

# Objects in JavaScript

BY JAROSLAV MOHAPL

## Abstract

Creating objects, using them and customizing, is an integral part of Java scripting. This page provides some introductory facts and illustrates their application on the built-in Array object.

**Key Phrases** Constructor in JavaScript, array in JavaScript, sorting an array in JavaScript.

**Key Words** Array, constructor, JavaScript, object, new, sorted, syntax.

## Introduction

Object-oriented programming is a powerful tool for writing efficient and logically correct codes. Though JavaScript coding relies heavily on objects exposed to the compiler by the browser, the knowledge of creating your own objects and customizing the built-in ones is definitely useful. This page thus recalls creation and use of objects in JavaScript.

## How to Create and Use Objects

Calling the constructor function creates an object. The call allocates the memory necessary for storage of all object's methods and properties. The call is realized by applying the new operator to the constructor function on the right side of the assignment operator. The constructor is an ordinary JavaScript function with a name that agrees with the name of the object. The next example defines an object called BusinessCard:

```
//constructor
function BusinessCard(fn, sn, ph){
    this.firstname=fn;
```

```

    this.surname=sn;
    this.phone=ph;
    this.print=print;

    //method print
    function print(){
        document.write("First name: "+this.firstname+"<br/>");
        document.write("Surname: "+this.surname+"<br/>");
        document.write("Phone: "+this.phone+"<br/><br/>");
    }
}

```

The constructor has three properties: `firstname`, `surname`, and `phone`. They are defined using the self-reference `this` and initialized by means of the constructor's arguments. The `print` method displays the properties in the browser window. Notice that the `print` method is also linked to the object using the self-referencing `this`. An instance of the `BusinessCard` object can be created and the content printed as follows:

```

//new instance of the Business card
var card=new BusinessCard("Tom", "Sawyer", "555 8822");

//print content
card.print();

```

## Arrays

The next line of code creates a new array named `a`:

```

var a = new Array();

```

`Array()` is the built-in JavaScript array constructor. An attempt to use methods or properties of an object that has not been initialized using the constructor function results in a runtime error. Consequently, it is not possible to create a JavaScript array setting

```
a[0] = "Apples";  
a[1] = "Bananas";  
a[3] = "Peaches";
```

without calling the constructor first. The above array `a` can also be created by setting

```
var a = new Array("Apples","Bananas","Peaches");
```

A simple loop lists the array content:

```
for (i=0; i<a.length; i++){  
    document.write(a[i]+" ");  
}
```

The number of elements in the array is retrieved using `a.length`, where `length` is a variable storing the count. The loop listing the array content can be replaced by the `join()` method of the Array object. We recall that `join()` produces a comma-delimited string. The function also accepts a string argument, where the string specifies a delimiter.

### Example: A Sorted Array

The Array object contains a sort function. We demonstrate its use by creating a wrapper object for an array of business cards, which allows sorting by surname. The BusinessCards object was introduced earlier. The code for the wrapper looks like this:

```
// The card array object
//constructor
function cardArray(){
    this.cards = [];
    this.addCard=addCard;
    this.print=print;
    this.sort_by_sn=sort_by_sn;

    //add a card in the array method
    function addCard(fn,sn,phone){
        this.cards[this.cards.length]
            =new BusinessCard(fn,sn,phone);
    }

    //print method
    function print(){
        for (i=0; i<this.cards.length; i++){
            this.cards[i].print();
        }
    }

    //sort method
    function sort_by_sn(){
        this.cards.sort(f);
    }
}
```

```
//auxiliary function
function f(a,b){
    return (b.surname<=a.surname) ? 1 : (-1);
}
}
```

The sort function accepts an argument, which is a function with two arguments and returns one or minus one. The definition of this auxiliary function depends on the context. In our case, we want sort by the surname, but we could use as well any other property whose arguments admit ordering. An application of the cards array follows:

```
var a = new cardArray();
a.addCard("Ernest", "Hemingway", "555 8822");
a.addCard("Scott","Fitzgerald","555 8721");
a.addCard("Mark","Twain","555 3633");

a.sort_by_sn()
a.print();
```

## Summary

This page recalls how to create and use objects in JavaScript. It also shows how to use the Array object and its sort function.