



Florida
Universitària

DOM

Desarrollo Web en Entorno Cliente

Curso 2025/2026

Paco Segura

DOM (Document Object Model)

- **Document Object Model**: estándar para la representación y acceso a la estructura de un documento HTML analizado desde JavaScript.
 - Estructura en forma de **árbol**.
 - Accesible a través del objeto **document**, disponible desde cualquier navegador.
-

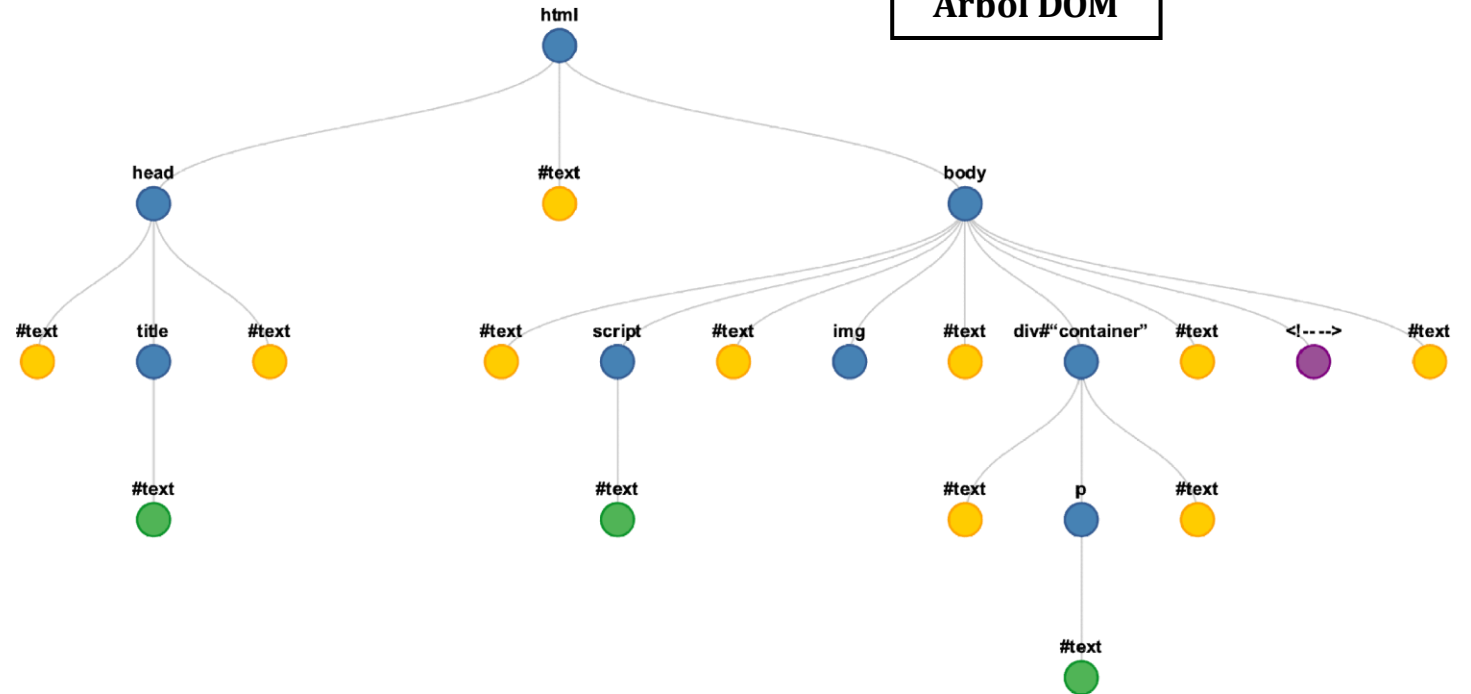
DOM (Document Object Model)

HTML

```
<!doctype html>
<html>
<head>
  <title>My First Web Page</title>
</head>
<body>

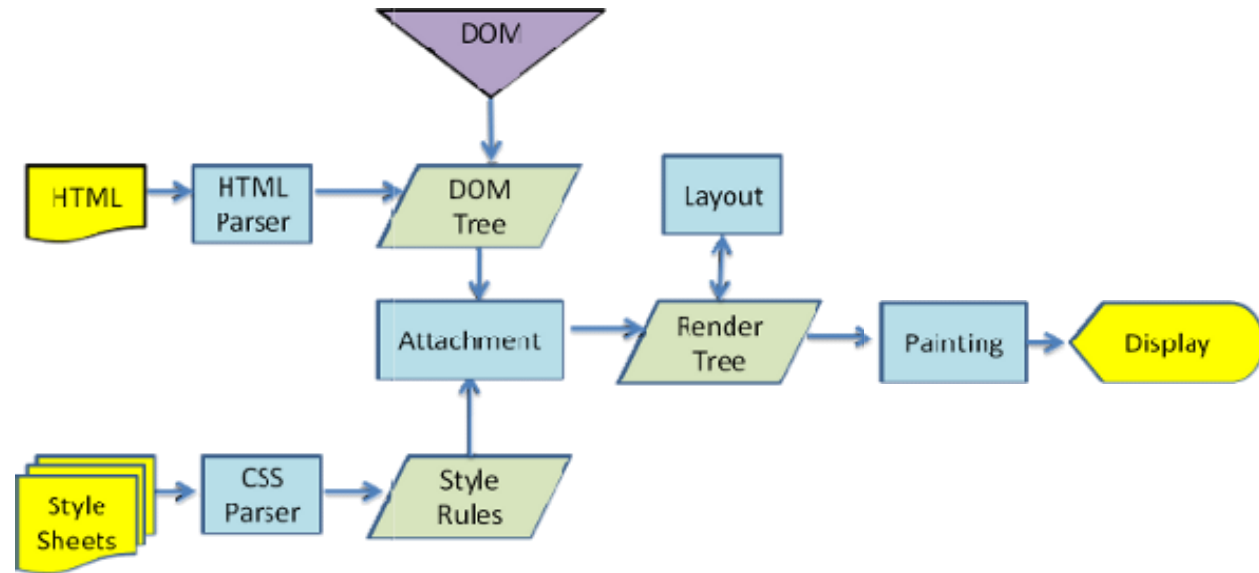
  <script>console.log("Hi");</script>
  <img src='cat.png' />
  <div id="container">
    <p>Un nuevo párrafo</p>
  </div>
  <!-- Un comentario-->
</body>
</html>
```

Árbol DOM



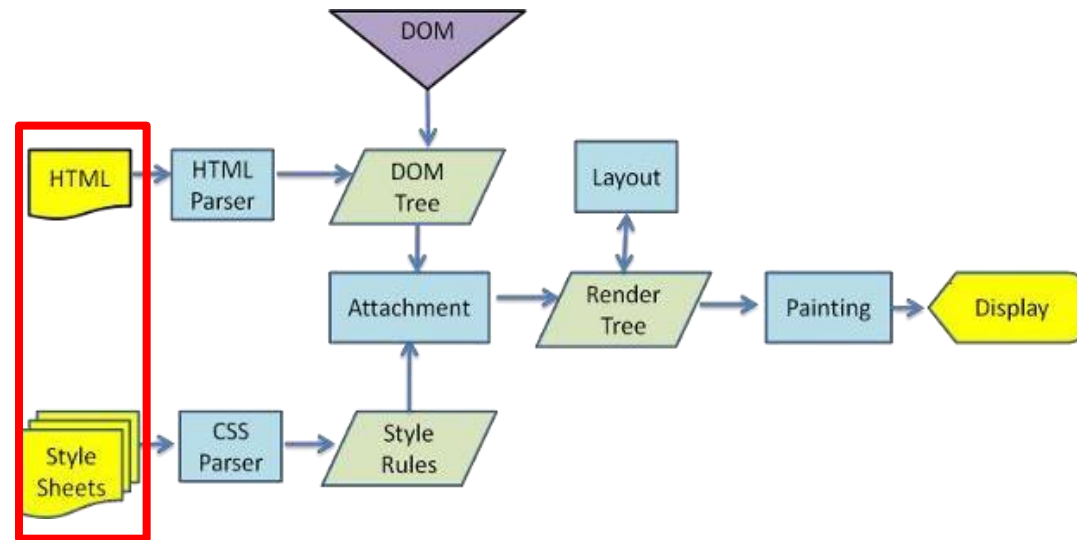
¿Cómo se visualiza un HTML?

- Ejemplo WebKit:



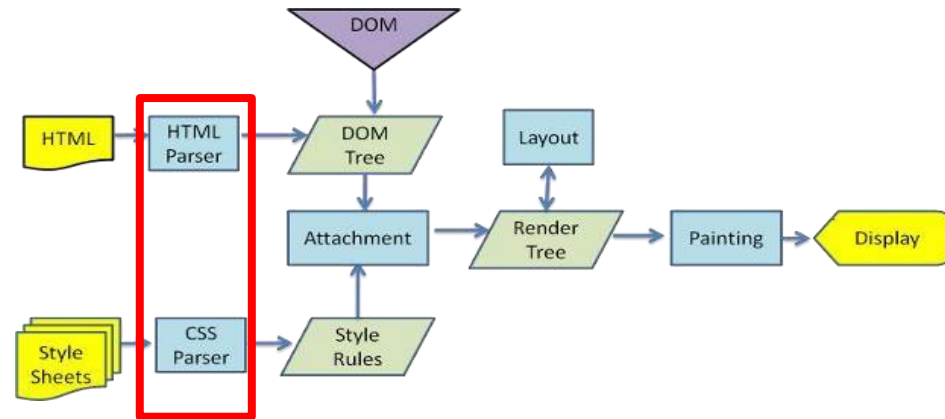
¿Cómo se visualiza un HTML?

- Ejemplo WebKit:
 - Llegan los documentos HTML y CSS desde el servidor mediante la respuesta HTTP.



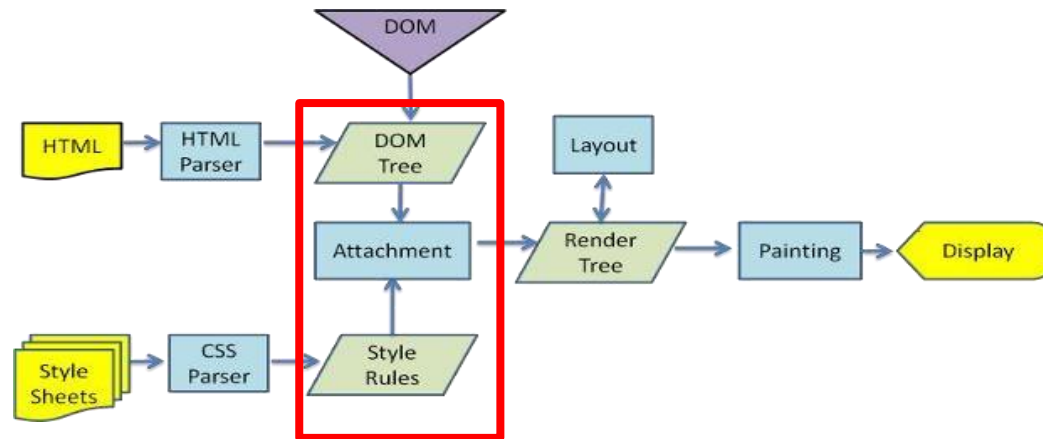
¿Cómo se visualiza un HTML?

- Ejemplo WebKit:
 - El parser de HTML comienza a procesar el HTML de arriba abajo para construir el DOM.
 - El parser de CSS comienza a procesar las reglas CSS.



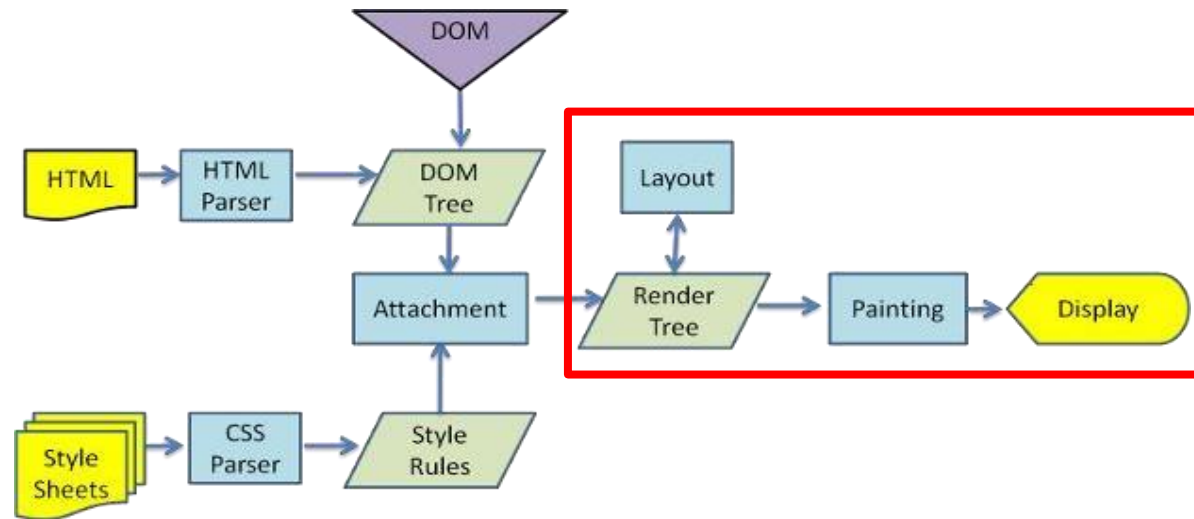
¿Cómo se visualiza un HTML?

- Ejemplo WebKit:
 - El DOM es procesado y los estilos CSS procesados son incrustados en los nodos del DOM.
 - El resultado es un árbol de visualización.



¿Cómo se visualiza un HTML?

- Ejemplo WebKit:
 - El árbol de visualización es procesado y pintado sobre la interfaz gráfica.



¿Y qué ocurre con JavaScript?

- Con los scripts de JavaScript podemos modificar el DOM...
 - Eso implica que las etiquetas `<script>` son procesadas con el HTML y ejecutadas (si es necesario) en ese momento.
 - Tras la ejecución pasa a la siguiente etiqueta a procesar.
-

¿Y qué ocurre con JavaScript?

- Las etiquetas se procesan en orden, de una en una.
- El DOM disponible depende de cuánto se haya procesado hasta ese momento.
- Hasta que la etiqueta `<script>` no ha terminado, no se procesan el resto de etiquetas.

- **Ver ejemplo 1.**

Objetos del navegador

- Hemos visto como se representa el árbol DOM. Con los objetos del navegador podemos interactuar con el DOM.



Objetos del navegador

- Al abrir una página Web, JavaScript crea a partir de ella una estructura de objetos:
 - **Navigator:** información sobre el navegador. Lo crea JavaScript, no es un objeto HTML DOM.
 - **Screen:** información sobre la pantalla del PC. Al igual que Navigator, lo crea JavaScript. No es un objeto HTML DOM.
 - **Window:** objeto más alto en la jerarquía HTML DOM. Representa una ventana del navegador. JavaScript crea uno por cada etiqueta 'body'.

```
console.log(navigator);  
console.log(screen);  
console.log(window);
```

```
▶ Navigator {vendorSub: '', productSub: '20030107', vendor: 'Google Inc.', maxTouchPoints: 0,  
▶ Screen {availWidth: 2560, availHeight: 1040, width: 2560, height: 1080, colorDepth: 24, ...}  
▶ Window {window: Window, self: Window, document: document, name: '', Location: Location, ...}
```

Objeto window

- **Document:** representa el documento HTML y permite acceder a todos los elementos de la página.
- **History:** consiste en un array de URLs con el historial de esa ventana del navegador.
- **Location:** información sobre la URL actual.

```
console.log(window.document);  
console.log(window.history);  
console.log(window.location);
```

▶ #document

▶ History {length: 5, scrollRestoration: 'auto', state: null}

▶ Location {ancestorOrigins: DOMStringList, href: 'file:///C:/U.

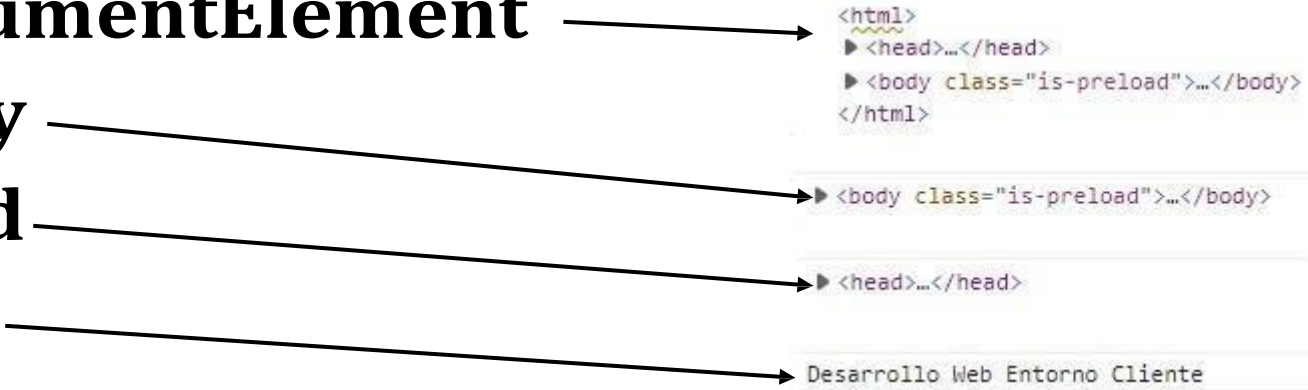


Document

- Desde `document` (`'window.document'` o directamente `'document'`) accedemos al árbol DOM. Podemos consultarlo, modificar nodos existentes y modificar la estructura del DOM.
 - En este apartado veremos cómo consultar el árbol DOM desde **`document`**.
-

Consultar el DOM

- Desde document ('window.document' o directamente 'document') accedemos al árbol DOM.
- Para navegar directamente a algunas partes del DOM:
 - **document.documentElement**
 - **document.body**
 - **document.head**
 - **document.title**



```
<html>  
  <head>...</head>  
  <body class="is-preload">...</body>  
</html>
```

```
<body class="is-preload">...</body>
```

```
<head>...</head>
```

```
Desarrollo Web Entorno Cliente
```

Consultar el DOM

- Para navegar desde cualquier nodo del DOM:
 - **parentNode:** el padre del nodo actual.
 - **previousSibling:** hermano del nodo, por la izquierda.
 - **nextSibling:** hermano del nodo, por la derecha.
 - **firstChild:** primer hijo.
 - **lastChild:** último hijo.
 - **childNodes:** Todos los hijos de un nodo. No es un array.
 - **Ver ejemplo 2.**
-

Consultar el DOM

- Para navegar directamente a nodos DOM:
 - **document.getElementById**
 - **document.getElementsByName**
 - **document.getElementsByTagName**
 - **document.getElementsByClassName**
-

document.getElementById

- El más utilizado. Selecciona el elemento HTML cuyo atributo 'id' coincide con el parámetro indicado:

- [Ver ejemplo 3.](#)
-

document.getElementsByName

- Busca los elementos cuyo atributo 'name' es igual al parámetro indicado:

- Ver ejemplo 4.
-

document.getElementsByTagName

- Obtiene todos los elementos de la página HTML cuya etiqueta es igual al parámetro que se le pasa.
- Ejemplo: obtener todos los párrafos de una página HTML.

• [Ver ejemplo 5.](#)

document.getElementsByClassName

- Devuelve un **array** con todos los nodos que coinciden con el parámetro indicado. Se debe procesar el array:

- Ver ejemplo 6.
-

Modificación de elementos

- Hemos visto como seleccionar elementos del DOM.
 - Ahora vamos a ver cómo modificar su apariencia y comportamiento.
-

Modificación de elementos

- Hemos visto que el árbol DOM se estructura a partir de nodos. Existen 12 tipos de nodos:
 - **Document**: nodo raíz del que derivan todos los demás.
 - **Element**: cada una de las etiquetas HTML. Único nodo que puede contener atributos. Único del que pueden derivar otros nodos.
 - **Attr**: se define para representar cada uno de los atributos de las etiquetas HTML.
 - **Text**: contiene el texto encerrado por una etiqueta HTML.
 - **Comment**: representa los comentarios incluidos en la página HTML.
 - **Resto**: `DocumentType`, `CDataSection`, `DocumentFragment`, `Entity`, `EntityReference`, `ProcessingInstruction` y `Notation`.
-

Element

- Atributos de Element:
 - className, clientHeight, clientWidth, id, innerHTML, tagName, textContent.
 - Atributo children: Todos los elementos hijos de un nodo.
 - firstElementChild, lastElementChild.
 - nextElementSibling, previousElementSibling.
 - parentElement.
 - innerHTML: seleccionar y reemplazar -ambas- el contenido HTML de un nodo.
 - Con los atributos de Element, podemos modificar los nodos del árbol DOM -> **Ver Ejemplo 7**
-

Element

- Algunos Elements especiales:
 - Tablas:
 - tableElement.rows -> Contiene los elements <tr>
 - tableElement.caption -> El caption de la tabla
 - tableElement.tBodies -> Colección de elements <tbody>
 - tableRow.cells -> La colección de <td> y <th>
 - tableRow.sectionRowIndex -> Posición de la fila dentro de la sección (body, head, etc.)
 - cell.cellIndex -> La posición de la celda en la fila.
 - **Ver Ejemplo 8**
-

Selectores CSS

- Con JavaScript también podemos utilizar la sintaxis propia de CSS para seleccionar elementos del árbol DOM, utilizando **querySelector** y **querySelectorAll**:
 - **querySelector**: devuelve el primer elemento del documento que coincida con el selector o selectores indicados.
 - **querySelectorAll**: devuelve todos los elementos del documento que coincidan con el selector o selectores indicados
-

Selectores CSS

- **.class:** Elementos de una clase llamada .class.
 - **#id:** Elemento con el id llamado id.
 - **Tag:** Elementos con un determinado tag.
 - **elem1, elem2:** Elementos que cumplen con elem1 o elem2.
 - **elem1 elem2:** Elementos de tipo elem2 dentro de elem1.
 - **elem1 > elem2:** Elementos de tipo elem2 que tienen un padre de tipo elem1.
 - **Elem1+elem2:** Todos los elementos de tipo elem2 inmediatamente después de un elem1.
 - **Elem1~elem2:** Elementos de tipo elem2 precedidos de elementos de tipo elem1.
-

Selectores CSS

- **[attribute=value]**: Todos los elementos con un atributo igual a un determinado valor
- **Elem1[attribute=value]**: Todos los elementos de tipo Elem1 con un atributo igual a un determinado valor
- **Elem1:focus**: El elemento de tipo elem1 que tiene el foco
- **Elem1:hover**: El elemento de tipo elem1 sobre el que está el ratón
- [Más](#)

• **Ver Ejemplo 9**

Modificación de propiedades CSS

- Hemos visto como modificar ciertos aspectos de los elementos seleccionados del árbol DOM con innerHTML, textContent, etc.
- Con el atributo 'style' podemos modificar las propiedades CSS del objeto de tipo 'Elements' devuelto.

- [Ver Ejemplo 10](#)
-