Fundamentos de Programación

JavaScript

Eventos

IES La Nía

Eventos

- Los eventos responden a acciones que suceden durante la navegación.
- El navegador lanza los eventos
  - Al cargar la página
  - Al pasar el ratón por encima de una imagen
  - Al terminar de cargar un vídeo
  - Al enviar un formulario
- Mediante *JavaScript* asociamos un manejador (una función) a un evento

Registrando mediante atributo

- Todas las etiquetas aceptan atributos **`on*`** con el nombre del evento
- El valor del atributo será una función que se invocará cuando se lance el evento
- Método no recomendable al acoplar *HTML* y *JavaScript*

```
<img onclick='cambiarBorde()' src='logo.jpg' />
```

Ejercicios

- A partir del contenido de `Eventos01`
- Modificar el archivo `index.html`
- (`ej701onclickatributo`) Añadir mediante un atributo a cada imagen una alerta para que al hacer *click* muestre su color

Registrado mediante notación punto

- Se realiza mediante *JavaScript*
- A partir de un nodo del DOM
  - Acceder a la propiedad del evento
  - Asignar a la propiedad una función. Esta función se conoce como **manejador** del evento

```
var nodo = document.getElementById('logo');
nodo.onclick = function() {
    // código del manejador
}
```

Ejercicios

- A partir del contenido de `Eventos01`
- Modificar el archivo `index.html`
- (`ej702onclickpunto`) Añade:
  - un atributo `id` a cada imagen
  - Mediante la notación punto, asigna un manejador a cada imagen para que al hacer *click* muestre una alerta con el color pulsado

Registrando mediante `addEventListener()`

- `addEventListener(eventos, funcionManejador, boolBubbling)`
- A partir de un nodo DOM
- Permite asignar más de un manejador a un mismo evento
- Permite asignar varios eventos al mismo tiempo

```
nodo.addEventListener('click', function() {
    // código del manejador
}, false);
```

Ejercicios

- A partir del contenido de `Eventos01`
- Modificar el archivo `index.html`
- (`ej703onclickpunto`) Añade:
  - un atributo `id` a cada imagen
  - Mediante `addEventListener`, asigna un manejador a cada imagen para que al hacer *click* muestre una alerta con el color pulsado

Objeto `Event`

- Al capturar un evento, se recibe como parámetro un objeto `Event`
- Contiene información sobre el entorno y el navegador

```
nodo.addEventListener('click', function(evt) {
    // evt es el objeto Event
}, false);
```

Propiedades del objeto `Event`

- **`type`**: tipo del evento (`click`, `mouseover`)
- `timestamp`: cuando sucedió el evento
- `defaultPrevented`: *booleano* sobre si se realiza el comportamiento predeterminado
- `currentTarget`: elemento al que se asignó el evento
- **`target`**: elemento que lanza el evento
- `fromElement / toElement`: se emplea en eventos que suceden entre elementos (mouseover, etc...)

Ejercicios

- A partir del contenido de `Eventos01`
- Sobre el archivo `index.html` (`ej704objetoEvent`):
  - Añade un evento a
    - una imagen
    - el elemento de lista que contiene la imagen
    - a la lista desordenada
  - En cada manejador, muestra las propiedades
    - `timestamp`
    - `currentTarget`
    - `target`

Propagación de Eventos

- Un elemento puede capturar eventos de sus hijos
- Al capturar el evento, mediante la propiedad `target` accedemos al elemento que lanzó el evento.
- Una vez accedido al elemento, mediante `nodeName` (o `tagName`) comprobaremos si es el nodo que esperamos.

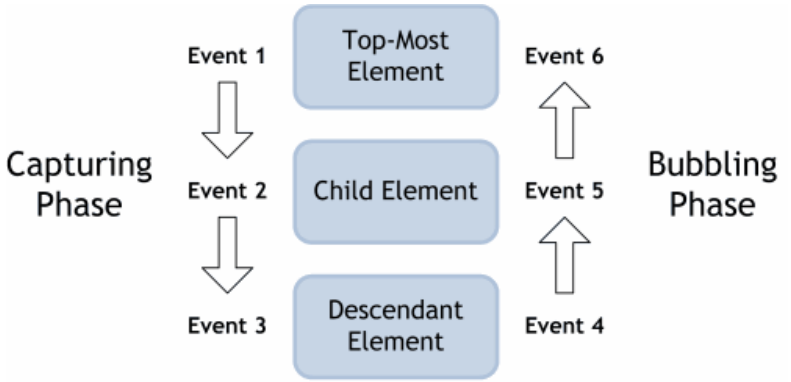
```
nodo.addEventListener('click', function(evt) {
    var etiq = evt.target;
    if (etiq.tagName == 'IMG') {...
}, false);
```

Ejercicios

- A partir del contenido de `Eventos01`
- Sobre el archivo `index.html` (`ej705propagacion`):
  - Añade un evento a
    - una imagen
    - a la lista desordenada
  - Dentro de la imagen, a partir del atributo `alt`, visualizar una alerta con el color de la imagen.

Tipos de Propagación

- **Captura de Eventos**: De arriba a abajo, y vuelve hacia arriba.
- **Burbujeo**: De abajo hacia arriba




```
<div>  <!-- Top-Most Element -->
  <div>  <!-- Child Element -->
    <div>Content</div>  <!-- Descendant Element -->
  </div>
</div>
```

- Tercer parámetro de `addEventListener`
  - `true`: captura de evento
  - `false`: burbujeo

Cancelando la Propagación

- Si queremos que un evento no se propague → `evt.stopPropagation()`
- Evita que se lancen otros manejadores.
- Para evitar que un evento lo capturen dos manejadores diferentes hay que cancelar la propagación en uno de ellos.

Ejercicios

- A partir del ejercicio `705`
- Sobre el archivo `index.html` (`ej706cancelarEvento`):
  - Modifica el código para que ambos manejadores se propaguen mediante burbujeo.
  - Al pulsar sobre la imagen que contiene un manejador, sólo se ha de lanzar un evento, no dos.

Cancelando el comportamiento predeterminado

- Los eventos pueden tener consecuencias
  - Al hacer click sobre un enlace
  - Al enviar un formulario
- Para evitar su comportamiento predeterminado → `evt.preventDefault()`

Ejercicios

- A partir del ejercicio `706`
- Sobre el archivo `index.html` (`ej707cancelarEvento`):
  - Rodea cada imagen mediante un enlace que apunte a www.ieslania.es
  - Modifica cada manejador para que al pulsar sobre la imagen NO vaya al destino del enlace.

Ejercicios

- A partir del contenido de `Eventos01` / `Eventos02`
- Sobre el archivo `script.js` (`ej708borrando`):
  - Añade un evento a la lista de modo que al pulsar sobre una imagen, ésta se elimine
    - TIP → con el atributo `target` podemos acceder al nodo que lanza el evento
    - TIP → para borrar un nodo → `removeChild`
  - Hay que comprobar que no suceda nada al pulsar entre dos imágenes

Tipos de Eventos

https://developer.mozilla.org/en-US/docs/Web/Events

- Carga
  - `load`
- Foco
  - `focus, blur`
- Ratón
  - `click, dblclick, mousedown, mouseup, clientX, clientY`
  - `mousemove, mouseover, mouseout`
- Teclado
  - `keydown, keypress, keyup`
- Formulario
  - `submit`

Evento de Carga

- **`window.onload`** → asegura que el documento ha cargado

```
function preparandoManejadores() {
  var miLogo =
document.getElementById("logo");
  miLogo.onclick() {
    alert("Has venido al sitio adecuado.");
  }
}

window.onload = function() {
  preparandoManejadores();
}
```

Ejercicios

- A partir del contenido de `Eventos02`
- Sobre el archivo `script.js` (`ej709lupa`):
  - Añade los manejadores mediante un evento de carga
  - Añade un evento a la lista de modo que al pasar por encima de una imagen, en la capa cuya clase es `preview`, se muestre la imagen en grande:
    - pequeña: acaba en `_tn`
    - grande: quitándole `_tn` a la pequeña

Ejercicios

- A partir del contenido de `Eventos03`
- Sobre el archivo `script.js` (`ej710fullscreen`):
  - Añade los manejadores mediante un evento de carga
  - Añade un evento a la imagen para que al hacer click, cargue la imagen en alta resolución en la capa `overlay`.
  - Al cargar la imagen, la capa `overlay` pondrá su atributo `display` a `block`
  - La imagen tendrá la clase CSS `bgImg`

Ejercicios

- A partir del ejercicio anterior
- Sobre el archivo `script.js` (`ej711spinner`):
  - Antes de cargar la imagen grande, cargar la imagen `spinner.gif` con clase CSS `spinner`
  - Añadir un evento `load` a la imagen grande, para que al dispararse elimine el *spinner.*