UNIDAD 6 - EVENTOS.

6.1 - DEFINICIÓN.

Podemos decir que los eventos son acciones que el usuario realiza mientras visita una página Web. Por ejemplo son eventos hacer clic en un botón, mover el puntero del ratón sobre un enlace, realizar una selección en una lista desplegable, etc. JavaScript para trabajar con eventos utiliza unos comandos denominados manejadores de eventos (event handler). Un manejador de eventos contiene instrucciones acerca de qué hacer cuando una clase particular de evento se dispara. El nombre del manejador de eventos es el nombre del evento precedido de la palabra on. Sin embargo en los eventos se pone de manifiesto uno de los principales problemas que implica desarrollar aplicaciones Web: las incompatibilidades entre navegadores. A pesar de que existen estándares para cada una de las tecnologías empleadas, los navegadores no los implementan intégra ni respetuosamente, haciendo que cada desarrollo se ejecute con diferencias sustanciales que hacen que algunas aplicaciones deban ser consideras incompatibles. Este es un problema muy grave de las tecnologías Web en general y además son un contrasentido ya que uno de los objetivos de las mismas era hacer que las aplicaciones fuesen independientes de la plataforma.

En el caso concreto de JavaScript y de los eventos, existen tres modelos diferentes para manejarlos dependiendo del navegador en el que se ejecute la aplicación. Por un lado el modelo básico de eventos que se introduce con la versión 4 del estándar HTML y que es considerado parte del nivel más básico de DOM. Es el único compatible con todos los navegadores pero es muy limitado. En segundo lugar el modelo de eventos estándar asociado al estándar DOM de nivel 2 que es implementado por la mayor parte de los navegadores con excepción de Internet Explorer. Y finalmente el modelo de eventos de Internet Explorer que como señala su nombre es específico de este

navegador. En el modelo básico de eventos, cada elemento tiene una lista de eventos que es capaz de manejar. El nombre de cada evento se construye con el prefijo "on" y el nombre de la acción que desencadena el evento. En siguiente tabla extraída de www.librosweb.es se muestran junto al nombre del manejador de eventos, la descripción y los elementos de interfaz de usuario más comunes en los documentos HTML.

Evento	Descripción	Elementos para los que está definido
onblur	Deseleccionar el elemento	<button>, <input>, <label>, <select>, <textarea>, <body>
onchange	Deseleccionar un elemento que se ha modificado	<input>, <select>, <textarea>
onclick	Pinchar y soltar el ratón	Todos los elementos
ondblclick	Pinchar dos veces seguidas con el ratón	Todos los elementos
onfocus	Seleccionar un elemento	<button>, <input>, <label>, <select>, <textarea>, <body>
onkeydown	Pulsar una tecla (sin soltar)	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	La página se ha cargado completamente	<body>
onmousedown	Pulsar (sin soltar) un botón del ratón	Todos los elementos
onmousemove	Mover el ratón	Todos los elementos
onmouseout	El ratón *"sale"* del elemento (pasa por encima de otro elemento)	Todos los elementos
onmouseover	El ratón *"entra"* en el elemento (pasa por encima del elemento)	Todos los elementos
onmouseup	Soltar el botón que estaba pulsado en el ratón	Todos los elementos
onreset	Inicializar el formulario (borrar todos sus datos)	<form>
onresize	Se ha modificado el tamaño de la ventana del navegador	<body>
onselect	Seleccionar un texto	<input>, <textarea>
onsubmit	Enviar el formulario	<form>
onunload	Se abandona la página (por ejemplo al cerrar el navegador)	<body>

El papel del programador Javascript consiste en desarrollar el código asociado a los manejadores de eventos para que una vez que se detecte el evento se ponga en ejecución el código previsto. El manejador de evento es por lo tanto una función Javascript diseñada por el programador para ese propósito. Para asociar a cada evento el manejador correspondiente se pueden usar diferentes estrategias que fundamentalmente se resumen en establecer el manejador como atributos de los elementos XHTML, usar funciones JavaScript externas o emplear manejadores "semánticos". El primer caso es el método más sencillo pero el menos recomendado. Consiste en definir el código del evento como el valor de un atributo de la marca XHTML.

```
<input type="button" value="Haga click..." onclick="alert('Gracias');" />
```

Con objeto de que desde el código se pueda hacer referencia al propio elemento XHTML que lo contiene, Javascript define el objeto this. En el ejemplo siguiente se observa el funcionamiento de este objeto de manera que un evento puede hacer que una propiedad del objeto XHTML se modifique durante la navegación.

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

El segundo método consiste en desarrollar funciones que agrupen las instrucciones sin que estas tengan que aparecer embebidas en el código XHTML.

```
function Mensaje() {alert('Gracias'); }

    <input type="button" value="Haga click" onclick="Mensaje()" />
```

Dado que en la función no tiene sentido, contextualmente hablando, utilizar el objeto this, es posible pasar dicho objeto como parámetro a la función.

```
function resalta(elemento) {
switch(elemento.style.borderColor) {
        case 'silver':
                elemento.style.borderColor = 'black';
                break;
        case 'black':
                elemento.style.borderColor='silver'; break; }
}
```

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

El tercer método para añadir manejadores de eventos tiene como principal objetivo separar si cabe aún más el código XHTML del código JavaScript, para facilitar su lectura y su mantenimiento futuro, al igual que es conveniente separar el contenido XHTML del diseño y presentación CSS. Este método consiste en utilizar las propiedades DOM de los elementos XHTML para asignar todas las funciones externas que actúan de manejadores de eventos. Por ejemplo:

```
// Función externa
function muestraMensaje() { alert('Gracias'); }
// Asignar la función externa al elemento
document.getElementById("pinchable").onclick = muestraMensaje;
// Elemento XHTML
<input id="pinchable" type="button" value="Haz click " />
```

El único inconveniente de este método es que la página se debe cargar completamente antes de que se puedan utilizar las funciones DOM que asignan los manejadores a los elementos XHTML. Una de las formas más sencillas de asegurar que cierto código se va a ejecutar después de que la página se cargue por completo es utilizar el evento `onload`:

```
window.onload = function() { document.getElementById("pinchable").onclick = muestraMensaje; }
```

Para obtener información más precisa sobre las circunstancias en las que se produce un evento, JavaScript dispone de un objeto especial llamado `event` pero cuyo uso depende del navegador. Los navegadores tipo Internet Explorer obtienen el objeto event directamente mediante: ***var evento = window.event;*** En el resto de navegadores, el objeto ***event*** se obtiene a partir del argumento que el navegador crea automáticamente:

```
function manejadorEventos(elEvento) { var evento = elEvento; }
```

Si se quiere programar una aplicación que funcione correctamente en todos los navegadores, es necesario obtener el objeto `event` de forma correcta según cada navegador. El siguiente código muestra la forma correcta de obtener el objeto en cualquier navegador:

```
function manejadorEventos(elEvento) {
 var evento = elEvento || window.event;}
```

6.2 - EVENTOS DE TECLADO

Los eventos relacionados con el teclado son los más incompatibles entre diferentes navegadores. Cada intervención del usuario en el teclado desencadena tres eventos diferentes (`onkeyup`, `onkeypress` y `onkeydown`). El evento `onkeydown` se corresponde con el acto de pulsar una tecla y mantenerla sin soltarla; el evento `onkeypress` refleja la pulsación de la tecla y el evento `onkeyup` representa el instante en que se suelta la tecla que pulsada. En función del navegador y del evento se pueden obtener información concreta a través de las propiedades ***keyCode y charCode***

```
function manejador(elEvento)
{
var evento = elEvento || window.event;
var caracter = evento.charCode || evento.keyCode;
alert("El carácter pulsado es: " + String.fromCharCode(caracter));
}
document.onkeypress = manejador;
```

6.3 - EVENTOS DE RATÓN

La información que habitualmente se captura del ratón son las coordenadas de su ubicación y los eventos asociados a la pulsación de los controles del mismo. Los sistemas de coordenadas permiten conocer la posición del ratón respecto a la pantalla del ordenador, respecto a la ventana del navegador y respecto a la propia página HTML (interesante si el usuario ha hecho *scroll* sobre la página). Las coordenadas más sencillas son las que se refieren a la posición del puntero respecto de la ventana del navegador, que se obtienen mediante las propiedades `clientX`y `clientY`:

```
function muestraInformacion(elEvento)
 {
var evento = elEvento || window.event;
var coordenadaX = evento.clientX;
var coordenadaY = evento.clientY;
alert("Ratón en la posición: " + coordenadaX + ", " + coordenadaY);
}
document.onclick = muestraInformacion;
```

Las coordenadas de la posición del puntero del ratón respecto de la pantalla completa del ordenador del usuario se obtienen de la misma forma, mediante las propiedades `screenX` y `screenY`:

```
var coordenadaX = evento.screenX; var coordenadaY = evento.screenY;
```

Las coordenadas respecto a la página HTML (hay diferencias entre navegadores)

```
function muestraInformacion(elEvento) {
 var evento = elEvento || window.event;}
var ie = navigator.userAgent.toLowerCase().indexOf('msie')!=-1;
if(ie) {
coordenadaX = evento.clientX + document.body.scrollLeft;
coordenadaY = evento.clientY + document.body.scrollTop;
}
else {
coordenadaX = evento.pageX;
coordenadaY = evento.pageY;
}
 alert("Ratón en la posición: " + coordenadaX + ", " + coordenadaY +  " respecto de la página web");
document.onclick=muestraInformacion;
```

```
<html>
<head>
<title> EVENTOS DE FORMULARIO</title>
<SCRIPT LANGUAGE="JAVASCRIPT">
function Comprobar(F)
{
        if(F.NOMBRE.value=="" && F.CURSO.value=="" && F.NOTA.value=="")
        {
                alert("Datos no válidos");
                return false;
        }
        return true;
}
</SCRIPT>
</head>
<body>
<p><b><font face="Verdana" size="4">EVENTOS DE FORMULARIO</font></b></p>
<form name="FORMU" onsubmit="return Comprobar(this)" onreset="return
document.FORMU.NOMBRE.value==''">
<p>Nombre y Apellidos: <input type="text" name="NOMBRE"
size="20"></p>
<p>Curso: <input type="text" name="CURSO" size="5"></p>
<p>Nota:<input type="text" name="NOTA" size="4"></p>
<p><input type="submit" value="Enviar" name="B1">
<input type="reset" value="Restablecer" name="B2"></p>
</form>
</body>
</html>
```

Ejemplo 42 – Eventos de formulario

```
<html>
<head>
<title> OTROS EVENTOS </title>
</head>
<body bgcolor="#66FFFF"
onresize='alert("HAS CAMBIADO EL TAMAÑO INICIAL DE LA VENTANA")'>
<p align="center"><B>BIENVENIDOS A MI PAGINA WEB</B>
<p align="center">
<img border="0" src="noesta.gif"
onerror='alert("NO EXISTE LA IMAGEN noesta.gif")'></p>
</body>
</html>
```

Ejemplo 43 – Otros eventos

Este ejemplo muestra los eventos del ratón. Se ha definido un formulario con dos cajas de texto, dos Botones de radio, un área de texto, una lista y el botón restablecer. Se han definido dos enlaces. Se ha definido la función Comprobar_seleccion(), que comprobará el botón pulsado del grupo de botones de radio con el elemento seleccionado de la lista de ciclos. Si se elige Ciclo de Grado Medio, solo pueden ser válidos los dos primeros elementos de la lista, si se elige Ciclo de Grado Superior, son válidos los elementos 3, 4 y 5 de la lista. Al elegir un elemento de la lista se invocará a la función. También se invocará al hacer clic en los botones de radio.

```
<html>
<head>
<title> Eventos de ratón</title>
<script language="JavaScript">
function Comprobar_seleccion(){
 n=document.FORMU.LISTA.selectedIndex;
 if(document.FORMU.LISTA.options[n].value=="0") //no ha seleccionado nada
 {
  alert("Debes seleccionar un elemento de la lista");
  document.FORMU.LISTA.focus();
 }
 else {

   if(document.FORMU.BOTONES[0].checked)  //grado medio
    if(document.FORMU.LISTA.options[n].value>"2")
    {
      alert("Debes seleccionar un Ciclo de Grado Medio");
      document.FORMU.LISTA.selectedIndex=5;//para que se visualice el elemento 5 de la lista
      document.FORMU.LISTA.focus();//devolver foco a la lista
      return;
    }
   if(document.FORMU.BOTONES[1].checked)    //grado superior
    if(document.FORMU.LISTA.options[n].value<"3")
    {
      alert("Debes seleccionar un Ciclo de Grado Superior");
      document.FORMU.LISTA.selectedIndex=5; //para que se visualice el elemento 5 de la lista
      document.FORMU.LISTA.focus();//devolver foco a la lista
      return;
    }
  }
}
</script>
</head>
<body>
<h2 align="center"><font face="Verdana">EVENTOS DEL RATÓN</font></h2>
<center>
<table border="1" cellpadding="0" cellspacing="0" width="576" bgcolor="#CCFFFF">
<tr>
<td width="572">
<form name="FORMU"><p align="center">
Nombre:<input type="text" name="NOMBRE" size="20"
onblur="if(this.value=='') this.value='********'"> Edad:<input type="text" name="EDAD"
size="5"
onblur="if(this.value=='') this.value='********'"></p>
<center>
<table border="1" cellpadding="0" cellspacing="0" width="490" bgcolor="#FFFFCC">
```

```
<tr>
<td width="243">
<input type="radio" name="BOTONES" value="CGM"
onclick="Comprobar_seleccion()">Ciclo de Grado Medio
<p><input type="radio" value="CGS" name="BOTONES"
onclick="Comprobar_seleccion()">Ciclo de Grado Superior
</td>
<td width="228"> Lista de ciclos:
<select size="1" name="LISTA"
onchange="Comprobar_seleccion()">
<option value="1">Gestión Administrativa</option>
<option value="2">Comercio</option>
<option value="3">Administración y Finanzas</option>
<option value="4">Secretariado</option>
<option value="5">Comercio Internacional</option>
<option value="0" selected>Selecciona un Ciclo</option>
</select>
</td>
</tr>
</table><p><textarea rows="4" name="TEXTO" cols="23"
ondblclick="this.value=this.value.toUpperCase()" onselect="alert('Has seleccionado texto del
area')">
El texto que escribas se convertirá en mayúsculas al hacer dobleclick.
</textarea>
<input type="reset" value="Restablecer" name="B2"
onmousedown="alert('Has presionado el botón Restablecer')">
</p>
</center>
</form>
<p align="center"><a href="cfgm.htm"
onmouseover="window.status='Ver ciclos de Grado Medio'; return true"
onmouseout="window.status='';return true"
onmouseup="alert('Has levantado el botón del enlace')"> Ver CFGM </a>
<a href="cfgs.htm"
onmouseup="alert('Has levantado el botón del enlace')">Ver CFGS</a>
</p>
</td>
</tr>
</table>
</center>
</body>
</html>
```

Ejemplo 44 – Eventos de ratón

Vamos a utilizar en este ejemplo el evento de teclado onkeypress (se ha pulsado una tecla ANSI) para cambiar el color de fondo del documento.

Definimos varios enlaces, nos situaremos en ellos usando el tabulador en el color deseado y al presionar una tecla cambiará el color de fondo.

El código asociado a los enlaces es el siguiente:

```
<a href="" onkeypress='document.bgColor="#0000FF"'>Azul</a>
<a href="" onkeypress='document.bgColor="yellow"'>Amarillo</a>
<a href="" onkeypress='document.bgColor="red"'>Rojo</a>
```

En este ejemplo al cambiar con el tabulador de un enlace a otro los colores no cambian. Si utilizamos el evento *onkeydown* veremos que al pulsar una tecla cualquiera sobre el enlace cambiará de color (antes nos situamos con el tabulador). Si nos movemos solo con el tabulador para ir de un color a otro, cambiará al color del enlace cuando el tabulador lo abandone, es decir cuando se presione la tecla. Si utilizamos el evento *onkeyup* nada mas entrar en el enlace con el tabulador el fondo cambia de color.

El siguiente ejemplo muestra el uso de una imagen mapeada, al pasar sobre los textos cambia el color de fondo de pantalla. Al hacer clic en los textos que se visualizan en la imagen se carga el archivo HTML correspondiente. Utilizamos el evento *onload* para abrir una ventana de bienvenida con un documento HTML.

anuncio.html

```
<html>
<head>
<title>BIENVENIDO A MI PAG WEB</title>
</SCRIPT>
</head>
<body bgcolor="#66FFFF">
<p align="center">
<marquee bgcolor="#FFFF00" width="212" height="42" style="font-size: 24 pt; font-family:
Verdana">BIENVENIDOS A MI PAGINA WEB</marquee>
</body>
</html>
```

```
<HTML>
<HEAD>
<TITLE>EVENTOS</TITLE>
<script language="javascript">

function  anuncio()
{window.open("anuncio.html","","top=300,left=300,width=250,height=150");}
</script>

</HEAD>
<BODY onload="anuncio()" >
<center>
<MAP NAME="imagen">
<AREA SHAPE=RECT COORDS="18, 20, 367, 64"  NOHREF
onMouseOver="document.body.style.backgroundColor='#ff0'"
onMouseOut="document.body.style.backgroundColor='#fff'"
onclick="window.location.href='pag1.htm'">

<AREA SHAPE=RECT COORDS="17, 80, 347, 123"  NOHREF
onMouseOver="document.body.style.backgroundColor='#0f0'"
onMouseOut="document.body.style.backgroundColor='#fff'"
onclick="window.location.href='pag1.htm'">

<AREA SHAPE=RECT COORDS="16, 147, 241, 191"  NOHREF
onMouseOver="document.body.style.backgroundColor='#f0f'"
onMouseOut="document.body.style.backgroundColor='#fff'"
onclick="window.location.href='pag1.htm'">

<AREA SHAPE=RECT COORDS="12, 210, 217, 256"  NOHREF
onMouseOver="document.body.style.backgroundColor='#0ff'"
onMouseOut="document.body.style.backgroundColor='#fff'"
onclick="window.location.href='pag1.htm'">

</MAP>
<b><font face="Verdana">HAZ CLIC SOBRE EL TEXTO DE LA IMAGEN </font></b>
</center>
<p><center>
<IMG src="Imag1.jpg"  width="413" height="295" border="0" USEMAP="#imagen" ISMAP>
</center></p>
</BODY>
</HTML>
```

Ejemplo 45 – Eventos de imágenes