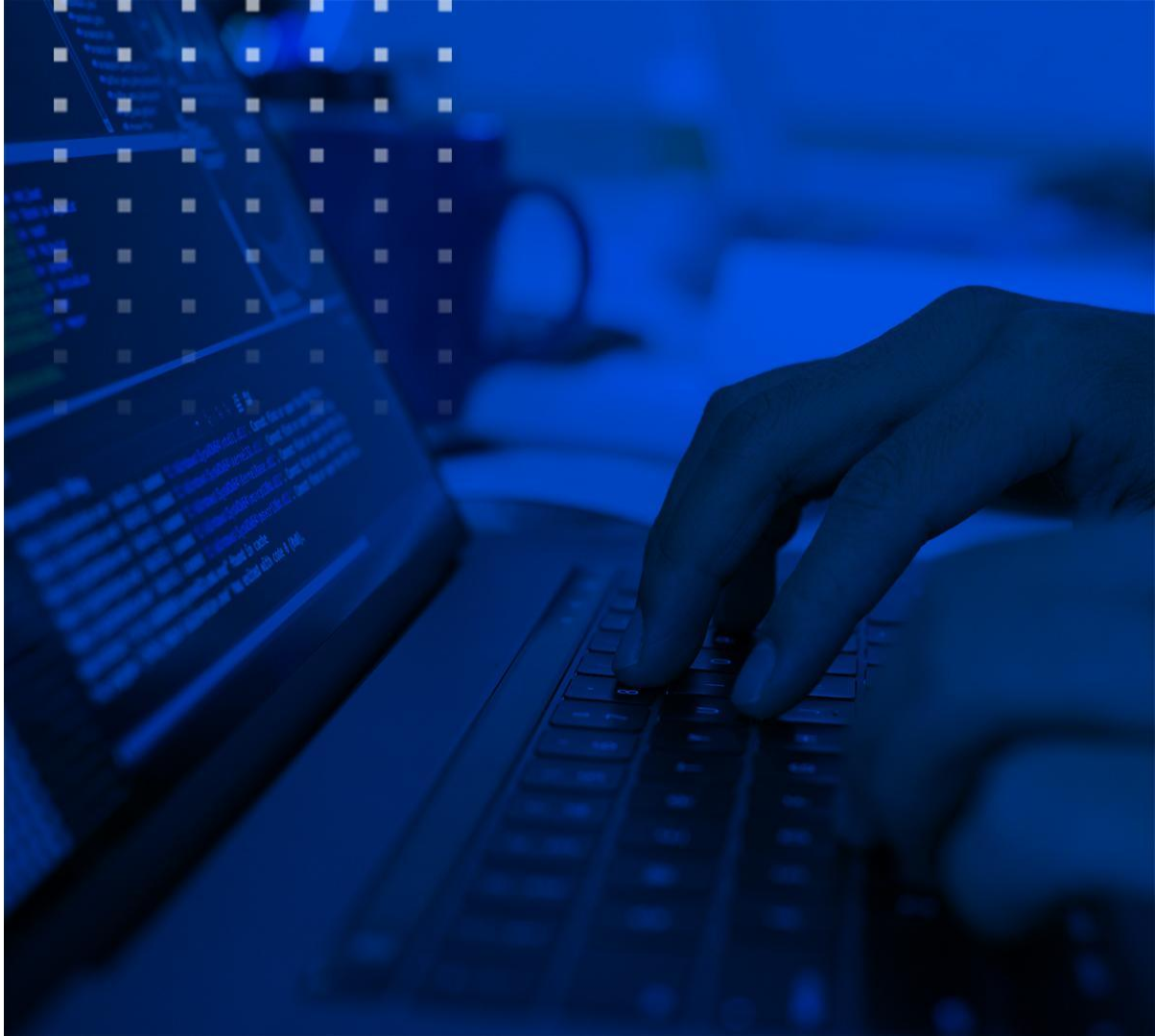
incluyeme.com

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

CURSO

FULL STACK DEVELOPER
NIVEL INICIAL



UNIDAD 7
JavaScript
Funciones, DOM y Eventos

¿Qué va a hacer una computadora si no la prendemos?

¿Qué va a hacer una computadora si no la prendemos?

Nada :(

FUNCIONES

```
function saludar() {
        alert("Hola  Mundo!");
}
```

Una función un conjunto de instrucciones que realiza una tarea y se define con la palabra **function** y luego el nombre que le queramos poner.

Luego entre las llaves indicaremos todas las instrucciones de la función.

FUNCIONES

```
function saludar() {
        alert("Hola  Mundo!");
}
```

¿Que hizo el código hasta ahora?

FUNCIONES

```
function saludar() {
        alert("Hola  Mundo!");
}
```

¿Que hizo el código hasta ahora?

NADA. *Como la computadora* -> Nos falta ~~prender la computadora~~ invocar la función.

FUNCIONES

```
function saludar() {
        alert("Hola  Mundo!")
}

saludar();
saludar();
saludar();
```

¡Ahora sí!

FUNCIONES

```
function saludar() {
        alert("Hola  Mundo!")
}
```

Agreguémosle ingredientes ahora

FUNCIONES

```
function saludar() {
        alert("Hola  Mundo!")
}
```

Agreguémosle ~~ingredientes~~ parámetros ahora

FUNCIONES

```
function saludar(nombre) {
  alert("Hola " + nombre);
}
```

FUNCIONES

```
function saludar(nombre) {
  alert("Hola " + nombre);
}
```

¡Importante!

El operador + sirve para concatenar strings

FUNCIONES

```
function saludar(nombre) {
  alert("Hola " + nombre);
}
```

¿Qué hace nuestro código ahora?

FUNCIONES

```
function saludar(nombre) {
  alert("Hola " + nombre);
}
```

¿Qué hace nuestro código ahora?

NADA. Pero tenemos una **función** que *sabe* saludar nombres

FUNCIONES

```
function saludar(nombre) {
  alert("Hola " + nombre);
}
```

Para que se ejecute, tenemos que **invocarla** con el **parámetro** que necesita

FUNCIONES

```
function saludar(nombre) {
  alert("Hola " + nombre);
}

saludar("Ana");
saludar("Carolina");
saludar("Ignacio");
```

¡Ahora sí!

FUNCIONES

```
function saludar(nombre, apellido) {
  alert("Hola " + nombre + " " + apellido);
}
```

Ahora tenemos una función más formal.

Sabe saludar con nombre y apellido.

FUNCIONES

```
function saludar(nombre, apellido) {
  alert("Hola " + nombre + " " + apellido);
}
```

Ahora tenemos una función más formal.

Sabe saludar con nombre y apellido.

¿Cuántos **parámetros** tiene esta función?

FUNCIONES

```
function saludar(nombre, apellido) {
  alert("Hola " + nombre + " " + apellido);
}

saludar("Harry", "Potter");
saludar("Hermione", "Granger");
saludar("Ron", "Weasley");
```

¡Bien!

¡A Practicar!

Crear una función que reciba el parámetros ***nombre, apellido*** *y* ***serie favorita*** y que devuelva un saludo:

- Un nombre
- Apellido
- Serie favorita

Hola, *nombre* + *apellido*, tu serie favorita es *serieFavorita*

Invocar la función con los parámetros requeridos.

VALORES DE RETORNO

VALORES DE RETORNO

Las funciones pueden devolver un resultado.
La sentencia *return* **finaliza** la ejecución de la función
y especifica un valor para ser devuelto.

```
function sumar(numero1, numero2) {
  return numero1 + numero2;
}
```

¡Importante!

El operador + también sirve para sumar números

Otros operadores: (-, *, /, %)

VALORES DE RETORNO

Las funciones pueden devolver un resultado

```
function sumar(numero1, numero2) {
  return numero1 + numero2;
}
```

¿Qué hace esta función?

VALORES DE RETORNO

Las funciones pueden devolver un resultado

```
function sumar(numero1, numero2) {
  return numero1 + numero2;
}
```

Suma dos números y nos **devuelve** el resultado

VALORES DE RETORNO

¿Qué hacemos con lo que nos devuelve una función?

```
function sumar(numero1, numero2) {
  return numero1 + numero2;
}
```

VALORES DE RETORNO

```
function sumar(numero1, numero2) {
  return numero1 + numero2;
}

let resultado = sumar(40, 2);
console.log("40 + 2 es igual a: " + resultado);
```

Podemos guardarlo en una variable

VALORES DE RETORNO

```
function sumar(numero1, numero2) {
  return numero1 + numero2;
}

console.log("40 + 2 es igual a: " + sumar(40, 2));
```

O usarlo directamente

¡A Practicar!

Crear una función que reciba el parámetro *nombre* y que devuelva el largo de ese nombre.

Guardar el valor de retorno en una variable y mostrarlo en consola.

Por ejemplo: si ingresamos “Ana”. El resultado debería ser 3.

SCOPE

SCOPE

Es el ***alcance*** que una variable tendrá en tu código. Existen dos tipos de scope, el **scope global** y el **scope local.**

SCOPE LOCAL

```
function sumarEdad(edad) {
  let num = 1;
  return edad + num;
}

//Acá ya no podemos usar num
```

Es cuando se puede acceder a una variable únicamente en **cierta parte del código**. Por ejemplo, cuando variable está declarada dentro de un **bloque o una función**.

SCOPE GLOBAL

```
let num = 1;
function sumarEdad(edad) {
  return edad + num;
}

//Acá podemos usar num
```

Es cuando la variable está declarada **fuera** de una función o de un bloque.
Se puede acceder a éstas desde **cualquier parte** del código.

DOM

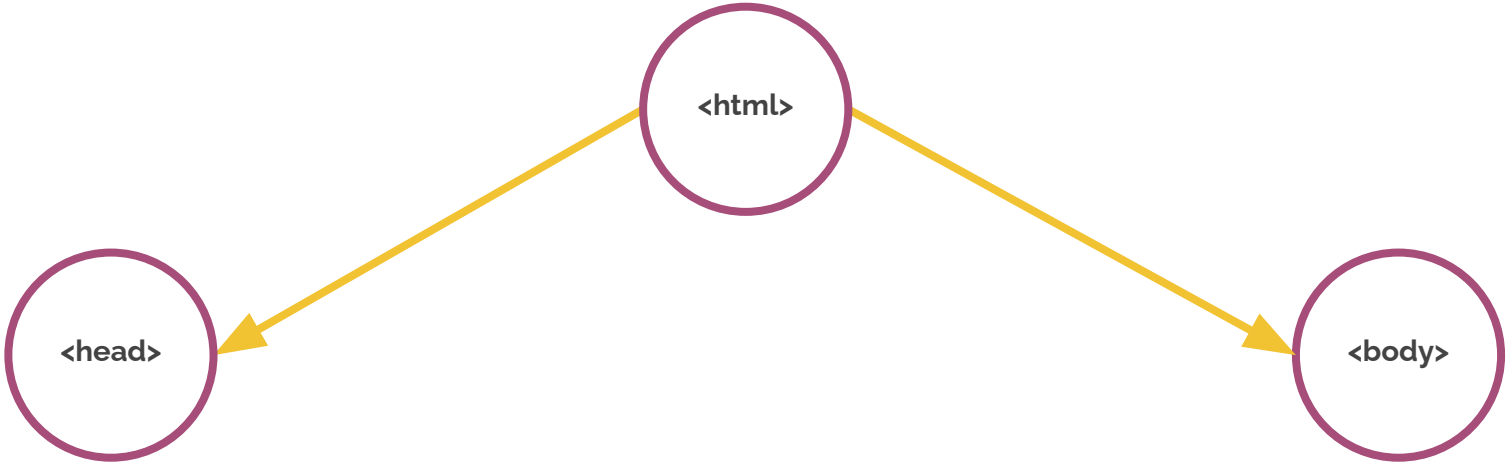
¿QUE ES EL DOM?

El DOM (Document Object Model, en español Modelo de Objetos del Documento) es la **estructura de objetos** que genera el navegador cuando se carga un documento y **se puede alterar mediante Javascript** para cambiar dinámicamente los contenidos y aspecto de la página.

VEAMOS NUESTROS HTML

```
<html>
    <head>
        <title>Mi página</title>
    </head>
    <body>
        <h1>Mi título principal</h1>
        <a href="https://google.com">Ir a Google</a>
    </body>
</html>
```

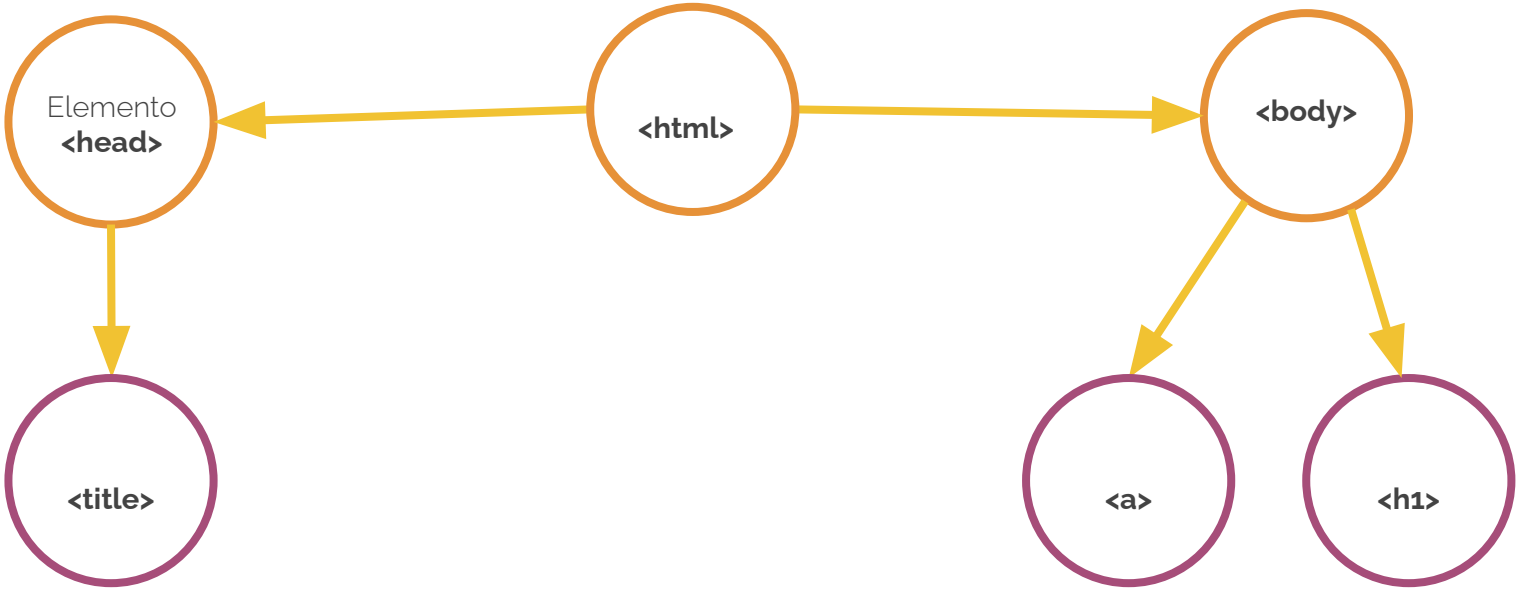
NUESTRO DOM



VEAMOS NUESTROS HTML

```
<html>
    <head>
        <title>Mi página</title>
    </head>
    <body>
        <h1>Mi título principal</h1>
        <a href="https://google.com">Ir a Google</a>
    </body>
</html>
```

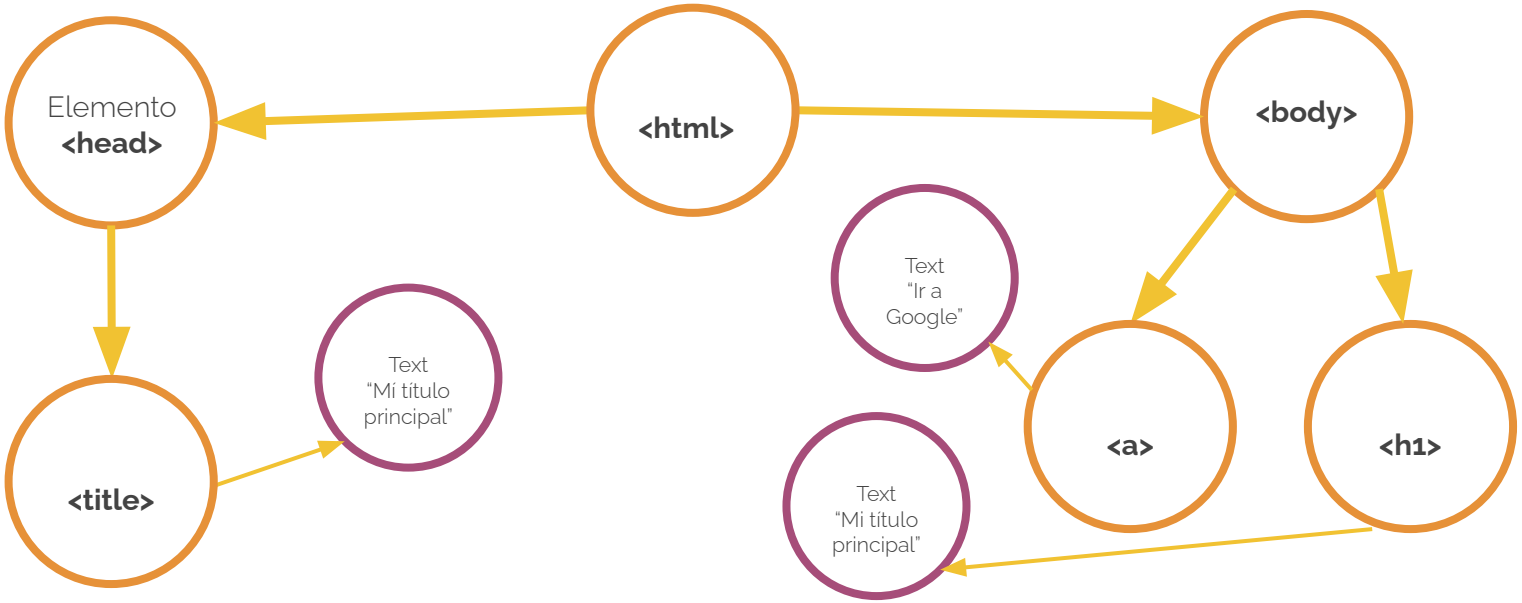
NUESTRO DOM



VEAMOS NUESTROS HTML

```
<html>
    <head>
        <title>Mi página</title>
    </head>
    <body>
        <h1>Mi título principal</h1>
        <a href="https://google.com">Ir a Google</a>
    </body>
</html>
```

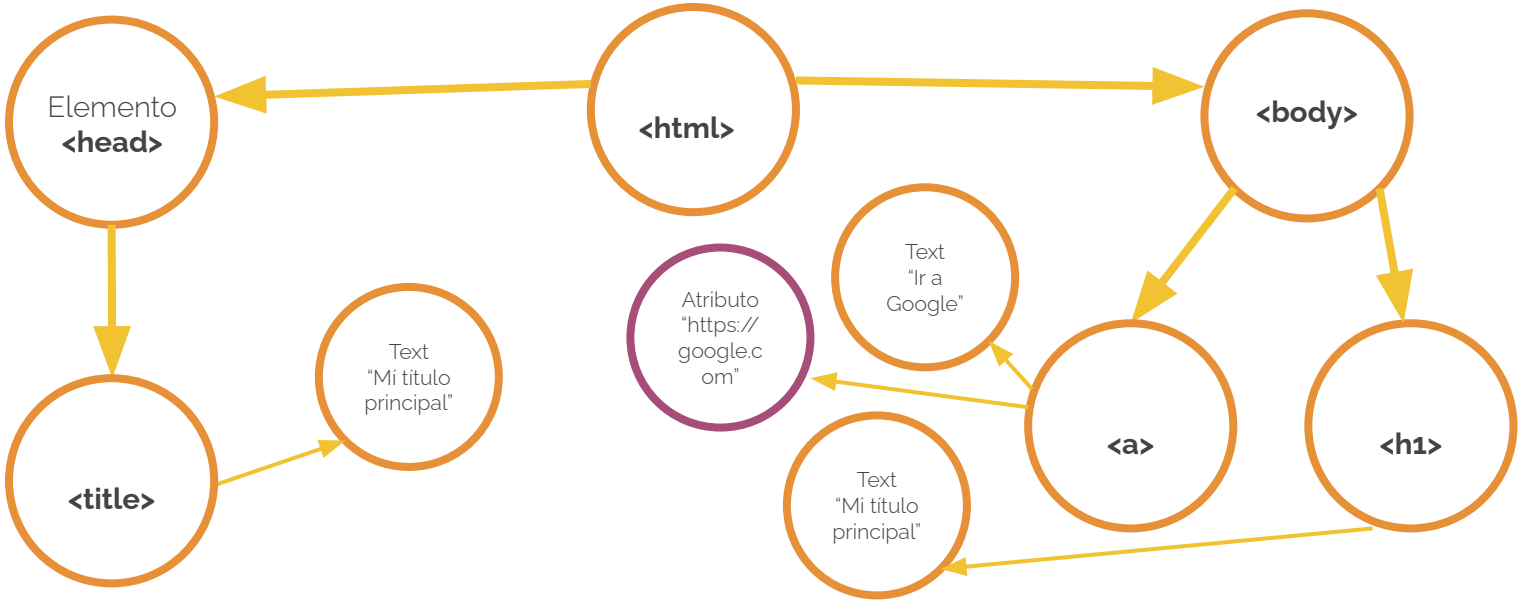
NUESTRO DOM


Elemento
<head>
<html>
<body>
Text
"Mi título
principal"
<title>
Text
"Ir a
Google"
<a>
<h1>
Text
"Mi título
principal"

VEAMOS NUESTROS HTML

```
<html>
    <head>
        <title>Mi página</title>
    </head>
    <body>
        <h1>Mi título principal</h1>
        <a href="https://google.com">Ir a Google</a>
    </body>
</html>
```

NUESTRO DOM


Elemento
<head>
<html>
<body>
Text
"Ir a
Google"
Atributo
"https://
google.c
om"
Text
"Mí título
principal"
<title>
<a>
<h1>
Text
"Mi título
principal"

MÉTODOS Y PROPIEDADES DEL DOM

MÉTODOS Y PROPIEDADES

En JS todos los elementos que manipulamos son objetos.
Éstos tienen **propiedades** y **métodos**.

Veamos un ejemplo...

MÉTODOS Y PROPIEDADES

MÉTODOS Y PROPIEDADES

Objeto: Gato

Propiedades: Color, Edad, Sexo

Métodos: ronronear(), maullar()

¿EL DOM ES UN OBJETO?

La forma de acceder al DOM es a través de un objeto llamado **document**, que representa el árbol DOM de la página o pestaña del navegador donde nos encontramos.

¿EL DOM ES UN OBJETO?

Esto quiere decir, que...

¡El DOM tiene **métodos** y **propiedades**!

Veamos algunos ejemplos de esto...

MÉTODOS

Los métodos sirven para hacer **acciones**, modificar o encontrar elementos.

MÉTODOS
document.querySelector()

Nos permite encontrar elementos dentro del documento.

- Para llamar por nombre de etiqueta:
  document.querySelector("h1")
- Para llamar por nombre de ID:
  document.querySelector("#titulo")
- Para llamar por nombre de clase:
  document.querySelector(".miClase")

PROPIEDADES

Las propiedades nos permiten interactuar con el elemento.

- Para acceder al texto dentro de la etiqueta: element.innerText
- Para acceder al estilo dentro de la etiqueta: document.style

¡Importante!

Estas propiedades pertenecen a los elementos dentro del DOM, no directamente al objeto document.

PROPIEDADES

¿Se te ocurre alguna propiedad que hayamos visto hasta ahora?

Sí, **length**. Es una propiedad que nos indica el largo de un string o array.

¡A Practicar!

1. Creá en un archivo un esqueleto HTML básico.
2. Incluí la etiqueta **<h1>** y **<p>**
3. Asignale un ID o CLASS a la etiqueta **<p>**
4. Enlazá tu archivo JS al HTML
5. Usando el método **querySelector** mostrá tu h1 y tu ID o CLASS en consola.
6. Mostrá en consola el texto dentro de la etiqueta h1.

ACCESO AL DOM

Existen distintos métodos para acceder a los elementos del DOM empleando en la clase Document. Los más utilizados son:

- getElementById()
- getElementsByClassName()
- getElementsByTagName()

EVENTOS

EVENTOS

Cuando un usuario visita una página web e interactúa con ella se producen los eventos y con Javascript podemos definir qué queremos que ocurra cuando se produzcan.

EVENTOS

Cuando un usuario visita una página web e interactúa con ella se producen los eventos y con Javascript podemos definir qué queremos que ocurra cuando se produzcan.

Por ejemplo: podemos definir qué pasa cuando un usuario hace click en un botón

EVENTOS on click

```
<button onClick="alert(‘¡Hola!’)">Saludar</button>
```

EVENTOS on click

```
<button onClick="alert('¡Hola!')">Saludar</button>
```

Si lo que tiene hacer nuestro código es más complejo podríamos reemplazar el valor del onClick por una función.

EVENTOS on click

```
<button onClick="alert(‘¡Hola!’)">Saludar</button>
```

Si lo que tiene hacer nuestro código es más complejo podríamos reemplazar el valor del onClick por una función.

```
<button onClick="saludar()">Saludar</button>
```

¡A Practicar!

1. Crea un elemento <button>
2. Al hacer click este botón se debe invocar una función llamada **saludar()** que nos salude.

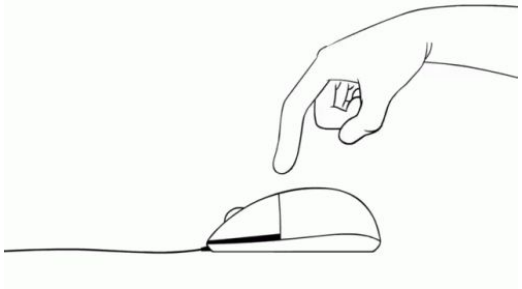
¿Y CUAL CONVIENE USAR?

Las opciones 1 y 2 son las recomendadas, si bien se pueden presentar casos de aplicación específico (por ejemplo, en la opción 1 el nombre del evento puede venir de una variable al usar la propiedad, y esto no puede hacerse en la 2), se identifican como formas de definición de evento equivalentes.

La opción 3, aunque es de fácil implementación, no es recomendada para proyectos en producción ya que no es considerada un buena práctica declarar funciones y código JavaScript dentro del HTML.

¿CÓMO FUNCIONA?

JavaScript permite asignar una función a cada uno de los eventos.
De esta forma, cuando se produce cualquier evento, JavaScript ejecuta su función asociada. Este tipo de funciones se denominan *event handlers* en inglés, y en castellano por "manejadores de eventos".
Los eventos se asocian a cada elemento al cual se lo quiere "escuchar".



```
<!DOCTYPE html>
<html>
  <head>
    <title>Mi primer App</title>
  </head>
  <body>
    <h2> Método: addEventListener </h2>
    <button id="btnPrincipal">CLICK</button>
    <script>
      let boton = document.getElementById("btnPrincipal")
      boton.addEventListener("click", respuestaClick)
      function respuestaClick(){
        console.log("Respuesta evento");
      }
    </script>
  </body>
</html>
```

Definir eventos: Opción 1 addEventListener()

El método addEventListener() permite definir qué evento escuchar sobre cualquier elemento en el código HTML.
El primer parámetro corresponde al nombre del evento y el segundo a la función de respuesta.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Mi primer App</title>
  </head>
  <body>
    <h2> Definir Eventos </h2>
<button id="btnPrincipal">CLICK</button>
    <script>
      let boton =
document.getElementById("btnPrincipal")
      boton.onclick = () =>{console.log("Respuesta
2")}
    </script>
  </body>
</html>
```

Definir eventos: opcion 2

Emplear una propiedad del nodo para definir la respuesta al evento. Las propiedades se identifican con el nombre del evento y el prefijo *on*.También es posible emplear funciones anónimas para definir los manejadores de eventos.

Sintaxis: opción 3

Determinar el evento especificando el manejador de evento en el atributo de una etiqueta HTML. La denominación del atributo es idéntica al de la propiedad de la opción 2 (prefijo *on*)

```
<input type="button" value="CLICK2" onclick="alert('Respuesta 3');" />
```

La función puede declararse entre la comillas o bien tomarse una referencia existente en el script.


[i]ncluyeme
.com


UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES