ECMA Script 6 JS

By : Vijay Shivakumar

www.dotjs.in

JS

ECMA Script 6

About the Program

Published on June 17, 2015

Formally "ECMA Script 2015"

6 years! from ES5

http://dotjs.in

JS ECMA Script 6
ECMA Timeline

ECMAScript 1 (1997)
variables functions control flow
ECMAScript 2 (1998)
minor changes
ECMAScript 3 (1999)
regex better error and string handling and XML
ECMAScript 4 (Never Released)
ECMAScript 5 (2009)
JSON support, array methods and strict mode, & loads of utilities
ECMAScript 5.1 (2011)
Mostly was standardized version of ES5
ECMAScript 6 (2015)
block scope, arrow functions, template literals, & classes
ECMAScript 7 (2016)
more array methods, promises
ECMAScript 8 (2017)
async/ await and new object methods
ECMAScript 9 (2018)
rest/spread and promise finally
ECMAScript 10 (2019)
array flat, flatMap and object methods
ECMAScript 11 (2020)
support for large integers, allSettled for promises
ECMAScript 12 (2021)
any for promises and replaceAll for strings
ECMAScript 13 (2022)
improvements and standardization continues
ECMAScript 14 (2023)
many features including typed array, array and objects methods

http://dotjs.in

JS

ECMA Script 6

Typical flow of a JavaScript in browser

JavaScript Browser

http://dotjs.in

JS

ECMA Script 6

Why ES6 ?

Modern syntax fixing pitfalls

Better support for large applications

No (or few) breaking changes

http://dotjs.in

JS

ECMA Script 6

Browser Support…

Modern browsers and io.js (node.js) support half of ES6 features.

Safari 9 (WebKit) will support many ES6 features soon.

Except for IE11…

Check http://www.caniuse.com

JS

ECMA Script 6

Libraries to bridge the support now…

BabelJs : https://babeljs.io/

Traceur : https://github.com/google/traceur-compiler

JS ECMA Script 6

What's new in ES6 ?`

Block Scope (let/const)
Destructuring
Functional Programming
- Default Params
- Rest parameters
- Spread Syntax
- Arrow Function (Fat Arrow Function)

Array Methods
Object & Extended Object
Iterator
Generator
Template Strings
Promise
Classes
Modules

http://dotjs.in

JS

ECMA Script 6

What's new in ES7 ?

Exponentiation Operator (**) ES7
Async / Await ES7 // deal with functions that may take time to execute
Array.includes() ES7 // to check if a value exists in an array
Object.entries() ES7 // returns object's key value pairs
Object.values()   ES7 // returns object's values
Optional Chaining ES7

http://dotjs.in

JS

ECMA Script 6

Few Array methods in ES5

forEach()
map()
filter()
every()
some()
reduce()
reduceRight()
indexOf(searchTerm, fromIndex)
lastIndexOf()

http://dotjs.in

JS

ECMA Script 6

Promises

```
loginUser(
    userinfo,
      function (result) {
                if(result){
                failedToLogin(
                function (result) {
                                forwardToSignup(
                                nestedResult,
                                function (deepNestedResult) {
                                                console.log("you were not logged in because ","deepNestedResult");
                                },
                                errorHandler
                                );
                },
                errorHandler
                );
    },
    errorHandler
  );
```

http://dotjs.in

JS ECMA Script 6

Thank you

please forward your comments and feedback

vijay.shivu@gmail.com

http://dotjs.in

JS

ECMA Script 6

Block Scope in ES6

http://dotjs.in

JS

ECMA Script 6

Block Scope in ES6

http://dotjs.in

JS

ECMA Script 6

Block Scope in ES6

VAR

- Globally Available in the function which it is declared
- Hoisted
- Names can be declared many times

LET / CONST

- Only available in the block in which it is declared
- Not Hoisted
- Names can only be declared once per block

JS

ECMA Script 6

Generics in ES6

Generics are templates that allow the same function to work with different data types.

Good when we create reusable code that preserve the data type that go in and out of them.

You can use data type markers to denote a data type and refer those markers to represent the same data type in side a function / program

---

```
function getType<T>(value:T):string{
    return typeof (value);
};
getType("hello there") ; // says string
getType(true) ;// says boolean
```

http://dotjs.in

JS

ECMA Script 6

Block Scope in ES6

VAR

- Globally Available in the function which it is declared
- Hoisted
- Names can be declared many times

LET / CONST

- Only available in the block in which it is declared
- Not Hoisted
- Names can only be declared once per block

JS

ECMA Script 6

Generics in ES6

Generics are templates that allow the same function to work with different data types.

Good when we create reusable code that preserve the data type that go in and out of them.

You can use data type markers to denote a data type and refer those markers to represent the same data type in side a function / program

---

```
function getType<T>(value:T):string{
    return typeof (value);
};
getType("hello there") ; // says string
getType(true) ;// says boolean
```

http://dotjs.in

JS

ECMA Script 6

About me

Vijay Shivakumar

Designer | Developer | Trainer

Training on web and Adobe products from past 10+ years

http://dotjs.in