APRENDERAPROGRAMAR.COM

CREAR OBJETOS JAVASCRIPT CON THIS Y NEW. EJEMPLOS PARA ENTENDER QUÉ SON LOS OBJETOS Y PARA QUÉ SIRVEN. (CU01143E)

Sección: Cursos
Categoría: Tutorial básico del programador web: JavaScript desde cero

Fecha revisión: 2029

Resumen: Entrega nº43 del Tutorial básico "JavaScript desde cero".

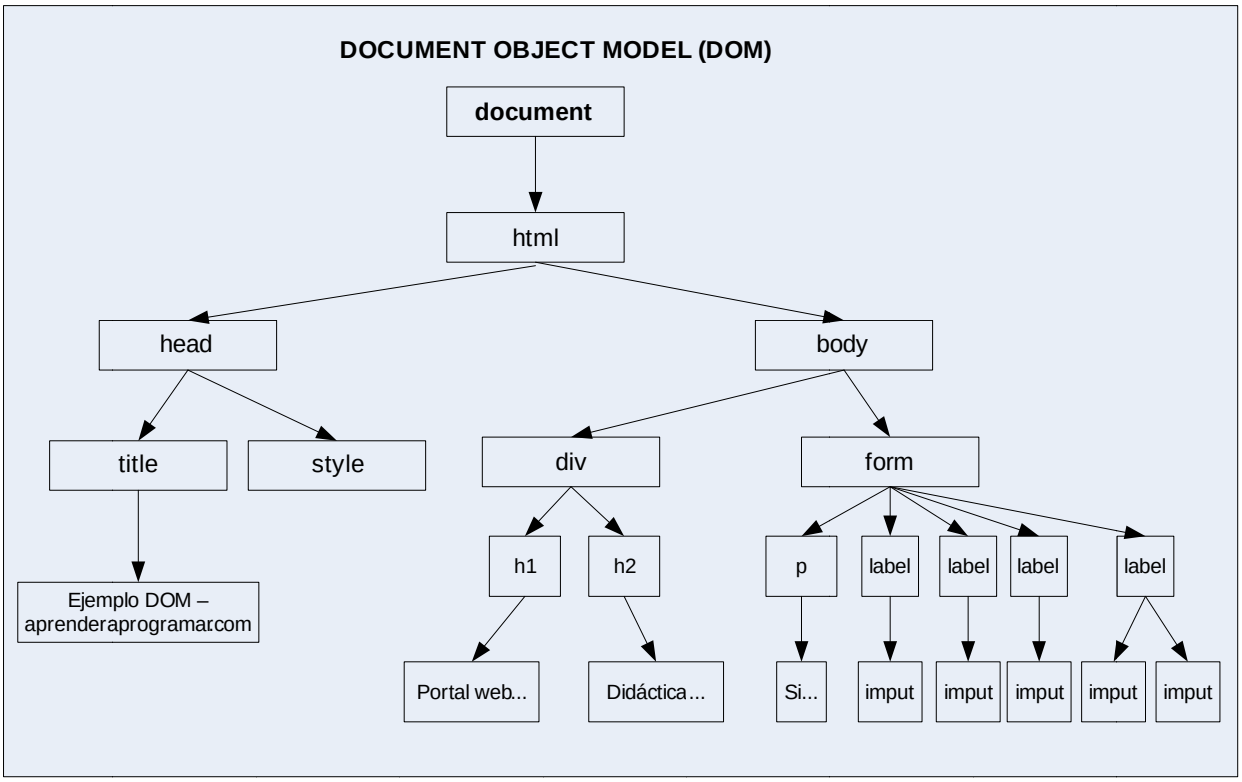
Autor: César Krall

OBJETOS JAVASCRIPT

En muchas ocasiones nos hemos referido a "el objeto document de JavaScript" o hemos dicho que un NodeList es un tipo de objeto JavaScript. Nos interesa ahora centrarnos en estudiar con mayor detenimiento el concepto de objeto JavaScript.

Un objeto es un tipo avanzado de datos que admite atributos (propiedades) y funciones (métodos). Tendremos que ir paso a paso para comprender esta definición.

Si recordamos cómo considera JavaScript que se estructura una página web a través del DOM, la representación que hacíamos era de este tipo:



Podemos señalar que el propio DOM lleva la palabra objeto contenida en sí mismo: Document **Object** Model. Esto ya nos indica que para JavaScript todos los elementos presentes en el DOM son objetos. Por ejemplo el elemento document es un objeto que representa la página web en su conjunto (el documento HTML) y a su vez consta de numerosos nodos que también son objetos. JavaScript utiliza los objetos como estructura de datos fundamental: prácticamente todo (casi todo) en JavaScript son objetos.

Un objeto puede tener valores asociados a los que denominamos propiedades o atributos. Por ejemplo en un nodo al que hayamos denominado nodoDiv el atributo id nos indica el valor de id en el documento html para la etiqueta asociada.

En el código HTML tendremos por ejemplo:

Y rescatamos el nodo así: var nodo = document.getElementById('cabecera');

Y luego nos referimos al atributo id del objeto nodo como nodo.id.

Un atributo podría ser una entidad simple, por ejemplo un atributo id con valor 'cabecera', pero también podría ser una entidad compleja, es decir, otro objeto. Por ejemplo la propiedad doctype del objeto document (que invocamos como document.doctype) nos devuelve otro objeto.

Un objeto puede tener métodos (funciones) asociados, que permiten realizar operaciones invocando dicho objeto. Por ejemplo getElementById es un método disponible para el objeto document.

Escribe este código, guárdalo como documento HTML y trata de razonar sobre lo que se visualiza.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html><head><title>Ejemplo aprenderaprogramar.com</title><meta charset="utf-8">
<script type="text/javascript">
function ejemploObjetos() {
var elDocType = document.doctype;
var resultadoEjecutarMetodo = document.getElementById('cabecera');
var msg = 'document es un objeto. Una de sus propiedades es doctype, puede valer por ejemplo: ' + elDocType + ' que es
otro objeto, ';
msg = msg +'es decir, una propiedad puede ser un objeto. La propiedad name del objeto doctype devuelve: ' +
elDocType.name;
msg = msg + ' y la propiedad publicId devuelve ' + elDocType.publicId + ' \n\n';
msg = msg + 'document como objeto tiene métodos, por ejemplo getElementById nos devuelve ';
msg = msg + 'el elemento (objeto) con id indicado, que a su vez tiene atributos y métodos. Por ';
msg = msg + 'ejemplo el div con id cabecera objeto ' + resultadoEjecutarMetodo + ' tiene atributo nodeName:
'+resultadoEjecutarMetodo.nodeName + '\n\n';
alert (msg);
}
</script>
</head>
<body>
<div id="cabecera">
<h2>Cursos aprenderaprogramar.com</h2>
<h3>Ejemplo funciones JavaScript</h3>
</div>
<div style="color:blue;" id ="pulsador" onclick="ejemploObjetos()"> Probar </div>
</body>
</html>
```

Fijate cómo los objetos cuando se tratan de mostrar por pantalla (en modo texto) devuelven un corchete seguido de la palabra object y el tipo de objeto. Por ejemplo [object DocumentType] ó [object HTMLDivElement]. En cambio las propiedades o atributos únicamente devuelven un texto, sin los corchetes ni la palabra object.

Nosotros podremos definir "moldes" de objetos con sus propiedades y con sus métodos, crear uno o varios objetos del tipo definido y hacerlos trabajar dentro de nuestro código para facilitar la tarea de la programación.

Pensemos en la ventaja que esto nos supone: consideremos que tenemos un array como una serie de números y queremos conocer su suma. Podríamos escribir algo como:

```
var suma = 0;
var numero = [3, 2, 9];
for (var i=0; i<numero.length; i++){
suma = suma + numero[i];
}
alert ('La suma de los números es: ' + suma);
```

Pero esto nos obliga a usar un bucle cada vez que queremos conocer la suma de los números contenidos en el array.

¿Y si definiéramos el array como un objeto al que le añadimos un método por el cual sea capaz de sumar automáticamente el valor de sus números y devolvérnoslo?

Esto sería bastante útil. El interés entonces de usar objetos está en que nos permiten combinar datos y funciones en una misma entidad. El objeto almacenará unos datos (por ejemplo los valores numéricos contenidos en un array) y tendrá definidos unos métodos o funciones internas que permiten realizar operaciones o procesos (por ejemplo obtener la suma de los valores numéricos).

DEFINIR TIPOS DE OBJETOS SIMPLES CON JAVASCRIPT

Empecemos por definir un objeto muy simple, algo que representará una cuenta bancaria. Nuestro molde, patrón o definición de tipo se llamará cuentaBancaria y cada vez que queramos crear una cuenta bancaria escribiremos algo así como *new cuentaBancaria*.

Hay varias formas de definir tipos de objetos y de crear objetos. Empezaremos usando esta sintaxis para definir un tipo de objeto:

```
function nombreDelTipoDeObjeto (par1, par2, ..., parN) {
this.nombrePropiedad1 = valorPropiedad1;
this.nombrePropiedad2 = valorPropiedad2;
this.método1 = function () { ... código .... }
this.método2 = function (param1, param2, ..., paramN) { ... código ...}
}
```

Donde par1, par2, ... , parN son los posibles parámetros necesarios para crear un objeto.

nombrePropiedad1, nombrePropiedad2, etc. son los atributos o propiedades (información) que porta el objeto.

método1, método2, etc. son los métodos o funciones que pueden ser invocadas a través del objeto. Estos métodos pueden no requerir parámetros o sí requerir parámetros. El método1 sería un ejemplo de método que no requiere parámetros y el método2 un ejemplo de método que sí requiere parámetros.

Para crear un objeto del tipo definido escribiremos:

```
var nombreInstanciaObjetoCreada = new nombreTipoObjeto (param1, param2, ..., paramN);
```

Para invocar una propiedad escribiremos:

```
nombreDelObjeto.nombreDeLaPropiedad
```

Para invocar un método escribiremos:

```
nombreDelObjeto.nombreDelMétodo (param1, param2, ..., paramN)
```

Donde param1, param2, paramN son los parámetros necesarios para que el método (función) se pueda ejecutar, si es que hay parámetros. Si no los hubiera, simplemente escribiremos los paréntesis de apertura y cierre.

Dado que para invocar propiedades y métodos se usa el nombre del objeto seguido de un punto y el nombre de la propiedad o método, se dice que las propiedades y métodos se invocan usando el operador punto.

Para comprender todo lo anterior escribe este código y guárdalo con extensión html:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Ejemplo aprenderaprogramar.com</title>
<meta charset="utf-8">
<script type="text/javascript">
function CuentaBancaria () {
this.nombreTitular = "Barack";
this.apellidosTitular = "Obama";
this.saldo = 600000;
this.mostrarDatos = function () {
        var msg = 'Los datos de la cuenta son Nombre: ' + this.nombreTitular;
        msg = msg + '; Apellidos: ' + this.apellidosTitular + '; Saldo: ' + this.saldo;
        alert(msg);
        }
}
```

```
function ejemploCreaObjetos() {
var cuenta1 = new CuentaBancaria();
msg = 'El saldo en la cuenta bancaria de Obama es: ' + cuenta1.saldo + ' dólares';
alert (msg);
cuenta1.mostrarDatos();
}
</script>
</head>
<body><div id="cabecera"><h2>Cursos aprenderaprogramar.com</h2><h3>Ejemplo funciones JavaScript</h3></div>
<div style="color:blue;" id ="pulsador" onclick="ejemploCreaObjetos()"> Probar </div>
</body>
</html>
```

Cuando pulsamos sobre el texto “Probar” se crea el objeto cuenta1 mediante la invocación de new cuentaBancaria(). Al crearse el objeto, se establecen sus propiedades, en este ejemplo nombreTitular es una propiedad y su contenido es “Barack”. También se establecen métodos, en este ejemplo el método mostrarDatos que permite visualizar la información que contiene el objeto.

Una vez creado el objeto ya se pueden invocar sus propiedades y sus métodos con la sintaxis de punto.

Pero este ejemplo es poco útil, porque cada vez que creáramos un objeto cuentaBancaria, éste objeto siempre tendría el mismo titular y el mismo saldo.

Normalmente nos interesará crear objetos que respeten el patrón o molde (el tipo definido) pero donde cada objeto porte unos datos diferentes. Para ello lo más habitual será pasar esos datos como parámetros cuando se crea el objeto. Escribe el código de este ejemplo y pruébalo:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html><head><title>Ejemplo aprenderaprogramar.com</title><meta charset="utf-8">
<script type="text/javascript">
function CuentaBancaria (datoTitular, datoApellidos, datoSaldo) {
this.nombreTitular = datoTitular;
this.apellidosTitular = datoApellidos;
this.saldo = datoSaldo;
this.mostrarDatos = function () {
        var msg = 'Los datos de la cuenta son Nombre: ' + this.nombreTitular;
        msg = msg + '; Apellidos: ' + this.apellidosTitular + '; Saldo: ' + this.saldo;
        alert(msg);
        }
}
function ejemploCreaObjetos() {
var cuenta1 = new CuentaBancaria('Barack', 'Obama', 600000);
cuenta1.mostrarDatos();
var cuenta2 = new CuentaBancaria('Vladimir', 'Putin', 900000);
cuenta2.mostrarDatos();
}
</script></head>
<body><div id="cabecera"><h2>Cursos aprenderaprogramar.com</h2><h3>Ejemplo funciones JavaScript</h3></div>
<div style="color:blue;" id ="pulsador" onclick="ejemploCreaObjetos()"> Probar </div>
</body></html>
```

Resulta un tanto paradójico que para crear un objeto se use function. Y además para crear los métodos dentro del objeto, también se usa function. Hablaremos de ello más adelante.

EJERCICIO

Crea un documento HTML (página web) donde exista un botón "Crear cuenta bancaria". Cuando el usuario pulse sobre el botón debe:

a) Pedirse al usuario un nombre de titular, apellidos de titular y saldo de la cuenta.

b) Crear un nuevo objeto cuentaBancaria (similar al que hemos creado en los ejemplos) que se inicializará con los datos facilitados por el usuario.

c) Mostrar un mensaje informando de que se ha creado la nueva cuenta bancaria y de los datos asociados a la cuenta bancaria creada.

Para comprobar si tus respuestas y código son correctos puedes consultar en los foros aprenderaprogramar.com.

Próxima entrega: CU01144E

Acceso al curso completo en aprenderaprogramar.com -- > Cursos, o en la dirección siguiente: http://aprenderaprogramar.com/index.php?option=com_content&view=category&id=78&Itemid=206