# es6笔记

目录

[es6笔记 1](#_Toc10230691)

[ES6简介 1](#_Toc10230692)

[为什么要学习es6? 2](#_Toc10230693)

[ES6变化-let&const 2](#_Toc10230694)

[ES6变化-spreed&rest 3](#_Toc10230695)

[ES6变化- destructuring 3](#_Toc10230696)

[ES6变化-箭头函数 3](#_Toc10230697)

[箭头函数特点 3](#_Toc10230698)

[作用 4](#_Toc10230699)

[ES5-Object.defineProperty 4](#_Toc10230700)

[简介 4](#_Toc10230701)

[使用 4](#_Toc10230702)

[注意 4](#_Toc10230703)

[作用 4](#_Toc10230704)

[ES5- 数据劫持 4](#_Toc10230705)

[ES6变化-Proxy&Reflect 5](#_Toc10230706)

[简介 5](#_Toc10230707)

[总结 5](#_Toc10230708)

[ES6变化- Set 5](#_Toc10230709)

[简介 5](#_Toc10230710)

[特点 5](#_Toc10230711)

[用途 5](#_Toc10230712)

[ES6变化- Map 5](#_Toc10230713)

[简介 5](#_Toc10230714)

[特点 5](#_Toc10230715)

[用途 5](#_Toc10230716)

[原理实现 6](#_Toc10230717)

[ES6变化-Promise 6](#_Toc10230718)

[异步编程简述 6](#_Toc10230719)

[异步编程问题 6](#_Toc10230720)

[解决方案 6](#_Toc10230721)

## ES6简介

介绍：ES2015年推出的JavaScript新版本也叫ES2015

现在已经有ES7（2016）、 ES8（2017）、 ES9（2018）很多新增内容仍是提案。普遍使用的依然是ES6、ES7部分内容

ES7及后面版本浏览器或node环境未完全支持，但已有babel工具可对其进行编译支持 (转成ES5)

所以：新版本语言只是在原有基础上新增了一些语法糖，执行时还会转化成ES5

语言迭代目的：

是使JavaScript语言可以用于编写复杂的大型应用程序，成为企业级的开发语言。

### 为什么要学习es6?

ES6中引入的语言新特性，更具规范性，易读性、方便操作、简化了大型

项目开发的复杂程度、降低了出错概率，提升了开发效率。

大部分公司（不代表所有，代表着一个趋势）都在用ES6开发，已成为开发、求职必会内容

## ES6变化-let&const

1.回顾 var（变量声明）：变量声明提升、可重复定义、全局变量挂载到window

2.回顾作用域（变量生命周期）：全局作用域、函数作用域

3. let（块级变量声明）：没有变量声明提升、不能重复定义、不挂载到window

声明的变量和{}配合产生块级作用域-生命在大括号内部的变量无法在外部使用

产生临时Temporal Dead Zone （临时死区）

解决闭包问题（ES6规范后引入的）

4. const （块级常量声明）：其存储量的空间不可以被改变、其余和let一样

5.项目潜移推荐使用：先const，再let

6.小细节：ES6 函数声明、let var 混合使用、函数形参声明方式

## ES6变化-spreed&rest

… 展开&收集运算符：

此运算符在不同地方使用有不同的功效，可以从写和读两个角度考虑。

写：function test (…arg){}; test(1,2,3)；-收集作用 返回一个数组对象

读：var arg = [1,2,3]; console.log(…arg); -展开作用

作用：简化书写长度，提升开发效率

ES6/ES7:ES6可以处理数组，ES7能处理对象(目前只能浅克隆)

Object.assign:浅层clone，可以理解成从$.extend那里演化过来。

## ES6变化- destructuring

解构（结构化赋值）：

解构过程中，具备赋值和变量声明两个功能

目的在于把等号左右长的相似的两个东西内部的值取出来。

对象数组都可以参与解构：

let obj = {name: ‘duyi’，age: 18};

let {name, age} = obj;

或

let name，age;（{name，age} = obj）;

作用：简化书写长度，提升开发效率

默认参数：let {name , age, sex =‘male’} = obj；

更改变量名称：let {name：myName} = obj；

## ES6变化-箭头函数

箭头函数特点：

1.不用写function关键字

2.只能作为函数使用不能new，没有原型

3.参数不能重复命名

4.返回值可以不写return，但是有时需要配合{}

5.内部 arguments this 由定义时外围最接近一层的非箭头函数的

arguments和this决定其值

作用：函数目的指向性更强，可读性更好，简化代码，提升开发效率

## ES5-Object.defineProperty

简介：ES5规范开始后续版本迭代，也在致力于做一件事，就是把js底层已有的功能，提供给开发者使用。Object.defineProperty就是其中一个，此方法会可直接在一个对象上定义一个新的具有详细描述的属性，或者修改一个对象的现有属性， 并返回这个对象。

使用：

Object.defineProperty(对象，属性，描述符);描述符对，对象的属性的进行详细描述：

数据描述符：

value: ‘xxx’属性值。 默认‘’

writable：true 是否可写。默认false

configurable：true 是否可配置。默认false

enumerable：true 是否可枚举。默认false

存取描述符：

set：function(){} 属性访问器 进行写操作时调用该方法

get：function(){} 属性访问器 进行读操作时调用该方法

注意：如果描述符中同时出现，value、writable，和set、get两组的话，会出现异常。切记不要同时使用

作用：双向数据绑定的核心方法，主要做数据劫持操作（监控属性变化），同时是后期ES6中很多语法糖底层实现的核心方法。

## ES5- 数据劫持

VUE双向数据绑定核心功能由 Observer、Compile、Watcher三部分实现其中Observer部分功能实现有Object.defineProperty实现。

Observer：监测数据变化进行相应回调（数据劫持）；

实现一个简单的数据劫持，作为Object.defineProperty的练习。从而引出Proxy&Reflect。

## ES6变化-Proxy&Reflect

简介：植入代理模式的思想，以简洁易懂的方式控制对外部对象的访问。

总结：利用内置的set、get方法控制属性的读写功能用处较大

其余has deleProperty...等方法不太在工作开发中使用(兼容不好)

## ES6变化- Set

简介：Set是ES6提供给我们的构造函数，能够造出一种新的存储数据的结构

特点：只有属性值，成员值唯一（不重复）

用途：可以转成数组，其本身具备去重，交集，并集，差集的作用等

## ES6变化- Map

简介：Map 是ES6提供给我们的构造函数，能够造出一种新的存储数据的结构。本质上是键值对的集合。

特点：key对应value，key和value唯一，任何值都可以当属性。

用途：可以让对象当属性，去重等。

原理实现：链接链表、hash算法、桶

## ES6变化-Promise

异步编程简述：

无论是在浏览器环境中还是在node环境中，我们都会使用JavaScript

完成各种异步操作，如在浏览器环境中的定时器、事件、ajax等或是node环境中的文件读取、事件等。伴随着异步编程的就是回调机制。明确一点异步编程避免不了回调。

异步编程问题：

产生回调地狱，难于维护和扩展。

try catch只能捕获同步代码中出现的异常。

同步并发的异步存在一定的问题。

解决方案：

ES6 Promise可以解决回调地狱、以及同步并发的异步问题。（异步操作

的异常捕获有其他方式解决。）但依旧会有明显的回调痕迹，之后学习ES6的generator 、ES7的async await争取让异步代码看起来和同步一样。写起来更优雅一些。

Promise: new Promise(executor 函数); 同步执行

状态：pending 等待 Fulfilled 成功 Rejected 失败

executor 函数:参数 resolve reject 函数，对应触发Promise对象注册的成功和失败的函数。

返回值：Promise 对象