Intro to new JavaScript features. Comparison between old and new standards.

Hristijan Petreski, Software Engineer @ **GrabIT**

Table of Contents

Variable scope and modules

var

The var statement declares a function-scoped or globally-scoped variable, optionally initializing it to a value.

let

The let declaration declares a block-scoped local variable, optionally initializing it to a value.

var example

```
var x = 1;

if (x === 1) {
  var x = 2;

  console.log("The value of x inside the if clause is:", x);
}

console.log("The value of x at the top of the file is:", x);
```

let example

```
let x = 1;

if (x === 1) {
  let x = 2;

  console.log("The value of x inside the if clause is:", x);
}

console.log("The value of x at the top of the file is:", x);
```

var vs let example

```
var x = 6;
let y = 4;

function getValueX() {
  var x = 9;
  console.log("getValueX():", x);
}

function getValueY() {
  let y = 2;
  console.log("getValueY():", y);
}
```

```
getValueX();
console.log("Value for X:", x);
getValueY();
console.log("Value for Y:", x);
```

JavaScript Modules

JavaScript programs started off pretty small — most of its usage in the early days was to do isolated scripting tasks, providing a bit of interactivity to your web pages where needed, so large scripts were generally not needed. Fast forward a few years and we now have complete applications being run in browsers with a lot of JavaScript, as well as JavaScript being used in other contexts (Node.js, for example).

Functions and arrow functions

function vs arrow function syntax

```
function name() {
  // statements
}

function name(param) {
  // statements
}

function name(param, paramN) {
  // statements
}
```

```
param => expression

(param) => expression

(param1, paramN) => expression

param => {
  statements
}

(param1, paramN) => {
  statements
}
```

Useful **Array** methods

Methods

Array.prototype.forEach()

The forEach() method executes a provided function once for each array element.

Array.prototype.find()

The find() method returns the first element in the array that satisfies the provided testing function.

Array.prototype.filter()

The filter() method filters down to the elements that pass the test implemented by the provided function.

Array.prototype.map()

The map() method creates a new array populated with the results of calling a provided function.

Array.prototype.reduce()

The reduce() method reduces an array down to a single value based on a provided “reducer” function.

Useful **Object** methods

Methods

Object.assign()

The Object.assign() method copies all own properties from one or more objects to a target object.

Object.keys()

The Object.keys() method returns an array of a given object's own string-keyed property names.

Object.values()

The Object.values() method returns an array of a given object's own string-keyed property values.

Object.entries()

The Object.entries() method returns an array of a given object's own string-keyed property key-val pairs.

Object.fromEntries()

The Object.fromEntries() method transforms a list of key-value pairs into an object.

Closures and anonymous functions

Closures

A closure is the combination of a function bundled together (enclosed) with references to its surrounding state (the lexical environment). In other words, a closure gives you access to an outer function's scope from an inner function. In JavaScript, closures are created every time a function is created, at function creation time.

Learning Resources

Documentation and comparisons

https://developer.mozilla.org/en-US/

http://es6-features.org/

Courses

https://frontendmasters.com/teachers/kyle-simpson/