

AguaClara Website Manual

Jeremy Candelas

SPRING 2017

Purpose

The purpose of this website manual is to ensure that in future semesters, the AguaClara engineering project team will be able to continue to update and maintain its website. For the most part, the core pages and features of the website should not need to be modified, although there may be times in which the team may want to add or change certain aspects of the website. It is in these instances that where this website manual will be most beneficial.

For large-scale renovations and new features, it would be best to recruit a webmaster who has a considerable amount of experience with the necessary skills to undertake such a task. While it would be technically feasible for someone to learn how to accomplish these tasks over the course of the semester, it would be more beneficial to the team for someone to already have these skills. The necessary and recommended skills for who should be a webmaster are listed below.

Who Should be a Webmaster

The skills necessary to maintain and update the AguaClara website are not difficult to learn. Realistically, anyone who is passionate about AguaClara and is willing to take the time to learn a few basic web development skills would be suitable as a webmaster. However, it would be ideal that a webmaster possess the skills already, so that more time can be spent making general improvements and adding new features to the website. Below is a list of the necessary and recommended skills for a future webmaster:

HTML: The HTML code establishes the structure of the website. If any changes are to be made to the website, the webmaster absolutely must be familiar with HTML. There

are varying versions, with the most recent being HTML 5 as of the time of this document's writing. However, familiarity with even the most basic HTML is all that is necessary to operate the website as webmaster.

CSS: Cascading Style Sheets (CSS) establish how HTML elements should be displayed. While one could technically operate the website without knowing CSS, it is essential for the webmaster to be familiar with it in order for the website to be aesthetically-pleasing. As more and more users shift to mobile devices for viewing, it is important for a webmaster to be familiar with responsive designing with CSS.

Google: In all seriousness, knowing how to Google a solution is an absolutely necessary skill for any webmaster to have. Problems will come up that will seem all but impossible to figure out on one's own. Knowing where and how to find the solution is vitally important, and Google is often one of the best resources to find tutorials, forums, and other sources of possible solutions.

JavaScript: Not related to, nor to be confused with Java, this is a scripting language. For the most part, the website does not use JavaScript except for a few lines of code. Unless the webmaster knows what they're doing, they should not alter the code in the first place anyhow. Knowing JavaScript may be useful for implementing new features on the website in the future, depending on what the team determines to be potentially useful.

Adobe: The Adobe Creative Cloud (Photoshop, Lightroom, Illustrator, etc.) are incredibly useful and powerful tools for editing and designing. If photographs or graphics need to be featured on the website, the Adobe tools will make this far easier. Knowing how to use these tools is also useful when it comes to creating wireframes and other page mockups for the website before any actual coding occurs.

*Absolutely necessary skills in **bold**.*

Beyond these skills, a webmaster should be able to communicate ideas effectively. Websites should not be overloaded with information, but only with what is essential. A webmaster should also possess a strong hold on UX (user-experience) and design principles. If a website has a poor UX or design, the website cannot be effectively utilized. Lastly, it is recommended that the webmaster be someone who is open to - and actively seeks out - feedback. By employing [design thinking](#) in web design, one can ensure that the website is constantly improving, and is always a work-in-progress.

Gaining Access to the Website

Monroe will need to email webservices@cornell.edu to request that the designated webmaster be given access to the website files. After the webmaster has been given permission, they will need to establish a connection to the server.

On a Mac: Open the Finder, and in the top menu select “Go” then select “Connect to Server.”

Alternatively, upon opening the Finder window, press ⌘+K. A prompt will open; paste the

server address: <https://static-webdav.kproxy.cornell.edu/cee/aguaclara> Click “Connect” and the webmaster will be prompted to log in with their NetID and password.

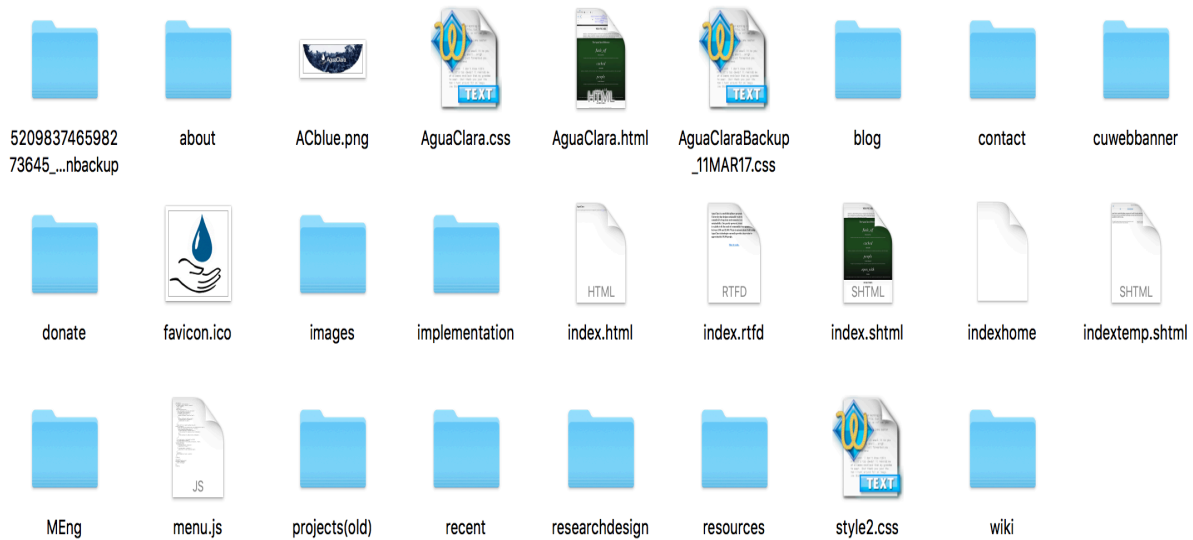
On a PC: It does not appear that there is quite as simple a way of connecting to the server while using a PC. The simpler solution might be to use a Mac computer in one of the libraries or renting out a Macbook from the library. Otherwise, the webmaster will need to install an FTP client such as FileZilla or Cyberduck (there are others that exist, these are just examples) to connect to the website.

Each FTP client operates a little differently, but for the most part they operate in the same way. The webmaster will connect to the host: <https://static-webdav.kproxy.cornell.edu/cee/aguaclara> with their username (NetID) and password (NetID password).

Editing the Website

The webmaster will need to install an application capable of opening, editing, and saving HTML and CSS documents in order to edit the website. TextWrangler or BBEdit are popular, free options. Paid options include the licensed version of BBEdit, Smultron, and others. Note that these are only a few options - it is ultimately up to personal preference. The only required capabilities are that the webmaster can 1) open, 2) edit, and 3) save the documents with the necessary file extension (.html, .css, .js, etc.).

File Hierarchy: When connected to the server, one will see a collection of files, as shown in the example below. It will look similar in an FTP client. Also note that the exact display format will vary based on user settings (e.g. showing in a list versus icon mode).



Example of the web folder, as shown on a Mac.

The homepage points to the file titled “AguaClara.html” — there are several other files that are named along the lines of “index” which are similar, but are there merely as reference points.

These extra files should not be moved or deleted as, if something were to go wrong, they could be used as backup files.

Each folder represents a different section of the website. Some point to content that website users will see and be able to easily click links to access. For example, the “about” folder contains the web pages for the About Us page, the project map page, etc. Someone hoping to access the

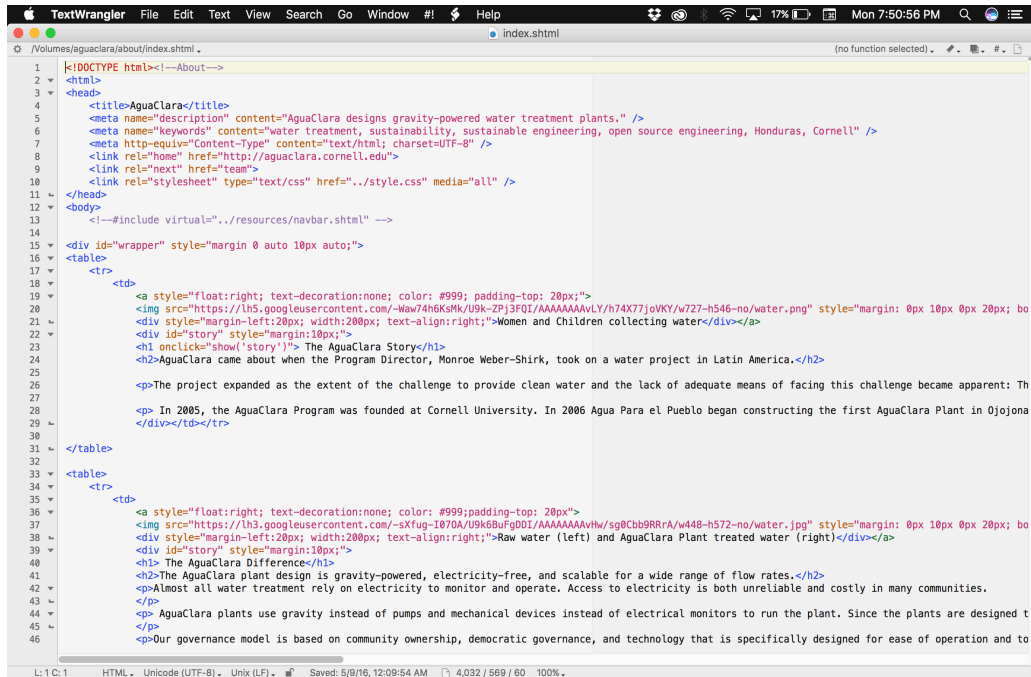
project maps would go to <http://aguaclara.cornell.edu/about/projectmap/>. The corresponding file path would be:

aguaclara > about > projectmap > index.shtml

Uploading: Save the document to the desired folder. It will take anywhere from a few seconds to a few minutes to upload depending upon the file size (photos and videos will take longer than a HTML or CSS file, which is simply a text-based file).

New Folders: To create a new folder, go to what *will be* the parent folder, right-click, and then click “Create new folder.” In an FTP client, there may be a button available to create a new folder. After giving the folder a name, it is ready to be used. For the sake of simplicity and organization, ensure that the folder will not be redundant before creating a new one. Keep in mind that the primary purpose of folders is for organization.

Opening a File for Editing: To open a file to edit it, simply double-click the file to open with the operating system’s default application, or right-click > Open With to choose a specific program. This is what editing an HTML document looks like:



```

1  |<!DOCTYPE html><!--About-->
2  |<html>
3  |<head>
4  |   <title>AquaClara</title>
5  |   <meta names="description" content="AquaClara designs gravity-powered water treatment plants." />
6  |   <meta names="keywords" content="water treatment, sustainability, sustainable engineering, open source engineering, Honduras, Cornell" />
7  |   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
8  |   <link rel="home" href="http://aguaclara.cornell.edu">
9  |   <link rel="next" href="team">
10 |   <link rel="stylesheet" type="text/css" href=" ../style.css" media="all" />
11 | </head>
12 | <body>
13 |   <!--#include virtual=" ../resources/navbar.shtml" -->
14 |
15 |   <div id="wrapper" style="margin 0 auto 10px auto;">
16 |   <table>
17 |     <tr>
18 |       <td>
19 |         <a style="float:right; text-decoration:none; color: #999; padding-top: 20px;">
20 |           Women and Children collecting water</div></a>
22 |         <div id="story" style="margin:10px;">
23 |           <h1 onclick="show('story')"> The AquaClara Story</h1>
24 |           <h2>AquaClara came about when the Program Director, Monroe Weber-Shirk, took on a water project in Latin America.</h2>
25 |           <p>The project expanded as the extent of the challenge to provide clean water and the lack of adequate means of facing this challenge became apparent: Th
26 |           <p> In 2005, the AquaClara Program was founded at Cornell University. In 2006 Agua Para el Pueblo began constructing the first AquaClara Plant in Ojojona
27 |           </div></td></tr>
28 |
29 |   </table>
30 |
31 |   <table>
32 |     <tr>
33 |       <td>
34 |         <a style="float:right; text-decoration:none; color: #999; padding-top: 20px">
35 |           Raw water (left) and AquaClara Plant treated water (right)</div></a>
37 |           <div id="story" style="margin:10px;">
38 |             <h1> The AquaClara Difference</h1>
39 |             <h2>The AquaClara plant design is gravity-powered, electricity-free, and scalable for a wide range of flow rates.</h2>
40 |             <p>Almost all water treatment rely on electricity to monitor and operate. Access to electricity is both unreliable and costly in many communities.
41 |             </p>
42 |             <p> AquaClara plants use gravity instead of pumps and mechanical devices instead of electrical monitors to run the plant. Since the plants are designed t
43 |             </p>
44 |             <p>Our governance model is based on community ownership, democratic governance, and technology that is specifically designed for ease of operation and to
45 |             </p>
46 |             <p>Our governance model is based on community ownership, democratic governance, and technology that is specifically designed for ease of operation and to

```

Editing a webpage (example shown with TextWrangler).

The example shows the overall HTML structure of the webpage. By simply editing and then saving the document, the website will update. The update is not quite instantaneous, but it will typically have updated within a matter of seconds.

Best Practices for Editing

When editing the website, it is important to have a backup plan in case something goes wrong. Simply forgetting to close an HTML tag, or a CSS-property can cause the website to “break.”

That is, the website's layout and/or design may radically change to the point where the website is unusable. In the vast majority of cases, these are simple fixes, but breaking the website can lead to minor moments of panic. The webmaster should employ proper practices and safeguards to fix errors when they go wrong, but also to prevent errors from occurring.

Backups: Whenever the webmaster is working on a page that is currently live on the website, a backup of the webpage should be created. Save the file to the AguaClara directory and/or a personal computer as: OriginalName_Backup_Date.EXTENSION

Offline Edits: An alternative method is to work offline. Note this does not literally mean working "offline." Rather, create a local version of the web file saved to a personal computer. Open this file in both a text editor, and open the file in a web browser. After changes to the local file have been saved, refresh the page in the web browser to see the changes put into effect. If the changes are satisfactory, overwrite the coding in the original file with that from the local file.

One Project at a Time: Avoid working on multiple pages at one time. This will significantly reduce the amount of errors experienced during the web development process. When editing a single page, avoid working on multiple aspects of the page at one time. In other words, do not work on updating the header, the body, and layout positioning at the same time. Finish one project and move on to the next.

Cut Once, Measure Twice: When making changes, it is best to add several line breaks around the section one is working on. These line breaks will not affect the overall layout of the page, but they will affect the loading times of the website. As such, once the changes are complete, the line breaks should be removed.

Leave Comments: Comment tags are an easy way to make notes to oneself when editing. These comments are only visible within the source document, and will not appear on the website. To leave a comment, place the comment within the following:

In an HTML document: `<!-- COMMENT HERE -->`

In a CSS document: `/* COMMENT HERE */`

While comment tags have no character limits, it's best to leave them as short as possible, and use them as sparingly as possible. Comments should not be overused, as one should remember that they *do* take up space. If a document is overloaded with complements, it will slow down the website to an extent, and after a certain point, they can make navigating the document more difficult.

```

<!-- AguaClara Difference Ends Here -->

<!-- Our Story Starts Here -->
<div class="w3-row" align="center" style="box-shadow:inset 0 0 75px #eee;">
  <div style="max-width:1100px;">

    <h1 align="center">OUR STORY</h1>
    </br>

    <div class="w3-half w3-container" align="center">
      <div style="max-width:560px;margin-bottom:10px;">
        <div class="video-container">
          <iframe width="560" height="315" src="http://www.youtube.com/embed/dwXXy_11bR4?wmode=transparent" frame
        </div>
      </div>
    </div>

    <div class="w3-half w3-container" align="center">
      <div style="max-width:560px;" align="center">
        <div style="max-width:560px;text-align:left;">
          AguaClara came about when the Program Director, Monroe Weber-Shirk, took on a water project in Latin Ame
          The project expanded as the extent of the challenge to provide clean water and the lack of adequate means of facing this cha
          In 2005, the AguaClara Program was founded at Cornell University. In 2006 Agua Para el Pueblo began constructing the first /
        </div>
      </div>
    </div>

  </div>
</div>
<!-- Our Story Ends Here -->

```

An example of editing a specific portion with line breaks and comments in HTML.

Basic Page Structure

While the overall structure of a webpage may vary depending on the type of content that needs to be displayed, the website tends to stay within the bounds of a fairly rough structure overall.

Some pages may employ a somewhat different structure depending on the content being displayed, and how it is wanted to be displayed. Once one is familiar with the basic page layout, and is familiar with basic CSS and HTML coding, it is easy to create more complex pages for the website. The basic structure of pages on the website is as follows:

```

<html>

  <head>

    <!--#include virtual="resources/header.shtml" -->

    <link rel="home" href="http://aguaclara.cornell.edu">

    <meta name="description" content="AguaClara designs gravity-powered water
treatment plants." />

    <meta name="keywords" content="water treatment, sustainability, sustainable
engineering, open source engineering, Honduras, Cornell" />

    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

    <title>AguaClara</title>

    <link rel="stylesheet" href="https://www.w3schools.com/w3css/3/w3.css">

  </head>

  <body>

    <!--#include virtual="resources/navbar.shtml" -->

    <div class="Feature" style="overflow:hidden;">&nbsp;</div>

      <!-- SectionOne -->

        <div class="SectionOneTitle">

          Section One Content

        </div>

      <!-- SectionOne Ends -->

      <!-- SectionTwo -->

        <div class="SectionTwoTitle">

          Section Two Content

```

```

        </div>

        <!-- SectionTwo Ends -->

    <footer>

        <!--#include virtual="resources/footer.shtml" -->

    </footer>

    <script>

        Any inline scripts</script>

    </body>

</html>

```

Explaining the Structure

What do each of these tags mean? A webmaster looking for a more in-depth explanation should begin with the courses and reference materials listed at the end of this manual.

`<HTML>` Defines the language of the document for the web browser. Every page must start with this tag, and be closed with `</HTML>`.

`<head>` This is the first information the page will load, and calls the page title, description, CSS, etc. that “defines” the website. The header ends where the content begins. End with `</head>`.

Commands such as `<!--#include virtual="resources/header.shtml" -->` can be used to “call” other HTML documents to be incorporated into the webpage. This is useful where an element is

repeated throughout the site and will not change from page to page. This specific example is the header which includes the main menu navigation.

The information leading up to the stylesheet `<link rel="stylesheet" href="https://www.w3schools.com/w3css/3/w3.css">` is not necessary to explain, just know that it helps in defining the site. The stylesheet tag calls a CSS document that allows for CSS3 (the latest version of CSS) properties to be utilized. This is what helps in creating the website's responsive design.

The section labeled `<body>` is where the main content is. After content for the page is finished, end with `</body>`.

The `<div>` tag is an element that designates a *division* - or a section of a webpage. Using classes and id's, the webmaster can style individual sections. Fundamentally, an id and a class can be used to accomplish the same effects. However, it is useful to use them for their intended purposes.

div class: This should be used for a "class" of divs, that is, recurring elements.

div id: The div id tag should be used for a div that will only appear once per page.

The stylings for the div tags are controlled in the CSS document, `AguaClara.css` in the main folder. To set the style for a div class or a div id, enter the style in the following format in the CSS document:

```
#DivID_Example {  
  color: red;  
  background-color: black;  
}  
  
.DivClass_Example {  
  color: black;  
  background-color: white;  
  padding: 10px;  
}
```

An element can have both a div class and a div id tag:

```
<div id="example" class="example2">Content</div>
```

In this case, the style for the div id would take precedence. In the examples of a div id and div class stylings in the CSS document formatting, this specific div would have red text, and a black background. The specifications given in the div id will take priority over the specifications given by the div class. However, the div would still have ten pixels of padding, as a declaration for “padding: 10px” is given for the class. Since no padding specifications are given for the div id, the class declaration will be used. If there was no declaration for padding at all, it will default to zero pixels of padding.

Guidelines

This should not be an issue in the foreseeable future, but in any instance where materials containing the Cornell logo are being utilized, one must follow the [Cornell Brand](#) Guidelines. Although the website is able to use any colors in its design, it is highly recommended to use colors which are complementary to Cornell's brand.

BLACK #222222		CARNELLIAN #B31B1B	
WHITE #FFFFFF	LIGHT GRAY #F7F7F7		
LINK COLOR #3787B0	SUSTAINABLE GREEN #6EB43F	IMPACT ORANGE #F8981D	#EF4035
#073949	#0068AC	#89CCE2	#C9D6A5
#A2998B	#D8D2C9	#9FAD9F	#BFB778

Above are the [recommended colors](#) for use in web design.

Resources

For further information concerning HTML, CSS, and web design and development in general, consult these resources:

Courses:

<https://www.codecademy.com/learn/web> — Beginner’s course on web development.

<https://www.khanacademy.org/computing/computer-programming/html-css> - Alternative

Reference Materials:

<https://www.w3schools.com> — Reference materials for CSS, HTML, JavaScript, and more.

<http://learn.shayhowe.com/html-css/building-your-first-web-page/> - CSS, HTML reference

<https://webdesign.tutsplus.com> - Several web design tutorials (free and paid)

Tools:

<https://www.google.com/webdesigner/> - Google web design tool

<https://material.io> - Google “Material Design” tools; the latest design trend

<https://kuler.adobe.com> - Adobe tool to find color schemes