educación IT
CODE YOUR FUTURE

Bootcamp Java Developer

Fase 2 - Java Web Developer

Módulo 18

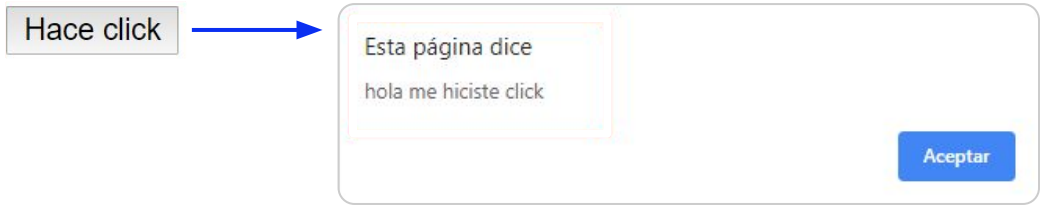
Eventos

Entender los eventos

Los eventos son algo que ***les pasa*** a los **elementos del HTML**:

```
<button onclick="alert('hola me hiciste click')"> Hace click </button>
```

Esa línea de código indica que **al hacer clic al botón, se disparará una alerta.**



Veamos otros eventos:

```
<button onclick="alert('hola me hiciste click')"> Hace click </button>
<button onmouseover="alert('hola me pasaste el mouse por encima')"> Acerca el cursor </button>
<button onmouseout="alert('Alejá el cursor')"> Aleja el cursor </button>
<button onmousemove="alert('Mové el cursor del mouse sobre mí')"> Mové el cursor </button>
```

Eventos desde el HTML

Es posible usar eventos relacionados con el teclado. En la imagen de abajo, se genera una **caja de texto (`input`)**:

```
10
11      <input type="text" onchange="console.log('Cambiaste el valor')">
12      <input type="text" onblur="console.log('Quitaste el foco de la caja')">
13      <input type="text" onfocus="console.log('Hiciste foco en la caja')">
14      <input type="text" onkeypress="console.log('Presionaste una tecla dentro de la caja')">
15      <input type="text" onkeyup="console.log('Dejaste de presionar una tecla dentro de la caja')">
16
```

Como vemos, hay diferentes eventos, pero también hay diferentes formas de trabajarlos, en el caso anterior **se hace desde el propio HTML**.

Eventos desde el código

También **se puede llamar a los eventos directamente desde el código**.

```
10
11      <button id="boton">Haceme click</button>
12
13      <script>
14
15          document.querySelector('button').onclick = alerta;
16
17          function alerta() {
18              console.log('Eventos desde el código');
19          }
20
21      </script>
22
```

Eventos con parámetros

Ahora toma especial interés lo antes visto, porque es posible **pasar parámetros** y hacer lo siguiente:

```
11      <button onclick="alerta('Ana')">Saludar a Ana</button>
12
13      <script>
14
15          function alerta(nombre) {
16              console.log('Hola '+nombre);
17          }
18
19      </script>
```

Lo interesante **es que podemos reutilizar esa misma función para diferentes botones y pasarle** distintos datos según se necesite.

```
<button onclick="saludarA('Ana')"> Ana </button>
<button onclick="saludarA('Pedro')"> Pedro </button>
<button onclick="saludarA('Juan')"> Juan </button>
<button onclick="saludarA('María')"> María </button>
```

El código anterior sigue intacto porque la función es una sola, no la hemos cambiado. El resultado será que, al hacer clic en el botón de *Juan,* saludará a Juan.

De esta forma no es necesario generar múltiples funciones, sino una sola y reutilizable.

También con el mismo propósito podemos hacer lo siguiente:

```
11
12      <button>Ana</button>
13      <button>Pedro</button>
14      <button>Juan</button>
15      <button>María</button>
16
17      <script>
18          var boton = document.getElementsByTagName('button')
19
20          for (var i = 0; i < 2; i++) {
21              boton[i].onclick = saludar;
22          }
23
24          function saludar() {
25              console.log('Hola');
26          }
27      </script>
28
```

Lo explicaremos en detalle:

Primero, se generan **cuatro (4) botones.**

Si se tratara de una interfaz en la que los botones fueran variando sus estados, sería muy tedioso modificar el HTML. La intención no es trabajar desde los botones, sino desde el código.

En JS, estos botones son partes del DOM, y sería poco práctico **asignarle a cada uno un `id` o `class`** porque serán dinámicos.

```
<button> Ana </button>
<button> Pedro </button>
<button> Juan </button>
<button> María </button>
```

Una mejor **forma de acceso a los elementos** es **`document.getElementsByTagName`**, esto es, tomamos a los elementos **por su nombre dentro del HTML:**

```
var boton = document.getElementsByTagName('button')
```

Simplifica la forma de acceso, pero es necesario **individualizar los elementos** para que al hacer clic en cada uno, hagan algo en forma individual.

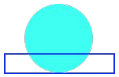
```
for(var i = 0; i <2; i++){

boton[i].onclick = saludar

}
```

Vemos que al hacer clic en cada botón el resultado es que se muestra el mensaje por consola.

Hemos avanzado mucho, pero la idea es que a su vez cada botón pase el valor de su contenido. Realizaremos las modificaciones pertinentes para lograrlo.



```
<button>Ana</button>
<button>Pedro</button>
<button>Juan</button>
<button>María</button>

<script>
    var boton = document.getElementsByTagName('button')

    for (var i = 0; i < 2; i++) {
        boton[i].onclick = function() {
            saludar(this.innerHTML)
        };
    }

    function saludar(nombre) {
        console.log(nombre);
    }
</script>
```

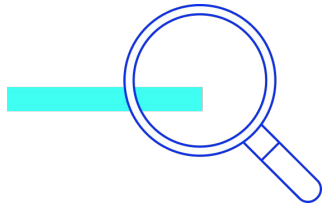
Se debe pasar el texto contenido en cada botón. Lo haremos con **`innerHTML()`**, que obtiene el texto del botón. Como lo hacemos dentro del **`for`**, e**l `innerHTML` será diferente según sobre qué botón se haga clic.**

Por otro lado, debemos pasar esto como parámetro. Por lo tanto, lo hacemos con esta sintaxis:

```
function(){ saludarA(this.innerHTML)}
```

Revisión

- Repasar el concepto de **Evento.**
- Investigar todos los **eventos posibles.**
- Combinar **funciones con eventos.**
- Comprender cómo funciona **`document.getElementsByTagName`**.
- Integrar un **bucle para reutilizar** una función desde el código.
- Trabajar en el **Proyecto Integrador.**
- Realizar las preguntas necesarias al/la docente antes de continuar.



educación IT
CODE YOUR FUTURE

¡Sigamos trabajando!