Publishing Your TEI as a Static Site

Using GitHub, GitHub Pages, and GitHub Desktop, you can publish your TEI as a static site online. This enables you to not only store your work online, but it also allows you to collaborate with others on your project, share your in-progress work prior to or in lieu of print publication, and promote accessibility.

Please note that these instructions presume that you have already installed and been using Atom Text Editor to encode your text-based data, and that you are working within a GitHub repository.

To proceed with publishing your TEI as a static site online, follow the steps below.

Getting Started with GitHub and Atom

1. To prepare GitHub: navigate to https://github.com and create an account. Follow the prompts to create a **free GitHub account**. Be sure to make the email address associated with your account public, so that you'll later be able to publish a static site. Stay logged in.
2. To prepare GitHub Desktop: navigate to https://desktop.github.com to download **GitHub Desktop**. Then, open the application and log in.
3. To prepare Atom, open the program. Click on "Atom" → "Preferences…" → "Install" (on a Mac) OR on "File" → "Settings" → "Install" (on a PC). Search for the "**atom-html-preview**" plug-in and install it. This will later allow you

to preview an HTML transformation of your TEI file within the Atom program.

If you already have an existing repository on GitHub…

1. Be sure that all your files are saved and up-to-date in Atom.
2. Verify that your GitHub repository has all of the necessary parts:
    a. A *data* folder with at least 1 valid TEI file with an .xml extension.
    b. A *css* folder with two files: “CETEIcean.css” and “style.css”
    c. A *js* folder with 1 file: “CETEI.js”
    d. An *index.html* page that includes a TEI/CETEIcean script, with placeholder text substituted with the appropriate information for your project/your files.
3. If you have to make any changes to your repository, do so and save everything in Atom.
4. Open GitHub desktop and verify that you’re logged in and working within your repository. You can check this by looking at the upper left-hand corner, where it says “Current repository.”
5. “Commit” all changes to master, and then “Fetch origin” to push (sync) the changes.
6. In your Internet browser, navigate to https://github.com. Navigate to your repository and then click the “Settings” tab near the top of the page.
7. Scroll down until you see “GitHub pages.” Pop open the first drop-down menu and click “master branch” to enable pages. Then click Save.
8. Wait a few moments to visit your new page!
9. If you’d like to change how you page is rendered, modify the “style.css” file in your *css* folder. Write your own css scripts to style different tagged

elements of your digital edition in different ways. For more, see the *Styling Your Page* section below.

If you're getting started from "scratch"...

1. Back in your internet browser, make sure you're logged into your GitHub account on github.com.
2. Find Jessica Lu's emptyproject repository by navigating to https://github.com/lujessica/emptyproject.
3. To add the skeleton repository in its current form to *your own* GitHub account, click "**Fork**" in the upper right-hand corner. Now, the "emptyproject" repo will have been replicated to your own account.
4. Since you likely have a title for your project, you'll want to change "emptyproject" to a more appropriate name. To do so, look for your avatar/icon in the upper right-hand corner of your GitHub screen. Click it, and select "Your repositories" in the drop-down menu. Select the "emptyproject" repo that is now housed by your account. On the repository screen, look for and select the "Settings" tab in the top menu. Once you've accessed the settings, create a new name for your project repository and click "Rename."
5. Return to/open your GitHub Desktop program (if needed, log into using your GitHub login information). Click on "File" → "**Clone Repository**…"
6. Select your newly-named repository from the list and **clone** it to your own computer.
7. Locate the repository on your computer (it will likely be saved in a GitHub folder in a default location, e.g. "My Documents").

Customizing the Skeleton Repo For Your Project

1. Within the project repository on your computer, you will find several folders that you can use to organize, store, style, and publish your work. Use Atom to work within your repository on your local machine as you the repository for your own projects:
    a. The *data* folder is where you should save your encoded digital editions. Be sure that each file has an .xml file extension, and that you have named them in an ordered and efficient way. (The folder currently a two TEI files in it, named “basicdigitaledition.xml” and “blackDHdigitaledition.xml.” These are templates that you can use to jumpstart/adapt for your own projects. You are also free to delete them if/when you do not find them useful.)
    b. The *css* folder is where you should modify and/or write CSS code to control how your digital editions are rendered online. The “CETEIcean.css” file is vital and should not be deleted (though it can be modified); it allows your machine to read and then display TEI as if it were HTML-type markup. The “style.css” file is currently blank, and is where you would add your own CSS code to style your page. Any CSS code added to the “style.css” file will override the scripts in the foundational CETEIcean.css file, so this is where you should be doing most, if not all, of your styling work.
    c. The *js* folder houses your Javascript code. It currently contains a “CETEI.js” file that, together with the CSS file, allows your machine to read and then render TEI tags online.
    d. The *pages* folder will house the files and code that will control the individual pages displaying your encoded digital editions. In it, you’ll

currently find a file titled “pagename.html.” This file should be renamed to match the title of a digital edition you wish to display. The contents of the file should be further modified to substitute any placeholder text with the appropriate text for your project/files.

   e. The *txt* folder should house all of the plain-text transcriptions of your text-based primary material.
   f. The “index.html” file should be modified to substitute any placeholder text with the appropriate text for your project/files.

2. As you work within Atom on your project, remember to save often, and use GitHub Desktop to “commit” and “push” your changes.
3. When you’re ready to publish and share your work, use the *“If you already have an existing repository on GitHub…”* section above to guide you.

Styling Your Page

1. You can begin to manipulate the way your page looks by adding CSS (Cascading Style Sheets) script. In Atom, open the “style.css” file from your css folder. It should be blank, because you haven’t yet added any additional styling to the page.
2. Let’s try adding a basic border around your TEI header content with the following script:

```
tei-teiHeader {
    border: 2px solid black;
}
```

3. To preview the changes, open your index.html file and make sure your cursor is active within the file window. Then hold down **Shift + Ctrl + H** keys (pressed and held order) to activate the plug-in you installed at the start of this tutorial. A window should appear in the right-hand side of the

Atom program. It will show you what the HTML transformation of your page would look like.

4. As you continue to manipulate the styling of your page, be sure to return to the GitHub Desktop application to commit and push your new changes to your GitHub repository online.
5. For more practice with CSS, you can visit code-generators at http://enjoycss.com or https://webcode.tools. You can also view introductory tutorials for CSS at https://developer.mozilla.org/en-US/docs/Learn/CSS/Introduction_to_CSS.