JavaScript

Programa del curso

- Clase 1: Introducción y sintaxis
- Clase 2: Funciones, arrays y objetos
- Clase 3: Javascript integrado a HTML
- Clase 4: DOM, selectores y elementos
- **Clase 5: Eventos**
- Clase 6: Formularios
- Clase 7: Ajax
- Clase 8: Ejercicio integrador

1. Eventos

DOM - Eventos

Un evento es algo que pasa en el navegador o algo que hace el usuario.

Algunos ejemplos:

- La página terminó de cargar
- El input de un formulario cambió
- Clickearon un botón

Javascript me permite actuar cuando estos eventos pasan.

¿Se te ocurre que hace este botón?

```
<button onclick="alert(Date())">¿Qué hora es?</button>
```

DOM - Eventos

Existen dos formas de registrar un evento:

1) La primera es por medio de establecer una propiedad en el objeto o document

```
elemento.onNombreDelEvento = function () {}
```

Los más utilizados son:

- onclick
- onchange
- onmouseover
- onmouseout
- onkeydown
- onload

DOM - Eventos

2) La segunda es utilizando **addEventListener**.

```
objeto.addEventListener(tipoDeEvento, funcionQueLoManeja);
```

tipoDeEvento: es un string con el nombre del tipo de evento
funcionQueLoManeja: es una función que se invoca cuando suceda evento.

Un ejemplo: ¿Qué hará esto?

```
document.addEventListener("click", function(){
        alert("Ayy Me cliqueaste!");
});
```

Eventos - this

Podemos utilizar la palabra reservada **this** que en ese contexto hace referencia al objeto que ejecutó el evento.

```
function miFuncion() {
    console.log(this)  // this es el elemento que ejecutó el evento
}
elemento.addEventListener('click', miFuncion);
```

Eventos - removeEventListener

Para remover un *addEventListener* que hayamos ingresado se utiliza:

```
objeto.removeEventListener(tipoDeEvento, funcionQueManeja);
```

- tipoDeEvento: es un string con el nombre del tipo de evento
- funcionQueManeja: es una función que se invoca cuando suceda evento.

Onclick VS EventListener

La principal diferencia entre onclick y EventListener, es que addEventListener nos permite que un mismo elemento pueda tener muchos canales escuchándolo, es decir, muchos listeners.

El onclick encambio solamente tendrá un solo evento relacionado

Eventos - preventDefault()

Para prevenir la ejecución de un evento por defecto utilizamos:

```
<script>
document.getElementById("abc").addEventListener("click",
function(event){
    event.preventDefault(); // el link ya no irá a google
});
</script>
<body>
        <a id="abc" href="http://www.google.com.ar/">Google</a>
</body>
```

Eventos - Mouse

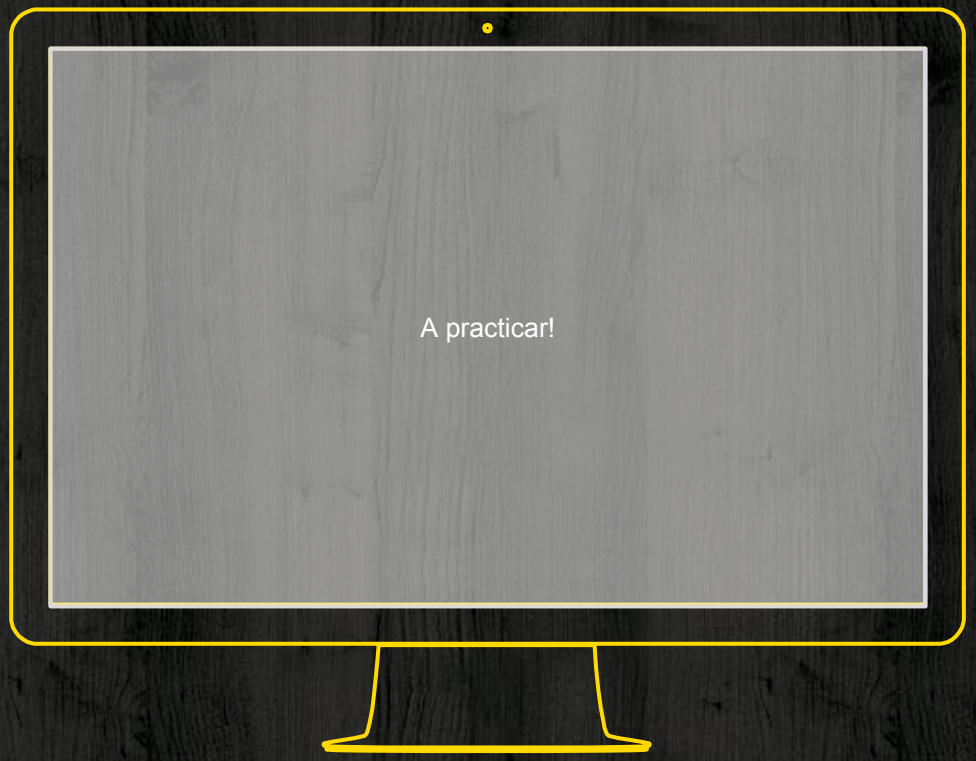
El objeto **event** asociado al mouse tiene atributos que nos permite saber la posición donde se encuentra con **clientX** y **clientY**.

```
elemento.addEventListener('click', function(event) {
    event.clientX;
    event.clientY;
});
```

Eventos - Teclado

También podemos controlar los eventos que se disparan cuando se presionan las teclas. Por medio del evento keypress, keydown y keyup.

```
elemento.addEventListener('keypress', function(event) {
        var x = event.keyCode;
        if (x == 27) { // 27 es el escape
                alert(“Presionaste el escape!!”);
        }
});
```



A practicar!

Práctica 5 - **Eventos**

2.
Timers

Timers - setTimeout

Javascript tiene funciones nativas que nos permiten retrasar la ejecución de un código que nosotros elegimos.

La función **setTimeout** se utiliza cuando queremos que nuestro código se ejecute una vez pasado un tiempo establecido.

```
setTimeout(función, retraso);
```

Muestra un alert luego de **3** segundos (3000 milisegundos):

```
setTimeout(function(){ alert("Hello"); }, 3000);
```

Timers - setInterval

Por medio de esta función podemos ejecutar varias veces el mismo código a un intervalo regular.

```
setInterval(función, retraso);
```

Cada 3 segundos salta la alerta:

```
setInterval(function(){ alert("Hello"); }, 3000);
```

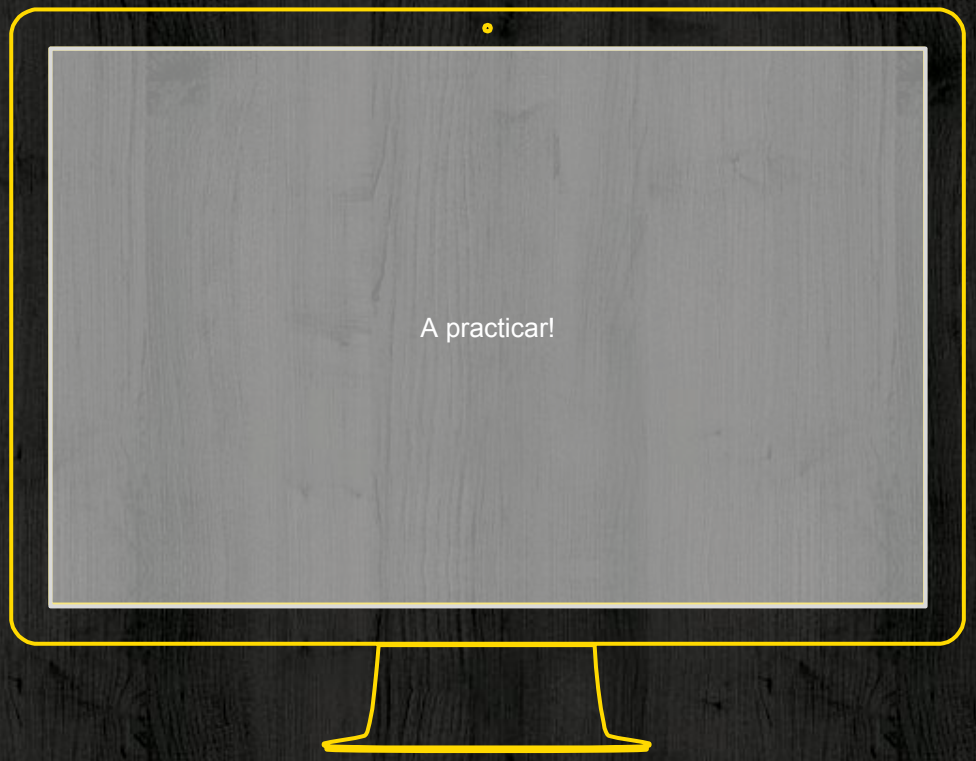
Timers - clearTimeout / clearInterval

Para detener un *timeout* utilizamos:

```
var myVar = setTimeout(function(){ alert("Hello"); }, 3000);
clearTimeout(myVar);
```

Para detener un *interval* utilizamos:

```
var myVar = setInterval(function(){ alert("Hello"); }, 3000);
clearInterval(myVar);
```



A practicar!

Práctica 5 - **Timers**

Gracias!

Preguntas?