**Constructor Functions**

This is the fifth and final article in JavaScript functions series. Earlier in the series we have covered –

Function Declaration

Function Expression

Anonymous Functions

Immediately Invoked Function Expression [IIFE]

Today we will focus on Constructor Functions.

**JavaScript without a class construct**

If you are an experienced C++/ Java / C# programmer you already know the importance of class construct in your program. Classes are composed of structural and behavioral elements like fields, properties, methods, events etc. and they are core part of the application. Now as I mentioned in the earlier articles in this series, current version of JavaScript language does not support class construct, but the future version [ECMAScript 6] will. Bu then, you must be thinking how it is even possible to create a complex program using JavaScript only and why anyone would do that? Trust me; I had similar question when I started digging into these concepts. So the purpose of this article is to provide some clarity around it and understand what the constructor functions are.

**Function as a class**

In general, when we compare two languages, we do it on the features it provides. For e.g. Java templates Vs C# generics, C# functions Vs JavaScript functions, dynamic nature of C# language and JavaScript language etc. If we don’t find any specific feature, for e.g. class in JavaScript, we might think that the language is not powerful enough to build large scale apps. However we really need to spend significant time to understand core concepts of these languages since they were designed for different purposes and each of the language has its pros and cons.

The most important part in JavaScript language is functions. They not only act as methods in your program, they can also be used to define a class construct which can have its internal fields, properties, method etc. Constructor function is one of the ways to use function as a class.

**Syntax**

A constructor is any function which is used as a constructor. The language doesn’t make a distinction. A function can be written to be used as a constructor or to be called as a normal function, or to be used either way. It can be define using function declaration or function expression syntax.

Below code snippet defines a class Customer using JavaScript function declaration syntax. It’s easy to understand that the constructor function accepts three arguments namely fname, lname and age which then instantiates the class properties.

A constructor function starting with new should always start with a capital letter. That’s the convention usually everyone follows to clearly identify which one is constructor function and which one is not.

var Customer = function (fname, lname, age) {

this.FirstName = fname;

this.LastName = lname;

this.Age = age;

}

**The new operator**

The new operator creates an instance of a user-defined object type or of one of the built-in object types that has a constructor function. In below defined code snippets, we are creating two instances of Customer type defined earlier.

var bob = new Customer("John","Smith", 50);

var mili = new Customer("Mili","Jha", 34);

You might be thinking how it is different than other programming languages as the syntax of creating an instance is exactly the same? Let’s understand what happens internally when the constructor function gets called using new operator.

When var mili = new Customer(“Mili”,”Jha”,34) gets called, JavaScript does following things.

1. **Creates empty object** - No magic here, JavaScript just creates a new empty object –

var mili = {};

1. **Set the constructor property of newly created object** - Each use defined object has one special property called as constructor, which is used to identify the type of the object using the instanceof operator

mili.constructor = Customer;

console.log(mili instanceof Customer); //true

1. **Set the prototype of newly created object** – Unlike functions in other programming languages like C++, Java or C#; functions in JavaScript can have its own properties and methods. All the JavaScript functions get a property called prototype, which his just an empty object. But once an object instance is created, it inherits all the properties of constructor’s prototype.

Customer.prototype.city = "London";

console.log(mili.city);

1. **Returns the object** – Finally constructor function itself gets called and this is set to the object we are constructing. It then initializes the values depending on function arguments and returns the newly constructed object.

So that was a quick overview of construction functions in JavaScript. I hope it was useful to you.