EDUCA2 – ¡Domina las Matemáticas y Programación!

Curso de JavaScript y el DOM

Preparador: Alexanyer Naranjo

Guía Teórica N°02

Selección de Nodos

Una vez construido automáticamente el árbol completo de nodos DOM, ya es posible utilizar las funciones DOM para acceder de forma directa a cualquier nodo del árbol. Como acceder a un nodo del árbol es equivalente a acceder a *"un trozo"* de la página, una vez construido el árbol, ya es posible manipular de forma sencilla la página: acceder al valor de un elemento, establecer el valor de un elemento, mover un elemento de la página, crear y añadir nuevos elementos, entre otros.

Veamos cuáles son las maneras que tenemos con JavaScript para acceder a cada uno de los nodos pertenecientes en el árbol.

getElementsByTagName():

La función getElementsByTagName(nombreEtiqueta) obtiene todos los elementos de la página XHTML cuya etiqueta sea igual que el parámetro que se le pasa a la función. El siguiente ejemplo muestra cómo obtener todos los párrafos de una página XHTML:

```
let parrafos = document.getElementsByTagName("p");
```

El valor que se indica delante del nombre de la función (en este caso, document) es el nodo a partir del cual se realiza la búsqueda de los elementos. En este caso, como se quieren obtener todos los párrafos de la página, se utiliza el valor document como punto de partida de la búsqueda.

El valor que devuelve la función es un array con todos los nodos que cumplen la condición de que su etiqueta coincide con el parámetro proporcionado. El valor devuelto es un array de nodos DOM, no un array de cadenas de texto o un array de objetos normales. Por lo tanto, se debe procesar cada valor del array de la forma que se muestra en las siguientes secciones.

getElementById():

La función getElementById() es la más utilizada cuando se desarrollan aplicaciones web dinámicas. Se trata de la función preferida para acceder directamente a un nodo y poder leer o modificar sus propiedades.

```
var cabecera = document.getElementById("cabecera");
```

La función getElementById() devuelve el elemento XHTML cuyo atributo id coincide con el parámetro indicado en la función. Como el atributo id debe ser único para cada elemento de una misma página, la función devuelve únicamente el nodo deseado.

getElementsByClassName():

Retorna un objecto similar a un array de los elementos hijos que tengan todos los nombres de clase indicados. Cuando es llamado sobre el objeto document , la búsqueda se realiza en todo el document, incluido el nodo raíz. También puedes llamar getElementsByClassName() sobre cualquier elemento; en ese caso retornara sólo los elementos hijos del elemento raíz indicado que contengan los nombres de clase indicados.

```
document.getElementsByClassName('prueba');
```

querySelector():

Devuelve el primer elemento del documento (utilizando un recorrido primero en **profundidad pre-ordenado** de los nodos del documento) que coincida con el grupo especificado de selectores.

```
let elemento = document.querySelector('.contenedor');
let especial = document.querySelector('#texto-especial');
```

querySelectorAll():

El método querySelectorAll () devuelve todos los elementos del documento que coinciden con un selector o selectores CSS especificados, como un objeto NodeList estático.

El objeto NodeList representa una colección de nodos. Se puede acceder a los nodos mediante números de índice. El índice comienza en 0.

```
let elementos = document.querySelectorAll(".items");
```

¡Nos vemos en la próxima!

Si deseas profundizar muchísimo más en el recorrido de árboles que se realiza al momento de utilizar el método **querySelector()** y **querySelectorAll()**, te invitamos a visitar el siguiente enlace donde se hace una excelente explicación sobre estos algoritmos de búsqueda en árboles.

https://runestone.academy/runestone/static/pythoned/Trees/RecorridosDeArboles.html