

Les avantages de l'es6

MEETUP 2017, Javascript, apéro web and co.

Franck WANG

13 janvier 2017

- 1 Introduction
- 2 Nouveautés de l'ES6
- 3 Création d'un projet en ES6
- 4 Conclusion
- 5 Références

Introduction

Intérêt de l'ES6

L'ECMAScript 6: dernière grosse mise à jour du JS

- ES6 approuvé en juin 2015
- Déjà implémenté en majeure partie dans Chrome / Firefox
- Très gros changements
- Beaucoup de nouvelles features

Objectif

- Éviter de dépendre de Framework (jquery) / autre langage (coffeeScript)
- Prototypage → orienté objet
- Futures release plus petites mais plus régulière

Nouveautés de l'ES6

Classe (ES5)

```
var Rectangle = function (id, x, y, width, height) {  
    Shape.call(this, id, x, y);  
    ...  
};  
  
Rectangle.prototype.toString = function () {  
    return "Rect " + Shape.prototype.toString.call(this);  
};
```

Classe (ES6)

```
class Rectangle extends Shape {  
  constructor (id, x, y, width, height) {  
    super(id, x, y)  
    ...  
  }  
  toString () {  
    return "Rectangle > " + super.toString()  
  }  
}
```

Arrow function

// ES5

```
var self = this;  
this.nums.forEach(function (v) {  
    if (v % 5 === 0) self.fives.push(v);  
});
```

// ES6

```
odds  = evens.map(v => v + 1)  
  
this.nums.forEach((v) => {  
    if (v % 5 === 0) this.fives.push(v);  
});
```


let vs var

let fonctionne comme **var** mais reste défini que localement dans chaque bloc.

```
let a, b, n;
...
for (let i = 0; i < n; i++) {
    let x = a[i]
    ...
}
for (let i = 0; i < n; i++) {
    let y = b[i]
    ...
}
// "i", "y", "x" n'existent pas ici
```

Import export

Permet d'éviter de travailler avec des variables globales.

```
// lib/math.js  
var pi = 3.141593  
export function sum (x, y) { return x + y }  
export pi
```

```
// someApp.js  
import * as math from "lib/math"  
console.log("2pi = " + math.sum(math.pi, math.pi))
```

```
// otherApp.js  
import { sum, pi } from "lib/math"  
console.log("2pi = " + sum(pi, pi))
```

For of

// Itération sur les valeurs

```
for (let n of [1, 2, 3, 4, 5]) {  
    if (n > 1000) break  
    console.log(n)  
}
```

// Itération sur les indices

```
let a = [1, 3, 4], b = [3, 4, 5];  
for (let i in a) {  
    console.log(a[i]+b[i])  
}
```

ES7: Object.entries, Object.values

Promesse

```
let fetchPromised = (url, timeout) => {  
  return new Promise((resolve, reject) => {  
    fetchAsync(url, timeout, resolve, reject)  
  })  
}  
  
Promise.all([  
  fetchPromised("http://backend/foo.txt", 500),  
  fetchPromised("http://backend/bar.txt", 500)  
]).then((data) => {  
  let [ foo, bar ] = data  
  console.log('success: foo=${foo} bar=${bar} baz=${baz}')  
}, (err) => {  
  console.log('error: ${err}')})
```

Autres

- Map: dictionnaire
- Set: Ensemble
- const
- Paramètre par défaut: `function f (x, y = 7) {...}`
- Array Matching: `var [a, , b] = [0, 1, 2]`
- Opérateur de décomposition: `function f (x, ...a) {...}`

Création d'un projet en ES6

npm / browserify / babel

Package.json

```
"scripts": {  
  "watch": "watchify index.js -o bundle.js -v --debug",  
  "build": "browserify index.js -v  
    -t [ babelify --presets [ es2015 ] ]  
    -o bundle.js"  
},  
"devDependencies": {  
  "babel-preset-es2015": "^6.6.0",  
  "babelify": "^7.2.0",  
  "browserify": "^13.0.0",  
  "watchify": "^3.7.0",  
}
```

script npm

npm run watch:

- “compile” le code a chaque fois qu’un fichier est sauvegardé.
- option `-debug` pour avoir l’arborescence des fichiers dans la console
- sans babel pour réduire le temps de “compilation”

npm run build

- compilation complète pour prod
- avec babel pour rétro-compatibilité
- sans `-debug`
- [bonus] ajouter uglifyjs

Conclusion

Conclusion

jQuery est devenu inutile ? NON

- bootstrap
- multitudes de plugins (datepicker, datatable. . .)

Avantages :

- pas à gérer la rétro-compatibilités pour les navigateurs
- syntaxe plus simple
- dépendre de moins de Framework

Références

Références

- Documentation + support navigateur : MDN
- Disponibilité :
`http://kangax.github.io/compat-table/es6/`
- Différences ES5 / ES6 : `http://es6-features.org`