Basic JavaScript Part 7: Static Properties and Methods

January 19th, 2011

Here are the links to the previous installments:

1. Functions
2. Objects
3. Prototypes
4. Enforcing New on Constructor Functions
5. Hoisting
6. Automatic Semicolon Insertion

In this post, I'm going to quickly show you how to set up static properties and methods in JavaScript. This can come in handy for constant values and/or utility functions. C# and most other class-based languages have a special syntax for static members. Not so for JavaScript. Although JavaScript doesn't provide a dedicated syntax for static members, we're still able to get what we want by adding properties and methods to a constructor function. This is possible because functions in JavaScript are plain objects that can have properties and methods of their own.

Let's show some code.

```
function Podcast() {};

Podcast.FILE_EXTENSION = 'mp3';
Podcast.download = function(podcast) {
    console.log('Downloading ' + podcast + ' ...');
};

Podcast.prototype.play = function() {
    console.log('Playing this podcast ...');
};
```

Here we created a constructor function named *Podcast* with a static property called*FILE_EXTENSION* and a static method named *download*. We also added the instance method*play()* to the prototype of our constructor function.

Getting the value of a static property or invoking a static method is done as follows:

```
Podcast.FILE_EXTENSION;                // 'mp3'
Podcast.download('Astronomy cast');    // 'Downloading Astronomy cast ...'
```

When we create a *Podcast* object and invoke the instance method *play()*, we obviously get the expected outcome.

```
new Podcast().play();    // 'Playing this podcast ...'
```

But it's surely not possible to statically invoke an instance method like so:

```
Podcast.play();        // undefined
```

It's also not possible to invoke a static method on a *Podcast* instance object either.

```
new Podcast().download('Railscasts');    // undefined
```

There you go. Knowing how to provide static members in JavaScript might come in handy when you feel inspired of writing your own Math class in JavaScript :-).

Until next time.