CFGS DAW

Mòdul 6: Desenvolupament web en entorn client

INS Joan d'Àustria

Eventos

En JavaScript, la interacción con el usuario se consigue mediante la captura de los eventos que éste produce. Un evento es una acción del usuario ante la cual el sistema debe responder de alguna manera. Un usuario hace clic en un botón y el sistema responde, pasa el ratón por encima de una imagen, realiza pulsaciones de teclado, etc.

Eventos

Existen 3 formas de asociar eventos a los diferentes elementos HTML.

a) Directamente como un atributo del elemento
b) Manejadores de eventos (Event Handlers). Se añaden en el DOM level 0.
c) Event Listeners. Se añaden en el DOM Level 2.

Eventos

a. AÑADIR EVENTOS DIRECTAMENTE

Es el método menos recomendado.

Consiste en añadir un atributo al elemento llamado como el evento y que toma como valor la función o código javascript a ejecutar.

```
<input type=button id="btn" value="prueba" onclick="unafuncion()">
```

Eventos

a. AÑADIR EVENTOS DIRECTAMENTE

Cuando se genera un evento de esta manera, automáticamente:

- se genera una función que dispone de un argumento llamado event (lo veremos más adelante)

```
<input type=button id="btn" value="prueba" onclick="alert(event.type)">
```

- La palabra this nos da acceso al objeto event, por lo que no es necesario pasarlo como parámetro a la función

```
<input type=button id="btn" value="prueba" onclick="alert(this.value)">
```

Eventos

a. AÑADIR EVENTOS DIRECTAMENTE

- Internamente, la función puede acceder tanto al documento como al elemento, pues se genera un mecanismo similar a:

```
function(){
        with(document){
                with(this){
                        //código
                }
        }
}
```

```
<input type=button id="btn" value="prueba" onclick="alert(value)">
```

Eventos

a. AÑADIR EVENTOS DIRECTAMENTE

- Si además el elemento está dentro de un formulario, este también se incluye :

```
function(){
        with(document){
                with(this.form){
                        with(this){
                                //código
                        }
                }
        }
}
```

```
<form method=”post”>
<input type=”text” name=”username” value=””>
<input type=”button” value=”prueba” onclick=”alert(username.value)”>
</form>
```

Eventos

b. EVENT HANDLERS

Es el método más recomendado.

Consiste en añadir la funcionalidad al elemento accediendo al DOM.

Sólo podemos añadir una función determinada a un evento de un elemento. Aunque no es habitual, en algunos momentos podremos necesitar 2 o más funciones asociadas a un evento, como por ejemplo validar un formulario y realizar una petición Ajax.

```
var btn = document.getElementById("btn");
btn.onclick = function(){
        alert('Hola mundo!');
};    //NO HAY QUE OLVIDAR ESTE PUNTO Y COMA
```

Eventos

b. EVENT HANDLERS

También podemos acceder al propio elemento usando la palabra this.

```
var btn = document.getElementById("btn");
btn.onclick = function(){
        alert(this.value);
};
```

Si queremos eliminar el event handler podemos asignarle el valor null

```
btn.onclick = null;
```

Eventos

c. EVENT LISTENERS

Permiten asociar más de una función a un evento.

IE8 y anteriores no soportan esta forma de asociación.

La sintaxis es:

```
element.addEventListenter("event", function(){
        // código javascript
}, false);
```

```
var btn = document.getElementById("btn");
btn.addEventListener("click", function(){
        alert('Hola mundo!');
}, false);
```

Eventos

c. EVENT LISTENERS

Se pueden eliminar mediante el método removeEventListener con los mismos argumentos que el addEventListener. Las funciones anónimas no se podrán eliminar.

```
function hola(){
        alert('Hola mundo');
}

var btn = document.getElementById("myBtn");
btn.addEventListener("click", hola, false);
// ...
 btn.removeEventListener("click", hola, false);
```

```
var btn = document.getElementById("myBtn");
btn.addEventListener("click", function(){
alert('Hola mundo');
}, false);
// ...Con funciones anónimas no lo quitará
btn.removeEventListener("click", function(){
        alert('Hola mundo');
}, false);
```

Eventos

EJERCICIO

Crear una página web con 3 párrafos y 3 enlaces (o botones). Estos tendrán un id tipo: contenido_1, enlace_1, contenido_2, enlace_2, etc. El texto de los enlaces será “ocultar contenido”

Utilizando una función común para los 3 enlaces, hacer que cuando se haga click sobre cada uno de ellos, se oculte el párrafo asociado y el enlace pase a tener como texto “mostrar contenido”. La siguiente vez que se haga click, se mostrará de nuevo el párrafo correspondiente.

Pista: Seguramente tendreis que utilizar this para saber el enlace que ha provocado el evento, split o otras funciones de string, innerHTML, y style para mostrar o ocultar.

Eventos

TIPOS DE EVENTOS

Hay muchos eventos que pueden ocurrir en un navegador. En DOM Level 3 se definen entre otras las siguientes categorías:

- De interface de usuario: eventos generales normalmente de interacción con el BOM
- De foco: Cuando un elemento recibe o pierde el foco
- De ratón: Cuando se realiza una acción con el ratón
- De teclado: Cuando se usa el teclado

Eventos

a) EVENTOS DE INTERFACE DE USUARIO

En esta categoría destacan:

- load: este evento ocurre en la página (window), en una imagen (img) o en un frameset cuando se cargan completamente
- select: ocurre cuando el usuario selecciona uno o más caracteres en un textbox
- resize: ocurre en la ventana (window) o frame cuando se cambia su tamaño

```
<body onload=”alert(‘Página cargada!!’)”>
```

Eventos

b) EVENTOS DE FOCO

En esta categoría destacan:

- blur: se produce cuando un elemento pierde el foco
- focus: se produce cuando un elemento recibe el foco

Eventos

c) EVENTOS DE RATÓN

En esta categoría destacan:

- click: se produce cuando el usuario presiona el botón primario del ratón o cuando se presiona la tecla Enter
- dblclick: se produce cuando se hace doble click en un elemento
- mousedown: se produce cuando el usuario presiona cualquiera de los botones del ratón

Eventos

c) EVENTOS DE RATÓN

- mouseenter: se produce cuando el usuario mueve el ratón y lo posiciona de fuera a dentro de un elemento
- mouseleave: se produce cuando el ratón sale de un elemento
- mousemove: se produce repetidamente cuando el ratón se mueve sobre un elemento

Eventos

d) EVENTOS DE TECLADO

- keydown: se produce cuando se presiona (sin soltar) una tecla
- keypress: se produce cuando se presiona una tecla
- keyup: se produce cuando se deja de presionar una tecla