Course Outline | JavaScript ES6

2 day(s)

Overview

The demands on the JavaScript language have increased dramatically over the last few years as developers are tasked with writing more and more complex scripts, using JavaScript, AngularJS, NodeJS and many more.

ECMAScript 6 completed in 2015 and formally dubbed "ECMAScript 2015." Takes JavaScript programming to new levels many developers have been asking for.

The features vary widely from completely new objects and patterns to syntax changes to new methods on existing objects. The exciting thing about ECMAScript 6 is that all of its changes are geared toward solving problems that developers actually face.

This course takes existing JavaScript developers through the new powerful capabilities of ECMAScript 6.

Prerequisites

Experience of developing JavaScript web pages/applications is required.

Course Outline

1 - Introduction

- The Road to ECMAScript 6
- About This Course

2 - Block Bindings

- Var Declarations and Hoisting
- Block-Level Declarations
- Block Binding in Loops
- Global Block Bindings
- Emerging Best Practices for Block Bindings
- Strings and Regular Expressions
- Better Unicode Support
- Other String Changes
- Other Regular Expression Changes
- Template Literals

3 - Functions

- Functions with Default Parameter Values
- Working with Unnamed Parameters
- Increased Capabilities of the Function Constructor
- The Spread Operator
- ECMAScript 6's name Property
- Clarifying the Dual Purpose of Functions
- Block-Level Functions
- Arrow Functions
- Tail Call Optimization
- Expanded Object Functionality
- Object Categories
- Object Literal Syntax Extensions
- New Methods
- Duplicate Object Literal Properties
- Own Property Enumeration Order
- More Powerful Prototypes
- A Formal Method Definition

4 - Destructuring for Easier Data Access

- Why is Destructuring Useful?
- Object Destructuring
- Array Destructuring
- Mixed Destructuring
- Destructured Parameters

5 - Symbols and Symbol Properties

- Creating Symbols
- Using Symbols
- Sharing Symbols
- Symbol Coercion
- Retrieving Symbol Properties
- Exposing Internal Operations with Well-Known Symbols

6 - Sets and Maps

- Sets and Maps in ECMAScript 5
- Problems with Workarounds
- Sets in ECMAScript 6
- Maps in ECMAScript 6

7 - Iterators and Generators

- The Loop Problem
- What are Iterators?
- What Are Generators?
- Iterables and for-of
- Built-in Iterators
- The Spread Operator and Non-Array Iterables
- Advanced Iterator Functionality
- Asynchronous Task Running

8 - Introducing JavaScript Classes

- Class-Like Structures in ECMAScript 5
- Class Declarations
- Class Expressions
- Classes as First-Class Citizens
- Accessor Properties
- Computed Member Names
- Generator Methods
- Static Members
- Inheritance with Derived Classes
- Using new.target in Class Constructors

9 - Improved Array Capabilities

- Creating Arrays
- New Methods on All Arrays
- Typed Arrays
- Similarities Between Typed and Regular Arrays
- Differences Between Typed and Regular Arrays

10 - Promises and Asynchronous Programming

- Asynchronous Programming Background
- Promise Basics
- Global Promise Rejection Handling

- Chaining Promises
- Responding to Multiple Promises
- Inheriting from Promises

11 - Encapsulating Code With Modules

- What are Modules?
- Basic Exporting
- Basic Importing
- Renaming Exports and Imports
- Default Values in Modules
- Re-exporting a Binding
- Importing Without Bindings
- Loading Modules