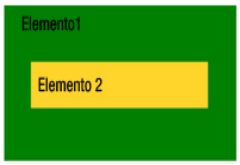
5.A. Gestión de eventos y formularios en JavaScript

3.- Eventos.

3.5.- Orden de disparo de los eventos.



Imagina que tenemos un elemento contenido dentro de otro elemento, y que tenemos programado el mismo tipo de evento para los dos (por ejemplo el evento clic). ¿Cuál de ellos se disparará primero? Sorprendentemente, ésto va a depender del tipo de navegador que tengamos.

El problema es muy simple. Imagina que tenemos el siguiente gráfico:

y ambos tienen programado el evento de clic. Si el usuario hace clic en el elemento2, provocará un clic en ambos: elemento1 y elemento2. ¿Pero cuál de ellos se disparará primero?, ¿cuál es el orden de los eventos?

Tenemos **dos Modelos propuestos** por Netscape y Microsoft en sus orígenes:

- Netscape dijo que, el evento en el elemento1 tendrá lugar primero. Es lo que se conoce como **"captura de eventos"**.
- Microsoft mantuvo que, el evento en el elemento2 tendrá precedencia. Es lo que se conoce como "**burbujeo de eventos**".

Los dos modelos son claramente opuestos. Internet Explorer sólo soporta burbujeo (bubbling). Mozilla, Opera 7 y Konqueror soportan los dos modelos. Y las versiones antiguas de Opera e iCab no soportan ninguno.

Modelo W3C

W3C decidió tomar una posición intermedia. Así supone que, cuando se produce un evento en su modelo de eventos, primero se producirá la fase de captura hasta llegar al elemento de destino, y luego se producirá la fase de burbujeo hacia arriba. Este modelo es el estándar, que todos los navegadores deberían seguir para ser compatibles entre sí.

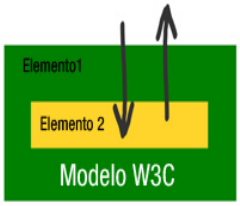
Tú podrás decidir cuando quieres que se registre el evento: en la fase de captura o en la fase de burbujeo. El tercer parámetro de `addEventListener` te permitirá indicar si lo haces en la **fase de captura** (`true`), o en la **fase de burbujeo**(`false`).

Por ejemplo:

```
elemento1.addEventListener('click',hacerAlgo1,true);

elemento2.addEventListener('click',hacerAlgo2,false);
```

Si el usuario hace clic en el elemento2 ocurrirá lo siguiente:



1. El evento de clic comenzará en la fase de captura. El evento comprueba si hay algún ancestro del elemento2 que tenga un evento de `onclick` para la fase de captura (`true`).
2. El evento encuentra un `elemento1.hacerAlgo1()` que ejecutará primero, pues está programado a `true`.
3. El evento viajará hacia el destino, pero no encontrará más eventos para la fase de captura. Entonces el evento pasa a la fase de burbujeo, y ejecuta `hacerAlgo2()`, el cuál hemos registrado para la fase de burbujeo (`false`).
4. El evento viaja hacia arriba de nuevo y chequea si algún ancestro tiene programado un evento para la fase de burbujeo. Éste no será el caso, por lo que no hará nada más.

Para **detener la propagación del evento** en la fase de burbujeo, disponemos del método `stopPropagation()`. En la fase de captura es imposible detener la propagación.

◀ Solución a la tarea para DWEC05

Ir a...

5.B. Eventos JavaScript en W3Schools ▶