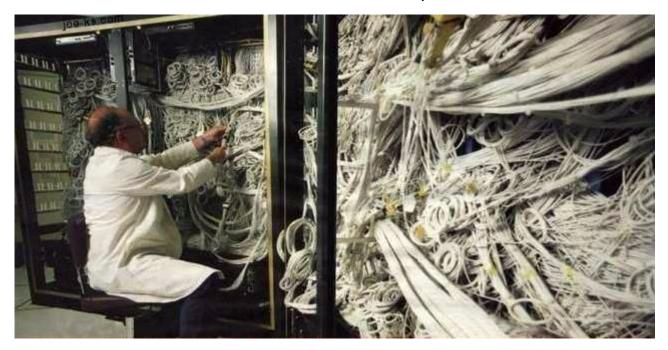
### 一、模块化简介

#### 1、模块化产生的背景

随着网站逐渐变成"互联网应用程序",嵌入网页的Javascript代码越来越庞大,越来越复杂。



Javascript模块化编程,已经成为一个迫切的需求。理想情况下,开发者只需要实现核心的业务逻辑,其他都可以加载别人已经写好的模块。

但是, Javascript不是一种模块化编程语言,它不支持"类" (class),包 (package)等概念,更遑论"模块" (module)了。

#### 2、什么是模块化开发

传统非模块化开发有如下的缺点:

- 命名冲突
- 文件依赖

#### 模块化规范:

- CommonJS模块化规范
- ES6模块化规范

### 二、CommonJS模块规范

每个文件就是一个模块,有自己的作用域。在一个文件里面定义的变量、函数、类,都是私有的,对其他文件不可见。

#### 1、创建 "module" 文件夹

#### 2、导出模块

创建 common-js模块化/四则运算.js

```
1 // 定义成员:
2 const sum = function(a,b){
3    return parseInt(a) + parseInt(b)
4 }
5 const subtract = function(a,b){
6    return parseInt(a) - parseInt(b)
7 }
8 const multiply = function(a,b){
9    return parseInt(a) * parseInt(b)
10 }
11 const divide = function(a,b){
12    return parseInt(a) / parseInt(b)
13 }
```

#### 导出模块中的成员

```
1 // 导出成员:
2 module.exports = {
3    sum: sum,
4    subtract: subtract,
5    multiply: multiply,
6    divide: divide
7 }
```

#### 简写

```
1 //简写
2 module.exports = {
3 sum,
```

```
subtract,
multiply,
divide
}
```

### 3、导入模块

创建 common-js模块化/引入模块.js

```
1 //引入模块,注意: 当前路径必须写 ./
2 const m = require('./四则运算.js')
3 console.log(m)
4
5 const result1 = m.sum(1, 2)
6 const result2 = m.subtract(1, 2)
7 console.log(result1, result2)
```

#### 4、运行程序

```
1 node common-js模块化/引入模块.js
```

CommonJS使用 exports 和require 来导出、导入模块。

## 三、ES6模块化规范

ES6使用 export 和 import 来导出、导入模块。

### 1、导出模块

创建 es6模块化/userApi.js

```
1 export function getList() {
2   console.log('获取数据列表')
3 }
```

```
export function save() {
    console.log('保存数据')
}
```

### 2、导入模块

创建 es6模块化/userComponent.js

```
1 //只取需要的方法即可,多个方法用逗号分隔
2 import { getList, save } from "./userApi.js"
3 getList()
4 save()
```

注意:这时的程序无法运行的,因为ES6的模块化无法在Node.js中执行,需要用Babel编辑成ES5后再执行。

#### 3、运行程序

```
1 node es6模块化-dist/userComponent.js
```

# 四、ES6模块化的另一种写法

### 1、导出模块

创建 es6模块化/userApi2.js

```
1 export default {
2    getList() {
3        console.log('获取数据列表2')
4    },
5    save() {
7        console.log('保存数据2')
8    }
```

# 2、导入模块

创建 es6模块化/userComponent2.js

```
import user from "./userApi2.js"
user.getList()
user.save()
```