
ICARO
Nuestro compromiso es con el futuro.

Front End I

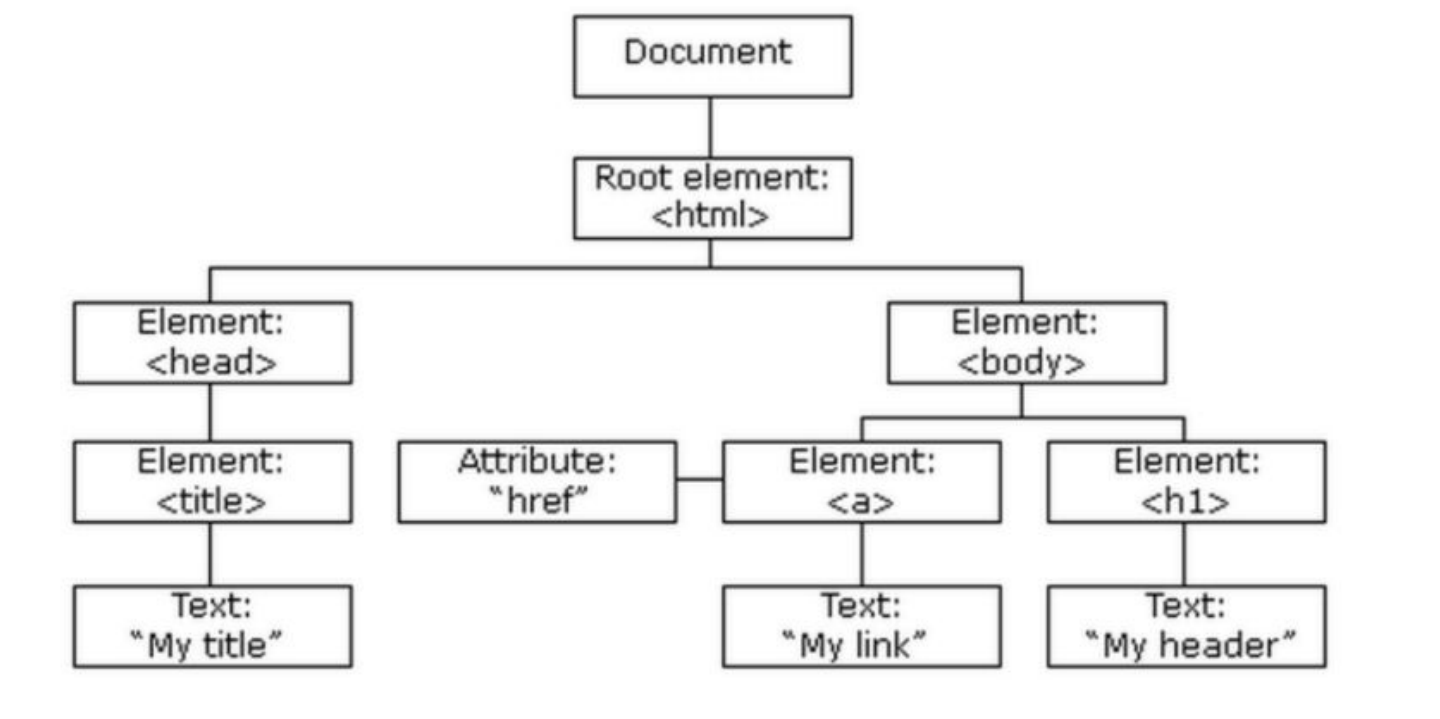
Clase 1

¿Qué es el DOM?

El **DOM** -Document Object Model- es una interfaz de programación para los documentos de **HTML** y **XML**. Facilita una representación estructurada del documento y define de qué manera los programas pueden acceder a él para modificarlo, tanto su estructura como en estilo y contenido.
El **DOM** da una representación del documento como un grupo de **nodos** y **objetos estructurados** que tienen propiedades y métodos. Esencialmente, conecta las páginas web a **scripts** o lenguajes de programación.
Es decir que:
Una página web es un documento **=>** Este documento puede exhibirse en la ventana de un navegador o también como código fuente HTML.

El DOM es una representación completamente orientada al objeto de la página web y puede ser modificado con un lenguaje de script como JavaScript.

Árbol DOM


Document
Root element:
<html>
Element:
<head>
Element:
<body>
Element:
<title>
Attribute:
"href"
Element:
<a>
Element:
<h1>
Text:
"My title"
Text:
"My link"
Text:
"My header"

El objeto Window

El objeto **window** representa una ventana que contiene un documento **DOM**; la propiedad document apunta al documento DOM cargado en esa ventana.
En los navegadores el **nodo raiz** o **document** define el principio y final de la página web. Sin embargo éste no es el objeto más alto de la jerarquía, ya que la página está incluída en el navegador.
El objeto document depende del objeto **window** el cual *es el objeto más alto en la jerarquía del navegador* y *del que dependen todos los demás.*
El nodo raíz del **DOM** (objeto Document) es una propiedad, y por tanto **hijo** del objeto window .
Nótese que hablamos aquí de objetos y no de nodos. Esto es así porque una vez que subimos un nivel por encima de document ya no estamos hablando del **DOM**, sino de un objeto window que engloba al **DOM** como una propiedad del mismo.

Dentro del **DOM** cada componente o nodo del esquema, se comporta como un objeto ya que puede tener propiedades y métodos.

JavaScript en el Front End

Casi todos los navegadores web pueden ejecutar **JavaScript**, lo que lo convierte en uno de los lenguajes de lenguajes de programación más populares del mundo. Tiene una barrera de entrada baja: todo lo que necesita para programar en **JavaScript** es un editor de texto y un navegador web.
JavaScript es un lenguaje de scripting de alto nivel que se interpreta y compila en en tiempo de ejecución. Esto significa que requiere un **motor** que es responsable de interpretar un programa y ejecutarlo.
Los motores de **JavaScript** más comunes se encuentran en navegadores como **Firefox, Chrome o Safari**, aunque **JavaScript** puede ejecutarse sin un navegador utilizando un motor como **Google V8**.
La fiebre por extender **JavaScript**, fue mucho más allá de lo imaginado y encontró en **Node.js** la pieza que faltaba para encajar en ciertos entornos como las aplicaciones de escritorio, o la gestión de redes.
Todo ello, permite que **JavaScript** deje de ser un lenguaje exclusivo de la web para ir mucho más allá, adentrándose incluso en el desarrollo del Internet of Things (IoT) y la Robótica.

HTML & JavaScript

Script en HTML

Con esta línea podremos incluir una pieza de **Javascript** en el HTML:

```
<h1>Hello World</h1>
<script>alert("hello!");</script>
```

Este script se ejecutará tan pronto como el browser lea el html.

Pero cuando tenemos muchas instrucciones para un programa no es recomendable que éste esté incluído directamente en el documento de HTML sino en un documento aparte que lleve toda la lógica del **JavaScript**

```
<h1>Hello World!</h1>
<script src="logica.js"></script>
```

Cuando un documento HTML hace referencia a una URL externa como este caso, el browser lo incluye en la página.

Recordar siempre cerrar el tag </script>, de lo contrario el resto de la página será interpretado como parte de este **script**. (y no queremos eso)

Selectores del DOM

¿Cómo manipulamos el DOM?

El primer paso para poder manipular el **DOM,** es adquirir cierta destreza en el manejo de los **selectores**, ya que siempre los **selectores** serán el primer paso, para realizar operaciones de **lectura** o **modificación** del **DOM**.

- **.getElementById()**

```
// <div id="miDiv"></div>
document.getElementById("miDiv");
```

por su id.

.getElementsByName()

Permite la selección de varios elementos por su atributo name.

```
1    // <form name="miForm"></form>
2    document.getElementsByName("miForm");
```

Selectores del DOM

• **.getElementsByTagName()**
Permite la selección de varios elementos por su etiqueta.

```
// <input>
document.getElementsByTagName("input");
```

• **.getElementsByClassName()**
Permite la selección de varios elementos por su clase.

```
// <div class="rojo"></div>
document.getElementsByClassName("rojo");
```

Existen muchas posibilidades para realizar selecciones dentro de un documento html que van mucho más alla de la clase, id, etiqueta o propiedades, ya que el soporte para ello es muy bueno en todos los navegadores.
JavaScript no se queda atrás y se pueden usar **querySelector** y **querySelectorAll** que además gozan de un gran soporte.

Selectores del DOM

.querySelector()
Devuelve el primer elemento que coincida con el selector.

```html
<div id="miDiv">
    <span id="miId5" class="miClase" title="cinco"></span>
    <span id="miId4" class="miClase" title="cuatro"></span>
    <span id="miId3" class="miClase" title="tres"></span>
    <span id="miId2" class="miClase" title="dos"></span>
    <span id="miId1" class="miClase" title="uno"></span>
</div>
```

```js
document.getElementById('miId1').title // uno
document.querySelector('#miDiv .miClase').title // cinco
document.querySelector('#miDiv #miId1.miClase').title // uno
document.querySelector('#miDiv .inventado').title // ERROR -> undefined
document.querySelector('#miDiv .miClase[title^=u]').title // uno
```

Selectores del DOM

.querySelectorAll()
Devuelve todos los elementos que coincidan con el selector en un pseudo-array.

```
1  document.querySelectorAll('p') // los párrafos
2  document.querySelectorAll('div, img') // divs e imágenes
3  document.querySelectorAll('a > img') // imágenes contenidas en enlaces
```

¿getElementBy… o querySelector?
Con una instrucción **querySelector**, puede seleccionar un elemento basado en un **selector de CSS**. Esto significa que puede seleccionar elementos por ID, clase o cualquier otro tipo de selector. Con el método **getElementById**, solo puede seleccionar *un elemento* por su ID.
Por lo general, debe optar por el selector que haga el trabajo con mayor claridad.
Si solo necesita seleccionar un elemento por ID o clase, puede usar getElementById o getElementsByClassName, respectivamente.
Si necesita utilizar una regla más elaborada para seleccionar elementos, el método querySelector es su mejor opción.

Manipulando el DOM

Manipular contenido HTML

```
<script>
document.getElementById("p1").innerHTML="New text!";
</script>
```

Cambiar el valor de un atributo

```
<script>
document.getElementById("image").src="landscape.jpg";
</script>
```

Modificando estilos y clases

Cambiar estilo CSS en línea

```
<script>
document.getElementById("p2").style.color="blue";
</script>
```

Cambiar clase CSS

```
document.getElementById('objeto').className="NombredeEstilo";
```

¡Vamos al código!

Bibliografía consultada y links para seguir investigando

- Jones, D. (2017) ***JavaScript - Novice to ninja*** - United States of America - SitePoint Pty.Ltd
- Gascón Gonzalez U. (2017) ***JavaScript, ¡Inspírate!-*** Leanpub
- Pérez Eguíluz J. (2009) ***Introducción a JavaScript -***Libroweb.es
- Puig Collel J. ***CSS3 y JavaScript avanzado*** - Universidad Oberta de Catalunya
- Rodriguez Jose A. ***Manual de JavaScript*** - Publicado en la página web www.internetmania.net
- ***https://uniwebsidad.com/libros/javascript?from=librosweb***
- ***https://developer.mozilla.org/es/docs/Learn/JavaScript***

Muchas gracias!

ICARO Asociación Civil
CUIT 30716564815
info@icaro.org.ar
www.icaro.org.ar