JavaScript	**MEDIA TÉCNICA DESARROLLO DE SOFTWARE** **GUIA JAVA SCRIPT No.1** **MODULO:** Elementos de software web **TEMA: eventos**	

DOCENTE: **Juan Carlos Pérez Pérez**
ESTUDIANTE : _____ FECHA:__/__/____
JUSTIFICACIÓN: Se pretende con esta guía hacer conozca y aplique manejo de java scripts y algunas funcionalidades en html al lado del cliente.

OBJETIVOS : Reconocer algunos eventos con javascript
Reconocer algunas utilidades de JavaScript

CONTENIDO:

Evento Onclick

El uso de JavaScript en los formularios HTML se hace fundamentalmente con el objetivo de validar los datos ingresados. Se hace esta actividad en el cliente (navegador) para desligar de esta actividad al servidor que recibirá los datos ingresados por el usuario.
Esta posibilidad de hacer pequeños programas que se ejecutan en el navegador, evitan intercambios innecesarios entre el cliente y el servidor (navegador y sitio web).

Codifique el siguiente ejemplo:

```
<script language="JavaScript">
var contador=0;
function incrementar()
{ contador++;
  alert('El contador ahora vale :' + contador);
}
</script>
<form> <input type="button"  onClick="incrementar()" value="incrementar">
</form></body></html>
```

Eventos onFocus y onBlur

El evento onFocus se dispara cuando el objeto toma foco y el evento onBlur cuando el objeto pierde el foco.
Ejemplo: Implementar un formulario que solicite la carga del nombre y la edad de una persona. Cuando el control tome foco borrar el contenido actual, al abandonar el mismo, mostrar un mensaje de alerta si el mismo está vacío.

```
<html><head></head>
<body>
<script language="JavaScript">
function vaciar(control){ control.value='';
}
function verificarEntrada(control)
{
  if (control.value=='')
    alert('Debe ingresar datos');
}</script>
<form name="form1">Ingrese su nombre:
<input type="text"  name="nombre"  onFocus="vaciar(this)"
onBlur="verificarEntrada(this)"><br>
```

```
Ingrese su edad:
<input type="text"  name="edad"  onFocus="vaciar(this)"
onBlur="verificarEntrada(this)"><br>
<input type="button"  value="Confirmar">
</form>
</body>
</html>
```

A cada control de tipo TEXT le inicializamos los eventos onFocus y onBlur. Le indicamos, para el evento onFocus la función vaciar, pasando como parámetro la palabra clave this que significa la dirección del objeto que emitió el evento. En la función propiamente dicha, accedemos a la propiedad value y borramos su contenido.

De forma similar, para el evento onBlur llamamos a la función verificarEntrada donde analizamos si se ha ingresado algún valor dentro del control, en caso de tener un string vacío procedemos a mostrar una ventana de alerta.

Eventos onMouseOver y onMouseOut

El evento onMouseOver se ejecuta cuando pasamos la flecha del mouse sobre un hipervínculo y el evento onMouseOut cuando la flecha abandona el mismo.

Para probar estos eventos implementaremos una página que cambie el color de fondo del documento.

Implementaremos una función que cambie el color con un valor que llegue como parámetro.

Cuando retiramos la flecha del mouse volvemos a pintar de blanco el fondo del documento:

```
<html><head></head>
<body><script language="JavaScript">
function pintar(col)
{
 document.bgColor=col;
}
</script>
<a href="pagina1.html" onMouseOver="pintar('#ff0000')"
onMouseOut="pintar('#ffffff')">Rojo</a>
-
<a href="pagina1.html" onMouseOver="pintar('#00ff00')"
onMouseOut="pintar('#ffffff')">Verde</a>
-
<a href="pagina1.html" onMouseOver="pintar('#0000ff')"
onMouseOut="pintar('#ffffff')">Azul</a><br><br><br>
<a href="pagina2.html">ver segundo problema</a>
</body></html>
```

Las llamadas a las funciones las hacemos inicializando las propiedades onMouseOver y onMouseOut:

```
<a href="pagina1.html" onMouseOver="pintar('#ff0000')"
onMouseOut="pintar('#ffffff')">Rojo</a>
```

La función 'pintar' recibe el color e inicializa la propiedad bgColor del objeto document.

```
function pintar(col)
{
 document.bgColor=col;
}
```

El segundo problema pinta de color el interior de una casilla de una tabla y lo regresa a su color original cuando salimos de la misma:

```
<html><head></head>
<body>
<script language="JavaScript">
function pintar(objeto,col)
{
  objeto.bgColor=col;
}
</script>
<table border="1">
<tr>
<td onMouseOver="pintar(this,'#ff0000')" onMouseOut="pintar(this,'#ffffff')">rojo</td>
<td onMouseOver="pintar(this,'#00ff00')" onMouseOut="pintar(this,'#ffffff')">verde</td>
<td onMouseOver="pintar(this,'#0000ff')" onMouseOut="pintar(this,'#ffffff')">azul</td>
</tr>
</table></body></html>
```

La lógica es bastante parecida a la del primer problema, pero en éste, le pasamos como parámetro a la función, la referencia a la casilla que queremos que se coloree (this):

```
<td onMouseOver="pintar(this,'#ff0000')" onMouseOut="pintar(this,'#ffffff')">rojo</td>
```

El objeto window.

Al objeto window lo hemos estado usando constantemente. Representa la ventana del navegador. window es un objeto global y tiene los siguientes métodos:

alert: Muestra un diálogo de alerta con un mensaje (a esta responsabilidad la hemos utilizado desde los primeros temas)
prompt: Muestra un diálogo para la entrada de un valor de tipo string (utilizado desde el primer momento)
confirm: Muestra un diálogo de confirmación con los botones Confirmar y Cancelar.
open y close: abre o cierra una ventana del navegador.

Podemos especificar el tamaño de la ventana, su contenido, etc.
[Variable=][window.]open(URL, nombre, propiedades)
Permite crear (y abrir) una nueva ventana. Si queremos tener acceso a ella desde la ventana donde la creamos, deberemos asignarle una variable, sino simplemente invocamos el método: el navegador automáticamente sabrá que pertenece al objeto window.
El parámetro URL es una cadena que contendrá la dirección de la ventana que estamos abriendo: si está en blanco, la ventana se abrirá con una página en blanco.
Las propiedades son una lista, separada por comas, de algunos de los siguientes elementos:

toolbar[=yes|no]
location[=yes|no]
directories[=yes|no]
status[=yes|no]
menubar[=yes|no]
scrollbars[=yes|no]

- **resizable[=yes|no]**
- **width=pixels**
- **height=pixels**

Es bueno hacer notar que a todas estas funciones las podemos llamar anteponiéndole el nombre del objeto window, seguida del método o en forma resumida indicando solamente el nombre del método (como lo hemos estado haciendo), esto es posible ya que el objeto window es el objeto de máximo nivel.
Ej: valor=window.prompt("Ingrese valor",""); ó valor=prompt("Ingrese valor","");

Para reducir la cantidad de caracteres que se escriben normalmente encontraremos los programas tipeados de la segunda forma.

El siguiente programa muestra varios de los métodos disponibles del objeto window:

```
<html><head><SCRIPT LANGUAGE="JavaScript">
function cerrar()
{
 close();// podemos escribir window.close();
}
function abrir()
{
 var ventana=open();
 ventana.document.write("Estoy escribiendo en la nueva ventana<br>");
 ventana.document.write("Segunda linea");
}
function abrirParametros()
{
 var ventana=open('','','status=yes,width=400,height=250,menubar=yes');
 ventana.document.write("Esto es lo primero que aparece<br>");
}
 function mostrarAlerta()
{
 alert("Esta ventana de alerta ya la utilizamos en otros problemas.");
}
function confirmar()
{
 var respuesta=confirm("Presione alguno de los dos botones");
 if (respuesta==true)
   alert("presionó aceptar");
 else
   alert("presionó cancelar");
}
function cargarCadena()
{
 var cad=prompt("cargue una cadena:","");
 alert("Usted ingreso "+cad);
}

</script></head>
<body>Este programa permite analizar la llamada a distintas responsabilidades del
objeto window.<br>
<form>
<input type="button" value="close()" onClick="cerrar()">
<br><br><input type="button" value="open()" onClick="abrir()">
<br><br><input type="button" value="open con parámetros" onClick="abrirParametros()">
<br><br><input type="button" value="alert" onClick="mostrarAlerta()">
<br><br><input type="button" value="confirm" onClick="confirmar()">
```

```
<br><br>
<input type="button"  value="prompt" onClick="cargarCadena()">
</form>
</body>
</html>
```

Tomado de y agradecimientos a : http://www.javascriptya.com.ar [07/02/2011 08:25]
https://www.w3schools.com/js/[05/11/2018 05:11]

Actividades Evaluativas:

- Realice todos los ejercicios propuestos para algunos ejercicios.
- Visualice ¿Qué aplicabilidad tiene para el proyecto PPI?