Python Programming

Day 17: Introduction to Web Programming

Introduction to Web Programming

DJANGO	HTML
CSS	BOOTSTRAP FRAMEWORK

Color and symbol meaning

Hint

Preferred

Student's activity

Practice code

	Keyword
	In-built functions
	Strings
	Output

Introduction to Web Development

HTML stands for Hypertext Markup Language, and it is the most widely used language to write Web Pages.

HTML Document Structure

```
<!DOCTYPE html>
<html>
  <head>
    Document header related tags
  </head>

  <body>
    Document body related tags
  </body>
</html>
```

Tags, Attributes, and Elements

Tags, **Attributes** and **Elements** are the stuff that makes up **HTML**.

<tag attribute="value">Margarine</tag>

Tag + *Attribute* + *Content* = *Element*

Tags, Attributes, and Elements

The first line on the top, **<!DOCTYPE html>**, is a document type declaration and it lets the browser know which flavor of HTML you're using (**HTML5**, in this case)

Basic Structure

```
<!DOCTYPE html>
<html>
  <head>
    Document header related tags
  </head>

  <body>
    Document body related tags
  </body>
</html>
```

Tags, Attributes, and Elements

<html> is the opening tag that kicks things off and tells the browser that everything between that and the **</html>** closing tag is an HTML document.

The stuff between <body> and </body> is the **main content** of the document that will appear in the browser window.

Basic Structure

```
<!DOCTYPE html>
<html>
  <head>
    Document header related tags
  </head>

  <body>
    Document body related tags
  </body>
</html>
```

Tags, Attributes, and Elements

Closing tags

The </body> and </html> put a close to their respective elements

Not all tags have closing tags like this (<html></html>) some tags, which do not wrap around content will close themselves. Example is the line-break tag <br>, horizontal rule <hr>, etc.

Basic Structure

```
<!DOCTYPE html>
<html>
  <head>
    Document header related tags
  </head>

  <body>
    Document body related tags
  </body>
</html>
```

Tags, Attributes, and Elements

Attributes

Tags can also have **attributes**, which are extra bits of information.

Attributes appear inside the opening tag and their values sit inside quotation marks. They look something like

Syntax

<tag attribute="value">
Margarine </tag>.

The <head> Element

The **head** element (that which starts with the <head> opening tag and ends with the </head> closing tag) appears before the body element.

It contains information about the page. The information in the head element does not appear in the browser window.

Syntax

<head>

</head>

Page Titles

All HTML pages should have a **page title**.

To add a title to your page, change your code so that it looks like this

Basic Structure

```
<!DOCTYPE html>
<html>
  <head>
     <title>My first web page</title>
  </head>

  <body>
     This is my first web page
  </body>
</html>
```

Paragraphs

Syntax

If you want text to appear on different lines or, rather, if you intend there to be two distinct blocks of text, you need to explicitly state that using the **<p> tag**.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>
  <body>
    <p>How exciting</p>
    <p>This is a paragraph.</p>
  </body>
</html>
```

Emphasis

Syntax

If you want text to appear on noticeable within a block of text you can use the **** and tags.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>
  <body>
    <p>Yes, that really <em>is</em> exciting. <strong>Warning:</strong> level of excitement may cause head to explode.</p>
  </body>
</html>
```

Line breaks

The **line-break tag** can also be used to separate lines like this:

Sample

```
<!DOCTYPE html>
<html>
  <head>
     <title>My first web page</title>
  </head>

  <body>
     This is my first web page<br>
     How exciting
  </body>
</html>
```

Heading Tags

Syntax

Documents start with heading. You can use different sizes for your headings. HTML also has **six levels of headings**, h1 being the almighty emperor of headings and h6 being the lowest **<h1>, <h2>, <h3>, <h4>, <h5>**, and **<h6>**.

While displaying any heading, browser adds one line before and one line after that heading.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>
  <body>

    <h1>My first web page</h1>
      <h2>What this is</h2>
      <p>A simple page put together using
    HTML</p>
      <h3>Why this is</h3>
      <p>To learn HTML</p>
    </body>
</html>
```

Class Activity 1

Create a new directory in your IDE named “webapp”. Now create a HTML file with title, paragraph, a link and 3 levels of header elements

Lists

There are three types of list; **unordered lists, ordered lists and definition lists**. We will look at the first two.

Unordered lists and ordered lists work the same way, except that the former is used for non-sequential lists with list items usually preceded by bullets and the latter is for sequential lists, which are normally represented by incremental numbers.

Include the following in your code

```
<ul>
    <li>To learn HTML</li>
    <li>To show off</li>
    <li>Because I've fallen in love with my computer and want to give her some HTML loving.</li>
  </ul>
```

Lists

Include the following in your code

Simply change the <ul> tags in the previous code to <ol> and you will see that the list will become numbered.

A list within a list.

```
<ul>
  <li>To learn HTML</li>
  <li>
    To show off
    <ol>
      <li>To my boss</li>
      <li>To my friends</li>
      <li>To my cat</li>
      <li>To the little talking duck in my brain</li>
    </ol>
  </li>
  <li>Because I've fallen in love with my computer and want to give her some HTML loving.</li>
</ul>
```

Links

The “H” and “T” in “HTML” stand for “hypertext”, which basically means a system of linked text.

An **anchor tag (a)** is used to define a link, but you also need to add something to the anchor tag — the destination of the link.

The destination of the link is defined in the ***href*** attribute of the tag.

Syntax

** HTML Dog **

Images

Things might seem a little bland and boring with only text content. **Images** spices up the web.

The **img tag** is used to put an image in an HTML document and it looks like this:

The ***src*** attribute tells the browser where to find the image

Syntax

```
<img
src="http://www.pythonsc
hool.com/badge1.gif"
width="120"   height="90"
alt="HTML Bug">
```

Tables

HTML Tables are best used to structure tabular data

The **table element** defines the table.

The **tr element** defines a table row.

The **td element** defines a data cell. These must be enclosed in tr tags, as shown above.

Basic Structure

```
<table>
  <tr>
    <td>Row 1, cell 1</td>
    <td>Row 1, cell 2</td>
    <td>Row 1, cell 3</td>
  </tr>
  <tr>
    <td>Row 2, cell 1</td>
    <td>Row 2, cell 2</td>
    <td>Row 2, cell 3</td>
  </tr>
</table>
```

Forms

Forms are used to collect data inputted by a user. They can be used as an interface for a web application, for example, or to send data across the web.

On their own, forms aren't usually especially helpful. They tend to be used in conjunction with a programming language to process the information inputted by the user, that is where **Python** comes in.

The basic tags used in the actual HTML of forms are form, input, textarea, select and option.

Forms

form defines the form and within this tag, if you are using a form for a user to submit information (which we are assuming at this level), an **action attribute** is needed to tell the form where its contents will be sent to.

The **method attribute** tells the form how the data in it is going to be sent and it can have the value **get**, or **post**, which invisibly sends the form's information to the server.

Syntax

```
<form
action="processingscript.py"
method="post">

</form>
```

Forms - input

The input tag has the most use case in a form. It can take a multitude of guises, the most common of which are outlined below.

- **<input type="text">** or simply <input> is a standard textbox. This can also have a value attribute, which sets the initial text in the textbox.
- **<input type="password">** is similar to the textbox, but the characters typed in by the user will be hidden.
- **<input type="checkbox">** is a checkbox, which can be toggled on and off by the user.
- **<input type="radio">** is similar to a checkbox, but the user can only select one radio button in a group.
- **<input type="submit">** is a button that when selected will submit the form.

Forms - textarea

Textarea is, basically, a large, **multi-line** textbox. The anticipated number of rows and columns can be defined with rows and cols attributes.

Basic Structure

```
<textarea rows="5" cols="20">
A big load of text
</textarea>
```

Forms - select

The select tag works with the option tag to make drop-down select boxes.

When the form is submitted, the value of the selected option will be sent

Basic Structure

```
<select>
   <option>Option 1</option>
   <option>Option 2</option>
   <option value="third option">Option 3</option>
</select>
```

Forms - Names

All of the tags mentioned above will look very nice presented on the page but if you hook up your form to a form-handling script, they will all be ignored.

This is because the form fields need names. So to all of the fields, the attribute name needs to be added.

Syntax:

```
<input type="text" name="pythonschool">
```

Class Activity 2

Create a new HTML file in the “webapp” directory having list element, image, table and a login form elements.

Python Programming

Cascading Style Sheet

Applying CSS

Cascading Style Sheets is used to **style** the HTML to make it look **appealing**

There are three ways to apply CSS to HTML: **Inline, internal, and external**.

Note that the best-practice approach is that the **HTML should be a stand-alone**, presentation free document.

Applying Inline-CSS

Inline styles are plonked straight into the HTML tags using the **style attribute**.

Syntax: ** text **

Applying Internal-CSS

Sample

Embedded or internal styles are used for the whole page. Inside the head element, the style tags surround all of the styles for the page.

```
<!DOCTYPE html>
<html>
<head>
<title>CSS Example</title>
<style>
  p {
    color: red;
  }

  a {
    color: blue;
  }
</style>
...
```

Applying Internal-CSS

Sample

This is preferable to soiling our HTML with inline styling, However, it is usually preferable to keep the HTML and the CSS files separate.

```
<!DOCTYPE html>
<html>
<head>
<title>CSS Example</title>
<style>
   p {
      color: red;
   }

   a {
      color: blue;
   }
</style>
...
```

Applying External-CSS

Sample

External styles are used for the whole, multiple-page website. There is a separate CSS file, which will simply look something like:

```
p {
   color: red;
}

a {
   color: blue;
}
```

If this file is saved as "style.css" in the same directory as your HTML page then it can be linked to in the HTML like this:

```
<!DOCTYPE html>
<html>
<head>
   <title>CSS Example</title>
   <link rel="stylesheet" href="style.css">
...
```

CSS – Text Formatting

You can alter the size and shape of the text on a web page with a range of properties.

Attributes	Description
font-family	This is the font itself, such as Times New Roman, Arial, or Verdana. font-family: arial, helvetica, serif
font-size	font-size sets the size of the font.
font-weight	font-weight states whether the text is bold or not. font-weight: bold|bolder|normal|lighter|100|200|300|400
font-style	font-style states whether the text is italic or not. font-style: italic|normal.
text-decoration	text-decoration states whether the text has got a line running under, over, or through it. text-decoration: underline|overline|line-through|none
text-transform	text-transform will change the case of the text. text-transform: capitalize|uppercase|lowercase|none

CSS – Text Formatting

Text spacing helps to space out the text on a page.

Attributes	Description
letter-spacing	This property is used for spacing between letters. The value can be a length or normal
word-spacing	This property is used for spacing between words. The value can be a length or normal
line-height	This property sets the height of the lines in an element, such as a paragraph, without adjusting the size of the font.
text-align	This property will align the text inside an element to left, right, center, or justify.

Sample

```
p {
   letter-spacing: 0.5em;
   word-spacing: 2em;
   line-height: 1.5;
   text-align: center;
}
```

CSS - Colors

CSS brings 16,777,216 colors to your disposal. They can take the form of a name, an RGB (red/green/blue) value or a hex code.

The displayed CSS color values produce the same result (RED colour):

- **red**
- **rgb(255,0,0)**
- **rgb(100%,0%,0%)**
- **#ff0000**
- **#f00**

CSS - color and background-color

Colors can be applied by using color and background-color (note that this must be the American English "color" and not "colour")

You can apply the color and background-color properties to most HTML elements, including body element.

Sample

```
h1 {
    color: yellow;
    background-color: blue;
}
```

CSS - Borders

Borders can be applied to **most HTML** elements within the body.

To make a border around an element, all you need is **border-style, border-width** and **border-color.**

Sample

```
h2 {
    border-style: dashed;
    border-width: 3px;
    border-left-width: 10px;
    border-right-width: 10px;
    border-color: red;
}
```

CSS - Margins and Padding

margin and padding are the two most commonly used properties for spacing-out elements.

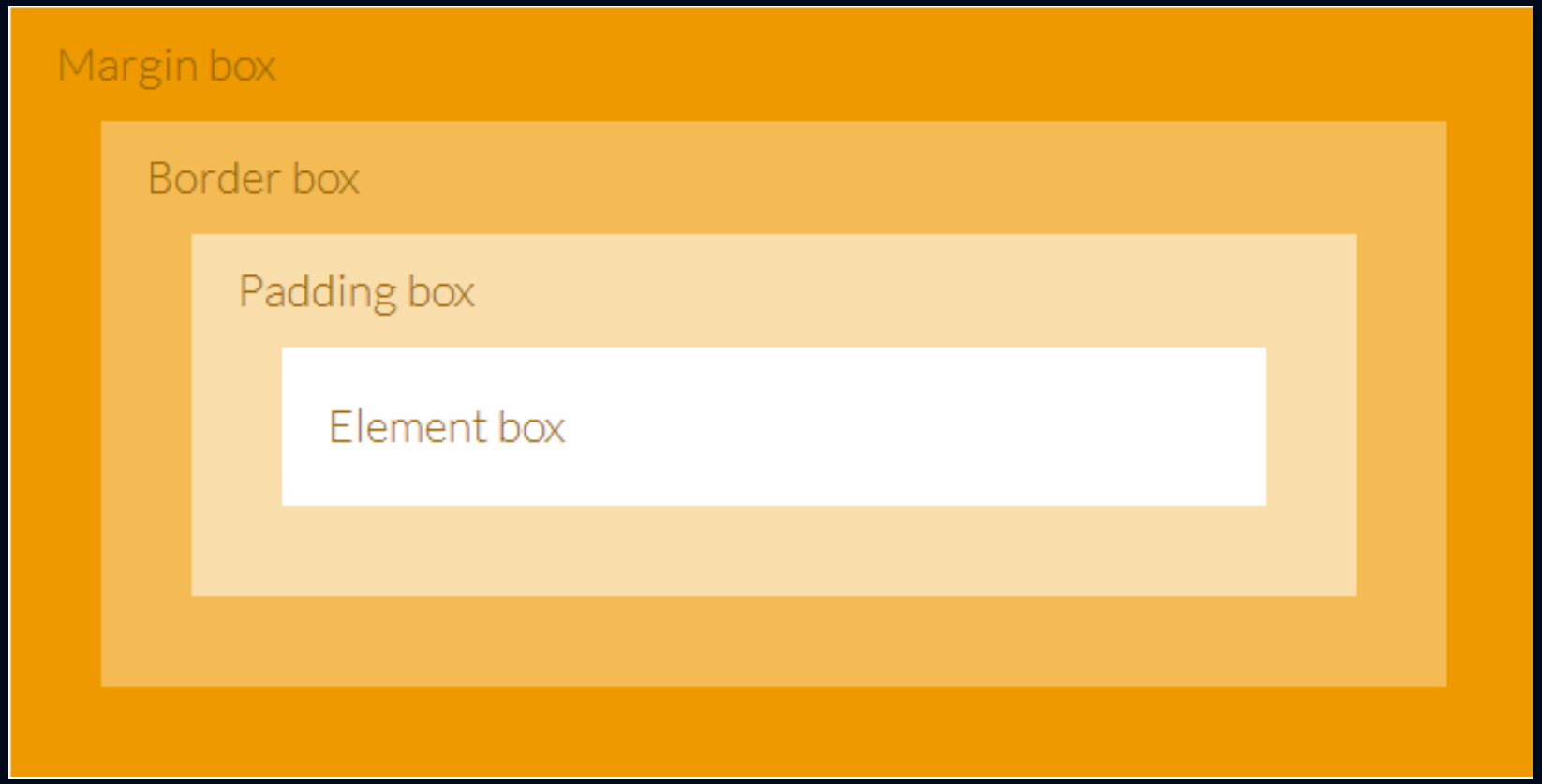
A margin is the space outside something, whereas padding is the space inside something.

```
h2 {
    font-size: 1.5em;
    background-color: #ccc;
    margin: 20px;
    padding: 40px;
}
```

The four sides of an element can also be set individually. margin-top, margin-right, margin-bottom, margin-left, padding-top, padding-right, padding-bottom and padding-left are the self-explanatory properties you can use.

CSS - The Box Model

Margins, padding and borders are all part of what's known as the Box Model.



CSS - Selectors, Properties, and Values

Whereas HTML has tags, CSS has selectors. Selectors are the names given to styles in internal and external style sheets.

There are 3 types of CSS selectors

- **HTML**
- **Class**
- **ID**

CSS - Selectors, Properties, and Values

For each selector (HTML, Class or ID) there are "properties" inside curly brackets, which simply take the form of words such as color, font-weight or background-color.

A value is given to the property following a colon (NOT an "equals" sign). Semi-colons are used to separate the properties.

Selector

Value

```
body {
    font-size: 14px;
    color: navy;
}
```

Property

CSS - Selectors, Properties, and Values

HTML selectors are simply the **names of HTML tags** and are used to change the style of a **specific type** of element.

Examples of HTML selectors

- ✓ **body**
- ✓ **div**
- ✓ **img**
- ✓ **p**
- ✓ **a**
- ✓ **table**
- ✓ **form**

CSS - Selectors, Properties, and Values

You can also define your own selectors in the form of **class** and **ID selectors** aside using the HTML selector.

The benefit of this is that you can have the **same HTML element**, but **present it differently** depending on its class or ID.

CSS - Selectors, Properties, and Values

In the CSS, a class selector is a name preceded by a full stop (“.”) and an ID selector is a name preceded by a hash character (“#”).

The difference between an ID and a class is that an ID can be used to identify one element, whereas a class can be used to identify more than one.

You can also apply a selector to a specific HTML element by simply stating the HTML selector first, so p.jam { /* whatever */ } will only be applied to paragraph elements that have the class “jam”.

CSS - Selectors, Properties, and Values

HTML

```
<div id="top">

<h1>Chocolate curry</h1>

<p class="intro">This is my recipe for making curry purely with chocolate</p>

<p class="intro">Mmmmm mmm</p>

</div>
```

CSS

```
#top {
    background-color: #ccc;
    padding: 20px
}
.intro {
    color: red;
    font-weight: bold;
}
```

Class Activity 3

Create a new directory named “css” inside the “webapp” directory and create a CSS file to style the two webpages created in class activity 1 & 2 to look more appealing.
Note. The css file must end in .css

Next Lecture ...

Python  Programming

Day 18: Introduction to Django Framework