COMP-5970/6970: Assignment 2

Assignment Submission Instructions

To complete this assignment, please follow these steps carefully:

1. **Develop Your Code based on the guidance (50% points):** Ensure your code meets the assignment requirements and is properly structured, documented, and tested.
2. **Publish Your Code on GitHub (50% points):** Upload your completed code to a public GitHub repository.
3. **Provide the GitHub Link:** Create a text file named **GithubLink.txt** containing the public GitHub repository link to your code. If you do not share the link, you will lose 40% points in section 2.
4. **Prepare Your Submission:**
    - Create a **ZIP file** under your **full name** and contain:
        - All your developed files for the assignment
        - The **GithubLink.txt** file:
        - This doc file that contains your solutions for "**Authoring World Wide Web Pages W3C Accessibility Guidelines"** section
5. **Submit on Canvas:** Upload your ZIP file to the designated Canvas assignment.

Note:
Failure to upload your code on Canvas or to provide a valid GitHub repository link in the GithubLink.txt file will result in a deduction of points. Ensure all components of your submission are complete and accurate to avoid losing marks.

Authoring World Wide Web Pages
W3C Accessibility Guidelines

Overview: The focus on the lab is on the creation of a website that follows the W3C Guidelines for Accessibility. Feel free to explore all the guidelines (https://www.w3.org/TR/2008/REC-WCAG20-20081211/).

It will be important for you to read the directions in this lab document carefully. Don't skip ahead and just type in the HTML code. You need to read the document to understand what the various tags are doing and why they are important to accessibility. You will be responsible for describing the various principles used in the W3C Guidelines and these principles are described in this document. Read and understand as you work through the activities.

The goal of this lab activity is to create a very simple accessible personal website that you will be posting live to the World Wide Web. Do not include content that you don't want everyone in the world to see or read.

1. Getting Organized

Organizing your files is critical to properly maintaining a web site. You will need to store your web pages in a folder. This "site" folder will contain ALL web pages, images, and resources needed by your site. For this lab we will call our site folder "MyWebsite" and store it on our USB drive (or your hard drive) so that we can have copies of the files to use when lab is over.

A. Guidelines for Completing the Assignment Using GitHub Codespaces

A.1 Set Up GitHub Codespaces:

- ✓ Create a GitHub repository specifically for this project.
- ✓ Launch GitHub Codespaces directly from the repository.
- ✓ In the Codespaces environment, create a folder named MyWebsite to organize all files related to the project.

A.2 Create and Edit HTML Files:

- ✓ Follow the assignment instructions to create index.html, hobby1.html, and hobby2.html.
- ✓ Use the Codespaces editor to write semantic HTML while meeting accessibility standards.

A.3 Preview the Website:

- ✓ Start a live server in the Codespaces environment (e.g., use npx serve or the Live Server extension).
- ✓ Use the provided URL from the live server to preview and verify the website's appearance.

A.4 Validate the Code:

- ✓ Upload the HTML files to the HTML Validator to ensure compliance with W3C accessibility guidelines.
- ✓ Address any validation errors or warnings before proceeding.

A.5 Publish Using GitHub Pages:

- ✓ Commit all finalized changes in Codespaces and push them to the GitHub repository.
- ✓ Enable GitHub Pages in the repository settings to publish the website.
- ✓ Share the live website link for review and evaluation.

Here's a tutorial video showing how to use GitHub Codespaces: YouTube Video.

B. Starting Your Website

B.1. In GitHub Codespaces, navigate to your project folder (or create a new one named "MyWebsite") to ensure it is active. Create a new file named index.html.
B.2. Open the index.html file in the Codespaces editor. Type in the following code exactly:

```
<!doctype HTML>
<HTML>
<BODY>

This is my website!

</BODY>
</HTML>
```

B.3. Save the file within Codespaces. Use a live preview server in Codespaces (e.g., npx serve or the Live Server extension) to view your webpage in a browser. Make note of the URL provided by the live preview server.
B.4. Make any necessary edits to improve your webpage. Save your changes and reload the live preview URL in your browser to see the updates.

Summary: You have created your first web page for your website! You have also used Safari to see what that page will look like once it is put on the World Wide Web (WWW). You will want to add some more content before you post it for everyone to see so keep going to find out how to add more information to your site.

The W3C Guidelines for Accessibility are based on 4 major principles: Perceivable, Operable, Understandable, and Robust. Each of these principles will be illustrated next with specific tags that help ensure that a website is accessible.

C. Principle 1 – Perceivable

This principle states that information should be presented in a way that is perceivable by any user. We will start with text information. A website that contains a lot of text should be segmented into sections using a header tag so that screen readers can easily identify sections to users browsing the site.

Header Tags

Header tags allow you to display text in a larger font, but more importantly screen readers will focus in on text in header tags when a user is browsing a site using a screen reader. Header tags are numbered H1-H6 where H1 will display in the largest font and H6 will display in the smallest font. The W3C Accessibility Guidelines say that you should use header tags to display hierarchical information.

C,1 Modify your index.html page as shown below, adding the text in red. You should modify the text to list your own hobbies and fun facts. Safari will not break text into separate lines unless you use a special
 tag. Remember this is your website so put the information that you would like on the site for everyone to see.

```
<!doctype HTML>
<HTML>

<BODY>
<H1>Hobbies</H1>
<H5>Cooking</H5>
I love to bake! My favorite thing to bake is cake.
<H5>Crafts</H5>
I enjoy knitting and crocheting. I also quilt when I have time.
<H1>Fun Facts</H1>
<H5>Birthplace</H5>
I was born in Oakland, CA but moved to Baltimore, MD when I was 2 years old.
<H5>Family</H5>
I have three sons who were all born in different states, but none of them where born in South Carolina where they grew up.
```

This is my website!
</BODY>
</HTML>

C.2. Saving and Previewing Your Webpage
Save your webpage in GitHub Codespaces. Reload the live preview URL in your browser to see your changes displayed on your webpage.
C.3. If you don't like the size of your headers, you can go back and change them. For example, maybe you want to use H3 instead of H5 (to make those lines bigger). Explore the different sizes and then preview them in Safari (or Chrome). When you have the text in the size that you like, continue on.

Summary: *You have added some content to your webpage using header tags. Header tags are recommended in the W3C Guidelines under the principle of "Perceivable" for hierarchical information so that screen readers can help users identify relationships between text elements on your site.*

D. Principle 2 – Operable
This principle states that your webpage should be easy to navigate. Many of the advanced features like forms, buttons, and other GUI components are beyond the scope of this lab but you will explore some basic tags suggested by the W3C for this principle.

Head and Title Tags
The first tag that relates to this principle is the TITLE tag that puts a title at the top of your browser window. Again, screen readers will identify your window using this title so it is important to have a descriptive title. The TITLE tag must appear inside the HEAD tag (as opposed to the BODY tag that contains the content of the window for your page.).

D.1. Add a descriptive title for your webpage by including the following code inside the <HEAD> tag after the opening <HTML> tag in your index.html file:

<HEAD>

<TITLE>Your Website Title</TITLE>

</HEAD>

D.2. Save your file in Codespaces and use the live preview server to confirm that your title appears at the top of your browser window.

E. Principle 3 – Understandable

This principle says that the information and user interface navigation need to be understandable. One of the guidelines for this principle suggests that the navigation used to switch from page to page in your website should be the same on every page. Users should not have to hunt around your page to find out how to navigate your site. Before we use this principle we need to add some pages to your site.

Adding more pages

You already know how to add pages to your website folder in *GitHub Codespaces*. See if you remember how to do the steps below. If you forget how to do any of these steps, refer back to the sections you completed earlier for help.

E.1 Create a new file in your MyWebsite folder and name it hobby1.html. You may want to copy and paste from your index.html page since the starting tags at the top of the page should be the same and you will want the same title on the window.

E.2 Pick one of your hobbies and provide a fuller description of what you do with this hobby. Feel free to use some header tags and type descriptions within the BODY tags. Save your file as you work.

E.3 View your new hobby1.html page in Safari. How does it look? Don't worry if it looks a little boring since it only contains text. We will learn how to make your pages more interesting later in the lab.

E.4. Create another new file in your MyWebsite folder and name it hobby2.html.

E.5 Pick another one of your hobbies and fill your third page with some content about this hobby. Save your file as you work.

E.6 View your new hobby2.html page in Safari.

Anchor Tag: Navigating to New Pages

Now we want to provide links between the hobbies you listed on your index.html page and your new pages so that when the user clicks on the text your new page is displayed. You need the anchor tag to accomplish this. The anchor tag has the following format:

<a HREF="hobby1.html">Cooking

E.7 Return to your index.html and find where you listed your hobbies. They should be inside of an H6 tag. You will want to put the anchor tag inside your H6 tag so it will look like this: <H6><a HREF="hobby1.html">Cooking</H6>. Don't forget the closing tag! Add an anchor tag for both of the hobbies you expanded on in hobby1.html and hobby2.html. Save your index.html file.

E.8 View your index.html page in Safari and click on your hobbies to see how you can now navigate from your index.html page to your new pages. The only way to return to your index.html page is to hit the Back navigation button in Safari. We need to fix that next.

Site Navigation

The W3C Guidelines within the Understandable Principle suggest that navigation is consistent across your pages. We do not want users to have to rely on the Safari Back button so we will add our own button and be sure that this button is in the same place on each page. We will use the FORM tag to create a button with the word "Home" on it. When the user clicks on the button the action will be to link to the index.html file. The form and button tags will have the following format:

```
<form style="display: inline" action="index.html" method="get">
    <button>Home</button>
</form>
```

E.9 Open your hobby1.html file and place your cursor right after the BODY tag. On a new line add your form tag as shown above so that Safari will put a button with the word "Home" on it at the top of your webpage.

E.10 Add a nice horizontal line below your button to help separate the navigation area from the rest of the page. The <HR> tag serves this purpose by adding a line or "horizontal ruler". Add a new line below the closing form tag and simply type in the <HR> tag on that line.

E.11 View your index.html page in Safari and see if you can navigate back and forth between index.html and hobby1.html.

E.12 Repeat this process in your hobby2.html file. You want your buttons to be in exactly the same place on each page!

Summary: *You have a real website now with three different pages that are linked together using a common button for navigating. You are now ready to add some more interesting content to your site but first let's make sure your site is Robust!*

F. Principle 4 – Robust

The final principle says that your website should be robust so that it is viewable by multiple browsers and multiple assistive technologies. If you follow the previous guidelines you should be well on your way. There are validators online that can check your webpages to determine if they are following the W3C Guidelines.

F.1 In Safari, visit validator.w3.org. This website allows web developers to validate their webpages before posting them online to ensure they are accessible to all.

F.2 Select the tab Validate by File Upload and a window should appear allowing you to select a file.

F.3 Click on Choose File and navigate to your index.html file in your MyWebsite folder.

F.4 Click on Check and make sure you do not have any errors on your page. You will probably get a warning that you haven't declared a language in your HTML tag. You will fix this for homework. You will also probably get an error that you haven't defined your character encoding. This is easy to fix and ensures that a browser knows that you are using ASCII characters (rather than more sophisticated text codes that we learned in class!). Add the tag <meta charset="UTF-8"> within your HEAD tag. This identifies the Unicode Transformation Format with 8 bits as the character set which is standard for HTML files. Remember that UTF allows extra characters that are not provided in standard ASCII.
F.5 Add the META tag to your other two pages and run them through the validator.

Summary: *You now have a website with 4 linked pages that follow the W3C Guidelines for accessibility! Unfortunately, only you can see your webpages because it is located on your USB drive rather than a webserver. You are ready to post your website to the World Wide Web so that others can see it.*

G. Posting to the World Wide Web

You simply need to transfer your files to the appropriate location on the server and your site will be "live".

G.1. Use GitHub Pages to publish your website:
Commit all changes to your repository in Codespaces.
Push the changes to the main branch of your repository.
In your GitHub repository settings, go to Settings > Pages.
Under "Source," select the branch where your files are located and save your changes. This will enable GitHub Pages for your repository.
G.2. Once GitHub Pages is enabled, a live link to your website will be generated. Test the link in your browser to confirm that your website is live.
G.3. Upload any remaining files, and ensure all links work correctly on the live site. Share your live website link as required.
G.4 Download the remaining pages to the server and test to make sure your pages still link correctly. Then, you can send your URL to all your family and friends!

Summary: *You now have several linked pages on a website that is public for anyone to view. You can continue working on your website by changing pages on your USB drive but the changes will not be live until you download the page to the cs.furman.edu server as you did in this step.*

H. Adding More Content (While Staying Accessible)
The next part of the lab will give you an opportunity to add some additional elements to your website that will make it more interesting. Some of these elements, like pictures, are often difficult to make accessible to everyone so there are some W3C Guidelines to help us. None of the techniques that are suggested in the W3C Guidelines are difficult but they should always be used so that all the elements on your website are accessible.

Images
Images are all over the World Wide Web and make our websites more interesting to browse. However, they are frustrating to visually impaired users of the web. It is very easy to fix this problem by providing text descriptions of an image when we add it to a webpage. This happens to be one of the guidelines under the Principle 1 – Perceivable. Non-text content should always have a text alternative.

H.1. Find an image that you want to put on your website. It might be a picture of you or some other picture that you download off the web. You need to put this image in your MyWebsite folder. Any image on the web can be copied and saved to your USB drive by Control-Clicking on the image and then saving it to your folder.
H.2 Open the web page that should contain the picture in Safari so you can visually see where you want the picture to display.
H.3 Now, open that web page in VSC and locate in the BODY tag where the image should go. Click your cursor in that location in your HTML code and on a new line enter the following IMG tag:

```
<img src="dog.jpg" alt="golden retriever dog">
```

Where you replace dog.jpg with the name of your image file and replace "golden retriever dog" with a text description of your picture. Save your webpage in VSC.

H.4 Reload your webpage in Safari and your picture should appear now. If the picture is too big go back to VSC and add height and/or width properties to your IMG tag. For example, add height=100 width=100 to keep your picture to 100x100 pixels. However, if your picture is too small you need to get a different picture. You don't have enough pixels to appropriately display on a website.
H.5 Once you get the picture looking nice on your webpage, upload it to the cs.furman.edu server and browse it using your URL.

Summary: *You can add as many images as you want to your website. Just be sure that you use the alt property to provide a text description so that your website is following the Perceivable Principle.*

Adding Color - bgcolor and font tags are obsolete … need to update
Although your images probably added a little color to your website you will probably want a little more color to make your pages more appealing. We will explore a few options now.

H.6 You can easily change the background color of any web page by adding the property bcolor to your BODY tag. Open your index.html file in VSC and locate your BODY tag. Modify it so it looks like the following:

```
<body bgcolor=”33FFDA”>
```

Save your index.html file. Do you recognize the 33FFDA? This is simply hex code for a color giving the red, green, and blue values (i.e. 2 hex digits per primary color).

H7 Open your index.html file in Safari and you should now have a different color background for the page. If you don’t like that color visit http://htmlcolorcodes.com and you can pick a color from the color wheel and the site will give you the hex code for that color to put into your bgcolor property.

H.8 Change the background colors for all the pages in your website. Don’t go crazy and make each page a different color … Your site will look more sophisticated if you stick with a uniform color scheme.

H.9 Add a new section to the bottom of your index.html page with the heading “Likes/Dislikes”. Underneath this heading add a list of things your like and dislike in no particular order. You can use the LI tag to do this. The LI tag allows you to put a bulleted list on your website. My list might look like this in my HTML file:

```
<H1>Likes/Dislikes</H1>
<UL> <LI>Pizza
        <LI>Oatmeal
        <LI>Babies out of carseats
</UL>
```

H.10. Now we want to add some color so that the things we like are written in green and the things we don’t like are written in red. You can surround any text with the FONT tag to change how it is displayed. For example, to make the last item in my list red I would modify the line as follows:

```
<LI><font color=”red”>Babies out of carseats</font></LI>
```

Notice that I can actually use words to describe color, or I can go back to my color wheel and find a hue of red that I like and type in the hex code rather than the word red (i.e. color=”FF5233”). Change your items so that they appear red or green depending on whether it is a like or dislike. Save your file.

H.11 Open your index.html file in Safari and you should see your likes and dislikes in different colors. Is your website accessible? Will someone with a visual impairment be able to tell which of your items in the list you like and which you don't like? Your answer should be NO! With this design you are using color to give information so someone who can not see (or even someone who can see but is color blind) will have a difficult time understanding this information. Hence, you are violating the Perceivable Principle as there is information on your site that is difficult to perceive. A screen reader may read the colors of the text but is it clear that red is "like" and green is "dislike". A better design would fix this problem.

H.12 Redesign your index.html content so that your Likes and Dislikes are more clearly identified without relying on color. Save your file.

H.13 Validate your website to make sure it is following the W3C Guidelines.

Summary: Color is an important feature to make your website more visually appealing. You now know how to add background colors and font colors to text on your site. Use these features appropriately and your site will be both visually appealing and accessible.

Linking to other pages on the WWW

I am sure you have seen examples on the web where someone has used a URL as the text to link on a webpage. This violates the Operable Principle in the W3C Guidelines and is also annoying to everyone! This is an example where an accessibility Guideline helps everyone.

H.14 Find a website on the web that relates to something in your website. For my website, I found a website about quilting that I want to link to.

H.15 Use the anchor tag to add a link to this website somewhere in the text for your description of that hobby. My text might look like this:

I also quilt when I have time. I get ideas from
The Spruce.

H.16. Save your index.html file in VSC and then open it in Safari. Try clicking on your link to make sure it takes you to the other webpage.

Summary: *You should always use descriptive text when linking to anything on your webpage so that your website is following the Operable Principle.*

Reference

CS0 for Accessibility Resources by *Paula Gabbert https://bookish.press/tac*