

Administration

School: SDSM&T

Course: CSC 468: GUI Programming

Semester: Spring 2016

Professor: Dr. John Weiss

Assignment: Programming Assignment 3

Team: Website Organization and Administration

Installation and Configuration

To use this code during development, you will need to do some initial setup.

1. Enable Apache URL rewriting module
2. Enable .htaccess files

1. Enable Apache URL rewriting module

Apache must be configured to allow URL rewriting in order for this code to work correctly. To enable URL rewriting, follow the below steps:

1. Open your Apache configuration file in a plain text editor. (Google it if you don't know where it is.)
2. Find the line containing the text "LoadModule rewrite_module".
3. If the line has a "#" at the beginning, remove the "#" to uncomment the line.
4. Save the file and restart Apache.

2. Enable .htaccess Files

Apache must be configured to allow the contents of .htaccess files to override the default Apache configuration. This is necessary for our code to execute properly.

1. Open your Apache configuration file in a plain text editor. (Google it if you don't know where it is.)
2. Find the line containing the text "AllowOverride None". If this line does not exist, then .htaccess files may already be enabled. If this line is found, then it should exist between opening and closing "<Directory>" tags.
3. Change the line to say "AllowOverride All"
4. Save the file and restart Apache.

Development Guidelines and Requirements

The goal of this project is to build a website framework upon which the other teams in this assignment can develop their parts of this project. This document contains guidelines and specifications when generating HTML content for pages and for organizing your page files.

1. All page files and content placed in that page's assigned subdirectory
2. Only use relative paths

3. Custom CSS and JavaScript is OK
4. Avoid style attributes and tags
5. No header or footer sections
6. Use standard HTML structure for navigation
7. Do not use tables for layouts
8. Avoid size attributes
9. Name files using common convention

1. All Page Files and Content Placed in That Page's Assigned Subdirectory

Each distinct page for the site will have its own unique directory under the `pages` directory. All files relevant for the page should be included in that page's directory.

2. Only Use Relative Links

When referencing links or images on a page, always use relative links, never absolute links. This will make incorporating your team's code into the rest of the site far easier.

Example of relative links (DO THIS):

```

<a href="../submit/">Submit</a>
<link rel="stylesheet" type="text/css" href="css/styles.css" />
<script type="text/javascript" src="../shared-scripts/myscript.js"></script>
```

Example of absolute links (DO NOT DO THIS):

```

<a href="http://dev.mcs.sdsmt.edu/~1234567/pages/submit/index.php">Submit</a>
<link rel="stylesheet" type="text/css" href="/~1234567/pages/my-page/css/styles.css" />
```

3. Custom CSS and JavaScript is Okay

Any custom CSS or JavaScript files can be placed in the same directory as the page in which it is used and referenced as normal in the HTML of the page. There are no restrictions on this, but there are some things to keep in mind.

1. You can include multiple CSS files. Use this to your advantage.
2. Try to keep CSS for overall page separate from CSS for specific elements.
3. Remember to always use relative paths.

4. Avoid Style Attributes/Tags

When styling your elements, refrain from using the `style=""` attribute or from putting `<style></style>` tags in your HTML. Move these rules to a separate CSS file and apply them to elements using CSS classes.

Example of inline CSS (DO NOT DO THIS):

```
<p style="color: red; font-size: 1.25em;"> This is my most favorite paragraph! </p>
<a href="../submit/" class="red-button"> This is a link that looks like a red
button! </a>

<style>
a.red-button {
  display: inline-block;
  margin: auto 1em;
  padding: 6px 10px;
  background-color: red;
  box-shadow: 0px 2px 6px 0px rgba(0,0,0,0.5);
}
</style>
```

Example of better CSS practice (DO THIS):

index.php contents:

```
<link type="text/css" rel="stylesheet" href="styles.css" />
<p class="favorite"> This is my most favorite paragraph! </p>
<a href="../submit/" class="submit-button"> This is a link that looks like a red
button! </a>
```

styles.css contents:

```
p.favorite {
  color: red;
  font-size: 1.25em;
}
a.submit-button {
  display: inline-block;
  margin: auto 1em;
  padding: 6px 10px;
  background-color: red;
  box-shadow: 0px 2px 6px 0px rgba(0,0,0,0.5);
}
```

5. No Header or Footer Sections

When displaying a page, please avoid adding any page headers or footers. Such site-wide navigation elements will be included by a standard site template and are thus redundant.

6. Use Standard HTML Structure for Navigation

If you need to include navigation between multiple pages in your section of the site, please structure the HTML of your navigation menu as follows:

```
<nav class="section-nav">
  <ul>
    <li><a href="link-to-page">Link 1</a></li>
    <li><a href="link to page">Link 2</a></li>
    <!-- repeat as necessary -->
  </ul>
</nav>
```

Not all teams will need this, but if your pages does, please place it at the very top of your HTML content.

If you would like this menu to navigate to different sections on the same page, use the class `page-nav` instead of `section-nav`.

7. Do Not Use Tables for Layouts

Please avoid using `<table>...</table>` tags for laying out your page. Use `<div>` elements for grouping elements in a related section and `` tags for displaying lists of elements. These are far easier to style than tables.

If you need to display tabular data, such as numbers and calculations, using a table to organize the data is okay.

8. Avoid Size Attributes

When adding images and other elements to your page, avoid including sizing attributes, such as `width=""` and `height=""`; this is something that should be placed in the CSS stylesheets.

9. File Naming

When naming files, please use all lowercase names with words separated by dashes (-).

- Valid names:
 - *index.php*
 - *my-javascript-file.js*
 - *images/logo.png*
 - *images/secondary-logo.png*
- Discouraged names:
 - *index.HTML*
 - *MyJavaScriptFile.js*
 - *Images/Logo.png*
 - *Images/Secondary Logo.png*