WRM Webmaster Guide

# Introduction

Whether or not you attended the webmaster skill session, you should look through this document because it has important information about how to handle the html for the website. I didn’t actually go to the skill session, so I don’t know what they teach you. I made the website according to industry standards. This ensures that the website is easy to understand and it is more educational for whoever edits it.

When you access a website, you are opening a file on the server that is mapped to a website address. If you go to “example.com”, by default example.com will send you its “index.html” file. This is considered the landing page, or home page, for all websites. Therefore index.html is the webpage that is always opened first when someone goes to our site.

If we want to take the user to a different page, they can access it like this: “example.com/other\_page.html”. Everything past the root directory (example.com/ ) can be accessed by giving a filepath. Since our pages will be in the root directory, we just put the file name right after the web domain (.com). If we want to access a file inside a folder, we use “example.com/folder\_name/file\_name.jpg”. If that folder is inside another folder: “example.com/folder1/folder\_inside\_folder1/file\_in\_folder2.jpg”. You should be able to understand the pattern from here.

It is also useful to note that you can open these html files on your own computer. If you make a copy of the folder with the website in it to your computer, and then try to open any of the files in your browser, you will notice that it looks and behaves the same as if you went to the website. All of the links will work and all pictures will show up… This is because we aren’t doing anything special here; it’s simply a file your browser is opening, and as long as we have the other files we are linking to with links and images, (the other webpages and the image files), everything will show up exactly the same.

It would probably be helpful to print this out so you can look at it easily when you work on the website.

# Basic Stuff

HTML stands for Hypertext Markup Language. All this means is it tells a browser how to display information. Html is NOT a coding language. JavaScript and PHP are coding languages that are widely used alongside html, but for our purposes we don’t need it. Bootstrap uses JavaScript, but we never need to touch or see that code. All the work is done for us.

Hopefully you went to the webmaster skill session, because I will not spend much time explaining the very basics of html. I will link some basic html tutorials at the bottom in the resources section if you need it. The html documents themselves are very very VERY well commented. If something doesn’t make sense, and you are unable to understand it through google or the resources, email me at wvanbure@purdue.edu.

An element in html consists of an opening tag and a closing tag. A tag is some word inside angle brackets like <div>. A closing tag for this div would look like </div>. ALL OPENING TAGS SHOULD BE CLOSED. If you are adding a lot of text, make sure you use a code editor (see below) to help make sure you don’t make mistakes with your opening and closing tags.

You will notice that links to files on our website are written in “relative” form. This means the filepath depends on where you start. For example, if we are in index.html in the root folder, and we want to put the filepath for an image in the “img” folder called “this.jpg”, we would write “./img/this.jpg”. In filepaths, ‘.’ Represents the current folder. If you don’t write it, the filepath will still work because if you write “img/this.jpg”, the ‘./’ will be assumed. It’s good practice to write the ‘./’ however, to make it more clear what file you are accessing. Also, if you need to link to something in a different folder, you might need to use ‘../’, which represents the folder above your current folder.

To illustrate this use, let’s imagine you are linking to the “this.jpg” again, but you are starting from a file called “hydroponics.html” in a folder called “projects”. Your filepath would be “../img/this.jpg”. Note that you can use ‘../’ more than once, too. If your “hydroponics.html” is inside “projects/hyd/” your filepath would be “../../img/this.jpg”. One ‘..’ is to get out of /hyd, another to get out of /projects.

As a quick note, you could write the filepaths in “absolute” form, but please do not do this. An example of this would be “<http://www.example.com/img/this.jpg>”, which would achieve the same result, but is bad practice. Imagine that you have made a copy to the website to your computer to edit and test. If in this copied website you try to use any links that are linked absolutely, it will take your browser to the “http://www.example.com/img/this.jpg” instead of taking you to the “/img/this.jpg” in your copied folder. If everything is written in relative form, these links will always work correctly provided you are not missing any files.

# Chrome Developer Console

Google Chrome is my recommended browser for editing the website. It has a great tool called the developer console you can open with (CTRL + SHIFT+ i), or by right clicking and selecting “Inspect”. It allows you to examine specific elements on a webpage and check where it is getting styles from and lots of other useful info. If something is going wrong on the website and you don’t understand how or why, using the dev console is the place to start.

IMPORTANT: If you are making changes to files and you are checking to see the results on the website, you may notice that it doesn’t seem to change sometimes. This is because web browsers will “cache” websites. This means if you have recently downloaded the website by going to it, it still has that downloaded file somewhere on your computer. So instead of downloading the same file again, your browser just uses the one you already have, meaning if you made changes you will not be looking at the document with your changes, instead the old one. To make sure you see the most recent version of your website, you need to clear your cache, or “force reload.” Refreshing the webpage does not work. In Windows on Chrome the key shortcut for this is “CTRL + R”. This forces the browser to get rid of the old copy and download it again.

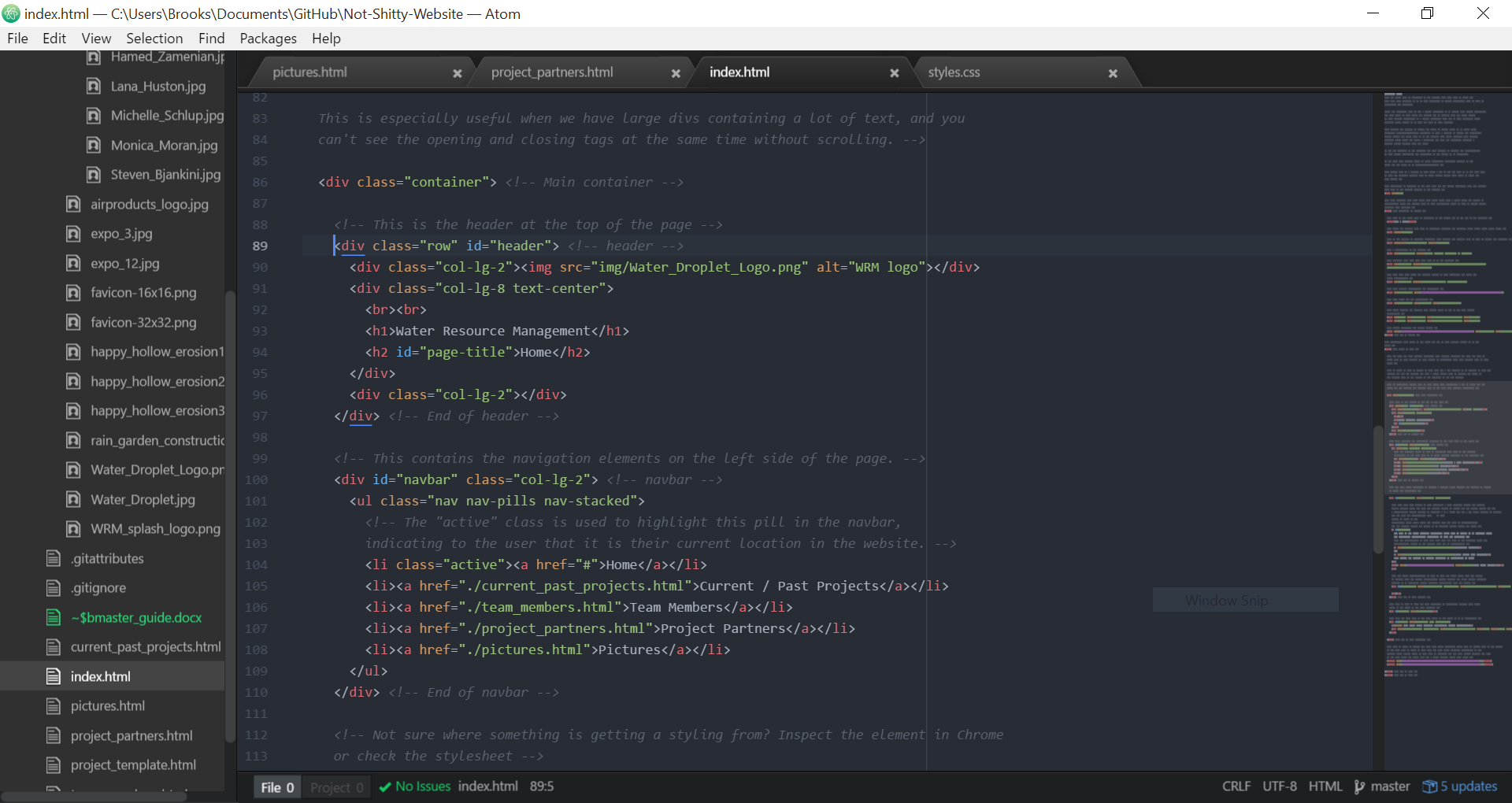
# Editing

An html file is just a file with text in it. That’s all. Meaning you can open up an html file in a plain text editor and start making changes to it. I DO NOT recommend using Microsoft Silverlight to edit the website. The nature of website editors like Silverlight is that they try to do things automatically when you add features, but they don’t always make the html formatted well, making it hard to interpret the text. It is not necessary to use any website editor like Silverlight; all you need is to edit the text. Don’t worry! It’s not as scary as it sounds.

One thing to note is you can’t edit html using something like Microsoft Word. This is because Word does not simply edit plain text; it adds in special formatting so you can change fonts and centering and colors and crazy stuff, which means the data for these files is read differently and it is not *just* text. However, Notepad works, as well as Wordpad.

However, these are not recommended. Trying to edit a website in Notepad is painful. Instead I suggest using a text editor made specifically for editing code and html. There are many options out there, like Notepad++, Sublime, Textwrangler, jEdit, etc… But the one I prefer, after trying some of the others, is Atom. It’s fully customizable with lots of optional plugins and stuff. This is nice for me since I’m a Computer Science major, and I work with code, but for you just editing the website, it’s not too necessary. It wouldn’t hurt though, either.

The nice thing about editors like these is that they highlight your text. You can see in my screenshot how Atom automatically highlights different parts of my text depending on what they are, and it makes it 5000 times easier to read. It also checks for me if I have proper opening and closing tags. With my cursor on the <div> tag, it underlines it, and the corresponding closing tag.

 Text editors like these also usually have the option to auto-indent files, which is useful for keeping the code looking clean. One of the first things I would recommend doing when you first use an editor is to CHANGE THE INDENT to 2 spaces. This is common on html files because larger indents make the text too hard to read (because so many indents are usually necessary).

# Standards

When editing the html of the website, please comment your changes by using the <!-- --> tags. It’s normally not necessary to have such detailed comments in anything, but for our purposes it is; future webmasters, who may or may not have html or coding experience, will have to look at the documents and be able to understand what it means. For this reason, please explain things well.

For the structure of a website, it is standard to have a format like this:

* Website folder
  + index.html (this is the file that is automatically loaded when someone goes to the site)
  + other pages in .html format
  + img (folder)
    - almost all your images should be stored in here
    - For more organization, you can have a more specific folder in here (for example, the team\_members folder in our website
  + css (folder)
    - styles.css (this file is a stylesheet that contains styling for elements in the html. It is standard to never include css in the html file, and keep it in a separate file.)
    - Any other stylesheets go in here, but we only need one
  + For more organization, you can group pages in folders, but for our site, we don’t have enough pages to justify that. Leave the structure as is.

It is also normal to make files and folders all lowercase, and words separated by an underscore. You can see this in practice in all the files and folders already existing in the website. Please follow this practice. For the team pictures you will notice they, however, are capitalized. This is one exception to the convention.

For html, and coding languages and other related text files, it is standard to not let your lines be really long. This is all done with the goal to make the text more readable. You will notice in the html files that some lines are broken onto the next line manually. When a browser reads html, it doesn’t matter how many extra spaces and carriage returns you have; it will ignore the extra space. Because of this, use extra space to make stuff easier to read. Break really long lines around the same length as the rest of the document.

IMPORTANT: The alt attribute is an important attribute you put on ALL images on your website. The text you put in the attribute is what will display if for any reason the browser fails to retrieve the linked image. This is also used by screen readers. Because of these reasons, it is considered extremely bad practice to not add an alt attribute to every image. The alt text should be a simple description of the image.

# Bootstrap

The website is built using Bootstrap. Bootstrap is a library of javascript and css that helps make it easy to layout websites and make them responsive. Responsive means the website responds to the user’s device; if it’s a small screen things will scale or rearrange to make themselves more readable on that screen.

For all the Bootstrap I used in making the website, I commented it fairly well. If there is a class you don’t know about, it’s very easy to learn about it by googling. I also highly suggest looking at Bootstrap’s documentation:

<http://getbootstrap.com/examples/theme/>

<http://getbootstrap.com/components/>

These show examples of all the things in Bootstrap in use, and shows the html as well. If you want to look at the html for a specific element, right click in chrome and select “Inspect”.

IMPORTANT: Note that if you try looking at the website without having internet (i.e. opening it on your own computer with no internet) it will not look right. Every once in a while if you are constantly opening and reloading it, this will also happen. Likely, this is because Bootstrap failed to download. In order for our website to look correct, the computer also downloads a file from the Bootstrap website containing the code to make all of our Bootstrap do what it is supposed to do in html. If it happens, make sure the internet is working and you are not loading a cached copy (use “CTRL + R”), and try again.

# Resources

[w3 Schools](http://www.w3schools.com/html/html_basic.asp)

* A great resource that has lots of tutorials about various parts of html
* Recommended as the first resource if you’re new to html
* Nice and quick to go through tutorial pages
* If you need to learn what a certain tag is for, this is often the best and quickest resource. Googling a tag usually brings up a w3 webpage.

[Learn HTML in 15 Minutes (youtube video)](https://youtu.be/Ggh_y-33Eso)

* “Learn HTML in 15 Minutes”
* Enough said. Takes you through the basics very quickly.
* If you want to learn more, I suggest watching more of the videos from this channel. He has videos on Bootstrap and more in depth HTML videos

# TO-DO

A list of things to do when you become webmaster:

1. Update the EPICS copyright year at the bottom of each webpage
2. Update the team member pictures, names, email addresses, and don’t forget the the MAILTO links
3. Update the “Current term” and if necessary the “Term completed” on each of the projects
4. Add the new team members to the team members list of their projects, if not already in the list
5. If there is a new project, add it to the project page, and add new project partners to the project partners page
6. Look for opportunities to take pictures that you can add to the pictures page on the website.

As a webmaster there should be some sort of “project” or goal you work on for the website over the semester. Some ideas for what you could work on that I planned on, but decided not to do considering how much time I already spent on rebuilding the website:

* Make more mobile responsive (scales to phone screens better, a little challenging)
* Create individual pages for projects (if you are looking for a challenge)
  + I had the idea for the pages to include
    - Pictures of the project
    - Schematics or other design documents
    - All the basic information already on the project description
    - A more detailed description and background
* Improve the pictures page by first getting some more and better pictures to display, and then maybe make the layout a little nicer (shouldn’t be too challenging)

If you see something that should be added to the guide, like a link, or a section, or some other random change you think is helpful to future webmasters, feel free to add it.

If you ever desperately need help, or can’t find anything to answer a question, or have some other reason to justify contacting me, you can do so by emailing me at wvanbure@purdue.edu

PROTIP: Read all the comments in the html files