Alexandria User Manual

JSLBrowning

September 27, 2022

1 What is Alexandria?

Named after the largest repository of information in the ancient world, Alexandria is meant to be an all-in-one online storytelling platform — a web-based library, the ultimate playlist, and an easily-accessible encyclopedia, all rolled together into an elegant, user-friendly package. The project was first developed as **Wall of History**, an online compendium of the LEGO BIONICLE story.

1.1 Licensing

Alexandria is currently published under the Apache License. The software is completely free to use, and you are free to make any changes you would like to the code (at your own risk, of course). You are *encouraged*, though not strictly required, to publish those changes yourself under the same license.

2 Getting Started

The first thing you'll want to do, of course, is acquire web hosting. I would recommend using HostGator.

Once you've acquired hosting, access your **cPanel**. This is the control panel from which almost all aspects of your website are controlled.

To install Alexandria as your website's front end, you'll need to download and install a program called FileZilla (click the quick download link labelled "FileZilla Client"). Once this is installed, click on **FTP Accounts** in your cPanel, and find the main account (it should have a path of something like `/home/[username]`). Click the button next to it that says "Configure FTP Client," and download the FTP configuration file for FileZilla. Open FileZilla, click **File**, then **Import**. Importing the config file will automatically configure FileZilla for uploading files to your server.

Once you've got FileZilla set up, locate the folder on your server named `public_html`. This is where the files that make up your website are stored. Upload the Alexandria files to this directory. Once you do this, the contents of `index.php` in the root directory should be visible when you visit your site with a web browser, and (for example), if you have an `index.html` file in a subdirectory named `about`, its contents should be visible when you visit `yoururl.com/about`. Index files are what are displayed to users when they visit web pages.

3 Setting Up Your Database

The contents of your site will be stored in a MySQL database. This guide should have come with an example `setup.sql` file that demonstrates how to format content for this database (more on this later), and once you've got everything loaded into your own setup file, you can go to your cPanel and click on `phpMyAdmin`. This is the control panel for your MySQL database. To upload the contents of your setup file, click on “Import,” the just select your file and click “Go.” Assuming your data is formatted correctly, phpMyAdmin should tell you that all insertions were successful, but if an error was encountered, the error message should be descriptive enough for you to fix it.

Once your data is imported, you need to set up a MySQL database user for the users of your site. Go back to your cPanel and click on “MySQL Databases,” then add a new user. Next, scroll down to “Add User to Database,” then add the user you just made to your database — when prompted to define their permissions, allow them to only perform `SELECT` operations. Once this is done, alter `db_connect.php` to contain the credentials for the database user you just added, and upload it to your site using FileZilla (overwriting the previous version of the file). Then you can visit your site and make sure everything'sworking!

4 The Alexandria Database

A very basic understanding of SQL will likely be necessary to actually build the database for your Alexandria library. The database uses a simple tree structure, with a small number of root nodes, numerous child nodes, and leaf nodes with no children of their own. “Routes” connecting leaves in various sequences are defined by you, the site administrator, and are how users navigate your site.

Each node in the tree has a unique, six-character `ID`, which is the most straightforward way of accessing pages on the site — for example…

`https://wallofhistory.com/read/?id=GNO2P6`

…will take you to *BIONICLE Chronicles #1: Tale of the Toa.* If you're worried about how unsightly that URL is, worry not — the tags database has support for *semantic tags*, allowing for cleaner URLs. For example…

`https://wallofhistory.com/read/?s=taleofthetoa`

These can be added like so:

```
INSERT INTO woh_tags VALUES (
    "GNO2P6",
    "semantic",
    "GN02P6.1",
    "taleofthetoa"
);
```

Setting up this database is the only thing that's really *required* for you to get your site up and running — everything else important is handled by the serverside code. The table of contents, for example, is automatically generated when a user visits the `read` page without specifying an ID — this triggers code that fetches all the root nodes (those without parents themselves), giving the user a full overview of content on the site.

5 Customization

For customizing your installation, we highly recommend using Visual Studio Code, with the recommended extensions for HTML, CSS, and JS installed. We also recommend putting the contents of your installation in the `root` folder of a USBWebserver, which will allow you to fully simulate the live version of the site, without actually changing anything on it until you're ready (after you've finished your changes, just use your FTP client to overwrite the versions of those files on your server).

6 Structure and Styling

The `main.css` file for Alexandria may be daunting, but the structure of the site is actually quite simple — far simpler than what you'd get using WordPress or a similar WYSIWYG platform. You can inspect the `test/index.html` page that should have come with your installation to familiarize yourself with how content on the site is formatted.

6.1 AutoCSS

One of Alexandria's useful features is **AutoCSS** — when a user visits a story page, the serverside code will automatically search the `css/id` directory for any CSS files with that ID as the filename, then traverse up the database tree two levels, also finding any files for the parent or grandparent. These are applied in the order **grandparent**, **parent**, **self**, meaning the leaf has final say on any conflicting styles.

7 Useful Features

7.1 Routes

Alexandria has support for multiple, simultaneous "routes" through the leaf nodes of a database. On Wall of History, there is one *main* route, and then individual novels or serials can be activated as additional routes. All these routs have unique saveplace variables, and when the "Read" button is pressed, the user will be prompted for which they want to resume. Options for *multiple* main routes are also possible (though we're putting some finishing touches on the code for this at the moment). Internally, routes are stored as `localStorage` variables on the end user's machine, and take the format `[id]:[boolean],[id]:[boolean],...` (though you should never have to interact with them in this raw format). You may have noticed the `content_version` variable in the database — this works f

7.2 Inline Content Swappers

On Wall of History, we faced a unique challenge — the official walkthrough of *MNOG* contained text descriptions of cutscenes, but as a web-based compendium, we could just include cutscenes in their original format. But then that text would simply be lost. The solution? Let users switch between blocks on the fly. These **inline content swappers** allows users to alternate between different versions of a piece of content *within* a single page, when differences aren't large enough to justify different versions in the database. An example of this can be seen here, where the cutscene videos can be swapped out for their text descriptions with a button underneath them.

An example of how these need to be formatted can be found in `format.html`.