INFO 101 Class Six

Thurs. April 14, 2022
Topic: Builder #3

Quiz Time!

01.

Builder #3
Stage 1

test.css

This is what you'll be creating

Step 1.1

1. When writing web pages, it is necessary to keep all relevant files in the same folder so that they can reference each other and stay organized. On your computer, create a folder called "Info101 Activities." Under that folder, create another folder named "Intro to CSS"

2. Right-click introToCss.html link and "save as" in your "Intro to CSS" folder. This is the file you will be formatting with CSS.

3. Download the images named icon.gif and price.jpg into your "Intro to CSS" folder. These will be displayed in your page.

Step 1.2

1. Open introToCss.html in your text editor (VSC)
2. Take a look at the HTML code. Notice how the code is divided into sections using div tags with ids.
   a. The div tag surrounds a set of other tags that you want to format as a set. The id attribute identifies the div and distinguishes it from other div tags.
3. Open introToCss.html in your browser. Take a look at what this web page looks like with no CSS applied to it. As we go on, the page will get more and more styled.
4. In your text editor, create a new file and name it **"test.css".** Save it in the "Intro to CSS" folder.

How does a CSS file work?

In a CSS file, you "target" HTML elements to receive the formatting that you want. There are a number of ways to do this targeting. In this activity, we will us tage names and ids to target styling.

The illustration on the next slideshows how different parts of the CSS file target/correspond to different HTML elements.

CSS

```css
body{

}

p{

}

h3{

}

li{

}

#Header{

}

#Header h1{

}

#LeftNav{

}

#LeftNav img{

}

#LeftNav ul{

}

#LeftNav h3{

}

#Content{

}

#Footer{

}
```

HTML

```html
<!DOCTYPE html>
    <head>
        <title> Keith the QB  </title>
        <link rel="stylesheet" type="text/css" href="test.css" />
        <link href='http://fonts.googleapis.com/css?family=Passion+One' rel='stylesheet' type='text/css'>
    </head>

    <body>
        <div id = "Header">
            <h1> Keith the QB </h1>
        </div>

        <div id = "LeftNav">
            <h3 class = "titles"> My Favorite Websites </h3>
            <ul>
                <li><a href="https://www.theverge.com">The Verge</a></li>
                <li><a href="https://www.surfline.com">Surfline</a></li>
                <li><a href="https://www.worldstarhiphop">WORLD STAR</a></li>
                <li><a href="https://www.amazon.com">Amazon</a></li>
                <li><a href="https://www.theonion.com">The Onion</a></li>
            </ul>
            <img src="icon.gif" width="350px" height="200px">
        </div>

        <div id = "Content">
            <p> This is Keith Price's number one Fan Site! He is a good leader, wins games, throws awesome passe:, h:
            Huskies have been on point this season with a starting record of 4-0. The Huskies smashed Boise State, p
            br /> <span id="bow">Bow Down to Washington!!</span></p>
            <img src="price.jpg">
        </div>

        <div id = "Footer">
            <h3 class = "titles">Connect With Me Online:</h3>
            <ul>
                <li><a href="https://www.Facebook.com">Facebook</a></li>
                <li><a href="https://www.Linkedin.com">Linked In</a></li>
                <li><a href="https://www.Spotify.com">Spotify Playlists</a></li>
            </ul>
            <p>Copyright of Keith the QB</p>
        </div>

    </body>
</html>
```

How does a CSS file work?

Notice the basic structure of the CSS file

- The file is broken into separate sections (body, p, h3, etc.). Each section targets a different HTML tag
- Each section has an optional symbol (# in this example), a name (body, p, h3, etc.), and a set of curly braces{}
- You will soon see the curly braces get filled with styling commands

Targeting by name, by id, and by both are shown in the illustration:

- You can target a tag by writing down the tag name
- You can also assign ID's to tags on HTML that you can reference from your external CSS. ie(HTML id = "header" | CSS id = #header)
- You can style child elements by referring to them like this on the external CSS:
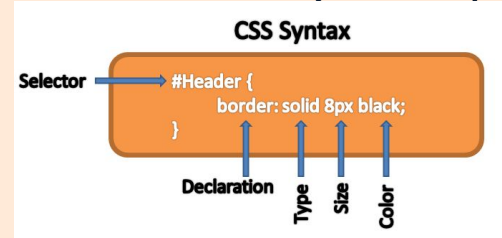 - #Header h1 {}

Step 2: Link your HTML and CSS Documents

Copy down the CSS exactly how it looks

Once you've copied:

In activity 2 I mentioned that you can link your HTML document to a CSS file in the <head> </head> section. To do this, copy this down in your <head> tag:

```
<head>
    <title> Keith the QB  </title>
    <link rel="stylesheet" type="text/css" href="test.css" />
</head>
```

Step 3

1. Find the **#Header** section of test.css and add the border style.
2. Save test.css, then refresh intro ToCss.html in the browser
3. Add the same border style to the **#LeftNav**, **#Content**, and **#Footer** divs
4. Change the width for the **#Header** section to 500px
   - **#LeftNav** width: **400px**
   - **#Content** width: **700px**
5. Add margin styles (Basically copy the image on gClass):
   - margin-left and margin-right auto centers the div
   - The margin-bottom of 10px makes space below the div
6. Feel free to mess around with the other divs to see how the margins affect them.

CSS Syntax

Selector → #Header {
    border: solid 8px black;
}

Declaration   Type   Size   Color

```
6   #Header{
7       width: 500px;
8       margin-left: auto;
9       margin-right: auto;
10      margin-bottom: 10px;
11      border: solid 8px black;
12  }
```

Step 4: Position the divs

1. CSS float pushes elements to the left or right, allowing other elements to wrap around it. Try this with **#LeftNav** div and see what happens:
   - float: left;
2. Your page should look similar to this (**Note: experiment with margins on the #Content**):

Step 5: Style Content Inside the Divs:

1. Align text by using **text-align** declaration with the values of **center**, **left** or **right.**
2. Center align text in your **#Header, #leftNav, #Content,** and **#Footer sections.**
3. The centered list looks a little awkward, remove bullets by using **list-style-type** with a value of **none.**
4. Similar to how **margin** orders spaces between divs, **padding** orders spaces within the div.
5. Use the padding to remove the indents in the list
6. Add **padding-bottom** of **8px** to **#LeftNav** for some extra space
7. Add overall **padding** of **15px** to the **#Content** div

Text-Align Syntax

```
#Header {
        text-align: center;
}
```

```
65  ul{
66      list-style-type: none;
67      padding-left: 0px;
68  }
```

Your page should now look like this:

Keith the QB

This is Keith Price's number one Fan Site! He is a good leader, wins games, throws awesome passes, hits his targets, can run when he needs to, and has a cheesy smile. Huskies have been on point this season with a starting record of 4-0. The Huskies smashed Boise State, pummeled Illinois, dominated Idaho State, and ran all over Arizona.

Bow Down to Washington!!

My Favorite Websites

The Verge
Surfline
WORLD STAR
Amazon
The Onion

Connect With Me Online:

Facebook
Linked In
Spotify Playlists

Copyright of Keith the QB

Step 6.1: Style Text and Color

1. Add some color and style to the text
2. Use the CSS style **color** with any value you want
3. Target the **h1** tag in your **#Header** :
   a. #Header h1{ font-size: 60px; font-family: "Passion One"; color: #FFBC00; font-weight: bold; }
4. Use this hex color site to get any color you want if you don't want to use gold:
   a. http://paletton.com/#uid=1000u0klllaFw0g0qFqFg0w0aF
5. Use this site to find fonts:
   a. https://fonts.google.com/
   b. Search for **"Passion One"** and click the **Quick Use** button.
   c. Find the **"Add Code to your Website"**
   d. Apply the font to your code like so:

```
2   <head>
3       <title> Keith the QB  </title>
4       <link rel="stylesheet" type="text/css" href="test.css" />
5       <link href='http://fonts.googleapis.com/css?family=Passion+One' rel='stylesheet' type='text/css'>
6   </head>
7
```

Step 6.2: Style Text and Color

6. Bold the <p> tags and give them a **color: white; bolded text**, and a **font-family: "Arial"**
7. Give all h3 tags a **color: #FFBC00; font-size: 40px; font-family: "Passion One"; margin-bottom: 0px;**
8. Target all divs and give them a light purple **background-color: #8E41D5**
9. Add an image for the page background:
   a. Download dark wood from Subtle Patterns and save the png in the same location as your working folder
10. Under the **body** section of your CSS add **background-image: url('yourimage.png');**

```
2  body{
3      background-image: url('dark_wood.png');
4  }
5
```

11. The link colors are kinda ugly. Target them with the <a> tags and give them:
    a. A white color
    b. Bolded text
    c. Font 24px

Stage 1 Homework

Add 5 more unique CSS styles to your test.css file! Then click the button to submit: Click to certify that all the above links Work

02.

TA Workshop

What is a Web Form?

Web forms allow users to fill in data that is sent to the server to be processed. The data is entered through controls such as text fields, checkboxes, radio buttons and more.

All controls for a single form go within a <form>...</form> tag.

Note! The examples I will go through are styled with Twitter Bootstrap, and will appear slightly different in plain HTML.

Text Field

One line input field where a user can enter text. The default width of the text field is 20 characters. The text field is meant for short text answers.

First name:

Last name:

First name: `<input type="text" name="firstname"><br>`

Last name: `<input type="text" name="lastname">`

Password Field

One line input field meant for a user to enter sensitive information. The characters are masked.

Password:

Password: <input type="password" name="pwd">

Text Area

A multiple line input field where a user can enter text. The text area is meant for long text answers.

Blah blah blah…

Text Area: <textarea name="essay">Blah blah blah...</textarea>

Radio Buttons

A list of choices with select buttons alongside the choices. The radio buttons should be used when only one choice should be selected.

Yes

No

```
<input type="radio" name="answer" value="yes"> Yes<br> <input type="radio" name="answer" value="no"> No
```

Checkboxes

A list of the choices with checkboxes alongside the choices

The checkboxes should be used when zero or more choices can be selected. For example a person can have a bike or a car, they can have neither or they can have both.

I have a bike

I have a car

```
<input type="checkbox" name="vehicle" value="Bike">I have a bike<br> <input
type="checkbox" name="vehicle" value="Car">I have a car
```

Drop-down list

A list of choices in a drop-down menu.

The drop-down list should be used when only one choice should be selected and there are a lot of options to choose from

```
<select name="cars">

        <option value="volvo">Volvo</option>

        <option value="saab">Saab</option>

        <option value="fiat">Fiat</option>

        <option value="audi">Audi</option>

</select>
```

Submit Button

Clicking the submit button sends the information in the form to the server.

`<input type="submit" value="Submit form">`

Form Action

When creating a form the user data needs to be sent to a specific place where the data can be processed. The action attribute details where the data should be sent and method determines how the data should be sent.

Form Action:

```
<form name="input" action="html_form_action.asp" method="post">

    ...

</form>
```

Server Data

When a server receives a submitted form, it reads the data based on the "name" attribute of the form elements. For this reason it is very important to be careful with your naming!

Get vs. Post Requests

- GET method sends the parameters in your form as if the user clicked on a link. Do not use this method if your form has sensitive info on it.
- The POST method packs up your form parameters and submits them in a lump to the server. It is good if you are sending text boxes or non-text data from your form (image uploads for example)

Get vs. Post Requests

- GET method sends the parameters in your form as if the user clicked on a link. Do not use this method if your form has sensitive info on it.
- The POST method packs up your form parameters and submits them in a lump to the server. It is good if you are sending text boxes or non-text data from your form (image uploads for example)

03. Original Work

Stage 3 Homework

1. Create a new HTML file called jobApplicationForm.html
2. Download one of the sample templates
3. Use the template as a reference to create your own job application
4. I will determine your grade based on the number of changes
5. Remember to click the button once your file is uploaded to GitHub

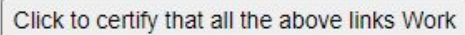Click to certify that all the above links Work

Points Breakdown

- **Level 1: Newbie (40XP) -** Your job application form has new elements in it.

- **Level 2: Apprentice (80XP) -** Your job application form has new elements in it, and you've added every element we talked about in the workshop.

- **Level 3: Master (120XP) -** Your job application form has new elements in it, you've added every element we talked about, you've given your elements good names so the server data makes sense ~~and you've uploaded your form to your Github server folder.~~

Homework Reminders

Before Sunday at Midnight:

- Builder: Upload your files from both stages (1 and 3) to GitHub and click the button [Click to certify that all the above links Work]
- Learner: Make sure you've completed 5 learner modules
- THINKER/SPEAKER: Your first modules are due!
  - Thinker: quiz, face-off, and learn and teach
  - Speaker: quiz, face-off, story
- Writer: Either stage two of first essay or stage one of second essay

Big Class tomorrow is on zoom!

Before you go...

I need to speak with PJ, Tommie, Dmitri, and Jane

If anyone else has questions about grades from last week or general questions, let me know now!