



UNIVERSIDAD DE EL SALVADOR
FACULTAD DE INGENIERÍA Y ARQUITECTURA
PROGRAM MICROSOFT IT ACADEMY
MTA-98-375

HTML 5 Application Development Fundamentals



Microsoft
IT Academy

Lección 9: Trabajando con gráficas y datos
con JavaScript



UNIVERSIDAD DE EL SALVADOR
FACULTAD DE INGENIERÍA Y ARQUITECTURA
PROGRAM MICROSOFT IT ACADEMY
MTA-98-375

CONTENIDO:

Animaciones con JavaScript	1
Trabajando con elementos gráficos.....	5
Enviando y recibiendo información	8
Cargando y guardando archivos.....	11
Validando entradas con JavaScript	13
Manejo de Cookies.....	15
Almacenamiento local (localStorage) con JavaScript.....	17

Animaciones con JavaScript

Ejercicio 1.

Se creara una animación de un párrafo utilizando JavaScript, con métodos se ira asignando diferentes posiciones para que se observe un movimiento, utilizando recursividad se verá que el movimiento de realiza poco a poco y no de una vez.

1. Crear un proyecto en WebMatrix y agregar un nuevo documento HTML, asignarle el nombre de index.html.
2. Se creara el documento HTML, sin estilo como se muestra a continuación:

```
<body onload = "init();">  
<h1>Animacion en JavaScript</h1>  
<p id = "original">Se observa como me muevo en pantalla.</p>  
</body>
```

Animacion en JavaScript

Se observa como me muevo en pantalla.

3. Ahora que se tiene los elementos que se desean animar, se establecerá la función init(), la cual ya es llamada por el manejador de eventos onLoad, en esta se declararan algunas variables, como también, se llamara a otro método que es el que realizara la animación.

```
function init() {  
    parrafo = document.getElementById("original");  
    parrafo.style.position = "absolute";  
    actual = 0;  
    moverParrafo();  
}
```

4. Se procede a agregar el método moverParrafo(), ya que es el que se encarga de mover el párrafo, y es convocado primeramente por la función init.

```
function moverParrafo() {  
    siguiente = actual + "px";  
    actual += 1;  
    if (actual > 300) {  
        bandera = 1;  
    }  
    parrafo.style.left = siguiente;  
    var velocidad = 18;  
    setTimeout(moverParrafo, velocidad);  
}
```

Con estas funciones ya se podrá observar el movimiento constante del párrafo.

Animacion en JavaScript

Se observa como me muevo en pantalla.

Animacion en JavaScript

Se observa como me muevo en pantalla.

5. Debido a que el elemento al llegar al límite establecido (300) regresa a su punto de partida, realizaremos unos cambios en las funciones para que no solo aparezca al inicio de nuevo, si no se observe esa transición también, por lo cual se observara como que el elemento rebotara de lado a lado.

```
function moverParrafo() {  
  if (bandera == 0) {  
    siguiente = actual + "px";  
    actual += 1;  
    if (actual > 300) {  
      bandera = 1;  
    }  
    parrafo.style.left = siguiente;  
    var velocidad = 18;  
    setTimeout(moverParrafo, velocidad);  
  }  
  else {  
    siguiente = actual + "px";  
    actual -= 1;  
    if (actual < 1) {  
      bandera = 0;  
    }  
    parrafo.style.left = siguiente;  
    var velocidad = 18;  
    setTimeout(moverParrafo, velocidad);  
  }  
}
```

```
function init() {  
  parrafo = document.getElementById("original");  
  parrafo.style.position = "absolute";  
  actual = 0;  
  bandera = 0;  
  moverParrafo();  
}
```

Ahora se observa que el elemento rebota de lado a lado sin detenerse.


Ejercicio 2.

Se creara un proyecto de animación, similar al anterior, pero en este se observara la interacción con el usuario para que este altere la velocidad de movimiento del elemento.

1. Crear un proyecto en WebMatrix y agregar un nuevo documento HTML, asignarle el nombre de index.html.
2. Buscar en internet una imagen, descargarla y luego copiarla y pegarla en el proyecto, asignarle el nombre de imagen y la terminación correspondiente.
3. Se creara el documento HTML, sin estilo como se muestra a continuación:

```
<body onload = "init();">
<h1>Animacion de imagen y control de velocidad</h1>
<form>
<input id = "velocidad" type = "number" value = "18" min = "5" max = "100"></input>
</form>
</img>
</body>
```

Animacion de imagen y control de velocidad



4. Teniendo los elementos que se quieren animar y los parámetros a utilizar se procede a crear el método init(), el cual es convocado cuando la página carga completamente.

```
function init() {
    imagen = document.getElementById("original");
    imagen.style.position = "absolute";
    actual = 0;
    moverImagen();
}
```

5. Luego se agrega el método moverImagen() que es el que se encarga de mover la imagen por la pantalla sin detenerse. Se observa la funcionalidad correcta de la aplicación, modificando el campo de texto asignado se observa que la velocidad aumenta si el valor disminuye y viceversa.

```
function moverImagen() {  
    sig = actual + "px";  
    actual += 1;  
    if (actual > 300) {  
        actual = 0;  
    }  
    imagen.style.left = sig;  
    var velocidad = document.getElementById("velocidad").value;  
    setTimeout(moverImagen, velocidad);  
}
```

Animacion de imagen y control de velocidad



Trabajando con elementos gráficos.

Ejercicio 1.

Se utilizarán los métodos de objetos en JavaScript para agregar elementos de tipo imagen, utilizando una función se toma ciertos parámetros de la imagen los cuales son asignados al elemento `` que va a ser agregado, si este no se puede agregar, se utiliza el atributo alternativo también.

1. Crear un proyecto en WebMatrix y agregar un nuevo documento HTML, asignarle el nombre de index.html.
2. Se creará el documento HTML, sin estilo como se muestra a continuación:

```
<body>
<form>
  <input id = "nombre" type = "text"></input>
</form>
<button onclick="mostrarImagen(276,110, 'Imagen');">Mostrar Imagen</button>
</body>
```



3. Descargue varias imágenes de su navegador, y agréguelas a su proyecto con los nombres que usted desee. Estas serán agregadas al documento.
4. Para que la aplicación funcione correctamente es necesario codificar la función `mostrarImagen()` con sus parámetros correspondientes, ancho, alto y un texto alternativo.

```
<script type="text/javascript">
  function mostrarImagen(width, height, alt) {
    var img = document.createElement("img");
    img.src = document.getElementById("nombre").value;
    img.width = width;
    img.height = height;
    img.alt = alt;
    document.body.appendChild(img);
  }
</script>
```

Ahora se puede observar el correcto funcionamiento de la aplicación web, si no se encuentra la imagen con el nombre especificado, siempre genera el recuadro con las dimensiones establecidas, pero coloca el texto alternativo en vez de la imagen. Asegúrese de escribir la terminación del tipo de dato de la imagen.



Ejercicio 2.

Se creara un proyecto para el uso de JavaScript en Canvas, con el cual se actualizara la pantalla del usuario dinámicamente cada momento.

1. Crear un proyecto en WebMatrix y agregar un nuevo documento HTML, asignarle el nombre de index.html.
2. Se creara el documento HTML, sin estilo como se muestra a continuación:

```
<body onload = "init();">
<h1>Reloj analogo utilizando JavaScript en canvas</h1>
<canvas id = "reloj" width = "200" height = "200"></canvas>
</body>
```

3. En el elemento Canvas creado, se pretende crear un reloj con actualización dinámica de las agujas, primeramente se creara la función init(), la cual establecerá ciertos elementos para iniciar a configurar nuestro elemento Canvas.

```
function init() {
    var canvas =
    document.getElementById("reloj");
    dctx = canvas.getContext('2d');
    dctx.fillStyle = "black";
    centroX = 100;
    centroY = 100;
    longitud = 100;
    mostrarAgujas();
}
```

Con esto se especifica el centro del reloj y la longitud de las agujas.

4. Al crear el init, se procederá a crear 3 funciones diferentes, una para mostrar las agujas, la cual calculara con la hora actual la dirección que debe de tener las agujas, dependiendo de los segundos, minutos y horas obtenidos, después llamara a otra función con la cual cada una de estas agujas es mostrada en pantalla, y esta otra función llamara a la función que con un cálculo matemático se encarga de dibujar ya en pantalla cada una de las agujas.

```
function mostrarAgujas() {  
    dctx.clearRect(0, 0, 200, 200);  
    var hora = new Date();  
    seg = hora.getSeconds();  
    min = hora.getMinutes() + seg / 60;  
    horas = hora.getHours() + min / 60;  
    mostrarAguja(seg / 60, 0.002);  
    mostrarAguja(min / 60, 0.005);  
    mostrarAguja(horas / 12, 0.01);  
    var vel = 1000;  
    setTimeout(mostrarAgujas, vel);  
}  
  
function mostrarAguja(fraccion, ancho) {  
    dctx.beginPath();  
    dctx.moveTo(centroX, centroY);  
    dibujarAguja(fraccion - ancho);  
    dibujarAguja(fraccion + ancho);  
    dctx.fill();  
}  
  
function dibujarAguja(fraccion) {  
    dctx.lineTo(centroX + longitud * Math.sin(2 * Math.PI * fraccion), centroY - longitud * Math.cos(2 * Math.PI * fraccion));  
}
```

Reloj analogo utilizando JavaScript en canvas



Enviando y recibiendo información

Ejercicio 1.

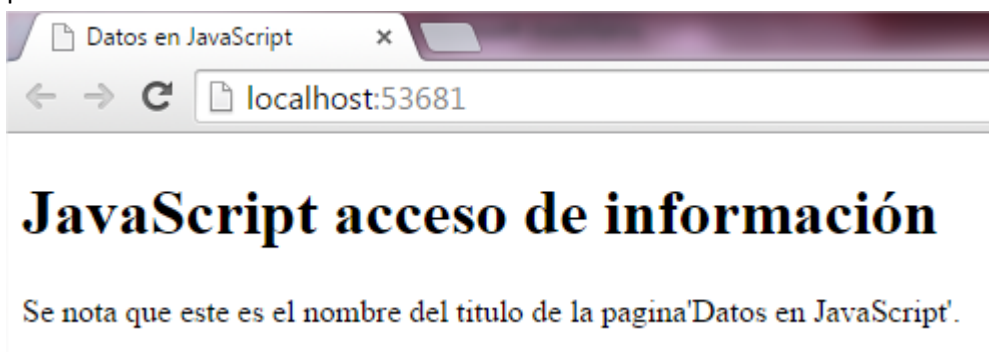
Se creara un archivo para observar el comportamiento de la página web para obtener información del navegador, utilizando diferentes métodos correspondientes a elementos en JavaScript.

1. Crear un proyecto en WebMatrix y agregar un nuevo documento HTML, asignarle el nombre de index.html.
2. Se creara el documento HTML, sin estilo como se muestra a continuación:

```
<!DOCTYPE html>

<html lang="en">
  <head>
    <meta charset="utf-8" />
    <title>Datos en JavaScript</title>
    <script type = "text/javascript">
      function init() {
        var parrafo = document.getElementById('parrafo');
        message = "Se nota que este es el nombre del titulo de la pagina'" + document.title + "'.";
        parrafo.innerHTML = message;
      }
    </script>
  </head>
  <body onload = "init();">
    <h1>JavaScript acceso de información</h1>
    <p id = "parrafo"></p>
  </body>
</html>
```

Se puede observar que la aplicación web, muestra el título de página, esto se realiza para reducir la cantidad de veces que el titulo se puede encontrar en nuestra página, par ano volver a escribirlo, accedemos a cierta información que ya se encuentra en nuestra página web y la presentamos adonde nosotros deseemos.



Ejercicio 2.

Se realizara un proyecto para el parseo de datos complicados, y se mostraran como corresponde, ya que a veces la transferencia de datos se recibe en cadenas de texto, las cuales poseen toda la información en una sola, siempre hay elementos que separan estas palabras, son los que se utilizan en este ejemplo.

1. Crear un proyecto en WebMatrix y agregar un nuevo documento HTML, asignarle el nombre de index.html.
2. Se creara el documento HTML, sin estilo como se muestra a continuación:

```
<body onload = "init();">
<h1>Parseando datos complejos</h1>
<p id = "parrafo">Bienvenido.</p>
</body>
```

3. Teniendo el elemento al cual se agregara texto, se procede a crear la función de init(), en la cual se realizara el parseo de una cadena de caracteres compleja, para esto crearemos dicha cadena e ingresaremos un valor determinado que se desea mostrar en específico.

```
datosEjemplo = "Lorena: 8.49; Maria: 9.36;Luis: 8.56; Salvador: 8.49; Andrea: 9.23; Rodrigo: 7.52";
datoAMostrar = "Lorena";
```

4. Luego creamos la función correspondiente, para que el elemento <script> se observe como a continuación.

```
<script type = "text/javascript">
    datosEjemplo = "Lorena: 8.49; Maria: 9.36;Luis: 8.56; Salvador: 8.49; Andrea: 9.23; Rodrigo: 7.52";
    datoAMostrar = "Lorena";
    function init() {
        var parrafo =
            document.getElementById("parrafo");
        var listaDatos = datosEjemplo.split(';');
        for (j = 0; j < listaDatos.length; j++) {
            elementos = listaDatos[j].split(':');
            var nombre = elementos[0].trim()
            if (nombre == datoAMostrar) {
                var message = "Dada la lista de nombres y notas '" +
                    datosEjemplo + "', el programa obtuvo la nota especifica de: " +
                    elementos[1].trim() + " que pertenece a " + nombre + ".";
                parrafo.innerHTML = message;
            }
        }
    }
</script>
```

Se observa el correcto funcionamiento de la aplicación, ya que se muestra todo el conjunto de elementos y luego se muestra el elemento específico, intenta cambiar el elemento a mostrar para ver si cambia el dato específico.



Parseando datos complejos

Dada la lista de nombres y notas 'Lorena: 8.49; Maria: 9.36; Luis: 8.56; Salvador: 8.49; Andrea: 9.23; Rodrigo: 7.52', el programa obtuvo la nota específica de: 8.49 que pertenece a Lorena.

Cargando y guardando archivos

Ejercicio 1.

Se creara una proyecto para la carga de un archivo, ese archivo será validado para que sea solamente de tipo imagen, por lo tanto tendrá ciertas posibilidades de extensiones de archivo, se mostrara mensaje cuando se una imagen, con su tamaño y también cuando no sea una imagen presentando un error.

1. Crear un proyecto en WebMatrix y agregar un nuevo documento HTML, asignarle el nombre de index.html.
2. Se creara el documento HTML, sin estilo como se muestra a continuación:

```
<h1>JavaScript accesses local files</h1>  
<input type = 'file' onchange = 'manejarSeleccionDeArchivo(this);' />
```

JavaScript accesses local files

Examinar...

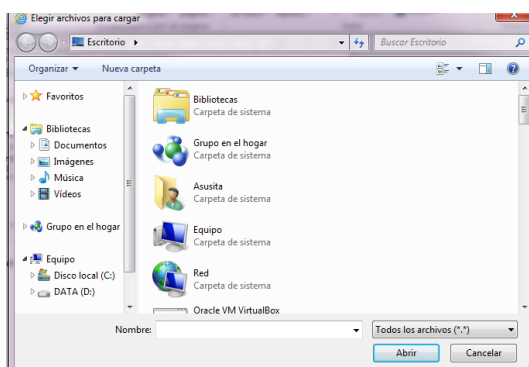
3. Después de tener los elementos necesarios, como se observa es una entrada de tipo “file”, por lo tanto se examinara para obtener un archivo de la computadora, se creara la función que es llamada cuando el valor de este input cambia (onchange), la cual se le colocara de nombre manejarSeleccionDeArchivo, como parámetro se lo enviare el nuevo dato que cambio. En esta se obtendrá la extensión del archivo y se validara.

```
function manejarSeleccionDeArchivo(item) {  
    var archivo = item.files[0];  
    var nombreArchivo = archivo.name;  
    var ultimoCaracter = nombreArchivo.lastIndexOf('.');  
    if (ultimoCaracter == -1) {  
        noCompatible(nombreArchivo);  
        return;  
    }  
    var ext = nombreArchivo.substr(ultimoCaracter + 1);  
    if (ext.toLowerCase() in { 'gif': '',  
        'jpg': '',  
        'png': '',  
        'tif': ''  
    }) {  
        reconocer(archivo);  
    } else {  
        noCompatible(nombreArchivo);  
    }  
}
```

4. Como se observa anteriormente esta función que valida, llama a otras dos funciones, una para cuando es compatible y la otra cuando no lo es, por lo tanto se crearan estas funciones, las cuales presentaran mensajes correspondientes con texto agregado.

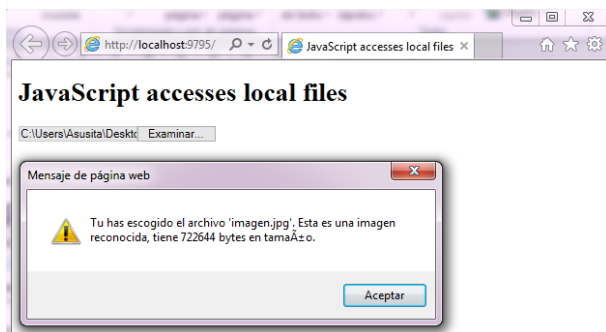
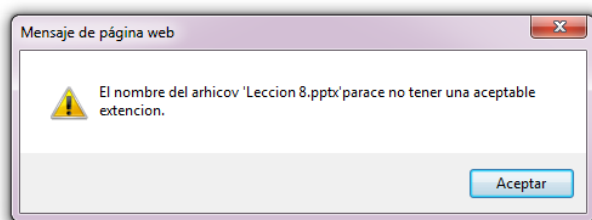
```
function noCompatible(nombreArchivo) {  
    var mensaje = "El nombre del arhico' " + nombreArchivo + "'parace no tener una aceptable extension.";  
    alert(mensaje);  
}  
  
function reconocer(manejoArchivo) {  
    var size = manejoArchivo.size;  
    var nombreArchivo = manejoArchivo.name;  
    var mensaje = "Tu has escogido el archivo '" + nombreArchivo + "'. Esta es una imagen reconocida, tiene " + size + " bytes en tamaño.";  
    alert(mensaje);  
}
```

Se observa la correcta funcionalidad de la aplicación.



JavaScript accesses local files

C:\Users\Asusita\Desktop Examinar...



Validando entradas con JavaScript

Ejercicio 1.

Se creara una validación de NIT, esta validación se hace del lado del cliente, es decir en el momento que el usuario está ingresando los datos se verifica que ingresa y se corrige si eso es necesario.

1. Crear un proyecto en WebMatrix y agregar un nuevo documento HTML, asignarle el nombre de index.html.
2. Se creara el documento HTML, sin estilo como se muestra a continuación:

```
<body>
  <h1>Validacion de formularios</h1>
  <form>
    Ingrese su NIT, en el format: ####-#####-###-#, en el cual cada # es un digito:
    <input id = "nit" type = 'text' size = '17' onkeyup="corregir();">.
  </form>
</body>
```

Validacion de formularios

Ingrese su NIT, en el format: ####-#####-###-#, en el cual cada # es un digito:

3. Teniendo los elementos necesarios, se observa que se ha declarado un manejador de eventos "onkeyup", esto quiere decir que se disparara la función corregir, cada vez que se presione una tecla y esta se suelte, en ese momento se validara la nueva cadena que genera esa tecla.

```
function corregir() {
  var nit = document.getElementById("nit");
  var value = nit.value;
  var longitudActual = value.length;
  if (longitudActual) {
    var ultimoCaracter = value.substring(longitudActual - 1);
    switch (longitudActual) {
      case 5:
      case 12:
      case 16:
        if (ultimoCaracter != '-') {
          value = value.substring(0, longitudActual - 1);
        }
        break;
      default:
        if (!/\d/.test(ultimoCaracter)) {
          value = value.substring(0, longitudActual - 1);
        }
    }
  }
}
```

```
if (longitudActual > 17) {  
    value = value.substring(0, longitudActual- 1);  
}  
longitudActual = value.length;  
switch (longitudActual) {  
    case 4:  
    case 11:  
    case 15:  
        value += "-";  
}  
nit.value = value;  
}
```

Teniendo el código completo se puede observar en el funcionamiento, que cada vez que se ingrese un valor no numero, este elimina ese valor, como también en ciertos momentos del ingreso, la cadena automáticamente agrega un guion (-). De igual manera si se excede la cantidad máxima de caracteres (17), la función elimina esos caracteres excedentes.

Validacion de formularios

Ingrese su NIT, en el format: ####-#####-###-#, en el cual cada # es un digito:

Validacion de formularios

Ingrese su NIT, en el format: ####-#####-###-#, en el cual cada # es un digito:

Manejo de Cookies

Ejercicio 1.

Se creara un proyecto para el manejo de cookies, en el cual se modificara un valor deseado por el usuario y se observara que ese cambio persiste en el futuro.

1. Crear un proyecto en WebMatrix y agregar un nuevo documento HTML, asignarle el nombre de index.html.
2. Se creara el documento HTML, sin estilo como se muestra a continuación:

```
<body onload = "init();">
<h1>Uso de Cookies</h1>
<p id = "welcome">Welcome.</p>
<form>
Puedes aumentar el nivel cuando tu quieras, ahora esta establecido en
<input id = "nivel" type = "number" min = "1" max = "100" oninput = "guardarNivel();" />.
</form>
</body>
```

Uso de Cookies

Parece que es primera vez que juegas, empiezas en nivel 1.

Puedes aumentar el nivel cuando tu quieras, ahora esta establecido en .

3. Ya teniendo los elementos podemos aplicar las funciones, se requiere la función init primeramente para que los valores sean establecidos, esta genera un valor 1 para el nivel por defecto.

```
function init() {
    var Mensaje;
    objetoNivel = document.getElementById("nivel");
    var welcome = document.getElementById("welcome");
    var Nivel = getCookie("Nivel");
    if (Nivel == null || Nivel == '') {
        Mensaje = "Parece que es primera vez que juegas, empiezas en nivel 1.";
        Nivel = 1;
    } else {
        Mensaje = "En el ultimo juego, llegas al nivel " + Nivel + ". Ahora empezaras desde ahi.";
    }
    welcome.innerHTML = Mensaje;
    objetoNivel.value = Nivel;
}
```

4. Si el nivel ya existe no lo coloca en 1, si no que obtiene el valor anterior utilizado, o colocado por el usuario, esto muestra una configuración realizada por el usuario, la cual permanece guardada en el navegador o computadora del usuario final, para que realice ese guardado y la recuperación respectiva se codifican las siguientes funciones.

```
function getCookie(nombreCookie) {  
    var i, x, y, ArregloCookies = document.cookie.split(";");  
    for (i = 0; i < ArregloCookies.length; i++) {  
        x = ArregloCookies[i].substr(0, ArregloCookies[i].indexOf("="));  
        y = ArregloCookies[i].substr(ArregloCookies[i].indexOf("=") + 1);  
        x = x.replace(/^\s+|\s+$/g, "");  
        if (x == nombreCookie) {  
            return unescape(y);  
        }  
    }  
}
```

5. Después cada vez que se realice un cambio en el apartado de texto (input) que se tiene en la página, esto generara una actualización de esta cookie creada, por lo tanto se crearan 2 funciones para dicha acción, las que se muestran a continuación.

```
function guardarNivel() {  
    setCookie("Nivel", objetoNivel.value, 10);  
}  
  
function setCookie(nombreCookie, value, exdays) {  
    var exdate = new Date();  
    exdate.setDate(exdate.getDate() + exdays);  
    var c_value = escape(value) + ((exdays == null) ? "" : "; expires=" + exdate.toUTCString());  
    document.cookie = nombreCookie + "=" + c_value;  
}
```

Con esto se podrá observar que los datos del usuario se guardan al ser actualizados, si se cierra el navegador siempre permanecen ese dato de nivel.

Uso de Cookies

Parece que es primera vez que juegas, empiezas en nivel 1.

Puedes aumentar el nivel cuando tu quieras, ahora esta establecido en .

Uso de Cookies

En el ultimo juego, llegas al nivel 10. Ahora empezaras desde ahi.

Puedes aumentar el nivel cuando tu quieras, ahora esta establecido en .

Almacenamiento local (localStorage) con JavaScript

Ejercicio 1.

Se creara el mismo proyecto anterior con la diferencia de que en este se utilizara localStorage en vez de cookies, lo cual se observara que es más sencillo y mucho más practico a la hora de guardar y acceder a él.

1. Crear un proyecto en WebMatrix y agregar un nuevo documento HTML, asignarle el nombre de index.html.
2. Se creara el documento HTML, sin estilo como se muestra a continuación:

```
<body onload = "init();">
<h1>Uso de Cookies</h1>
<p id = "welcome">Welcome.</p>
<form>
Puedes aumentar el nivel cuando tu quieras, ahora esta establecido en
<input id = "nivel" type = "number" min = "1" max = "100" oninput = "guardarNivel();" />.
</form>
</body>
```

Uso de localStorage

Parece que es primera vez que juegas, empiezas en nivel 1.

Puedes aumentar el nivel cuando tu quieras, ahora esta establecido en

3. Ya teniendo los elementos podemos podremos aplicar las funciones, se requiere la función init primeramente para que los valores sean establecidos, esta genera un valor 1 para el nivel por defecto.

```
function init() {
    var Mensaje;
    objetoNivel = document.getElementById("nivel");
    var welcome = document.getElementById("welcome");
    var Nivel = localStorage.nivel;
    if (Nivel == null || Nivel == '') {
        Mensaje = "Parece que es primera vez que juegas, empiezas en nivel 1.";
        Nivel = 1;
    } else {
        Mensaje = "En el ultimo juego, llegas al nivel " + Nivel + ". Ahora empezaras desde ahi.";
    }
    welcome.innerHTML = Mensaje;
    objetoNivel.value = Nivel;
}
```

Como se observa, a diferencia que el anterior no se llama a una función creada por nosotros para acceder al dato guardado, si no que se accede a una variable llamada nivel, la cual se encuentra en el localStorage.

4. Para lograr guardar la variable, se crea la función llamada guardarNivel, la cual es disparada cada vez que se ingresa nueva información en la caja de texto. La información seguirá teniendo el mismo valor aunque se cierre el navegador.

```
function guardarNivel() {  
    localStorage.nivel = objetoNivel.value;  
}
```

Uso de localStorage

Parece que es primera vez que juegas, empiezas en nivel 1.

Puedes aumentar el nivel cuando tu quieras, ahora esta establecido en .

Uso de localStorage

En el ultimo juego, llegas al nivel 25. Ahora empezaras desde ahi.

Puedes aumentar el nivel cuando tu quieras, ahora esta establecido en .

Uso de localStorage

En el ultimo juego, llegas al nivel 910. Ahora empezaras desde ahi.

Puedes aumentar el nivel cuando tu quieras, ahora esta establecido en .