## Funciones constructoras e instancias de objeto

Sobre la base de los Objetos Literales en Javascript se establecen los demás tipos de objetos personalizados que puedes diseñar.

Las **funciones constructoras**, conocidas como *clases* en la mayoría de lenguajes de POO, permiten crear **instancias de objeto** en base a una estructura determinada, establecida por la propia función.

## Sintaxis de una función constructora

Las funciones constructoras están a medio camino entre las funciones usuales de Javascript y los objetos literales, con una sintaxis como la que sigue:

```
var NombreConstructora = function(valor1, valor2, valor_n) {
    this.identificador1 = valor1;
    this.identificador2 = valor2;
    this.identificador_n = valor_n;
}
```

O su equivalente:

```
function NombreConstructora(valor1, valor2, valor_n) {
    this.identificador1 = valor1;
    this.identificador2 = valor2;
    this.identificador_n = valor_n;
}
```

Pudiendo, naturalmente, contener no solo propiedades con valores, sino métodos con funciones asociadas que dotan de capacidades a sus instancias:

```
function NombreConstructora(valor1) {
    this.identificador1 = valor1;
    this.identificador2 = function() {
          instruccion1;
          instruccion2;
    }
}
```

La declaración nominal de una constructora comienza con una mayúscula, una convención sin implicación técnica que nos permitirá diferenciarla de aquellas funciones que no tienen una finalidad constructora.

Cada uno de los identificadores que contiene la constructora siguen el mismo principio que los objetos literales con la salvedad que les precede la palabra clave *this* y que reciben los valores a través de los parámetros de la misma función.

Esta diferenciación **aumenta ampliamente el grado de reusabilidad** de un sistema que, a diferencia de la forma que nos permitía en la anterior entrega crear diferentes facturas, esta vez define una arquitectura (**función constructora**) a través de la que podremos crear infinitas facturas de similar morfología (**instancias**).

## Declaración de una función constructora

Reproduzcamos el diseño de facturas que realizamos mediante objetos literales, pero a través de una función constructora.

Lejos de crear facturas con información literal, diseñaremos una arquitectura estándar de facturas que permitirá producir infinidad de facturas independientes:

```
function Factura(par1, par2, par3, par4, par5) {
      this.numero = par1;
      this.cliente = par2;
      this.divisa = par3;
      this.subtotal = par4;
      this.IVA = par5;
      this.total = function(){
       return this.subtotal + this.IVA;
     }
}
```

## Creación de instancias de objeto

La palabra clave *new* determina la creación de un nuevo objeto en la aplicación.

Dado que las instancias son objetos, nos valdremos de esa palabra para crear dos facturas independientes a través de la función constructora única que hemos declarado previamente:

```
var factura1 = new Factura(854, "Bar Colón", "eur", 659, 138.39);
var factura2 = new Factura(855, "Pub María", "eur", 48, 10.08);
```

Estas facturas son ya instancias de objetos en la aplicación.

## Funcionalidades de instancias de objeto

Estas dos instancias son objetos que deben su denominación de *instancia* a que han sido formuladas a partir de una función constructora.

Visualizarlas por dentro nos muestra que no son más que dos objetos literales independientes que hemos creado de forma mucho más eficiente:

```
factura1 = {
    numero: 854;
    cliente: "Bar Colón";
    divisa: "eur";
    subtotal: 659;
    IVA: 138.39;
    total: function(){
    return this.subtotal + this.IVA;
    }
}

factura2 = {
    numero: 855;
    cliente: "Pub María";
    divisa: "eur";
    subtotal: 48;
    IVA: 10.08;
    total: function(){
    return this.subtotal + this.IVA;
    }
}
```

En este punto se comprende que, a efectos de uso de estas instancias, continúan vigentes las metodologías y procedimientos que utilizamos en los objetos literales:

```
var numeroFactura = factura1.numero;
var monedaFactura = factura1.divisa;
var totalFactura = factura1.total();
console.log('La factura ' + numeroFactura + ' es de ' + totalFactura + ' eur.');

// La factura 855 es de 58.08 eur.
```