Ebookbuild

Use a language such as Python, JavaScript or Go with a GUI toolkit such as Qt. Has to have libraries/or use APIs that can:

- zip files in order with no compression (so I can make an ePub)
- process text with regular expressions (to remove clutter from the documents when importing and minify the code when exporting)
- file management (to select files and folders for processing and then exporting)
- …

Might start in Python and eventually port it Go providing their is a substantial speed improvement.

Need a cool free icon-set too (such as font-awesome) to make the program more user-friendly.

The program must have minimal dependencies, be able to be ported to multiple operating systems (such as Windows, macOS and GNU/Linux).

Planned to be licensed under GNU GPLv3 or similar strict copyleft (https://www.gnu.org/licenses/gpl-3.0.en.html), although this isn't set in stone and I might consider more lenient MIT/BSD licensing instead.

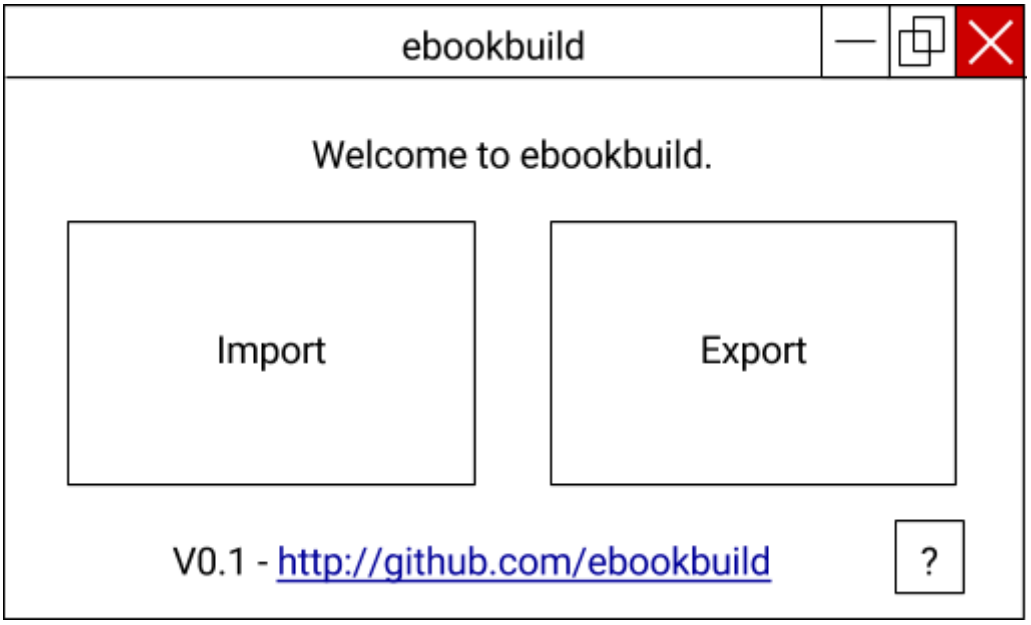
Abstract:

ebookbuild is a tool that imports book manuscript documents (such as .doc, .docx, .odt, .rtf, .tex, .pdf, .xps, .lyx, .docbook...) or using OCR technology (such as Tesseract) it takes the text from image and converts to XHTML. Also exports finished XHTML files and their CSS to a final ePub and Mobi (once I have figured out the licensing for it).

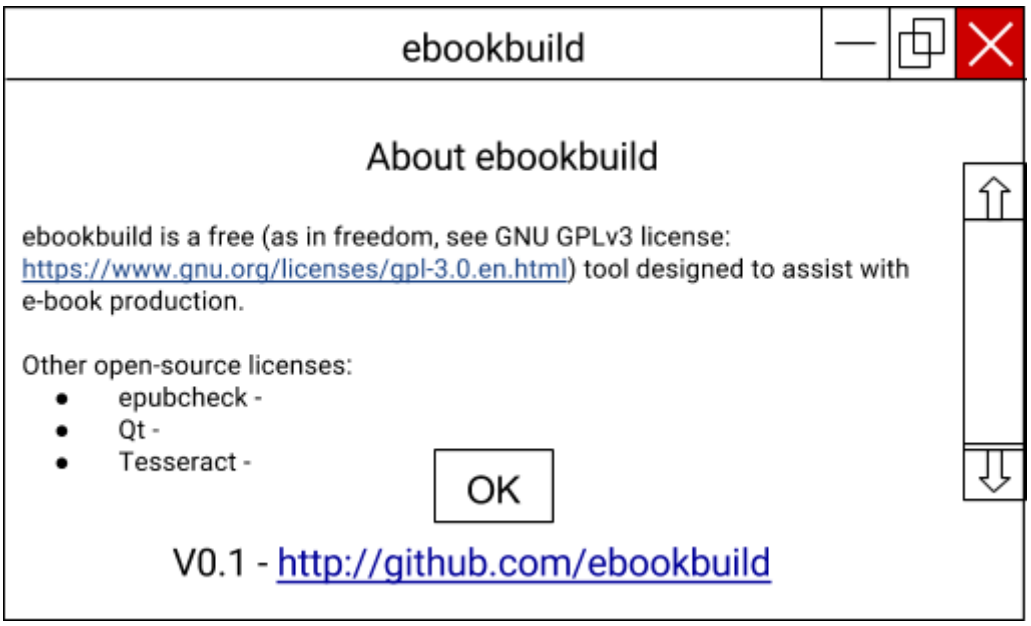
Dependencies:

To keep the software light, manageable and easily portable, a minimal set of dependencies is recommended.

- Python interpreter (if used) - Win, Mac, GNU/Linux and other (?, Python -license)
- QT/other UI toolkit - Win, Mac, GNU/Linux and other (C++, GNU GPL compatible)
- QT-Go binder ()
- epubcheck - Win, Mac, GNU/Linux and other (Java, BSD-license)
- kindlegen - Win, Mac, GNU/Linux (?, proprietary EULA) - need an alternative as EULA forbids distribution outside Amazon.com's store (try Calibre then Kindle Previewer, eventually will implement own solution.
- Tesseract - Win, Mac, GNU/Linux (C/C++, Apache 2.0-license)
- Tesseract wrapper for Go ()
- Pandoc - Win, Mac, GNU/Linux (Haskell with Python wrapper, GNU GPLv2 license)



Start mock-up screen for ebookbuild



About screen

Import allows the user to convert a document or use OCR to produce XHTML. Export takes the XHMTL and creates then checks the ePub before optionally making a Mobi counterpart.

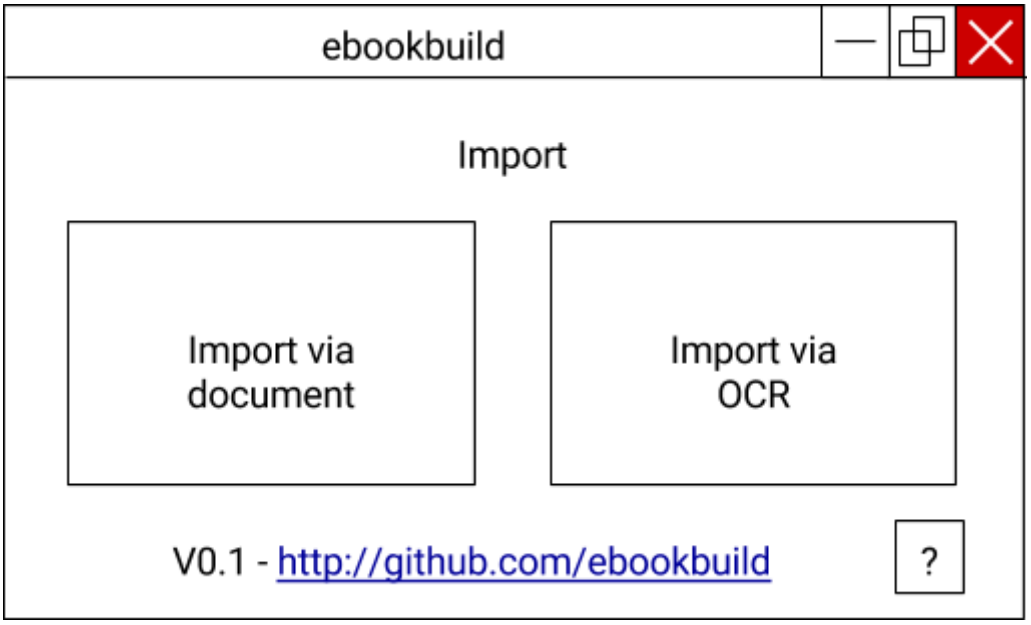
Import document file support:

This is a planned list of formats that most people would use to write book manuscripts on a computer (users are welcome to suggest more).

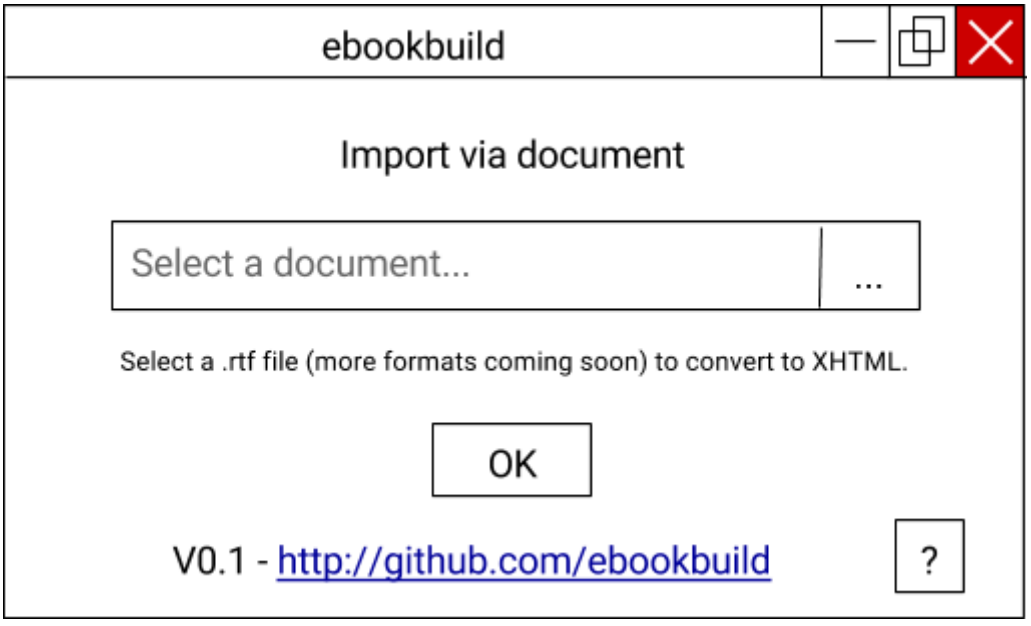
- .abw
- .doc
- .docx
- .docbook
- .html
- .lyx
- .odt
- .pages
- .pdf
- .rtf (start with this file format at v0.1)
- .tex
- .txt
- .wiki (to allow Wiki formatting to be converted)
- .xps

Export file support

- .epub - ePub v2.x and 3.y
- .mobi - Mobi for Kindle devices
- .crb - Comic Book Archive format and associated files (possibly)

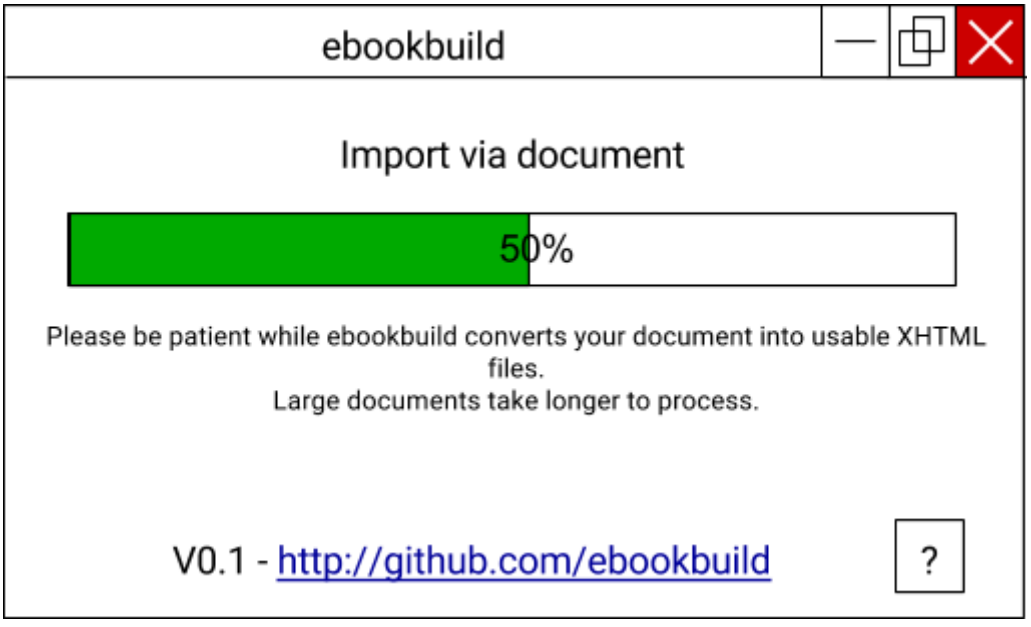


Mockup of the first screen encountered after clicking (or tapping) **Import.**

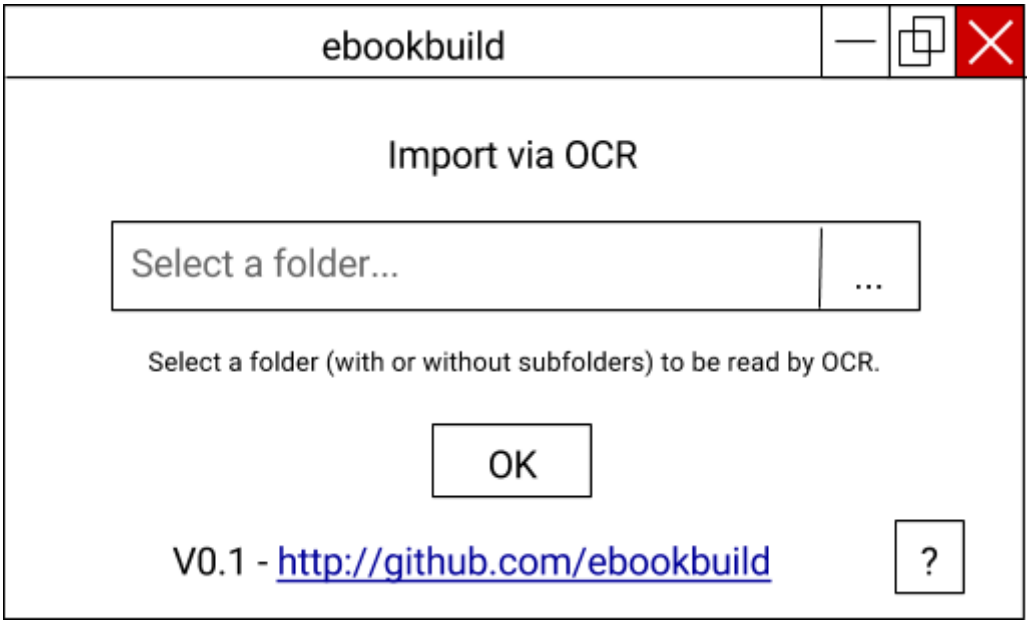


User selects a document file to import, future builds will convert most document files to .rtf or .xhtml depending. There will also be eventual folder and batch document support too.

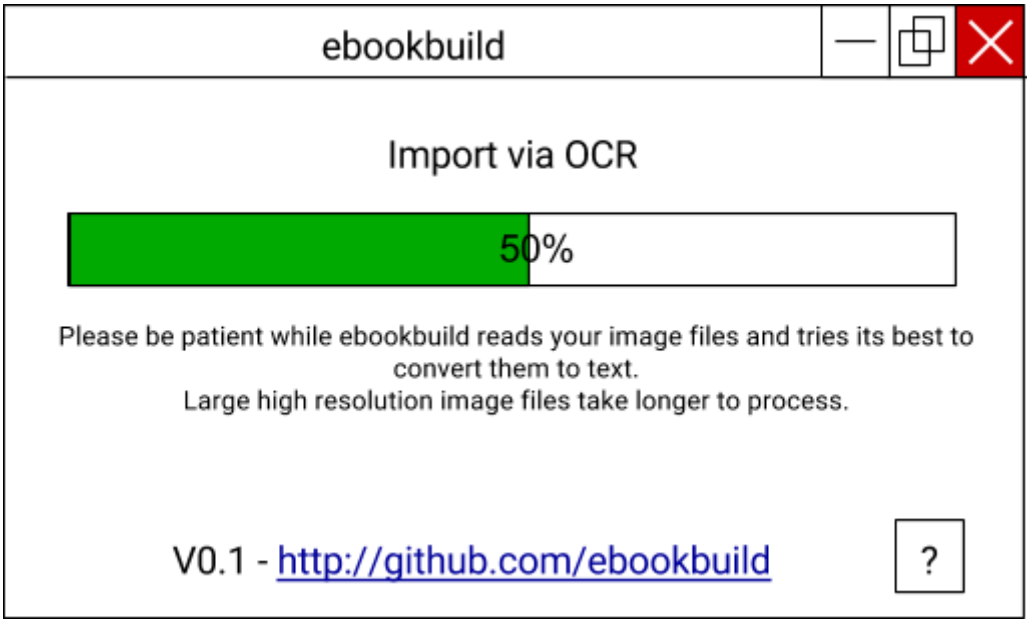
This screen is used to select a destination folder for the imported XHTML files to be placed in.



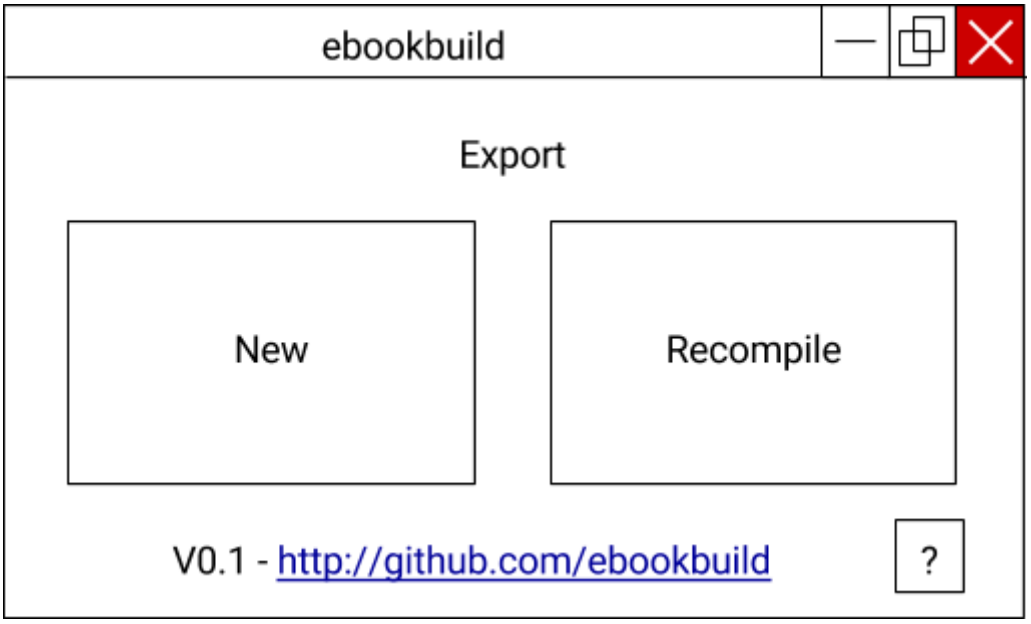
The files will be found in the folder provided by user ready for editing.
Alternatively…

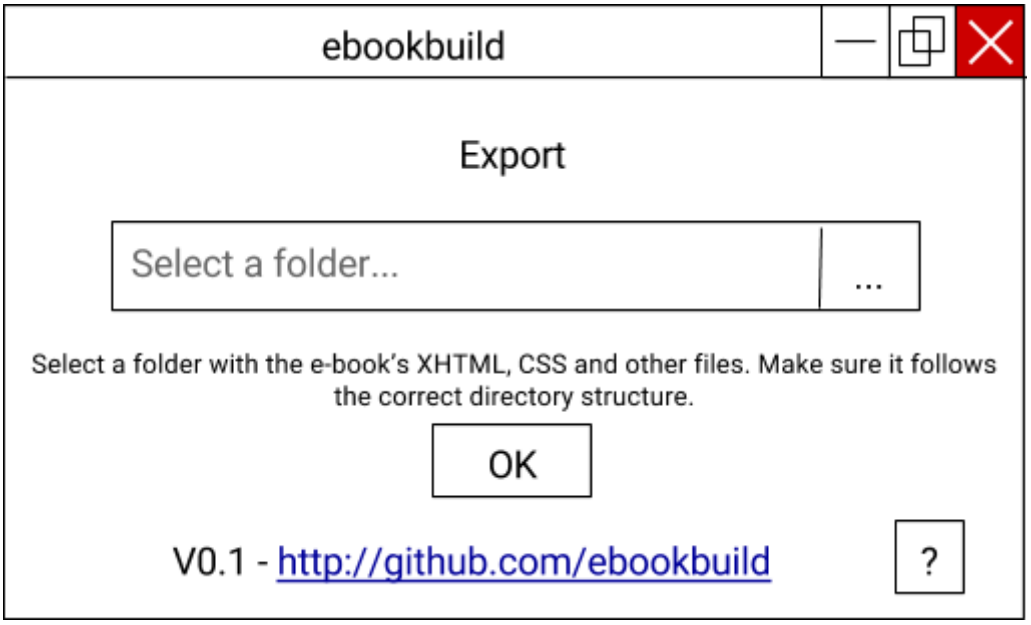


While not supported initially and not a first priority, built-in OCR support starts with the user choosing a folder with the images to be scanned (make sure images are placed in one folder, subfolders allowed)



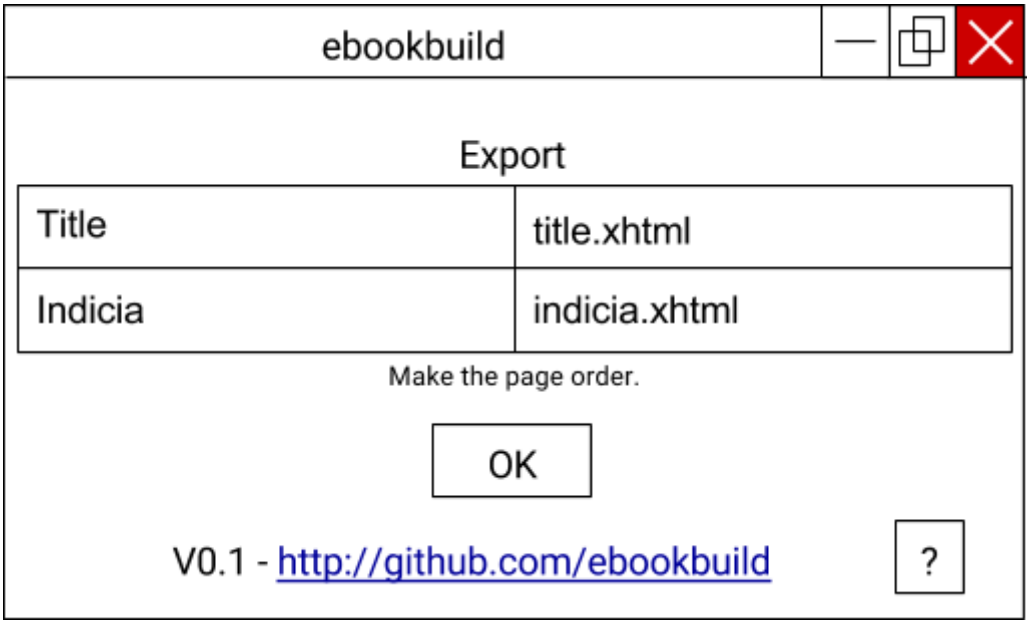
Process is finished and files are ready to be used in a production.





User selects folder to make as an e-book (a similar screen also appears to select the folder for re-compiling)

User enters in the book's metadata to write to the opf file and clicks OK to proceed.



User inputs the page order using a drag and drop interface (eventually it will pull the names of the xhtml files in and use drop-down menus)

Export options for the user, here they can decide via checkboxes if they want a .mobi book made or a fingerprint (eventually) that they can MD5/SHA 256 with ease (maybe show the MD5/SHA 256 on-screen?).
They can also minify the XHTML/CSS code by removing spaces and comments using the dropdown menu provided (all options catered to).

Next, epubcheck and kindlegen (if applicable) are run and errors/warnings are reported to user while the book is built and ready for testing.

In documentation, mention that e-books should have no warnings on epubcheck and few on kindlegen. ePub files can be tested with Adobe Digital Editions (Windows, Mac) Mobi files are limited to Kindle Previewer (Windows, Mac) and devices that support the official Kindle app.