ES6 In Modern Development

By

汤桂川 @广发证券

About me

QCon 全球软件开发大会

汤桂川

资深汕头人
写代码、爱旅游

- Github : @lightningtgc
- Weibo : @lightning川
- Zhihu : @汤桂川
- Email : tangguichuan@gmail.com

↑

一起实现金融报国梦请砸简历

Agenda - 大纲

① ES6介绍

- JavaScript简史
- 重点新特性
- Coding Style
- Best Practice

② ES6实战开发

- Angular - ES6
- Angular 2.0
- Babel
- jspm, Webpack, Gulp
- Linting, Debug , Test
- React, Node, Meteor - ES6

③ 发展趋势

- ES7
- Babel support
- Trending

Part 1 - ES6介绍

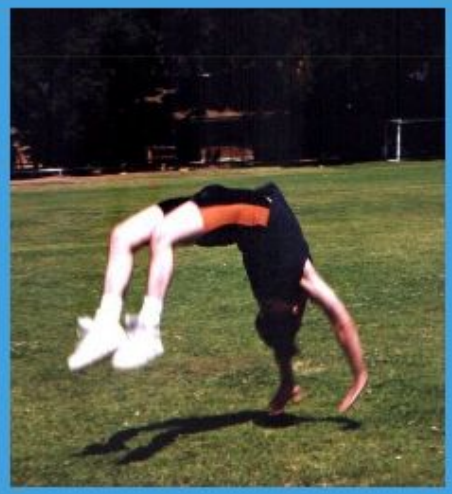
JavaScript简史

- 1995 LiveScript发布

  随后改名为 JavaScript
- 1997 ECMAScript 1
- 1999 ES3 (大变化）
- ES4 流产 (harmony)
- 2009 ES5（ES3.1）
- 2015. 06. 17 ES6

  ECMAScript 2015

Brendan Eich @Netscape



->1993年的小鲜肉

Via

《JavaScript at 20》

ECMAScript 5 在浏览器的实现情况

V8 SpiderMonkey JavaScriptCore Chakra Carakan KJS Other

Feature name	Current browser	iOS7/8	IE 10+	PhantomJS 2.0	CH 23+, OP 15+	FF 21+	BESEN	WebKit	SF 6+	OP 12.10	IE 9	EJS	Rhino 1.7	Konq 4.13
	100%	100%	100%	100%	100%	100%	100%	100%	100%	97%	97%	94%	92%	92%
Object.create	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.defineProperty	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.defineProperties	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.getPrototypeOf	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.keys	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.seal	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.freeze	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.preventExtensions	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.isSealed	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.isFrozen	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.isExtensible	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.getOwnPropertyDescriptor	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.getOwnPropertyNames	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Date.prototype.toISOString	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes
Date.now	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Array.isArray	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
JSON	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

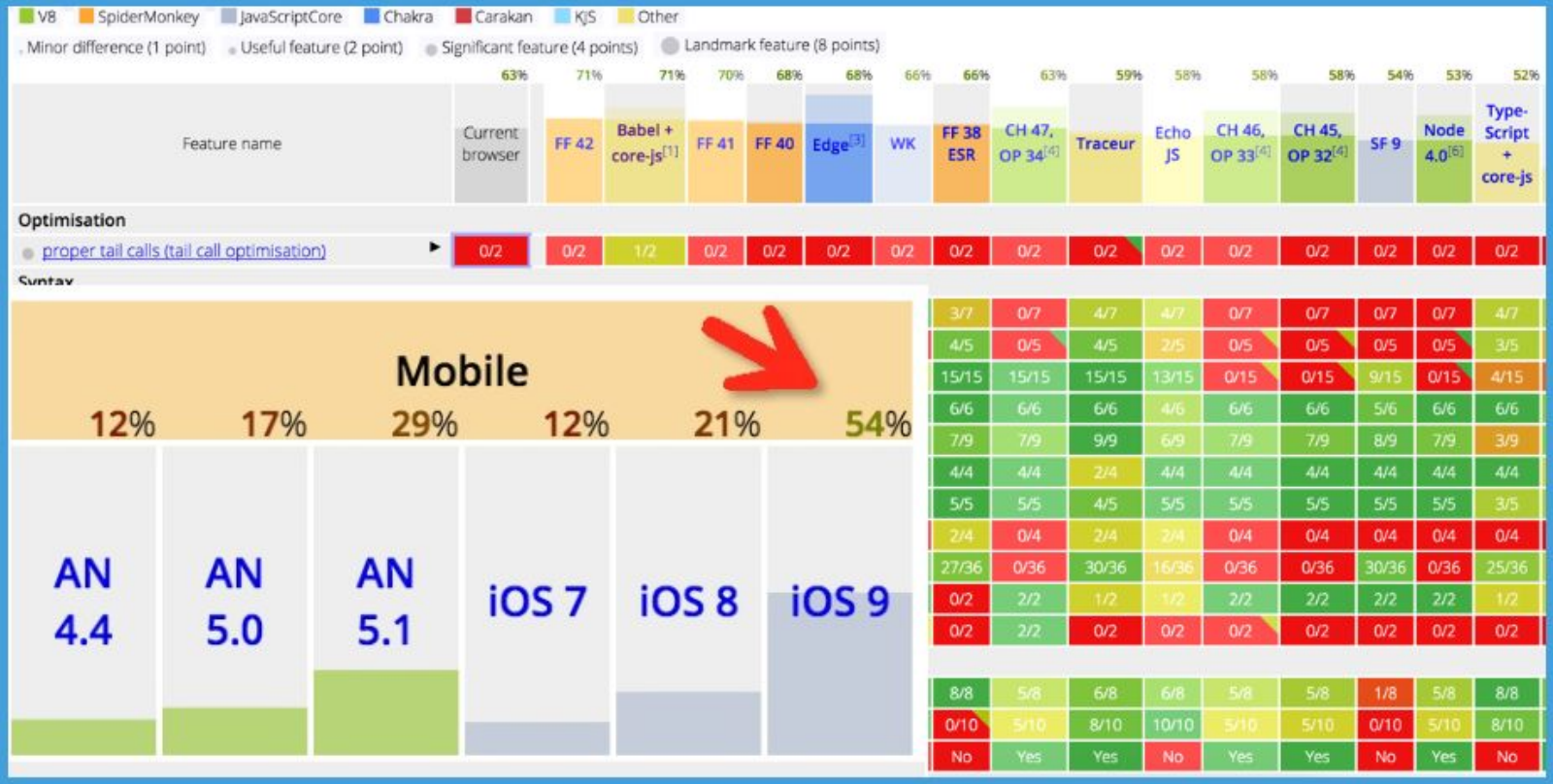
数据来源：http://kangax.github.io/compat-table/es5/

ECMAScript 6 在浏览器的实现情况

Feature name	Current browser 63%	FF 42 71%	Babel + core-js[1] 71%	FF 41 70%	FF 40 68%	Edge[3] 68%	WK 66%	FF 38 ESR 66%	CH 47, OP 34[4] 63%	Traceur 59%	Echo JS 58%	CH 46, OP 33[4] 58%	CH 45, OP 32[4] 58%	SF 9 54%	Node 4.0[6] 53%	Type-Script + core-js 52%
Optimisation																
proper tail calls (tail call optimisation)	0/2	0/2	1/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2
Syntax																
default function parameters	0/7	3/7	6/7	3/7	3/7	0/7	7/7	3/7	0/7	4/7	4/7	0/7	0/7	0/7	0/7	4/7
rest parameters	5/5	4/5	4/5	4/5	4/5	5/5	0/5	4/5	0/5	4/5	2/5	0/5	0/5	0/5	0/5	3/5
spread (...) operator	15/15	15/15	13/15	15/15	15/15	12/15	9/15	15/15	15/15	15/15	13/15	0/15	0/15	9/15	0/15	4/15
object literal extensions	6/6	6/6	6/6	6/6	6/6	6/6	6/6	6/6	6/6	6/6	4/6	6/6	6/6	5/6	6/6	6/6
for..of loops	7/9	7/9	9/9	7/9	7/9	6/9	8/9	7/9	7/9	9/9	6/9	7/9	7/9	8/9	7/9	3/9
octal and binary literals	4/4	4/4	4/4	4/4	4/4	4/4	4/4	4/4	4/4	2/4	4/4	4/4	4/4	4/4	4/4	4/4
template strings	5/5	5/5	4/5	5/5	5/5	4/5	5/5	5/5	5/5	4/5	5/5	5/5	5/5	5/5	5/5	3/5
RegExp "y" and "u" flags	0/4	2/4	2/4	2/4	2/4	2/4	0/4	2/4	0/4	2/4	2/4	0/4	0/4	0/4	0/4	0/4
destructuring	0/36	28/36	33/36	28/36	27/36	0/36	32/36	27/36	0/36	30/36	16/36	0/36	0/36	30/36	0/36	25/36
Unicode code point escapes	2/2	1/2	1/2	1/2	1/2	2/2	2/2	0/2	2/2	1/2	1/2	2/2	2/2	2/2	2/2	1/2
new.target	0/2	2/2	0/2	2/2	0/2	0/2	1/2	0/2	2/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2
Bindings																
const	5/8	8/8	6/8	8/8	8/8	8/8	8/8	8/8	5/8	6/8	6/8	5/8	5/8	1/8	5/8	8/8
let	5/10	0/10	8/10	0/10	0/10	8/10	10/10	0/10	5/10	8/10	10/10	5/10	5/10	0/10	5/10	8/10
block-level function declaration[14]	Yes	No	Yes	No	No	Yes	No	No	Yes	Yes	No	Yes	Yes	No	Yes	No

数据来源： http://kangax.github.io/compat-table/es6/

ECMAScript 6 在浏览器的实现情况



数据来源：http://kangax.github.io/compat-table/es6/

ES6 重点新特性

EMAScript 2015 规范文档》》

爱看规范的男生运气不会太差。

-- 乔布斯？？

Arrows

Syntax

```
const name = 'object name';
let obj = {
  name,
  init() {
    // Lexical this
    $.on('click', () => this.name);
  }
};
// 以上代码等同于 ES5
var name = 'object name';
var obj = {
  name: name,
  init: function init() {
    var _this = this;

    $.on('click', function () {
      return _this.name;
    });
  }
};
```

- C# 3.0 lambda
- Lexical this
- 单行隐性return value
- No arguments object

注意点(keng)：

```
//改造一下上面例子，this变成什么？
let obj = {
     init: ()=> this
};

// 上面代码等同于：
var obj = {
     init: function test() {
          return undefined;
     }
};
```

严格模式下，模块里顶层的this是undefined

- 5.5 对象中的函数方法使用缩写形式

> 更加简洁
>
> 函数方法不要使用箭头函数，避免this指向的混乱

```
// 不好
const shopObj = {
  des: '对象模块写法',
  foo: function() {
    console.log(this.des);
  }
};

const shopObj = {
  des: '对象模块写法',
  foo: () => {
    console.log(this.des); // 此处会变成undefined.des，因为指向顶层模块的this
  }
};

// 好
const des = '对象模块写法'; // 使用对象属性值简写方式
const shopObj = {
  des,
  foo() {
    console.log(this.des);
  }
};
```

More detail in：

https://github.com/gf-rd/es6-coding-style

Template String

Syntax

```
// 多行字符串，引号
var multiStr = `In 'JavaScript' this is
                not legal.`;
```

```
// 字符串插值
var name = "QCon", time = "today";
console.log(`Hello ${name},
how are you ${time}?`);
```

注意点(keng)？

字符串

- 2.1 处理多行字符串,使用模板字符串

> 注意排版引起空格的问题，使用场景为声明HTML模板字符串

```
// 不好
const tmpl = '<div class="content"> \n' +
               '<h1>这是换行了。</h1> \n' +
             '</div>';


// 好
const tmpl = `
<div class="content">
  <h1>这是换行了。</h1>
</div>`;
```

More detail in：

https://github.com/gf-rd/es6-coding-style

- 2.2 处理字符串拼接变量时,使用模板字符串

```
// 不好
function sayHi(name) {
  return 'How are you, ' + name + '?';
}


// 好
function sayHi(name) {
  return `How are you, ${name}?`;
}
```

Classes

```
class Animal {
  constructor(name) {
    this.name = name;
  }
  say(msg){
    alert(`${this.name} says ${msg}`);
  }
  static staticMethod() {
    return '不能被实例继承，子类可继承父类';
  }
}
class Panda extends Animal {
  say(msg = 'hi') {
    super(msg); // 调用父类say方法
  }
}
let panda = new Panda('Tuan');
panda.say(); // 'Tuan says hi'
panda.say('Ooooh'); // 'Tuan says Ooooh
```

Syntax

- constructor
- this
- default
- extends
- super
- 静态方法

```
// 上面代码转换为ES5
'use strict';

function _inherits(subClass, superClass) {
if (typeof superClass !== 'function' && superClass !== null) {
throw new TypeError('Super expression must either be null or a function, not ' + typeof
} subClass.prototype = Object.create(superClass && superClass.prototype, { constructor:
if (superClass) Object.setPrototypeOf ? Object.setPrototypeOf(subClass, superClass) : su
}
function _classCallCheck(instance, Constructor) { if (!(instance instanceof Constructor)
var Animal = (function () {
  function Animal(name) {
    _classCallCheck(this, Animal);
    this.name = name;
  }
  Animal.prototype.say = function say(msg) {
    alert(this.name + ' says ' + msg);
  };
  Animal.staticMethod = function staticMethod() {
    return '不能被实例继承，子类可继承父类';
  };
  return Animal;
})();
var Panda = (function (_Animal) {
  _inherits(Panda, _Animal);
  function Panda() {
    _classCallCheck(this, Panda);
    _Animal.apply(this, arguments);
  }
  Panda.prototype.say = function say() {
    var msg = arguments.length <= 0 || arguments[0] === undefined ? 'hi' : arguments[0];
    _Animal.prototype.say.call(this, msg); // 调用父类say方法
  };
  return Panda;
})(Animal);
var panda = new Panda('Tuan');
```

类

- 6.1 类名应使用帕斯卡写法(`PascalCased`)

```
// 好
class SomeClass {

}
```

- 6.2 定义类时，方法的顺序如下：
  - `constructor`
  - public `get/set` 公用访问器，`set` 只能传一个参数
  - public methods 公用方法，公用相关命名使用小驼峰式写法(`lowerCamelCase`)
  - private `get/set` 私有访问器，私有相关命名应加上下划线 `_` 为前缀
  - private methods 私有方法

Modules

Syntax

```
// 模块导入
import angular from 'angular';
import { lightRed } from './colors';
import * as UserModule from './user.module';

let userCtrl = UserModule.ctrl;

// 模块导出
export default lightRed;
export { userCtrl };
```

~~AMD, CommonJS, UMD~~

More Detail in: 下一代模块化的js

- 7.2 应确保每个module有且只有一个默认导出模块

> 方便调用方使用

```
// 不好
const lightRed = '#F07';

export lightRed;


// 好
const lightRed = '#F07';

export default lightRed;
```

- 7.4 不要将 `import` 与 `export` 混合在一行

> 分开导入与导出，让结构更清晰，可读性更强

```
// 不好
export { lightRed as default } from './colors';

// 好
import { lightRed } from './colors';
export default lightRed;
```

- 7.5 多变量要导出时应采用对象解构形式

> `export` 置于底部，使欲导出变量更加清晰

```
// 不好
export const lightRed = '#F07';
export const black  = '#000';
export const white  = '#FFF';

// 好
const lightRed = '#F07';
const black  = '#000';
const white  = '#FFF';

export default { lightRed, black, white };
```

ES6Features

查看详情: ES6features(中文版)

- let + const - 块级作用域
- destructuring - 解构
- arrows - 箭头函数
- classes - 类
- modules - 模块
- template strings
- default + rest + spread
- iterators + for..of - 遍历器
- generators函数
- promises对象
- map + set
- weakmap + weakset
- enhanced object literals - 增强对象字面量
- proxies - 代理
- symbols - 第七种数据类型
- binary and octal literals
- unicode - 字符串扩展
- reflect api - 反射api
- tail calls - 尾调用

More Info About ES6:

- ES6features(中文版)
- es6features(Github)
- 《Exploring ES6》- Free Online Book
- 《Understanding ECMAScript 6》- Book
- 《ECMAScript 6 入门》
- Mozilla: ES6 In Depth
- Ecma TC39 On Github

Part 2 - ES6实战开发

Workflow:

- Angular - ES6
- Babel
- Webpack & Gulp & JSPM
- ESlint
- Debug & Unit Test
- React , Node, Meteor - ES6

面向开发者编程

Angular - ES6

↓

Babel

↓

Webpack & Gulp & JSPM

↓

ESlint

↓

Debug & Unit Test

↓

React , Node, Meteor - ES6

Angular.js

- Angular ES5
- Angular ES6
- Angular 2.0

Angular ES6 VS ES5

ES5 Angular 1.x Service

```
angular.module('myapp', ['localInfos'])
.service('locationService', ['localInfo', '$q',
 function(localInfo, $q) {
    $scope.localInfo = localInfo;

    // 打印当前状态
    $scope.printInfo = function(msg) {
       msg = msg || 'unknown';
       console.log('State:' + msg);
    };

    // 寻找当前位置
    $scope.findLocation = function() {
      var deferred = $q.defer();
      var self = this;

      navigator.geolocation.getCurrentPosition(function(pos) {
        pos = pos || {};
        self.printInfo('Location');
        self.localInfo.save(pos);
        deferred.resolve('You are at ' + pos.coords.latitude);
      }, deferred.reject);

      return deferred.promise;
    };
  }
```

ES6 Angular 1.x Service

```
angular.module('myapp', ['localInfos'])
.service('locationService', LocationService);

class LocationService {
    constructor(localInfo, $q) {
        'ngInject';
        this.localInfo = localInfo;
        this.$q = $q;
    }

    // 打印当前状态
    printInfo(msg = 'unknown') {
       console.log(`State: ${msg}`);
    }

    // 寻找位置
    findLocation() {
        return this.$q((resolve, reject) => {

            navigator.geolocation.getCurrentPosition((pos = {}) => {
                this.printInfo('Location');
                this.localInfo.save(pos);
                resolve(`You are at ${pos.coords.latitude}`);
            }, reject);
        });
    }
```

Angular with es6 In Depth

```
// 'app/index.js'
import angular from 'angular';
import UserService from './user.service';
import UserController from './user.controller';

angular.module('myApp',[])
.factory('userService', UserService)
.controller('userController', UserController);
```

使用gulp-inject进行自动import与组装angular模块

Note: 文件遵循自动化命名规范(如: shop.controller.auto.js)

```
// 实现自动注入与Angular模块组装
import ShopController from './shop.controller.auto'

angular.module('shop',[])
.controller('shopController', ShopController);
```

另一种按页面进行分模块导入导出，
并可DIY暴露特定方法的模式

```
// "shop/shop.module.js"
import { ShopController } from './shop/shop.controller';
import { ShopService } from './shop/shop.service';

let ctrl = ShopController;
let svc = ShopService.init;

export { svc };
export { ctrl };
```

```
// 调用方 'index.js'
import * as ShopModules from './shop/shop.module';

angular.module('microshop',[])
.factory('shopSvc', ShopModules.svc)
.controller('shopCtrl', ShopModules.ctrl);
```

```
// 'user.service.js'
class UserService {
    constructor($http) {
        'ngInject';

        this.$http = $http;
        this.userUrl = '/serve/user/';
    }
    getUser(name = 'Hanmeimei') {
        return this.$http.get(this.userUrl + name)
            .then((response) => response.data);
    }
}

export default UserService;
```

'ngInject': 使用ng-annotate进行DI依赖自动注入

```
<div ng-app="myApp" ng-strict-di>
```

'ng-strict-di': missing DI annotations; ng 1.3 or later

Tips: Directives - controller vs controllerAs

Angular ES6 VS 2.0

Angular 2.0

- Why
- What

  AtScript vs TypeScript

  No $scope, controller, 2-way data binding

  No $watch, $apply: Zone.js
- When

  迁徙计划：

  Angular 1.4 Router

```
// Angular 2.0
import { UnicornHorn } from "../animals/unicorn";
class UnicornHype {
    constructor(unicornHorn:UnicornHorn) {
        this.horn = unicornHorn;
    }
    findUnicorn() {
        return new Promise((resolve, reject) => {
            navigator.geolocation.getCurrentPosition((pos) => {
                this.horn.thrust();
                resolve(`The unicorn is at ${pos.coords.latitude}`);
            }, reject);
        });
    }
}
// Angular ES6
angular.module('test', ['unicorn']).service('unicornHype', UnicornHype);
class UnicornHype {
    constructor(unicornHorn, $q) {
        'ngInject';
        this.horn = unicornHorn;
        this.$q = $q;
    }
    findUnicorn() {
        return this.$q((resolve, reject) => {
            navigator.geolocation.getCurrentPosition((pos) => {
                this.horn.thrust();
                resolve(`The unicorn is at ${pos.coords.latitude}`);
            }, reject);
        });
    }
}
```

More Info about Angular 2.0：

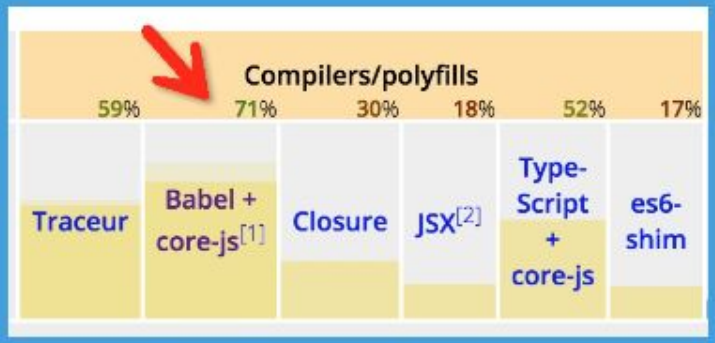
- 广发证券技术博客
- Angualr.js 2.0 官网
- try-angular2-today
- asm.js , Flow
- Change detection in angular 2
- Angular2 template syntax(ng-model)
- building applications in Angular


Angular - ES6
Babel
Webpack & Gulp & JSPM
ESlint
Debug & Unit Test
React , Node, Meteor - ES6

- WHY

1.Compatibility(支持度高):

2.生态圈与社区成熟:



Babel中有哪些优雅的实现？

Eg:引擎优化

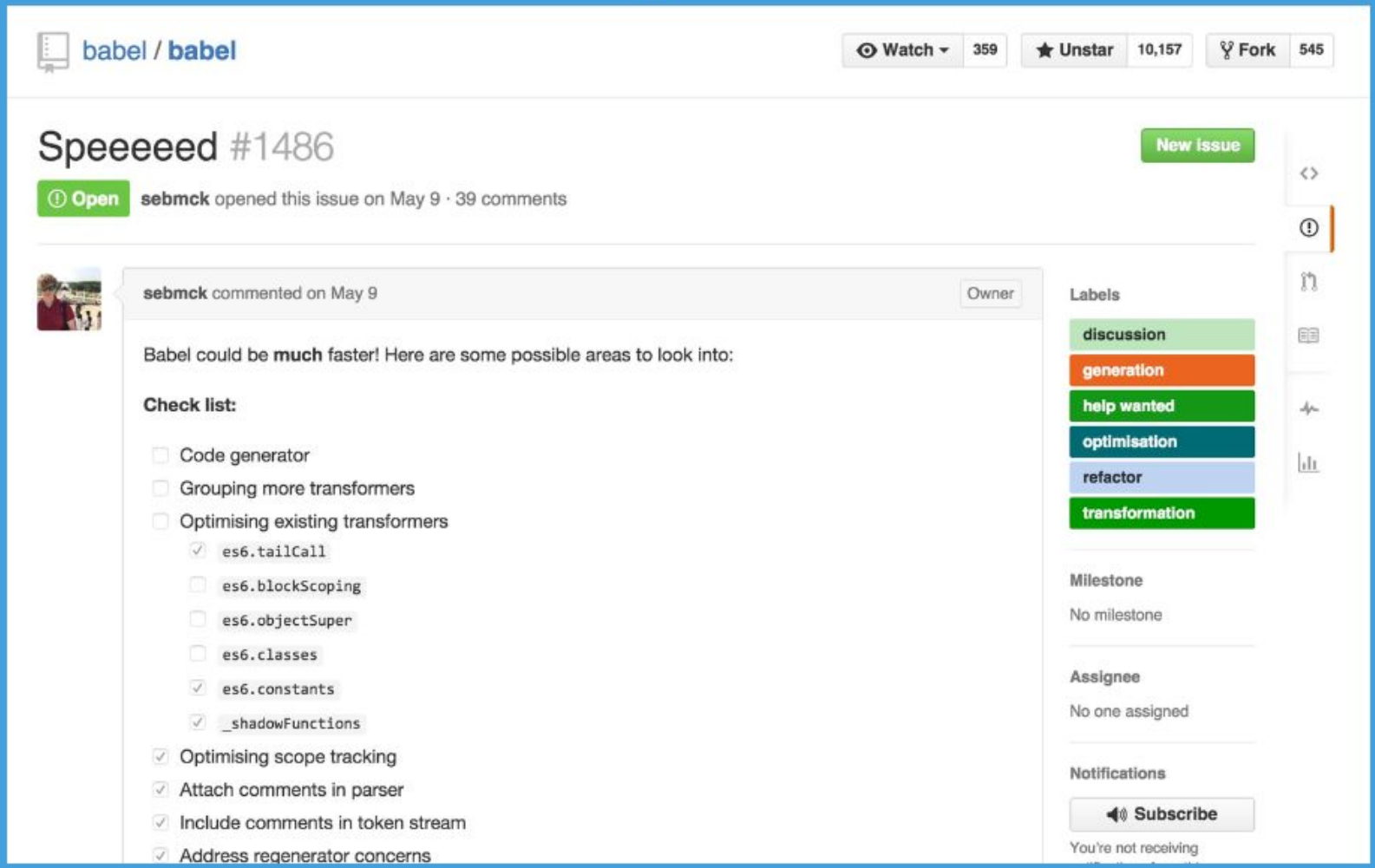
```
// ES6
(...args) => args
```

```
// 转换为ES5
(function () {
  for (
        var _len = arguments.length,
        args = Array(_len),
          _key = 0;
        _key < _len;
          _key++
      ) {

    args[_key] = arguments[_key];
  }

  return args;
});
```

More Detail：

- Don't leak arguments
- Optimization-killers
- immutable-js

Babel Speeeeed



More Information:

- Babel 官网
- Babel Polyfill For Old Browser 兼容
- Babel 6.0 新版本
- Babel在线转换ES6代码

Angular - ES6

↓

Babel

↓

Webpack & Gulp & JSPM

↓

ESlint

↓

Debug & Unit Test

↓

React , Node, Meteor - ES6

```
// gulp与babel结合

var gulp = require("gulp");
var babel = require("gulp-babel");

gulp.task("default", function () {

  return gulp.src("src/app.js")

    .pipe(babel())

    .pipe(gulp.dest("dist"));
});
```

More Information:

- Webpack官网
- Gulp.js官网
- Babel wtih Gulp and Webpack
- Writing client-side ES6 with webpack
- Webpack babel-loader

Angular - ES6

↓

Babel

↓

Webpack & Gulp & JSPM

↓

ESlint

↓

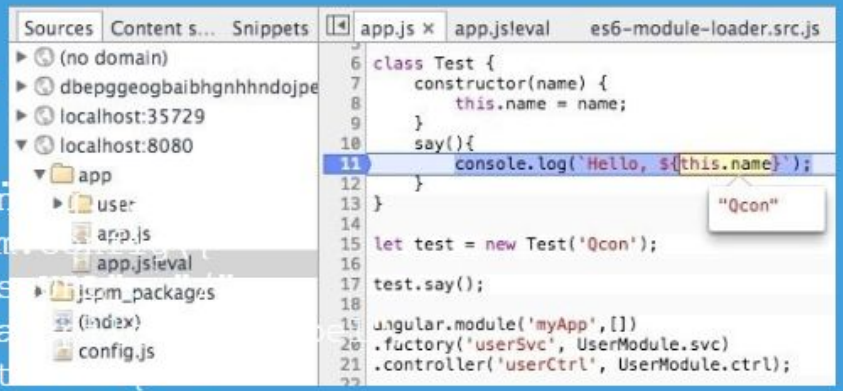
Debug & Unit Test

↓

React , Node, Meteor - ES6

What

- SystemJS - Module Loader
  - -- ES6,AMD,CommonJS...
  - -- babel-runtime
- Package Manager
  - -- local git repos, Github, NPM, Bower




```
// con
System
  "bas
  "tra
  "pat
    "github:*": "jspm_packages/github/*.js"
  }
});
```

How

- jspm
- Getting Started
- Plugins

```
<!-- index.html -->

<script src="jspm_packages/system.js">
</script>
<script src="config.js"></script>
<script>
  System.import('./app/index');
</script>
```

Angular - ES6

↓

Babel

↓

Webpack & Gulp & JSPM

↓

ESlint

↓

Debug & Unit Test

↓

React , Node, Meteor - ES6

ESLint

By Nicholas C. Zakas

The pluggable linting utility for JavaScript and JSX

Configuring:

- rules
- `ecmaFeatures`
- Parser:
    1. espree(Default)
    2. esprima (Base)
    3. esprima-fb (JSX)
    4. babel-eslint

babel-eslint

- Why
  1. Support Babel

     Transformed: ES7...
  2. Error Location Info

- How

Install

```
$ npm install -g eslint babel-eslint
```

Setup

.eslintrc

```
{
  "parser": "babel-eslint",
  "rules": {
    "strict": 0
  }
}
```

Check out the ESLint docs for all possible rules.

Run

```
$ eslint your-files-here
```

Angular - ES6

↓

Babel

↓

Webpack & Gulp & JSPM

↓

ESlint

↓

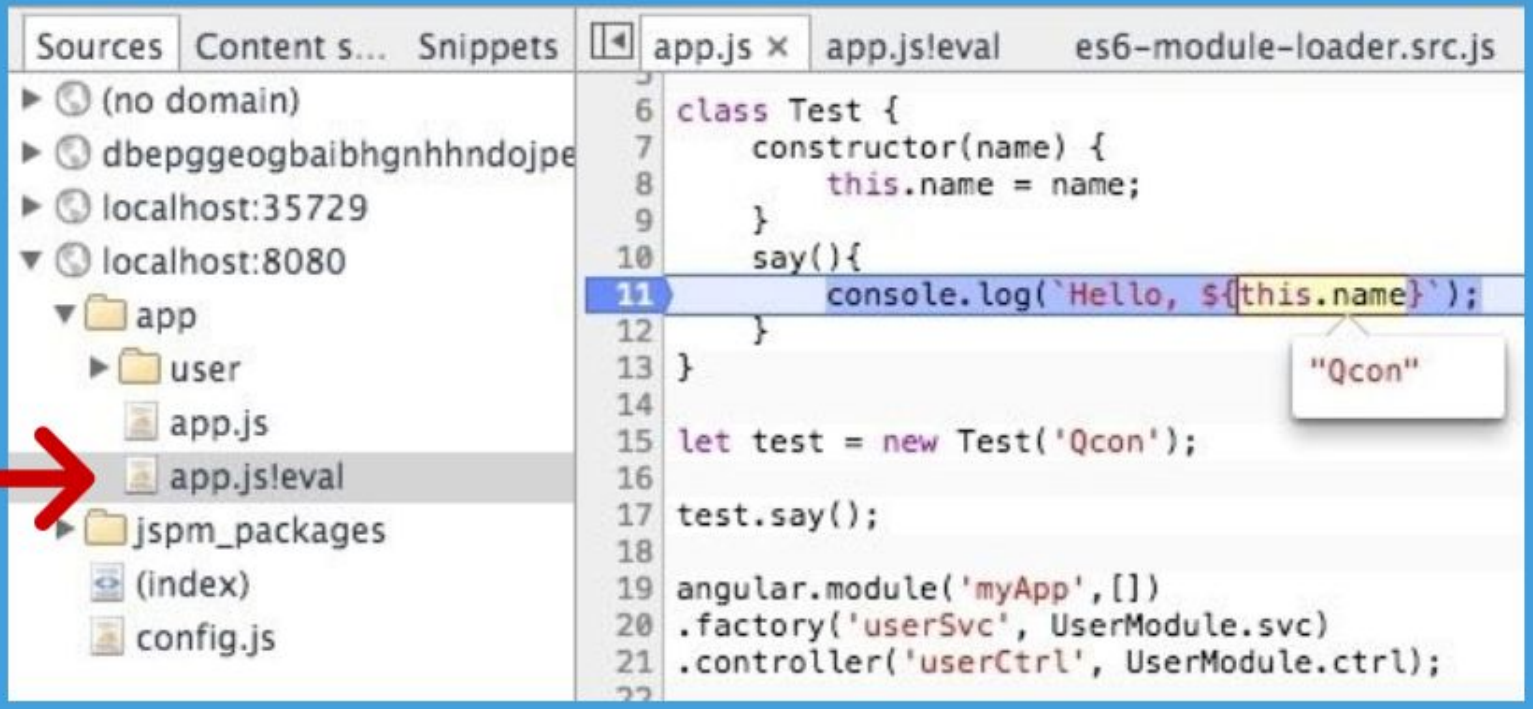
Debug & Unit Test

↓

React , Node, Meteor - ES6

HOW

- - > ES5
- Source Maps
- Runtime Transpiler



Angular - ES6

↓

Babel

↓

Webpack & Gulp & JSPM

↓

ESlint

↓

Debug & Unit Test

↓

React , Node, Meteor - ES6

Tools

- jasmine.js
- karma.js

Setting up the test environment:

- package.json
- karma.config.js
- test-context.js

```
// karma.config.js
module.exports = function(config) {
    config.set({
        browsers: ['PhantomJS'],
        frameworks: ['jasmine'],
        webpack: {
            module: {
                loaders: [{
                    test: /\.js/,
                    loader: 'babel-loader',
                    exclude: /node_modules/
                }]
            },
            watch: true
        }
    });
};
```

Target File For Test:

```
// test.js
export class Calculator{

    add(op1,op2){
        return op1 + op2;
    }

    subtract(op1,op2){
        return op1 - op2;
    }
}
```

Unit Test File:

```
// calculator-spec.js
import {Calculator} from './calculator';

describe('Calculator', () => {
    it('should add two numbers', () => {
        let calculator = new Calculator();
        let sum = calculator.add(1,4);
        expect(sum).toBe(5);
    });
     it('should subtract two numbers', () => {
         let calculator = new Calculator();
         let sum = calculator.subtract(4,1);
         expect(sum).toBe(3);
     });
});
```

Angular - ES6

↓

Babel

↓

Webpack & Gulp & JSPM

↓

ESlint

↓

Debug & Unit Test

↓

React , Node, Meteor - ES6

React v0.13

```
export class Counter extends React.Component {
  constructor(props) {
    super(props);
    this.state = {count: props.initialCount};
  }
  tick() {
    this.setState({count: this.state.count + 1});
  }
  render() {
    return (
      <div onClick={this.tick.bind(this)}>
        Clicks: {this.state.count}
      </div>
    );
  }
}
Counter.propTypes = {
    initialCount: React.PropTypes.number
};
Counter.defaultProps = { initialCount: 0 };
```

Tools for react, jsx

More Info:

- React On ES6 Plus
- React v0.14 (Symbol - XSS)
- Jest -PAINLESS JS UNIT TESTING
- Redux
- Refactoring React Components To ES6 Classes

Angular - ES6

↓

Babel

↓

Webpack & Gulp & JSPM

↓

ESlint

↓

Debug & Unit Test

↓

React , Node, Meteor - ES6

- ES6 on io.js

  No more --harmony (Node v0.12)

- Node V4
- ES6 in Node.js
  - shipping
  - staged
  - in progress

- 7 reasons to upgrade to node v4 now
  1. Template Strings
  2. Classes
  3. Arrow Functions
  4. Object Literals
  5. Promises
  6. String Methods
  7. Let and const

Angular - ES6

↓

Babel

↓

Webpack & Gulp & JSPM

↓

ESlint

↓

Debug & Unit Test

↓

React , Node, Meteor - ES6

Announcing Meteor 1.2
– ES6, Angular, React, and More

ES6 is now the official JavaScript
of the Meteor platform.

Every new Meteor project now uses ES6
by default, in every JS file.

Projects and Info

- 广发Angular ES6骨架
- 广发前端开源项目
- Angular2-ES6-Starter
- essential-react
- NG6-starter
- react-seed
- NgBabelJspm
- ES6 Tools

Part 3 - 发展趋势

ECMAScript 2016 (ES7)

方案生命周期 - stages：

Stage 0 ：Strawman 建议方案

Stage 1 ：Proposal 提案

Stage 2： Draft 草案

Stage 3： Candidate 候选方案

Stage 4： Finished 定案

- See: ECMAScript Current Status

Babel 支持

Stage 0

- es7.comprehensions - [for (i of [1, 2, 3]) i*i]
- es7.classProperties - 静态类属性
- es7.doExpressions
- es7.functionBind - ::

Stage 1

- es7.decorators - 装饰器语法
- es7.exportExtensions
- es7.trailingFunctionCommas

Stage 2

~~Object observe~~

- es7.exponentiationOperator
- es7.async-await - 异步编程模型
- es7.objectRestSpread - ...

使用Babel玩转ES7

- 开启支持stage里的特性

```
// 命令行
$ babel --stage 0

// 从代码里开启
babel.transform("code", { stage: 0 });
```

注意：Stage 2 及以上是默认开启支持的

- 单独开启某某ES7特性

```
// 命令行
$ babel --optional es7.decorators

// 从代码里开启
babel.transform("code", { optional: ["es7.decorators"] });
```

More Information:

- 开启Babel ES7支持
- ecmascript-7
- Brendan Eich Blog
- Mozilla JavaScript Reference
- TC39 On Github
- ES7支持性

Thank you!

By 汤桂川 with love

In Sydney Fall 2015

广发证券前端团队愿景：

做地球上最有钱的前端团队

简历请砸：

↓

tangguichuan@gmail.com