# CTI 110

# M4HW1: Java House Coffee Shop, Part 1

# Introduction

Congratulations, you’ve been hired to create a web site for the local coffee shop, Java House! Now you’ll need to learn how to make that site.

Fortunately, you’ll have some guidance from this document and from the website W3Schools (<https://www.w3schools.com> ).

# Explanation

In the two homework assignments in this module, you’ll build the two most important pages that Java House needs. (You know they’re most important because in your negotiations with them, they told you they needed them first, and could wait on the rest of the site.) Those pages are:

* A “home page” or “landing page”, leading to the other pages. (This will have some basic information about the coffee shop.)
* Their menu (they get a lot of phone calls about this, and are hoping to save their customers confusion)

For this assignment, you’ll just be building their home page. This version is pretty plain looking – this early, they just want to make sure that the information is available online. This kind of prototype version is sometimes called a “wireframe”, so you’ll sometimes hear the verb “wireframing” used to describe the process of laying out a site in its early stages.

# Assignment Directions

**Setting up your project**

First, go to where you keep your CTI 110 files (such as a **cti110** folder on your flash drive or other storage.) Create a folder named **m4hw1\_lastname** (using your own last name). This is known as the “project folder” or the “assignment folder”.

The project folder will hold all the HTML files used for the website (and, later, extra files such as images, stylesheets, and scripts.)

For this assignment you’ll want to use a **programmer’s text editor.** The programmer’s text editor we use in class is Notepad++, and that program is free, so you may choose to install that. (Notepad++ is Windows-only. If you’re on a Mac, consider Sublime Text or Atom as alternatives.)

**Creating the home page**

Open your chosen text editor. Start with an empty file, and save it as **index.html** in your project folder. (Make sure that the file type is set to HTML, not plain text. If the file is named something like **index.html.txt** then your file type is incorrect!)

**Creating the basic wireframe**

Before we start building the page, we’ll add a comment block. HTML comments <!-- look like this --> , and everything between the <!-- and --> tags is ignored by the browser.

Use the standard comment block text, using HTML comment tags. It should look something like this:

<!--

CTI 110

M4HW1 Java House Home Page

Your Name

Date

-->

Consult W3Schools and the slides for the basic structure of an empty web page. This includes the **doctype** information, which comes first, and <html> and </html> tags at the beginning and end of the document.

Inside the <html> tags, add a <head> and <body> element. Each element should have an opening and closing tag.

Inside the <head> element, add the web page title. Consulting W3Schools, we see that the title element is called <title>. Use a descriptive title, as this is the text that shows up in the browser tab. The company name is a good choice.

Inside the <body> element, the first item should be a **header**. This involves adding the opening and closing tags as follows, inside the <body> element:

<body>

<header>

</header>

</body>

(Be sure to not confuse “header” with the “head” element. The <head> comes *before* the <body> tag, while header tags are always *inside* the <body> tag.)

Next, inside the header element, we want to place header text. From W3Schools, we see that the largest header text uses the <h1> tag, and smaller headers use larger numbers. Since this is supposed to be eye-catching, we use **heading 1**, or <h1>. Again, the company name is the right choice here for the text inside the header.

After the header, we’ll place a navigation element. (This will consist of several links near the top of the page that the user can click to navigate from page to page.) This uses the <nav> element. For now, we’ll just put an HTML comment that says “Navigation” inside the <nav> tags. In the next section, we’ll actually add the navigation links.

Next comes the main content of the page. Use the <main> tag to create a portion of the page that will contain this content. Place an HTML comment inside this tag pair that says “Main content”.

Last is the footer. This uses the <footer> element. Again, add a HTML comment inside the footer.

Save your page, and open it in a Web browser. You should be able to see the company name in the title (the text in the tab at the top of the browser, above the page) and in the header (in large text at the top of the page itself.) If not, go back and check your tags.

Next, we’ll add the actual content inside the page.

# Adding content

**Navigation**

Go back to the navigation element. Inside it, place the following bolded text (using the <b> tag):

**Home Menu Music Jobs**

If you save and preview the page, you’ll find that even if you add several spaces between these words, the browser will still only show one space. To better space the links, use the non-breaking space character &nbsp; several times between each word.

Next step is to turn those pieces of text into hyperlinks using the anchor tag. Remember, the anchor tag <a> turns text into a link, but it needs to have an href portion added inside the tag in order for the link to point somewhere. Make the links point to the following pages:

* **Home** links to index.html (this page)
* **Menu** links to menu.html
* **Music** links to music.html
* **Jobs** links to jobs.html

Obviously only one of those pages exists so far, so the other links will give you an error if you follow them. That’s fine, we’ll add pages for them later.

**Main Content**

Find the main tag you used for the main content. The majority of the page text will go inside here.

Add a header tag using the second-largest header size. Inside it put:

**Relax at the Java House**

Underneath that, add a paragraph tag. Place inside that tag this text in **bold:**

**Located in historic downtown Fayetteville, Java House has been open since 2007.**

After that introductory sentence, the client wants a short list of reasons to come to Java House. (This is a good decision, because a home page should grab user’s attention quickly. An overly wordy home page can lead to what’s called a “bounce”, where the user loads the page but doesn’t read it or click the links.)

Create an unordered list. (Remember, there is one tag that describes the list, and a different tag that describes list elements. You have one list with multiple elements.)

Add to the list these items:

* Specialty coffee from around the world
* Muffins, bagels, and sandwiches
* Live music
* Open Mic Thursday

Now go to the bottom of the <main> element, just before the closing tag, and add a <div> element. This will contain the coffee shop’s contact information as follows:

Java House

323 Hay Street

Fayetteville, NC 28301

(910) 555-7890

Open 8am-10pm

Your line breaks should match the ones shown, which means you’ll need to add a line break tag after each line. (Note that line break tags break the usual rule that tags need to be paired.)

**Footer**

The final content within the <body> element is the footer. This contains copyright information and contact information for the webmaster (you).

The footer should have two lines, separated by a line break tag.

The first line of the footer should say:

© 2017 Java House Inc

(You’ll need to look up the HTML symbol used to create the © symbol.)

The second line of the footer should display your student email address. You should then turn it into an email link (using the <a> tag with a href that uses the mailto: protocol.)

**Testing the page**

Save your HTML file. Open it in a a browser (in Notepad++, the menu option **Run -> Open with [browser]** is a useful shortcut). If you’ve already got it open, and you don’t see your latest changes, do the following:

* Make sure you’ve saved the HTML file recently (you can check the timestamp on the file)
* Refresh the page in the browser (keyboard shortcut F5)

If something looks incorrect in your version, double-check your work against the directions. Missing a closing tag or placing content in the wrong tag is the most common error.

# Submitting M4HW1

All your work should now be in a folder named **m4hw1\_lastname**. (If it isn’t, rename the folder now.)

Go to GitHub, and navigate to your **cti110** repository. Select “upload files” and upload the entire folder (drag and drop is probably easiest).

Finally, zip up the assignment folder. On the Blackboard submission page, submit your **m4hw1\_lastname.zip** file in the appropriate location.