5.A. Gestión de eventos y formularios en JavaScript

3.- Eventos.

La mayoría de las veces que un usuario interactúa con un formulario está generando eventos. Por ejemplo, cuando hace clic con el ratón, cuando sitúa el cursor en un campo, cuando mueve el ratón sobre algún objeto, etc.

Usando JavaScript podremos programar que, cuando se produzca alguno de esos eventos se realice una tarea determinada. Es lo que se conoce en programación como capturar un evento.

Hay que tener en cuenta que en casi todas las páginas web que incorporan JavaScript, suele haber eventos programados que disparan la ejecución de dichos scripts. La razón es muy simple, JavaScript fue diseñado para añadir interactividad a las páginas: el usuario realiza algo y la página reacciona.

Por lo tanto, JavaScript necesita detectar de alguna forma las acciones del usuario para saber cuando reaccionar. También necesita saber las funciones, que queremos que ejecute cuando se produzcan esas acciones.

Cuando el usuario hace algo se produce un evento. También habrá algunos eventos que no están relacionados directamente con acciones de usuario: por ejemplo el evento de carga (`load`) de un documento, que se producirá automáticamente cuando un documento ha sido cargado.

Todo el tema de gestión de eventos se popularizó a raíz de la versión 2 de Netscape, que también soportaba eventos. Netscape 2 soportaba solamente algunos eventos. Los eventos `mouseOver` y `mouseOut`, se hicieron muy famosos a raíz de su utilización para hacer el efecto de sustitución de una imagen por otra al pasar el ratón por encima. Todo el resto de navegadores, incluido Explorer, tuvieron que adaptarse a la forma en la que Netscape 2 y 3 gestionaban los eventos.

Aunque hoy en día la técnica de gestión de eventos varía con el objetivo de independizar el código de JavaScript de la estructura HTML, los navegadores todavía son compatibles con las técnicas utilizadas por Netscape.

Anteriormente, a las versiones 4 de los navegadores, los eventos (interacciones del usuario y del sistema), eran capturados preferentemente por gestores de eventos definidos como atributos en las etiquetas HTML (modelo de eventos en línea). Por ejemplo, cuando un usuario hacía clic en un botón, el evento de `onclick` que se había programado en la etiqueta HTML era disparado. Ese evento hacía una llamada a una función en la que se realizarían las operaciones programadas por el usuario.

Aunque todo ese modo de gestión de eventos sigue funcionando, los navegadores modernos ya incorporan un modelo de eventos, que proporciona un montón de información sobre cómo ocurre un evento. Estas propiedades son accesibles a través de JavaScript, permitiendo programar respuestas más inteligentes a las interacciones del usuario con los objetos del documento.

Incompatibilidades entre navegadores

Muchas veces, lo que se hacía en un principio antes de programar cualquier evento, era detectar qué navegador estábamos utilizando, para saber si nuestro navegador soportaba, o no, los métodos y propiedades que queríamos usar. Por ejemplo:

```
if (Netscape) {

        utilizar modelo Netscape

}

else if (Explorer) {

        utilizar modelo Microsoft

}
```

Pero hoy en día ni ésto ni el modelo de detección basado en DHTML se recomiendan debido a las diferencias que hay entre todos los navegadores actuales. Por lo tanto hay que intentar usar modelos de detección de eventos estándar y que sean los navegadores los que tengan que adaptarse a ese modelo.