

6.A. Modelo de objetos del documento en JavaScript

2.- Gestión de eventos.

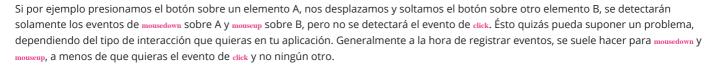
2.6.- Eventos del ratón en JavaScript.

Los eventos del ratón son uno de los eventos más importantes en JavaScript.

Cada vez que un usuario hace clic en un elemento, al menos se disparan tres eventos y en el siguiente orden:

- 1. mousedown, cuando el usuario presiona el botón del ratón sobre el elemento.
- 2. mouseup, cuando el usuario suelta el botón del ratón.
- 3. click, cuando el usuario pulsa y suelta el botón sobre el elemento.

En general, los eventos de mousedown y mouseup son mucho más útiles que el evento clic.



El evento de dibiclick no se usa muy a menudo. Incluso si lo usas, tienes que ser muy prudente y no registrar a la vez click y dibiclick sobre el mismo elemento, para evitar complicaciones.

El evento de mousemove funciona bastante bien, aunque tienes que tener en cuenta que la gestión de este evento le puede llevar cierto tiempo al sistema para su procesamiento. Por ejemplo si el ratón se mueve 1 pixel, y tienes programado el evento de mousemove, para cada movimiento que hagas, ese evento se disparará, independientemente de si el usuario realiza o no realiza ninguna otra opción. En ordenadores antiguos ésto puede ralentizar el sistema, ya que para cada movimiento del ratón estaría realizando las tareas adicionales programadas en la función. Por lo tanto se recomienda utilizar este evento sólo cuando haga falta, y desactivarlo cuando hayamos terminado.

Otros eventos adicionales del ratón son los de mouscover y mouscout, que se producen cuando el ratón entra en la zona del elemento o sale del elemento. Si, por ejemplo, tenemos tres contenedores anidados divA, divB y divC: si programamos un evento de mouscover sobre el divA y nos vamos moviendo hacia el contenedor interno, veremos que ese evento sigue disparándose cuando estemos sobre divB o entremos en divC. Ésta reacción se debe al burbujeo de eventos. Ni en divB o divC tenemos registrado el evento de mouscover, pero cuando se produce el burbujeo de dicho evento, se encontrará que tenemos registrado ese evento en el contenedor padre divA y por eso se ejecutará.

Muchas veces es necesario saber de dónde procede el ratón y hacia dónde va, y para ello W3C añadió la propiedad related Target a los eventos de mouseover y mouseout. Esta propiedad contiene el elemento desde dónde viene el ratón en el caso de mouseover, o el elemento en el que acaba de entrar en el caso de mouseout.

Para saber más

Propiedades de destino y origen del objeto Event.

Para saber los botones del ratón que hemos pulsado, disponemos de las propiedades which y button. Y para detectar correctamente el botón pulsado, lo mejor es hacerlo en los eventos de mousedown o mouseup. Which es una propiedad antigua de Netscape, así que simplemente vamos a citar button que es la propiedad propuesta por el W3C:

Los valores de la propiedad button pueden ser:

- Botón izquierdo: 0
- Botón medio: 1
- Botón derecho: 2

También es muy interesante conocer la posición en la que se encuentra el ratón, y para ello disponemos de un montón de propiedades que nos facilitan esa información:

- clientX, clientY: devuelven las coordenadas del ratón relativas a la ventana.
- offsetX, offsetY: devuelven las coordenadas del ratón relativas al objeto destino del evento.
- pageX, pageY: devuelven las coordenadas del ratón relativas al documento. Estas coordenadas son las más utilizadas.
- screenX, screenY: devuelven las coordenadas del ratón relativas a la pantalla.



document.getElementById("coordenadas").value=elEvento.clientX+" : "+elEvento.clientY;

document.addEventListener('mousemove', mostrarCoordenadas, false);

</script>

}

Para saber más

Eventos de ratón

Créditos de la imagen

Autoría: Davichi.

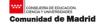
Licencia: CC BY 2.0.

◀ Solución a la tarea para DWEC06

Ir a...

6.B. HTML DOM en W3Schools ▶

EducaMadrid - Consejería de Educación, Ciencia y Universidades - <u>Ayuda</u>





\$