Options**

#### --force-max-line-length

Force splitting on the max-line-length value when no space is present. Also allows max-line-length to be below the minimum

# --inline-toc

Add Table of Contents to beginning of the book.

# --keep-color

Do not remove font color from output. This is only useful when TXT output formatting is set to textile. Textile is the only formatting that supports setting font color. If this option is not specified font color will not be set and default to the color displayed by the reader (generally this is black).

#### --keep-image-references

Do not remove image references within the document. This is only useful when paired with a TXT output formatting option that is not none because links are always removed with plain text output.

# --keep-links

Do not remove links within the document. This is only useful when paired with a TXT output formatting option that is not none because links are always removed with plain text output.

#### --max-line-length

The maximum number of characters per line. This splits on the first space before the specified value. If no space is found the line will be broken at the space after and will exceed the specified value. Also, there is a minimum of 25 characters. Use 0 to disable line splitting.

# --newline, -n

Type of newline to use. Options are ['old\_mac', 'system', 'unix', 'windows']. Default is 'system'. Use 'old\_mac' for compatibility with Mac OS 9 and earlier. For macOS use 'unix'. 'system' will default to the newline type used by this OS.

# --pretty-print

If specified, the output plugin will try to create output that is as human readable as possible. May not have any effect for some output plugins.

## --txt-output-encoding

Specify the character encoding of the output document. The default is utf-8.

### --txt-output-formatting

Formatting used within the document. \* plain: Plain text \* markdown: Markdown formatted text \* textile: Textile formatted text

# **TXTZ Output Options**

### --force-max-line-length

Force splitting on the max-line-length value when no space is present. Also allows max-line-length to be below the minimum

# --inline-toc

Add Table of Contents to beginning of the book.

#### --keep-color

Do not remove font color from output. This is only useful when TXT output formatting is set to textile. Textile is the only formatting that supports setting font color. If this option is not specified font color will not be set and default to the color displayed by the reader (generally this is black).

# --keep-image-references

Do not remove image references within the document. This is only useful when paired with a TXT output formatting option that is not none because links are always removed with plain text output.

#### --keep-links

Do not remove links within the document. This is only useful when paired with a TXT output formatting option that is not none because links are always removed with plain text output.

### --max-line-length

The maximum number of characters per line. This splits on the first space before the specified value. If no space is found the line will be broken at the space after and will exceed the specified value. Also, there is a minimum of 25 characters. Use 0 to disable line splitting.

# --newline, -n

Type of newline to use. Options are ['old\_mac', 'system', 'unix', 'windows']. Default is 'system'. Use 'old\_mac' for compatibility with Mac OS 9 and earlier. For macOS use 'unix'. 'system' will default to the newline type used by this OS.

# --pretty-print

If specified, the output plugin will try to create output that is as human readable as possible. May not have any effect for some output plugins.

### --txt-output-encoding

Specify the character encoding of the output document. The default is utf-8.

### --txt-output-formatting

Formatting used within the document. \* plain: Plain text \* markdown: Markdown formatted text \* textile: Textile formatted text

# 13.1.8 ebook-edit

ebook-edit [opts] [path\_to\_ebook] [name\_of\_file\_inside\_book ...]

Launch the calibre Edit book tool. You can optionally also specify the names of files inside the book which will be opened for editing automatically.

Whenever you pass arguments to **ebook-edit** that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

# [options]

```
--detach
```

Detach from the controlling terminal, if any (Linux only)

```
--help, -h
```

show this help message and exit

# --select-text

The text to select in the book when it is opened for editing

### --version

show program's version number and exit

# 13.1.9 ebook-meta

```
ebook-meta ebook_file [options]
```

Read/Write metadata from/to e-book files.

Supported formats for reading metadata: azw, azw1, azw3, azw4, cb7, cbc, cbr, cbz, chm, docx, epub, fb2, fbz, html, htmlz, imp, kepub, lit, lrf, lrx, mobi, odt, oebzip, opf, pdb, pdf, pml, pmlz, pobi, prc, rar, rb, rtf, snb, tpz, txt, txtz, updb, zip

Supported formats for writing metadata: azw, azw1, azw3, azw4, docx, epub, fb2, fbz, htmlz, kepub, lrf, mobi, odt, pdb, pdf, prc, rtf, tpz, txtz

Different file types support different kinds of metadata. If you try to set some metadata on a file type that does not support it, the metadata will be silently ignored.

Whenever you pass arguments to **ebook-meta** that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

# [options]

# --author-sort

String to be used when sorting by author. If unspecified, and the author(s) are specified, it will be auto-generated from the author(s).

```
--authors, -a
```

Set the authors. Multiple authors should be separated by the & character. Author names should be in the order Firstname Lastname.

```
--book-producer, -k
```

Set the book producer.

--category

Set the book category.

```
--comments, -c
```

Set the e-book description.

# --cover

Set the cover to the specified file.

# --date, -d

Set the published date.

# --disallow-rendered-cover

For formats like EPUB that use a "default cover" of the first page rendered, disallow such default covers

# --from-opf

Read metadata from the specified OPF file and use it to set metadata in the e-book. Metadata specified on the command line will override metadata read from the OPF file

### --get-cover

Get the cover from the e-book and save it at as the specified file.

# --help, -h

show this help message and exit

# --identifier

Set the identifiers for the book, can be specified multiple times. For example: --identifier (page 349) uri:https://acme.com --identifier (page 349) isbn:12345 To remove an identifier, specify no value, --identifier (page 349) isbn: Note that for EPUB files, an identifier marked as the package identifier cannot be removed.

# --index, -i

Set the index of the book in this series.

### --isbn

Set the ISBN of the book.

```
--language, -l
```

Set the language.

```
--lrf-bookid
```

Set the BookID in LRF files

## --publisher, -p

Set the e-book publisher.

### --rating, -r

Set the rating. Should be a number between 1 and 5.

# --series, -s

Set the series this e-book belongs to.

### --tags

Set the tags for the book. Should be a comma separated list.

```
--title, -t
```

Set the title.

# --title-sort

The version of the title to be used for sorting. If unspecified, and the title is specified, it will be auto-generated from the title.

#### --to-opf

Specify the name of an OPF file. The metadata will be written to the OPF file.

#### --version

show program 's version number and exit

# 13.1.10 ebook-polish

ebook-polish [options] input\_file [output\_file]

Polishing books is all about putting the shine of perfection onto your carefully crafted e-books.

Polishing tries to minimize the changes to the internal code of your e-book. Unlike conversion, it does not flatten CSS, rename files, change font sizes, adjust margins, etc. Every action performs only the minimum set of changes needed for the desired effect.

You should use this tool as the last step in your e-book creation process.

Note that polishing only works on files in the AZW3 or EPUB or KEPUB formats.

Whenever you pass arguments to **ebook-polish** that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

# [options]

# --add-soft-hyphens, -H

Add soft hyphens to all words in the book. This allows the book to be rendered better when the text is justified, in readers that do not support hyphenation.

# --compress-images, -i

Losslessly compress images in the book, to reduce the filesize, without affecting image quality.

--cover, -c

Path to a cover image. Changes the cover specified in the e-book. If no cover is present, or the cover is not properly identified, inserts a new cover.

# --download-external-resources, -d

Download external resources such as images, stylesheets, etc. that point to URLs instead of files in the book. All such resources will be downloaded and added to the book so that the book no longer references any external resources.

--embed-fonts, -e

Embed all fonts that are referenced in the document and are not already embedded. This will scan your computer for the fonts, and if they are found, they will be embedded into the document. Please ensure that you have the proper license for embedding the fonts used in this document.

```
--help, -h
```

show this help message and exit

```
--jacket, -j
```

Insert a "book jacket" page at the start of the book that contains all the book metadata such as title, tags, authors, series, comments, etc. Any previous book jacket will be replaced.

### --opf, -o

Path to an OPF file. The metadata in the book is updated from the OPF file.

### --remove-jacket

Remove a previous inserted book jacket page.

# --remove-soft-hyphens

Remove soft hyphens from all text in the book.

#### --remove-unused-css, -u

Remove all unused CSS rules from stylesheets and <style> tags. Some books created from production templates can have a large number of extra CSS rules that don't match any actual content. These extra rules can slow down readers that need to parse them all.

# --smarten-punctuation, -p

Convert plain text dashes, ellipsis, quotes, multiple hyphens, etc. into their typographically correct equivalents. Note that the algorithm can sometimes generate incorrect results, especially when single quotes at the start of contractions are involved.

# --subset-fonts, -f

Subsetting fonts means reducing an embedded font to contain only the characters used from that font in the book. This greatly reduces the size of the font files (halving the font file sizes is common). For example, if the book uses a specific font for headers, then subsetting will reduce that font to contain only the characters present in the actual headers in the book. Or if the book embeds the bold and italic versions of a font, but bold and italic text is relatively rare, or absent altogether, then the bold and italic fonts can either be reduced to only a few characters or completely removed. The only downside to subsetting fonts is that if, at a later date you decide to add more text to your books, the newly added text might not be covered by the subset font.

### --upgrade-book, -U

Upgrade the internal structures of the book, if possible. For instance, upgrades EPUB 2 books to EPUB 3 books.

#### --verbose

Produce more verbose output, useful for debugging.

#### --version

show program's version number and exit

# 13.1.11 ebook-viewer

```
ebook-viewer [options] file
```

### View an e-book.

Whenever you pass arguments to **ebook-viewer** that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

# [options]

#### --continue

Continue reading the last opened book

# --detach

Detach from the controlling terminal, if any (Linux only)

# --force-reload

Force reload of all opened books

# --full-screen, --fullscreen, -f

If specified, the E-book viewer window will try to open full screen when started.

# --help, -h

show this help message and exit

#### --new-instance

Open a new viewer window even when the option to use only a single viewer window is set

### --open-at

The position at which to open the specified book. The position is a location or position you can get by using the Go to->Location action in the viewer controls. Alternately, you can use the form toc:something and it will open at the location of the first Table of Contents entry that contains the string "something". The form toc-href:something will match the href (internal link destination) of toc nodes. The matching is exact. If you want to match a substring, use the form toc-href-contains:something. The form ref:something will use Reference mode references. The form search:something will search for something after opening the book. The form regex:something will search for the regular expression something after opening the book.

### --raise-window

If specified, the E-book viewer window will try to come to the front when started.

#### --version

show program's version number and exit

# 13.1.12 fetch-ebook-metadata

fetch-ebook-metadata [options]

Fetch book metadata from online sources. You must specify at least one of title, authors or ISBN.

Whenever you pass arguments to **fetch-ebook-metadata** that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

# [options]

#### --allowed-plugin, -p

Specify the name of a metadata download plugin to use. By default, all metadata plugins will be used. Can be specified multiple times for multiple plugins. All plugin names: Google, Google Images, Amazon.com, Edelweiss, Open Library, Big Book Search

```
--authors, -a
```

Book author(s)

#### --cover, -c

Specify a filename. The cover, if available, will be saved to it. Without this option, no cover will be downloaded.

```
--help, -h
```

show this help message and exit

```
--identifier, -I
```

Identifiers such as ASIN/Goodreads id etc. Can be specified multiple times for multiple identifiers. For example: --identifier (page 352) asin:B0082BAJA0

### --isbn, -i

Book ISBN

# --opf, -o

Output the metadata in OPF format instead of human readable text.

# --timeout, -d

Timeout in seconds. Default is 30

```
--title, -t
Book title
```

```
--verbose, -v
```

Print the log to the console (stderr)

```
--version
```

show program's version number and exit

# 13.1.13 lrf2lrs

lrf2lrs book.lrf

Convert an LRF file into an LRS (XML UTF-8 encoded) file

Whenever you pass arguments to **lrf2lrs** that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

# [options]

# --dont-output-resources

Do not save embedded image and font files to disk

--help, -h

show this help message and exit

```
--output, -o
```

Output LRS file

--verbose

Be more verbose

--version

show program's version number and exit

# 13.1.14 lrfviewer

lrfviewer [options] book.lrf

Read the LRF e-book book.lrf

Whenever you pass arguments to **lrfviewer** that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

# [options]

```
--disable-hyphenation
```

Disable hyphenation. Should significantly speed up rendering.

--help, -h

show this help message and exit

--profile

Profile the LRF renderer

# --verbose

Print more information about the rendering process

### --version

show program's version number and exit

#### --visual-debug

Turn on visual aids to debugging the rendering engine

# --white-background

By default the background is off white as I find this easier on the eyes. Use this option to make the background pure white.

# 13.1.15 lrs2lrf

lrs2lrf [options] file.lrs

Compile an LRS file into an LRF file.

Whenever you pass arguments to **lrs2lrf** that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

# [options]

```
--help, -h
```

show this help message and exit

#### --lrs

Convert LRS to LRS, useful for debugging.

```
--output, -o
```

Path to output file

#### --verbose

Verbose processing

--version

show program 's version number and exit

# 13.1.16 web2disk

web2disk URL

Where URL is for example https://google.com

Whenever you pass arguments to **web2disk** that have spaces in them, enclose the arguments in quotation marks. For example: "/some path/with spaces"

# [options]

```
--base-dir, -d
```

Base folder into which URL is saved. Default is .

# --delay

Minimum interval in seconds between consecutive fetches. Default is 0 s

### --dont-download-stylesheets

Do not download CSS stylesheets.

# --encoding

The character encoding for the websites you are trying to download. The default is to try and guess the encoding.

# --filter-regexp

Any link that matches this regular expression will be ignored. This option can be specified multiple times, in which case as long as any regexp matches a link, it will be ignored. By default, no links are ignored. If both filter regexp and match regexp are specified, then filter regexp is applied first.

# --help, -h

show this help message and exit

# --match-regexp

Only links that match this regular expression will be followed. This option can be specified multiple times, in which case as long as a link matches any one regexp, it will be followed. By default all links are followed.

--max-files, -n

The maximum number of files to download. This only applies to files from <a href> tags. Default is 9223372036854775807

# --max-recursions, -r

Maximum number of levels to recurse i.e. depth of links to follow. Default 1

```
--timeout, -t
```

Timeout in seconds to wait for a response from the server. Default: 10.0 s

# --verbose

Show detailed output information. Useful for debugging

# --version

show program's version number and exit

# **13.2 Undocumented commands**

- · ebook-device
- markdown-calibre

You can see usage for undocumented commands by executing them without arguments in a terminal.

CHAPTER

FOURTEEN

# SETTING UP A CALIBRE DEVELOPMENT ENVIRONMENT

calibre is completely open source, licensed under the GNU GPL  $v3^{121}$ . This means that you are free to download and modify the program to your heart's content. In this section, you will learn how to get a calibre development environment set up on the operating system of your choice. calibre is written primarily in Python<sup>122</sup> with some C/C++ code for speed and system interfacing. Note that calibre requires at least Python 3.8.

#### Contents

- Design philosophy (page 358)
  - Code layout (page 358)
- *Getting the code* (page 359)
  - Submitting your changes to be included (page 359)
- Windows development environment (page 360)
- macOS development environment (page 361)
- *Linux development environment* (page 361)
- Having separate "normal" and "development" calibre installs on the same computer (page 362)
- *Debugging tips* (page 363)
  - Using print statements (page 363)
  - Using an interactive Python interpreter (page 363)
  - Using the Python debugger as a remote debugger (page 363)
  - Using the debugger in your favorite Python IDE (page 364)
  - Executing arbitrary scripts in the calibre Python environment (page 364)
- Using calibre in your projects (page 364)
  - Binary install of calibre (page 364)
  - Source install on Linux (page 364)
- API documentation for various parts of calibre (page 365)

<sup>121</sup> https://www.gnu.org/licenses/gpl.html

<sup>122</sup> https://www.python.org

# 14.1 Design philosophy

calibre has its roots in the Unix world, which means that its design is highly modular. The modules interact with each other via well defined interfaces. This makes adding new features and fixing bugs in calibre very easy, resulting in a frenetic pace of development. Because of its roots, calibre has a comprehensive command line interface for all its functions, documented in *Command Line Interface* (page 301).

The modular design of calibre is expressed via Plugins. There is a *tutorial* (page 253) on writing calibre plugins. For example, adding support for a new device to calibre typically involves writing less than a 100 lines of code in the form of a device driver plugin. You can browse the built-in drivers<sup>123</sup>. Similarly, adding support for new conversion formats involves writing input/output format plugins. Another example of the modular design is the *recipe system* (page 33) for fetching news. For more examples of plugins designed to add features to calibre, see the Index of plugins<sup>124</sup>.

# 14.1.1 Code layout

All the calibre Python code is in the calibre package. This package contains the following main sub-packages

- devices All the device drivers. Just look through some of the built-in drivers to get an idea for how they work.
  - For details, see: devices.interface which defines the interface supported by device drivers and devices.usbms which defines a generic driver that connects to a USBMS device. All USBMS based drivers in calibre inherit from it.
- e-books All the e-book conversion/metadata code. A good starting point is calibre.ebooks.conversion. cli which is the module powering the ebook-convert command. The conversion process is controlled via conversion.plumber. The format independent code is all in ebooks.oeb and the format dependent code is in ebooks.format\_name.
  - Metadata reading, writing, and downloading is all in ebooks.metadata
  - Conversion happens in a pipeline, for the structure of the pipeline, see *Introduction* (page 63). The pipeline consists of an input plugin, various transforms and an output plugin. The code that constructs and drives the pipeline is in plumber.py. The pipeline works on a representation of an e-book that is like an unzipped epub, with manifest, spine, toc, guide, html content, etc. The class that manages this representation is OEBBook in ebooks.oeb.base. The various transformations that are applied to the book during conversions live in oeb/transforms/\*.py. And the input and output plugins live in conversion/plugins/\*.py.
  - E-book editing happens using a different container object. It is documented in *API documentation for the e-book editing tools* (page 375).
- db The database back-end. See *API documentation for the database interface* (page 365) for the interface to the calibre library.
- Content server: srv is the calibre Content server.
- gui2 The Graphical User Interface. GUI initialization happens in gui2.main and gui2.ui. The e-book-viewer is in gui2.viewer. The e-book editor is in gui2.tweak\_book.

If you want to locate the entry points for all the various calibre executables, look at the  $entry_points$  structure in  $linux.py^{125}$ .

If you need help understanding the code, post in the development forum<sup>126</sup> and you will most likely get help from one of calibre's many developers.

<sup>&</sup>lt;sup>123</sup> https://github.com/kovidgoyal/calibre/tree/master/src/calibre/devices

<sup>&</sup>lt;sup>124</sup> https://www.mobileread.com/forums/showthread.php?p=1362767#post1362767

<sup>125</sup> https://github.com/kovidgoyal/calibre/blob/master/src/calibre/linux.py

<sup>&</sup>lt;sup>126</sup> https://www.mobileread.com/forums/forumdisplay.php?f=240

# 14.2 Getting the code

You can get the calibre source code in two ways, using a version control system or directly downloading a tarball<sup>127</sup>.

calibre uses Git<sup>128</sup>, a distributed version control system. Git is available on all the platforms calibre supports. After installing Git, you can get the calibre source code with the command:

git clone https://github.com/kovidgoyal/calibre.git

On Windows you will need the complete path name, that will be something like C:\Program Files\Git\git.exe.

calibre is a very large project with a very long source control history, so the above can take a while (10 mins to an hour depending on your internet speed).

If you want to get the code faster, the source code for the latest release is always available as an archive<sup>129</sup>.

To update a branch to the latest code, use the command:

git pull --no-edit

You can also browse the code at GitHub<sup>130</sup>.

# 14.2.1 Submitting your changes to be included

If you only plan to make a few small changes, you can make your changes and create a "merge directive" which you can then attach to a ticket in the calibre bug tracker<sup>131</sup>. To do this, make your changes, then run:

git commit -am "Comment describing your changes"
git format-patch origin/master --stdout > my-changes

This will create a my-changes file in the current folder, simply attach that to a ticket on the calibre bug tracker<sup>132</sup>. Note that this will include *all* the commits you have made. If you only want to send some commits, you have to change origin/master above. To send only the last commit, use:

git format-patch HEAD~1 --stdout > my-changes

To send the last *n* commits, replace 1 with *n*, for example, for the last 3 commits:

git format-patch HEAD~3 --stdout > my-changes

Be careful to not include merges when using HEAD~n.

If you plan to do a lot of development on calibre, then the best method is to create a GitHub<sup>133</sup> account. Below is a basic guide to setting up your own fork of calibre in a way that will allow you to submit pull requests for inclusion into the main calibre repository:

- Setup git on your machine as described in this article: Setup Git<sup>134</sup>
- Setup ssh keys for authentication to GitHub, as described here: Generating SSH keys<sup>135</sup>
- Go to https://github.com/kovidgoyal/calibre and click the Fork button.

129 https://calibre-ebook.com/dist/src

<sup>127</sup> https://calibre-ebook.com/dist/src

<sup>128</sup> https://www.git-scm.com/

<sup>130</sup> https://github.com/kovidgoyal/calibre

<sup>&</sup>lt;sup>131</sup> https://bugs.launchpad.net/calibre

<sup>132</sup> https://bugs.launchpad.net/calibre

<sup>133</sup> https://github.com

<sup>134</sup> https://help.github.com/articles/set-up-git

<sup>135</sup> https://help.github.com/articles/generating-ssh-keys

• In a Terminal do:

```
git clone git@github.com:<username>/calibre.git
git remote add upstream https://github.com/kovidgoyal/calibre.git
```

Replace <username> above with your GitHub username. That will get your fork checked out locally.

• You can make changes and commit them whenever you like. When you are ready to have your work merged, do a:

git push

and go to https://github.com/<username>/calibre and click the *Pull Request* button to generate a pull request that can be merged.

• You can update your local copy with code from the main repo at any time by doing:

git pull upstream

You should also keep an eye on the calibre development forum<sup>136</sup>. Before making major changes, you should discuss them in the forum or contact Kovid directly (his email address is all over the source code).

# 14.3 Windows development environment

1 Note

You must also get the calibre source code separately as described above.

Install calibre normally, using the Windows installer<sup>137</sup>. Then open a Command Prompt and change to the previously checked out calibre code folder. For example:

cd C:\Users\kovid\work\calibre

calibre is the folder that contains the src and resources sub-folders.

The next step is to set the environment variable CALIBRE\_DEVELOP\_FROM to the absolute path of the src folder. So, following the example above, it would be C:\Users\kovid\work\calibre\src. Here is a short guide<sup>138</sup> to setting environment variables on Windows.

Once you have set the environment variable, open a new command prompt and check that it was correctly set by using the command:

echo %CALIBRE\_DEVELOP\_FROM%

Setting this environment variable means that calibre will now load all its Python code from the specified location.

That's it! You are now ready to start hacking on the calibre code. For example, open the file src\calibre\\_\_init\_\_. py in your favorite editor and add the line:

print("Hello, world!")

near the top of the file. Now run the command calibredb. The very first line of output should be Hello, world!.

<sup>136</sup> https://www.mobileread.com/forums/forumdisplay.php?f=240

<sup>137</sup> https://calibre-ebook.com/download\_windows

<sup>138</sup> https://docs.python.org/using/windows.html#excursus-setting-environment-variables

You can also setup a calibre development environment inside the free Microsoft Visual Studio, if you like, following the instructions here<sup>139</sup>.

# 14.4 macOS development environment

### 1 Note

You must also get the calibre source code separately as described above.

Install calibre normally using the provided .dmg<sup>140</sup>. Then open a Terminal and change to the previously checked out calibre code folder, for example:

cd /Users/kovid/work/calibre

calibre is the folder that contains the src and resources sub-folders. The calibre command line tools are found inside the calibre app bundle, in /Applications/calibre.app/Contents/MacOS you should add this folder to your PATH environment variable, if you want to run the command line tools easily.

The next step is to create a bash script that will set the environment variable CALIBRE\_DEVELOP\_FROM to the absolute path of the src folder when running calibre in debug mode.

Create a plain text file:

```
#!/bin/sh
export CALIBRE_DEVELOP_FROM="/Users/kovid/work/calibre/src"
calibre-debug -g
```

Save this file as /usr/local/bin/calibre-develop, then set its permissions so that it can be executed:

chmod +x /usr/local/bin/calibre-develop

Once you have done this, run:

calibre-develop

You should see some diagnostic information in the Terminal window as calibre starts up, and you should see an asterisk after the version number in the GUI window, indicating that you are running from source.

# 14.5 Linux development environment

#### 1 Note

You must also get the calibre source code separately as described above.

calibre is primarily developed on Linux. You have two choices in setting up the development environment. You can install the calibre binary as normal and use that as a runtime environment to do your development. This approach is similar to that used in Windows and macOS. Alternatively, you can install calibre from source. Instructions for setting up a development environment from source are in the INSTALL file in the source tree. Here we will address using the binary as a runtime, which is the recommended method.

<sup>139</sup> https://www.mobileread.com/forums/showthread.php?t=251201

<sup>140</sup> https://calibre-ebook.com/download\_osx

Install calibre using the binary installer<sup>141</sup>. Then open a terminal and change to the previously checked out calibre code folder, for example:

cd /home/kovid/work/calibre

calibre is the folder that contains the src and resources sub-folders.

The next step is to set the environment variable CALIBRE\_DEVELOP\_FROM to the absolute path of the src folder. So, following the example above, it would be /home/kovid/work/calibre/src. How to set environment variables depends on your Linux distribution and what shell you are using.

#### 1 Note

It is recommended to use the binary installer provided from upstream. Should you insist on using a package provided by your distribution, use the CALIBRE\_PYTHON\_PATH and CALIBRE\_RESOURCES\_PATH variables instead. You can get these by running calibre-debug --paths. But note that distribution calibre packages are often terminally broken and are completely unsupported.

Once you have set the environment variable, open a new terminal and check that it was correctly set by using the command:

echo \$CALIBRE\_DEVELOP\_FROM

Setting this environment variable means that calibre will now load all its Python code from the specified location.

That's it! You are now ready to start hacking on the calibre code. For example, open the file src/calibre/\_\_init\_\_\_. py in your favorite editor and add the line:

print("Hello, world!")

near the top of the file. Now run the command calibredb. The very first line of output should be Hello, world!.

# 14.6 Having separate "normal" and "development" calibre installs on the same computer

The calibre source tree is very stable and rarely breaks, but if you feel the need to run from source on a separate test library and run the released calibre version with your everyday library, you can achieve this easily using .bat files or shell scripts to launch calibre. The example below shows how to do this on Windows using .bat files (the instructions for other platforms are the same, just use a shell script instead of a .bat file)

To launch the release version of calibre with your everyday library:

calibre-normal.bat:

calibre.exe "--with-library=C:\path\to\everyday\library folder"

calibre-dev.bat:

```
set CALIBRE_DEVELOP_FROM=C:\path\to\calibre\checkout\src
calibre.exe "--with-library=C:\path\to\test\library folder"
```

<sup>141</sup> https://calibre-ebook.com/download\_linux

# 14.7 Debugging tips

Python is a dynamically typed language with excellent facilities for introspection. Kovid wrote the core calibre code without once using a debugger. There are many strategies to debug calibre code:

# 14.7.1 Using print statements

This is Kovid's favorite way to debug. Simply insert print statements at points of interest and run your program in the terminal. For example, you can start the GUI from the terminal as:

```
calibre-debug -g
```

Similarly, you can start the E-book viewer as:

```
calibre-debug -w /path/to/file/to/be/viewed
```

The e-book editor can be started as:

```
calibre-debug --edit-book /path/to/be/edited
```

# 14.7.2 Using an interactive Python interpreter

You can insert the following two lines of code to start an interactive Python session at that point:

```
from calibre import ipython
ipython(locals())
```

When running from the command line, this will start an interactive Python interpreter with access to all locally defined variables (variables in the local scope). The interactive prompt even has Tab completion for object properties and you can use the various Python facilities for introspection, such as dir(), type(), repr(), etc.

# 14.7.3 Using the Python debugger as a remote debugger

You can use the builtin Python debugger (pdb) as a remote debugger from the command line. First, start the remote debugger at the point in the calibre code you are interested in, like this:

```
from calibre.rpdb import set_trace
set_trace()
```

Then run calibre, either as normal, or using one of the calibre-debug commands described in the previous section. Once the above point in the code is reached, calibre will freeze, waiting for the debugger to connect.

Now open a terminal or command prompt and use the following command to start the debugging session:

calibre-debug -c "from calibre.rpdb import cli; cli()"

You can read about how to use the Python debugger in the Python stdlib docs for the pdb module<sup>142</sup>.

### Note

By default, the remote debugger will try to connect on port 4444. You can change it, by passing the port parameter to both the set\_trace() and the cli() functions above, like this: set\_trace(port=1234) and cli(port=1234).

<sup>142</sup> https://docs.python.org/library/pdb.html#debugger-commands

#### \rm 1 Note

The Python debugger cannot handle multiple threads, so you have to call set\_trace once per thread, each time with a different port number.

# 14.7.4 Using the debugger in your favorite Python IDE

It is possible to use the builtin debugger in your favorite Python IDE, if it supports remote debugging. The first step is to add the calibre src checkout to the PYTHONPATH in your IDE. In other words, the folder you set as CALIBRE\_DEVELOP\_FROM above, must also be in the PYTHONPATH of your IDE.

Then place the IDE's remote debugger module into the src sub-folder of the calibre source code checkout. Add whatever code is needed to launch the remote debugger to calibre at the point of interest, for example in the main function. Then run calibre as normal. Your IDE should now be able to connect to the remote debugger running inside calibre.

# 14.7.5 Executing arbitrary scripts in the calibre Python environment

The **calibre-debug** command provides a couple of handy switches to execute your own code, with access to the calibre modules:

calibre-debug -c "some Python code"

is great for testing a little snippet of code on the command line. It works in the same way as the -c switch to the Python interpreter:

calibre-debug myscript.py

can be used to execute your own Python script. It works in the same way as passing the script to the Python interpreter, except that the calibre environment is fully initialized, so you can use all the calibre code in your script. To use command line arguments with your script, use the form:

calibre-debug myscript.py -- --option1 arg1

The -- causes all subsequent arguments to be passed to your script.

# 14.8 Using calibre in your projects

It is possible to directly use calibre functions/code in your Python project. Two ways exist to do this:

### 14.8.1 Binary install of calibre

If you have a binary install of calibre, you can use the Python interpreter bundled with calibre, like this:

```
calibre-debug /path/to/your/python/script.py -- arguments to your script
```

# 14.8.2 Source install on Linux

In addition to using the above technique, if you do a source install on Linux, you can also directly import calibre, as follows:

```
import init_calibre
import calibre
```

(continues on next page)

(continued from previous page)

```
print(calibre.__version__)
```

It is essential that you import the init\_calibre module before any other calibre modules/packages as it sets up the interpreter to run calibre code.

# 14.9 API documentation for various parts of calibre

# 14.9.1 API documentation for the database interface

This API is thread safe (it uses a multiple reader, single writer locking scheme). You can access this API like this:

```
from calibre.library import db
db = db('Path to calibre library folder').new_api
```

If you are in a calibre plugin that is part of the main calibre GUI, you get access to it like this instead:

```
db = self.gui.current_db.new_api
```

class calibre.db.cache.Cache (backend, library\_database\_instance=None)

An in-memory cache of the metadata.db file from a calibre library. This class also serves as a threadsafe API for accessing the database. The in-memory cache is maintained in normal form for maximum performance.

SQLITE is simply used as a way to read and write from metadata.db robustly. All table reading/sorting/searching/caching logic is re-implemented. This was necessary for maximum performance and flexibility.

#### class EventType (\*values)

```
book_created = 4
```

When a new book record is created in the database, with the book id as the only argument

#### book\_edited = 8

When a book format is edited, with arguments: (book\_id, fmt)

#### books\_removed = 5

When books are removed from the database with the list of book ids as the only argument

#### format\_added = 2

When a format is added to a book, with arguments: (book\_id, format)

#### formats\_removed = 3

When formats are removed from a book, with arguments: (mapping of book id to set of formats removed from the book)

#### indexing\_progress\_changed = 9

When the indexing progress changes

#### items\_removed = 7

When items such as tags or authors are removed from some books. Arguments: (field\_name, affected book ids, ids of removed items)

#### items\_renamed = 6

When items such as tags or authors are renamed in some or all books. Arguments: (field\_name, affected book ids, map of old item id to new item id)

#### $links_changed = 11$

When the links associated with items(s) are changed, with arguments: (field\_name, item\_ids)

#### metadata\_changed = 1

When some metadata is changed for some books, with arguments: (name of changed field, set of affected book ids)

#### $notes_changed = 10$

When the notes associated with item(s) are changed, with arguments: (field\_name, item\_ids)

Add the specified books to the library. Books should be an iterable of 2-tuples, each 2-tuple of the form (mi, format\_map) where mi is a Metadata object and format\_map is a dictionary of the form {fmt: path\_or\_stream}, for example: {'EPUB': '/path/to/file.epub'}.

Returns a pair of lists: ids, duplicates. ids contains the book ids for all newly created books in the database. duplicates contains the (mi, format\_map) for all books that already exist in the database as per the simple duplicate detection heuristic used by *has\_book()* (page 371).

#### add\_custom\_book\_data (name, val\_map, delete\_first=False)

Add data for name where val\_map is a map of book\_ids to values. If delete\_first is True, all previously stored data for name will be removed.

# add\_extra\_files (book\_id, map\_of\_relpath\_to\_stream\_or\_path, replace=True, auto\_rename=False) Add extra data files

#### add\_format (book\_id, fmt, stream\_or\_path, replace=True, run\_hooks=True, dbapi=None)

Add a format to the specified book. Return True if the format was added successfully.

#### **Parameters**

- **replace** If True replace existing format, otherwise if the format already exists, return False.
- **run\_hooks** If True, file type plugins are run on the format before and after being added.
- dbapi Internal use only.

#### add\_listener(event\_callback\_function, check\_already\_added=False)

Register a callback function that will be called after certain actions are taken on this database. The function must take three arguments: (*EventType* (page 365), library\_id, event\_type\_specific\_data)

#### add\_notes\_resource ( $path_or\_stream\_or\_data$ , name: str, mtime: float = None) $\rightarrow$ int

Add the specified resource so it can be referenced by notes and return its content hash

#### all\_annotation\_types()

Return a tuple of all annotation types in the database.

#### all\_annotation\_users()

Return a tuple of all (user\_type, user name) that have annotations.

#### 

Return a tuple of all annotations matching the specified criteria. *ignore\_removed* controls whether removed (deleted) annotations are also returned. Removed annotations are just a skeleton used for merging of annotations.

#### all\_annotations\_for\_book(book\_id)

Return a tuple containing all annotations for the specified book\_id as a dict with keys: *format*, *user\_type*, *user*, *annotation*. Here, annotation is the annotation data.

all\_book\_ids (type=<class 'frozenset'>)

Frozen set of all known book ids.

all\_field\_for (field, book\_ids, default\_value=None)

Same as field\_for, except that it operates on multiple books at once

all\_field\_ids (name)

Frozen set of ids for all values in the field name.

```
all_field_names (field)
```

Frozen set of all fields names (should only be used for many-one and many-many fields)

```
annotation_count_for_book(book_id)
```

Return the number of annotations for the specified book available in the database.

annotations\_map\_for\_book (book\_id, fmt, user\_type='local', user='viewer')

Return a map of annotation type -> annotation data for the specified book\_id, format, user and user\_type.

#### author\_data (author\_ids=None)

Return author data as a dictionary with keys: name, sort, link

If no authors with the specified ids are found an empty dictionary is returned. If author\_ids is None, data for all authors is returned.

#### author\_sort\_from\_authors (authors, key\_func=<function</pre>

make\_change\_case\_func.<locals>.change\_case>)

Given a list of authors, return the author\_sort string for the authors, preferring the author sort associated with the author over the computed string.

#### books\_for\_field(name, item\_id)

Return all the books associated with the item identified by item\_id, where the item belongs to the field name.

Returned value is a set of book ids, or the empty set if the item or the field does not exist.

```
books_in_virtual_library(vl, search_restriction=None, virtual_fields=None)
```

Return the set of books in the specified virtual library

#### compress\_covers (book\_ids, jpeg\_quality=100, progress\_callback=None)

Compress the cover images for the specified books. A compression quality of 100 will perform lossless compression, otherwise lossy compression.

The progress callback will be called with the book\_id and the old and new sizes for each book that has been processed. If an error occurs, the new size will be a string with the error details.

copy\_cover\_to(book\_id, dest, use\_hardlink=False, report\_file\_size=None)

Copy the cover to the file like object dest. Returns False if no cover exists or dest is the same file as the current cover. dest can also be a path in which case the cover is copied to it if and only if the path is different from the current path (taking case sensitivity into account).

copy\_format\_to(book\_id, fmt, dest, use\_hardlink=False, report\_file\_size=None)

Copy the format fmt to the file like object dest. If the specified format does not exist, raises NoSuchFormat error. dest can also be a path (to a file), in which case the format is copied to it, iff the path is different from the current path (taking case sensitivity into account).

**cover** (book\_id, as\_file=False, as\_image=False, as\_path=False, as\_pixmap=False)

Return the cover image or None. By default, returns the cover as a bytestring.

WARNING: Using as\_path will copy the cover to a temp file and return the path to the temp file. You should delete the temp file when you are done with it.

#### **Parameters**

- **as\_file** If True return the image as an open file object (a SpooledTemporaryFile)
- as\_image If True return the image as a QImage object
- as\_pixmap If True return the image as a QPixmap object
- as\_path If True return the image as a path pointing to a temporary file

#### data\_for\_find\_identical\_books()

Return data that can be used to implement *find\_identical\_books()* (page 369) in a worker process without access to the db. See db.utils for an implementation.

#### data\_for\_has\_book()

Return data suitable for use in *has\_book()* (page 371). This can be used for an implementation of *has\_book()* (page 371) in a worker process without access to the db.

#### delete\_annotations (annot\_ids)

Delete annotations with the specified ids.

```
delete_custom_book_data(name, book_ids=())
```

Delete data for name. By default deletes all data, if you only want to delete data for some book ids, pass in a list of book ids.

#### delete\_trash\_entry(book\_id, category)

Delete an entry from the trash. Here category is 'b' for books and 'f' for formats.

embed\_metadata (book\_ids, only\_fmts=None, report\_error=None, report\_progress=None)

Update metadata in all formats of the specified book\_ids to current metadata in the database.

#### expire\_old\_trash()

Expire entries from the trash that are too old

#### **export\_note** (*field*, *item\_id*) $\rightarrow$ str

Export the note as a single HTML document with embedded images as data: URLs

#### fast\_field\_for (field\_obj, book\_id, default\_value=None)

Same as field\_for, except that it avoids the extra lookup to get the field object

#### field\_for (name, book\_id, default\_value=None)

Return the value of the field name for the book identified by book\_id. If no such book exists or it has no defined value for the field name or no such field exists, then default\_value is returned.

default\_value is not used for title, title\_sort, authors, author\_sort and series\_index. This is because these always have values in the db. default\_value is used for all custom columns.

The returned value for is\_multiple fields are always tuples, even when no values are found (in other words, default\_value is ignored). The exception is identifiers for which the returned value is always a dictionary. The returned tuples are always in link order, that is, the order in which they were created.

#### field\_ids\_for(name, book\_id)

Return the ids (as a tuple) for the values that the field name has on the book identified by book\_id. If there are no values, or no such book, or no such field, an empty tuple is returned.

#### $field\_supports\_notes(field=None) \rightarrow bool$

Return True iff the specified field supports notes. If field is None return frozenset of all fields that support notes.

#### find\_identical\_books (mi, search\_restriction=", book\_ids=None)

Finds books that have a superset of the authors in mi and the same title (title is fuzzy matched). See also data\_for\_find\_identical\_books() (page 368).

#### format (book\_id, fmt, as\_file=False, as\_path=False, preserve\_filename=False)

Return the e-book format as a bytestring or *None* if the format doesn't exist, or we don't have permission to write to the e-book file.

#### Parameters

- **as\_file** If True the e-book format is returned as a file object. Note that the file object is a SpooledTemporaryFile, so if what you want to do is copy the format to another file, use *copy\_format\_to()* (page 367) instead for performance.
- as\_path Copies the format file to a temp file and returns the path to the temp file
- **preserve\_filename** If True and returning a path the filename is the same as that used in the library. Note that using this means that repeated calls yield the same temp file (which is re-created each time)

#### format\_abspath(book\_id, fmt)

Return absolute path to the e-book file of format *format*. You should almost never use this, as it breaks the threadsafe promise of this API. Instead use, *copy\_format\_to()* (page 367).

Currently used only in calibredb list, the viewer, edit book, compare\_format to original format, open with, bulk metadata edit and the catalogs (via get\_data\_as\_dict()).

Apart from the viewer, open with and edit book, I don't believe any of the others do any file write I/O with the results of this call.

#### format\_hash(book\_id, fmt)

Return the hash of the specified format for the specified book. The kind of hash is backend dependent, but is usually SHA-256.

#### format\_metadata(book\_id, fmt, allow\_cache=True, update\_db=False)

Return the path, size and mtime for the specified format for the specified book. You should not use path unless you absolutely have to, since accessing it directly breaks the threadsafe guarantees of this API. Instead use the  $copy\_format\_to()$  (page 367) method.

#### Parameters

- allow\_cache If True cached values are used, otherwise a slow filesystem access is done. The cache values could be out of date if access was performed to the filesystem outside of this API.
- update\_db If True The max\_size field of the database is updated for this book.

#### formats (book\_id, verify\_formats=True)

Return tuple of all formats for the specified book. If verify\_formats is True, verifies that the files exist on disk.

#### get\_all\_items\_that\_have\_notes (*field\_name=None*) $\rightarrow$ set[int] | dict[str, set[int]]

Return all item\_ids for items that have notes in the specified field or all fields if field\_name is None

#### get\_all\_link\_maps\_for\_book(book\_id)

Returns all links for all fields referenced by book identified by book\_id. If book\_id doesn't exist then the method returns {}.

Example: Assume author A has link X, author B has link Y, tag S has link F, and tag T has link G. If book 1 has author A and tag T, this method returns {'authors':{'A':'X'}, 'tags':{'T', 'G'}. If book 2's author is neither A nor B and has no tags, this method returns {}.

#### Parameters

**book\_id** – the book id in question.

#### Returns

{field: {field\_value, link\_value}, ... for all fields with a field\_value having a non-empty link value for that book

#### 

Used internally to implement the Tag Browser

#### get\_custom\_book\_data (name, book\_ids=(), default=None)

Get data for name. By default returns data for all book\_ids, pass in a list of book ids if you only want some data. Returns a map of book\_id to values. If a particular value could not be decoded, uses default for it.

get\_id\_map(field)

Return a mapping of id numbers to values for the specified field. The field must be a many-one or many-many field, otherwise a ValueError is raised.

### get\_ids\_for\_custom\_book\_data(name)

Return the set of book ids for which name has data.

#### get\_item\_id (field, item\_name, case\_sensitive=False)

Return the item id for item\_name or None if not found. This function is very slow if doing lookups for multiple names use either get\_item\_ids() or get\_item\_name\_map(). Similarly, case sensitive lookups are faster than case insensitive ones.

#### get\_item\_ids (field, item\_names, case\_sensitive=False)

Return a dict mapping item\_name to the item id or None

#### get\_item\_name (field, item\_id)

Return the item name for the item specified by item\_id in the specified field. See also get\_id\_map() (page 370).

#### get\_item\_name\_map (field, normalize\_func=None)

Return mapping of item values to ids

#### get\_link\_map (for\_field)

Return a dictionary of links for the supplied field.

#### Parameters

for\_field - the lookup name of the field for which the link map is desired

#### Returns

{field\_value:link\_value, ...} for non-empty links

get\_metadata (book\_id, get\_cover=False, get\_user\_categories=True, cover\_as\_data=False)

Return metadata for the book identified by book\_id as a *calibre.ebooks.metadata.book.base. Metadata* (page 208) object. Note that the list of formats is not verified. If get\_cover is True, the cover is returned, either a path to temp file as mi.cover or if cover\_as\_data is True then as mi.cover\_data.

#### get\_next\_series\_num\_for (series, field='series', current\_indices=False)

Return the next series index for the specified series, taking into account the various preferences that control next series number generation.

#### Parameters

- field The series-like field (defaults to the builtin series column)
- current\_indices If True, returns a mapping of book\_id to current series\_index value instead.

#### $get_notes_resource(resource\_hash) \rightarrow dict | None$

Return a dict containing the resource data and name or None if no resource with the specified hash is found

#### get\_proxy\_metadata(book\_id)

Like get\_metadata() (page 370) except that it returns a ProxyMetadata object that only reads values from the database on demand. This is much faster than get\_metadata when only a small number of fields need to be accessed from the returned metadata object.

#### get\_usage\_count\_by\_id(field)

Return a mapping of id to usage count for all values of the specified field, which must be a many-one or many-many field.

```
has_book(mi)
```

Return True iff the database contains an entry with the same title as the passed in Metadata object. The comparison is case-insensitive. See also *data\_for\_has\_book()* (page 368).

### has\_format (book\_id, fmt)

Return True iff the format exists on disk

has\_id(book\_id)

Return True iff the specified book\_id exists in the db

import\_note (field, item\_id, path\_to\_html\_file, path\_is\_data=False)

Import a previously exported note or an arbitrary HTML file as the note for the specified item

#### init()

Initialize this cache with data from the backend.

#### items\_with\_notes\_in\_book ( $book_id: int$ ) $\rightarrow dict[str, dict[int, str]]$

Return a dict of field to items that have associated notes for that field for the specified book

#### link\_for (field, item\_id)

Return the link, if any, for the specified item or None if no link is found

**list\_extra\_files** (*book\_id*, *use\_cache=False*, *pattern="*)  $\rightarrow$  tuple[ExtraFile, ...]

Get information about extra files in the book's directory.

#### Parameters

- book\_id the database book id for the book
- **pattern** the pattern of filenames to search for. Empty pattern matches all extra files. Patterns must use / as separator. Use the DATA\_FILE\_PATTERN constant to match files inside the data directory.

#### Returns

A tuple of all extra files matching the specified pattern. Each element of the tuple is ExtraFile(relpath, file\_path, stat\_result). Where relpath is the relative path of the file to the book directory using / as a separator. stat\_result is the result of calling os.stat() on the file. merge\_annotations\_for\_book (book\_id, fmt, annots\_list, user\_type='local', user='viewer')
Merge the specified annotations into the existing annotations for book\_id, fm, user\_type, and user.

#### merge\_extra\_files (dest\_id, src\_ids, replace=False)

Merge the extra files from src\_ids into dest\_id. Conflicting files are auto-renamed unless replace=True in which case they are replaced.

#### move\_book\_from\_trash(book\_id)

Undelete a book from the trash directory

#### move\_format\_from\_trash(book\_id, fmt)

Undelete a format from the trash directory

#### multisort (fields, ids\_to\_sort=None, virtual\_fields=None)

Return a list of sorted book ids. If ids\_to\_sort is None, all book ids are returned.

fields must be a list of 2-tuples of the form (field\_name, ascending=True or False). The most significant field is the first 2-tuple.

#### $notes_data_for(field, item_id) \rightarrow str$

Return all notes data as a dict or None if note does not exist

#### **notes\_for** (*field*, *item\_id*) $\rightarrow$ str

Return the notes document or an empty string if not found

#### notes\_resources\_used\_by (field, item\_id)

Return the set of resource hashes of all resources used by the note for the specified item

#### pref (name, default=None, namespace=None)

Return the value for the specified preference or the value specified as default if the preference is not set.

#### read\_backup(book\_id)

Return the OPF metadata backup for the book as a bytestring or None if no such backup exists.

#### remove\_books (book\_ids, permanent=False)

Remove the books specified by the book\_ids from the database and delete their format files. If permanent is False, then the format files are placed in the per-library trash directory.

**remove\_extra\_files** (*book\_id: int, relpaths: Iterable[str], permanent=False*)  $\rightarrow$  dict[str, Exception | None]

Delete the specified extra files, either to Recycle Bin or permanently.

#### remove\_formats (formats\_map, db\_only=False)

Remove the specified formats from the specified books.

#### **Parameters**

- formats\_map A mapping of book\_id to a list of formats to be removed from the book.
- db\_only If True, only remove the record for the format from the db, do not delete the actual format file from the filesystem.

#### Returns

A map of book id to set of formats actually deleted from the filesystem for that book

#### remove\_items (field, item\_ids, restrict\_to\_book\_ids=None)

Delete all items in the specified field with the specified ids. Returns the set of affected book ids. restrict\_to\_book\_ids is an optional set of books ids. If specified the items will only be removed from those books. rename\_extra\_files (book\_id, map\_of\_relpath\_to\_new\_relpath, replace=False)
Rename extra data files

rename\_items (field, item\_id\_to\_new\_name\_map, change\_index=True, restrict\_to\_book\_ids=None)

Rename items from a many-one or many-many field such as tags or series.

### Parameters

- change\_index When renaming in a series-like field also change the series\_index values.
- **restrict\_to\_book\_ids** An optional set of book ids for which the rename is to be performed, defaults to all books.

restore\_book (book\_id, mi, last\_modified, path, formats, annotations=())

Restore the book entry in the database for a book that already exists on the filesystem

#### restore\_original\_format (book\_id, original\_fmt)

Restore the specified format from the previously saved ORIGINAL\_FORMAT, if any. Return True on success. The ORIGINAL\_FORMAT is deleted after a successful restore.

### property safe\_read\_lock

A safe read lock is a lock that does nothing if the thread already has a write lock, otherwise it acquires a read lock. This is necessary to prevent DowngradeLockErrors, which can happen when updating the search cache in the presence of composite columns. Updating the search cache holds an exclusive lock, but searching a composite column involves reading field values via ProxyMetadata which tries to get a shared lock. There may be other scenarios that trigger this as well.

This property returns a new lock object on every access. This lock object is not recursive (for performance) and must only be used in a with statement as with cache.safe\_read\_lock: otherwise bad things will happen.

#### save\_original\_format(book\_id, fmt)

Save a copy of the specified format as ORIGINAL\_FORMAT, overwriting any existing ORIGI-NAL\_FORMAT.

#### search (query, restriction=", virtual\_fields=None, book\_ids=None)

Search the database for the specified query, returning a set of matched book ids.

#### **Parameters**

- **restriction** A restriction that is ANDed to the specified query. Note that restrictions are cached, therefore the search for a AND b will be slower than a with restriction b.
- virtual\_fields Used internally (virtual fields such as on\_device to search over).
- **book\_ids** If not None, a set of book ids for which books will be searched instead of searching all books.

Return of a tuple of annotations matching the specified Full-text query.

Search the text of notes using an FTS index. If the query is empty return all notes.

set\_annotations\_for\_book (book\_id, fmt, annots\_list, user\_type='local', user='viewer')
Set all annotations for the specified book\_id, fmt, user\_type and user.

set\_conversion\_options (options, fmt='PIPE')

options must be a map of the form {book\_id:conversion\_options}

set\_cover(book\_id\_data\_map)

Set the cover for this book. The data can be either a QImage, QPixmap, file object or bytestring. It can also be None, in which case any existing cover is removed.

set\_field(name, book\_id\_to\_val\_map, allow\_case\_change=True, do\_path\_update=True)

Set the values of the field specified by name. Returns the set of all book ids that were affected by the change.

#### **Parameters**

- book\_id\_to\_val\_map Mapping of book\_ids to values that should be applied.
- **allow\_case\_change** If True, the case of many-one or many-many fields will be changed. For example, if a book has the tag tag1 and you set the tag for another book to Tag1 then the both books will have the tag Tag1 if allow\_case\_change is True, otherwise they will both have the tag tag1.
- do\_path\_update Used internally, you should never change it.

set\_link\_map (field, value\_to\_link\_map, only\_set\_if\_no\_existing\_link=False)

Sets links for item values in field. Note: this method doesn't change values not in the value\_to\_link\_map

#### Parameters

- field the lookup name
- value\_to\_link\_map dict(field\_value:link, ...). Note that these are values, not field ids.

#### Returns

books changed by setting the link

Set metadata for the book id from the Metadata object mi

Setting force\_changes=True will force set\_metadata to update fields even if mi contains empty values. In this case, 'None' is distinguished from 'empty'. If mi.XXX is None, the XXX is not replaced, otherwise it is. The tags, identifiers, and cover attributes are special cases. Tags and identifiers cannot be set to None so they will always be replaced if force\_changes is true. You must ensure that mi contains the values you want the book to have. Covers are always changed if a new cover is provided, but are never deleted. Also note that force\_changes has no effect on setting title or authors.

**set\_notes\_for** (field, item\_id, doc: str, searchable\_text: str = ", resource\_hashes=(), remove\_unused\_resources=False)  $\rightarrow$  int

Set the notes document. If the searchable text is different from the document, specify it as searchable\_text. If the document references resources their hashes must be present in resource\_hashes. Set remove\_unused\_resources to True to cleanup unused resources, note that updating a note automatically cleans up resources pertaining to that note anyway.

```
set_pref (name, val, namespace=None)
```

Set the specified preference to the specified value. See also pref() (page 372).

```
split_if_is_multiple_composite(f, val)
```

If f is a composite column lookup key and the column is is\_multiple then split v into unique non-empty values. The comparison is case sensitive. Order is not preserved. Return a list() for compatibility with proxy metadata field getters, for example tags.

tags\_older\_than (tag, delta=None, must\_have\_tag=None, must\_have\_authors=None)

Return the ids of all books having the tag tag that are older than the specified time. tag comparison is case insensitive.

#### Parameters

- delta A timedelta object or None. If None, then all ids with the tag are returned.
- must\_have\_tag If not None the list of matches will be restricted to books that have this tag
- **must\_have\_authors** A list of authors. If not None the list of matches will be restricted to books that have these authors (case insensitive).

```
\texttt{unretire\_note\_for}(\mathit{field}, \mathit{item\_id}) \rightarrow \mathsf{int}
```

Unretire a previously retired note for the specified item. Notes are retired when an item is removed from the database

#### update\_annotations (annot\_id\_map)

Update annotations.

user\_categories\_for\_books (book\_ids, proxy\_metadata\_map=None)

Return the user categories for the specified books. proxy\_metadata\_map is optional and is useful for a performance boost, in contexts where a ProxyMetadata object for the books already exists. It should be a mapping of book\_ids to their corresponding ProxyMetadata objects.

# 14.9.2 API documentation for the e-book editing tools

The e-book editing tools consist of a *calibre.ebooks.oeb.polish.container.Container* (page 375) object that represents a book as a collection of HTML + resource files, and various tools that can be used to perform operations on the container. All the tools are in the form of module level functions in the various calibre.ebooks.oeb.polish.\* modules. You obtain a container object for a book at a path like this:

```
from calibre.ebooks.oeb.polish.container import get_container
container = get_container('Path to book file', tweak_mode=True)
```

If you are writing a plugin for the E-book editor, you get the current container for the book being edited like this:

```
from calibre.gui2.tweak_book import current_container
container = current_container()
if container is None:
    report_error # No book has been opened yet
```

### The Container object

class calibre.ebooks.oeb.polish.container.Container (rootpath=None, opfpath=None,

log=<calibre.utils.logging.Log object>,

*clone\_data=None*)

A container represents an open e-book as a folder full of files and an OPF file. There are two important concepts:

- The root folder. This is the base of the e-book. All the e-books files are inside this folder or in its sub-folders.
- Names: These are paths to the books' files relative to the root folder. They always contain POSIX separators and are unquoted. They can be thought of as canonical identifiers for files in the book. Most methods on the container object work with names. Names are always in the NFC Unicode normal form.

• Clones: the container object supports efficient on-disk cloning, which is used to implement checkpoints in the e-book editor. In order to make this work, you should never access files on the filesystem directly. Instead, use raw\_data() (page 378) or open() (page 378) to read/write to component files in the book.

When converting between hrefs and names use the methods provided by this class, they assume all hrefs are quoted.

#### abspath\_to\_name (fullpath, root=None)

Convert an absolute path to a canonical name relative to root

#### Parameters

**root** – The base folder. By default the root for this container object is used.

Add a file to this container. Entries for the file are automatically created in the OPF manifest and spine (if the file is a text document)

#### add\_name\_to\_manifest (name, process\_manifest\_item=None, suggested\_id=")

Add an entry to the manifest for a file with the specified name. Returns the manifest id.

#### add\_properties (name, \*properties)

Add the specified properties to the manifest item identified by name.

#### apply\_unique\_properties (name, \*properties)

Ensure that the specified properties are set on only the manifest item identified by name. You can pass None as the name to remove the property from all items.

#### book\_type = 'oeb'

The type of book (epub for EPUB files and azw3 for AZW3 files)

#### commit (outpath=None, keep\_parsed=False)

Commit all dirtied parsed objects to the filesystem and write out the e-book file at outpath.

#### **Parameters**

- **output** The path to write the saved e-book file to. If None, the path of the original book file is used.
- **keep\_parsed** If True the parsed representations of committed items are kept in the cache.

#### commit\_item (name, keep\_parsed=False)

Commit a parsed object to disk (it is serialized and written to the underlying file). If keep\_parsed is True the parsed representation is retained in the cache. See also: parsed() (page 378)

#### dirty(name)

Mark the parsed object corresponding to name as dirty. See also: parsed() (page 378).

#### exists (name)

True iff a file/folder corresponding to the canonical name exists. Note that this function suffers from the limitations of the underlying OS filesystem, in particular case (in)sensitivity. So on a case insensitive filesystem this will return True even if the case of name is different from the case of the underlying filesystem file. See also has\_name() (page 377)

#### filesize(name)

Return the size in bytes of the file represented by the specified canonical name. Automatically handles dirtied parsed objects. See also: *parsed()* (page 378)

#### generate\_item (name, id\_prefix=None, media\_type=None, unique\_href=True)

Add an item to the manifest with href derived from the given name. Ensures uniqueness of href and id automatically. Returns generated item.

#### get\_file\_path\_for\_processing(name, allow\_modification=True)

Similar to open() except that it returns a file path, instead of an open file object.

#### property guide\_type\_map

Mapping of guide type to canonical name

#### has\_name (name)

Return True iff a file with the same canonical name as that specified exists. Unlike *exists()* (page 376) this method is always case-sensitive.

#### href\_to\_name (href, base=None)

Convert an href (relative to base) to a name. base must be a name or None, in which case self.root is used.

#### insert\_into\_xml (parent, item, index=None)

Insert item into parent (or append if index is None), fixing indentation. Only works with self closing items.

#### is\_dir = False

If this container represents an unzipped book (a directory)

### iterlinks (name, get\_line\_numbers=True)

Iterate over all links in name. If get\_line\_numbers is True the yields results of the form (link, line\_number, offset). Where line\_number is the line\_number at which the link occurs and offset is the number of characters from the start of the line. Note that offset could actually encompass several lines if not zero.

#### make\_name\_unique (name)

Ensure that *name* does not already exist in this book. If it does, return a modified version that does not exist.

#### manifest\_has\_name(name)

Return True if the manifest has an entry corresponding to name

#### property manifest\_id\_map

Mapping of manifest id to canonical names

#### manifest\_items\_of\_type (predicate)

The names of all manifest items whose media-type matches predicate. *predicate* can be a set, a list, a string or a function taking a single argument, which will be called with the media-type.

#### manifest\_items\_with\_property (property\_name)

All manifest items that have the specified property

#### property manifest\_type\_map

Mapping of manifest media-type to list of canonical names of that media-type

#### property mi

The metadata of this book as a Metadata object. Note that this object is constructed on the fly every time this property is requested, so use it sparingly.

#### name\_to\_abspath(name)

Convert a canonical name to an absolute OS dependent path

#### name\_to\_href(name, base=None)

Convert a name to a href relative to base, which must be a name or None in which case self.root is used as the base

#### property names\_that\_must\_not\_be\_changed

Set of names that must never be renamed. Depends on the e-book file format.

#### property names\_that\_must\_not\_be\_removed

Set of names that must never be deleted from the container. Depends on the e-book file format.

#### property names\_that\_need\_not\_be\_manifested

Set of names that are allowed to be missing from the manifest. Depends on the e-book file format.

### **open** (*name*, *mode='rb'*)

Open the file pointed to by name for direct read/write. Note that this will commit the file if it is dirtied and remove it from the parse cache. You must finish with this file before accessing the parsed version of it again, or bad things will happen.

#### property opf

The parsed OPF file

#### opf\_get\_or\_create(name)

Convenience method to either return the first XML element with the specified name or create it under the opf:package element and then return it, if it does not already exist.

#### property opf\_version

The version set on the OPF's <package> element

### property opf\_version\_parsed

The version set on the OPF's <package> element as a tuple of integers

#### opf\_xpath(expr)

Convenience method to evaluate an XPath expression on the OPF file, has the opf: and dc: namespace prefixes pre-defined.

#### parsed(name)

Return a parsed representation of the file specified by name. For HTML and XML files an lxml tree is returned. For CSS files a css\_parser stylesheet is returned. Note that parsed objects are cached for performance. If you make any changes to the parsed object, you must call *dirty()* (page 376) so that the container knows to update the cache. See also *replace()* (page 379).

#### raw\_data (name, decode=True, normalize\_to\_nfc=True)

Return the raw data corresponding to the file specified by name

#### Parameters

- decode If True and the file has a text based MIME type, decode it and return a unicode object instead of raw bytes.
- normalize\_to\_nfc If True the returned unicode object is normalized to the NFC normal form as is required for the EPUB and AZW3 file formats.

#### relpath (path, base=None)

Convert an absolute path (with os separators) to a path relative to base (defaults to self.root). The relative path is *not* a name. Use *abspath\_to\_name()* (page 376) for that.

#### remove\_from\_spine (spine\_items, remove\_if\_no\_longer\_in\_spine=True)

Remove the specified items (by canonical name) from the spine. If remove\_if\_no\_longer\_in\_spine is True, the items are also deleted from the book, not just from the spine.

#### remove\_from\_xml(item)

Removes item from parent, fixing indentation (works only with self closing items)

#### remove\_item (name, remove\_from\_guide=True)

Remove the item identified by name from this container. This removes all references to the item in the OPF manifest, guide and spine as well as from any internal caches.

#### rename (current\_name, new\_name)

Renames a file from current\_name to new\_name. It automatically rebases all links inside the file if the folder the file is in changes. Note however, that links are not updated in the other files that could reference this file. This is for performance, such updates should be done once, in bulk.

#### replace (name, obj)

Replace the parsed object corresponding to name with obj, which must be a similar object, i.e. an lxml tree for HTML/XML or a css\_parser stylesheet for a CSS file.

#### replace\_links(name, replace\_func)

Replace all links in name using replace\_func, which must be a callable that accepts a URL and returns the replaced URL. It must also have a 'replaced' attribute that is set to True if any actual replacement is done. Convenient ways of creating such callables are using the LinkReplacer and LinkRebaser classes.

#### serialize\_item(name)

Convert a parsed object (identified by canonical name) into a bytestring. See parsed () (page 378).

#### set\_spine (spine\_items)

Set the spine to be spine\_items where spine\_items is an iterable of the form (name, linear). Will raise an error if one of the names is not present in the manifest.

#### property spine\_items

An iterator yielding the path for every item in the books' spine. See also: *spine\_iter* (page 379) and *spine\_items* (page 379).

#### property spine\_iter

An iterator that yields item, name is\_linear for every item in the books' spine. item is the lxml element, name is the canonical file name and is\_linear is True if the item is linear. See also: *spine\_names* (page 379) and *spine\_items* (page 379).

### property spine\_names

An iterator yielding name and is\_linear for every item in the books' spine. See also: *spine\_iter* (page 379) and *spine\_items* (page 379).

#### Managing component files in a container

### 

Replace links to files in the container. Will iterate over all files in the container and change the specified links in them.

#### Parameters

- link\_map A mapping of old canonical name to new canonical name. For example: {'images/old.png': 'images/new.png'}
- **frag\_map** A callable that takes two arguments (name, anchor) and returns a new anchor. This is useful if you need to change the anchors in HTML files. By default, it does nothing.
- replace\_in\_opf If False, links are not replaced in the OPF file.

calibre.ebooks.oeb.polish.replace.rename\_files(container, file\_map)

Rename files in the container, automatically updating all links to them.

#### **Parameters**

file\_map - A mapping of old canonical name to new canonical name, for example: {'text/
chapter1.html'; 'chapter1.html'}.

calibre.ebooks.oeb.polish.replace.get\_recommended\_folders (container, names)

Return the folders that are recommended for the given filenames. The recommendation is based on where the majority of files of the same type are located in the container. If no files of a particular type are present, the recommended folder is assumed to be the folder containing the OPF file.

#### Pretty printing and auto fixing parse errors

calibre.ebooks.oeb.polish.pretty.fix\_html(container, raw)

Fix any parsing errors in the HTML represented as a string in raw. Fixing is done using the HTML5 parsing algorithm.

calibre.ebooks.oeb.polish.pretty.fix\_all\_html(container)

Fix any parsing errors in all HTML files in the container. Fixing is done using the HTML5 parsing algorithm.

calibre.ebooks.oeb.polish.pretty.pretty\_html(container, name, raw)

Pretty print the HTML represented as a string in raw

calibre.ebooks.oeb.polish.pretty.pretty\_css(container, name, raw)

Pretty print the CSS represented as a string in raw

calibre.ebooks.oeb.polish.pretty.pretty\_xml (container, name, raw)

Pretty print the XML represented as a string in raw. If name is the name of the OPF, extra OPF-specific prettying is performed.

calibre.ebooks.oeb.polish.pretty.pretty\_all(container)

Pretty print all HTML/CSS/XML files in the container

### Managing book jackets

calibre.ebooks.oeb.polish.jacket.remove\_jacket(container)

Remove an existing jacket, if any. Returns False if no existing jacket was found.

calibre.ebooks.oeb.polish.jacket.add\_or\_replace\_jacket (container)

Either create a new jacket from the book's metadata or replace an existing jacket. Returns True if an existing jacket was replaced.

#### Splitting and merging of files

calibre.ebooks.oeb.polish.split.**split** (container, name, loc\_or\_xpath, before=True, totals=None)

Split the file specified by name at the position specified by loc\_or\_xpath. Splitting automatically migrates all links and references to the affected files.

#### Parameters

- **loc\_or\_xpath** Should be an XPath expression such as //h:div[@id="split\_here"]. Can also be a *loc* which is used internally to implement splitting in the preview panel.
- before If True the split occurs before the identified element otherwise after it.
- totals Used internally

calibre.ebooks.oeb.polish.split.multisplit (container, name, xpath, before=True)

Split the specified file at multiple locations (all tags that match the specified XPath expression). See also: *split()* (page 380). Splitting automatically migrates all links and references to the affected files.

#### **Parameters**

before – If True the splits occur before the identified element otherwise after it.

calibre.ebooks.oeb.polish.split.merge (container, category, names, master)

Merge the specified files into a single file, automatically migrating all links and references to the affected files. The file must all either be HTML or CSS files.

#### Parameters

- category Must be either 'text' for HTML files or 'styles' for CSS files
- names The list of files to be merged
- **master** Which of the merged files is the *master* file, that is, the file that will remain after merging.

#### **Managing covers**

calibre.ebooks.oeb.polish.cover.set\_cover(container, cover\_path, report=None, options=None)

Set the cover of the book to the image pointed to by cover\_path.

#### Parameters

- **cover\_path** Either the absolute path to an image file or the canonical name of an image in the book. When using an image in the book, you must also set options, see below.
- **report** An optional callable that takes a single argument. It will be called with information about the tasks being processed.
- **options** None or a dictionary that controls how the cover is set. The dictionary can have entries: **keep\_aspect**: True or False (Preserve aspect ratio of covers in EPUB) **no\_svg**: True or False (Use an SVG cover wrapper in the EPUB titlepage) **existing**: True or False (cover\_path refers to an existing image in the book)

calibre.ebooks.oeb.polish.cover.mark\_as\_cover(container, name)

Mark the specified image as the cover image.

calibre.ebooks.oeb.polish.cover.mark\_as\_titlepage(container, name, move\_to\_start=True)

Mark the specified HTML file as the titlepage of the EPUB.

#### Parameters

move\_to\_start - If True the HTML file is moved to the start of the spine

### Working with CSS

calibre.ebooks.oeb.polish.fonts.change\_font (container, old\_name, new\_name=None)

Change a font family from old\_name to new\_name. Changes all occurrences of the font family in stylesheets, style tags and style attributes. If the old\_name refers to an embedded font, it is removed. You can set new\_name to None to remove the font family instead of changing it.

calibre.ebooks.oeb.polish.css.remove\_unused\_css (container, report=None,

remove\_unused\_classes=False, merge\_rules=False, merge\_rules\_with\_identical\_properties=False, remove\_unreferenced\_sheets=False)

Remove all unused CSS rules from the book. An unused CSS rule is one that does not match any actual content.

#### Parameters

• **report** – An optional callable that takes a single argument. It is called with information about the operations being performed.

- **remove\_unused\_classes** If True, class attributes in the HTML that do not match any CSS rules are also removed.
- merge\_rules If True, rules with identical selectors are merged.
- merge\_rules\_with\_identical\_properties If True, rules with identical properties are merged.
- **remove\_unreferenced\_sheets** If True, stylesheets that are not referenced by any content are removed

calibre.ebooks.oeb.polish.css.filter\_css(container, properties, names=())

Remove the specified CSS properties from all CSS rules in the book.

#### Parameters

- properties Set of properties to remove. For example: { 'font-family', 'color' }.
- names The files from which to remove the properties. Defaults to all HTML and CSS files in the book.

#### Working with the Table of Contents

calibre.ebooks.oeb.polish.toc.from\_xpaths(container, xpaths, prefer\_title=False)

Generate a Table of Contents from a list of XPath expressions. Each expression in the list corresponds to a level of the generate ToC. For example: ['//h:h1', '//h:h2', '//h:h3'] will generate a three level Table of Contents from the <h1>, <h2> and <h3> tags.

calibre.ebooks.oeb.polish.toc.from\_links(container)

Generate a Table of Contents from links in the book.

calibre.ebooks.oeb.polish.toc.from\_files(container)

Generate a Table of Contents from files in the book.

calibre.ebooks.oeb.polish.toc.create\_inline\_toc(container, title=None)

Create an inline (HTML) Table of Contents from an existing NCX Table of Contents.

#### Parameters

title – The title for this table of contents.

#### Edit book tool

```
class calibre.gui2.tweak_book.plugin.Tool
```

```
Bases: object
```

The base class for individual tools in an Edit Book plugin. Useful members include:

- self.plugin: A reference to the *calibre.customize.Plugin* (page 254) object to which this tool belongs.
- self. boss (page 383)
- self. gui (page 383)

Methods that must be overridden in sub classes:

- create\_action() (page 383)
- register\_shortcut() (page 383)

#### name = None

Set this to a unique name it will be used as a key

#### allowed\_in\_toolbar = True

If True the user can choose to place this tool in the plugins toolbar

#### allowed\_in\_menu = True

If True the user can choose to place this tool in the plugins menu

#### toolbar\_button\_popup\_mode = 'delayed'

The popup mode for the menu (if any) of the toolbar button. Possible values are 'delayed', 'instant', 'button'

#### property boss

The calibre.gui2.tweak\_book.boss.Boss (page 384) object. Used to control the user interface.

#### property gui

The main window of the user interface

#### property current\_container

Return the current *calibre.ebooks.oeb.polish.container.Container* (page 375) object that represents the book being edited.

Register a keyboard shortcut that will trigger the specified qaction. This keyboard shortcut will become automatically customizable by the user in the Keyboard shortcuts section of the editor preferences.

#### **Parameters**

- qaction A QAction object, it will be triggered when the configured key combination is pressed by the user.
- unique\_name A unique name for this shortcut/action. It will be used internally, it must not be shared by any other actions in this plugin.
- **default\_keys** A list of the default keyboard shortcuts. If not specified no default shortcuts will be set. If the shortcuts specified here conflict with either builtin shortcuts or shortcuts from user configuration/other plugins, they will be ignored. In that case, users will have to configure the shortcuts manually via Preferences. For example: default\_keys=('Ctrl+J', 'F9').
- **short\_text** An optional short description of this action. If not specified the text from the QAction will be used.
- **description** An optional longer description of this action, it will be used in the preferences entry for this shortcut.

#### create\_action (for\_toolbar=True)

Create a QAction that will be added to either the plugins toolbar or the plugins menu depending on for\_toolbar. For example:

```
def create_action(self, for_toolbar=True):
    ac = QAction(get_icons('myicon.png'), 'Do something')
    if for_toolbar:
        # We want the toolbar button to have a popup menu
        menu = QMenu()
        ac.setMenu(menu)
        menu.addAction('Do something else')
        subaction = menu.addAction('And another')
        # Register a keyboard shortcut for this toolbar action be
```

(continues on next page)

(continued from previous page)

```
# careful to do this for only one of the toolbar action or
# the menu action, not both.
self.register_shortcut(ac, 'some-unique-name', default_keys=('Ctrl+K',
→))
return ac
```

#### Arr See also

```
Method register_shortcut() (page 383).
```

#### Controlling the editor's user interface

The e-book editor's user interface is controlled by a single global *Boss* object. This has many useful methods that can be used in plugin code to perform common tasks.

```
class calibre.gui2.tweak_book.boss.Boss(parent, notify=None)
```

#### add\_savepoint (msg)

Create a restore checkpoint with the name specified as msg

```
apply_container_update_to_gui(mark_as_modified=True)
```

Update all the components of the user interface to reflect the latest data in the current book container.

#### **Parameters**

**mark\_as\_modified** – If True, the book will be marked as modified, so the user will be prompted to save it when quitting.

#### close\_editor(name)

Close the editor that is editing the file specified by name

#### commit\_all\_editors\_to\_container()

Commit any changes that the user has made to files open in editors to the container. You should call this method before performing any actions on the current container

#### property currently\_editing

Return the name of the file being edited currently or None if no file is being edited

edit\_file(name, syntax=None, use\_template=None)

Open the file specified by name in an editor

#### **Parameters**

- **syntax** The media type of the file, for example, 'text/html'. If not specified it is guessed from the file extension.
- use\_template A template to initialize the opened editor with

#### open\_book (path=None, edit\_file=None, clear\_notify\_data=True, open\_folder=False, search\_text=None)

Open the e-book at path for editing. Will show an error if the e-book is not in a supported format or the current book has unsaved changes.

#### Parameters

edit\_file - The name of a file inside the newly opened book to start editing. Can also be a
list of names.

#### rewind\_savepoint()

Undo the previous creation of a restore checkpoint, useful if you create a checkpoint, then abort the operation with no changes

#### $save_book()$

Save the book. Saving is performed in the background

### set\_modified()

Mark the book as having been modified

#### show\_current\_diff(allow\_revert=True, to\_container=None)

Show the changes to the book from its last checkpointed state

#### **Parameters**

- **allow\_revert** If True the diff dialog will have a button to allow the user to revert all changes
- to\_container A container object to compare the current container to. If None, the previously checkpointed container is used

#### show\_editor(name)

Show the editor that is editing the file specified by name

### sync\_preview\_to\_editor()

Sync the position of the preview panel to the current cursor position in the current editor

CHAPTER

# FIFTEEN

# **DIGITAL RIGHTS MANAGEMENT (DRM)**

Digital Rights Management (DRM) is a generic term for access control technologies that can be used by hardware manufacturers, publishers, copyright holders and individuals to try to impose limitations on the usage of digital content and devices. It is also, sometimes, disparagingly described as Digital Restrictions Management. The term is used to describe any technology which inhibits uses (legitimate or otherwise) of digital content that were not desired or foreseen by the content provider. The term generally doesn't refer to other forms of copy protection which can be circumvented without modifying the file or device, such as serial numbers or key-files. It can also refer to restrictions associated with specific instances of digital works or devices. DRM technologies attempt to control use of digital media by preventing access, copying or conversion to other formats by end users. See Wikipedia<sup>143</sup>.

# 15.1 What does DRM imply for me personally?

When you buy an e-book with DRM you don't really own it but have purchased the permission to use it in a manner dictated to you by the seller. DRM limits what you can do with e-books you have "bought". Often people who buy books with DRM are unaware of the extent of these restrictions. These restrictions prevent you from reformatting the e-book to your liking, including making stylistic changes like adjusting the font sizes, although there is software that empowers you to do such things for non DRM books. People are often surprised that an e-book they have bought in a particular format cannot be converted to another format if the e-book has DRM. So if you have an Amazon Kindle and buy a book sold by Barnes and Nobles, you should know that if that e-book has DRM you will not be able to read it on your Kindle. Notice that I am talking about a book you buy, not steal or pirate but BUY.

# 15.2 What does DRM do for authors?

Publishers of DRMed e-books argue that the DRM is all for the sake of authors and to protect their artistic integrity and prevent piracy. But DRM does NOT prevent piracy. People who want to pirate content or use pirated content still do it and succeed. The three major DRM schemes for e-books today are run by Amazon, Adobe and Barnes and Noble and all three DRM schemes have been cracked. All DRM does is inconvenience legitimate users. It can be argued that it actually harms authors as people who would have bought the book choose to find a pirated version as they are not willing to put up with DRM. Those that would pirate in the absence of DRM do so in its presence as well. To reiterate, the key point is that DRM *does not prevent piracy*. So DRM is not only pointless and harmful to buyers of e-books but also a waste of money.

# 15.3 DRM and freedom

Although digital content can be used to make information as well as creative works easily available to everyone and empower humanity, this is not in the interests of some publishers who want to steer people away from this possibility of freedom simply to maintain their relevance in world developing so fast that they can't keep up.

<sup>143</sup> https://en.wikipedia.org/wiki/Digital\_rights\_management

# 15.4 Why does calibre not support DRM?

calibre is open source software while DRM by its very nature is closed. If calibre were to support opening or viewing DRM files it could be trivially modified to be used as a tool for DRM removal which is illegal under today's laws. Open source software and DRM are a clash of principles. While DRM is all about controlling the user, open source software is about empowering the user. The two simply can not coexist.

# 15.5 What is calibre's view on content providers?

We firmly believe that authors and other content providers should be compensated for their efforts, but DRM is not the way to go about it. We are developing this database of DRM-free e-books from various sources to help you find DRM-free alternatives and to help independent authors and publishers of DRM-free e-books publicize their content. We hope you will find this useful and we request that you do not pirate the content made available to you here.

# 15.6 How can I help fight DRM?

As somebody who reads and buys e-books you can help fight DRM. Do not buy e-books with DRM. There are some publishers who publish DRM-free e-books. Make an effort to see if they carry the e-book you are looking for. If you like books by certain independent authors that sell DRM-free e-books and you can afford it make donations to them. This is money well spent as their e-books tend to be cheaper (there may be exceptions) than the ones you would buy from publishers of DRMed books and would probably work on all devices you own in the future saving you the cost of buying the e-book again. Do not discourage publishers and authors of DRM-free e-books by pirating their content. Content providers deserve compensation for their efforts. Do not punish them for trying to make your reading experience better by making available DRM-free e-books. In the long run this is detrimental to you. If you have bought books from sellers that carry both DRMed as well as DRM-free books, not knowing if they carry DRM or not make it a point to leave a comment or review on the website informing future buyers of its DRM status. Many sellers do not think it important to clearly indicate to their buyers if an e-book carries DRM or not. Here<sup>144</sup> you will find a guide to DRM-free living.

<sup>&</sup>lt;sup>144</sup> https://www.defectivebydesign.org/guide/ebooks

# CHAPTER

# SIXTEEN

# GLOSSARY

### RSS

**RSS** (*Really Simple Syndication*) is a web feed format that is used to publish frequently updated content, like news articles, blog posts, etc. It is a format that is particularly suited to being read by computers, and is therefore the preferred way of getting content from the web into an e-book. There are many other feed formats in use on the Internet, and calibre understands most of them. In particular, it has good support for the *ATOM* format, which is commonly used for blogs.

#### recipe

A recipe is a set of instructions that teach calibre how to convert an online news source, such as a magazine or a blog, into an e-book. A recipe is essentially Python<sup>145</sup> code. As such, it is capable of converting arbitrarily complex news sources into e-books. At the simplest level, it is just a set of variables, such as URLs, that give calibre enough information to go out onto the Internet and download the news.

### HTML

**HTML** (*Hyper Text Mark-Up Language*), a subset of Standard Generalized Mark-Up Language (SGML) for electronic publishing, is the specific standard used for the World Wide Web.

### CSS

**CSS** (*Cascading Style Sheets*) is a language used to describe how an *HTML* document should be rendered (visual styling).

#### API

**API** (*Application Programming Interface*) is a source code interface that a library provides to support requests for services to be made of it by computer programs.

### LRF

LRF The e-book format that is read by the SONY e-book readers.

#### URL

URL (Uniform Resource Locator) for example: http://example.com

#### regexp

**Regular expressions** provide a concise and flexible means for identifying strings of text of interest, such as particular characters, words, or patterns of characters. See *the tutorial* (page 212) for an introduction to regular expressions.

<sup>145</sup> https://www.python.org

# **PYTHON MODULE INDEX**

### С

calibre.customize, 253 calibre.customize.conversion, 263 calibre.db.cache,365 calibre.devices.interface, 266 calibre.ebooks.metadata.book.base, 208 calibre.ebooks.metadata.sources.base, 259 calibre.ebooks.oeb.polish.container,375 calibre.ebooks.oeb.polish.cover, 381 calibre.ebooks.oeb.polish.css, 381 calibre.ebooks.oeb.polish.jacket, 380 calibre.ebooks.oeb.polish.pretty, 380 calibre.ebooks.oeb.polish.replace, 379 calibre.ebooks.oeb.polish.split, 380 calibre.ebooks.oeb.polish.toc, 382 calibre.gui2.tweak\_book.boss, 384 calibre.utils.formatter\_functions, 182 calibre.web.feeds.news, 41

# INDEX

### Symbols

-1

calibredb-add command line option, 313-H ebook-polish command line option, 350– I calibredb-add command line option, 312 fetch-ebook-metadata command line option, 352 -S calibredb-add command line option, 313 -Т calibredb-add command line option, 313 -U ebook-polish command line option, 351--access-log calibre-server command line option, 305 --add calibredb-add command line option, 313 --add-alt-text-to-img ebook-convert command line option, 328--add-plugin calibre-customize command line option, 302 --add-simple-plugin calibre-debug command line option, 303--add-soft-hyphens ebook-polish command line option, 350 --ajax-timeout calibre-server command line option, 305--all calibredb-backup\_metadata command line option, 320 calibredb-export command line option, 315 --allow-local-files-outside-root ebook-convert-html-input command line option, 333 --allowed-plugin fetch-ebook-metadata command line option, 352 --append

calibredb-set\_custom command line option, 319 --as-extra-data-file calibredb-add\_format command line option, 313 --as-opf calibredb-show\_metadata command line option, 314 --ascending calibredb-list command line option, 311 --asciiize ebook-convert command line option, 325 --attachment calibre-smtp command line option, 309 --auth-mode calibre-server command line option, 305 --author-sort ebook-convert command line option, 330ebook-meta command line option, 348--authors calibredb-add command line option, 312 ebook-convert command line option, 330 ebook-meta command line option, 348 fetch-ebook-metadata command line option, 352 --auto-reload calibre-server command line option, 305 --automerge calibredb-add command line option, 312 --ban-after calibre-server command line option, 305 --ban-for calibre-server command line option, 305 --base-dir web2disk command line option, 354 --base-font-size ebook-convert command line option, 325 --book-list-mode calibre-server command line option, 305--book-producer ebook-convert command line option, 330 ebook-meta command line option, 348

--breadth-first ebook-convert-html-input command line option, 333 --build-plugin calibre-customize command line option, 302 --cafile calibre-smtp command line option, 309 --catalog-title calibredb-catalog command line option, 316 --categories calibredb-list\_categories command line option, 320 --category ebook-meta command line option, 348 --change-justification ebook-convert command line option, 325 --chapter ebook-convert command line option, 328 --chapter-mark ebook-convert command line option, 328 --colors ebook-convert-comic-input command line option, 331 --comic-image-size ebook-convert-comic-input command line option, 331 --command calibre-debug command line option, 303--comments ebook-convert command line option, 330ebook-meta command line option, 348 --compress-images ebook-polish command line option, 350--compress-min-size calibre-server command line option, 305 --continue ebook-viewer command line option, 351 --cover calibredb-add command line option, 312 ebook-convert command line option, 330 ebook-meta command line option, 349 ebook-polish command line option, 350fetch-ebook-metadata command line option, 352 --cross-reference-authors calibredb-catalog command line option, 316 --csv calibredb-check\_library command line option, 319 calibredb-list\_categories command line option, 320

--custom-list-template calibre-server command line option, 305--custom-size ebook-convert-pdf-output command line option, 343 --customize-plugin calibre-customize command line option, 302 --daemonize calibre-server command line option, 305--date ebook-meta command line option, 349 --debug-device-driver calibre-debug command line option, 303--debug-pipeline calibredb-catalog command line option, 316 ebook-convert command line option, 331 --default-programs calibre-debug command line option, 303--delay web2disk command line option, 354 --despeckle ebook-convert-comic-input command line option, 331 --detach calibre command line option, 301ebook-edit command line option, 348ebook-viewer command line option, 351 --details calibredb-custom\_columns command line option, 318 --dialect calibredb-list\_categories command line option, 320 --diff calibre-debug command line option, 303 --disable-allow-socket-preallocation calibre-server command line option, 305 --disable-auth calibre-server command line option, 306--disable-dehyphenate ebook-convert command line option, 327 --disable-delete-blank-paragraphs ebook-convert command line option, 327 --disable-fallback-to-detected-interface calibre-server command line option, 306--disable-fix-indents ebook-convert command line option, 327 --disable-font-rescaling ebook-convert command line option, 325 --disable-format-scene-breaks ebook-convert command line option, 327 --disable-hyphenation

lrfviewer command line option, 353 --disable-italicize-common-cases ebook-convert command line option, 327 --disable-local-write calibre-server command line option, 306--disable-log-not-found calibre-server command line option, 306--disable-markup-chapter-headings ebook-convert command line option, 327 --disable-plugin calibre-customize command line option, 302 --disable-remove-fake-margins ebook-convert command line option, 328 --disable-renumber-headings ebook-convert command line option, 327 --disable-trim ebook-convert-comic-input command line option, 331 --disable-unwrap-lines ebook-convert command line option, 327 --disable-use-bonjour calibre-server command line option, 306--disable-use-sendfile calibre-server command line option, 306--disallow-rendered-cover ebook-meta command line option, 349 --display calibredb-add\_custom\_column command line option, 318 --displayed-fields calibre-server command line option, 305--do-not-match-on-related-words calibredb-fts\_search command line option, 322 --docx-custom-page-size ebook-convert-docx-output command line option, 337 --docx-inline-subsup ebook-convert-docx-input command line option, 332 --docx-no-cover ebook-convert-docx-input command line option, 332 ebook-convert-docx-output command line option, 337 --docx-no-pagebreaks-between-notes ebook-convert-docx-input command line --dont-sharpen option, 332 --docx-no-toc ebook-convert-docx-output command line --dont-split-on-page-breaks option, 337 --docx-page-margin-bottom

ebook-convert-docx-output command line option, 337 --docx-page-margin-left ebook-convert-docx-output command line option, 337 --docx-page-margin-right ebook-convert-docx-output command line option, 337 --docx-page-margin-top ebook-convert-docx-output command line option, 337 --docx-page-size ebook-convert-docx-output command line option, 338 --dont-add-comic-pages-to-toc ebook-convert-comic-input command line option, 331 --dont-asciiize calibredb-export command line option, 315 --dont-compress ebook-convert-azw3-output command line option, 337 ebook-convert-mobi-output command line option, 342 --dont-download-recipe ebook-convert-recipe-input command line option, 335 --dont-download-stylesheets web2disk command line option, 354 --dont-grayscale ebook-convert-comic-input command line option, 331 --dont-normalize ebook-convert-comic-input command line option, 331 --dont-output-resources lrf2lrs command line option, 353 --dont-package ebook-convert-html-input command line option, 333 --dont-replace calibredb-add\_format command line option. 314 --dont-save-cover calibredb-export command line option, 315 --dont-save-extra-files calibredb-export command line option, 315ebook-convert-comic-input command line option, 332 ebook-convert-epub-output command line option, 338 ebook-convert-kepub-output command line

option, 340 --dont-update-metadata calibredb-export command line option, 315 --epub-max-image-size --dont-verify-server-certificate calibre-smtp command line option, 309--dont-write-opf calibredb-export command line option, 315 --download-external-resources ebook-polish command line option, 350 --duplicate-links-in-toc ebook-convert command line option, 329 --duplicates calibredb-add command line option, 312--edit-book calibre-debug command line option, 303--embed-all-fonts ebook-convert command line option, 325 --embed-font-family ebook-convert command line option, 325 --embed-fonts ebook-polish command line option, 350 --empty calibredb-add command line option, 312 --enable-allow-socket-preallocation calibre-server command line option, 305--enable-auth calibre-server command line option, 306--enable-autorotation ebook-convert-lrf-output command line option, 341 --enable-fallback-to-detected-interface calibre-server command line option, 306--enable-heuristics ebook-convert command line option, 327 --enable-local-write calibre-server command line option, 306--enable-log-not-found calibre-server command line option, 306--enable-plugin calibre-customize command line option, 302 --enable-use-bonjour calibre-server command line option, 306--enable-use-sendfile calibre-server command line option, 306--encoding web2disk command line option, 355 --encryption-method calibre-smtp command line option, 309 --epub-flatten ebook-convert-epub-output command line --filter-regexp option, 338 --epub-inline-toc

ebook-convert-epub-output command line option, 338 ebook-convert-epub-output command line option, 338 --epub-toc-at-end ebook-convert-epub-output command line option, 338 --epub-version ebook-convert-epub-output command line option, 338 --exclude-genre calibredb-catalog command line option, 316 --exclusion-rules calibredb-catalog command line option, 316 --exec-file calibre-debug command line option, 303--expand-css ebook-convert command line option, 325 --explode-book calibre-debug command line option, 303--export-all-calibre-data calibre-debug command line option, 303--extra-css ebook-convert command line option, 325--extract-to ebook-convert-azw3-output command line option, 337 ebook-convert-docx-output command line option, 338 ebook-convert-epub-output command line option, 338 ebook-convert-html-output command line option, 339 ebook-convert-kepub-output command line option, 340ebook-convert-mobi-output command line option, 342 --fb2-genre ebook-convert-fb2-output command line option, 339 --field calibredb-set\_metadata command line option, 314 --fields calibredb-list command line option, 311 --filter-css ebook-convert command line option, 325 web2disk command line option, 355 --flow-size

ebook-convert-epub-output command line option, 338 ebook-convert-kepub-output command line --genre-source-field option, 340 --font-size-mapping ebook-convert command line option, 325 --for-machine calibredb-list command line option, 311 --force calibredb-remove\_custom\_column command line option, 318 --force-max-line-length ebook-convert-txt-output command line option, 346 ebook-convert-txtz-output command line --header-format option, 347 --force-reload ebook-viewer command line option, 351 --fork calibre-smtp command line option, 308--format ebook-convert-pdb-output command line option, 343 --formats calibredb-export command line option, 315 --formatting-type ebook-convert-txt-input command line option, 336 --from-opf ebook-convert command line option, 330 ebook-meta command line option, 349 --full-image-depth ebook-convert-pml-output command line option, 345 --full-screen ebook-viewer command line option, 351 --fullscreen ebook-viewer command line option, 351 --generate-authors calibredb-catalog command line option, 316 --generate-descriptions calibredb-catalog command line option, 316 --generate-genres calibredb-catalog command line option, 316 --generate-recently-added calibredb-catalog command line option, 316 --generate-series calibredb-catalog command line option, 317 --generate-titles

calibredb-catalog command line option, 317 calibredb-catalog command line option, 317 --get-cover ebook-meta command line option, 349 --a11 i calibre-debug command line option, 303--gui-debug calibre-debug command line option, 304--header ebook-convert-lrf-output command line option, 341 ebook-convert-lrf-output command line option, 341 --header-note-source-field calibredb-catalog command line option, 317 --header-separation ebook-convert-lrf-output command line option, 341 --help calibre command line option, 301calibre-customize command line option, 302 calibre-debug command line option, 304calibre-server command line option, 306calibre-smtp command line option, 308command line option, 310ebook-convert command line option, 324 ebook-edit command line option, 348 ebook-meta command line option, 349ebook-polish command line option, 350ebook-viewer command line option, 351 fetch-ebook-metadata command line option, 352 lrf2lrs command line option, 353 lrfviewer command line option, 353 lrs2lrf command line option, 354 web2disk command line option, 355 --html-unwrap-factor ebook-convert command line option, 327 --htmlz-class-style ebook-convert-htmlz-output command line option, 340 --htmlz-css-type ebook-convert-htmlz-output command line option, 340 --htmlz-title-filename ebook-convert-htmlz-output command line option, 340 --identifier

calibredb-add command line option, 312 ebook-meta command line option, 349 fetch-ebook-metadata command line option, 352 --ids calibredb-catalog command line option, 316 --ignore calibredb-add command line option, 313--ignore\_extensions calibredb-check\_library command line option, 319 --ignore\_names calibredb-check\_library command line option, 319 --ignore-plugins calibre command line option, 301 --ignore-wmf ebook-convert-rtf-input command line option, 335 --ignored-fields calibre-server command line option, 306--implode-book calibre-debug command line option, 304 --import-calibre-data calibre-debug command line option, 304--include-snippets calibredb-fts\_search command line option, 322 --index ebook-meta command line option, 349 --indexing-speed calibredb-fts\_index command line option, 321 --indexing-threshold calibredb-fts\_search command line option. 322 --inline-toc ebook-convert-pdb-output command line option, 343 ebook-convert-pml-output command line option, 345 ebook-convert-rb-output command line option, 345 ebook-convert-txt-output command line option, 346 ebook-convert-txtz-output command line option, 347 --input-encoding ebook-convert-azw4-input command line option, 331 ebook-convert-chm-input command line option, 331 ebook-convert-comic-input command line

option, 332 ebook-convert-djvu-input command line option, 332 ebook-convert-docx-input command line option, 332 ebook-convert-epub-input command line option, 333 ebook-convert-fb2-input command line option, 333 ebook-convert-htlz-input command line option, 333 ebook-convert-html-input command line option, 333 ebook-convert-lit-input command line option, 333 ebook-convert-lrf-input line command option, 334 ebook-convert-mobi-input command line option, 334 ebook-convert-odt-input command line option, 334 ebook-convert-pdb-input line command option, 334 ebook-convert-pdf-input line command option, 334 ebook-convert-pml-input command line option, 335 ebook-convert-rb-input command line option, 335 ebook-convert-recipe-input command line option, 335 ebook-convert-rtf-input command line option, 335 ebook-convert-snb-input command line option, 336 command line ebook-convert-tcr-input option, 336 ebook-convert-txt-input command line option, 336 --input-profile ebook-convert command line option, 324 --insert-blank-line ebook-convert command line option, 325 --insert-blank-line-size ebook-convert command line option, 325 --insert-metadata ebook-convert command line option, 329 --inspect-mobi calibre-debug command line option, 304--is-multiple calibredb-add\_custom\_column command line option, 318 --isbn calibredb-add command line option, 312

ebook-convert command line option, 330 ebook-meta command line option, 349fetch-ebook-metadata command line option, 352 --item\_count calibredb-list\_categories command line option. 320 --jacket ebook-polish command line option, 350 --keep-aspect-ratio ebook-convert-comic-input command line option, 332 --keep-color ebook-convert-txt-output command line option, 346 ebook-convert-txtz-output command line --library-path option, 347 --keep-image-references ebook-convert-txt-output command line option, 346 ebook-convert-txtz-output command line --line-height option, 347 --keep-ligatures ebook-convert command line option, 326 --keep-links ebook-convert-txt-output command line option, 346 ebook-convert-txtz-output command line option, 347 --kepub-affect-hyphenation ebook-convert-kepub-output command line option, 340 --kepub-disable-hyphenation ebook-convert-kepub-output command line option, 340 --kepub-hyphenation-limit-lines ebook-convert-kepub-output command line --localhost option, 340 --kepub-hyphenation-min-chars ebook-convert-kepub-output command line option, 340 --kepub-hyphenation-min-chars-after ebook-convert-kepub-output command line option, 340 --kepub-hyphenation-min-chars-before ebook-convert-kepub-output command line option, 341 --kepub-max-image-size ebook-convert-kepub-output command line option, 341 --kepub-prefer-justification ebook-convert-kepub-output command line --margin-left option, 341 --kepubify

calibre-debug command line option, 304--landscape ebook-convert-comic-input command line option, 332--language ebook-convert command line option, 330 ebook-meta command line option, 349 --languages calibredb-add command line option, 312--level1-toc ebook-convert command line option, 329 --level2-toc ebook-convert command line option, 329--level3-toc ebook-convert command line option, 329 command line option, 310--limit calibredb-list command line option, 311 calibredb-search command line option, 321  ${\tt ebook-convert\ command\ line\ option,\ }326$ --line-width calibredb-list command line option, 311 --linearize-tables ebook-convert command line option, 326--list-fields calibredb-set\_metadata command line option, 314 --list-plugins calibre-customize command line option, 302 --list-recipes ebook-convert command line option, 324 --listen-on calibre-server command line option, 306calibre-smtp command line option, 308--loq calibre-server command line option, 306--lrf ebook-convert-recipe-input command line option, 335 --lrf-bookid ebook-meta command line option, 349--lrs lrs2lrf command line option, 354 --manage-users calibre-server command line option, 306--margin-bottom ebook-convert command line option, 326 ebook-convert command line option, 326 --margin-right

ebook-convert command line option, 326 --margin-top ebook-convert command line option, 326 --markdown-extensions ebook-convert-txt-input command line option, 336 --match-end-marker calibredb-fts\_search command line option, 322 --match-regexp web2disk command line option, 355 --match-start-marker calibredb-fts\_search command line option, 322 --max-files web2disk command line option, 355 --max-header-line-size calibre-server command line option, 306--max-job-time calibre-server command line option, 306--max-jobs calibre-server command line option, 307 --max-levels ebook-convert-html-input command line option, 333 --max-line-length ebook-convert-txt-output command line option, 346 ebook-convert-txtz-output command line option, 347 --max-log-size calibre-server command line option, 307--max-opds-items calibre-server command line option, 307 --max-opds-ungrouped-items calibre-server command line option, 307 --max-recursions web2disk command line option, 355 --max-request-body-size calibre-server command line option, 307 --max-toc-links ebook-convert command line option, 329 --merge-comments-rule calibredb-catalog command line option, 317 --minimum-indent ebook-convert-lrf-output command line option, 341 --minimum-line-height ebook-convert command line option, 326 --mobi-file-type ebook-convert-mobi-output command line --only-formats option, 342 --mobi-ignore-margins

ebook-convert-mobi-output command line option, 342 --mobi-keep-original-images ebook-convert-mobi-output command line option, 342 --mobi-toc-at-start ebook-convert-azw3-output command line option, 337 ebook-convert-mobi-output command line option, 342 --mono-family ebook-convert-lrf-output command line option, 341 --new-instance ebook-viewer command line option, 351 --newline ebook-convert-txt-output command line option, 346 ebook-convert-txtz-output command line option, 347 --no-chapters-in-toc ebook-convert command line option, 329 --no-default-epub-cover ebook-convert-epub-output command line option, 338 --no-images ebook-convert-pdf-input command line option, 334 --no-inline-fb2-toc ebook-convert-fb2-input command line option, 333 --no-inline-toc ebook-convert-azw3-output command line option, 337 ebook-convert-mobi-output command line option, 342 --no-process ebook-convert-comic-input command line option, 332 --no-sort ebook-convert-comic-input command line option, 332 --no-svq-cover ebook-convert-epub-output command line option, 339 --no-update-check calibre command line option, 301--num-per-page calibre-server command line option, 307--one-book-per-directory calibredb-add command line option, 313calibredb-embed\_metadata command line option, 321

--open-at ebook-viewer command line option, 352 --opf ebook-polish command line option, 350 fetch-ebook-metadata command line option, 352 --outbox calibre-smtp command line option, 308--output lrf2lrs command line option, 353 lrs2lrf command line option, 354 --output-format calibredb-fts\_search command line option, 322 ebook-convert-comic-input command line --pdf-mark-links option, 332 --output-profile calibredb-catalog command line option, 317 ebook-convert command line option, 324 --page-breaks-before ebook-convert command line option, 329--paper-size ebook-convert-pdf-output command line option, 343 --paragraph-type ebook-convert-txt-input command line option, 336 --password calibre-smtp command line option, 309 command line option, 311 ebook-convert-recipe-input command line option, 335 --paths calibre-debug command line option, 304 --pdb-output-encoding ebook-convert-pdb-output command line option, 343 --pdf-add-toc ebook-convert-pdf-output command line option, 343 --pdf-default-font-size ebook-convert-pdf-output line command option, 343 --pdf-engine ebook-convert-pdf-input command line option, 334 --pdf-footer-regex ebook-convert-pdf-input command line option, 334 --pdf-footer-skip ebook-convert-pdf-input command line option, 334 --pdf-footer-template

ebook-convert-pdf-output command line option, 343 --pdf-header-regex ebook-convert-pdf-input command line option, 334 --pdf-header-skip ebook-convert-pdf-input command line option, 334 --pdf-header-template ebook-convert-pdf-output command line option, 343 --pdf-hyphenate ebook-convert-pdf-output command line option, 343 ebook-convert-pdf-output command line option, 343 --pdf-mono-family ebook-convert-pdf-output command line option, 343 --pdf-mono-font-size ebook-convert-pdf-output command line option, 344 --pdf-no-cover ebook-convert-pdf-output command line option, 344 --pdf-odd-even-offset ebook-convert-pdf-output command line option, 344 --pdf-page-margin-bottom ebook-convert-pdf-output command line option, 344 --pdf-page-margin-left ebook-convert-pdf-output command line option, 344 --pdf-page-margin-right ebook-convert-pdf-output command line option, 344 --pdf-page-margin-top ebook-convert-pdf-output command line option, 344 --pdf-page-number-map ebook-convert-pdf-output command line option, 344 --pdf-page-numbers ebook-convert-pdf-output command line option, 344 --pdf-sans-family ebook-convert-pdf-output command line option, 344 --pdf-serif-family ebook-convert-pdf-output command line option, 344 --pdf-standard-font

ebook-convert-pdf-output command line option, 344 --pdf-use-document-margins ebook-convert-pdf-output command line option, 344 --permanent calibredb-remove command line option, 313 --personal-doc ebook-convert-mobi-output command line option, 342 --pidfile calibre-server command line option, 307 --pml-output-encoding ebook-convert-pml-output command line option, 345 --port calibre-server command line option, 307 calibre-smtp command line option, 309--prefer-author-sort ebook-convert-azw3-output command line option, 337 ebook-convert-mobi-output command line option, 342 --prefer-metadata-cover ebook-convert command line option, 329 --prefix calibredb-list command line option, 311 --prefix-rules calibredb-catalog command line option, 317 --preserve-cover-aspect-ratio ebook-convert-docx-output command line option, 338 ebook-convert-epub-output command line option, 339 ebook-convert-pdf-output command line option, 344 --preserve-spaces ebook-convert-txt-input command line option, 336 --preset calibredb-catalog command line option, --rating 317 --pretty-print ebook-convert-azw3-output command line --read-metadata-from-opf option, 337 ebook-convert-docx-output command line --really-do-it option, 338 ebook-convert-epub-output command line option, 339 ebook-convert-fb2-output command line option, 339 ebook-convert-html-output command line --recurse option, 339

ebook-convert-htmlz-output command line option, 340 ebook-convert-kepub-output command line option, 341 ebook-convert-lit-output command line option, 341 ebook-convert-lrf-output command line option, 341 ebook-convert-mobi-output command line option, 342 ebook-convert-oeb-output command line option, 343 ebook-convert-pdb-output command line option, 343 ebook-convert-pdf-output command line option, 344 ebook-convert-pml-output command line option, 345 command line ebook-convert-rb-output option, 345 ebook-convert-rtf-output command line option, 345 ebook-convert-snb-output command line option, 345 ebook-convert-tcr-output command line option, 346 ebook-convert-txt-output command line option, 347 ebook-convert-txtz-output command line option, 347 --profile lrfviewer command line option, 353 --progress calibredb-export command line option, 315--pubdate ebook-convert command line option, 330 --publisher ebook-convert command line option, 330 ebook-meta command line option, 349--raise-window ebook-viewer command line option, 352 ebook-convert command line option, 330 ebook-meta command line option, 349 ebook-convert command line option, 330 calibredb-restore\_database command line option, 319 --recipe-specific-option ebook-convert-recipe-input command line option, 335 calibredb-add command line option, 313

--relav calibre-smtp command line option, 309--remove-first-image ebook-convert command line option, 329 --remove-jacket ebook-polish command line option, 350 --remove-paragraph-spacing ebook-convert command line option, 326 --remove-paragraph-spacing-indent-size ebook-convert command line option, 326--remove-plugin calibre-customize command line option, 302 --remove-soft-hyphens ebook-polish command line option, 350--remove-unused-css ebook-polish command line option, 351 --render-tables-as-images ebook-convert-lrf-output command line --shutdown-running-calibre option, 341 --replace-scene-breaks ebook-convert command line option, 328--replace-whitespace calibredb-export command line option, 315 --single-dir --report calibredb-check\_library command line --smarten-punctuation option, 319 --restrict-to calibredb-fts\_search command line op- --snb-dont-indent-first-line tion. 322 --right2left ebook-convert-comic-input command line --snb-full-screen option, 332 --run-plugin calibre-debug command line option, 304--run-test calibre-debug command line option, 304 --run-without-debug calibre-debug command line option, 304--sans-family ebook-convert-lrf-output command line option, 341 --search calibredb-catalog command line option, 316 calibredb-list command line option, 311 --search-replace ebook-convert command line option, 328 --search-the-net-urls calibre-server command line option, 307 --sectionize ebook-convert-fb2-output command line option, 339 --select-text

ebook-edit command line option, 348 --separator calibredb-list command line option, 311 --series calibredb-add command line option, 312ebook-convert command line option, 330 ebook-meta command line option, 349 --series-index calibredb-add command line option, 313ebook-convert command line option, 330 --serif-family ebook-convert-lrf-output command line option, 342 --share-not-sync ebook-convert-azw3-output command line option, 337 ebook-convert-mobi-output command line option, 342 calibre command line option, 301 calibre-debug command line option, 304--shutdown-timeout calibre-server command line option, 307 calibredb-export command line option, 315 ebook-convert command line option, 326 ebook-polish command line option, 351ebook-convert-snb-output command line option, 345 ebook-convert-snb-output command line option, 346 --snb-hide-chapter-name ebook-convert-snb-output command line option, 346 --snb-insert-empty-line ebook-convert-snb-output command line option, 346 --snb-max-line-length ebook-convert-snb-output command line option, 346 --snb-output-encoding ebook-convert-snb-output command line option, 346 --sort-by calibredb-list command line option, 312 --sr1-replace ebook-convert command line option, 328--sr1-search ebook-convert command line option, 328 --sr2-replace ebook-convert command line option, 328

--sr2-search ebook-convert command line option, 328 --sr3-replace ebook-convert command line option, 328 --sr3-search ebook-convert command line option, 328 --ssl-certfile calibre-server command line option, 307 --ssl-keyfile calibre-server command line option, 307--start-in-tray calibre command line option, 302--start-reading-at ebook-convert command line option, 329 --subject calibre-smtp command line option, 309--subset-embedded-fonts ebook-convert command line option, 327 --subset-font calibre-debug command line option, 304 --subset-fonts ebook-polish command line option, 351 --tags calibredb-add command line option, 313 ebook-convert command line option, 330 ebook-meta command line option, 349--tcr-output-encoding ebook-convert-tcr-output command line option, 346 --template calibredb-export command line option, 315 calibredb-list command line option, 312--template\_file calibredb-list command line option, 312--template heading calibredb-list command line option, 312 --template-css ebook-convert-html-output command line option, 340 --template-html ebook-convert-html-output command line option, 340 --template-html-index ebook-convert-html-output command line --transform-html-rules option, 340 --test ebook-convert-recipe-input command line option, 335 --test-build calibre-debug command line option, 304 --text-size-multiplier-for-rendered-tables ebook-convert-lrf-output command line option, 342 --thumb-width

calibredb-catalog command line option, 317 --timefmt calibredb-export command line option, 315 --timeout calibre-server command line option, 307 calibre-smtp command line option, 308command line option, 311 fetch-ebook-metadata command line option, 352 web2disk command line option, 355 --timestamp ebook-convert command line option, 330--title calibredb-add command line option, 313ebook-convert command line option, 330 ebook-meta command line option, 349fetch-ebook-metadata command line option. 352 --title-sort ebook-convert command line option, 331 ebook-meta command line option, 349--to-dir calibredb-export command line option, 315 --to-lowercase calibredb-export command line option, 315 --to-opf ebook-meta command line option, 349--toc-filter ebook-convert command line option, 329 --toc-threshold ebook-convert command line option, 329 --toc-title ebook-convert-azw3-output command line option, 337 ebook-convert-epub-output command line option, 339 ebook-convert-mobi-output command line option, 342 ebook-convert-pdf-output command line option, 345 --transform-css-rules ebook-convert command line option, 327 ebook-convert command line option, 327 --trusted-ips calibre-server command line option, 307--txt-in-remove-indents ebook-convert-txt-input command line option, 336 --txt-output-encoding ebook-convert-txt-output command line option, 347

ebook-convert-txtz-output command line option, 347 --txt-output-formatting ebook-convert-txt-output command line option, 347 ebook-convert-txtz-output command line option, 347 --un-kepubify calibre-debug command line option, 304--uncompressed-pdf ebook-convert-pdf-output command line option, 345--unit ebook-convert-pdf-output command line option, 345 --unsmarten-punctuation ebook-convert command line option, 327 --unwrap-factor ebook-convert-pdf-input command line option, 335 --upgrade-book ebook-polish command line option, 351 --url-prefix calibre-server command line option, 307 --use-auto-toc ebook-convert command line option, 330--use-existing-cover calibredb-catalog command line option, 317 --use-profile-size ebook-convert-pdf-output command line option, 345 --userdb calibre-server command line option, 308--username calibre-smtp command line option, 309 command line option, 311 ebook-convert-recipe-input command line option, 336 --vacuum-fts-db calibredb-check\_library command line option, 319 --verbose calibre command line option, 302calibre-smtp command line option, 308calibredb-catalog command line option, 316 ebook-convert command line option, 331 ebook-polish command line option, 351 fetch-ebook-metadata command line option, 353 lrf2lrs command line option, 353 lrfviewer command line option, 353lrs2lrf command line option, 354

web2disk command line option, 355 --version calibre command line option, 302calibre-customize command line option, 302 calibre-debug command line option, 304calibre-server command line option, 308calibre-smtp command line option, 308command line option, 311ebook-convert command line option, 324 ebook-edit command line option, 348 ebook-meta command line option, 350ebook-polish command line option, 351 ebook-viewer command line option, 352 fetch-ebook-metadata command line option, 353 lrf2lrs command line option, 353 lrfviewer command line option, 354 lrs2lrf command line option, 354 web2disk command line option, 355 --viewer calibre-debug command line option, 304--visual-debug lrfviewer command line option, 354 --wait-for-completion calibredb-fts\_index command line option, 321 --white-background lrfviewer command line option, 354 --wide ebook-convert-comic-input command line option, 332 --width calibredb-list\_categories command line option, 320 --with-library calibre command line option, 302command line option, 310--wordspace ebook-convert-lrf-output command line option, 342 --worker-count calibre-server command line option, 308-a calibre-customize command line option, 302 calibre-smtp command line option, 309calibredb-add command line option, 312 calibredb-set\_custom command line option, 319 ebook-meta command line option, 348fetch-ebook-metadata command line option. 352 -b

calibre-customize command line option, 302 -C calibre-debug command line option, 303calibredb-add command line option, 312 calibredb-check\_library command line option. 319 calibredb-list\_categories command line option, 320 ebook-meta command line option, 348 ebook-polish command line option, 350 fetch-ebook-metadata command line option, 352 - i -d calibre-debug command line option, 303calibredb-add command line option, 312 calibredb-custom\_columns command line option, 318 ebook-convert command line option, 331 ebook-meta command line option, 349 ebook-polish command line option, 350fetch-ebook-metadata command line option, 352 web2disk command line option, 354 -j -0 calibre-debug command line option, 303-k calibre-smtp command line option, 309calibredb-add command line option, 312-1 calibredb-check\_library command line option, 319 ebook-polish command line option, 350 -f calibre-debug command line option, 304calibre-smtp command line option, 308calibredb-embed metadata command line option, 321 calibredb-list command line option, 311 -m calibredb-remove\_custom\_column command line option, 318 calibredb-set\_metadata command line option, 314 -n ebook-convert-pdb-output command line option, 343 ebook-polish command line option, 351 ebook-viewer command line option, 351 -q calibre-debug command line option, 303-h calibre command line option, 301-0 calibre-customize command line option, 302 calibre-debug command line option, 304 calibre-server command line option, 306calibre-smtp command line option, 308

command line option, 310ebook-convert command line option, 324 ebook-edit command line option, 348 ebook-meta command line option, 349ebook-polish command line option, 350ebook-viewer command line option, 351 fetch-ebook-metadata command line option, 352 lrf2lrs command line option, 353 lrfviewer command line option, 353 lrs2lrf command line option, 354 web2disk command line option, 355 calibre-debug command line option, 304 calibredb-add command line option, 312calibredb-catalog command line option, 316 calibredb-list\_categories command line option, 320 ebook-meta command line option, 349 ebook-polish command line option, 350fetch-ebook-metadata command line option, 352 ebook-polish command line option, 350 ebook-meta command line option, 348calibre-customize command line option, 302 calibre-smtp command line option, 308calibredb-add command line option, 312calibredb-search command line option, 321 calibredb-set\_metadata command line option, 314 ebook-meta command line option, 349 calibre-debug command line option, 304calibredb-add command line option, 312ebook-convert command line option, 330calibredb-check\_library command line option, 319 ebook-convert-txt-output command line option, 346 ebook-convert-txtz-output command line option, 347 web2disk command line option, 355 calibre-smtp command line option, 308ebook-polish command line option, 350fetch-ebook-metadata command line option.352

```
lrf2lrs command line option, 353
```

```
lrs2lrf command line option, 354
-p
   calibre-smtp command line option, 309
   ebook-meta command line option, 349
   ebook-polish command line option, 351
   fetch-ebook-metadata command line op-
       tion. 352
-r
   calibre-customize command line option,
       302
   calibre-debug command line option, 304
   calibre-smtp command line option, 309
   calibredb-add command line option, 313
   calibredb-check_library command
                                       line
       option, 319
   calibredb-list_categories command line
       option, 320
   calibredb-restore database command line
       option, 319
   ebook-meta command line option, 349
   web2disk command line option, 355
- 5
   calibre command line option, 301
   calibre-debug command line option, 304
   calibre-smtp command line option, 309
   calibredb-add command line option, 312
   calibredb-catalog command line option,
       316
   calibredb-list command line option, 311
   ebook-meta command line option, 349
-t.
   calibre-debug command line option, 304
   calibre-smtp command line option, 308
   calibredb-add command line option, 313
   calibredb-list command line option, 312
   ebook-meta command line option, 349
   fetch-ebook-metadata command line op-
       tion, 352
   web2disk command line option, 355
-u
   calibre-smtp command line option, 309
   ebook-convert-pdf-output command line
       option, 345
   ebook-polish command line option, 351
-77
   calibre command line option, 302
   calibre-smtp command line option, 308
   calibredb-catalog command line option,
       316
   ebook-convert command line option, 331
   fetch-ebook-metadata command line op-
       tion, 353
-w
   calibre-debug command line option, 304
```

```
calibredb-list command line option, 311
calibredb-list_categories command line
    option, 320
```

```
-x
```

calibre-debug command line option, 303

#### Α

<pre>abort_article()</pre>		(cali-
bre.web.feed	ls.news.BasicNew	vsRecipe method),
42		1 ,,
abort_recipe_pro	cossing()	(cali-
42	ls.news.BasicNew	vsRecipe method),
abspath_to_name(	)	(cali-
bre.ebooks.o method), 37	<i>eb.polish.contain</i> 6	er.Container
accept_drag_move	_event()	(cali-
bre.gui2.acti 280	ons.InterfaceActi	on method),
accept_enter_eventer_eventer_eventer_eventer	nt()	(cali-
	ons.InterfaceActi	
accepts_drops (ca. tribute), 280	libre.gui2.actions	InterfaceAction at-
action_add_menu <i>attribute</i> ),28		ions.InterfaceAction
action_menu_clone	e_qaction	(cali-
	ons.InterfaceActi	
action_shortcut_	name	(cali-
	ons.InterfaceActi	,
action_spec <i>attribute</i> ),28		ions.InterfaceAction
action_type		ions.InterfaceAction
attribute), 28	-	5
add_annotation_to		(cali-
	sbms.device.Dev	
277	isomis.uevice.Dev	ice memou),
add_book() ( <i>calibre</i> 273	.devices.interface	e.BookList method),
add_books() ( <i>calibr</i>	e dh cache Cach	e method) 366
add_books_to_meta		(cali-
270	nterface.DeviceP	lugin class method),
add_books_to_meta	adata()	(cali-
bre.devices.u 278	usbms.driver.USB	MS method),
add_custom_book_o <i>method</i> ), 36		ibre.db.cache.Cache
add_extra_files(		che.Cache method),
add_file()( <i>calibre</i> .	ebooks oeb notist	h container Containe
method), 37	-	

	mat() (calibre.db.cache.Cache method), 366	
add_lis	tener() (calibre.db.cache.Cache method), 36	6
add_nam	ne_to_manifest() (ca	li-
	bre.ebooks.oeb.polish.container.Container	
	<i>method</i> ), 376	
add_not	es_resource() (calibre.db.cache.Cach	he
	method), 366	
add_or_	<pre>replace_jacket() (in module ca bre.ebooks.oeb.polish.jacket), 380</pre>	li-
add_pro	operties() (ca	li-
_	bre.ebooks.oeb.polish.container.Container	
	method), 376	
add sav	repoint() (calibre.gui2.tweak_book.boss.Bo	255
	method), 384	
add toc		li-
aaa_000	bre.web.feeds.news.BasicNewsRecipe method	
	42	
adeify_	_images() (ca	
	bre.web.feeds.news.BasicNewsRecipe cla	ISS
	method), 41	
all_ann	<pre>notation_types() (calibre.db.cache.Ca</pre>	he
all_ann	otation_users() (calibre.db.cache.Ca	he
	<i>method</i> ), 366	
all_ann	otations() ( <i>calibre.db.cache.Cache method</i> 366	1),
all_ann	otations_for_book() (ca	li-
	bre.db.cache.Cache method), 366	
all_boo	k_ids() (calibre.db.cache.Cache method), 36	7
	<pre>eld_for() (calibre.db.cache.Cache method), 30</pre>	
	eld_ids() (calibre.db.cache.Cache method), 30	
	eld_keys() (ca	
····_	bre.ebooks.metadata.book.base.Metadata	
	method), 209	
all_fie	<pre>sld_names() (calibre.db.cache.Cache method 367</pre>	<i>l</i> ),
all non	_none_fields() (ca	li
	bre.ebooks.metadata.book.base.Metadata	<i>u</i> -
	method), 209	,
allowed	<pre>L_in_menu (calibre.gui2.tweak_book.plugin.To attribute), 383</pre>	001
allowed	l_in_toolbar (ca	li-
	bre.gui2.tweak_book.plugin.Tool attribute 382	e),
annotat	ion_count_for_book() (ca	li-
	bre.db.cache.Cache method), 367	
annotat	<pre>ions_map_for_book() (ca</pre>	li-
	bre.db.cache.Cache method), 367	
API, 389	· · ·	
·	container_update_to_gui() (ca	li-
11-1-0	bre.gui2.tweak_book.boss.Boss method), 384	
יי עוממה	nique_properties() (ca	li-
~rr+1_u	bre.ebooks.oeb.polish.container.Container	
	method), 376	

article	es_are_obfuscated (cali-
	bre.web.feeds.news.BasicNewsRecipe attribute),
	46
ASK_TO_	_ALLOW_CONNECT (cali-
	bre.devices.interface.DevicePlugin attribute),
	267
author	(calibre.customize.InterfaceActionBase attribute),
	282
author	(calibre.customize.MetadataReaderPlugin at-
	tribute), 258
author	(calibre.customize.MetadataWriterPlugin at-
	tribute), 258
author	(calibre.customize.Plugin attribute), 254
author	(calibre.customize.PreferencesPlugin attribute),
	283
author	(calibre.devices.usbms.driver.USBMS attribute),
	277
author	(calibre.ebooks.metadata.sources.base.Source at-
	tribute), 259
author_	_data() (calibre.db.cache.Cache method), 367
author_	_sort_from_authors() (cali-
	bre.db.cache.Cache method), 367
auto_cl	Leanup (calibre.web.feeds.news.BasicNewsRecipe
	attribute), 46
auto_cl	Leanup_keep (cali-
	bre.web.feeds.news.BasicNewsRecipe attribute),
	46
auto_re	epeat (calibre.gui2.actions.InterfaceAction
	attribute), 280
auto_tr	rim_covers (cali-
	bre.ebooks.metadata.sources.base.Source at-
	tribute), 260

#### В

BasicNewsRecipe (class in calibre.web.feeds.news), 41 BCD (calibre.devices.interface.DevicePlugin attribute), 266 BCD (calibre.devices.usbms.device.Device attribute), 275 book\_class (calibre.devices.usbms.driver.USBMS attribute), 277 book\_created (calibre.db.cache.Cache.EventType attribute), 365 book\_edited (calibre.db.cache.Cache.EventType attribute), 365 book\_type (calibre.ebooks.oeb.polish.container.Container attribute), 376 BookList (class in calibre.devices.interface), 273 booklist\_class (calibre.devices.usbms.driver.USBMS attribute), 277 books() (calibre.devices.interface.DevicePlugin method), 270 books() (calibre.devices.usbms.driver.USBMS method), 278 books\_for\_field() (calibre.db.cache.Cache method), 367

books\_in\_virtual\_library() (calibre.db.cache.Cache method), 367 books\_removed (calibre.db.cache.Cache.EventType attribute), 365 boss (calibre.gui2.tweak\_book.plugin.Tool property), 383 Boss (class in calibre.gui2.tweak book.boss), 384 browser\_type (calibre.web.feeds.news.BasicNewsRecipe attribute), 46 BuiltinAdd (class in calibre.utils.formatter\_functions), 182 BuiltinAnd (class in calibre.utils.formatter\_functions), 183 BuiltinAnnotationCount (class in calibre.utils.formatter\_functions), 191 BuiltinApproximateFormats in (class calibre.utils.formatter\_functions), 191 BuiltinArguments (class in calibre.utils.formatter functions), 201 BuiltinAssign (class in calibre.utils.formatter functions), 201 BuiltinAuthorLinks (class in calibre.utils.formatter functions), 191 BuiltinAuthorSorts (class in calibre.utils.formatter functions), 192 BuiltinBookCount (class in calibre.utils.formatter functions), 184 (class BuiltinBooksize in calibre.utils.formatter\_functions), 192 BuiltinBookValues (class in calibre.utils.formatter functions), 185 BuiltinCapitalize (class in calibre.utils.formatter\_functions), 184 BuiltinCeiling (class caliin bre.utils.formatter\_functions), 182 BuiltinCharacter (class in calibre.utils.formatter functions), 203 BuiltinCheckYesNo (class in calibre.utils.formatter\_functions), 203 BuiltinCmp (class in calibre.utils.formatter\_functions), 202BuiltinConnectedDeviceName (class in calibre.utils.formatter\_functions), 192 BuiltinConnectedDeviceUUID (class in calibre.utils.formatter\_functions), 192 BuiltinContains (class in calibre.utils.formatter\_functions), 203 BuiltinCount (class in calibre.utils. formatter\_functions), 197 BuiltinCurrentLibraryName (class in calibre.utils.formatter\_functions), 192 BuiltinCurrentLibraryPath (class in calibre.utils.formatter functions), 192 BuiltinCurrentVirtualLibraryName (class in calibre.utils.formatter functions), 193

	BuiltinDateArithmetic (class bre.utils.formatter_functions), 187	in	cali-
	BuiltinDaysBetween (class	in	cali-
	bre.utils.formatter_functions), 188		
	BuiltinDivide (class in		cali-
	bre.utils.formatter_functions), 182		
2	BuiltinEncodeForURL (class	in	cali-
	bre.utils.formatter_functions), 206		
,	BuiltinEval ( <i>class in calibre.utils.format.</i> 202	ter_func	ctions),
,	BuiltinExtraFileModtime (class	in	cali-
	bre.utils.formatter_functions), 185		
	BuiltinExtraFileNames (class	in	cali-
	bre.utils.formatter_functions), 185		
	BuiltinExtraFileSize (class	in	cali-
	bre.utils.formatter_functions), 186		
	BuiltinField ( <i>class in calibre.utils.format</i> 193	ter_func	ctions),
	BuiltinFieldExists (class	in	cali-
	bre.utils.formatter_functions), 204		
	BuiltinFieldListCount ( <i>class</i>	in	cali-
	bre.utils.formatter_functions), 197		
	BuiltinFinishFormatting (class	in	cali-
	bre.utils.formatter_functions), 188		
	BuiltinFirstMatchingCmp (class	in	cali-
	bre.utils.formatter_functions), 202		
	BuiltinFirstNonEmpty (class	in	cali-
	bre.utils.formatter_functions), 195		
	BuiltinFloor (class in calibre.utils.formate 183	ter_func	ctions),
	BuiltinFormatDate (class	in	cali-
	bre.utils.formatter_functions), 188		
	BuiltinFormatDateField (class	in	cali-
	bre.utils.formatter_functions), 189		
	BuiltinFormatDuration (class	in	cali-
	bre.utils.formatter_functions), 190		
	BuiltinFormatNumber (class	in	cali-
	bre.utils.formatter_functions), 191		
,	BuiltinFormatsModtimes (class	in	cali-
	bre.utils.formatter_functions), 193		
	BuiltinFormatsPaths (class	in	cali-
	bre.utils.formatter_functions), 194		
	BuiltinFormatsPathSegments (class	s in	cali-
	bre.utils.formatter_functions), 193		
	BuiltinFormatsSizes (class	in	cali-
	bre.utils.formatter_functions), 194		
,	BuiltinFractionalPart (class	in	cali-
	bre.utils.formatter_functions), 183		
	BuiltinGetLink (class in	ı	cali-
	bre.utils.formatter_functions), 186		
	BuiltinGetNote (class in	1	cali-
	bre.utils.formatter_functions), 186		1.
	BuiltinGlobals (class in	ı	cali-
	bre.utils.formatter_functions), 201		

	1.	
BuiltinHasCover ( <i>class in bre.utils.formatter_functions</i> ), 194		BuiltinMod (class in calibre.utils.formatter_functions), 183
BuiltinHasExtraFiles (class in bre.utils.formatter_functions), 187	cali-	BuiltinMultiply (class in cali- bre.utils.formatter_functions), 183
BuiltinHasNote (class in	cali-	
bre.utils.formatter_functions), 187		183
BuiltinHumanReadable (class in	cali-	BuiltinOndevice (class in cali-
	cun-	
bre.utils.formatter_functions), 191	1:	bre.utils.formatter_functions), 195
BuiltinIdentifierInList (class in	cali-	
bre.utils.formatter_functions), 196		BuiltinPrint (class in calibre.utils.formatter_functions),
BuiltinIfempty (class in	cali-	202
bre.utils.formatter_functions), 204		BuiltinQueryString (class in cali-
BuiltinInList (class in	cali-	bre.utils.formatter_functions), 207
bre.utils.formatter_functions), 196		BuiltinRange (class in calibre.utils.formatter_functions),
BuiltinIsDarkMode (class in	cali-	200
bre.utils.formatter_functions), 201		BuiltinRatingToStars (class in cali-
BuiltinIsMarked (class in	cali-	bre.utils.formatter_functions), 191
bre.utils.formatter_functions), 194		BuiltinRawField (class in cali-
BuiltinLanguageCodes (class in	cali-	bre.utils.formatter_functions), 195
bre.utils.formatter_functions), 194	cun-	
· · · · · · · · · · · · · · · · · · ·	1:	
BuiltinLanguageStrings (class in	cali-	bre.utils.formatter_functions), 195
bre.utils.formatter_functions), 194		BuiltinRe (class in calibre.utils.formatter_functions), 204
BuiltinListCountMatching ( <i>class in</i>	cali-	1
bre.utils.formatter_functions), 197		bre.utils.formatter_functions), 204
BuiltinListDifference (class in	cali-	BuiltinRound (class in calibre.utils.formatter_functions),
bre.utils.formatter_functions), 198		183
BuiltinListEquals (class in	cali-	BuiltinSelect (class in cali-
bre.utils.formatter_functions), 198		bre.utils.formatter_functions), 197
BuiltinListIntersection (class in	cali-	BuiltinSeriesSort (class in cali-
bre.utils.formatter_functions), 198		bre.utils.formatter_functions), 195
BuiltinListitem ( <i>class in</i>	cali-	BuiltinSetGlobals (class in cali-
bre.utils.formatter_functions), 196	cun	bre.utils.formatter_functions), 202
BuiltinListJoin (class in	cali	BuiltinShorten (class in cali-
	cuu-	
bre.utils.formatter_functions), 198	1.	bre.utils.formatter_functions), 204
BuiltinListRe (class in	cali-	BuiltinStreat (class in cali-
bre.utils.formatter_functions), 199		bre.utils.formatter_functions), 204
BuiltinListReGroup ( <i>class in</i>	cali-	BuiltinStrcatMax (class in cali-
bre.utils.formatter_functions), 199		bre.utils.formatter_functions), 205
BuiltinListRemoveDuplicates (class in	cali-	BuiltinStrcmp (class in cali-
bre.utils.formatter_functions), 199		bre.utils.formatter_functions), 203
BuiltinListSort ( <i>class in</i>	cali-	BuiltinStrcmpcase (class in cali-
bre.utils.formatter_functions), 199		bre.utils.formatter_functions), 203
BuiltinListSplit (class in	cali-	BuiltinStrInList (class in cali-
bre.utils.formatter_functions), 199		bre.utils.formatter_functions), 197
BuiltinListUnion (class in	cali-	BuiltinStrlen (class in cali-
bre.utils.formatter_functions), 199		bre.utils.formatter_functions), 205
BuiltinLookup ( <i>class in</i>	cali-	BuiltinSubitems (class in cali-
	cun-	
bre.utils.formatter_functions), 196	ogl:	bre.utils.formatter_functions), 200
BuiltinLowercase (class in	cali-	BuiltinSublist (class in cali-
bre.utils.formatter_functions), 184	1.	bre.utils.formatter_functions), 201
BuiltinMakeUrl (class in	cali-	BuiltinSubstr (class in cali-
bre.utils.formatter_functions), 206	_	bre.utils.formatter_functions), 205
BuiltinMakeUrlExtended (class in	cali-	BuiltinSubtract (class in cali-
bre.utils.formatter_functions), 206		bre.utils.formatter_functions), 183

Index

BuiltinSwapAroundArt	icles	(class	in	cali-
bre.utils.formatter	functic	ons), 205		
BuiltinSwapAroundCom	ma	(class	in	cali-
bre.utils.formatter	functic	ons), 205		
BuiltinSwitch	(class	in		cali-
bre.utils.formatter	-functio	ons), 196		
BuiltinSwitchIf	(class		ı	cali-
bre.utils.formatter	• functio	ons), 196		
BuiltinTemplate			ı	cali-
bre.utils.formatter				
BuiltinTest (class in ca	-v	<i>,</i> .	er fund	ctions).
205		2	-	
BuiltinTitlecase	(class	i i	n	cali-
bre.utils.formatter	• functio	ons), 184		
BuiltinToday (class in ca	-v	<i>,</i> .	er fund	ctions).
188		2	-	
BuiltinToHex (class in ca	libre.util	s.formatt	er fund	ctions).
208		2	-	
BuiltinTransliterate	(c	lass	in	cali-
bre.utils.formatter				
	(class		n	cali-
bre.utils.formatter				
BuiltinUrlsFromIdent	•		in	cali-
bre.utils.formatter				
BuiltinUserCategorie			in	cali-
bre.utils.formatter				
BuiltinVirtualLibrar	•		in	cali-
bre.utils.formatter				
0		// -		
$\cap$				

C	
Cache ( <i>class in calibre.db.cache</i> ), 365	
Cache.EventType (class in calibre.db.cache), 365	
cached_cover_url_is_reliable	(cali-
bre.ebooks.metadata.sources.base.Source	at-
tribute), 260	
calibre command line option	
detach, 301	
help, 301	
ignore-plugins, 301	
no-update-check, 301	
shutdown-running-calibre, $301$	
start-in-tray, 302	
verbose, 302	
version, $302$	
with-library, 302	
-h, 301	
-s, 301	
-v, 302	
calibre.customize	
module, 253	
calibre.customize.conversion	
module, 263	
calibre.db.cache	
module, 365	

calibre.devices.interface
module, 266
calibre.ebooks.metadata.book.base
module, 208
calibre.ebooks.metadata.sources.base
module, 259
calibre.ebooks.oeb.polish.container
module, 375
calibre.ebooks.oeb.polish.cover
module, 381
calibre.ebooks.oeb.polish.css
module, 381
calibre.ebooks.oeb.polish.jacket
module, 380
calibre.ebooks.oeb.polish.pretty
module, 380
calibre.ebooks.oeb.polish.replace
module, 379
calibre.ebooks.oeb.polish.split
module, 380
calibre.ebooks.oeb.polish.toc
module, 382
calibre.gui2.tweak_book.boss
module, 384
calibre.utils.formatter_functions
module, 182
calibre.web.feeds.news
module, 41
module, 41
$\begin{array}{c} \texttt{module, 4l} \\ \texttt{calibre-customize command line option} \end{array}$
module, $41$ calibre-customize command line option $-\text{-}\text{add-plugin},302$
<pre>module, 41 calibre-customize command line option    add-plugin, 302    build-plugin, 302</pre>
<pre>module,41 calibre-customize command line optionadd-plugin,302build-plugin,302customize-plugin,302</pre>
<pre>module, 41 calibre-customize command line optionadd-plugin, 302build-plugin, 302customize-plugin, 302disable-plugin, 302</pre>
<pre>module, 41 calibre-customize command line optionadd-plugin, 302build-plugin, 302customize-plugin, 302disable-plugin, 302enable-plugin, 302</pre>
<pre>module, 41 calibre-customize command line option    add-plugin, 302    build-plugin, 302    customize-plugin, 302    disable-plugin, 302    enable-plugin, 302    help, 302    list-plugins, 302</pre>
<pre>module, 41 calibre-customize command line option    add-plugin, 302    build-plugin, 302    customize-plugin, 302    disable-plugin, 302    enable-plugin, 302    help, 302    list-plugins, 302    remove-plugin, 302</pre>
<pre>module, 41 calibre-customize command line option    add-plugin, 302    build-plugin, 302    customize-plugin, 302    disable-plugin, 302    enable-plugin, 302    help, 302    list-plugins, 302    remove-plugin, 302    remove-plugin, 302    version, 302</pre>
<pre>module, 41 calibre-customize command line option    add-plugin, 302    build-plugin, 302    customize-plugin, 302    disable-plugin, 302    enable-plugin, 302    help, 302    list-plugins, 302    remove-plugin, 302    version, 302     -a, 302</pre>
<pre>module, 41 calibre-customize command line optionadd-plugin, 302build-plugin, 302customize-plugin, 302disable-plugin, 302enable-plugin, 302help, 302list-plugins, 302remove-plugin, 302version, 302 -a, 302 -b, 302</pre>
<pre>module, 41 calibre-customize command line option    add-plugin, 302    build-plugin, 302    customize-plugin, 302    disable-plugin, 302    enable-plugin, 302    help, 302    list-plugins, 302    remove-plugin, 302    version, 302     -a, 302     -b, 302     -h, 302</pre>
<pre>module, 41 calibre-customize command line option    add-plugin, 302    build-plugin, 302    customize-plugin, 302    disable-plugin, 302    help, 302    help, 302    list-plugins, 302    remove-plugin, 302    version, 302     -a, 302     -b, 302     -h, 302     -l, 302</pre>
<pre>module, 41 calibre-customize command line option    add-plugin, 302    build-plugin, 302    customize-plugin, 302    disable-plugin, 302    enable-plugin, 302    help, 302    list-plugins, 302    remove-plugin, 302    version, 302     -a, 302     -b, 302     -h, 302     -1, 302     -r, 302</pre>
<pre>module, 41 calibre-customize command line option    add-plugin, 302    build-plugin, 302    customize-plugin, 302    disable-plugin, 302    enable-plugin, 302    help, 302    list-plugins, 302    remove-plugin, 302    version, 302     -a, 302     -b, 302     -h, 302     -l, 302     calibre-debug command line option</pre>
<pre>module, 41 calibre-customize command line option    add-plugin, 302    build-plugin, 302    customize-plugin, 302    disable-plugin, 302    enable-plugin, 302    help, 302    list-plugins, 302    remove-plugin, 302    version, 302     -a, 302     -b, 302     -h, 302     -l, 302 calibre-debug command line option    add-simple-plugin, 303</pre>
<pre>module, 41 calibre-customize command line option    add-plugin, 302    build-plugin, 302    customize-plugin, 302    disable-plugin, 302    enable-plugin, 302    help, 302    list-plugins, 302    remove-plugin, 302    version, 302     -a, 302     -b, 302     -h, 302     -1, 302     -r, 302 calibre-debug command line option    add-simple-plugin, 303    command, 303</pre>
<pre>module, 41 calibre-customize command line option    add-plugin, 302    build-plugin, 302    customize-plugin, 302    disable-plugin, 302    help, 302    help, 302    list-plugins, 302    version, 302     -a, 302     -b, 302     -h, 302     -l, 302     calibre-debug command line option    add-simple-plugin, 303    command, 303    debug-device-driver, 303</pre>
<pre>module, 41 calibre-customize command line option    add-plugin, 302    build-plugin, 302    customize-plugin, 302    disable-plugin, 302    help, 302    help, 302    list-plugins, 302    version, 302     -a, 302     -b, 302     -h, 302     -l, 302     calibre-debug command line option    add-simple-plugin, 303    command, 303    debug-device-driver, 303    default-programs, 303</pre>
<pre>module, 41 calibre-customize command line option    add-plugin, 302    build-plugin, 302    customize-plugin, 302    disable-plugin, 302    enable-plugin, 302    help, 302    help, 302    remove-plugin, 302    version, 302     -a, 302     -b, 302     -h, 302     -1, 302     -r, 302 calibre-debug command line option    add-simple-plugin, 303    command, 303    debug-device-driver, 303    default-programs, 303    diff, 303</pre>
<pre>module, 41 calibre-customize command line option    add-plugin, 302    build-plugin, 302    customize-plugin, 302    disable-plugin, 302    enable-plugin, 302    help, 302    list-plugins, 302    remove-plugin, 302    version, 302     -a, 302     -b, 302     -h, 302     -l, 302     calibre-debug command line option    add-simple-plugin, 303    command, 303    default-programs, 303    diff, 303    edit-book, 303</pre>
<pre>module, 41 calibre-customize command line option    add-plugin, 302    build-plugin, 302    customize-plugin, 302    disable-plugin, 302    enable-plugin, 302    help, 302    list-plugins, 302    remove-plugin, 302    version, 302     -a, 302     -b, 302     -h, 302     -l, 302     calibre-debug command line option    add-simple-plugin, 303    command, 303    debug-device-driver, 303    diff, 303    edit-book, 303    exec-file, 303</pre>
<pre>module, 41 calibre-customize command line option    add-plugin, 302    build-plugin, 302    customize-plugin, 302    disable-plugin, 302    enable-plugin, 302    help, 302    list-plugins, 302    remove-plugin, 302    version, 302     -a, 302     -b, 302     -h, 302     -l, 302     calibre-debug command line option    add-simple-plugin, 303    command, 303    debug-device-driver, 303    diff, 303    edit-book, 303    explode-book, 303 </pre>
<pre>module, 41 calibre-customize command line option    add-plugin, 302    build-plugin, 302    customize-plugin, 302    disable-plugin, 302    enable-plugin, 302    help, 302    list-plugins, 302    remove-plugin, 302    version, 302     -a, 302     -b, 302     -h, 302     -l, 302     calibre-debug command line option    add-simple-plugin, 303    command, 303    debug-device-driver, 303    diff, 303    edit-book, 303    exec-file, 303</pre>

--qui-debug, 304 --help, 304 --implode-book, 304 --import-calibre-data, 304 --inspect-mobi, 304 --kepubify, 304 --paths, 304 --run-plugin, 304 --run-test, 304 --run-without-debug, 304 --shutdown-running-calibre, 304 --subset-font, 304 --test-build, 304 --un-kepubify, 304 --version, 304--viewer, 304 -c, 303 -d, 303 -e.303 -f, 304 -q, 303 -h, 304 -i.304 -m. 304 -r.304 -s.304 -t.304 -w, 304 -x, 303 calibre-server command line option --access-log, 305 --ajax-timeout, 305 --auth-mode, 305 --auto-reload, 305 --ban-after, 305 --ban-for.305 --book-list-mode, 305 --compress-min-size, 305 --custom-list-template, 305 --daemonize, 305 --disable-allow-socket-preallocation, 305 --disable-auth. 306 --disable-fallback-to-detectedinterface, 306 --disable-local-write, 306 --disable-log-not-found, 306 --disable-use-bonjour, 306 --disable-use-sendfile, 306 --displayed-fields, 305 --enable-allow-socket-preallocation, 305 --enable-auth, 306 --enable-fallback-to-detectedinterface, 306

--enable-local-write, 306 --enable-log-not-found, 306 --enable-use-bonjour, 306 --enable-use-sendfile, 306--help, 306 --ignored-fields, 306--listen-on.306 --log, 306 --manage-users, 306 --max-header-line-size, 306 --max-job-time, 306 --max-jobs, 307 --max-log-size, 307 --max-opds-items, 307 --max-opds-ungrouped-items, 307 --max-request-body-size, 307 --num-per-page, 307 --pidfile, 307 --port, 307 --search-the-net-urls, 307 --shutdown-timeout, 307 --ssl-certfile, 307 --ssl-keyfile, 307 --timeout. 307 --trusted-ips, 307 --url-prefix, 307 --userdb, 308 --version, 308 --worker-count, 308 -h, 306 calibre-smtp command line option --attachment, 309 --cafile, 309 --dont-verify-server-certificate, 309 --encryption-method, 309 --fork, 308 --help, 308 --localhost, 308--outbox, 308 --password, 309 --port, 309 --relay, 309 --subject, 309 --timeout, 308 --username, 309 --verbose, 308 --version, 308 -a, 309 -e, 309 -f, 308 -h. 308 -1,308-0.308

-p, 309

-r.309 -s, 309 -t,308 -u, 309 -v, 308 calibredb-add command line option -1.313-I, 312 -s, 313 -т, 313 --add, 313 --authors, 312 --automerge, 312 --cover, 312 --duplicates, 312 --empty, 312 --identifier, 312 --ignore, 313 --isbn, 312 --languages, 312 --one-book-per-directory, 313 --recurse, 313 --series, 312 --series-index, 313 --tags, 313 --title, 313 -a, 312 -c, 312 -d, 312 -e, 312 -i,312 -1,312 -m, 312 -r.313 -s, 312 -t,313 calibredb-add\_custom\_column command line calibredb-export command line option option --display, 318 --is-multiple, 318 calibredb-add\_format command line option --as-extra-data-file, 313 --dont-replace, 314 calibredb-backup\_metadata command line option --all, 320 calibredb-catalog command line option --catalog-title, 316 --cross-reference-authors, 316 --debug-pipeline, 316 --exclude-genre, 316 --exclusion-rules, 316 --generate-authors, 316

```
--generate-genres, 316
    --generate-recently-added, 316
    --generate-series, 317
    --generate-titles, 317
    --genre-source-field, 317
    --header-note-source-field, 317
    --ids. 316
    --merge-comments-rule, 317
    --output-profile, 317
    --prefix-rules, 317
    --preset, 317
    --search, 316
    --thumb-width, 317
    --use-existing-cover, 317
    --verbose, 316
    -i,316
    -s, 316
    -v. 316
calibredb-check_library command line option
    --csv, 319
    --ignore_extensions, 319
    --ignore_names, 319
    --report, 319
    --vacuum-fts-db. 319
    -c, 319
    -e. 319
    -n, 319
    -r.319
calibredb-custom_columns command line op-
        tion
    --details, 318
    -d. 318
calibredb-embed_metadata command line op-
        tion
    --only-formats, 321
    -f.321
    --all, 315
    --dont-asciiize, 315
    --dont-save-cover, 315
    --dont-save-extra-files, 315
    --dont-update-metadata, 315
    --dont-write-opf, 315
    --formats, 315
    --progress, 315
    --replace-whitespace, 315
    --single-dir, 315
    --template, 315
    --timefmt, 315
    --to-dir, 315
    --to-lowercase, 315
calibredb-fts_index command line option
    --indexing-speed, 321
    --wait-for-completion, 321
```

--generate-descriptions, 316

```
calibredb-fts_search command line option
    --do-not-match-on-related-words, 322
    --include-snippets, 322
    --indexing-threshold, 322
    --match-end-marker, 322
    --match-start-marker, 322
    --output-format, 322
    --restrict-to, 322
calibredb-list command line option
   --ascending, 311
    --fields, 311
    --for-machine, 311
   --limit, 311
   --line-width, 311
    --prefix, 311
    --search, 311
   --separator, 311
    --sort-by, 312
    --template, 312
    --template_file, 312
    --template_heading, 312
    -f,311
    -s,311
    -t,312
    -w, 311
calibredb-list_categories command line op-
       tion
    --categories, 320
    --csv, 320
    --dialect, 320
    --item_count, 320
    --width, 320
    -c, 320
    -i,320
    -r, 320
    -w, 320
calibredb-remove command line option
    --permanent, 313
calibredb-remove_custom_column command line
       option
    --force, 318
    -f, 318
calibredb-restore_database command line op-
       tion
    --really-do-it, 319
    -r, 319
calibredb-search command line option
    --limit, 321
    -1,321
calibredb-set_custom command line option
    --append, 319
    -a, 319
calibredb-set_metadata command line option
    --field, 314
```

1	ist-fields,314	
-f,	314	
-1,	314	
	db-show_metadata command line s-opf,314	e option
	disabled	(cali-
cun_be_	bre.customize.conversion.InputFormatl	
	attribute), 263	lugui
can he	disabled	(cali-
can_be_	bre.customize.conversion.OutputForma	
	attribute), 264	ii iagin
can he	disabled	(cali-
can_bc_	bre.customize.InterfaceActionBase	attribute),
	282	auribuic),
can_be_	disabled ( <i>calibre.customize.Plugin</i> 255	attribute),
can_be_	disabled ( <i>calibre.customize.Prefere attribute</i> ), 283	ncesPlugin
CAN_DO_	DEVICE_DB_PLUGBOARD	(cali-
	bre.devices.interface.DevicePlugin	attribute),
	266	
can_get	_multiple_covers	(cali-
	bre.ebooks.metadata.sources.base.Sour	ce at-
	tribute), 260	
can_han	dle() (calibre.devices.interface.De method), 268	evicePlugin
can_han	dle_windows()	(cali-
_	bre.devices.interface.DevicePlugin	method),
can han	dle_windows()	(cali-
04.11_11411	bre.devices.usbms.device.Device	method),
	276	
CAN_SET	_METADATA	(cali-
	bre.devices.interface.DevicePlugin 266	attribute),
CAN CDT		(agli
CAN_SEI	_METADATA bre.devices.usbms.driver.USBMS	(cali-
	277	attribute),
canonic	alize_internal_url()	(cali-
	bre.web.feeds.news.BasicNewsRecipe 42	method),
capabil	ities (calibre.ebooks.metadata.source attribute), 260	es.base.Source
card_pr	efix() (calibre.devices.interface.De method), 269	evicePlugin
card_pr		vice.Device
Cataloo	Plugin (class in calibre.customize), 25	59
	y (calibre.customize.PreferencesPlugin 283	
categor	y_order (calibre.customize.Preference tribute), 283	sPlugin at-
centor	navbar ( <i>calibre.web.feeds.news.Basici</i>	NewsRecine
CEIICET	attribute), 46	n woncerpe

change_		(in	module	cali-
	bre.ebooks	.oeb.polish.f	onts), 381	(
changed	d_signal			(cali-
2	-	references Co	onfig WidgetBas	
	284	ejel enees.ee	ingis in taget Bas	e autroute),
				<i>( 1</i> <sup>•</sup>
changed	d_signal			(cali-
	bre.gui2.pr	references.Co	onfig WidgetInte	rface
	attribute), 1	283		(
clean d	downloade	d_metadat	a()	(cali-
		_	ources.base.Sou	
			<i>urees.ouse.sou</i>	
	method), 2		1	
cleanup			eds.news.Basicl	VewsRecipe
	method), 4			
CLI (clas	ss in calibre.	devices.usbn	ıs.cli), 277	
cli_mai	in() <i>(calibr</i>	e.customize.	Plugin method)	, 256
			e.CatalogPlugir	
off_obc	259	or ciclisionni,	cuulogi lugu	( and to the ),
				(1:
clone_k	prowser()	7 D		(cali-
	bre.web.fe	eds.news.Ba	sicNewsRecipe	method),
	42			(
close_e	editor()	(calibre.g	ui2.tweak_boo	k.boss.Boss
	method), 3			(
command	d line op			
		01011		
	elp, 310	210		(
	ibrary-pa			(
p	assword,	311		
t	imeout, 3	11		
u	sername,	311		(
	version, $3$			
	ith-libra			
		ary, 510		(
,	310			<i>a</i> .
commit			polish.containe	r.Container
	method), 3	576		(
commit	() (cali	bre.gui2.pref	erences.Config	WidgetBase
	method), 2	285		
commit	· · ·		ces.ConfigWidg	petInterface
COMMITC	method), 2		ces.conjig mag	,ennergace
				<i>( 1</i> <sup>•</sup>
commit_		ors_to_cc		(cali-
			oss.Boss method	
commit_	_item()( <i>ca</i>	alibre.ebooks	.oeb.polish.com	tainer.Contain
	method), 3	576		
common	_options			(cali-
001111011	-	ize conversio	n.InputFormat	
			n.mpun orman	lugin
	attribute), 1	205		( / 1·
common_	_options			(cali-
	bre.custom	ize.conversic	n.OutputForma	ıtPlugin
	attribute), 1	265		(
compres	ss covers	() (calibre.	db.cache.Cach	e method).
1 - 50	367			(
				(cali
compres	ss_news_i		·	(cali - cali -
		eas.news.Ba	sicNewsRecipe	attribute),
	46			
compres	ss_news_i	mages_aut	o_size	(cali-
			sicNewsRecipe	attribute),

		4	6
 _	_	_	

46
compress_news_images_max_size (cali-
bre.web.feeds.news.BasicNewsRecipe attribute),
46
config_help_message (cali-
bre.ebooks.metadata.sources.base.Source at- tribute), 260
<pre>config_widget (calibre.customize.PreferencesPlugin at- tribute), 283</pre>
<pre>config_widget() (calibre.customize.Plugin method),</pre>
config_widget() (cali-
<i>bre.devices.interface.DevicePlugin class method</i> ), 271
config_widget() (cali-
bre.ebooks.metadata.sources.base.Source
method), 260
ConfigWidgetBase ( <i>class in calibre.gui2.preferences</i> ), 284
ConfigWidgetInterface (class in cali-
bre.gui2.preferences), 283
Container ( <i>class in calibre.ebooks.oeb.polish.container</i> ), 375
contains(), <b>158</b>
conversion_options (cali-
bre.web.feeds.news.BasicNewsRecipe attribute), 47
<pre>convert() (calibre.customize.conversion.InputFormatPlugin</pre>
method), 264
<pre>convert() (calibre.customize.conversion.OutputFormatPlugin</pre>
copy_cover_to() (calibre.db.cache.Cache method), 367
<pre>copy_format_to() (calibre.db.cache.Cache method),</pre>
core_usage (calibre.customize.conversion.InputFormatPlugin attribute), 263
cover () ( <i>calibre.db.cache.Cache method</i> ), 367
cover_margins ( <i>calibre.web.feeds.news.BasicNewsRecipe</i> attribute), 47
<pre>imereate_action() (calibre.gui2.tweak_book.plugin.Tool</pre>
create_inline_toc() (in module cali-
bre.ebooks.oeb.polish.toc), 382
create_menu_action() (cali-
bre.gui2.actions.InterfaceAction method),
281
<pre>create_widget() (calibre.customize.PreferencesPlugin</pre>
CSS, <b>389</b>
current_container (cali-
bre.gui2.tweak_book.plugin.Tool property), 383
currently_editing (cali-
bre.gui2.tweak_book.boss.Boss property),

#### 384

custom_field_keys()	(cali-	
bre.ebooks.metadata.	book.base.Metadata	(
<i>method</i> ), 209		
customization_help()	(calibre.customize.Plugin	(
<i>method</i> ), 255		
customization_help()	(cali-	(
bre.ebooks.metadata	sources.base.Source	
<i>method</i> ), 260		
		,

# D

data_for_find_identical_books()	(can-
bre.db.cache.Cache method), 368	
data_for_has_book() (calibre.db	.cache.Cache
method), 368	
<pre>debug_managed_device_detection()</pre>	(cali-
bre.devices.interface.DevicePlugin	method),
268	
deepcopy() (calibre.ebooks.metadata.book.l	vase.Metadata
<i>method</i> ), 208	
default_cover()	(cali-
bre.web.feeds.news.BasicNewsRecip	e method),
42	
delay (calibre.web.feeds.news.BasicNewsRec	ipe attribute),
47	
	.cache.Cache
method), 368	
delete_books() (calibre.devices.interface.	DevicePlugin
method), 270	
delete_books() (calibre.devices.usbms.a	lriver. USBMS
method), 279	
delete_custom_book_data() (calibre.db	.cache.Cache
method), 368	
	.cache.Cache
method), 368	Destance to Frances at Dle
<pre>description(calibre.customize.conversion.C property), 265</pre>	πιραιτοιπαιτι
description (calibre.customize.Plugin attrib	(12) $(254)$
description (calibre.customize.Fugin and description (calibre.customize.Prefe	
attribute), 283	erencesi iugin
description (calibre.devices.usbms.a	lriver USRMS
attribute), 277	
description (calibre.web.feeds.news.Basi	icNewsRecine
attribute), 47	ertemsteerpe
detect_managed_devices()	(cali-
bre.devices.interface.DevicePlugin	method),
267	,,
Device (class in calibre.devices.usbms.device)	), 274
DevicePlugin (class in calibre.devices.inter)	
dirty() (calibre.ebooks.oeb.polish.contain	
method), 376	
do_user_config() (calibre.customize.Plug	gin method),
255	

dont_add_to (calibre.gui2.actions.InterfaceAction
attribute), 280
dont_remove_from (calibre.gui2.actions.InterfaceAction
attribute), 280
<pre>download() (calibre.web.feeds.news.BasicNewsRecipe</pre>
method), 42
download_cover() (cali-
bre.ebooks.metadata.sources.base.Source
method), 262
drop_event() (calibre.gui2.actions.InterfaceAction
<i>method</i> ), 281
-

# Е

```
ebook-convert command line option
   --add-alt-text-to-img, 328
   --asciiize, 325
   --author-sort, 330
   --authors, 330
   --base-font-size, 325
   --book-producer, 330
   --change-justification, 325
   --chapter, 328
   --chapter-mark, 328
   --comments, 330
   --cover, 330
   --debug-pipeline, 331
   --disable-dehyphenate, 327
   --disable-delete-blank-paragraphs, 327
   --disable-fix-indents, 327
   --disable-font-rescaling, 325
   --disable-format-scene-breaks, 327
   --disable-italicize-common-cases, 327
   --disable-markup-chapter-headings, 327
   --disable-remove-fake-margins, 328
   --disable-renumber-headings, 327
ugin --disable-unwrap-lines, 327
   --duplicate-links-in-toc, 329
   --embed-all-fonts, 325
   --embed-font-family, 325
   --enable-heuristics, 327
   --expand-css, 325
   --extra-css, 325
   --filter-css, 325
   --font-size-mapping, 325
   --from-opf, 330
   --help, 324
   --html-unwrap-factor, 327
   --input-profile, 324
   --insert-blank-line, 325
   --insert-blank-line-size, 325
   --insert-metadata, 329
   --isbn, 330
   --keep-ligatures, 326
   --language, 330
```

--level1-toc, 329 --level2-toc, 329 --level3-toc, 329 --line-height, 326 --linearize-tables, 326 --list-recipes, 324 --margin-bottom, 326 --margin-left, 326 --margin-right, 326 --margin-top, 326 --max-toc-links, 329 --minimum-line-height, 326 --no-chapters-in-toc, 329 --output-profile, 324 --page-breaks-before, 329 --prefer-metadata-cover, 329 --pubdate, 330 --publisher, 330 --rating, 330 --read-metadata-from-opf, 330 --remove-first-image, 329 --remove-paragraph-spacing, 326 --remove-paragraph-spacing-indent-size, 326 --replace-scene-breaks, 328 --search-replace, 328 --series, 330 --series-index, 330 --smarten-punctuation, 326 --sr1-replace, 328 --sr1-search, 328 --sr2-replace, 328 --sr2-search, 328 --sr3-replace, 328 --sr3-search, 328 --start-reading-at, 329 --subset-embedded-fonts, 327 --tags, 330 --timestamp, 330 --title, 330 --title-sort, 331 --toc-filter, 329 --toc-threshold, 329 --transform-css-rules, 327 --transform-html-rules, 327 --unsmarten-punctuation, 327 --use-auto-toc, 330 --verbose, 331 --version, 324 -d, 331 -h, 324 -m, 330 -v. 331

ebook-convert-azw3-output command line option --dont-compress, 337 --extract-to, 337 --mobi-toc-at-start, 337 --no-inline-toc, 337 --prefer-author-sort, 337 --pretty-print, 337 --share-not-sync, 337 --toc-title, 337 ebook-convert-azw4-input command line option --input-encoding, 331 ebook-convert-chm-input command line option --input-encoding, 331 ebook-convert-comic-input command line option --colors, 331 --comic-image-size, 331 --despeckle, 331 --disable-trim, 331 --dont-add-comic-pages-to-toc, 331 --dont-grayscale, 331 --dont-normalize.331 --dont-sharpen, 332 --input-encoding, 332 --keep-aspect-ratio, 332 --landscape, 332 --no-process, 332 --no-sort, 332 --output-format, 332 --right2left, 332 --wide, 332 ebook-convert-djvu-input command line option --input-encoding, 332 ebook-convert-docx-input command line option --docx-inline-subsup, 332 --docx-no-cover, 332 --docx-no-pagebreaks-between-notes, 332 --input-encoding, 332 ebook-convert-docx-output command line option --docx-custom-page-size, 337 --docx-no-cover, 337 --docx-no-toc, 337 --docx-page-margin-bottom, 337 --docx-page-margin-left, 337 --docx-page-margin-right, 337 --docx-page-margin-top, 337 --docx-page-size, 338 --extract-to, 338 --preserve-cover-aspect-ratio, 338

```
--pretty-print, 338
ebook-convert-epub-input command line op-
       tion
    --input-encoding, 333
ebook-convert-epub-output command line op-
       tion
   --dont-split-on-page-breaks, 338
   --epub-flatten, 338
   --epub-inline-toc, 338
   --epub-max-image-size, 338
   --epub-toc-at-end, 338
   --epub-version, 338
   --extract-to, 338
   --flow-size, 338
   --no-default-epub-cover, 338
   --no-svg-cover, 339
   --preserve-cover-aspect-ratio, 339
   --pretty-print, 339
   --toc-title.339
ebook-convert-fb2-input command line option
   --input-encoding, 333
   --no-inline-fb2-toc, 333
ebook-convert-fb2-output command line op-
       tion
   --fb2-genre, 339
   --pretty-print, 339
   --sectionize, 339
ebook-convert-htlz-input command line op-
       tion
   --input-encoding, 333
ebook-convert-html-input command line op-
       tion
   --allow-local-files-outside-root, 333
   --breadth-first, 333
   --dont-package, 333
   --input-encoding, 333
   --max-levels, 333
ebook-convert-html-output command line op-
       tion
   --extract-to, 339
   --pretty-print, 339
   --template-css, 340
   --template-html, 340
   --template-html-index, 340
ebook-convert-htmlz-output command line op-
       tion
   --htmlz-class-style, 340
   --htmlz-css-type, 340
   --htmlz-title-filename, 340
   --pretty-print, 340
ebook-convert-kepub-output command line op-
       tion
   --dont-split-on-page-breaks, 340
```

```
--flow-size, 340
    --kepub-affect-hyphenation, 340
    --kepub-disable-hyphenation, 340
    --kepub-hyphenation-limit-lines, 340
    --kepub-hyphenation-min-chars, 340
    --kepub-hyphenation-min-chars-after, 340
    --kepub-hyphenation-min-chars-before,
        341
    --kepub-max-image-size, 341
    --kepub-prefer-justification, 341
    --pretty-print, 341
ebook-convert-lit-input command line option
    --input-encoding, 333
ebook-convert-lit-output command line op-
        tion
    --pretty-print, 341
ebook-convert-lrf-input command line option
    --input-encoding, 334
ebook-convert-lrf-output command line op-
        tion
    --enable-autorotation, 341
    --header, 341
    --header-format, 341
    --header-separation, 341
    --minimum-indent, 341
    --mono-family, 341
    --pretty-print, 341
    --render-tables-as-images, 341
    --sans-family, 341
    --serif-family, 342
    --text-size-multiplier-for-rendered-
        tables, 342
    --wordspace, 342
ebook-convert-mobi-input command line op-
        tion
    --input-encoding, 334
ebook-convert-mobi-output command line op-
        tion
    --dont-compress, 342
    --extract-to, 342
    --mobi-file-type, 342
    --mobi-ignore-margins, 342
    --mobi-keep-original-images, 342
    --mobi-toc-at-start, 342
    --no-inline-toc, 342
    --personal-doc, 342
    --prefer-author-sort, 342
    --pretty-print, 342
    --share-not-sync, 342
    --toc-title, 342
ebook-convert-odt-input command line option
    --input-encoding, 334
ebook-convert-oeb-output command line op-
        tion
```

--extract-to, 340

--pretty-print, 343 ebook-convert-pdb-input command line option --input-encoding, 334 ebook-convert-pdb-output command line option --format, 343 --inline-toc. 343 --pdb-output-encoding, 343 --pretty-print, 343 -f, 343 ebook-convert-pdf-input command line option --input-encoding, 334 --no-images, 334 --pdf-engine, 334 --pdf-footer-regex, 334 --pdf-footer-skip, 334 --pdf-header-regex, 334 --pdf-header-skip, 334 --unwrap-factor, 335 ebook-convert-pdf-output command line option --custom-size, 343 --paper-size, 343 --pdf-add-toc, 343 --pdf-default-font-size, 343 --pdf-footer-template, 343 --pdf-header-template, 343 --pdf-hyphenate, 343 --pdf-mark-links, 343 --pdf-mono-family, 343 --pdf-mono-font-size, 344 --pdf-no-cover, 344 --pdf-odd-even-offset, 344 --pdf-page-margin-bottom, 344 --pdf-page-margin-left, 344 --pdf-page-margin-right, 344 --pdf-page-margin-top, 344 --pdf-page-number-map, 344 --pdf-page-numbers, 344 --pdf-sans-family, 344 --pdf-serif-family, 344 --pdf-standard-font, 344 --pdf-use-document-margins, 344 --preserve-cover-aspect-ratio, 344 --pretty-print, 344 --toc-title, 345 --uncompressed-pdf, 345 --unit, 345 --use-profile-size, 345 -u, 345 ebook-convert-pml-input command line option --input-encoding, 335 ebook-convert-pml-output command line option

```
--full-image-depth, 345
   --inline-toc, 345
   --pml-output-encoding, 345
   --pretty-print, 345
ebook-convert-rb-input command line option
    --input-encoding, 335
ebook-convert-rb-output command line option
    --inline-toc, 345
    --pretty-print, 345
ebook-convert-recipe-input command line op-
       tion
    --dont-download-recipe, 335
   --input-encoding, 335
   --lrf, 335
   --password, 335
    --recipe-specific-option, 335
   --test, 335
    --username, 336
ebook-convert-rtf-input command line option
    --ignore-wmf, 335
   --input-encoding, 335
ebook-convert-rtf-output command line op-
       tion
    --pretty-print, 345
ebook-convert-snb-input command line option
   --input-encoding, 336
ebook-convert-snb-output command line op-
       tion
   --pretty-print, 345
   --snb-dont-indent-first-line, 345
    --snb-full-screen, 346
   --snb-hide-chapter-name, 346
   --snb-insert-empty-line, 346
   --snb-max-line-length, 346
    --snb-output-encoding, 346
ebook-convert-tcr-input command line option
   --input-encoding, 336
ebook-convert-tcr-output command line op-
       tion
   --pretty-print, 346
   --tcr-output-encoding, 346
ebook-convert-txt-input command line option
   --formatting-type, 336
   --input-encoding, 336
   --markdown-extensions, 336
   --paragraph-type, 336
   --preserve-spaces, 336
   --txt-in-remove-indents, 336
ebook-convert-txt-output command line op-
       tion
    --force-max-line-length, 346
   --inline-toc, 346
   --keep-color, 346
    --keep-image-references, 346
```

```
--keep-links, 346
                                                      -h, 349
    --max-line-length, 346
                                                      -i, 349
    --newline, 346
                                                      -k, 348
                                                      -1, 349
    --pretty-print, 347
    --txt-output-encoding, 347
                                                      -p, 349
    --txt-output-formatting, 347
                                                      -r, 349
                                                      -s. 349
    -n. 346
ebook-convert-txtz-output command line op-
                                                      -t,349
                                                  ebook-polish command line option
        tion
                                                      -н, 350
    --force-max-line-length, 347
    --inline-toc, 347
                                                      -U, 351
    --keep-color, 347
                                                      --add-soft-hyphens, 350
    --keep-image-references, 347
                                                      --compress-images, 350
    --keep-links, 347
                                                      --cover, 350
    --max-line-length, 347
                                                      --download-external-resources, 350
    --newline, 347
                                                      --embed-fonts, 350
    --pretty-print, 347
                                                      --help, 350
    --txt-output-encoding, 347
                                                      --jacket, 350
                                                      --opf, 350
    --txt-output-formatting, 347
    -n, 347
                                                      --remove-jacket, 350
ebook-edit command line option
                                                      --remove-soft-hyphens, 350
    --detach, 348
                                                      --remove-unused-css, 351
    --help, 348
                                                      --smarten-punctuation, 351
    --select-text. 348
                                                      --subset-fonts, 351
    --version, 348
                                                      --upgrade-book, 351
    -h. 348
                                                      --verbose, 351
ebook-meta command line option
                                                      --version, 351
    --author-sort, 348
                                                      -c, 350
    --authors, 348
                                                      -d, 350
    --book-producer, 348
                                                      -e, 350
                                                      -f.351
    --category, 348
    --comments, 348
                                                      -h, 350
    --cover, 349
                                                      -i,350
    --date, 349
                                                      -j, 350
    --disallow-rendered-cover, 349
                                                      -0,350
    --from-opf, 349
                                                      -p, 351
    --get-cover, 349
                                                      -u, 351
    --help, 349
                                                  ebook-viewer command line option
    --identifier, 349
                                                      --continue, 351
                                                      --detach, 351
    --index, 349
    --isbn, 349
                                                      --force-reload, 351
    --language, 349
                                                      --full-screen, 351
    --lrf-bookid. 349
                                                      --fullscreen, 351
    --publisher, 349
                                                      --help, 351
    --rating, 349
                                                      --new-instance, 351
    --series, 349
                                                      --open-at, 352
    --tags, 349
                                                      --raise-window, 352
    --title, 349
                                                      --version, 352
    --title-sort, 349
                                                      -f.351
    --to-opf, 349
                                                      -h, 351
    --version, 350
                                                                      (calibre.gui2.tweak_book.boss.Boss
                                                  edit_file()
    -a, 348
                                                          method), 384
    -c. 348
                                                  eject() (calibre.devices.interface.DevicePlugin method),
    -d, 349
                                                          269
```

(cali-

(cali-

cali-

(cali-

method),

file\_types (calibre.customize.MetadataReaderPlugin at-

file\_types (calibre.customize.MetadataWriterPlugin at-

filesize() (calibre.ebooks.oeb.polish.container.Container

filter\_css() (in module calibre.ebooks.oeb.polish.css),

bre.web.feeds.news.BasicNewsRecipe attribute),

bre.devices.usbms.device.Device

FileTypePlugin (class in calibre.customize), 256

256

filename\_callback()

277

382

47

filter\_regexps

tribute), 258

tribute), 258

method), 376

- eject() (calibre.devices.usbms.device.Device method), file\_types (calibre.customize.FileTypePlugin attribute), 277
- embed\_metadata() (calibre.db.cache.Cache method), 368
- encoding (calibre.web.feeds.news.BasicNewsRecipe attribute), 47
- exists() (calibre.ebooks.oeb.polish.container.Container method), 376
- expire\_old\_trash() (calibre.db.cache.Cache method), 368
- export\_note() (calibre.db.cache.Cache method), 368
- extra\_css (calibre.web.feeds.news.BasicNewsRecipe attribute), 47
- extract\_readable\_article() (calibre.web.feeds.news.BasicNewsRecipe method), 42

#### F

(calibre.db.cache.Cache find identical books() method), 369 fast\_field\_for() (calibre.db.cache.Cache method), 368 fix\_all\_html() (in module bre.ebooks.oeb.polish.pretty), 380 feeds (calibre.web.feeds.news.BasicNewsRecipe attribute), fix\_html() (in module calibre.ebooks.oeb.polish.pretty), 47 380 fetch-ebook-metadata command line option for\_viewer(calibre.customize.conversion.InputFormatPlugin -I.352 --allowed-plugin, 352 attribute), 263 --authors, 352 format() (calibre.db.cache.Cache method), 369 format\_abspath() (calibre.db.cache.Cache method), --cover, 352 369 --help, 352 (calibre.db.cache.Cache.EventType --identifier, 352 format\_added attribute), 365 --isbn, 352 --opf, 352 format\_field() bre.ebooks.metadata.book.base.Metadata --timeout, 352 method), 209 --title, 352 format\_hash() (calibre.db.cache.Cache method), 369 --verbose, 353 format\_metadata() (calibre.db.cache.Cache method), --version, 353 -a, 352 369 -c. 352 FORMATS (calibre.devices.interface.DevicePlugin attribute), -d, 352 266 FORMATS (calibre.devices.usbms.driver.USBMS attribute), -h, 352 277 -i, 352 formats() (calibre.db.cache.Cache method), 369 -0,352 (calibre.db.cache.Cache.EventType -p, 352 formats\_removed -t,352 attribute), 365 (calibre.devices.interface.DevicePlugin -v, 353 free\_space() method), 269 field\_for() (calibre.db.cache.Cache method), 368 (calibre.devices.usbms.device.Device field\_ids\_for() (calibre.db.cache.Cache method), 368 free\_space() method), 276 (calibre.db.cache.Cache field\_supports\_notes() method), 368 from\_files() (in module calibre.ebooks.oeb.polish.toc), 382 file\_type (calibre.customize.conversion.OutputFormatPlugin from\_links() (in module calibre.ebooks.oeb.polish.toc), attribute), 265 382 file\_types (calibre.customize.CatalogPlugin attribute), 259 from\_xpaths() (in module bre.ebooks.oeb.polish.toc), 382 file\_types (calibre.customize.conversion.InputFormatPlugin attribute), 263

cali-

# G

generat	e_item()	(cali-
	bre.ebooks.oeb.polish.container.Container	
	<i>method</i> ), 376	
genesis	() (calibre.gui2.actions.InterfaceAction me	thod),
	282	
genesis	() (calibre.gui2.preferences.ConfigWidgetIn	terface
2	<i>method</i> ), 284	5
det all	_items_that_have_notes()	(cali-
gee_arr	bre.db.cache.Cache method), 369	(cun
got all	_link_maps_for_book()	(cali-
yet_all	<i>bre.db.cache.Cache method</i> ), 369	(cun-
		(agl:
get_all	_standard_metadata()	(cali-
	bre.ebooks.metadata.book.base.Metadata	
	<i>method</i> ), 209	
get_all	_user_metadata()	(cali-
	bre.ebooks.metadata.book.base.Metadata	
	<i>method</i> ), 209	
get_ann	otations()	(cali-
	bre.devices.usbms.device.Device me	thod),
	277	
get_art	icle_url()	(cali-
-	bre.web.feeds.news.BasicNewsRecipe me	thod),
	42	
get aut	hor_tokens()	(cali-
gee_aae	bre.ebooks.metadata.sources.base.Source	(cun
	method), 261	
get_boo		(cali-
get_boo		(cuii-
	bre.ebooks.metadata.sources.base.Source	
	method), 261	<i>(</i> <b>1</b> <sup>•</sup>
get_boo	k_url_name()	(cali-
	bre.ebooks.metadata.sources.base.Source	
	<i>method</i> ), 261	
get_boo	k_urls()	(cali-
	bre.ebooks.metadata.sources.base.Source	
	<i>method</i> ), 261	
get_bro	wser()(calibre.web.feeds.news.BasicNews	Recipe
	method), 42	
get_cac	hed_cover_url()	(cali-
	bre.ebooks.metadata.sources.base.Source	
	<i>method</i> ), 261	
get cat	egories() (calibre.db.cache.Cache me	thod).
goo_ouo	370	,,
det col	<pre>lections() (calibre.devices.interface.Bo</pre>	okList
get_cor	method), 273	OKLISI
		(aali
yer_cov	er_url() brawab faada nawa BasiaNawaBacina ma	(cali- thod),
	v *	moa),
		C 1
get_cus	tom_book_data() ( <i>calibre.db.cache</i> .	Cache
	method), 370	/ 1-
get_dev	ice_information()	(cali-
		thod),
	269	
get_dev	ice_information()	(cali-

bre.devices.u 277	sbms.driver.	USBMS	<i>method</i> ),
<pre>get_device_uid()</pre>			(cali-
	nterface.Devi	icePlugin	method),
get_driveinfo()			(cali-
-	nterface.Devi	icePlugin	method),
get_extra_css()			(cali-
-	ls.news.Basic	NewsRecipe	method),
get_feeds() ( <i>calil</i> <i>method</i> ), 43	bre.web.feeds	.news.BasicN	ewsRecipe
get_file() (c method),27		s.interface.De	vicePlugin
get_file_path_fo:	r_processi	ng()	(cali-
bre.ebooks.o method), 37	*	tainer.Contair	ıer
get_id_map()( <i>calib</i>	pre.db.cache.	Cache method	), 370
get_identifiers()			(cali-
bre.ebooks.n method), 20		k.base.Metada	ta
get_ids_for_cust	om_book_da	ata()	(cali-
bre.db.cache	.Cache metho	od), 370	
get_images()( <i>calib</i> <i>method</i> ),26		conversion.Inp	outFormatPlugin
get_item_id() ( <i>cal</i>	ibre.db.cache	.Cache metho	d), 370
get_item_ids()( <i>cc</i>	alibre.db.cach	ne.Cache meth	od), 370
get_item_name()(	calibre.db.cad	che.Cache met	hod), 370
get_item_name_map <i>method</i> ),37	p()	(calibre.db.ca	
get_link_map() (a		ne.Cache meth	od), 370
get_masthead_tit			(cali-
		NewsRecipe	method),
get_masthead_url	()		(cali-
	ls.news.Basic	NewsRecipe	method),
get_metadata()			(cali-
	e.MetadataRe	eaderPlugin	method),
get_metadata() (ca	libre.db.cacl	he.Cache meth	od), 370
get_next_series_u <i>method</i> ),37		(calibre.db.ca	che.Cache
get_notes_resour <i>method</i> ), 37		(calibre.db.ca	che.Cache
get_obfuscated_a:	rticle()		(cali-
bre.web.feed 43	ls.news.Basic	NewsRecipe	method),
get_open_popup_me	essage()		(cali-
		cePlugin class	,
get_option() ( <i>c</i> <i>method</i> ),272		s.interface.De	vicePlugin

I

<pre>get_proxy_metadata() (calibre.db.cache.Cache method), 371</pre>
<pre>get_recommended_folders() (in module cali- bre.ebooks.oeb.polish.replace), 380</pre>
get_standard_metadata() (cali- bre.ebooks.metadata.book.base.Metadata
<pre>method), 209 get_title_tokens() (cali-</pre>
bre.ebooks.metadata.sources.base.Source method), 261
<pre>get_url_specific_delay() (cali- bre.web.feeds.news.BasicNewsRecipe method), 44</pre>
<pre>get_usage_count_by_id() (calibre.db.cache.Cache</pre>
<pre>get_user_blacklisted_devices() (cali- bre.devices.interface.DevicePlugin method), 272</pre>
get_user_metadata() (cali- bre.ebooks.metadata.book.base.Metadata method), 209
gui (calibre.gui2.tweak_book.plugin.Tool property), 383 gui_category (calibre.customize.PreferencesPlugin at- tribute), 283
<pre>gui_configuration_widget() (cali- bre.customize.conversion.InputFormatPlugin method), 264</pre>
<pre>gui_configuration_widget() (cali- bre.customize.conversion.OutputFormatPlugin method), 265</pre>
gui_layout_complete() (cali- bre.gui2.actions.InterfaceAction method), 282
gui_name ( <i>calibre.customize.PreferencesPlugin attribute</i> ), 283
guide_type_map (cali- bre.ebooks.oeb.polish.container.Container property), 377

# Н

cipe
cali-
at-
ainer
cali-

icon ( <i>cal</i>	libre.customize.PreferencesPlugin attribi	<i>ute</i> ), 283
icon (cal	libre.devices.interface.DevicePlugin attri	ibute), 266
id_from	_url()( <i>calibre.ebooks.metadata.sourd</i>	ces.base.Source
	method), 261	
identif	y()(calibre.ebooks.metadata.sources.b	pase.Source
	method), 262	
identif	y_results_keygen()	(cali-
	bre.ebooks.metadata.sources.base.Sour	ce
	method), 261	
ignore_	connected_device()	(cali-
	bre.devices.interface.DevicePlugin	method),
	272	
ignore_	duplicate_articles	(cali-
	bre.web.feeds.news.BasicNewsRecipe	attribute),
	48	
ignore_	ssl_errors	(cali-
	bre.ebooks.metadata.sources.base.Sour	ce at-
	tribute), 260	
image_u	rl_processor()	(cali-
	bre.web.feeds.news.BasicNewsRecipe	class
	method), 41	
import_	note() (calibre.db.cache.Cache metho	od), 371
	o_soup()	(cali-
	bre.web.feeds.news.BasicNewsRecipe	method),
	44	, .
indexin	g_progress_changed	(cali-
	bre.db.cache.Cache.EventType	attribute),
	365	
init()(	calibre.db.cache.Cache method), 371	
	_tab_changed()	(cali-
	bre.gui2.preferences.ConfigWidgetInter	face
	method), 284	- -
initial	ization_complete()	(cali-
	bre.gui2.actions.InterfaceAction	method),
	282	
initial	ize() (calibre.customize.CatalogPlugi	n method),
	259	
initial	ize() (calibre.customize.Plugin metho	d), 255
initial	<pre>ize() (calibre.gui2.preferences.Config</pre>	WidgetBase
	method), 285	
initial	ize() (calibre.gui2.preferences.Config	WidgetInterface
	method), 284	1:
Inputro	rmatPlugin ( <i>class in</i>	cali-
	bre.customize.conversion), 263	( agli
insert_	into_xml()	(cali-
	bre.ebooks.oeb.polish.container.Contai	ner
1	method), 377	attribute
install	ation_type ( <i>calibre.customize.Plugin</i> 254	attribute),
Interfa	ceAction (class in calibre.gui2.action	s), 279
Interfa	ceActionBase (class in calibre.custor	nize), 282
Interna	lMetadataCompareKeyGen ( <i>class</i>	in cali-
	bre.ebooks.metadata.sources.base), 262	2

is_configured() (cali-	10
bre.ebooks.metadata.sources.base.Source	
method), 260	10
is_dir (calibre.ebooks.oeb.polish.container.Container at-	
tribute), 377	
is_dynamically_controllable() (cali-	L
bre.devices.interface.DevicePlugin method),	1:
272	
is_image_collection (cali-	
bre.customize.conversion.InputFormatPlugin	
attribute), 263	
is_link_wanted() (cali-	
bre.web.feeds.news.BasicNewsRecipe method),	
44	
<pre>is_null() (calibre.ebooks.metadata.book.base.Metadata</pre>	1:
method), 208	
<pre>is_running() (calibre.devices.interface.DevicePlugin</pre>	
<i>method</i> ), 272	
is_usb_connected() (cali-	
bre.devices.interface.DevicePlugin method),	
267	
<pre>items_removed (calibre.db.cache.Cache.EventType at-</pre>	
tribute), 365	
<pre>items_renamed (calibre.db.cache.Cache.EventType at-</pre>	1:
tribute), 365	
items_with_notes_in_book() (cali-	
bre.db.cache.Cache method), 371	
<pre>iterlinks() (calibre.ebooks.oeb.polish.container.Container</pre>	er
method), 377	

# Κ

keep_only_tags	(cali-
bre.web.feeds.news.BasicNewsRecipe	attribute),
48	

# L

language (calibre.web.feeds.news.BasicNewsR	Recipe at-
tribute), 48	
library_about_to_change()	(cali-
bre.gui2.actions.InterfaceAction	method),
282	
library_changed()	(cali-
bre.gui2.actions.InterfaceAction	method),
282	
<pre>link_for() (calibre.db.cache.Cache method), 3</pre>	371
links_changed (calibre.db.cache.Cache.Even	tType at-
tribute), 365	••
<pre>list_extra_files() (calibre.db.cache.Cache</pre>	e method),
371	
load_actual_plugin()	(cali-
bre.customize.InterfaceActionBase	method),
282	<i>,</i> .
<pre>load_resources() (calibre.customize.Plugin</pre>	method),
255	

load_resources() (calibre.gui2.actions.In	terfaceAction
<i>method</i> ), 281	
location_selected()	(cali-
bre.gui2.actions.InterfaceAction	<i>method</i> ),
282	
LRF, <b>389</b>	
lrf2lrs command line option	
dont-output-resources, 353	
help, 353	
output, 353	
verbose, 353	
version, 353	
-h, 353	
-0, 353	
lrfviewer command line option	
disable-hyphenation, 353	
help, 353	
profile, 353	
verbose, 353	
version, 354	
visual-debug,354	
white-background, 354	
-h, 353	
lrs2lrf command line option	
help, 354	
lrs, 354	
output, 354	
verbose, 354	
version, 354	
-h, 354	
-0, 354	

# Μ

make_na	me_unique()	(cali
	bre.ebooks.oeb.polish.container.Container	
	<i>method</i> ), 377	
MANAGES	_DEVICE_PRESENCE	(cali
	bre.devices.interface.DevicePlugin attri	bute)
	267	
manifes	t_has_name()	(cali
	bre.ebooks.oeb.polish.container.Container	
	<i>method</i> ), 377	
manifes	t_id_map	(cali
	bre.ebooks.oeb.polish.container.Container	
	property), 377	
manifes	t_items_of_type()	(cali
	bre.ebooks.oeb.polish.container.Container	
	<i>method</i> ), 377	
manifes	t_items_with_property()	(cali
	bre.ebooks.oeb.polish.container.Container	
	<i>method</i> ), 377	
manifes	t_type_map	(cali
	bre.ebooks.oeb.polish.container.Container	
	property), 377	

module calimark as cover() (in bre.ebooks.oeb.polish.cover), 381 mark\_as\_titlepage() (in module calibre.ebooks.oeb.polish.cover), 381 masthead\_url (calibre.web.feeds.news.BasicNewsRecipe attribute), 48 match\_regexps (calibre.web.feeds.news.BasicNewsRecipe attribute), 48 (calimax\_articles\_per\_feed bre.web.feeds.news.BasicNewsRecipe attribute), 48 MAX\_PATH\_LEN (calibre.devices.usbms.device.Device attribute), 275 merge() (in module calibre.ebooks.oeb.polish.split), 381 merge\_annotations\_for\_book() (calibre.db.cache.Cache method), 371 (calibre.db.cache.Cache merge\_extra\_files() method), 372 Metadata (class in calibre.ebooks.metadata.book.base), 208metadata\_changed (calibre.db.cache.Cache.EventType attribute), 366 metadata\_for\_field() (calibre.ebooks.metadata.book.base.Metadata method), 209 MetadataReaderPlugin (class in calibre.customize), 258MetadataWriterPlugin (class in calibre.customize), 258(calibre.ebooks.oeb.polish.container.Container propmi erty), 377 minimum\_calibre\_version (calibre.customize.Plugin attribute), 254 model\_metadata() (calibre.devices.interface.DevicePlugin class method), 267model\_metadata() (calibre.devices.usbms.device.Device class method), 275 module calibre.customize, 253 calibre.customize.conversion, 263 calibre.db.cache, 365 calibre.devices.interface, 266 calibre.ebooks.metadata.book.base, 208 calibre.ebooks.metadata.sources.base, 259 calibre.ebooks.oeb.polish.container, 375 calibre.ebooks.oeb.polish.cover, 381 calibre.ebooks.oeb.polish.css, 381 calibre.ebooks.oeb.polish.jacket, 380 calibre.ebooks.oeb.polish.pretty, 380 calibre.ebooks.oeb.polish.replace, 379 calibre.ebooks.oeb.polish.split, 380 calibre.ebooks.oeb.polish.toc, 382

calibre.gui2.tweak_bo	ook.boss,384
calibre.utils.formatt	er_functions, 182
calibre.web.feeds.new	rs, 41
<pre>move_book_from_trash()</pre>	(calibre.db.cache.Cache
<i>method</i> ), 372	
<pre>move_format_from_trash()</pre>	(calibre.db.cache.Cache
<i>method</i> ), 372	
multisort() ( <i>calibre.db.cache</i> .	Cache method), 372
multisplit()( <i>in module calibr</i>	e.ebooks.oeb.polish.split)
380	
N	

name (calibre.customize.Plugin attribute), 254 name (calibre.gui2.actions.InterfaceAction attribute), 280 name (calibre.gui2.tweak\_book.plugin.Tool attribute), 382 name(), 158 (calibre.customize.PreferencesPlugin name\_order attribute), 283 name\_to\_abspath() (calibre.ebooks.oeb.polish.container.Container method), 377 (caliname\_to\_href() bre.ebooks.oeb.polish.container.Container method), 377 names\_that\_must\_not\_be\_changed (calibre.ebooks.oeb.polish.container.Container property), 377 (calinames\_that\_must\_not\_be\_removed bre.ebooks.oeb.polish.container.Container property), 378 names\_that\_need\_not\_be\_manifested (calibre.ebooks.oeb.polish.container.Container property), 378 needs subscription (calibre.web.feeds.news.BasicNewsRecipe attribute), 48 (calibre.devices.usbms.device.Device NEWS\_IN\_FOLDER attribute), 275 no\_stylesheets (calibre.web.feeds.news.BasicNewsRecipe attribute), 48 normalize\_path() (calibre.devices.usbms.driver.USBMS class method), 279 notes\_changed (calibre.db.cache.Cache.EventType attribute), 366 notes\_data\_for() (calibre.db.cache.Cache method), 372 notes\_for() (calibre.db.cache.Cache method), 372 notes\_resources\_used\_by() (calibre.db.cache.Cache *method*), 372 NUKE\_COMMENTS (calibre.devices.interface.DevicePlugin attribute), 267

# 0

oldest_	article ( bre.web.feeds.news.BasicNewsRecipe attrib 48	cali- ute),
on_impo	rt (calibre.customize.FileTypePlugin attrib 256	ute),
on_post	<pre>convert (calibre.customize.FileTypePlugin tribute), 256</pre>	at-
on_post	delete (calibre.customize.FileTypePl attribute), 256	ugin
-	<pre>import (calibre.customize.FileTypePl attribute), 256</pre>	
on_post	process (calibre.customize.FileTypePlugin tribute), 256	at-
on_prep	attribute), 256 (calibre.customize.FileTypePl	ugin
open()	(calibre.devices.interface.DevicePlugin meth 268	od),
open() (	(calibre.devices.usbms.device.Device method),	276
open ()	(calibre.ebooks.oeb.polish.container.Conto method), 378	liner
open_bo	ok() (calibre.gui2.tweak_book.boss. method), 384	Boss
OPEN_FE	EDBACK_MESSAGE ( bre.devices.interface.DevicePlugin attrib	cali- ute),
	266	
opf (cal	libre.ebooks.oeb.polish.container.Container p erty), 378	rop-
opf_get	bre.ebooks.oeb.polish.container.Container	cali-
	method), 378	
-	<pre>sion(calibre.ebooks.oeb.polish.container.Co property), 378</pre>	ntaine
		cali-
obr_ver	sion_parsed ( bre.ebooks.oeb.polish.container.Container	cuu-
	property), 378	
opf_xpa	th () (calibre.ebooks.oeb.polish.container.Co method), 378	ntaine
options	(calibre.customize.conversion.InputFormatPl attribute), 263	ugin
options	(calibre.customize.conversion.OutputFormatF attribute), 265	lugin
options	(calibre.ebooks.metadata.sources.base.Source tribute), 260	e at-
OSX_MAI		cali-
	bre.devices.usbms.device.Device attrib 275	
output_	-	cali-
	$bre.customize.conversion. {\it Input Format Plugin}$	
	attribute), 263	
OutputF		cali-
	bre.customize.conversion), 264	

## Ρ

parse_f	eeds()(calibre.web.feeds.news.BasicNewsRecipe method), 44	?
parse_i	ndex()(calibre.web.feeds.news.BasicNewsRecipe method), 44	2
parsed(	) (calibre.ebooks.oeb.polish.container.Container method), 378	
path_se	p (calibre.devices.interface.DevicePlugin at-	
-	tribute), 266 class in calibre.customize), 254	
populat	e_article_metadata() (cali- bre.web.feeds.news.BasicNewsRecipe method), 44	
popup_t	ype (calibre.gui2.actions.InterfaceAction at- tribute), 280	
post_ya	nk_cleanup() (cali- bre.devices.interface.DevicePlugin method), 269	
post_ya	nk_cleanup() (cali- bre.devices.usbms.device.Device method), 277	
postadd	<ul><li>() (calibre.customize.FileTypePlugin method),</li><li>257</li></ul>	
postcon	<pre>vert() (calibre.customize.FileTypePlugin method), 257</pre>	
postdel	ete() (calibre.customize.FileTypePlugin method), 257	
postimp	ort() (calibre.customize.FileTypePlugin method), 257	
postpro	cess_book() (cali- bre.customize.conversion.InputFormatPlugin	
r	method), 264	
postpro	cess_book() (cali- bre.web.feeds.news.BasicNewsRecipe method), 45	
postpro	cess_html() (cali-	
er -	bre.web.feeds.news.BasicNewsRecipe method), 45	
pref() (	calibre.db.cache.Cache method), 372	
prefer_	results_with_isbn (cali- bre.ebooks.metadata.sources.base.Source at-	
	tribute), 260	
	ncesPlugin ( <i>class in calibre.customize</i> ), 283	
prepare	_addable_books() (cali-	
	bre.devices.interface.DevicePlugin method), 271	
preproc	ess_html() (cali-	
	bre.web.feeds.news.BasicNewsRecipe method), 45	
preproc	ess_image() (cali-	
	bre.web.feeds.news.BasicNewsRecipe method), 45	
preproc	ess_raw_html() (cali- bre.web.feeds.news.BasicNewsRecipe method),	

45			
preprocess_reg	lexps		(cali-
bre.web.j	feeds.news.Ba	sicNewsRecipe	attribute),
49			
<pre>pretty_all()</pre>	(in	module	cali-
bre.ebool	ks.oeb.polish.p	pretty), 380	
pretty_css()	(in	module	cali-
bre.ebool	ks.oeb.polish.j	pretty), 380	
pretty_html()	(in	module	cali-
bre.ebool	ks.oeb.polish.j	pretty), 380	
pretty_xml()	(in	module	cali-
bre.ebool	ks.oeb.polish.j	pretty), 380	
print_version(	()		(cali-
bre.web.j	feeds.news.Ba	sicNewsRecipe	class
method),	41		
priority (calibre	e.customize.Pli	ugin attribute), 2	54
priority (calibre	e.gui2.actions	InterfaceAction	attribute),
280			
PRODUCT_ID	(calibre.dev	ices.interface.De	evicePlugin
attribute)	, 266		-
PRODUCT_ID (c	alibre.devices	.usbms.device.D	evice at-
tribute), 2	274		
publication_da	ite()		(cali-
-		sicNewsRecipe	method),
45		*	
publication_ty	'pe		(cali-
	-	sicNewsRecipe	attribute),
49		1	,,

# R

<pre>raw_data() (calibre.ebooks.oeb.polish.container.Container</pre>	re
method), 378	
re:test(), <b>158</b>	re
read_backup() (calibre.db.cache.Cache method), 372	re
recipe, <b>389</b>	
recipe_disabled (cali-	re
bre.web.feeds.news.BasicNewsRecipe attribute), 49	
recipe_specific_options (cali-	re
bre.web.feeds.news.BasicNewsRecipe attribute), 49	
recommendations (cali-	re
bre.customize.conversion.InputFormatPlugin attribute), 264	re
recommendations (cali-	
bre.customize.conversion.OutputFormatPlugin attribute), 265	re
recursions (calibre.web.feeds.news.BasicNewsRecipe at- tribute), 49	
<pre>refresh_gui() (calibre.gui2.preferences.ConfigWidgetInter method), 284</pre>	fca
regexp, <b>389</b>	re
<pre>register() (calibre.gui2.preferences.ConfigWidgetBase method), 285</pre>	re

register_shortcut()	(cali-
bre.gui2.tweak_book.plugin.Tool 383	method),
relpath() (calibre.ebooks.oeb.polish.contai method), 378	iner.Container
remove_attributes	(cali-
 bre.web.feeds.news.BasicNewsReci 49	
remove_book() (calibre.devices.inter method), 273	face.BookList
remove_books() (calibre.db.cache.Cache )	nethod), 372
remove_books_from_metadata()	(cali-
bre.devices.interface.DevicePlugin o 270	class method),
remove_books_from_metadata()	(cali-
bre.devices.usbms.driver.USBMS 279	method),
remove_empty_feeds	(cali-
bre.web.feeds.news.BasicNewsReci 49	
remove_extra_files() (calibre.dl method), 372	b.cache.Cache
remove_formats() ( <i>calibre.db.cache.Cae</i> 372	che method),
remove_from_spine()	(cali-
bre.ebooks.oeb.polish.container.Com method), 378	
remove_from_xml()	(cali-
bre.ebooks.oeb.polish.container.Con	ntainer
<pre>method), 378 remove_item() (calibre.ebooks.oeb.polish.c</pre>	container.Container
method), 378	4 0 272
<pre>method), 378 remove_items() (calibre.db.cache.Cache r</pre>	
<pre>method), 378 remove_items() (calibre.db.cache.Cache r remove_jacket() (in modul </pre>	
<pre>method), 378 remove_items() (calibre.db.cache.Cache r remove_jacket() (in modu bre.ebooks.oeb.polish.jacket), 380</pre>	le cali-
<pre>method), 378 remove_items() (calibre.db.cache.Cache r remove_jacket() (in modu</pre>	le cali- (cali-
<pre>method), 378 remove_items() (calibre.db.cache.Cache r remove_jacket() (in modu</pre>	le cali- (cali- pe attribute),
<pre>method), 378 remove_items() (calibre.db.cache.Cache r remove_jacket() (in modu</pre>	le cali- (cali- pe attribute), (cali-
<pre>method), 378 remove_items() (calibre.db.cache.Cache r remove_jacket() (in modu</pre>	le cali- (cali- pe attribute), (cali-
<pre>method), 378 remove_items() (calibre.db.cache.Cache r remove_jacket() (in modu bre.ebooks.oeb.polish.jacket), 380 remove_javascript bre.web.feeds.news.BasicNewsReci 49 remove_stale_user_metadata() bre.ebooks.metadata.book.base.Met method), 209 remove_tags (calibre.web.feeds.news.BasicNewsBasicN</pre>	le cali- (cali- pe attribute), (cali- tadata
<pre>method), 378 remove_items() (calibre.db.cache.Cache r remove_jacket() (in modu</pre>	le cali- (cali- pe attribute), (cali- tadata sicNewsRecipe
<pre>method), 378 remove_items() (calibre.db.cache.Cache r remove_jacket() (in modu bre.ebooks.oeb.polish.jacket), 380 remove_javascript bre.web.feeds.news.BasicNewsReci 49 remove_stale_user_metadata() bre.ebooks.metadata.book.base.Met method), 209 remove_tags (calibre.web.feeds.news.BasicNewsBasicN</pre>	le cali- (cali- pe attribute), (cali- tadata sicNewsRecipe (cali-
<pre>method), 378 remove_items() (calibre.db.cache.Cache r remove_jacket() (in modu</pre>	le cali- (cali- pe attribute), (cali- tadata sicNewsRecipe (cali- pe attribute),
<pre>method), 378 remove_items() (calibre.db.cache.Cache r remove_jacket() (in modu</pre>	le cali- (cali- pe attribute), (cali- tadata sicNewsRecipe (cali- pe attribute), (cali-
<pre>method), 378 remove_items() (calibre.db.cache.Cache r remove_jacket() (in modu</pre>	le cali- (cali- pe attribute), (cali- tadata sicNewsRecipe (cali- pe attribute), (cali- pe attribute),
<pre>method), 378 remove_items() (calibre.db.cache.Cache r remove_jacket() (in modu</pre>	le cali- (cali- pe attribute), (cali- tadata sicNewsRecipe (cali- pe attribute), (cali- pe attribute),
<pre>method), 378 remove_items() (calibre.db.cache.Cache r remove_jacket() (in modu</pre>	le cali- (cali- pe attribute), (cali- tadata sicNewsRecipe (cali- pe attribute), (cali- pe attribute),

	method), 372			
rename_	files()	(in	module	cali-
	bre.ebooks.oeb	.polish.rep	place), 379	
rename_	items() ( <i>calil</i>	bre.db.cad	che.Cache meth	hod), 373
replace	() (calibre.ebo	oks.oeb.po	olish.container	.Container
	method), 379			
replace	_links()			(cali-
	bre.ebooks.oeb	.polish.co	ntainer.Contai	ner
	method), 379			
replace	_links()	(in	module	cali-
	bre.ebooks.oeb	.polish.rep	place), 379	
require	s_version			(cali-
	bre.web.feeds.r	news.Basi	cNewsRecipe	attribute),
	50			
reset()	( <i>calibre.device</i> 268	es.interfac	e.DevicePlugir	ı method),
reset ()	(calibre.devic	es ushms.	device.Device	method),
	275			
resolve	_internal_l			(cali-
	bre.web.feeds.r	news.Basi	cNewsRecipe	attribute),
	50			/ <b>1</b> •
restart	_critical	G		(cali-
	bre.gui2.prefer	ences.Con	ifig WidgetBase	attribute),
	284			<i>(</i> 1'
restart.	_critical	C		(cali-
	bre.gui2.preferative attribute), 284	ences.Con	ifig WidgetInter	face
restore	_book() (calil	bre.db.cad	che.Cache met	hod), 373
restore	_defaults()			(cali-
	bre.gui2.prefer	ences.Con	nfig WidgetBase	e method),
	285			
restore	_defaults()			(cali-
	bre.gui2.prefer	ences.Con	fig WidgetInter	face
	<i>method</i> ), 284			
restore	_defaults_d	esc		(cali-
	bre.gui2.prefer	ences.Con	nfig WidgetInter	face
	attribute), 284			
restore	_original_f	ormat()	(calibre.db.ca	che.Cache
	method), 373			
reverse	_article_or	der		(cali-
	bre.web.feeds.r	news.Basi	cNewsRecipe	attribute),
	50			
rewind_	savepoint()			(cali-
	bre.gui2.tweak	_book.bos	ss.Boss method	!), 384
rss, <b>389</b>				
	alibre.customize			
run() ( <i>ca</i>	alibre.customize	e.FileType	Plugin method	!), 256

# S

safe\_read\_lock (calibre.db.cache.Cache property), 373
sanitize\_callback() (calibre.devices.usbms.device.Device method),
277

sanitiz	<pre>se_path_components()     bre.devices.usbms.device.Device     277</pre>	(cali- method),
save_bo	<pre>ook() (calibre.gui2.tweak_book. method), 385</pre>	boss.Boss
save_or	<pre>iginal_format() (calibre.db.cad method), 373</pre>	che.Cache
save_se	255 (calibre.customize.Plugin	method),
save_se	ettings() bre.devices.interface.DevicePlugin class 271	(cali- method),
save_se	<pre>ettings() bre.ebooks.metadata.sources.base.Sourc method), 261</pre>	(cali- ce
scale n	news_images	(cali-
_	bre.web.feeds.news.BasicNewsRecipe 50	
scale_n	<pre>wews_images_to_device bre.web.feeds.news.BasicNewsRecipe 50</pre>	(cali- attribute),
search(	() (calibre.db.cache.Cache method), 373	
search_	annotations() (calibre.db.cad method), 373	che.Cache
search_	notes() (calibre.db.cache.Cache meth	od), 373
seriali	<pre>ze_item() bre.ebooks.oeb.polish.container.Contain method), 379</pre>	(cali- ver
set_all	user_metadata() bre.ebooks.metadata.book.base.Metada method), 209	(cali- ta
set_ann	<pre>notations_for_book() bre.db.cache.Cache method), 373</pre>	(cali-
set_con	<pre>version_options() (calibre.db.cad method), 373</pre>	che.Cache
set_cov	<pre>rer() (calibre.db.cache.Cache method), rer() (in module calibre.ebooks.oeb.poli. 381</pre>	
set_dri	veinfo_name() <i>bre.devices.interface.DevicePlugin</i> 271	(cali- method),
set_dri	<pre>.veinfo_name() bre.devices.usbms.driver.USBMS 278</pre>	(cali- method),
	<pre>278 eld() (calibre.db.cache.Cache method), entifier()</pre>	374 (cali-
	bre.ebooks.metadata.book.base.Metada method), 209	
set_ide	entifiers () bre.ebooks.metadata.book.base.Metada method), 208	(cali- ta
set_lib	prary_info()	(cali-
	bre.devices.interface.DevicePlugin 272	method),

<pre>set_link_map() (calibre.db.cache.Cache method), 374</pre>	;
set_metadata() (cali-	
bre.customize.MetadataWriterPlugin method), 258	:
<pre>set_metadata() (calibre.db.cache.Cache method), 374</pre>	
<pre>set_modified() (calibre.gui2.tweak_book.boss.Boss method), 385</pre>	:
<pre>set_notes_for() (calibre.db.cache.Cache method), 374 set_option() (calibre.devices.interface.DevicePlugin</pre>	:
<pre>set_plugboards() (cali-</pre>	
bre.devices.interface.DevicePlugin method), 271	:
<pre>set_pref() (calibre.db.cache.Cache method), 374</pre>	:
<pre>set_progress_reporter() (cali-</pre>	
bre.devices.interface.DevicePlugin method), 269	:
<pre>set_progress_reporter() (cali-</pre>	:
bre.devices.usbms.device.Device method), 275	:
<pre>set_spine() (calibre.ebooks.oeb.polish.container.Containe method), 379</pre>	r
<pre>set_user_blacklisted_devices() (cali-</pre>	:
bre.devices.interface.DevicePlugin method), 272	
<pre>set_user_metadata() (cali- bre.ebooks.metadata.book.base.Metadata</pre>	
<i>method</i> ), 209	:
<pre>settings() (calibre.devices.interface.DevicePlugin class</pre>	:
show_current_diff() (cali-	
bre.gui2.tweak_book.boss.Boss method), 385	:
<pre>show_editor() (calibre.gui2.tweak_book.boss.Boss method), 385</pre>	:
<pre>shutdown() (calibre.devices.interface.DevicePlugin     method), 271</pre>	
<pre>shutting_down() (calibre.gui2.actions.InterfaceAction</pre>	:
simultaneous_downloads (cali-	
bre.web.feeds.news.BasicNewsRecipe attribute), 50	:
skip_ad_pages() (cali-	
bre.web.feeds.news.BasicNewsRecipe method), 45	:
SLOW_DRIVEINFO (calibre.devices.interface.DevicePlugin attribute), 267	:
<pre>smart_update() (cali- bre.ebooks.metadata.book.base.Metadata method) 200</pre>	
<pre>method), 209 sort_index_by() (cali-</pre>	:
<pre>sort_index_by() (cali- bre.web.feeds.news.BasicNewsRecipe method), 46</pre>	
Source (class in calibre.ebooks.metadata.sources.base), 259	

special	ize() (calibre.customize.conversion.In method), 264	putFormatPlugin
special	<pre>ize_css_for_output() bre.customize.conversion.OutputForma</pre>	(cali- tPlugin
	method), 265	
special	<pre>ize_global_preferences()</pre>	(cali-
	bre.devices.interface.DevicePlugin 272	method),
special	<pre>ize_options() bre.customize.conversion.OutputForma</pre>	(cali- tPlugin
	method), 265	
spine_i	<pre>tems (calibre.ebooks.oeb.polish.contain property), 379</pre>	ner.Container
spine_i	ter (calibre.ebooks.oeb.polish.containe property), 379	er.Container
spine_n	ames (calibre.ebooks.oeb.polish.contain	ner.Container
split()	property), 379 (in module calibre.ebooks.oeb.polish.sp	plit), 380
	f_is_multiple_composite()	(cali-
-	bre.db.cache.Cache method), 374	
split_j	obs () (calibre.ebooks.metadata.source method), 261	es.base.Source
standar	d_field_keys()	(cali-
oounuur	bre.ebooks.metadata.book.base.Metada method), 209	
STANDAR	D_METADATA_FIELDS ( <i>in modu</i> bre.ebooks.metadata.book.base), 209	ule cali-
start_p	<pre>lugin() (calibre.devices.interface.De method), 272</pre>	evicePlugin
startup	() ( <i>calibre.devices.interface.De</i> <i>method</i> ), 271	evicePlugin
stop_pl	ugin() ( <i>calibre.devices.interface.De method</i> ), 272	evicePlugin
summary	_length	(cali-
	bre.web.feeds.news.BasicNewsRecipe 50	attribute),
	ed_platforms	(cali-
	bre.customize.conversion.InputFormatl attribute), 263	Plugin
support	ed_platforms	(cali-
	bre.customize.conversion.OutputForma attribute), 264	tPlugin
support	ed_platforms	(cali-
11	bre.customize.InterfaceActionBase 282	attribute),
support	ed_platforms	(cali-
	bre.customize.MetadataReaderPlugin 258	attribute),
support	ed_platforms	(cali-
	bre.customize.MetadataWriterPlugin 258	attribute),
support	ed_platforms (calibre.custor attribute), 254	nize.Plugin
support	ed_platforms	(cali-

	bre.customize.PreferencesPlugin 283	attribute),
support	ed_platforms	(cali-
	bre.devices.usbms.driver.USBMS 277	attribute),
support	ed_platforms	(cali-
	bre.ebooks.metadata.sources.base.Sour tribute), 259	ce at-
support	s_collections()	(cali-
	bre.devices.interface.BookList method)	, 273
support	s_gzip_transfer_encoding	(cali-
	bre.ebooks.metadata.sources.base.Sour tribute), 260	ce at-
support	s_restoring_to_defaults	(cali-
	bre.gui2.preferences.ConfigWidgetBase 284	e attribute),
support	s_restoring_to_defaults	(cali-
	bre.gui2.preferences.ConfigWidgetInter attribute), 284	face
sync_bo	oklists()	(cali-
	bre.devices.interface.DevicePlugin 271	method),
sync_bo	oklists()	(cali-
	bre.devices.usbms.driver.USBMS 279	method),
sync_pr	eview_to_editor()	(cali-
	bre.gui2.tweak_book.boss.Boss method	l), 385
synchro	nize_with_db()	(cali-
	bre.devices.interface.DevicePlugin 273	method),
т		

# Т

<pre>tag_browser_context_action()</pre>	(cali-
bre.gui2.actions.InterfaceAction	method),
282	
tag_to_string()	(cali-
bre.web.feeds.news.BasicNewsRecipe	class
method), 41	
tags_older_than() (calibre.db.cache.Cach	he method),
374	
template_css (calibre.web.feeds.news.Basic	NewsRecipe
attribute), 51	
<pre>template_to_attribute()</pre>	(cali-
bre.ebooks.metadata.book.base.Meta	data
<i>method</i> ), 209	
<pre>temporary_file() (calibre.customize.Plugi</pre>	in method),
256	
<pre>test_fields() (calibre.ebooks.metadata.sou</pre>	rces.base.Sour
<i>method</i> ), 261	
THUMBNAIL_COMPRESSION_QUALITY	(cali-
bre.devices.interface.DevicePlugin	attribute),
266	
THUMBNAIL_HEIGHT	(cali-
bre.devices.interface.DevicePlugin	attribute),

266
timefmt (calibre.web.feeds.news.BasicNewsRecipe
attribute), 51
timeout (calibre.web.feeds.news.BasicNewsRecipe
attribute), 51
<pre>title(calibre.web.feeds.news.BasicNewsRecipe attribute)</pre>
51
to_html() (calibre.ebooks.metadata.book.base.Metadata method), 209
Tool (class in calibre.gui2.tweak_book.plugin), 382
toolbar_button_popup_mode (cali
bre.gui2.tweak_book.plugin.Tool attribute) 383
<pre>total_space() (calibre.devices.interface.DevicePlugin method), 269</pre>
total_space() (calibre.devices.usbms.device.Device
method), 275
touched_fields (cali
bre.ebooks.metadata.sources.base.Source at-
tribute), 260
type (calibre.customize.CatalogPlugin attribute), 259
type (calibre.customize.conversion.InputFormatPlugin at- tribute), 263
type (calibre.customize.conversion.OutputFormatPlugin at-
tribute), 264
type (calibre.customize.FileTypePlugin attribute), 256
type (calibre.customize.InterfaceActionBase attribute), 282
type (calibre.customize.MetadataReaderPlugin attribute) 258
type (calibre.customize.MetadataWriterPlugin attribute) 258
type (calibre.customize.Plugin attribute), 255
type (calibre.customize.PreferencesPlugin attribute), 283
type (calibre.devices.interface.DevicePlugin attribute), 266
type (calibre.ebooks.metadata.sources.base.Source at- tribute), 259
U
unretire_note_for() (calibre.db.cache.Cache method), 375
update_annotations() (calibre.db.cache.Cache method), 375

upload\_books() (*calibre.devices.interface.DevicePlugin method*), 270

upload\_books() (calibre.devices.usbms.driver.USBMS method), 278

upload\_cover() (calibre.devices.usbms.driver.USBMS rce method), 278

#### URL, **389** USBMS (*class in calibre.devices.usbms.driver*), 277

use\_embedded\_content (calibre.web.feeds.news.BasicNewsRecipe attribute), 51

user_categories_for_books()	(cali-
bre.db.cache.Cache method), 375	
user_feedback_after_callback	(cali-
bre.devices.interface.DevicePlugin	attribute),
267	
UserAnnotation (calibre.devices.interface.D	evicePlugin
attribute), 266	

# V

VENDOR_	ID (calibre.devices.interface.DevicePlugin	at-
	tribute), 266	
VENDOR_	ID (calibre.devices.usbms.device.Device	at-
	tribute), 274	
version	(calibre.customize.MetadataReaderPlugin	at-
	tribute), 258	
version	(calibre.customize.MetadataWriterPlugin	at-
	tribute), 258	
version	(calibre.customize.Plugin attribute), 254	
VIRTUAL	_BOOK_EXTENSION_MESSAGE	cali-
	bre.devices.interface.DevicePlugin attrib	oute),
	267	
VIRTUAL	_BOOK_EXTENSIONS	cali-
	bre.devices.interface.DevicePlugin attrib	oute),
	267	

### W

WANTS_UPDATED_THUMBNAILS bre.devices.interface.DevicePlugin	(cali- attribute),
266	
web2disk command line option	
base-dir,354	
delay, <b>354</b>	
dont-download-stylesheets, 354	
encoding, 355	
filter-regexp, 355	
help, 355	
match-regexp, 355	
max-files, 355	
max-recursions, 355	
timeout, 355	
verbose, 355	
version, 355	
-d, 354	
-h, 355	
-n, 355	
-r, 355	
-t,355	
WINDOWS_CARD_A_MEM	(cali-
bre.devices.usbms.device.Device	attribute),
275	
WINDOWS_CARD_B_MEM	(cali-
bre.devices.usbms.device.Device	attribute),
275	

WINDOWS_	_MAIN_	MEM	(calibre.devices.usbms.device.Device
	attribut	te), 27	5

windows_sort_drives()	(cali-
bre.devices.usbms.device.Device	method),
276	