**Js模块化总结**

# 前言

仅供参考

<https://www.jianshu.com/p/e99033028b16>

上面网站为commonJS、AMD、CMD三者的定义及使用。面试问到结合本文档进行交流。

下载第三方库：

npm install jquery@3【设置下载版本】 --save;

# 一、commonJS\_Node 模块化方法

必备环境，node.js

## 1.创建项目结构

创建项目根目录node；

node下创建项目目录modules；

node下创建js索引文件app.js；

node下创建package.JSON文件；【必须有】

modules下创建三个js文件，分别为module1，module2, module3;

package.JSON文件中写入项目名称和项目版本；

{

“name”:”commonJS-node”

“version”:”1.1.0”

}

## 2.模块化编码

编码使用在三个module文件下分贝使用不同的编码方式进行编译【属于同一种方法】。

### 1）编码基本语法及格式

module.exports module为全局，exports为暴露；

### 2）对象直接暴露

只能赋值一次，下次赋值会覆盖已有的数据；

modeule.exports=function(){ };

### 3）对象传属性暴露

modeule.exports={

fun:function(){ }

}

可设置多个，调用时对象.属性名；

### 4）直接暴露

exports.fun(){ }

## 3.索引js文件引入

在app.js文件下引入其他文件；

let module1=require(“./modules/module1”);

let module2=require(“./modules/module2”);

let module3=require(“./modules/module3”);

调用方法

module1();

module2.fun();

module3.fun();

## 4.运行监测

通过node运行索引js文件，app.js;

\*命令 node app.js;

\*工具：右键--->运行；

# 二、commonJS-Browserify模块化方法

## 1.创建构建项目结构

创建项目根目录commonJS；

根目录下创建目录js;

根目录下创建html文件index.html;

根目录下创建JSON文件package.JSON；【必须有】

js目录下创建目录dist；【打包生成文件目录】；

js目录下创建目录src；【源码所在目录】；

src目录下创建js文件，分别为module1，module2, module3,app.js【应用主源文件】;

package.JSON文件中写入项目名称和项目版本；

{

“name”:”commonJS-Browserify”;

“version”:”1.1.0”;

}

## 2.下载browserify

\*全局：npm install browserify –g 【下载一次后以后不用再下载】；

\*局部：npm install browserify --save-deg;【不同的项目需要时，都需要重新下载】；

## 3.模块化编码

编码使用在三个module文件下分贝使用不同的编码方式进行编译【属于同一种方法】。

### 1）编码基本语法及格式

module.exports module为全局，exports为暴露；

### 2）对象直接暴露

只能赋值一次，下次赋值会覆盖已有的数据；

modeule.exports=function(){ };

### 3）对象传属性暴露

modeule.exports={

fun:function(){ }

}

可设置多个，调用时对象.属性名；

### 4）直接暴露

exports.fun(){ }

## 4.索引js文件引入

在app.js文件下引入其他文件；

let module1=require(“./ module1”);

let module2=require(“./ module2”);

let module3=require(“./ module3”);

调用方法

module1();

module2.fun();

module3.fun();

## 5.打包处理js

\*browserify js/src/app.js –o js/dist/bundle.js ,在node环境中执行【注意根目录】

## 6.页面引用使用

<script type=”text/javascript” src=”js/dist/bundle.js”></script>

# 三、AMD-RequireJS模式化方法

## 1.准备工作：

下载require.js，并引入；

\*官网：<http://www.requirejs.cn>;

\*github: <http://github.com/requires/requeirejs>

\*将require.js导入项目：js/libs/require.js

## 2.创建项目结构

创建根目录require

根目录下创建目录js

根目录下创建html文件index.html

js目录下创建目录libs

js目录下创建目录modules

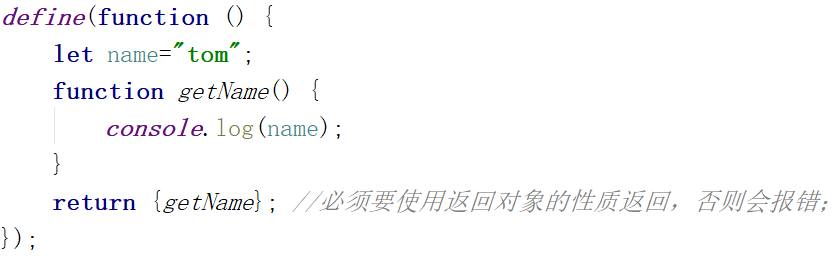
js目录下创建索引主源js文件main.js

libs目录下放入文件require.js,juqery.js【可选】,angular.js【可选】

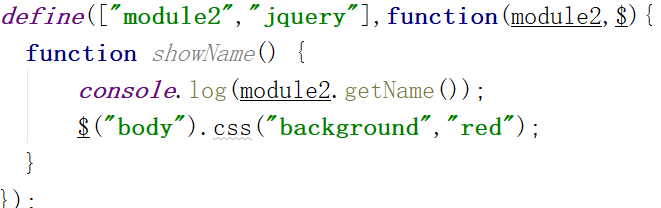
modules目录下创建两个文件js文件，分别为module1,module2;

## 3.模块化编码

### 1）定义没有依赖的编码



### 2）定义有依赖的代码



中括号当中为需要引入的文件名；需要return需要暴露的方法；

## 4.索引js文件mian.js文件引入

### 1)引入模块代码

(**function** () {  
 *require*.config({ *//配置* **baseUrl**: **'js/'**, *//基本路径  
 //映射: 模块标识名: 路径* **paths**: { *//自定义模块* **'module1'**: **'modules/module1'**,  
 **'module2'**: **'modules/module2'**,  
 *//库模块* **'jquery'**: **'libs/jquery-1.10.1'**,  
 **'angular'**: **'libs/angular'** },  
 *//配置不兼容AMD的模块* **shim**: {  
 **angular**: {  
 **exports**: **'angular'** }  
  
 }  
 });

### 2）引入模块使用代码

*require*([**'module1'**, **'module2'**], **function** (module1, module2) {  
 module2.*showName*();  
 ***console***.log(angular);  
 });  
 *require*([**"jquery"**],**function** ($) {  
 $(**"body"**).css(**"background"**,**"red"**);  
 })  
})();

注：和上面的代码是一体的，需要一起使用；

## 5.页面引用使用模块

<script data-main="js/main" src="js/libs/require.js"></script>

# 四、ES6模块化方法

## 1.创建项目结构

创建根目录ES6demo

根目录下创建目录js

根目录下创建html文件index.html

根目录下创建JSON文件package.JSON【必须有】

根目录下创建JSON文件. babelrc【必须有】

Js目录下创建目录lib

Js目录下创建目录src

Src目录下创建js文件，分别为app.js,module1.js,module2.js,module3.js;

定义package文件

{

“name”：”es6-babel-browify”;

“version”：”1.1.0”;

}

定义. babelrc文件

{

“presets”:[“es2015”];

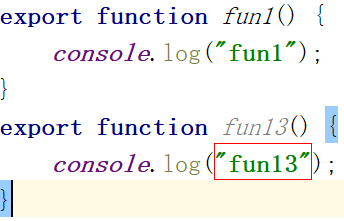
}

## 2.安装必要组件

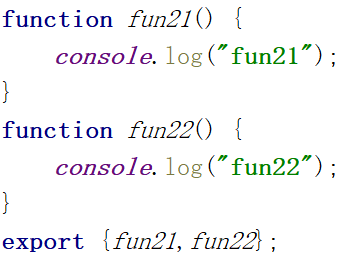
\* npm install babel-cli browserify -g //装在全局的之后不用再装  
\* npm install babel-preset-es2015 --save-dev //只要有新的项目，就必须重新安装  
\* preset 预设(将es6转换成es5的所有插件打包)

## 3.模块化编码

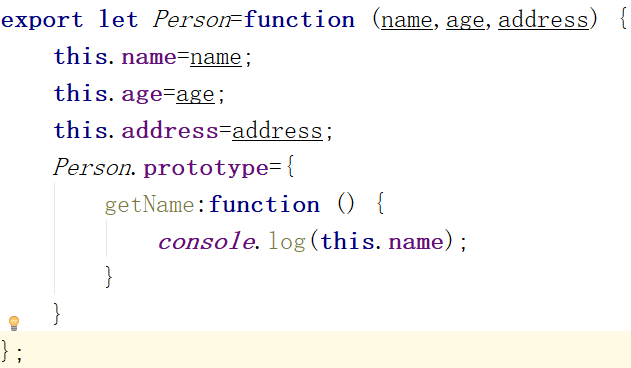
### 1）分别暴露



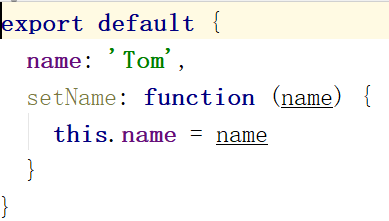
### 2）统一暴露



### 3）单个对象暴露

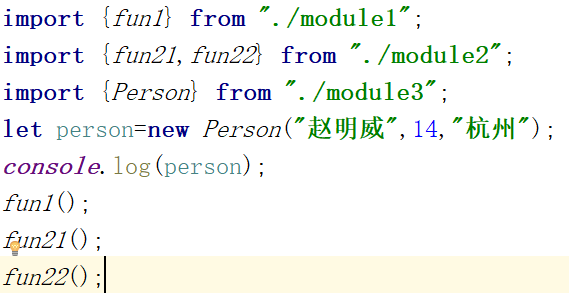


### 4)default默认暴露



可以暴露多个方法；

### 5）索引js文件代码配置



主要是使用import 方法，然后将函数或者对象放进中括号内，通过from引用对应文件的地址；

分别暴露，同一暴露和单个对象暴露的方法一致；default暴露使用：

**import** parent **from './module3' //主要区别在于不加大括号,import后面的可以自定义；**

## 5.编译

\* 使用Babel将ES6编译为ES5代码(但包含CommonJS语法) : babel js/src -d js/lib  
\* 使用Browserify编译js : browserify js/lib/app.js -o js/lib/bundle.js

## 6.引入第三方库（jquery）

1). 下载jQuery模块:  
 \* npm install jquery@1 --save  
2). 在app.js中引入并使用  
 import $ from 'jquery'  
 $('body').css('background', 'red')