Dory Zidon
Oct 2016
doryz@coreteam.io
Linkedin: http://goo.gl/39r2py
Facebook: https://www.facebook.com/doryzi
https://www.facebook.com/CoreTeamIo/
https://www.coreteam.io

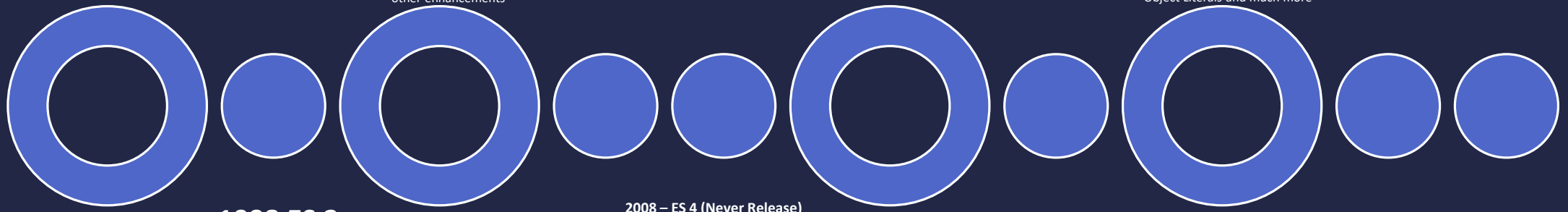
JavaScript

PAST, CURRENT AND FUTURE

JavaScript - Soap Opera History

- Prototype developed by Brendan Eich (Netscape) in 10 days
- JavaScript was initially called Mocha then LiveScript.
- Result of a partnership with Sun, against Microsoft
- Released in Netscape Navigator 2.0, in Sep 1995.
- Introduced as server side technology in Dec 1995.
- Microsoft, Google, Sun all tried to kill JavaScript yet it continues to live.

ECMAScript (ES) Standards



1997 ES 1

Netscape delivered JavaScript to Ecma International for standardization and the work on the specification

1998 ES 2

Editorial changes to keep the specification fully aligned with ISO/IEC 16262 international standard

1999 –ES 3

Baseline for modern JavaScript. Added regular expressions, better string handling, new control statements, try/catch exception handling, tighter definition of errors, formatting for numeric output and other enhancements

2008 – ES 4 (Never Release)

Disagreements lead to abandonment Many features we see in ES6 (classes, generators, iterators, destructing assignments and more)

2009 – ES 5 (3.1)

Yahoo, Microsoft, Google, and other 4th edition dissenters form community and get agreement for less complex version –JSON support, get /set, remove ambiguities, etc. New features Harmony Codename

2015 – ES 6

Significant update – Promises, Generators, Classes, Modules Arrow Functions, Blocked – Scope Constructs, Destructing Assignment, Template Literals, Enhanced Object Literals and much more

2016 – ES 7

Array.prototype.include,
** Exponentiation operator

ES.Next

Move to much Quicker Releases, more often..

Things like async functions, trailing commas, improved global and others.

About Us – Making Geeks Happy!!

- Work with Customers – New Languages / US Customers
- Work on own projects!
- Focus on happiness
- Control life – work balance
- Mix of remote + on-site
- Work with very smart people
- Work on projects you like
- Control your future and freedom

JOIN US TODAY: doryz@coreteam.io

Linkedin: http://goo.gl/39r2py

Facebook: https://www.facebook.com/CoreTeamIo/ | https://www.facebook.com/doryzi

https://www.coreteam.io

Some ES6 Features

- Default Parameters
- Template Literals
- Multiline Strings
- Arrow Functions
- Scoping
- Map / Set
- Classes
- Modules
- Promises
- Generators
- Destructing Assignment

Default Parameters

ECMAScript 5

```
var link = function (height, color, url) {
    var height = height || 50;
    var color = color || 'red‘;
    var url = url || 'https://coreteam.io‘;
            ...
}
```

ECMAScript 6

```
var link = function(height=50, color='red', url = 'http://coreteam.io') {
    ...
}
```

Template Literals

ECMAScript 5

```
var name = 'Your name is ' + first + ' ' + last + '.';
var url = 'http://localhost:3000/api/messages/' + id;
```

ECMAScript 6

```
var name = `Your name is ${first} ${last}.`
var url = `http://localhost:3000/api/messages/${id}`
```

Multiline Strings

ECMAScript 5

```
var roadPoem = 'Then took the other, as just as fair,\n\t'
    + 'And having perhaps the better claim\n\t'
    + 'Because it was grassy and wanted wear,\n\t'
    + 'Though as for that the passing there\n\t'
    + 'Had worn them really about the same,\n\t‘;

var fourAgreements = 'You have the right to be you.\n\
    You can only be you when you do your best.‘;
```

ECMAScript 6

```
var roadPoem = `Then took the other, as just as fair,
    And having perhaps the better claim
    Because it was grassy and wanted wear,
    Though as for that the passing there
    Had worn them really about the same,`

var fourAgreements = `You have the right to be you.
    You can only be you when you do your best.`
```

Arrow Functions

ECMAScript 5

```
var _this = this;
$('.btn').click(function(event){
    _this.sendData()
});
// or
$('.btn').click(function(event){
    this.sendData()
}.bind(this));
```

ECMAScript 6

```
$('.btn').click(event =>{
    return this.sendData()
});
```

Arrow Functions

ECMAScript 5

```
odds  = evens.map(function (v) { return v + 1; });
pairs = evens.map(function (v) { return { even: v, odd: v + 1 }; });
nums  = evens.map(function (v, i) { return v + i; });
```

ECMAScript 6

```
odds  = evens.map(v => v + 1)
pairs = evens.map(v => ({ even: v, odd: v + 1 }))
nums  = evens.map((v, i) => v + i)
```

Scoping

ECMAScript 5

```
function calculateTotalAmount (vip) {
    var amount = 0;
    if (vip) {
        var amount = 1;  // first amount is 1
    }
    { // more crazy blocks!
        var amount = 100;  // first amount is 100
        {
            var amount = 1000;  //first amount is 1000
        }
    }
    return amount;
}
console.log(calculateTotalAmount(true);
```

ECMAScript 6

```
function calculateTotalAmount (vip) {
    var amount = 0 // probably should also be let, but you can mix var and let
    if (vip) {
        let amount = 1;// first amount is still 0
    }
    { // more crazy blocks!
        let amount = 100;// first amount is still 0
        {
            let amount = 1000;// first amount is still 0
        }
    }
    return amount
}
console.log(calculateTotalAmount(true))
```

Scoping

ECMAScript 6

```
// valid assignment
const foo = {};
foo.bar = 42;
console.log(foo.bar);
// → 42
```

```
//invalid assignment
const foo = 27;
// Any of the following uncommented lines throws an exception.
// Assignment operators:
foo = 42;
foo += 42;
foo /= 42;
// All bit operations:
foo <<= 0b101010;
foo >>= 0b101010;
foo |= 0b101010;
// Unary `--` and `++`, prefix and postfix:
--foo;
foo++;
```

```
//making an object with immutable values
const foo = Object.freeze({
    'bar': 27
});
foo.bar = 42; // throws a TypeError exception in strict mode; // silently fails in sloppy mode
console.log(foo.bar);
// → 27
```

Classes - Definition

ECMAScript 5

```
var Shape = function (id, x, y) {
    this.id = id;
    this.move(x, y);
};
Shape.prototype.move = function (x, y) {
    this.x = x;
    this.y = y;
};
```

ECMAScript 6

```
class Shape {
    constructor (id, x, y) {
        this.id = id
        this.move(x, y)
    }
    move (x, y) {
        this.x = x
        this.y = y
    }
}
```

Set

ECMAScript 5

```
var s = {};
s["hello"] = true; s["goodbye"] = true; s["hello"] = true;
Object.keys(s).length === 2;
s["hello"] === true;
for (var key in s) // arbitrary order
    if (s.hasOwnProperty(key))
        console.log(s[key]);
```

ECMAScript 6

```
let s = new Set()
s.add("hello").add("goodbye").add("hello")
s.size === 2
s.has("hello") === true
for (let key of s.values()) // insertion order
    console.log(key)
```

Map

ECMAScript 5

```
var m = {};
m["hello"] = 42;
// no equivalent in ES5
// no equivalent in ES5
Object.keys(m).length === 2;
for (key in m) {
    if (m.hasOwnProperty(key)) {
        var val = m[key];
        console.log(key + " = " + val);
    }
}
```

ECMAScript 6

```
let m = new Map()
m.set("hello", 42)
m.set(s, 34)
m.get(s) === 34
m.size === 2
for (let [ key, val ] of m.entries())
    console.log(key + " = " + val)
```

Classes - Inheritance

ECMAScript 5

```
var Rectangle = function (id, x, y, width, height) {
    Shape.call(this, id, x, y);
    this.width  = width;
    this.height = height;
};
Rectangle.prototype = Object.create(Shape.prototype);
Rectangle.prototype.constructor = Rectangle;
var Circle = function (id, x, y, radius) {
    Shape.call(this, id, x, y);
    this.radius = radius;
};
Circle.prototype = Object.create(Shape.prototype);
Circle.prototype.constructor = Circle;
```

ECMAScript 6

```
class Rectangle extends Shape {
    constructor (id, x, y, width, height) {
        super(id, x, y)
        this.width  = width
        this.height = height
    }
}
class Circle extends Shape {
    constructor (id, x, y, radius) {
        super(id, x, y)
        this.radius = radius
    }
}
```

Classes – Static Members

ECMAScript 5

```
var Rectangle = function (id, x, y, width, height) {
    …
};
Rectangle.defaultRectangle = function () {
    return new Rectangle("default", 0, 0, 100, 100);
};
var defRectangle = Rectangle.defaultRectangle();
```

ECMAScript 6

```
class Rectangle extends Shape {
    …
    static defaultRectangle () {
        return new Rectangle("default", 0, 0, 100, 100)
    }
}
var defRectangle = Rectangle.defaultRectangle();
```

Modules

ECMAScript 5 (using common.js, the base for node modules today)

```
// hello.js
module.exports = function() {}

// main.js
var helloWorld = require('./hello-world');
var anotherFunction = require('./hello-world');

helloWorld();
console.log(helloWorld === anotherFunction);
```

ECMAScript 6

```
// hello-world.js
export default function() {}

// main.js
import helloWorld from './hello-world';
import anotherFunction from './hello-world';

helloWorld();
console.log(helloWorld === anotherFunction);
```

Promises

ECMAScript 5

```
setTimeout(function(){
    console.log('Yay!');
    setTimeout(function(){
        console.log('Wheeyee!');
    }, 1000)
}, 1000);
```

ECMAScript 6

```
const wait1000 =  ()=> new Promise((resolve, reject)=> {setTimeout(resolve, 1000)})

wait1000()
.then(function() {
    console.log('Yay!')
    return wait1000()
})
.then(function() {
    console.log('Wheeyee!')
})
.catch (function (err) {
  //handle failure, or exception thrown anywhere in the promise chain!
})
```

Generators

ECMAScript 5

```
fs.readFile('blog_post_template.html', function(err, tpContent){
    if (err){
        res.end(err.message);
        return;
    }
    fs.readFile('my_blog_post.md', function(err, mdContent){
        if (err){
            res.end(err.message);
            return;
        }
        res.end(template(tpContent, markdown(String(mdContent))));
    });
});
```

ECMAScript 6

```
try{
    var tpContent = yield readFile('blog_post_template.html');
    var mdContent = yield readFile('my_blog_post.md');
    res.end(template(tpContent, markdown(String(mdContent))));
}catch(e){
    res.end(e.message);
}
```

Destructing Assignment

ECMAScript 5 (using common.js, the base for node modules today)

```
//array examples
var list = [ 1, 2, 3 ];
var a = list[0], b = list[2];
var tmp = a; a = b; b = tmp;

//object examples
var tmp = getUserInfo();
var name  = tmp.name;
var address = tmp.address;
var email = tmp.email;
```

ECMAScript 6

```
//array examples
var list = [ 1, 2, 3 ]
var [ a, , b ] = list
    [ b, a ] = [ a, b ]

//object examples
var { name, address, email } = getUserInfo()
```

Q &A

doryz@coreteam.io | Linkedin: http://goo.gl/39r2py | Facebook: https://www.facebook.com/CoreTeamIo/
+359 899930786
https://coreteam.io