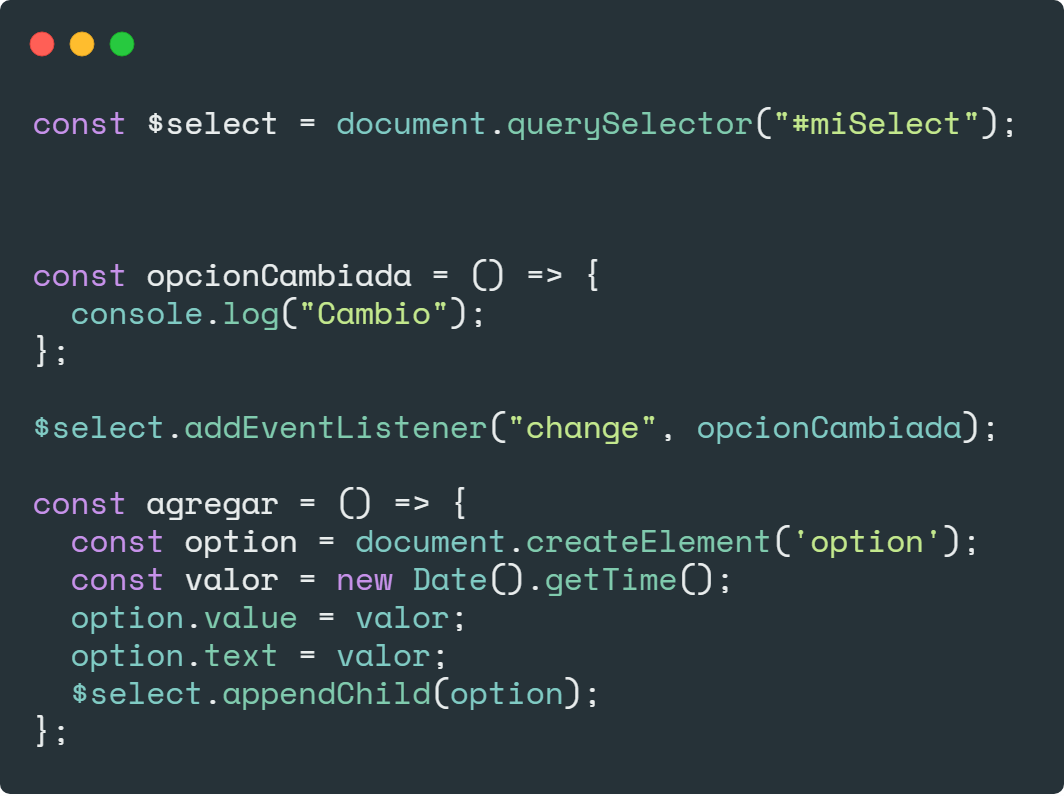
# Select y JavaScript: agregar, limpiar, obtener y escuchar cambios del elemento HTML

El select de HTML **se maneja a través de JavaScript**. Al select también se le conoce como **lista desplegable**, **Dropdown list** o simplemente select.

[](https://parzibyte.me/blog/wp-content/uploads/2019/04/Manipular-select-con-JavaScript-puro.png)

**Manipular select con JavaScript puro**

Hoy veremos cómo trabajar con **JavaScript y los elementos de tipo select** con **option**. Vamos a ver cómo:

1. Obtener valor seleccionado de un **select**
2. Agregar opciones al elemento **select**
3. Obtener **opción** seleccionada
4. Limpiar el **select**
5. Escuchar cuando el valor del select **cambia**, es decir, que se selecciona otro valor

Está de más mencionar que esto lo haremos con **JavaScript puro**, sin usar frameworks ni librerías externas.

Al final tendremos un **ejemplo** en donde manejamos un **select con JavaScript**, el cual se puede probar [en este enlace](https://select-con-javascript--parzibyte.repl.co/).

Primero: obtener referencia al elemento

Podemos usar document.getElementById o document.querySelector, yo prefiero la segunda forma pues es más moderna, aunque su soporte no está para navegadores antiguos.

Suponiendo que el select tiene el id “miSelect” podemos obtenerlo así:

document.getElementById("miSelect");

O así:

document.querySelector("#miSelect");

Podemos guardar el resultado en una [variable o constante](https://parzibyte.me/blog/2018/08/24/const-var-let-en-javascript/), recomiendo una constante pero eso es cosa de cada desarrollador.

Yo lo haré así:

|  |  |
| --- | --- |
|  | const $select = document.querySelector("#miSelect"); |

[**view raw**](https://gist.github.com/parzibyte/8afdd1cfbcf0280ebe988ec67607a7e2/raw/5aa8bc8e18f2fab17ec7c751b54bacefcc038941/definir.js)[**definir.js**](https://gist.github.com/parzibyte/8afdd1cfbcf0280ebe988ec67607a7e2#file-definir-js)hosted with  by [**GitHub**](https://github.com/)

Estoy **prefijando con el signo de dólar** para saber que no es un valor normal, sino un elemento.

Agregar option a select con JavaScript

Para agregar una opción a la lista desplegable debemos crear un elemento HTML de tipo option, establecer el text y value para finalmente llamar al método appendChild del select.

Aquí el ejemplo:

|  |  |
| --- | --- |
|  | const agregar = () => { |
|  | const option = document.createElement('option'); |
|  | const valor = new Date().getTime(); |
|  | option.value = valor; |
|  | option.text = valor; |
|  | $select.appendChild(option); |
|  | }; |

[**view raw**](https://gist.github.com/parzibyte/aadb86a1fd4db578e978ed8edd98580b/raw/23163da44a66989113cacd773db85fd325382bad/agregar.js)[**agregar.js**](https://gist.github.com/parzibyte/aadb86a1fd4db578e978ed8edd98580b#file-agregar-js)hosted with  by [**GitHub**](https://github.com/)

El texto y valor pueden obtenerse de cualquier lugar, yo le pongo los milisegundos actuales para que no se repitan, pero es un simple ejemplo. De esta manera podemos agregar opciones al elemento.

Diferencia entre text y value

El texto es lo que ve el usuario, y el valor algo así como el identificador. Ambas cosas pueden ser las mismas, pero depende de tus necesidades.

Por ejemplo, si tenemos una **lista de empleados** yo pondría el nombre en text, y el id en value, así el usuario vería el nombre pero yo buscaría por id.

Escuchar cuando cambia la opción

Para saber si el usuario elige otra opción o ***el select cambia de opción***agregamos un listener en el evento change del select.

Antes de ello definimos la función que se ejecuta cuando cambie, la cual solo imprime un mensaje.

|  |  |
| --- | --- |
|  | const opcionCambiada = () => { |
|  | console.log("Cambio"); |
|  | }; |
|  |  |
|  | $select.addEventListener("change", opcionCambiada); |

[**view raw**](https://gist.github.com/parzibyte/70144387f5642c40142d157121e645d7/raw/fe6fdad5f7e1e6d9f5f48fa86a70c37884112d1d/escuchar.js)[**escuchar.js**](https://gist.github.com/parzibyte/70144387f5642c40142d157121e645d7#file-escuchar-js)hosted with  by [**GitHub**](https://github.com/)

A continuación veremos cómo obtener el valor seleccionado, también puedes llamar a esa función en el evento que acabamos de ver.

Obtener opción seleccionada de select con JavaScript

El elemento tiene una propiedad llamada selectedIndex, y otra llamada options. La propiedad options es un arreglo de las opciones, y selectedIndex el índice que está seleccionado.

Por lo tanto solo **accedemos al arreglo en esa posición**. Es importante mencionar que si no hay elementos, selectedIndex estará **en -1**.

|  |  |
| --- | --- |
|  | const mostrar = () => { |
|  | const indice = $select.selectedIndex; |
|  | if(indice === -1) return; // Esto es cuando no hay elementos |
|  | const opcionSeleccionada = $select.options[indice]; |
|  | alert(`Texto: ${opcionSeleccionada.text}. Valor: ${opcionSeleccionada.value}`); |
|  | }; |

[**view raw**](https://gist.github.com/parzibyte/90f6bd8ca09451e110e5fce171d6d7f7/raw/f46bbfbf6e9e086ff3e219b3f3a93e5d2d5b24a2/mostrar.js)[**mostrar.js**](https://gist.github.com/parzibyte/90f6bd8ca09451e110e5fce171d6d7f7#file-mostrar-js)hosted with  by [**GitHub**](https://github.com/)

En el ejemplo obtenemos la opción seleccionada, la cual es un objeto que tiene text y value. Lo estoy mostrando en una alerta pero podríamos hacer cualquier otra cosa.

Limpiar select con JavaScript

Ahora veamos la última operación, la cual es **limpiar completamente el select**. Para eso usamos el método remove en un ciclo. De esta manera quitamos cada elemento.

Es importante hacer el ciclo desde la cantidad de opciones hasta 0, ya que la función remove reordena el arreglo y si lo hacemos desde 0 hasta la cantidad de opciones no funcionará.

|  |  |
| --- | --- |
|  | const limpiar = () => { |
|  | for (let i = $select.options.length; i >= 0; i--) { |
|  | $select.remove(i); |
|  | } |
|  | }; |

[**view raw**](https://gist.github.com/parzibyte/586d9ad4981e1434ec73ca0c3a14be5c/raw/27bf4e3fca55bc56c43d4d62e871f0ceae7dbf2b/limpiar.js)[**limpiar.js**](https://gist.github.com/parzibyte/586d9ad4981e1434ec73ca0c3a14be5c#file-limpiar-js)hosted with  by [**GitHub**](https://github.com/)

Igualmente si solamente quieres quitar un elemento de la lista desplegable llama a remove sin el ciclo, pasándole **el índice de la opción**.

Poniendo todo junto

Podemos poner la declaración de los elementos, sus listeners y funciones juntas. Queda así:

|  |  |
| --- | --- |
|  | const $select = document.querySelector("#miSelect"); |
|  |  |
|  |  |
|  |  |
|  | const opcionCambiada = () => { |
|  | console.log("Cambio"); |
|  | }; |
|  |  |
|  | $select.addEventListener("change", opcionCambiada); |
|  |  |
|  | const agregar = () => { |
|  | const option = document.createElement('option'); |
|  | const valor = new Date().getTime(); |
|  | option.value = valor; |
|  | option.text = valor; |
|  | $select.appendChild(option); |
|  | }; |
|  |  |
|  | const limpiar = () => { |
|  | for (let i = $select.options.length; i >= 0; i--) { |
|  | $select.remove(i); |
|  | } |
|  | }; |
|  |  |
|  | const mostrar = () => { |
|  | const indice = $select.selectedIndex; |
|  | if(indice === -1) return; // Esto es cuando no hay elementos |
|  | const opcionSeleccionada = $select.options[indice]; |
|  | alert(`Texto: ${opcionSeleccionada.text}. Valor: ${opcionSeleccionada.value}`); |
|  | }; |
|  |  |
|  | document.addEventListener("DOMContentLoaded", () => { |
|  | document.querySelector("#btnAgregar").addEventListener("click", agregar); |
|  | document.querySelector("#btnLimpiar").addEventListener("click", limpiar); |
|  | document.querySelector("#btnMostrar").addEventListener("click", mostrar); |
|  | }); |

[**view raw**](https://gist.github.com/parzibyte/72253e3b950f5805c331dd45571eb7d1/raw/e3e840307291888e45dcf4243da5c3cb5076e972/select.js)[**select.js**](https://gist.github.com/parzibyte/72253e3b950f5805c331dd45571eb7d1#file-select-js)hosted with  by [**GitHub**](https://github.com/)

El código HTML que tiene el select es el siguiente:

|  |  |
| --- | --- |
|  | <!DOCTYPE html> |
|  | <html> |
|  |  |
|  | <head> |
|  | <meta charset="utf-8"> |
|  | <meta name="viewport" content="width=device-width"> |
|  | <title>Select en JavaScript</title> |
|  | </head> |
|  |  |
|  | <body> |
|  | <h1>Select en JavaScript</h1> |
|  | <p> |
|  | <a target="\_blank" href="//parzibyte.me/blog">By Parzibyte</a> |
|  | </p> |
|  | <select id="miSelect"></select> |
|  | <br> |
|  | <button id="btnAgregar">Agregar</button> |
|  | <button id="btnLimpiar">Limpiar</button> |
|  | <button id="btnMostrar">Mostrar seleccionado</button> |
|  | <script src="script.js"></script> |
|  | </body> |
|  | </html> |

[**view raw**](https://gist.github.com/parzibyte/7ec666267524b25645490abbc83247a6/raw/3c3802a9560ee7047e742b7aaffc6c64de6e5159/index.html)[**index.html**](https://gist.github.com/parzibyte/7ec666267524b25645490abbc83247a6#file-index-html)hosted with  by [**GitHub**](https://github.com/)

Puedes probar el ejemplo [aquí](https://select-con-javascript--parzibyte.repl.co/).

Conclusión

Sin importar el framework que utilicemos, todo es manejado con JavaScript en los niveles más bajos.

Además, al hacer todo con JavaScript evitamos sobrecargar nuestras páginas con librerías de terceros que hacen las cosas más fáciles pero a su vez más pesadas.