DESARROLLO WEB FULL STACK

Eventos en JavaScript

Introducción a los eventos

DWFS COR

Introducción

En la programación **tradicional**, las aplicaciones se ejecutan **secuencialmente** de principio a fin para producir sus resultados. Sin embargo, en la actualidad el **modelo predominante** es el de la **programación basada en eventos**. Los scripts y programas esperan sin realizar ninguna tarea hasta que se produzca un evento. Una vez producido, ejecutan alguna tarea asociada a la aparición de ese evento y cuando concluye, el script o programa vuelve al estado de espera.

JavaScript permite realizar scripts con ambos métodos de programación: secuencial y basada en eventos. Los eventos de JavaScript permiten la interacción entre las aplicaciones JavaScript y los usuarios. Cada vez que se pulsa un botón, se produce un evento. Cada vez que se pulsa una tecla, también se produce un evento. No obstante, para que se produzca un evento no es obligatorio que intervenga el usuario, ya que por ejemplo, cada vez que se carga una página, también se produce un evento.

Introducción

En la programación **tradicional**, las aplicaciones se ejecutan **secuencialmente** de principio a fin para producir sus resultados. Sin embargo, en la actualidad el **modelo predominante** es el de la **programación basada en eventos**. Los scripts y programas esperan sin realizar ninguna tarea hasta que se produzca un evento. Una vez producido, ejecutan alguna tarea asociada a la aparición de ese evento y cuando concluye, el script o programa vuelve al estado de espera.

JavaScript permite realizar scripts con ambos métodos de programación: secuencial y basada en eventos. Los eventos de JavaScript permiten la interacción entre las aplicaciones JavaScript y los usuarios. Cada vez que se pulsa un botón, se produce un evento. Cada vez que se pulsa una tecla, también. No obstante, para que se produzca un evento no es obligatorio que intervenga el usuario, ya que por ejemplo, cada vez que se carga una página, también se produce un evento.

Tipos de eventos

Cada elemento HTML tiene definida su propia lista de posibles eventos que se le pueden asignar.

El nombre de los eventos se construye mediante el prefijo **on**, seguido del nombre en inglés de la acción asociada al evento. Así, el evento de clickear un elemento con el mouse se denomina **onclick** y el evento asociado a la acción de mover el mouse se denomina **onmousemove**.

Tipos de eventos

Evento	Descripción	Elementos para los que está definido
`onblur`	Un elemento pierde el foco	`<button>`, `<input>`, `<label>`, `<select>`, `<textarea>`, `<body>`
`onchange`	Un elemento ha sido modificado	`<input>`, `<select>`, `<textarea>`
`onclick`	Pulsar y soltar el ratón	Todos los elementos

Tipos de eventos

Evento	Descripción	Elementos para los que está definido
`ondblclick`	Pulsar dos veces seguidas con el ratón	Todos los elementos
`onchange`	Un elemento ha sido modificado	`<input>`, `<select>`, `<textarea>`
`onfocus`	Un elemento obtiene el foco	`<button>`, `<input>`, `<label>`, `<select>`, `<textarea>`, `<body>`

Tipos de eventos

Evento	Descripción	Elementos para los que está definido
`onkeydown`	Pulsar una tecla y no soltarla	Elementos de formulario y `<body>`
`onkeypress`	Pulsar una tecla	Elementos de formulario y `<body>`
`onkeyup`	Soltar una tecla pulsada	Elementos de formulario y `<body>`
`onload`	Página cargada	`<body>`

Tipos de eventos

Evento	Descripción	Elementos para los que está definido
`onmousedown`	Pulsar un botón del ratón y no soltarlo	Todos los elementos
`onmousemove`	Mover el ratón	Todos los elementos
`onmouseout`	El ratón "sale" del elemento	Todos los elementos

Tipos de eventos

Evento	Descripción	Elementos para los que está definido
`onmouseover`	El ratón "entra" en el elemento	Todos los elementos
`onmouseup`	Soltar el botón del ratón	Todos los elementos
`onreset`	Inicializar el formulario	`<form>`

Tipos de eventos

Evento	Descripción	Elementos para los que está definido
`onresize`	Modificar el tamaño de la ventana	`<body>`
`onselect`	Seleccionar un texto	`<input>`, `<textarea>`
`onsubmit`	Enviar el formulario	`<form>`

Manejadores de eventos

Un evento de JavaScript por sí mismo carece de utilidad. Para que los eventos resulten útiles, se deben asociar **funciones** o **código** JavaScript a cada evento. De esta forma, cuando se produce un evento se ejecuta el código indicado.

Las funciones o código JavaScript que se definen para cada evento se denominan **manejador de eventos (event handlers en inglés)** y como JavaScript es un lenguaje muy flexible, existen varias formas diferentes de indicar los manejadores:

- Manejadores como atributos de los elementos HTML.
- Manejadores como funciones JavaScript externas.
- Manejadores "semánticos".

Manejadores como atributos HTML

Se trata del método más sencillo y a la vez menos indicado de manejar un evento.

```
<input type="button" value="¡Click acá!"
onclick="console.log('¡Gracias! :)');" />


<div onclick="console.log('Hiciste click');"
onmouseover="console.log('Acabas de pasar el
mouse por encima');">
 Podés hacer click o pasar el mouse por encima
</div>


<body onload="console.log('La página se cargó
completamente');"></body>
```

Variable “this”

JavaScript define una variable especial llamada **this** que se crea automáticamente.

En los eventos, se puede utilizar la variable **this** para referirse al elemento HTML que ha provocado el evento.

```
<div id="contenidos" style="width:150px;
height:60px; border:thin solid silver"
onmouseover="document.getElementById('contenido
s').style.borderColor='black';">
    Sección de contenidos
</div>


// Con “this” podemos llevarlo a


<div id="contenidos" style="width:150px;
height:60px; border:thin solid silver"
onmouseover="this.style.borderColor='black';">
    Sección de contenidos
</div>
```

Manejadores de eventos como funciones

Cuando el código de la función manejadora es más complejo, es aconsejable agrupar todo el código JavaScript en una **función** externa que se invoca desde el código HTML cuando se produce el evento.

```
<input type="button" value="¡Click acá!"
onclick="console.log('¡Gracias! :)');" />

// Se puede transformar en

<input type="button" value="¡Click acá!"
onclick="darGracias();" />


// Código JS

function darGracias() {
    console.log('¡Gracias! :)');
}
```

Variable “this”

A las funciones podemos pasar la variable this.

```
<div id="contenidos" style="width:150px;
height:60px; border:thin solid silver"
onmouseover="resalta(this);">
    Sección de contenidos
</div>

// Código JS

function resalta(elemento) {
    elemento.style.borderColor= 'black';
}
```

Práctica

A una imágen agregar eventos para que cuando pongo el mouse arriba se cambie por otra, pero cuando lo saco vuelve a la que estaba.

A partir de la página web proporcionada, completar el código JavaScript para que:

1. Cuando se pinche sobre el primer botón, se oculte su sección relacionada.
2. Cuando se vuelva a pinchar sobre el mismo botón, se muestre otra vez esa sección de contenidos.
3. Completar el resto de enlaces de la página para que su comportamiento sea idéntico al del primer botón.
4. Cuando una sección se oculte, debe cambiar el mensaje del botón asociado.