Lecture 0

Temas	Bootstrap CSS DOM Forms HTML Responsive
Date	@February 20, 2024

Una página web en escencia está compuesta por 3 capas principales para poder proporcionar una experiencia y contenido al usuario:

1. Capa de contenido: Información que veo en la página.
2. Capa de presentación: CSS (Cascade styles sheet) Estilos
3. Capa de comportamiento: JavaScript. Permite al usuario interactuar con la página.

Html

¿Qué es HTML?

Hypertext Markup Language. Es el lenguaje que nos permite realizar la capa de contenido, y forma la estructura fundamental de la página.

HTML es un lenguaje de marcado que define la estructura de una página web. Este es implementado por tu navegador web (Safari, Google Chrome, Firefox, etc.) en orden para mostrar contenido en tu pantalla.

`Código: Hello World`

```
<!DOCTYPE html>
<html lang="en"> /* En las busquedas de google filtra resultados
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-s
    <title>Hola</title>
</head>
<body>
```

```
    <h1>¡Hola, mundo!</h1>
</body>
</html>
```

Explicación:

1. `<!DOCTYPE html>`: Esta es una declaración que indica la versión de HTML que el documento está utilizando. En este caso, `html` es la versión más reciente y estándar.
2. `<html lang="en">`: Abre el elemento raíz del documento HTML. El atributo `lang` se establece en "en" para indicar que el idioma principal del contenido es el inglés.
3. `<head>`: Inicia la sección del encabezado del documento. Esta sección contiene información sobre el documento pero no se muestra directamente en la página.
4. `<meta charset="UTF-8">`: Especifica la codificación de caracteres utilizada en el documento. `UTF-8` es una codificación que admite una amplia gama de caracteres.
5. `<meta name="viewport" content="width=device-width, initial-scale=1.0">`: Define la configuración de la vista en dispositivos móviles. Este meta tag es crucial para hacer que el contenido se vea bien en pantallas de diferentes tamaños. `width=device-width` significa que el ancho de la página se ajustará al ancho del dispositivo, y `initial-scale=1.0` establece el nivel de zoom inicial.
    a. `width=device-width`: Hace que el ancho de la página se ajuste al ancho del dispositivo. Esto significa que el contenido se adapta a la pantalla del dispositivo, ya sea un teléfono, una tableta o una computadora de escritorio.
6. `<title>Hola</title>`: Define el título de la página, que se mostrará en la pestaña del navegador.
7. `</head>`: Cierra la sección del encabezado.
8. `<body>`: Inicia la sección del cuerpo del documento. Esta es la parte principal de la página que se mostrará en el navegador.

DOM

Si queremos pensar en cómo se estructuran y se relacionan los elementos de una página web, es conveniente pensar en que el archivo HTML se estructura en forma de árbol (estructura vista en CS50). A este árbol, se le conoce como **Document Object Model.**

El DOM es un api.

> El DOM representa un archivo HTML en una estructura de árbol.

`https://software.hixie.ch/utilities/js/live-dom-viewer/`

Es utilizado para que de forma sencilla podamos leer, acceder y actualizar el contenido de una página web.

Eventualmente esta información tendrá más sentido de recordar, una vez utilicemos Javascript veremos lo poderoso que puede ser utilizar el DOM al modificar o acceder a los elementos de una página web.

Elementos html

Headings

`Explicar headings`

```
<!DOCTYPE html>
<html>
    <head>
        <title>My Web Page!</title>
    </head>
    <body>
        <h1>This is the largest headline</h1>
        <h2>This is also a large headline</h2>
        <h3>This is a slightly smaller headline</h3>
        <h4>This is an even smaller headline</h4>
        <h5>This is the second-smallest headline</h5>
```

```
        <h6>This is the smallest headline</h6>
    </body>
</html>
```

Listas ordenadas y desordenadas

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>My Web Page!</title>
    </head>
    <body>
        An Unordered List:
        <ul>
            <li>One Item</li>
            <li>Another Item</li>
            <li>Yet Another Item</li>
        </ul>
        An Ordered List:
        <ol>
            <li>First Item</li>
            <li>Second Item</li>
            <li>Third Item</li>
        </ol>
    </body>
</html>
```

Imagenes

```
<h2>Imagen</h2>
    <img width="400" src="/url.jpg" alt="Un perrito muy cariñoso
```

Tablas

```
<table>
    <thead>
        <th>Nombre</th>
        <th>Edad</th>
    </thead>
    <tbody>
        <tr>
            <td>Lia</td>
            <td>19</td>
        </tr>
        <tr>
            <td>Omar</td>
            <td>22</td>
        </tr>
    </tbody>
</table>
```

> Hay muchos mas elementos html para utilizar, la documentación está en la pagina https://www.w3schools.com/html/html_elements.asp

Formularios

Otros conjuntos de elementos que son realmente importantes cuando creamos un sitio web es cómo recolectamos la información de los usuarios. Puedes permitir al usuario ingresar información usando un formulario de HTML, el cual puede contener varios diferentes tipos de inputs. Luego en el curso aprenderás a cómo manejar la información una vez el formulario haya sido enviado.

```
<form>
        <input type="text" name="nombre" placeholder="Nombre">
        <input type="password" name="password" placeholder="Con
        <div>
            <input type="radio" name="rojo"> Rojo
```

```
            <input type="radio" name="verde"> verde
            <input type="radio" name="azul" disabled> azul
        </div>
        <div>
            <input name="carrera" list="lista-carreras" placeho
            <datalist id="lista-carreras">
                <option value="sistemas"></option>
                <option value="computacion"></option>
                <option value="civil"></option>
                <option value="quimica"></option>
                <option value="arquitectura"></option>

            </datalist>
        </div>
        <input type="submit" value="Enviar">
    </form>
```

Google ejm

```
<!DOCTYPE html><html lang="es">
<head>
 <meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale
<title>Clon de Google</title>
</head>
<body>
 <form action="https://www.google.com/search" method="GET">
<input type="text" name="q" placeholder="Buscar en Google">
  <input type="submit" value="Buscar">
</form>
</body>
</html>
```

CSS (Hojas de estilo en cascada)

- CSS es usado para personalizar la apariencia de un sitio web.
- Mientras recién estamos comenzando, podemos agregar un atributo de estilo a cualquier elemento HTML para aplicarle algo de CSS.
- Cambiaremos el estilo alterando las propiedades CSS de un elemento, escribiendo algo como **color: blue;** o **text-align: center;**
- En el ejemplo de abajo, haremos un ligero cambio a nuestro primer archivo para darle algo de color a nuestro encabezado.

`Inline Style`

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Hello!</title>
    </head>
    <body>
        <h1 style="color: blue; text-align: center;">A Colorful
        Hello, world!
    </body>
<html>
```

Si damos estilo a un elemento exterior, todos los elementos internos automaticamente tomaran ese estilo. Podemos ver esto si movemos los estilos aplicados en el encabezado a la etiqueta body.

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Hello!</title>
    </head>
    <body style="color: blue; text-align: center;">
        <h1 >A Colorful Heading!</h1>
        Hello, world!
```

```
    </body>
</html>
```

- Mientras podemos dar estilo a nuestras páginas como lo hemos venido haciendo, para obtener un mejor diseño, deberíamos de poder alejar nuestros estilos de las líneas individuales.
    - Una manera de hacer esto es añadiendo los estilos entre la etiqueta <style> en el head. Dentro de estas etiquetas, escribiremos que tipos de elementos queremos que tengan estilos, y el estilo que queremos que se les aplique. Por ejemplo:

Etiqueta Style

```
<html lang="en">
  <!DOCTYPE html>
  <head>
      <title>Hello!</title>
      <style>
          h1 {
              color: blue;
              text-align: center;
          }
      </style>
  </head>
  <body>
      <h1 >A Colorful Heading!</h1>
      Hello, world!
  </body>
</html>
```

Otra manera de hacer es incluyendo un elemento <link> en el head con un enlace a un archivo styles.css el cual contenga ciertos estilos. Esto significa que el archivo HTML lucirá algo así:

```
<html lang="en">
  <!DOCTYPE html>
  <head>
      <title>Hello!</title>
      <link rel="stylesheet" href="styles.css">
  </head>
  <body>
      <h1 >A Colorful Heading!</h1>
      Hello, world!
  </body>
  </html>
```

Hay demasiadas propiedades CSS para revisar aquí, pero al igual que los elementos HTML, normalmente es fácil buscar en Google algo como "cambiar fuente a CSS azul" para obtener el resultado. Algunos de los más comunes son:

- color: el color del texto
- alineación de texto: donde se colocan los elementos en la página
- color de fondo: se puede configurar en cualquier color
- ancho: en píxeles o porcentaje de una página
- altura: en píxeles o porcentaje de una página
- relleno: cuánto espacio se debe dejar dentro de un elemento
- margen: cuánto espacio se debe dejar fuera de un elemento
- familia de fuentes: tipo de fuente para el texto de la página
- tamaño de fuente: en píxeles
- borde: tipo de tamaño (sólido, discontinuo, etc.) color

Ejemplo: Mejorar tabla

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

```
        <title>Nicer Table</title>
    </head>
    <body>
        <table>
            <thead>
                <th>Ocean</th>
                <th>Average Depth</th>
                <th>Maximum Depth</th>
            </thead>
            <tbody>
                <tr>
                    <td>Pacific</td>
                    <td>4280 m</td>
                    <td>10911 m</td>
                </tr>
                <tr>
                    <td>Atlantic</td>
                    <td>3646 m</td>
                    <td>8486 m</td>
                </tr>
            </tbody>
        </table>
    </body>
<html>
```

CSS

```
table {
    border: 1px solid black;
    border-collapse: collapse;
}

td, th {
    border: 1px solid black;
```

```
    padding: 2px;
}
```

Selectores CSS

Esta es una buena introducción a lo que ahora se conoce como los selectores de CSS. Hay muchas maneras de determinar que elementos HTML queremos que tengan estilos, algunas en las cuales mencionaremos aquí:

- **tipo de elemento:** esto es lo que hemos venido haciendo hasta ahora: estilizar todos los elementos del mismo tipo.
- **id:** Otra opción es darle a nuestros elementos HTML un id tal como: <h1 id="first-header">Hello!</h1> y luego aplicarle estilos usando #first-header{...} usando el hashtag para mostrar que estamos buscando un id. Es importante mencionar que dos elementos no deben de tener el mismo id y un elemento no puede tener más de un id.
- **class:** Este es similar al id, pero las clases pueden ser compartidas por más de un elemento, y un solo elemento puede tener muchas clases. Añadimos clases a un elemento HTML de esta manera: <h1 class="page-text muted">Hello!</h1> (nota que hemos añadido dos clases al elemento: page-text y muted). Entonces damos el estilo basado en clases usando un punto en vez del hashtag: .muted {...}.

`Especificación`

Ahora, también tenemos que tratar con el problema de un conflicto de CSS potencial. ¿Qué pasará cuando un encabezado debería de ser rojo basado en una clase pero azul basado en su id? CSS tiene un orden específico que va de la siguiente manera:

- **estilos en línea**
- **id**
- **class**
- **tipo de elemento**

`Selectores CSS`

En adición a la coma para multiples selectores, hay muchas maneras para especificar que elementos te gustaría darle estilo. En la siguiente tabla de la lecture da unos pocos.

a,b	Selector de Elementos Múltiples
a b	Selector Descendiente
a > b	Selector de Hijo Directo
a + b	Selector de Hermano Adyacente
[a=b]	Selector de Atributo
a:b	Selector de Pseudo Clase
a::b	Selector de Pseudo Elemento

`Selector descendiente`

Selecciona todos los elementos descendientes del elemento especificado, sin importar su nivel de anidamiento.

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Using Selectors</title>
        <style>
            ul li {
                color: blue;
            }
        </style>
    </head>
    <body>
        <ol>
            <li>foo</li>
            <li> bar
                <ul>
                    <li>hello</li>
                    <li>goodbye</li>
```

```
                    <li>hello</li>
                </ul>
            </li>
            <li>baz</li>
        </ol>

    </body>
</html>
```

Selector hijo directo

Selecciona todos los elementos hijos directos del elemento especificado.

Selector hermano adyacente

Selecciona el elemento que es adyacente inmediato al elemento especificado

Selector de atributo

Selecciona elementos basados en sus atributos.

Selector de pseudoclase

Selecciona elementos en función de su estado o posición en relación con el documento.

Selector de pseudoelemento

Selecciona partes específicas de un elemento.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
  <style>
    /* Selector descendiente */
    ul li {
      color: blue;
    }
```

```
    /* Selector hijo directo */
    ul > li {
      color: #170752;
    }

    /* Selector hermano adyacente */
    li + li {
      font-weight: bold;
    }

    /* Selector de atributo */
    li[data-level="2"] {
      background-color: rgb(255, 253, 188);
    }

    /* Selector de pseudoclase */
    li:hover {
      background-color: lightblue;
    }

    /* Selector de pseudoelemento */
    /* Agrega un contenido adicional antes del primer párrafo
          elemento <li> */

    li::before {
      content: "• ";
    }
  </style>
</head>
<body>

  <ul>
    <li>Item 1</li>
    <li>Item 2
      <ul>
```

```
        <li data-level="2">Subitem 2.1</li>
        <li data-level="2">Subitem 2.2</li>
      </ul>
    </li>
    <li>Item 3</li>
  </ul>

</body>
</html>
```

Diseño responsivo

- Hoy en día, muchas personas miran los sitios web en otros dispositivos que no son computadoras, tales como un smartphone y tablets. Es importante asegurarse que tu sitio web sea leible para todas las personas en todos los dispositivos.
- Una forma de lograrlo es mediante el conocimiento de la ventana gráfica. La ventana gráfica es la parte de la pantalla que realmente es visible para el usuario en un momento dado. De forma predeterminada, muchas páginas web asumen que la ventana gráfica es la misma en cualquier dispositivo, lo que hace que sea difícil interactuar con muchos sitios (especialmente los más antiguos) en dispositivos móviles.
- Una manera simple para mejorar la apariencia de un sitio en un dispositivo movil es añadiendo la siguiente linea en el head de nuestro archivo HTML. Esta línea le dice al dispositivo mobil use la ventana gráfica que es el mismo ancho que la del dispositivo que se está usando en lugar de una mucho más grande.

```
<meta name="viewport" content="width=device-width, initial-sc
```

- Otra manera en la que podemos lidiar con diferentes dispositivos es con las medias queries. Media queries son una forma de cambiar los estilos de una página basado en como la página está siendo visualizada.

- Un ejemplo de una media query, tratemos de simplemente cambiar el color de la pantalla cuando se reduce a un tamaño determinado. Determinamos una media query escribiendo @media seguido del tipo de query en paréntesis:

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Screen Size</title>
        <style>
            @media (min-width: 600px) {
                body {
                    background-color: red;
                }
            }

            @media (max-width: 599px) {
                body {
                    background-color: blue;
                }
            }
        </style>
    </head>
    <body>
        <h1>Welcome to the page!</h1>
    </body>
</html>
```

• Otra forma de lidiar con diferentes tamaños de pantalla es usar un nuevo atributo CSS conocido como flexbox. Esto nos permite hacer que los elementos pasen fácilmente a la siguiente línea si no encajan horizontalmente. Hacemos esto poniendo todos nuestros elementos en un div al que llamaremos nuestro contenedor. Luego agregamos algo de estilo a ese div especificando que queremos usar una pantalla flexbox para los elementos dentro de él.

También agregamos algunos estilos adicionales a los divs internos para ilustrar mejor la envoltura que ocurre aquí.

Flexbox

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Screen Size</title>
        <style>
            #container {
                display: flex;
                flex-wrap: wrap;
            }

            #container > div {
                background-color: green;
                font-size: 20px;
                margin: 20px;
                padding: 20px;
                width: 200px;
            }
        </style>
    </head>
    <body>
        <div id="container">
            <div>Some text 1!</div>
            <div>Some text 2!</div>
            <div>Some text 3!</div>
            <div>Some text 4!</div>
            <div>Some text 5!</div>
            <div>Some text 6!</div>
            <div>Some text 7!</div>
            <div>Some text 8!</div>
            <div>Some text 9!</div>
            <div>Some text 10!</div>
```

```
            <div>Some text 11!</div>
            <div>Some text 12!</div>
        </div>
    </body>
</html>
```

- Otra forma popular de estilizar una página es usando un grid en HTML. En este grid, podemos especificar atributos tales como columnas, ancho y espaciado entre las columnas y las filas, como se demuestra en el ejemplo debajo. Nota que cuando nosotros especificamos el ancho de las columnas, decimos que el tercero es automatico, significando que debería de llenar el resto de la página.