Resumen JavaScript

Ginna Solano

¿Qué es JavaScript?

Es un lenguaje de programación orientado a objetos, es utilizado para crear eventos en páginas web.

Diferencia entre java y javaScript:

JavaScript:
Esta orientada a objetos. No se distingue entre tipos de objetos. La herencia es a través del mecanismo de prototipo, y las propiedades y métodos se puede añadir a cualquier objeto de forma dinámica. Los tipos de variables no se declaran.

Java:

Java es un lenguaje de programación basada en clases.
Los objetos se dividen en clases e instancias con toda herencia a través de la jerarquía de clases. Las clases y los casos no pueden tener propiedades o métodos añadidos dinámicamente. Los tipos de variables deben ser declaradas.

Que es scripting language?

Es un lengueje de programación que soporta scripts, es un lenguaje de control operando en alto nivel interpreta el codigo en vez de compilar asi automaticamente ejecuta las tareas.

Que es V8?

Es un motor de javascript utilizado en chrome.

En programación orientada a objetos:

Objeto: Instancia de una clase, contiene propiedades o atributos con comportamientos los cuales reaccionan a eventos.

Clase: Propiedades y comportamientos de un objeto.

Herencia: (Por ejemplo, herencia de la clase C a la clase D) es la facilidad mediante la cual la clase D hereda en ella cada uno de los atributos y operaciones de C.

Método: Es lo que el objeto puede hacer, puede producir cambios en las propiedades del objeto.

Evento: Es un suceso, la reacción que genera un objeto.

Atributos: Características de una clase.

Constructor: Es una función cuya misión es inicializar un objeto de una clase.

Definición

Alert: Es una función que se muestra en forma de ventana al usuario para dar un aviso.

Prompt: Al igual que el Alert se muestra en forma de ventana, pero permite al usuario ingresar información.

Eval(): Es utilizada para evaluar una cadena de código.

ToUppercase: Este método convierte una cadena de string a mayúsculas.

ToLowercase: Este método se utiliza para convertir una cadena de string a mayúsculas.

<noscript>: Esta etiqueta solo se mostrara al usuario cuando este no tiene soporte javaScript.

Window.onunload: El evento onunload ocurre una vez al perfil ha descargado (o la ventana del navegador se ha cerrado). Se produce cuando el usuario se desplaza fuera de la página (haciendo clic en un enlace, envía un formulario, cierre la ventana del navegador, etc.).

Undefined: Es cuando se declara la variable pero sin valor.

Null: El ***valor nulo*** es un valor simple que representa la referencia nula, vacía o inexistente.

Palabras reservadas:

Break:

Esta sentencia termina una acción.

Function:

Declaración de una función.

Return:

Finaliza una función devolviendo un valor.

Var:

Declaración de variable.

Case:

Nos permite tener varias opciones o simplificar un poco el código.

Delete:

Elimina una propiedad de un objeto.

Ejemplo:

```
var person = {firstName:"John", lastName:"Doe",
    age:50, eyeColor:"blue"};
    delete person.age;   // or delete person["age"];
```

If: Ejecuta una sentencia si una condición especificada es evaluada como verdadera.

Switch:

Permite tener varias opciones o simplificar un poco el código.

Void: Se utiliza para tomar decisiones en función de distintos estados o valores de una variable.

Catch: Permite manejar el error. Emitiendo código.

This: This hace referencia al objeto global. No se puede establecer por la asignación durante la ejecución, y puede ser diferente cada vez que la función se llama.

While:

Ejecuta una sentencia, hasta que la condición sea evaluada como verdadera.

In:

El operador in retorna true si la propiedad especificada está en el objeto especificado.

Debugger: Es como colocar un punto de interrupción en el código.

For: Se utiliza para repetir una secuencia de instrucciones sobre todo cuando se conoce la cantidad exacta de veces que se quiere ejecutar cierta instrucción.

New: Constructor de un objeto.

Typeoff: Devuelve una cadena que identifica el tipo de datos de una expresión. ej: booleano,number, string.

Futuras palabras reservadas:

Class
Enum
Export
Extends
Import
Super
Static

Implements
Interface
Package
Private
Protected
Public
Yield

Lista de eventos JavaScript:

Nombre con prefijo on

Relacionados con el ratón:

Onclick:
Click sobre un elemento.

Ondblclick:
Doble click sobre un elemento.

Onmousedown:
Se pulsa un botón del ratón sobre un elemento.

Onmouseenter:
El puntero del ratón entra en el área de un elemento.

Onmousemove:

El puntero del ratón se está moviendo sobre el área de un elemento.

Onmouseleave:

El puntero del ratón sale del área de un elemento.

Onmouseout:

El puntero del ratón sale fuera del área del elemento o fuera de uno de sus hijos

Onmouseup:

Un botón del ratón se libera estando sobre un elemento

Contextmenu:

Se pulsa el botón derecho del ratón (antes de que aparezca el menú contextual)

Relacionados con el teclado

Onkeydown:

El usuario tiene pulsada una tecla (para elementos de formulario y body)

Onkeypress:

El usuario pulsa una tecla (momento justo en que la pulsa) (para elementos de formulario y body)

Relacionados con cambios en el DOM

DOMAttrModified,
DOMCharacterDataModified,
DOMContentLoaded,
DOMElementNameChanged,
DOMNodeInserted,
DOMNodeInsertedIntoDocument,
DOMNodeRemoved,
DOMNodeRemovedFromDocument,
DOMSubtreeModified

Diferencia entre Window.onload y document ready:

Document ready se produce después de que el documento HTML se ha cargado, mientras que el onload evento se produce más tarde, cuando también se ha cargado todo el contenido (por ejemplo, imágenes).

El onload evento es un evento normal en el DOM, mientras que la ready de eventos es específica para jQuery.

El propósito de la ready de eventos es que debe ocurrir tan pronto como sea posible después de que el documento se ha cargado, por lo que el código que añade funcionalidad a los elementos de la página no tiene que esperar a que todo el contenido se cargue.

Otra diferencia entre los dos es que podemos tener más de un document.ready() función en una página web, pero sólo un onload función.

Guardar funciones en una variable.

Es posible porque en javaScript las funciones son de primer clase esto quiere decir que permite devolver funciones dentro de otra función y asignar funciones a variables. Esta variable seria de tipo function.

Objeto Date:

El objeto Date se utiliza para trabajar con fechas y horas.

Los objetos de la clase Date no tienen propiedades pero si un montón de métodos:

getDate() Devuelve el día del mes.

getDay() Devuelve el día de la semana.

getHours() Retorna la hora.

getMinutes() Devuelve los minutos.

getMonth() Devuelve el mes (atención al mes que empieza por 0).

getSeconds() Devuelve los segundos.

getTime() Devuelve los milisegundos transcurridos entre el día 1 de enero de 1970 y la fecha correspondiente al objeto al que se le pasa el mensaje.

getYear() Retorna el año, al que se le ha restado 1900. Por ejemplo, para el 1995 retorna 95, para el 2005 retorna 105. Este método está obsoleto en Netscape a partir de la versión 1.3 de Javascript y ahora se utiliza

getFullYear()

Retorna el año con todos los dígitos. Usar este método para estar seguros de que funcionará todo bien en fechas posteriores al año 2000.

Hay 4 formas de crear instancias de una fecha:

Ejemplo:

var d = new Date();

var d = new Date(*milliseconds*);

var d = new Date(*dateString*);

var d = new Date

(*year* , *month* , *day* , *hours* , *minutes* , *seconds* , *mil liseconds*)

toFixed:

Mantiene un número con los decimales que pongamos entre los paréntesis a su vez los redondea.

ejemplo:

```
var num = 35.678;
 num.toFixed(2)
alert(num);
```

Esto nos mostraría solo 2 decimales:

3.68

toPrecision()

Muestra un número en una longitud especificada.

Ejemplo:

```
var num = 13.3714;
num.toPrecision(5);
alert(num);
```

Esto nos mostraría:

13.371

toString();

Convierte un número en una cadena.

```
var num = 15;
var n = num.toString();
```

```
El resultado de n sera:
         15
```

Number();

Esta función convierte un objeto a un número que representa el valor del objeto.

Ejemplo:

```
var x1 = true;
var x2 = false;
var x3 = new Date();
var x4 = "999";
var x5 = "999 888";

var n =
Number(x1) + "<br>" +
Number(x2) + "<br>" +
Number(x3) + "<br>" +
Number(x4) + "<br>" +
Number(x5);
```

El resultado será:

```
1
0
1382704503079
999
NaN
```

Prototype:

Cada objeto posee un prototype.

Puede añadir propiedades o métodos a una clase existente mediante la adición de ellos al prototipo asociado con la función constructora para esa clase.

String:

Almacena una cadena, una cadena puede ser cualquier texto entre comillas dobles o simples.

Manejo de errores:

try : Código a intentar ejecutar .

catch: Bloque de código que se ejecuta si falla lo que esta dentro de try.

Finally: Se ejecuta sin importar si hay errores o no.

```
try {
  // sentencias a ejecutar que pueden generar excepciones
}
catch (e) {
  // De generarse alguna excepcion este es capturado
  // y puede ser accederse a traves de la variable e.
}
finally {
  // codigo que se ejecuta independientemente de si se
  // ejecuta el try o el catch o ambos.
}
```

throw: Lanza una excepción definida por el usuario. Contiene una serie de características, sintaxis:

```
throw{
    name: Nombre del error o excepcion.
    message: Descripción del error
  }
```