

**Proyecto de Medio Curso Javascript.**

**Coordinador:** Andrés Díaz Soto.

**Estudiante:** Cristian Felipe.

**Objetivo:** Investigar sobre los métodos document.querySelector () y document.querySelectorAll (), y crear ejemplos de cada uno.

**1-¿Para qué sirven los métodos document.querySelector () y document.querySelectorAll ()?**

**document.querySelector ():** Este método nos retorna el primer elemento del documento, especificado por medio de un o varios selectores, ya sea por medio de la clase o bien, por medio del Id (solo debería de haber un Id por documento).

Si el selector no hace match con los elementos a buscar, nos retorna Null o nulo.

**Sintaxis:** element = document.querySelector (selectorName);

**document.querySelectorAll ():** Este método nos retorna todos los elementos en un documento, especificado por medio de un o varios selectores (se separan por comas). El resultado se retorna en un NodeList.

NodeList es una colección de objetos, no es un Array, pero se puede recorrer como un Array, por medio de forEach (), también al igual que los Arrays, los NodeList se accesan por medio de los números de Index, inicia en [0].

**Sintaxis:** element = document.querySelectorAll (selectorsName);



**Diferencias entre .querySelector () y .querySelectorAll ()?**

Para entender más las diferencias entre estos métodos, quise compararlos con **.getElementsByClassName ()**, **.getElementById ()** y **.getElementsByTagName ().** Ya que cada uno de estos métodos nos retorna un elemento o una colección de elementos, ya sea por medio de Array o NodeList.

**.querySelector ()** : Nos retorna un solo elemento en el documento, que coincida con el selector CSS.

**document.querySelector ("p.example").style.backgroundColor = "red";**

Retorna un elemento (**p**) con la clase (**.example**), y le aplica un color de fondo, rojo.

**.querySelectorAll () :** Retorna una colección de todos los elementos en el documento , que coincidan con el selector.

**let x = document.querySelectorAll ("p.example");**

**x[1].style.backgroundColor = "red";**

Retorna un elemento (p) con la clase (**.example**), y le aplica un color de fondo rojo. Se especifica el numero de indexado.

En una colección de (**x**) cantidad de (**párrafos**), cambia el color del index [**1**].

**.getElementsByClassName ():** Retorna una colección de elementos, que coincidan por la Clase, no por el selector.

**let x = document.getElementsByClassName ("example");**

**x [0] .style.backgroundColor = "red";**

Retorna un elemento (**x**) con la clase (**.example**), y le aplica un color de fondo rojo. Se especifica el numero de indexado.

**.getElementById () :** Retorna un solo elemento en el documento que se especifica por el Id, solo hay uno en el documento.

**let x = document.getElementById ("example");**

**x.style.color = "red";**

Retorna un elemento (**x**) con el Id (**#example**), y le aplica el color. Solo debería haber un elemento (**x**) con el Id.

**.getElementsByTagName ():** Retorna una colección de elementos que tienen un nombre de etiqueta específico.

**let x = document.getElementsByTagName ("p");**

**let i;**

**for (i = 0; i < x.length; i++) {**

**x[i].style.backgroundColor = "red";**

Retorna (**x**) cantidad de elementos que responden a la etiqueta (**p**), por cada etiqueta (**p**) en el documento se va a aplicar un color de fondo rojo.



**2-Suponga que se obtiene un elemento HTML usando document.getElementById(‘elemento1’) y se almacena en una variable.**

**let miElemento = document.getElementById (‘Elemento1’);**

Averigüe como obtener la siguiente información:

El elemento Padre de miElemento:

**miElemento.parentElement;**

**miElemento.parentNode;**

Los elementos hijos de miElemento:

**miElemento.children;**

Cantidad de elementos hijos de miElemento:

**miElemento.childElementCount;**

El elemento inmediatamente anterior a miElemento (descendiente del mismo padre):

**miElemento.previousElementSibling;**

El elemento inmediatamente posterior a miElemento (descendiente del mismo padre):

**miElemento.nextElementSibling;**

Insertar un elemento como primer descendiente, o bien en una posición arbitraria dentro del elemento padre.

**miElemento.append(“ ”);**

**miElemento.prepend(“ ”);**

**miElemento.InsertBefore(“ ”);**