JavaScript
parte 2

Carolina Caballero

Agenda

- Control de Flujo
- Funciones
- Fechas
- Ventanas

Control de Flujo

Comando if

- Define la ejecución de un bloque de comandos en base a una condición.

```
if (condition) {
    block of code to be executed if the condition is true
}
```

```
if (hour < 18) {
    greeting = "Good day";
}
```

Comando if...

```
if (condition) {
    block of code to be executed if the condition is true
} else {
    block of code to be executed if the condition is false
}
```

```
if (hour < 18) {
    greeting = "Good day";
} else {
    greeting = "Good evening";
}
```

Comando if...

```
if (condition1) {
    block of code to be executed if condition1 is true
} else if (condition2) {
    block of code to be executed if the condition1 is false and condition2
is true
} else {
    block of code to be executed if the condition1 is false and condition2
is false
}
```

```
if (time < 10) {
    greeting = "Good morning";
} else if (time < 20) {
    greeting = "Good day";
} else {
    greeting = "Good evening";
}
```

- File: Sample10.html

Comando switch

- Define la ejecución de muchas alternativas de bloques de comandos, dependiendo del valor de una expresión

```
switch(expression) {
    case n:
        code block
        break;
    case n:
        code block
        break;
    default:
        code block
}
```

```
switch (new Date().getDay()) {
    case 4:
    case 5:
        text = "Soon it is Weekend";
        break;
    case 0:
    case 6:
        text = "It is Weekend";
        break;
    default:
        text = "Looking forward to the Weekend";
}
```

- File: Sample11.html

Comando for

- Sirve para crear un ciclo (loop), es el mas comunmente usado.

```
for (statement 1; statement 2; statement 3) {
    code block to be executed
}
```

```
for (i = 0; i < 5; i++) {
    text += "The number is " + i + "<br>";
}
```

- File: Sample12.html

Comando while

- Ejecuta un bloque de comandos mientra la condición sea verdadera.

```
while (condition) {
    code block to be executed
}
```

```
while (i < 10) {
    text += "The number is " + i;
    i++;
}
```

- File: Sample13.html

Comando do ... while

- Ejecuta un bloque de comandos mientra la condicion sea verdadera. Se ejecuta al menos 1 vez.

```
do {
    code block to be executed
}
while (condition);
```

```
do {
    text += "The number is " + i;
    i++;
}
while (i < 10);
```

- File: Sample14.html

Ejercicios

- Haz un loop para sacar la tabla del 5 (sin usar operador de multiplicación) (usa for).
- Haz un loop para contar de 10 a 1. Despliega los valores con salto de línea. (usa while).
- Haz una función para adivinar un número. El usuario captura un valor, la función debe dar pistas si es menor o mayor, y si adivina debe mandar un mensaje de que ha acertado.

Agenda

- Control de Flujo
- Funciones
- Fechas
- Ventanas

Funciones

Funciones

- Una función de JavaScript es un bloque de código diseñado para ejecutar una tarea en particular.
- Se pueden reutilizar fácilmente.
- El orden de los parámetros es fundamental.
- Los nombres de las funciones deben seguir las reglas que las variables.

```
function name(parameter1, parameter2, parameter3) {
    code to be executed
}
```

Función - párametros

- Esta función recibe un parámetro de entrada.

```
function sayHello (msg)
{
   alert ("Hello " + msg);
}

sayHello ("World!");
```

- File: Sample15.html

Función - return

- Esta función recibe 2 parámetros y regresa un resultado.

```
function suma (num1, num2) {
  var resultado = num1 + num2;
  return resultado
}

var numero1 = 5;
var numero2 = 6;
var suma = suma(numero1, numero2);
```

- File: Sample15.html

Funciones como variables

- Una variable puede ser una función

```
var hola = function (msg) {
    alert('Hola ' + msg);
}

hola('Mundo');
```

- File: Sample15.html

Pasando Funciones a Funciones

- Una función puede pasarse como parámetro a otra función como si fuera otra variable

```
var mostrar = function(que) {
   que('funcion mostrar');
}

var hola = function (quien) {
   alert('Hola '+quien);
}

mostrar(hola);  // "Hola funcion mostrar"
```

- File: Sample15.html

Pasando funciones

- Una función puede asignarse a una variable.

```
function hacerAlgo() {
   alert('diste click!');
}

document.onclick=hacerAlgo;
```

- File: Sample15.html

Pasando funciones

- Llamado entre funciones y asignación de función.

```
function presionaBoton() {
   alert('presionaste boton!');
}

function hazAlgo() {
   return presionaBoton;
}

document.onclick = hazAlgo();
```

- File: Sample15.html

Funciones Anónimas

- Las funciones se pueden asignar directamente a una variable, sin definirle nombre.

```
document.onmousedown = function() {
alert("Presionaste el boton");
}
```

- File: Sample15.html

Autoinvocación de Funciones

- La función se invoca al desplegar la variable x.

```
var x = (function () {return(5+6)})();
document.writeln(typeof(x)+'<BR>');
document.writeln(x);
```

- File: Sample15.html

Autoinvocación de Funciones

```
function alerta(msg) {
   alert(msg);
}

alerta( (function(animal1,animal2) { return animal1 + ' ama ' +
animal2; } ) ('gato', 'perro') );
```

- File: Sample15.html

Autoinvocación de Funciones

```
function mostrar(msg) {
   document.writeln(msg+'<br>');
}

function crearCadena(num1, num2) {
   return num1 + ' + ' + num2 + ' = ' + (num1+num2);
}

for (i=0; i<10; i++) {
  mostrar( crearCadena(i, i+5) );
}
```

- File: Sample15.html

Apuntador de las Funciones

```
var original = function () {
   alert('Hola Mundo');
}

var copia = original;

var original = function () {
   alert('adios mundo');
}

copia();
original();
```

- File: Sample15.html

Visibilidad de una Función (Scope)

```
var global = function () {
   alert('Hola Mundo');
}
var contenedora = function() {
  var subFuncion = function() {
    alert("Soy Local");
    global();
  }
  global();
  subFuncion();
}
contenedora();
subFuncion();
```

- File: Sample15.html

Ejercicio

- Crear una función que calcule y regrese el factorial pasando el número como argumento. Capturar un número usando la función prompt y llamar la función factorial pasando como argumento el número capturado por medio la función prompt. Desplegar el resultado regresado en un alert.

Agenda

- Control de Flujo
- Funciones
- Fechas
- Ventanas

Fechas Julianas

- Son una manera de referenciar la hora y fecha basado en cuanto tiempo ha pasado desde un número arbitrario en el pasado.
- El punto de partida en Javascript es 1 de Enero de 1970 GMT
- Javascript cuenta el número de milisegundos que han pasado desde el 1 de Enero de 1970
- Javascript puede manejar alrededor de 285,616 años a ambos lados de 1970

Creando una Fecha

- Existen 4 diferentes maneras para crear una fecha

```
new Date()
new Date(milliseconds)
new Date(dateString)
new Date(year, month, day, hours, minutes, seconds, milliseconds)
```

- var fecha = new Date();
- var fecha = new Date('May 8, 1984');

Creando una Fecha

- var fecha = new Date('May 8, 1984 8:14:15 pm');
- var fecha = new Date('May 8, 1984 8:14:15 pm GMT');
- var fecha = new Date('May 8, 1984 8:14:15 pm GMT-7');
- var fecha = new Date('January 1, 1970'); // Parser
- var fecha = new Date(0); // Número Juliano
- var fecha = new Date(1970, 0, 1); // Argumentos

- File: Sample16.html

Creando una Fecha

- Sintaxis:

```
new Date(year, month, day, hours, minutes, seconds, milliseconds)
```

fecha = new Date(2009, 12, 25);

fecha = new Date(1984, 5, 8, 14, 54, 10, 30);

Creando un reloj

```
function reloj() {
    var hoy = new Date();
    var cadena = hoy.getHours()+':'+hoy.getMinutes()+':'+hoy.getSeconds();
    document.getElementById('relojDiv').innerHTML=cadena;
    setTimeout('reloj()',1000);
}
reloj();
```

- File: Sample16.html

Obtener número de días de un Mes

```
Date.prototype.diasEnMes = function () {
    return new Date(this.getFullYear(),
                    this.getMonth()+1,
                    0)
                    .getDate();
}

var d = new Date();
document.writeln('Numeros de días en este mes: '+d.diasEnMes());
```

- File: Sample16.html

Las Fechas en Javascript son Pasadas por Referencia

```
var fechaOriginal = new Date('January 1, 1970');

function cambiarYear(fecha) {
  fecha.setFullYear(1980);
}
cambiarYear(fechaOriginal);
document.writeln(fechaOriginal.getFullYear() + "<br/>"); //1980
```

- File: Sample16.html

Las Fechas en Javascript son Pasadas por Referencia

```
var fechaOriginal1 = new Date('January 1, 1970');
var nuevaFecha = fechaOriginal1;

nuevaFecha.setFullYear(1980);

document.writeln(fechaOriginal1.getFullYear()); //1980
```

- File: Sample16.html

Métodos de Date

Método	Descripción
getDate()	Regresa el Día: 12/15/2006
getDay()	Regresa el día de la semana (0=domingo, 1=lunes, etc)
getFullYear()	Regresa el año en 4 digitos
getHours()	Regresa la hora en formato militar (0-23)
getMilliseconds()	Regresa los milisegundos
getMinutes()	Regresa los minutos (0-59)
getMonth()	Regresa el mes (0=Ene, 1=Feb, etc)
getSeconds()	Regresa los segundos (0-59)
getTime()	# de milisegundos desde 1 de Enero de 1970
getTimezoneOffset()	Regresa el desplazamiento en minutos desde el GMT
getUTCDate()	Regresa el día como UTC
getUTCDay()	Regresa el día de la semana como UTC

Métodos de Date

Método	Descripción
getUTCFullYear()	Regresa el año en UTC de 4 digitos
getUTCHours()	Regresa las horas en UTC
getUTCMiliseconds()	Regresa los milisegundos en UTC
getUTCMinutes()	Regresa los minutos en UTC
getUTCMonth()	Regresa el mes en UTC
getUTCSeconds()	Regresa los segundos en UTC
parse(cadena)	Static-Convierte la cadena en timestamp
setDate(dia)	Establece el día de un mes
setFullYear(año)	Establece el año de 4 digitos
setHours(hora)	Establece la hora
setMiliseconds(milisegundos)	Establece los milisegundos
setMinutes(minutos)	Establece los minutos

Métodos de Date

Método	Descripción
setMonth(mes)	Establece el mes
setSeconds(segundos)	Establece los segundos
setTime(timestamp)	Establece la fecha a partir de un timestamp
setUTCDate(dia)	Establece el dia del mes en UTC
setUTCFullYear(año)	Establece el año de 4 digitos en UTC
setUTCHours(hora)	Establece la hora en UTC
setUTCMiliseconds(segs)	Establece los segundos en UTC
setUTCMinutes(minutos)	Establece los minutos en UTC
setUTCMonth(mes)	Establece el mes en UTC
setUTCSeconds(segs)	Establece los segundos en UTC
toLocaleDateString()	Imprime la fecha usando la localización del navegador
toLocaleString()	Imprime la fecha y hora usando la localización del navegador
toLocaleTimeString()	Imprime la hora usando la localización del navegador

Métodos de Date

Método	Descripción
toString()	Imprime la fecha como un String
toTimeString()	Regresa la hora como un String
toUTCString()	Regresa la fecha como un String en GMT
UTC(año, mes, dia, horas, min, seg, ms)	Static-Regresa la fecha como un timestamp UTC

Ejercicios

- Captura los campos día, mes y año y crea un objeto fecha.
- En base a la fecha capturada, programa 6 citas futuras, para los proximos 6 meses, guardarlo en array y desplegarlo en una tabla.

Agenda

- Control de Flujo
- Funciones
- Fechas
- Ventanas

Ventanas

- Para abrir una pagina web en una nueva ventana se usa :

```
window.open(URL, name, specs, replace)
```

- **URL:** Opcional. URL de la página a abrir. Si no se especifíca se abre una about:blank

Ventanas

- **name:** Opcional. Especifica el atributo target o el nombre de la ventana. Soporta:
  - _blank - URL is loaded into a new window. This is default
  - _parent - URL is loaded into the parent frame
  - _self - URL replaces the current page
  - _top - URL replaces any framesets that may be loaded
  - *name* - The name of the window (**Note:** the *name* does not specify the title of the new window)

Ventanas

- **specs:** Opcional. Lista de atributos de la ventana separados por coma:

channelmode=yes|no|1|0	Whether or not to display the window in theater mode. Default is no. IE only
directories=yes|no|1|0	Obsolete. Whether or not to add directory buttons. Default is yes. IE only
fullscreen=yes|no|1|0	Whether or not to display the browser in full-screen mode. Default is no. A window in full-screen mode must also be in theater mode. IE only
height=pixels	The height of the window. Min. value is 100
left=pixels	The left position of the window. Negative values not allowed
location=yes|no|1|0	Whether or not to display the address field. Opera only

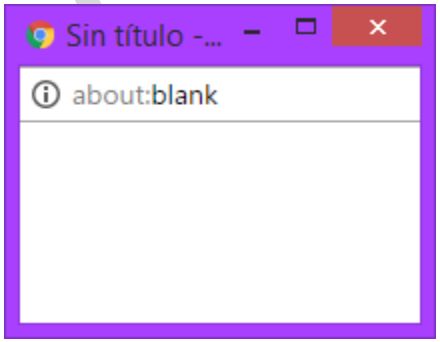
Ventanas

menubar=yes|no|1|0	Whether or not to display the menu bar
resizable=yes|no|1|0	Whether or not the window is resizable. IE only
scrollbars=yes|no|1|0	Whether or not to display scroll bars. IE, Firefox & Opera only
status=yes|no|1|0	Whether or not to add a status bar
titlebar=yes|no|1|0	Whether or not to display the title bar. Ignored unless the calling application is an HTML Application or a trusted dialog box
toolbar=yes|no|1|0	Whether or not to display the browser toolbar. IE and Firefox only
top=pixels	The top position of the window. Negative values not allowed
width=pixels	The width of the window. Min. value is 100

Ventanas

- Esta instrucción abre una nueva ventana de tamaño 200x100

```
var myWindow = window.open("", "", "width=200,height=100");
```



- File: Sample17.html

Ventanas

- Abre una ventana y modifica directamente el contenido de la misma:

```
var myWindow = window.open("", "MsgWindow", "width=200,height=100");
myWindow.document.write("<p>This is 'MsgWindow'. I am 200px wide and 100px
tall!</p>");
```

- Remplaza la ventana actual:

```
var myWindow = window.open("", "_self");
myWindow.document.write("<p>I replaced the current window.</p>");
```

- File: Sample17.html

Ventanas

- Ventana con diferentes atributos:

```
window.open("https://www.w3schools.com", "_blank",
"toolbar=yes,scrollbars=yes,resizable=yes,top=500,left=500,width=400,height=400
");
```

- Abrir multiples ventanas:

```
window.open("http://www.google.com/");
window.open("https://www.w3schools.com/");
```

- File: Sample17.html

Ventanas

- Abrir y Cerrar:

```
function openWin() {
    myWindow = window.open("", "myWindow",
"width=200,height=100");    // Opens a new window
}

function closeWin() {
    myWindow.close();    // Closes the new window
}
```

- File: Sample17.html

Ventanas

- Abrir y Cerrar:

```
function openWin() {
    myWindow = window.open("", "myWindow",
"width=200,height=100");    // Opens a new window
}

function closeWin() {
    myWindow.close();    // Closes the new window
}
```

- File: Sample17.html

- Despliega nombre de la ventana:

```
var myWindow = window.open("", "MsgWindow", "width=200,height=100");
myWindow.document.write("<p>This window's name is: " + myWindow.name +
"</p>");
```

- File: Sample17.html

Ejercicios

- Despliega una ventana que apunte alguna pagina web:
  - de 400 x 500
  - que aparezca a 100,300
  - que no despliegue location
  - que no despliegue la barra de menu
  - que pueda cambiar de tamaño
  - que muestre los scrolls
  - que muestre estatus
  - que no despliegue el titulo
  - que no se vea la barra de herramientas

Continua en JS_3