`mbib` Documentation

As of March 10, 2018, the tutorial section is nearing completion, but the reference section is still missing.

`mbib` is a literature manager, with capabilities similar to `JabRef`, but intended to better cope with large databases containing thousands of references. Key features and limitations are:

- Import from and export to `BibTex`
- Import records from DOI and PubMed identifiers
- Push citations to `Texmaker/TexStudio`
- Push citations to OpenOffice.org/LibreOffice (collectively referred to as "`OOo`" below). Formatting of bibliographies in `OOo` piggybacks on `JabRef`. Thus, you need to have `JabRef` installed in order to fully use `mbib` with `OOo`.
- Written in Python3.
- Console-based GUI, based on the `urwid` and `urwidtrees` Python libraries.
- Data are stored in a `SQLite` database.
- Developed and tested only on Linux; will most likely not work out of the box on other platforms. (Volunteers for porting it to other platforms are welcome.)

1 Preliminary notes

1.1 Motivation

I started writing `mbib` after my previous literature manager (`bibus`) became unusable because of growing incompatibilities with LibreOffice, wxPython etc. I tried using `JabRef` for a while, and while it's really pretty good in many ways, it bogs down once you have several thousand references in your database. (The same goes for Zotero and Mendeley.)

My main objectives for `mbib` were

- Keep all references in a single database
- Use a proper database engine for flexibility and performance
- Work seamlessly with both LaTeX (my tool of choice) and `OOo` (which I use when I have to).

To date 2.5 of these 3 objectives have been met. Working with `OOo` is possible but a little clunky and will hopefully improve in future versions.

1.2 Development status

At present, `mbib` is alpha software, yet production-ready ;-) – it has almost all the features I need, but incompatible changes will likely happen to the code and the configuration, and they might happen to the database structure also. However, in the latter case, I will provide a script for migrating the database to the new format (which I will have to do with my own database anyway).

The program was written primarily with my own needs in mind. I work in biochemistry and use PubMed as my main online literature source, and I have therefore given interaction with that database priority. People in other fields might miss tighter integration with other databases. I'm open to adding support for those, but likely won't do so unless prodded. Similarly, some other

bits of functionality are tailored to my own personal preferences and may seem a little narrow or idiosyncratic to other users. Let me know if you need or want to contribute enhancements.

The code violates all manner of software engineering gospel. There are no unit tests, and I don't plan to add them; the doc strings are a bit spotty and not formatted for automatic conversion into API docs. That said, I will give an overview of the program structure below, which hopefully will help you find your way through the code.

2 Installation

In the following, I am going to describe how things work on *my* system. I am running Debian with a KDE desktop. I don't suppose there will be any major differences with other Linux distros or window managers, but I am not going to verify this by experiment. If you manage to get it to work on other systems and have some specific tricks to share, please let me know,[1] and I will include them here.

2.1 Prerequisites

In order to run `mbib`, you first need to install these programs and libraries:

- Bash
- Python3
- `SQLite`
- The `urwid`, `urwidtrees`, and `requests` libraries for Python3
- If you intend to use `mbib` with `OOo`, you also need the `PyUNO` bridge for Python3, as well as `JabRef`
- If you want to copy items to the X clipboard, you will need `xsel`
- For viewing or emailing PDF files, `mbib` relies on `xdg-open` and `xdg-email`

Bash is probably present on any Linux system by default. On Debian, all other prerequisites can be installed through the system's package manager. A copy of `SQLite` already comes as part of the standard library when you install Python3, but you may also want to install the `sqlite3` package, which provides the command line client that lets you run SQL statements directly on your database.

The `xdg-open` and `xdg-email` utilities may already be installed by default on your graphical Linux desktop; in Debian, they reside in the `xdg-utils` package.

2.2 Installing `mbib`

Clone (or download and unzip) the `git` repository. Add the main directory (`mbib`) to your bash `$PATH`.

2.3 Configuration

The first program start will generate a configuration `.mbib.ini` file in a default location. The available settings are explained in comments inside the file itself.[2]

[1] mpalmeruw@gmail.com

[2] You can have multiple configuration files and select one on the command line. This is explained in section ...

3 Tutorial

Here we give an overview of the general work flow. We assume that the default configuration settings are in effect. A reference section describing all program features and configuration options will follow later.

3.1 Starting the program

Assuming you have installed all prerequisites and added the `mbib` directory to your shell's `$PATH`, you should now be able to open a console window and run

```
mpalmer@rehakles:~$ mbib.sh
```

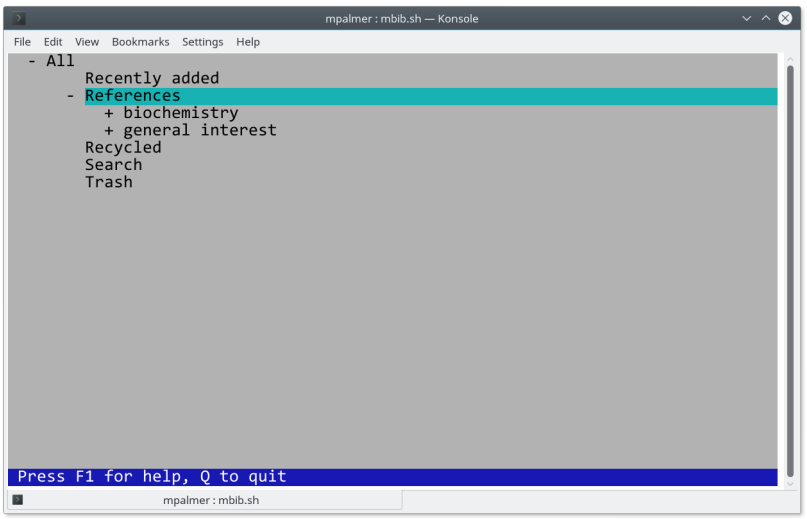
After the first start, the expected output should be

```
Config file (...)/mbib/data/mbib.ini not found!
Create default configuration file and proceed (1) or exit (2)?
1) proceed
2) exit
#?
```

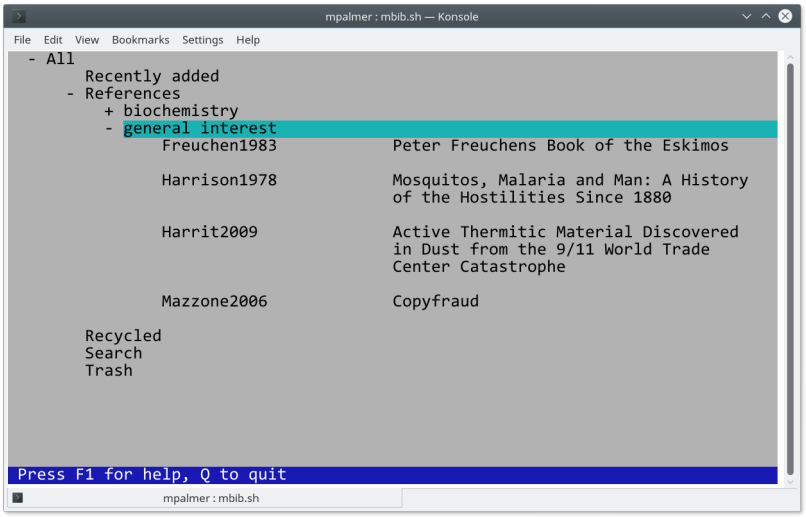
Press `1` and `Enter`, and you should see

```
#? 1
OK
Database file (...)/mbib/data/mbib.sqlite not found. Create? (y/n)
```

Press `y` and the program should start. The interface should look like this:



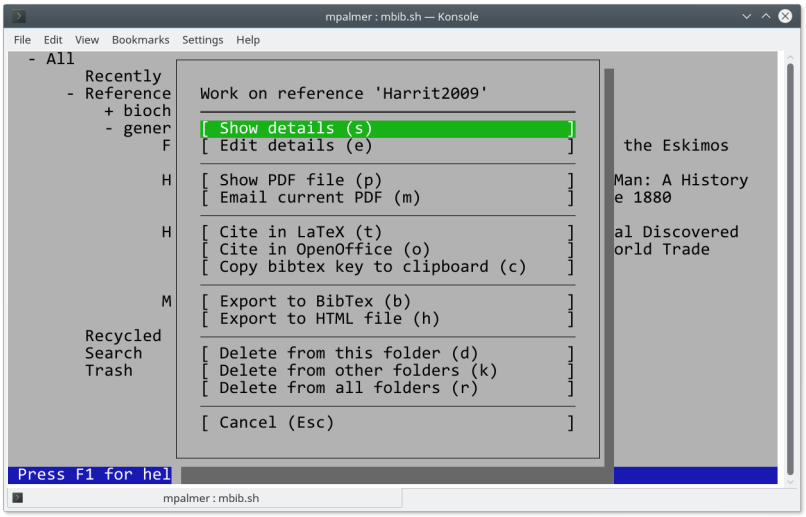
`mbib` displays all references in a tree structure. The main `References` folder at this time has two sub-folders. Move between the displayed items using the arrow keys, the mouse wheel, or click directly on the desired folder. Go to the "general interest" folder and press `F2`. You should now see the references contained in this folder:



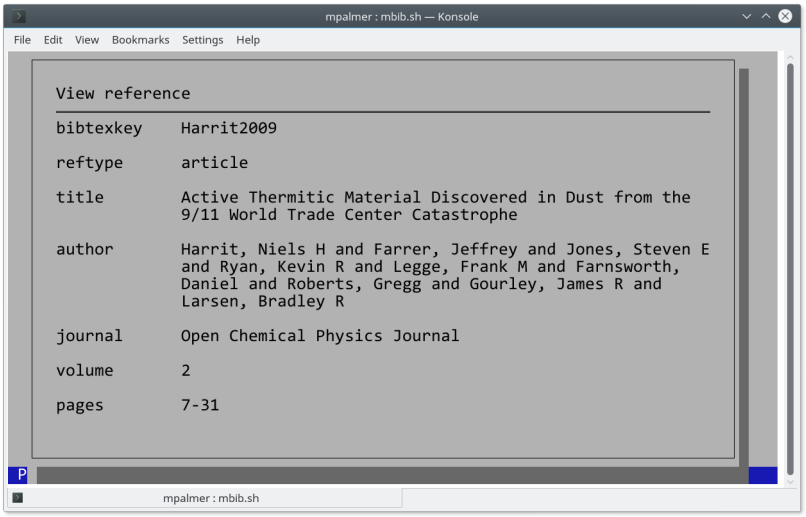
Pressing `F2` once more will close the folder again. However, for now leave it open, and use the arrow keys or the mouse again to highlight the record with the label `Harrit2009` (which by the way also functions as its `BibTex` key).

3.2 Working with references

When `Harrit2009` is highlighted, press `Enter` (or use a double mouse click), and you should see this context menu that presents all operations available for a single reference:



Navigate the menu using ↑, ↓ and select an item with `Enter`, or alternatively use the indicated shortcut keys or a single mouse click. If we select `Show details`, the `View reference` dialog comes up:

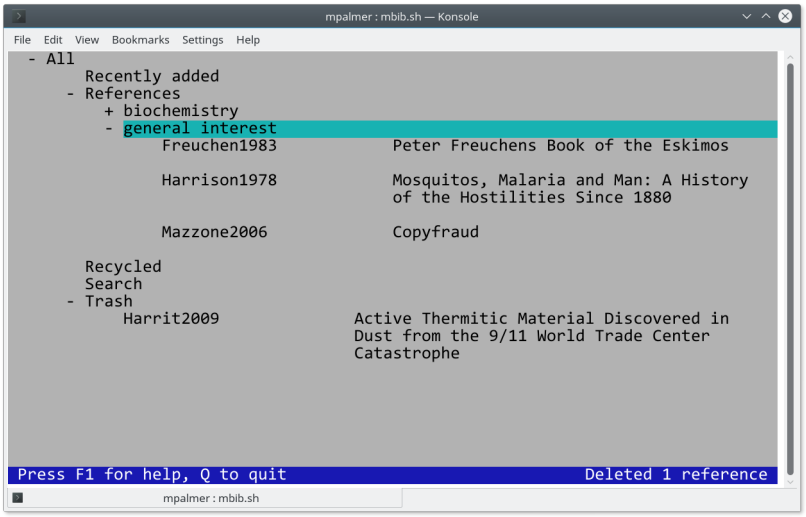


Use the ↑/↓ keys again to scroll through this dialog. The abstract is at the bottom. Above it, there are three fields that are hyperlinks. The one currently in focus is highlighted in blue; you can activate it by pressing `Enter` or again using the mouse, which will open a browser window and take you to the corresponding URL.

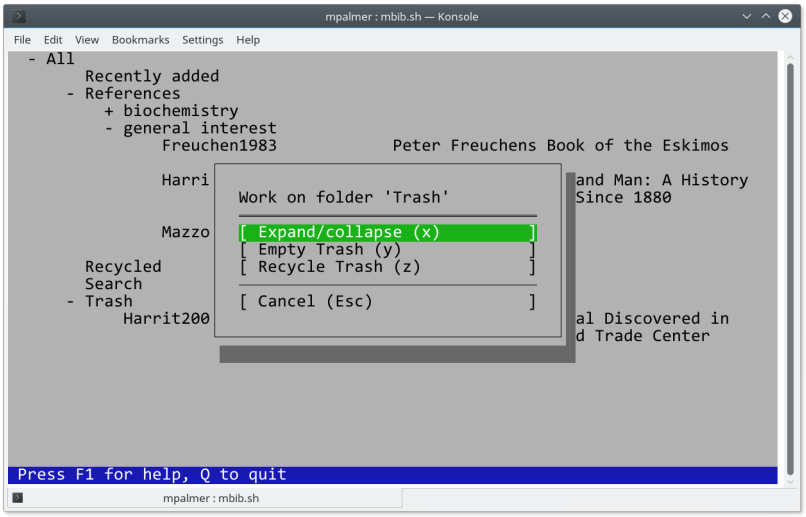
Press `Esc` to close the dialog, and then `Enter` to reopen the menu. This time, choose `Edit details`. The dialog will look similar, but now the content of all fields is editable, and fields that are currently empty are displayed as well. Make some changes—for example, add a comment: "That stuff they found in the dust must of been paint peeled off from them box cutters." Press `Tab` to switch to the buttons at the bottom; use the left and right arrow keys to select "Cancel" or "OK", and `Enter` to confirm or abort the edit.

3.3 Moving references around

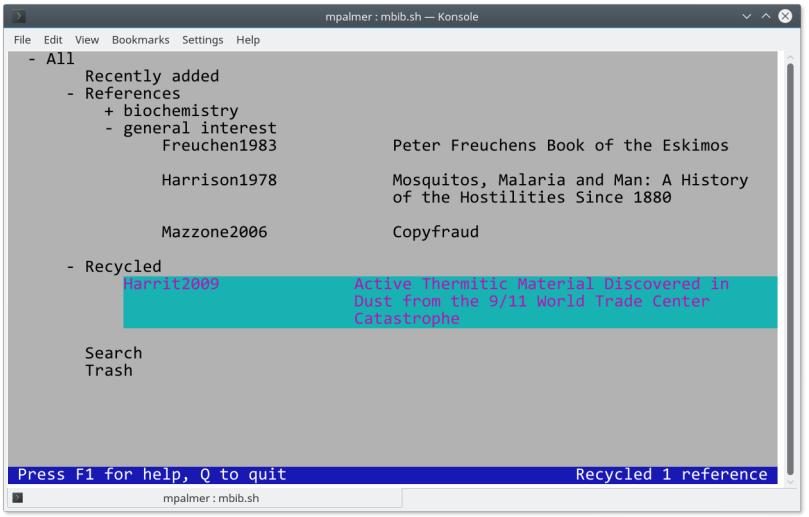
The paper we are looking at presents clear scientific evidence demonstrating that the WTC towers were rigged with explosives. This notion might disturb you, and you might therefore want to delete this reference. To do so, bring up the menu again and press `d` to delete the reference from this one folder, or `r` to also delete any copies residing in other folders (which in this case don't exist). After confirmation, the reference now shows up in the Trash folder:



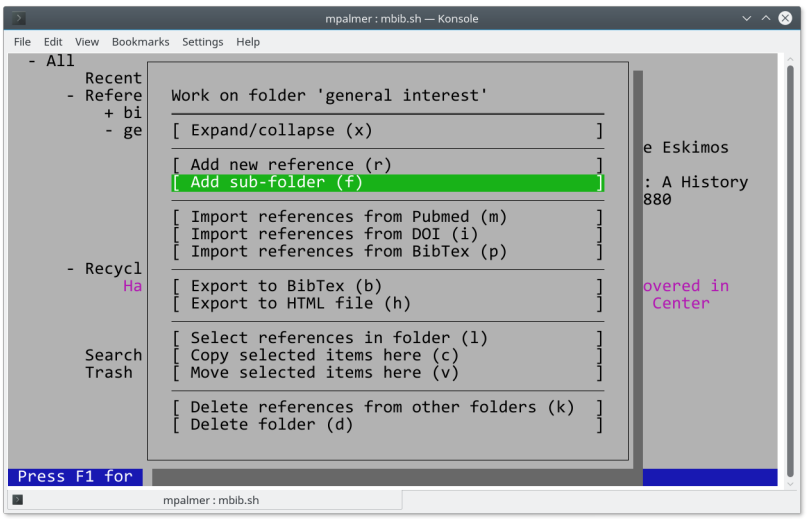
Navigate down to the Trash folder and press `Enter` or double-click on it to bring up this folder's context menu:



Selecting the "Empty Trash" option will get rid of the deleted reference entirely and irreversibly. However, for the purpose of this tutorial, use the "Recycle" option to move the deleted reference to the Recycled folder. Navigate to it and press the Space key to select it:

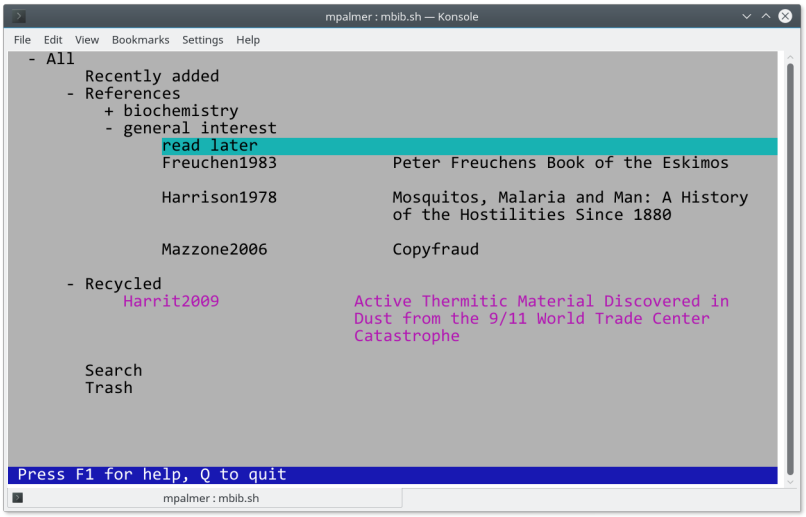


Let's assume that you plan to look at this paper some other time, and you want to collect references for later study in a separate folder. Navigate to the "general interest" folder and activate its menu (`Enter` or mouse double-click), then activate the option "Add sub-folder":[3]

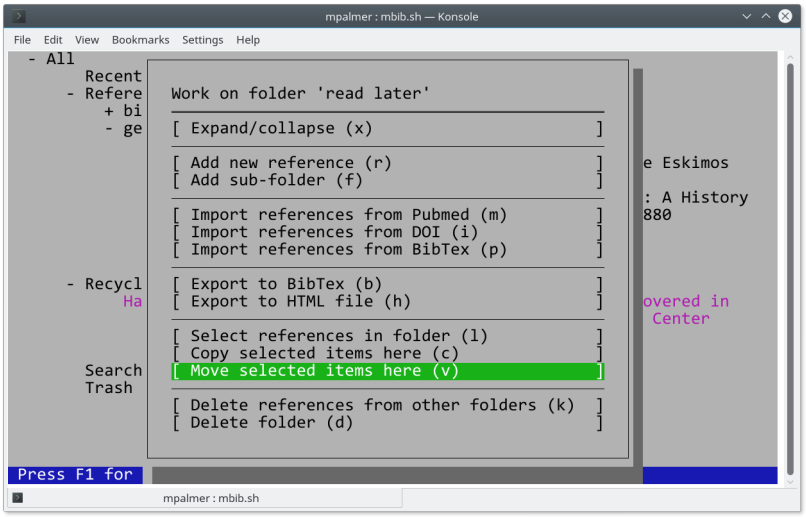


This will bring up a dialog that prompts you for the name for the new sub-folder. After entering "read later" and confirming, the display should look like this:

[3] The folder context menu is quite long. In these screen shots, which show a small window, some items at the bottom are hidden. If such should be the case on your screen also, use the ↓ key to scroll them into view.



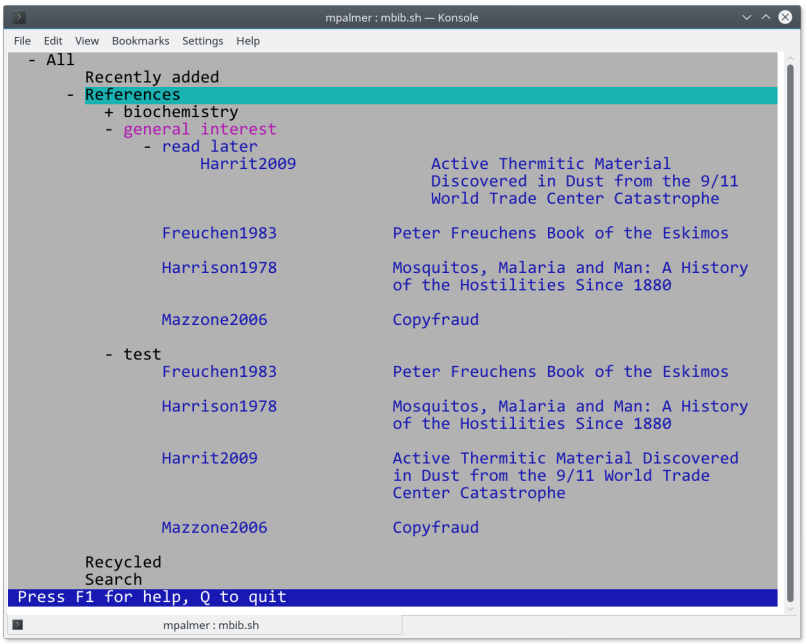
The new folder has no +/− switch before its name because it is still empty. To move the previously recycled reference into this folder, bring up the new folder's context menu and select the "Move selected items here" option:



After completing this, you should now have the recycled reference in the new folder. (Use the F2 key to display the folder's contents.)

Note that you can select more than one reference at a time; all of them will be copied or moved to a given destination. You can also select and then move or copy entire folders. As a practical exercise, create a folder named "test" directly below "References". Navigate back to "general interest" and select this folder with the space bar. Go back to "test", open its menu and select "Copy selected items here".

You will now have cloned the entire folder with all of its sub-folders and references. If you would rather have all the copied references directly underneath "test", without any sub-folders in between, open the menu for the "test" folder again and select "Flatten folder." This will give you the following result:



You will notice that, while the “general interest” folder is selected, the references in both “test” and “general interest” are shown in blue. If a reference appears in this shade, it indicates that *this or another instance* is contained in a selected folder *somewhere* in the tree—in this case, the “general interest” folder—and thus part of the currently active selection.[4] Deselecting “general interest” by hitting `Space` on it again will also deselect the references.

3.4 More on selections

3.5 Importing references

References can be imported using PubMed identifiers, DOI identifiers, and `BibTex` text. We will start with PubMed. Go to the PubMed website (`https://www.ncbi.nlm.nih.gov/pubmed/`) and type the following into the search bar:

```
blue-native[ti] anal-biochem[so]
```

This will give you (at the time of this writing) 16 results. Choose the “PMID List” option from the page’s display format control to see the identifiers for these papers. Use your mouse to select some or all of them.

In `mbib`, create a new sub-folder “blue-native” within “biochemistry”. Open its menu and choose “Import references from PubMed”. Hold down the Shift key and middle-click your mouse[5] to paste the selected PMIDs from the clipboard into the dialog. Click OK to start the import. The references will be imported one by one, and a unique `BibTex` key will be automatically generated from the first author’s last name and the year of publication.

[4] To be explicit about it: multiple instances of the same reference in different folders point to the same single instance of the reference in the database; this means that changes made to any instance will be shared by all others. That is, after all, what relational databases are good for.

[5] This behaviour is provided by the X environment, not implemented by `mbib`; it seems possible to me that some desktop environments might modify it, but I don’t know for sure.

Notice that the imported references will also show up in the "Recently added" folder at the top of the tree. This can be convenient for quicker access, since references newly imported into a folder don't float to the top; they just get sorted according to either year or BibTex key (press F7 to toggle) and thus may "get lost in the crowd" within a folder that already contains many references.

The procedure for importing references via DOI is similar to that for PubMed identifiers; if you have a list of such identifiers, separated by white space, you can just paste them. Alternatively, you can load both types of identifiers from a plain text file by specifying the file name; mbib will first treat the input as a file name, and failing that will attempt to use the input directly.

As an example for importing BibTex, go to a paper on sciencedirect (for example this one) and export the BibTex record for it. Copy it into your clipboard by mouse-selecting it, or save it to file. Open "Import references from BibTex" from the folder menu and then paste the BibTex text or give the file name.

Pasting works fine for one or a handful of references; be warned, however, that pasting longer text will be quite slow. In this case, it is better to first save the BibTex to file, and then enter the file name into the BibTex import dialogue.

3.6 Exporting references

References can be exported to BibTex or HTML. The formatting of the former can be fine-tuned through various settings in the .mbib.ini file; the latter, for the time being, cannot. Duplicates are weeded out; any folder trees are not preserved, that is, a flattened list of records is generated.

The export operations are available in the context menu of the "References" folder, in which case the entire database is exported, as well as from the menu of each sub-folder, which will export the references within and below it. Additionally, export operations can also be applied to the currently selected references (including those residing in folders). Press F5 to bring up a context menu that contains this and other operations pertaining to the current selection.

3.7 Making the mbib database available to LaTeX

Aside from exporting part or all of the mbib database to BibTex from an interactive session, you can also access the references you need from the command line. There are various ways to do this.

3.7.1 Export the references listed in the .aux file

You can generate an up-to-date BibTex file that contains all cited references (and only these[6]) on the fly by letting mbib parse the .aux file of your document. Assume that in mydocument.tex you have declared \bibliography{myproject}, which instructs BibTex to use a file named myproject.bib. Then, after latexing your document, but *before* running BibTex, you can run the following command to create or update myproject.bib:

```
mpalmer@rehakles:~$ mbib.sh -b auxexport -t mydocument.aux
```

After that, you run BibTex, again on the .aux file:

```
mpalmer@rehakles:~$ bibtex mydocument.aux
```

[6]The only trick that won't work is \nocite{*}, which is supposed to include all entries listed in the bibliography file. The best way to accommodate \nocite{*} would be to keep all references you do want to list within a single folder inside mbib and then to sync this folder on demand—see next section.

You can also ask mbib to first export and run BibTex for you:

```
mpalmer@rehakles:~$ mbib.sh -b auxbibtex -t mydocument.aux
```

If you include this line in your build process, BibTex will always transparently access the current state of your database.[7] Thanks to SQLite, this is easily fast enough even with large databases and numbers of citations, and it is how I use mbib myself with LaTeX.

3.7.2 Syncing the mbib database to a BibTex file

You can also export the entire database, or selected folders contained in it, to BibTex from the command line. In the simplest case, you can use

```
mpalmer@rehakles:~$ mbib.sh -b sync
```

This will export all references in the mbib database to a single BibTex file, the path of which is configured in the .mbib.ini file. You can override these settings by passing additional options, for example

```
mpalmer@rehakles:~$ mbib.sh -b sync -t myproject.bib -f myproject
```

Now, only the references which, inside mbib, are stored in the folder named “myproject” will be exported, and they will be saved in the file myproject.bib. This allows you to keep all your references in a single, central mbib database, and at the same time use smaller, project-specific BibTex files for your multiple separate manuscripts. A few more points to note:

- If you have multiple folders named “myproject", all of them will be exported to the same file.
- If the name of the folder contains spaces, you need to enclose in quotes on the command line:

  ```
  mpalmer@rehakles:~$ mbib.sh -b sync -t myproject.bib -f "my project"
  ```

- The mbib -b sync command will only export the database if its time stamp is more recent than that on the BibTex file (or if the latter does not yet exist, obviously).

The last point means that, regardless of file size, it costs very little to run this command routinely during document compilation. Therefore, this is another method to ensure that your document gets compiled against the current state of the mbib database.[8]

3.8 Using mbib with OOo

External programs can communicate with OOo via either sockets or TCP. mbib supports both variants. Here, I will illustrate the set-up and program I use on my own laptop. I use LibreOffice, but this should work the same with OpenOffice.org, too.

To make LibreOffice listen to connection attempts, I have configured my (KDE) desktop menu entry for LibreOffice to use the command

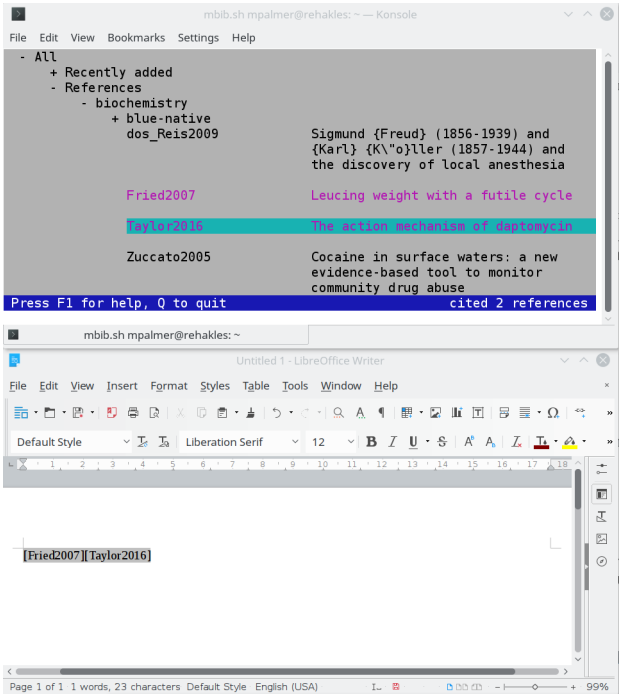
```
libreoffice  --accept="pipe,name=abraxas;urp" %U
```

[7] mbib makes no attempt to determine if your .bib file is up to date (as this would require looking at all files that you \include or \input) and just overwrites the .bib file every time.

[8] Of course, if you do the full database dump, BibTex may have to scan through a very large .bib file. While BibTex itself is very fast even with large files, other tools like biber reportedly can be noticeably slower.

The %U argument variable will be substituted by KDE with a file name to open as applicable. If you start LibreOffice from a command line, use the actual file name instead, or omit it and then open a text file through the menu.

If you left the default settings inside the `.mbib.ini` file in force, then `mbib` should now be able to insert citations into an open OOo document. Select some references (by hitting `Space` on them). Bring up the selection context menu with `F5` and select "Cite in OpenOffice." The selected references should now show up in the text document:



The above procedure inserts the citations into the document in a preliminary form; it does not generate the actual bibliography. Creating the latter takes a little extra work. First save the document to file as, say, `mydoc.odt`. Then, on the command line, invoke

```
mpalmer@rehakles:~$ mbib.sh -b jabrefy -t mydoc.odt
```

This will produce two files: `mydoc_jr.odt` and `mydoc_jr.bib`. The first will contain the same citation entries as the original `mydoc.odt`, but now formatted in a manner that `JabRef` understands. The second file contains the `BibTex` items that match the citations in the `.odt` file.

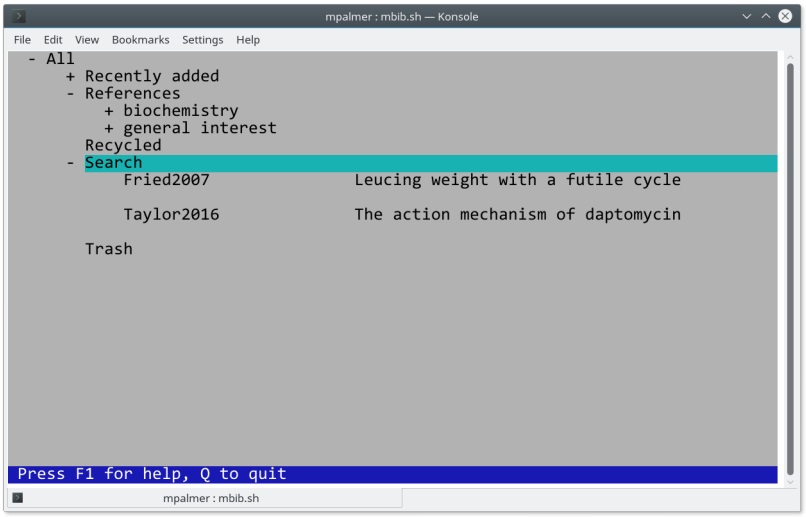
Actually formatting the bibliography requires that you open `mydocs_jr.odt` in OOo, and `mydocs_jr.bib` in `JabRef`. Connect from `JabRef` to OOo, and hit the "refresh" button. This should generate the bibliography; use `JabRef`'s facilities for defining and applying styles.

The process of inserting references and then formatting is, admittedly, a bit cumbersome. At least, it is "round-trippable"—if you insert additional references into an already `JabRef`-formatted document and then reformat it, the new references will be properly merged with the old ones.

3.9 Searching and filtering

Coming back to the interactive interface, one important task we have not yet touched upon is searching for references. Navigate to the "Search" folder, bring up its context menu, and select

"New search". This will bring up an empty search mask. Into the `abstract` field, type "synthesis", and hit `OK`. You should now see the following screen:



The search results have been appended to the "Search" folder. You can work with them the usual way—edit them, select, copy, and move them. You cannot delete them individually, because it would be inconsequential, as they are merely copies of instances stored elsewhere. However, you can use "Delete from all folders", in which case those other instances will be deleted also, and the references will then show up in the Trash.

If you press `F2` to bring up the details view for either reference in the Search Folder, you will see an "also in" entry that is a hyperlink. Following it will take you to the directory in which this record resides.

3.9.1 Multiple search terms

You can modify your search by selecting "Edit last search" from the "Search" folder's context menu. If you fill in multiple fields at the same time, all constraints will be applied in conjunction ("and"). You can also apply multiple constraints to a single field using boolean operators. For example, typing the phrase: `respiratory || cytochrome && PAGE` into the "abstract" field of the search mask will look for records that contain the words "respiratory" or "cytochrome", as well as "PAGE" in the abstract. Note that

- `||` and `&&` represent the boolean `or` and `and` operators, respectively.
- `||` takes precedence over (joins more strongly than) `&&`. While this deviates from convention in mathematics, I find it more useful in database queries; it pretty much does away with the need for using brackets at all (which `mbib` anyway doesn't support at this time). You can, however, change the precedence in the `.mbib.ini` file if you wish.

3.9.2 Finding folders

There is no proper search facility for folders. However, there are two aids for finding your way through the brushwork:

- You can filter folders by name. Press F9 to enter a word or phrase to search for. Any folder whose name contains the entered phrase, as well as its ancestors and descendants, will be displayed, while all other regular folders (those below "References") will be hidden. Press F10 to revert to the full display.
- You can also hide all references from view, displaying only the folders, using F8. (If you do so, folders that contain only references but no sub-folders will be displayed without the +/- expansion marker before their names.) This condensed view lets you explore the folder tree more speedily. Press F8 again to revert to the full display.

Finally, if you remember neither the name nor the location of the folder you are looking for, it may be useful to search for a pertinent reference instead, display its details (F2), and then follow the "Also in" link to its location in the tree (see section 3.9, page 13).

3.10 Viewing PDF files

If you select "Show PDF file" from a reference's context menu, mbib will look for a PDF file whose name matches its BibTex key; e.g. the PDF for the record Harrit2009 should be named Harrit2009.pdf. Furthermore, by default, mbib looks for PDF files only below the data/pdf/ directory that itself resides in the mbib installation directory.[9]

If mbib fails to find the requested file in the preconfigured directory, it can use the locate program[10] to find it elsewhere. This feature is disabled by default, but you can enable it in the .mbib.ini file (change option pdf_locate = false to true). This may be convenient if you prefer to store your PDF files not in a single centralized location but closer to the various projects and manuscripts you may be working on.

3.11 Multiple databases

One of my key objectives with mbib was to organize all my references within in a single database. However, if you prefer, you can have multiple databases and select the one to use from the command line:

```
mpalmer@rehakles:~$ mbib.sh -d my/wonderful/database.sqlite
```

This also works in conjunction with the various other command line operations that were described above in this tutorial.

4 Reference

This section will—some day, far, far into the future, maybe about the same time that the U.S. and Israeli governments finally come clean about 9/11—describe everything, including configuration and code structure.

5 Some gotchas

Here are some things to keep in mind.

[9] If you don't want to move your PDF files there, you can change the path in the .mbib.ini file, or you can create symlinks from inside data/pdf to your actual storage locations.

[10] On Debian, locate resides in the package of the same name. It requires the updatedb command to be run periodically in order to catch up with added or deleted files on your system.

- All changes to the data—imports, edits, deletions—are saved to the database immediately, and there is no undo function. (However, as discussed in section 3.3, deleted references can still be retrieved from the “Trash” folder.)
- On a related note: `mbib`, by default, writes a backup version of the `SQLite` file (with the appended extension `.bak`) next to the current database. (You can disable automatic backups in the `.mbib.ini` file.)

  The backup file gets overwritten at program start if it deviates from the current database file.

  Therefore, if you want to revert to the backup file, exit `mbib` and then replace the current database file with the backup file *before* restarting `mbib`.
- All the special top-level folders are special and protected just inside `mbib`, because specific operations are available and others excluded; in particular, they cannot be deleted, moved, or renamed. However, no such protective restrictions apply when working with the database using an `SQLite` shell.

  Generally speaking, it is a good idea to make a backup copy of the database file and stash it away in a safe place before performing major surgery in SQL.