HTML & CSS

- **Overview**
  - HTML (**H**yper **T**ext **M**arkup **L**anguage) is not a programming language. It is a meta-language
  - It is an application of SGML (standard generalized markup language)
  - HTML uses paired tags to markup different elements of a page
  - Language that drives web pages in WWW
  - It describes how the web-browser should present (render) the page.
  - HTML is Not Case Sensitive.
- **Editing Tools**
  - HTML files are text documents and, therefore, can be edited by any text-based editor such as Notepad or TextEdit.
  - Professional HTML editors are recommended to edit HTML documents.
  - Examples:
    - Adobe Dreamweaver, Sublime, Microsoft Expression Web Coffee Cup HTML.
    - Visual Studio Code:
      - Download https://code.visualstudio.com/download
      - Extension to preview your html document.

- **HTML Syntax**
  - The HTML document layout:

```
<!DOCTYPE html>
<html>
<head>

</head>
<body>
<!-- Content of the web page -->

</body>
</html>
```

- **HTML Elements**
  - HTML consists of a series of elements
  - They label the content of the page HTML:
    - Headings, paragraphs, images, etc.
  - Every HTML element must start and end with a tag:

    <tagname>Content goes here...</tagname>
  - HTML Tag References:
    - <html>:
      - Defines the root of an HTML document.
    - <body>:
      - It defines the document's body.
    - <head>:
      - It contains meta information about the HTML page including javascript code.
    - <title>:
      - It specifies a title for the HTML page shown in the browser's title bar or in the page's tab.
    - Headings:
      - It defines HTML headings. There are six headings:

        <h1>Heading 1</h1>
        <h2>Heading 2</h2>

<h3>Heading 3</h3>
<h4>Heading 4</h4>
<h5>Heading 5</h5>
<h6>Heading 6</h6>

- :
  - It defines paragraphs.
  - Any extra spaces and lines within a paragraph are ignored when the page is displayed.
- <hr>:
  - It defines a break in an HTML page, displayed as a horizontal rule.
  - The <hr> element is used to separate content (or define a change) in an HTML page.
- <br\>:
  - It produces a line break in text (carriage-return).
- <a>: HTML Links:
  - It defines a hyperlink.
  - Example:

    UniProt
  - Absolute URLs vs. Relative URLs:
    - Absolute URL: link to a full website

      UniProt
    - Relative URL: link to a file in the current site

      CSS Tutorial

- **HTML Attributes**
  - All HTML elements can have attributes.
  - Attributes provide additional information about elements.
  - Attributes are always specified in the start tag.
  - Attributes usually come in name/value pairs like name="value".
  - HTML Links - The target Attribute
    - It specifies where to open the linked document.
    - By default, it is displayed in the current browser window.
    - The target attribute values:
      _self - Default.

_blank - Opens the document in a new window or tab
_parent - Opens the document in the parent frame
_top - Opens the document in the full body of the window

- The src Attribute
  - The <img> tag is used to embed an image in an HTML page. The src attribute specifies the path to the image to be displayed:

    <img src="uniprot.jpg" >
- The width and height Attributes
  - They are used to specify the width and height in pixels of the <img> tag.
  - Example:

    <img src="uniprot.jpg" width="500" height="600" >
- The alt Attribute:
  - It is a highly recommended attribute for the <img> tag.
  - It specifies a text for an image to be used if the user uses a screen reader or if the image cannot be displayed due to slow connection.
  - Example:

    <img src="uniprot.jpg" alt="UniProt">
- The style Attribute:
  - The style attribute is used to add styles to an element, such as color, font, size, and more.
  - Example:

    This is a red paragraph.
- The lang Attribute
  - It is an attribute of the <html> tag and it is used to identify the language of the page to assist search engines and browsers.
  - Example:

    <html lang="en">
    <html lang="en-US">

- The title Attribute:
  - It shows additional information about the element.
  - It is displayed as a tooltip when you hover over the element:

- **HTML Lists**
  - HTML lists allow web developers to group a set of related items in lists.
  - Unordered (Unnumbered) List:
    - An unordered list starts with the <ul> tag and each list item starts with the <li> tag.
    - The items of the list are marked with bullets (small black circles) by default.
    - Example:

```
<UL type="square" >
    <LI> Smith </LI>
    <LI> Allen </LI>
    <LI> Drake </LI>
</UL>
```

  - Ordered (Numbered) List:
    - An ordered list starts with the <ol> tag and each list item starts with the <li> tag.
    - The list items will be number by default.
    - Example:

```
<OL start="10" >
   <LI> Smith </LI>
   <LI> Allen </LI>
   <LI> Drake </LI>
</OL >
<OL Type="A" >
   <LI> Smith </LI>
   <LI> Allen </LI>
   <LI> Drake </LI>
</OL >
```

- The type of a list can be one of the following:
  type="a": a, b, c
  type="A": A, B, C
  type="i": i, ii, iii
  type="I": I, II, III

- **HTML Special Characters**
  - Reserved characters that belong to HTML language.
  - If used, the browser will interpret them as HTML characters, such as double quotations.
  - Examples ofspecial characters:

Character	**Entity Number**	**Entity Name**	**Description**
Space			Space character
"	&#34;	&quot;	quotation mark
'	&#39;	&apos;	apostrophe
&	&#38;	&amp;	ampersand
<	&#60;	&lt;	less-than
>	&#62;	&gt;	greater-than

- **HTML Formatting Elements**
  - They are used to format to display special types of text.
  - List of formatting elements:
    <b> - Bold text
     - Important text
    <i> - Italic text
     - Emphasized text

<mark> - Marked text
<small> - Smaller text
<del> - Deleted text
<ins> - Inserted text
<sub> - Subscript text
<sup> - Superscript text

- Example:
  - myfirsthtml.html

- **Coding Links: Absolute & Relative URLs**
  - Anchor tags & hrefs
  - Linking to other websites
  - Linking to pages within a website
  - Opening a link in a new browser window/tab

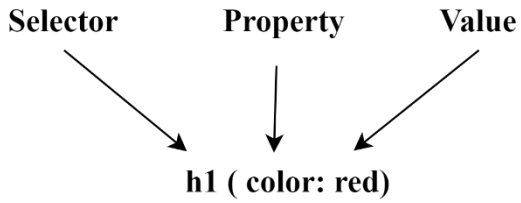
- **HTML Style**
  - CSS attributes are used to add styles to an element, such as color, font, size, and more.
  - Syntax;

    <tagname style="property:value;">

    Where the property is a CSS property and the value is a CSS value.
  - Different style attributes:
    - Use background-color for background color.
    - Use color for text colors.
    - Use font-family for text fonts.
    - Use font-size for text sizes.
    - Use text-align for text alignment.

- **Cascading Style Sheets (CSS)**
  - CSS stands for Cascading Style Sheets.
  - CSS can be shared by multiple web pages.
  - CSS Syntax:

- 3 Elements to a CSS Statement:
  - Selector: What HTML sections does it affect?
  - Property: What attribute of that HTML section will be affected?
  - Value: What change will be made to that attribute?



- There are many kinds of selectors and many ways to reference them: Type, Class, ID, Pseudo, etc.
  - HTML Type Tag – selected with the tag type

```
p {font-size: 10px;
        font-color: White; }
<p>Content</p>
```

  - The Class Attribute – precede the class with a period

```
.myinfo {   font-size: 10px;
            font-color: White; }
            <p class="myinfo">Content</p>
            <div class="myinfo">Other content</div>
```

- Three CSS Definition Locations:
  - Inline: the "style" attribute:

```
<p style="font-color:red;font-size:10px;">Content</p>
```

  - Internal: the <style> markup tag:

```
<html>
<head>
        <style>
                p {background-color: Red;
                   font-family: serif;
                  font-color: White; }
        </style>
</head>
```

```
    <body>
        <p>Content</p>
    </body>
</html>
```

- External: the .css stylesheet file
  <link rel="stylesheet" type="text/css" href=*"mystylesheet.css"* />

- Cascading Inheritance:
  - Nested elements inherit the properties from the its parent
  - If you specify a style for the <body> tag it will affect all content in your HTML page.
  - If you want to override inherited settings, you need to specify a style in a more local element

- **HTML ID**
  - The HTML id attribute is used to specify a unique id for an HTML element.
  - You cannot have more than one element with the same id in an HTML document.
  - It is also used by JavaScript to access and manipulate the element with the specific id.
  - The id name is case-sensitive!
  - Syntax:
    - An is prefixed with a hash character (#), followed by an id name. Then, define the CSS properties within curly braces {}.

```
#id_name {
    CSS properties
}
```

    - Example:

```
.db {
    font-size: 14px;
    color: greenyellow;
```

```
}
```

- **HTML Tables**
  - HTML tables allow web developers to arrange data into rows and columns.
  - Example:

```
<table class="db" >
    <tr bgcolor="black" >
      <th>empno</th>
      <th>ename </th>
      <th>sal </th>
    </tr>
    <tr>
      <td> 7369</td>
      <td> SMITH</td>
      <td> 8000</td>
    </tr>
    <tr>
      <td> 7499</td>
      <td> ALLEN</td>
      <td> 1600</td>
    </tr>
  </table>
```

- **Block vs. Inline Elements**
  - Block Element:
    - Block elements add a line break before and after them
    - Two commonly used block elements are: <p> and .
    - The <p> element defines a paragraph in an HTML document.
    - element:
      - It defines a division or a section in an HTML document.
      - It is often used as a container for other HTML elements.
  - Inline Element:

- Does not add a line break before and after.
- An inline element only takes up as much width as necessary.
- This is a element inside a paragraph.
- Most HTML elements are inline, e.g. <a>

- **HTML JavaScript**
  - Adding JavaScript makes HTML pages dynamic and interactive.
  - The HTML <script> Tag:
    - It is used to define a client-side script (JavaScript)
    - JavaScript uses the following methos to select an HTML element:

      document.getElementById()
    - Example:

```
<h2>Use JavaScript to Change Text</h2>
<p>Use the following database to search for proteins: <span
id="database"></span></p> --
<script>
  document.getElementById("database").innerHTML = "UniProt!";
</script>
```

- **HTML Forms**
  - Form is another HTML tag.
  - It is used to develop web applications that take user input and interact with back-end servers.
  - A form is an area that can contain form elements such as buttons, checkboxes, text fields, radio buttons, drop-down menus, etc.
  - Syntax:

    <form *arguments*> *...form elements...* </form>

- Arguments:
  - Where arguments dictate what to do with the user input
    - action="url" (required)
      - Specifies where to send the data when the Submit button is clicked
    - method="get"(default)
      - Form data is sent as a URL with ?form_data info appended to the end
      - Can be used only if data is all ASCII and not more than 100 characters
    - method="post"
      - It is used to send the body of the URL request
      - Cannot be bookmarked by most browsers
    - target="target"
      - It specifies the location of the page sent as a result of the request
      - target= _blank means open in a new window
      - target= _top means use the same window

- Submit button:
  - It sends the information in the form elements to the server
- <input> Element
  - It indicates the data that the user wants to send to the server.
  - Types of input:

Types	Description
<input type="text">	Displays a single-line text input field
<input type="radio">	Displays a radio button (for selecting one of many choices)
<input type="checkbox">	Displays a checkbox (for selecting zero or more of many choices)

<input type="submit">	Displays a submit button (for submitting the form)
<input type="button">	Displays a clickable button

- HTML form processing;
  - Require web-server to handle action
  - Input element name attributes provide key-value pairs, method attribute indicates GET or POST.
  - Example: html_forms.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title> HTML Forms</title>
    <style>
        .myDiv {
                border: 2px outset rgb(0, 162, 255);
                padding: 5px;
                width: 500px;
                background-color: aqua;
        }
    </style>
</head>
<body>
<form>
    <dd><div   class="myDiv" >
        <div>
            <label for="fname">First name:</label>
            <input type="text"  id="fname"
name="fname" >
        </div><br/>
        <div>
            <label for="lname">Last name:</label>
```

```
            <input type="text" id="lname"
name="lname">
        </div><br/>
        <div>
            <label for="gender">Gender:</label>
            <label for="male">Male</label>:
            <input type="radio" id="male"
name="gender" value="Male">
            <label for="female">Female</label>
            <input type="radio" id="Female"
name="gender" value="Female">
        </div><br/>
        <div>
            <input type="checkbox" id="uniprot"
name="uniprot" value="UniProt">
            <label for="UniProt"> UniProt</label><br>
            <input type="checkbox" id="gene"
name="Gene" value="Gene">
            <label for="Gene"> Gene</label><br>
        </div><br/>
        <div>
            <button type="button">Submit</button>
            <button type="button">Reset</button>

        </div>
    </div></dd><d></d>
</form>
</body>
</html>
```

- **In-page Dynamic Content**
  - Display and process the content of a form using Javascript.

- Example 1: Increment an input value and display the result in the page.

```
<!DOCTPE html>
<html>
<head>
  <style>
    .button {
      background-color: #f79a10;
      color: rgb(2, 2, 2);
      text-align: center;
      font-size: 16px;
      font-weight: bold;
    }
    </style>
  <script>
    function increment(self) {
      var elt = document.getElementById('increment');
      let x = parseInt(self.value)+1;
      elt.innerHTML = "Input value:"+x;
    };
  </script>
</head>
<body>
  <div align="center">
    <h1 >Increment Values</h1><br/>
    <input type="number" class="button"
onchange="increment(this);"/>
    <div id="increment"/>
  </div>
</body>
</html>
```

- Example 2: html_form_button_onclick.html

```
<!DOCTYPE html>
<html>
  <script>
    function clicked() {
      date = new Date();
      var elt = document.getElementById('demo');
      elt.innerHTML = date.getFullYear();
    };
    function changeColor(color) {
      document.getElementById("changeColor").style.ba
ckgroundColor = color;

    };
    </script>
<body>
  <button type="button"  style="background-color:
lightblue;" onclick="changeColor('orange')">
    Change My Color!
    </button><br/><br/>
    <div style="background-color: aqua; width:
500px;" id="changeColor" >
        Hello, <br/>
        Please change my color.

    </div>
    <hr>
    <button type="button" onclick="clicked();">
      Show Me Current Year!
      </button>
      <div id="demo"/>

</body>
</html>
```

- Example 3: fetchomim_console.html
  - Javascript Fetch to read a Json file and display the results on the console (use Ctrl-Shift-J) - J is Capital.

```
<html lang="en">
<head>

<title> Fetch Jason Files</title>

<script>
    const api_url ='omim.json';
    async function getomim(){
        //await can only be sued inside an async
function
        //async makes a function return a Promise
        //await makes a function wait for a Promise
        const response =   await fetch(api_url) ;
        let  data =   await response.json();
       console.log(data);
       //console.log(data.docs[1].MIM);

    }
    getomim();
</script>
</head>

<body>
<div align="center">
<h1>List of MIMs</h1>
</div>
</body>
```

```
</html>
```

- Example 4: fetchmim_html_table.html
  - JavaScript Fetch to read a Json file and display the results in a table.

```
<html lang="en">
<head>

<title> Query Json Files Using JS Fetch</title>
<style>
    .button {
      background-color: #f79a10;
      border: none;
      color: rgb(2, 2, 2);
      padding: 15px 32px;
      text-align: center;
      font-size: 16px;
      font-weight: bold;
      margin: 4px 2px;
    }
    </style>
<script>

    const api_url ='omim.json';
    async function getomim(){
        //await can only be sued inside an async
function
        //async makes a function return a Promise
        //await makes a function wait for a Promise
        var table = "<table border=3 bgcolor=lighblue
width=500 align=center>";
        table +=
"<tr><th>MIM</th><td>preferredTitle</td></tr>";
```

```
        const response =   await fetch(api_url, {
mode: 'no-cors'}) ;
        let  data =   await response.json();
        let size = await data.docs.length;
        for (var i = 0; i < size; i++ ) {
            table +="<tr>";
            table += "<td>" +  data.docs[i].MIM; +
"</td>";
            table += "<td>"
+  data.docs[i].preferredTitle; + "</td>";
        }
        table += "</tr>";
        table += "</table>"
        document.getElementById('container').innerHTM
L = table;
    }
    function clicked() {
        getomim();
    };
    //getomim();
</script>
</head>

<body>

<h1 align="center"> List of MIMs</h1>
<div align="center">
 <button type="button" class="button"
onclick="clicked();">
    Click Here!
    </button>
    <div id="container"/>
</div>
```

```
</body>

</html>
```

- **Homework: Uploaded to course file directory.**