JavaScript y ECMAScript6

Tutorial para Aspirantes del Club de Informatica

Incluir JavaScript externo en HTML.

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Ejemplo de código JavaScript en el propio documento</title>
                        [se puede incluir como un archivo externo]
<script type="text/javascript" src="/js/codigo.js"></script>

</head>
<body>
<script type="text/javascript">
  alert("Un mensaje de prueba");
</script>
<p>Un párrafo de texto.</p>
</body>
</html>
```


Incluir JavaScript dentro de HTML.

codigo.js

```
// Comentario de una linea

alert("Un mensaje de prueba");

/* Los comentarios de varias líneas son muy
   útiles cuando se necesita incluir bastante
   información en los comentarios */
```

Incluir JavaScript en los elementos HTML

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Ejemplo de código JavaScript en el propio documento</title>
</head>

<body>
<p onclick="alert('Un mensaje de prueba')">Un párrafo de texto.</p>
</body>
</html>
```

Declarar e inicializar variables(Ambito o Escope="Locales y globales")

var numero_1 = 3;

let numero_2 = 1; //LET es el NEW "var" important

let resultado = let numero_1 + ler numero_2;

El nombre de una variable también se conoce como identificador y debe cumplir las siguientes normas: Sólo puede estar formado por letras, números y los símbolos $ (dólar) y _ (guión bajo). El primer carácter no puede ser un número.

Tipos de variables

Numéricas

```
let iva = 16;        // variable tipo entero
let total = 234.65;  // variable tipo decimal
```

Cadenas de texto

```
let mensaje = "Bienvenido a nuestro sitio web";
let letraSeleccionada = 'c';
```

Arreglos

```
let dias = ["Lunes", "Martes", "Miércoles", "Jueves",
    "Viernes", "Sábado", "Domingo"];
```

Booleanos

```
let verdadero = true;
let falso = false;
```

Palabras reservadas

son las palabras (en inglés) que se utilizan para construir las sentencias de JavaScript y que por tanto no pueden ser utilizadas libremente. Las palabras actualmente reservadas por JavaScript son: break, case, catch, continue, default, delete, do, else, finally, for, function, if, in, instanceof, new, return, switch, this, throw, try, typeof, var, void, while, with.

Operadores en "JavaScript = C++"

Asignacion "=" ;
Igualdad "=="
Incremento "++"
Asignacion e incremento "+="
Decremento "--"
Asignacion y decremento "-="

Operadores Logicos

Negacion "!" o "not"
operación lógica AND "&&"
operación lógica OR "||"

Operadores matematicos

suma "+"
resta "-"
multiplicacion "*"
division "/"
Modulo "%"

Operadores relacionales

Mayor que ">"
Menor que "<"
Mayor igual que ">="
Menor igual que "<="
Igual que "=="
Distinto "!="d

Estructuras de control de flujo en "JavaScript == C++"

IF

```
let mostrarMensaje = true;
if(mostrarMensaje) {
  alert("Hola Mundo");
}
```

ELSE

```
let mostrarMensaje = true;
if(mostrarMensaje) {
  alert("Hola Mundo");
}ELSE
{
alert("Fuck World");
}
```

FOR

```
let mensaje = "Hola, estoy dentro de
    un bucle";
for(let i = 0; i < 5; i++) {
  alert(mensaje);
}
```

¿Cuantas veces se muestra el mensaje?

FOR...IN

```
let dias = ["Lunes", "Martes",
    "Miércoles", "Jueves", "Viernes",
    "Sábado", "Domingo"];

for(i in dias) {
  alert(dias[i]);
}
```

Funciones y Scope o ambito de las variables (Locales y Globales).

Ambito Local

```
function creaMensaje() {
  let mensaje = “Mensaje de prueba”;
//mensaje es una variable local, lo
    que significa que solo puede ser
    usada dentro de la funcion.
}
creaMensaje();
//como la variable local no existe al
    llamar a la funcion alert,
    pasandole la variable mensaje,
    dira "undefined              "
alert(mensaje);
```

Ambito Global

```
let mensaje = “Mensaje de prueba”;
 function muestraMensaje() {
   alert(mensaje);
}
//como la variable mensaje esta
    declarada fuera de la funcion,
    como lo que significa que es una
    variable global
alert(mensaje);
```

Ejemplo de variables locales y globales.

```
let mensaje = "gana la de Fuera";
function muestraMensaje() {
  mensaje = "gana la de Dentro";
  alert(mensaje);
}

alert(mensaje);
muestraMensaje();
alert(mensaje);
```

Sentencias break y continue

BREAK

```
let cadena = "En un lugar de la Mancha de cuyo
      nombre no quiero acordarme...";
let letras = cadena.split("");
let resultado = "";

for(i in letras) {
  if(letras[i] == 'a') {
    break;
  }
  else {
    resultado += letras[i];
  }
}
alert(resultado);
// muestra "En un lug"
```

CONTINUE

```
let cadena = "En un lugar de la Mancha de cuyo
      nombre no quiero acordarme...";
let letras = cadena.split("");
let resultado = "";

for(i in letras) {
  if(letras[i] == 'a') {
    continue;
  }
  else {
    resultado += letras[i];
  }
}
alert(resultado);
// muestra "En un lugr de l Mnch de cuyo nombre
      no quiero cordrme..."
```

while, do...while.

WHILE

```
let resultado = 0;
let numero = 100;
let i = 0;

while(i <= numero) {
  resultado += i;
  i++;
}
 alert(resultado);
```

DO...WHILE

```
let resultado = 1;
let numero = 5;
do {
  resultado *= numero;  //
     resultado = resultado * numero
  numero--;
} while(numero > 0);

alert(resultado);
```

switch

Con IF.

```
if(numero == 5) {
  ...
}
else if(numero == 8) {
  ...
}
else if(numero == 20) {
  ...
}
else {
  ...
}
```

Con SWITCH

```
switch(numero) {
  case 5:
    ...
    break;
  case 8:
    ...
    break;
  case 20:
    ...
    break;
  default:
    ...
    break;
}
```

CLASES
del nuevo JavaScript ECMAScript 6 (Sale en Junio 2015)

```
class Persona {
  // constructor donde definir las variables que se reciben
        y guardarlas en el objeto usando this
  constructor(nombre, edad) {
    this.nombre = nombre;
    this.edad   = edad;
  }
 //getters y setters
get verNombre() {
    return this.nombre;
  }
  set nuevoNombre(nuevo) {
    this.nombre = nuevo;
  }
}

var Instancia= new Persona('Gilberta', 22);
Instancia.verNombre; //devuelve Gilberta
```

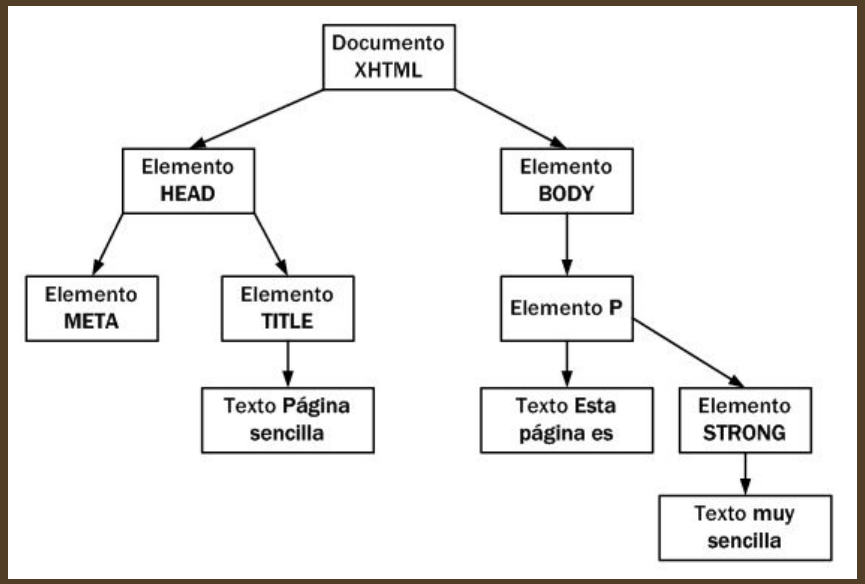
Herencia

```
class Desarrollador extends Persona {
  constructor(nombre, edad, cargo) {
    super(nombre, edad);
    this.cargo = cargo;
  }
  presentarse() { //funcion
    return super.presentarse() + ' y soy desarrollador ' + this.cargo;
  }
//getters y setters
get verNombre() {
    return this.nombre;
  }
  set nuevoNombre(nuevo) {
    this.nombre = nuevo;
  }
}
var instancia = new Persona('Ruben', 22);
instancia.verNombre; // devuelve Ruben
instancia.nuevoNombre('Daniel'); // cambia el valor de nombre a
        Daniel
instancia.verNombre; // devuelve Daniel
```

Árbol de nodos

```
<!DOCTYPE html PUBLIC "-//W3C//DTD
    XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
    transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type"
    content="text/html; charset=iso-8859-1" />
<title>Página sencilla</title>
</head>

<body>
<p>Esta página es <strong>muy
    sencilla</strong></p>
</body>
</html>
```



Tipos de nodos

La especificación completa de DOM define 12 tipos de nodos, aunque las páginas XHTML habituales se pueden manipular manejando solamente cuatro o cinco tipos de nodos:

Document, nodo raíz del que derivan todos los demás nodos del árbol.

Element, representa cada una de las etiquetas XHTML. Se trata del único nodo que puede contener atributos y el único del que pueden derivar otros nodos.

Attr, se define un nodo de este tipo para representar cada uno de los atributos de las etiquetas XHTML, es decir, uno por cada par atributo=valor.

Text, nodo que contiene el texto encerrado por una etiqueta XHTML.

Comment, representa los comentarios incluidos en la página XHTML.

Los otros tipos de nodos existentes que no se van a considerar son :

DocumentType,
CDataSection,
DocumentFragment,
Entity,
EntityReference,
ProcessingInstruction y Notation.

Acceso directo a los nodos

La función getElementsByTagName(nombreEtiqueta) obtiene todos los elementos de la página XHTML cuya etiqueta sea igual que el pará metro que se le pasa a la función.

```
let parrafos = document.getElementsByTagName("p");
```

La función getElementsByName() es similar a la anterior, pero en este caso se buscan los elementos cuyo atributo name sea igual al pará metro proporcionado. En el siguiente ejemplo, se obtiene directame nte el único párrafo con el nombre indicado:

```
let parrafoEspecial = document.getElementsByName("especial");
```

```
<p name="prueba">...</p>
<p name="especial">...</p>
<p>...</p>
```

Acceso directo a los nodos

getElementById() es la más utilizada cuando se desarrollan aplicaciones web dinámicas. Se trata de la función preferida para acceder directamente a un nodo y poder leer o modificar sus propiedades.

```
var cabecera = document.getElementById("cabecera");
```

```
<div id="cabecera">
  <a href="/" id="logo">...</a>
</div>
```

Creacion y eliminacion de NODOS

```
// Crear nodo de tipo Element
var parrafo =
    document.createElement("p");

// Crear nodo de tipo Text
var contenido =
    document.createTextNode("Hola
    Mundo!");

// Añadir el nodo Text como hijo del
    nodo Element
parrafo.appendChild(contenido);

// Añadir el nodo Element como hijo
    de la pagina
document.body.appendChild(parrafo);
```

solamente es necesario utilizar la función removeChild():

```
var parrafo =
    document.getElementById("provisi
    onal");
parrafo.parentNode.removeChild(parr
    afo);
```

... q

Acceso directo a los atributos XHTML

```
let enlace =
    document.getElementById("enlace
    ");
alert(enlace.href); // muestra
    http://www...com

<a id="enlace"
    href="http://www...com">Enlace</
    a>
```

```
let imagen =
    document.getElementById("image
    n");
alert(imagen.style.margin);

<img id="imagen" style="margin:0;
    border:0;" src="logo.png" />
```

Modelo básico de eventos

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>

Ejercicio.