Week 5/6 – Website Design and Hosting

Lab Assignment – 100 pts

Due 11:59 pm November 4, 2019

Objectives:

1. To practice the professional practice of paired programming.
2. To learn about how the components of a webpage, namely HTML, CSS, and JavaScript, combine to make a functioning site. (Week 5)
3. To learn about serving dynamic webpages and interacting with databases. (Week 6)

Procedure:

You will be working in teams of 2 for this lab. You will be using a professional practice called Paired Programming in which two people work together on one file. Both partners will hold both roles at various points and will switch roles on request or about every 15 minutes. The Driver is the partner with the keyboard. The Driver is primarily concerned with the details such as syntax and keywords. The Navigator is the partner who is ‘over the shoulder’ of the driver. The Navigator is primarily concerned with the big picture of the problem being solved or the task being performed. However, the Navigator has to be following along with the Driver, so they can take over as needed without having to be caught up. In a good Paired Programming experience, the two partners will be in constant communication. In industry this leads to fewer errors and more robust development projects. In an academic environment it leads to deeper understanding in less time for both partners.

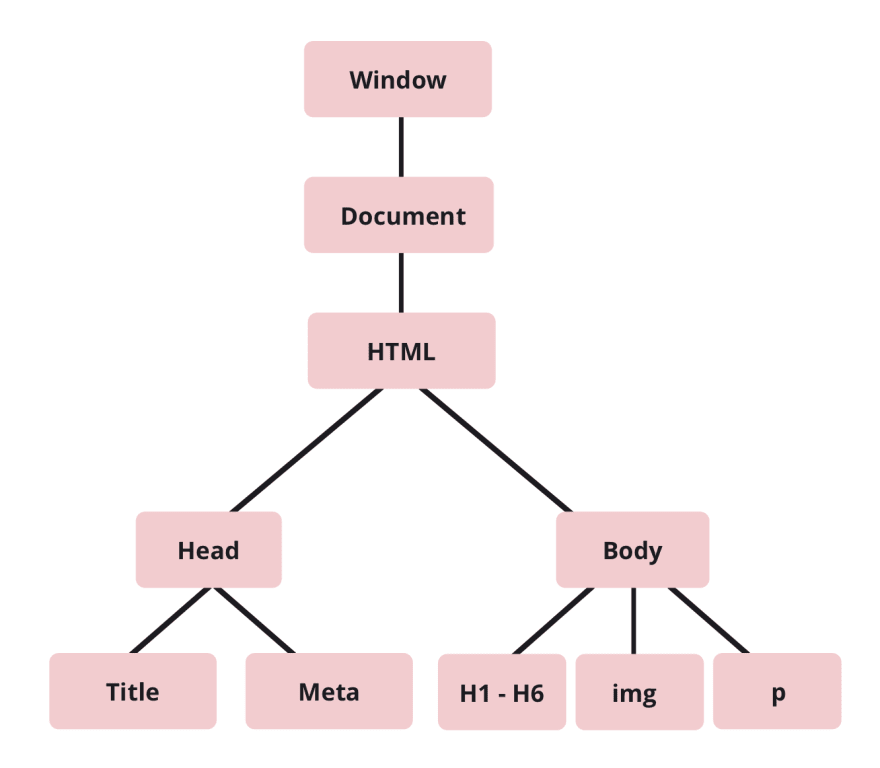
Since both people will work on both parts, it would make sense to put the answer sheet in your group’s git repository. This ensures that everyone will always have the most up-to-date version.

Before we begin, it should be noted that this lab has some required programs to run it properly. You will use remote desktop to Drexel’s Unix server (Tux) which is an environment that has all the required software already installed. You may have already linked your Gitlab to Tux, in which case you are good to go!

If you previously connected your git to your own computer, you will need to go through the ssh process again from tux.

Your lab TA will demonstrate how to create a new repo for this project and add your partner, your grading TA, your lab TA’s and your professor to it as developers. You will then clone the source code from another repo into your own.

HTML, CSS, and JavaScript (Week 5)

1. Getting Set Up
   1. You will access tux through a remote desktop application (instead of Putty or ssh). On Windows the application comes pre-installed, but on Mac you must download it. Search for **Microsoft Remote Desktop** on the Mac OS app store and it should come up as the first option.
   2. Once it’s open, you will be prompted for a computer name. Enter **tux2.cci.drexel.edu**.
   3. You will then be prompted to log on, this is just your normal tux logon username/password. (Note, it can be glitchy and prompt you to enter your password multiple times. This does not mean you messed it up. After a few tries it will let you in).
   4. You should now see a desktop not too unlike your own.
   5. If you right click on the screen, a menu will open. One of the options is to open the terminal. If you select that option, you will get a terminal just like PuTTY or ssh. The advantage of the remote desktop is that you also have a visualization of the file system and can open a browser to look at / interact with your website.
   6. Answer Question 1 in the Answer Sheet
2. Create Your Group and Clone the Repository
   1. Choose one partner, the driver, to create a group in GitLab that includes both partners, the lab professor, the grading TA and the lab TA’s as developers.
   2. Using the terminal, the driver will clone the repository at <https://gitlab.cci.drexel.edu/trp74/CI101Week56Lab> into your directory.
   3. Next, you have to disconnect the cloned repository from the original. Make sure you are in the CI101Week56Lab directory. Use the command **git remote rm origin**.
   4. You also need to delete the old .git folder. Use **rm -r .git** to do this.
   5. Now in GitLab make a new repository, but don’t put anything in it.
   6. Back in the terminal, in the directory of the folder, type **git init**. This tells Git that the folder will be used for a git project.
   7. Next you will stage the files ( **git add .** ), and commit them ( **git commit -m “**first commit message” ).
   8. Copy the ssh or http url from your new repository. In the terminal type (**git remote add origin your repo url** .If you would like to check that it worked, use **git remote -v**. You should see the new url under origin.
   9. The last step is to push your changes using **git push origin master**
   10. You will also want to go into the package.json file and change the git url to your new repository.
   11. Paste the link to your repo in the answer sheet question 2
3. The Project Structure
   1. This project contains a lot of files, take a minute to look at the structure. While these files are all important, most won’t be touched in the next two labs. The important ones for our purposes are:
      1. Index.html – This is where all of the HTML goes. The HTML is the skeleton of the website, keeping everything organized and in place. To open the file for editing, right click and choose open with notepad++ (or another editor). If you just click on it, it will try to open the rendered version of the page in the browser, which is not what we want.
      2. Index.js – This is one of the two JavaScript files in the project. This file is used for storing JavaScript that will help with website functionality.
      3. Stylesheet.css – This is where the CSS goes, the CSS controls all of the stylings of the website. Things such as color, font, and relative positioning all fall under this category.
      4. The other two files to note are server.js, and CLCBooksSQL.sql, but we will not work with them until week 6.
   2. After exploring the file structure, answer question 3 on the answer sheet.
4. HTML
   1. As mentioned earlier, html is the structure of the website. It’s all of the content, but without positioning or styling. If you had just html on your webpage, it would be very black text on a white background. HTML is written in what we call **tags**. A tag is a keyword enclosed by < and >. Most tags need to be closed after the content that goes into them is finished. This is done using </ the tag name again >. Different tags have different behaviors. Some of the most common ones are:
      1. There are comments in html. Comments are denoted by <!-- to open, and --> to close. Everything between that opening and closing gets ‘commented out’ and will no longer be shown if the webpage is loaded on a browser.
      2. <div> – the basic box tag. It purposefully has very little behavior of its own, as it’s manipulated to do what the user wants. Can contain text, an image, or even other tag elements within it.
      3. <h1> – this tag is in a group with 5 others (h1, h2,…, h6), but they all do basically the same thing. They are headers, or titles. If you need big text for a title, you can use one of them. They are in decreasing order of size (h1 is the biggest), but all are bigger or at least bolder than normal text.
         1. Open index.html in an editor like Notepad++. Find the comment that says Question 4a and add your names to the document between the opening h3 and closing h3 tags. Replace the underline. When done, save the file and check it in a browser. Once you have confirmed it works, copy it and paste it in the answer sheet under Question 4a.
      4. <p> – the paragraph tag. This one is used for text. You can think of it almost like a textbox.
         1. In index.html, find the comment Question 4b. Write a paragraph that introduces a site visitor to the page. Don’t forget to add the opening and closing paragraph tags. Once you have confirmed it worked, copy it and paste it in the answer sheet under Question 4b.
      5. <img src=”#”/> – this tag is different from the other ones on the list in two major ways, firstly it’s self-closing (meaning that it doesn’t need an </img> tag to close it), and secondly it has an attribute. In this case the attribute is src, which stands for source. In between the quotes, you put the address of an image, whether it’s a URL or file path. The tag knows to go to that source and pull the image. Similar tags are <link> (which uses href instead), and <script>, which loads a JavaScript into the HTML page.
         1. Find the comment Question 4c. You will need to find the logo choices in the files provided and agree between the two of you which one you want to use in your site. Replace the # with the relative address to your chosen logo. Once you have confirmed it worked, copy that line and paste it in the answer sheet under Question 4c.
      6. <html>, <head>, and <body> – These three tags are also special in that they show up in every html file. The <html> tag wraps the entire page, and the <head> and <body> tags separate the content. The <head> tag contains information about the page such as the character set in use and loads dependencies. The <body> tag is used for the actual content. Everything you see on a webpage is in the <body> tag. Along with these three tags, you must also always include <!DOCTYPE html> as the first thing on the page. Together they keep the structure of HTML uniform across developers, browsers, and generations.
      7. The same way we can “link” an image to the index.html file, we can “link” other types of files. The link element allows us to link supporting files, such as CSS, to the html. The rel attribute holds a value indicating what type of supporting file is being linked while the href attribute holds a value that is the location of that file.
         1. Find the comment Question 4d. Add the name of the css file that is in the project. Save the html file and test it in a browser, then go to the Answer sheet with your edited line of code.
   2. The combination of opening tag, content, and closing tag is often referred to as an element. These elements comprise what is called **The DOM** or **D**ocument **O**bject **M**odel. The browser keeps track of the hierarchy of elements in the DOM. Since elements can be inside of each other, it’s important to keep track of who is who’s parent. Languages like JavaScript use the DOM to interact with html elements. Unfortunately, with the DOM you can only see inward. From a given element, you can find it’s children, but not its parent.
   3. 
   4. The table in this file is an example of elements nested inside of each other. The table contains rows and each row contains cells.
      1. Add a new row to the table, including cells with the name of the book, the author, the isbn and the books availability. Every piece of information should line up with the others above it. If you don’t have a favorite CCI type book that you want to add, you can find all this information from an Amazon search. When done, save and check your new addition in a browser. When it works and looks good, copy your new row and take it to the answer sheet.
5. CSS
   1. CSS is where the styling lives. If you want an element be in a different position, or have a different color, you want to use CSS.
   2. In CSS you target elements based on their tag, id, and or class. While the tag is implicitly defined when you add an element, ids and classes are added by the programmer as attributes. If you want to add an id or class in html, you simply include it <div id=”X” > or <p class=”Y”>
      1. Id – must be unique within the DOM, so only one item can have a specific id. This makes them useful if you want to target a specific item of many.
      2. Class – they are basically the opposite of ids. If you target a class in CSS what you add will affect all of elements that have that class.
   3. The general format of a CSS rule is the name of the element(s) you’re trying to target, opening curly brackets, the properties you wish to change for that element, and closing brackets.
   4. 
      1. Open the stylesheet.css file in an editor. In the html document, the header row of the table has an id attribute called “header-row”. Find the corresponding CSS rule. It has a comment with Question 5a. Change the color to one that works well with your chosen logo. Save both files and test the changes in the browser. When you are happy with your choice, copy the rule from the CSS and take it to the answer sheet, question 5a.
      2. Change the background color of the next rule (.rows). On the answer sheet under 5b answer the questions and paste the changed rule.
      3. Any designer will tell you that whitespace or negative space matters. Find the comment with Question 5c and play with the margin and padding. Then go to the answer sheet and share your insights.
      4. Many CSS rules simply apply new style rules to the existing html tag. Let’s add a subtle background image to our page. Add a background-image to the body element. You can look up how to do this on w3schools.com. You can find a good image by searching for images with the keyword “seamless”. Save and check your page. When you like the result, copy and paste the rule into the answer sheet under Question 5d.
6. JavaScript
   1. JavaScript is the action center of web development. Just HTML and CSS gives a purely static website. It might look nice, but it would have little to no functionality. CSS can do some animation and movement, but nothing like JavaScript. HTML and CSS are not actually programming languages, and you can’t use them for logic. JavaScript comes in to solve these problems in a big way. By having access to the DOM (through document.getElementById(“id-X”)), JavaScript brings websites ‘to life’.
   2. Since this is just an introductory lab, we are not going to dive too deep into JavaScript, instead showing one example of how to attach it to an already existing element. Some tags, like button, have built in interfaces with JavaScript. Button for example has the onclick attribute which lets it perform an action when a user clicks it. This attribute can take a JavaScript function as an argument, calling the function when it’s clicked.
   3. A lot of people start off with what is called inline JavaScript. You can technically write your code inside of the HTML files as long as you use <script> tags. While this might be easier for a beginner to understand, it’s a really bad habit. It’s very easy for files to get large and unruly, and everything generally works better when it’s split up into parts.
   4. We will apply an existing function to an existing button. Open the index.html page and find the comment with Question 6 in it. Replace the # with the goToCCI() function call. Go to the Answer Sheet and answer question 6, include your changed line of code.

What to Turn in:

Nothing this week. Keep your answer sheet in your repo so both partners can access it. You will add to it next week. Instructions for week 6 will be posted on Blackboard in week 6.

How you will be graded:

Your answer sheet must be submitted to Blackboard on time for full credit to be possible

You will be graded on the correctness of your edited code and the completeness of your answers on the answer sheet. You will also be graded on your implementation of paired programming.