



Lección 4

Introducción a JavaScript

HTML y CSS incluyen instrucciones para indicar al navegador cómo debe organizar y visualizar un documento y su contenido, pero la interacción de estos lenguajes con el usuario y el sistema se limita solo a un grupo pequeño de respuestas predefinidas.

JavaScript es un lenguaje de programación que se usa para procesar información y manipular documentos, en esta lección aprenderemos a manipular el contenido HTML, cambiar estilos, mostrar alertas, y algunas otras características utilizando JavaScript.

Objetivos de la lección

Al finalizar esta lección, los participantes podrán:

- Describir las 3 distintas formas de incluir JavaScript
- Conocer los diferentes eventos que se pueden utilizar
- Manipular el DOM
- Cambiar estilos CSS a través de JavaScript
- Incorporar librerías JavaScript de terceros

Implementando JavaScript

JavaScript es un lenguaje de programación que se usa para procesar información y manipular documentos. Al igual que cualquier otro lenguaje de programación, JavaScript provee instrucciones que se ejecutan de forma secuencial para indicarle al sistema lo que queremos que haga (realizar una operación aritmética, asignar un nuevo valor a un elemento, etc.). Cuando el navegador encuentra este tipo de código en nuestro documento, ejecuta las instrucciones al momento y cualquier cambio realizado en el documento se muestra en pantalla.

Siguiendo el mismo enfoque que CSS, el código JavaScript se puede incorporar al documento mediante tres técnicas diferentes: el código se puede insertar en un elemento por medio de atributos (En línea), incorporar al documento como contenido del elemento `<script>` o cargar desde un archivo externo.

La técnica En línea aprovecha atributos especiales que describen un evento, como un clic del ratón. Para lograr que un elemento responda a un evento usando esta técnica, todo lo que tenemos que hacer es agregar el atributo correspondiente con el código que queremos que se ejecute.

Por ejemplo, en el siguiente código se muestra una alerta cuando el usuario da clic en el enlace:

```
<a class="btn"onclick="alert('Bienvenido!')"><i class="font-icon icon-download"></i> Hola!</a>
```

Los atributos de evento son útiles cuando queremos probar código o implementar una función de inmediato, pero no son apropiados para aplicaciones importantes. Para trabajar con códigos extensos y personalizar las funciones, tenemos que agrupar el código con el elemento `<script>`. El elemento `<script>` actúa igual que el elemento `<style>` para CSS, organizando el código en un solo lugar y afectando al resto de los elementos en el documento usando referencias.

```
<script>
  function SayWelcome(){
    alert('Bienvenido!')
  };
</script>
```

```
<a class="btn"onclick="SayWelcome()"><i class="font-icon icon-download"></i> Hola!</a>
```

Introducir JavaScript en el documento con el elemento `<script>` puede resultar práctico cuando tenemos un grupo pequeño de instrucciones, pero el código JavaScript crece con rapidez en

aplicaciones profesionales. Si usamos el mismo código en más de un documento, tendremos que mantener diferentes versiones del mismo programa y los navegadores tendrán que descargar el mismo código una y otra vez con cada documento solicitado por el usuario. Una alternativa es introducir el código JavaScript en un archivo externo y luego cargarlo desde los documentos que lo requieren. De esta manera, solo los documentos que necesitan ese grupo de instrucciones deberán incluir el archivo, y el navegador tendrá que descargar el archivo una sola vez.

Para incluir un archivo JavaScript utilizamos la etiqueta `script` y con ayuda del atributo `src` colocando la ruta deseada.

```
| <script src="js/site.js"></script>
```

Eventos JavaScript

los siguientes son los atributos más usados asociados con el ratón.

- **onclick**—Este atributo responde al evento click. El evento se ejecuta cuando el usuario hace clic con el botón izquierdo del ratón. HTML ofrece otros dos atributos similares llamados **ondblclick** (el usuario hace doble clic con el botón izquierdo del ratón) y **oncontextmenu** (el usuario hace clic con el botón derecho del ratón).
- **onmousedown**—Este atributo responde al evento **mousedown**. Este evento se desencadena cuando el usuario pulsa el botón izquierdo o el botón derecho del ratón.
- **onmouseup**—Este atributo responde al evento **mouseup**. El evento se desencadena cuando el usuario libera el botón izquierdo del ratón.
- **onmouseenter**—Este atributo responde al evento **mouseenter**. Este evento se desencadena cuando el ratón se introduce en el área ocupada por el elemento.
- **onmouseleave**—Este atributo responde al evento **mouseleave**. Este evento se desencadena cuando el ratón abandona el área ocupada por el elemento.
- **onmouseover**—Este atributo responde al evento **mouseover**. Este evento se desencadena cuando el ratón se mueve sobre el elemento o cualquiera de sus elementos hijos.
- **onmouseout**—Este atributo responde al evento **mouseout**. El evento se desencadena cuando el ratón abandona el área ocupada por el elemento o cualquiera de sus elementos hijos.
- **onmousemove**—Este atributo responde al evento **mousemove**. Este evento se desencadena cada vez que el ratón se encuentra sobre el elemento y se mueve.

- **onwheel**—Este atributo responde al evento wheel. Este evento se desencadena cada vez que se hace girar la rueda del ratón.

Los siguientes son los atributos disponibles para responder a eventos generados por el teclado. Estos tipos de atributos se aplican a elementos que aceptan una entrada del usuario.

- **onkeypress**—Este atributo responde al evento keypress. Este evento se desencadena cuando se activa el elemento y se pulsa una tecla.
- **onkeydown**—Este atributo responde al evento keydown. Este evento se desencadena cuando se activa el elemento y se pulsa una tecla.
- **onkeyup**—Este atributo responde al evento keyup. Este evento se desencadena cuando se activa el elemento y se libera una tecla.

También contamos con otros dos atributos importantes asociados al documento:

- **onload**—Este atributo responde al evento load. El evento se desencadena cuando un recurso termina de cargarse.
- **onunload**—Este atributo responde al evento unload. Este evento se desencadena cuando un recurso termina de cargarse.

Los eventos no solo los produce el usuario, sino también el navegador. Un evento útil desencadenado por el navegador es load. Este evento se desencadena cuando se ha terminado de cargar un recurso y, por lo tanto, se utiliza frecuentemente para ejecutar código JavaScript después de que el navegador ha cargado el documento y su contenido.

¿Qué es el DOM?

El DOM (Document Object Model, en español Modelo de Objetos del Documento) es una API definida para representar e interactuar con cualquier documento HTML o XML. El DOM es un modelo de documento que se carga en el navegador web y que representa el documento como un árbol de nodos, en donde cada nodo representa una parte del documento (puede tratarse de un elemento, una cadena de texto o un comentario).

El DOM es una de las APIs más usadas en la Web, pues permite ejecutar código en el navegador para acceder e interactuar con cualquier nodo del documento. Estos nodos pueden crearse, moverse o modificarse. Pueden añadirse a estos nodos manejadores de eventos (event listeners en inglés) que se ejecutarán/activarán cuando ocurra el evento indicado en este manejador.

Laboratorio 6

Accediendo a los elementos del DOM.

En este laboratorio incorporarás un archivo JavaScript que te permitirá agregar un menú interactivo, además de cambiar la posición de algunos elementos para fijar la barra lateral.

1. Agrega el archivo `site.js` dentro del folder llamado JS
2. Abre tu archivo `index.html` y agrega el siguiente código dentro de la etiqueta `head`.

```
<script src="js/site.js"></script>
```

3. Guarda los cambios en tu archivo `html`.
4. Agrega el siguiente código dentro de tu archivo `site.js`

```
function toggleMenu() {  
    var button = document.querySelector('.gg-menu-oreos');  
    button.classList.toggle('gg-close');  
    var menu = document.getElementsByClassName('js-menu');  
    menu[0].classList.toggle("active");  
}
```

El código anterior define una función llamada `toggleMenu`, la cual accede a 2 elementos del DOM:

- El botón que tiene como clase `.gg-menu-oreos`: Una vez que lo encuentra, coloca o elimina la clase `.gg-close` del botón con el método `toggle`.
- Los elementos que contengan la clase `js-menu`, una vez que los obtenga, selecciona el primero de la lista, y coloca o elimina la clase `active`.

Tanto `document.querySelector`, como `getElementsByClassName` son métodos que nos permiten acceder y modificar los elementos que tenemos situados en nuestro archivo `html`.

5. Agrega el siguiente código debajo del código agregado en el paso anterior.

```
function fixPhotoOnScroll () {  
    var photo = document.getElementById("photo-profile");  
    this.scrollY > 50 ? photo.style.marginTop = 0 : photo.style.marginTop = '-7rem';  
}
```

La función `fixPhotoOnScroll`, accede al elemento que tenga como identificador "photo-profile" y lo almacena en la variable **photo**. La palabra **this**, representa la ventana del navegador, y la propiedad `ScrollY` es el número de pixeles que el usuario se ha desplazado utilizando el `scroll`.

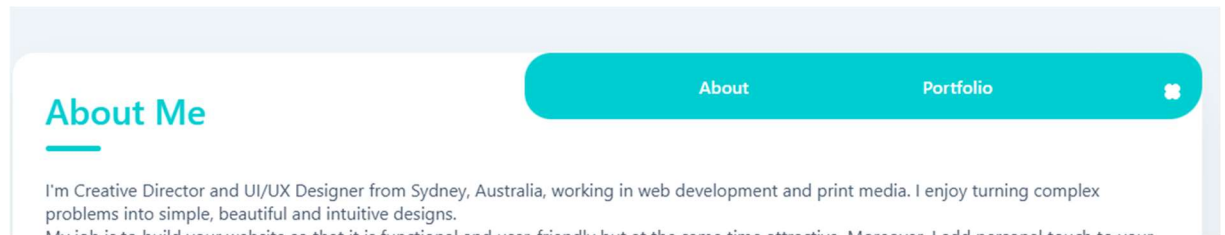
Por lo que sí es mayor a 50 cambia la propiedad margin-top del elemento “photo-profile” a 0, en caso contrario con -7rem;

- Finalmente, escribe el siguiente código

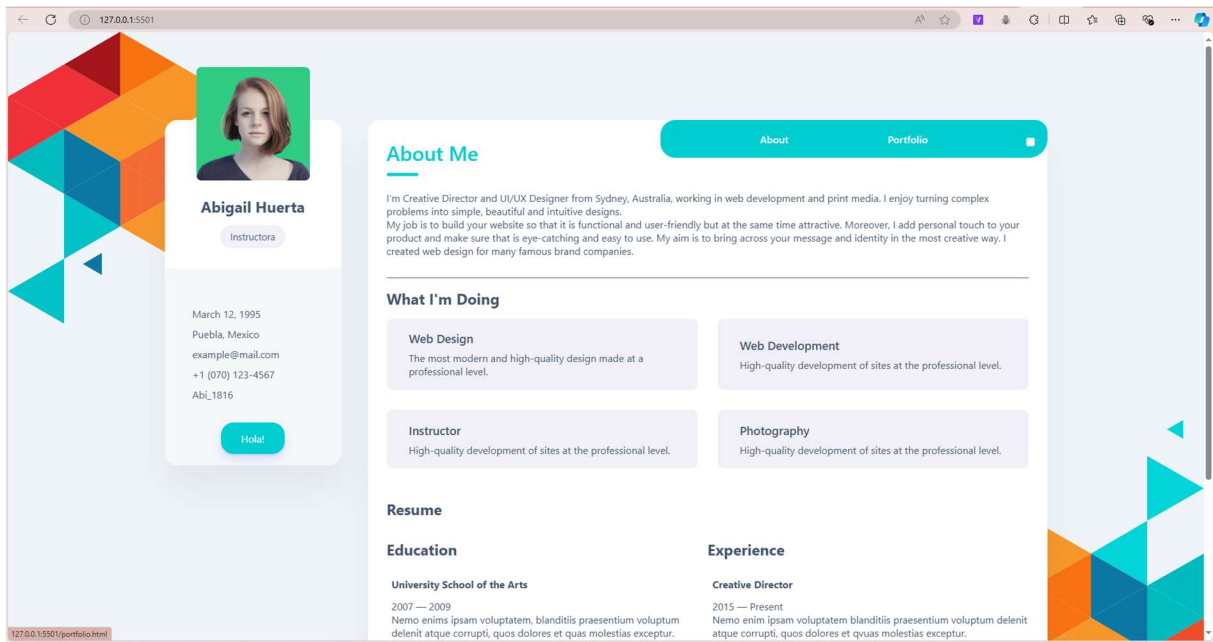
```
window.addEventListener("scroll", fixPhotoOnScroll , false);
```

Este código te permite enlazar la función fixPhotoOnScroll al evento scroll de la ventana.

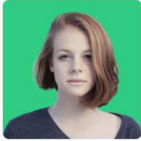
- Guarda los cambios realizados y visualiza el resultado en el navegador.
- Da clic en el menú de la página. Te mostrara las opciones del menú como se muestra a continuación.



- Desplázate hacia la parte inferior de tu página, observa cómo cambia la posición de tu foto.







Abigail Huerta

Instructora

March 12, 1995
Puebla, Mexico
example@mail.com
+1 (070) 123-4567
Abi_1816

Hola!

What I'm Doing

Web Design

The most modern and high-quality design made at a professional level.

Web Development

High-quality development of sites at the professional level.

Instructor

High-quality development of sites at the professional level.

Photography

High-quality development of sites at the professional level.

Resume

Education

University School of the Arts

2007 — 2009

Nemo enim ipsam voluptatem, blanditiis praesentium voluptum delenit atque corrupti, quos dolores et quas molestias exceptur.

New York Academy of Art

2005 — 2007

Ratione voluptatem sequi nesciunt, facere quisquams facere menda ossimus, omnis voluptas assumenda est omnis..

High School of Art and Design

2003 — 2005

Duis aute irure dolor in reprehenderit in voluptate, quila voluptas mag odit aut fugit, sed consequuntur magni dolores eos.

Experience

Creative Director

2015 — Present

Nemo enim ipsam voluptatem blanditiis praesentium voluptum delenit atque corrupti, quos dolores et quuas molestias exceptur.

Art Director

2013 — 2015

Nemo enim ipsam voluptatem, blanditiis praesentium voluptum delenit atque corrupti, quos dolores et quas molestias exceptur.

Web Designer

2010 — 2013

Nemo enim ipsam voluptatem, blanditiis praesentium voluptum delenit atque corrupti, quos dolores et quas molestias exceptur.

Autor: Abigail Huerta De Los Santos

Laboratorio 7

Creando la página portfolio.html.

En este laboratorio utilizaras la plantilla creada en laboratorios anteriores para crear una segunda página, en la cual podrás crear una galería de imágenes. Sigue los siguientes pasos.

1. Crea una copia del archivo template.html con el nombre de portfolio.html.
2. Sustituye el código de la sección de información por el siguiente. Modifica el valor del atributo src de cada imagen, puedes utilizar las imágenes que desees.

```
<header>
  <div class="menu-circle">
    <i class="gg-menu-oreos"></i>
  </div>
  <div class="inner-menu js-menu">
    <ul class="nav">
      <li class="nav_item"><a class="active" href="about.html">About</a></li>
      <li class="nav_item"><a href="portfolio.html">Portfolio</a></li>
    </ul>
  </div>
</header>
<div>
  <h2 class="title bordered">Portafolio</h2>
  <div class="portfolio">
    
    
    
    
    
    
    
  </div>
  <div class="modal" id="preview"></div>
</div>
```

3. Agrega el atributo onclick a cada una de las imágenes.

```

```

4. Agrega los siguientes estilos en tu archivo site.css

```
.portfolio{
  column-count: 3;
  column-gap: 10px;
  row-gap: 12px;
}
.portfolio img{
  max-width: 100%;
}
```

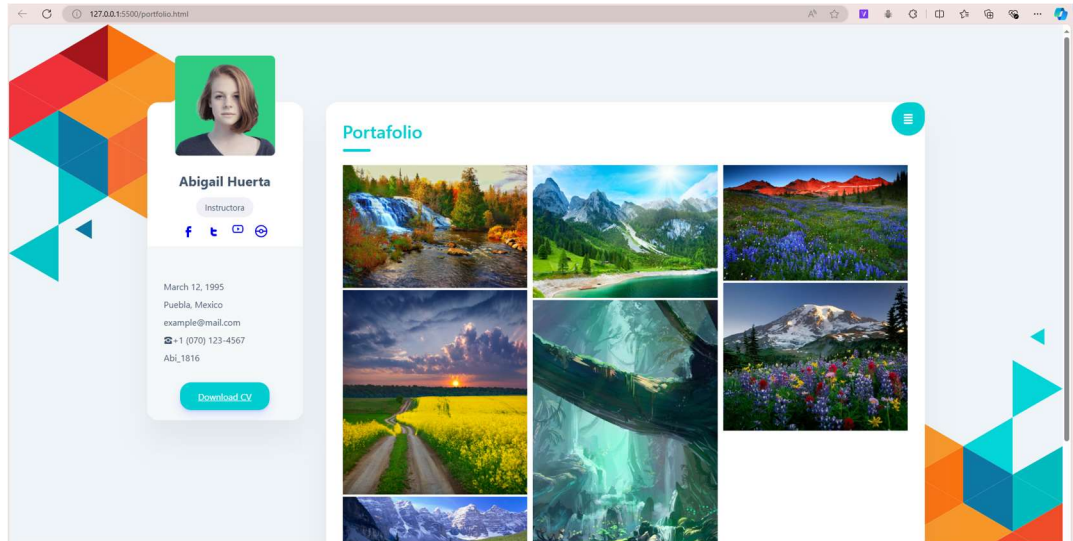


```
.modal{  
  position: fixed;  
  top:0;  
  left: 0;  
  width: 100%;  
  height: 100%;  
  padding: 40px 20px;  
  background-color: rgba(0, 0, 0, 0.7);  
  z-index: 999;  
  display: none;  
  text-align: center;  
}  
  
.modal img{  
  max-height: 100%;  
  margin: auto;  
}  
  
.showModal{  
  display: block;  
  transition: 0.5s ease-in;  
}
```

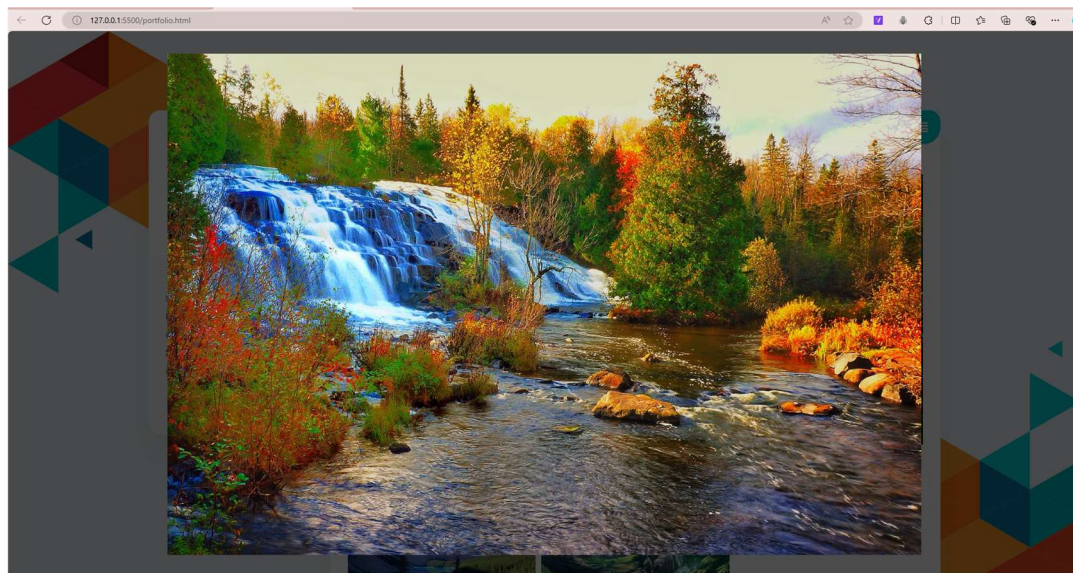
5. Abre el archivo site.js y agrega el siguiente código.

```
function showImage(image)  
{  
  var newImage=document.createElement('img');  
  newImage.src=image.src;newImage.removeAttribute('onclick');  
  var modal=document.getElementById('preview');  
  modal.classList.toggle('showModal');modal.appendChild(newImage);  
}
```

6. Guarda los cambios realizados en todos los archivos.
7. Previsualiza los cambios de este nuevo archivo, veras algo similar a lo siguiente.



Al dar clic en una imagen



Actividades.

1. Agrega la funcionalidad correspondiente que te permita cerrar la “ventana modal” y ver la galería de imágenes
2. Crea una 3er pagina, el contenido es libre y agrega el enlace al menú principal.