5.A. Gestión de eventos y formularios en JavaScript

3.- Eventos.

3.2.- Modelo de registro de eventos tradicional.

En los navegadores antiguos, el modelo que se utilizaba era el modelo en línea. Con la llegada de DHTML, el modelo se extendió para ser más flexible. En este nuevo modelo el evento pasa a ser una propiedad del elemento, así que por ejemplo los navegadores modernos ya aceptan el siguiente código de JavaScript:

```
elemento.onclick = hacerAlgo;  // cuando el usuario haga clic en el objeto, se llamará a la función hacerAlgo()
```

Esta forma de registro, no fue estandarizada por el W3C, pero debido a que fue ampliamente utilizada por Netscape y Microsoft, todavía es válida hoy en día. La ventaja de este modelo es que podremos asignar un evento a un objeto desde JavaScript, con lo que ya estamos separando el código de la estructura. Fíjate que aquí los nombres de los eventos si que van siempre en minúsculas.


```
elemento.onclick = hacerAlgo;
```

Para eliminar un gestor de eventos de un elemento u objeto, le asignaremos *null*:

```
elemento.onclick = null;
```

Otra gran ventaja es que, como el gestor de eventos es una función, podremos realizar una llamada directa a ese gestor, con lo que estamos disparando el evento de forma manual. Por ejemplo:

```
elemento.onclick();

// Al hacer ésto estamos disparando el evento clic de forma manual y se ejecutará la función hacerAlgo()
```

Sin paréntesis

Fíjate que en el registro del evento no usamos paréntesis (). El método *onclick* espera que se le asigne una función completa. Si haces: `element.onclick = hacerAlgo();` la función será ejecutada y el resultado que devuelve esa función será asignado a *onclick*. Pero ésto no es lo que queremos que haga, queremos que se ejecute la función cuando se dispare el evento.

Uso de la palabra reservada this

En el modelo en línea podemos utilizar la palabra reservada *this* cuando programamos el gestor de eventos. Por ejemplo:

```
<a href="pagina.html" ID="mienlace" onClick="alertar(this)">Pulsa aqui</a>

<script type="text/javascript">

function alertar(objeto) {

    alert("Te conectaremos con la página: " + objeto.href);

}

</script>


Su equivalente en el <strong>modelo tradicional</strong> sería:

<a id="mienlace" href="pagina.html">Pulsa aqui</a>


<script type="text/javascript">
```

```
document.getElementById("mienlace").onclick = alertar;


function alertar() {

     alert("Te conectaremos con la página: " + this.href);

}

</script>
```

Fíjate que estamos usando `this` dentro de la función, sin pasar ningún objeto (tal y como hacíamos en el modelo en linea). En el modelo tradicional el *this* que está dentro de la función, hace referencia al objeto donde hemos programado el evento. También hay que destacar que en este modelo es importante que el hiperenlace sea declarado antes de programar la asignación `onclick`, ya que si por ejemplo escribimos el hiperenlace después del bloque de código de JavaScript, éste no conocerá todavía el objeto y no le podrá asignar el evento de `onclick`. Ésto también se podría solucionar, programando la asignación de eventos a los objetos, en una función que se ejecute cuando el documento esté completamente cargado. Por ejemplo con `window.onload=asignarEventos`.

Si por ejemplo queremos que cuando se produzca el evento se realicen llamadas a más de una función lo podremos hacer de la siguiente forma:

```
elemento.onclick = function ( ) {

    llamada1( );

    llamada2( );

};
```

Créditos de la imagen
Autoría: tilyckr.

◀ Solución a la tarea para DWEC05

Ir a...

5.B. Eventos JavaScript en W3Schools ▶