陌上兮月

日拱一卒，功不唐捐

EsModule VS CommonJS

关于模块化

说一说 js 模块化这回事儿吧。

一开始 js 并没有模块化这个概念，但是没有模块化在应对一些大型前端应用开发时是非常不好管理的。所以社区催生出了一个野生模块化规范，叫做 *CommonJS*。至今这个规范仍然被应用在 NodeJS 中。

后来，ECMA 也意识到了模块化是必须的，在 ES6 中，官方性地将模块化加入到 ES 标准中，这就是大名鼎鼎的*ES6 模块*。

初识 CommonJS 用法

在 CommonJS 中，我们只需要知道两种用法就可以了，也就是 `require` 和 `module.exports` 。

```
// A文件
var firstName = "Michael";
var lastName = "Jackson";
var year = 1958;

module.exports = {
  firstName,
  lastName,
  year
};

// B文件
const Info = require('./A);
```

< 2024年5月 >

日	一	二	三	四	五	六
28	29	30	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	1
2	3	4	5	6	7	8

搜索

找找看

随笔分类 (153)

Git(8)

HTML+CSS(37)

JavaScript(42)

MongoDB(1)

Node.js(5)

React(6)

```
console.log(Info); // {firstName: "Michael", lastName: "Jackson", year: 1958
}

// 当然，也可以使用解构赋值
const { year } = require('./A);
console.log(year); // 1958
```

用法方面，知道上面示例这样的，我觉得就差不多可以了。

同时我们也可以看到，其实上述代码是把整个 A 模块都加载了，然后包装在一个对象里面导入进来。这种加载，我们称之为“运行时加载”。（代码跑起来的时候再去加载模块）

ES6 模块用法探究

像 ES6 模块的话，它的花样就更多一些，比如单单导出这方面，就分什么单独导出、默认导出、转发导出。我们这里可以看看现在比较常见的几种玩法。

单独导出

```
// A文件
// 单独导出第一种写法
export var firstName = "Michael";
export var lastName = "Jackson";
export var year = 1958;
```

```
// 单独导出第二种写法
var firstName = "Michael";
var lastName = "Jackson";
var year = 1958;

export { firstName, lastName, year };
```

这两种写法的共同特点就是：export 后面不能直接接一个变量或者值，而是要接一个“对应关系”，对于第二种写法来讲，尾部的大括号其实和 CommonJS 中 module.export 导出的大括号是不同的，`module.export =` 后面接的真的是个对象，而 `export` 后面接的，虽然长得很像对象，但是其实并不是对象。

```
export { nn: firstName, lastName, year }; // 报错
❌ Syntax Error: SyntaxError

export { firstName: 'zhang', lastName, year }; //❌ 报
错 Syntax Error: SyntaxError
```

所以，**这个大括号只是大括号而已，它圈定哪些变量要被输出去。**

而且像这种单独导出的，由于我们确实不是导出对象；在其他模块静态导入文件时，直接打印也是没东西的。

TypeScript(1)
Vue(15)
服务器(7)
计算机网络(19)
架构思想(3)
前端可视化(2)
三小记(2)
微信小程序(5)

阅读排行榜

1. 移动端布局之postcss-px-to-viewport（兼容vant）【更新于2021/09/27】(67577)
2. ping、网络抖动与丢包(34667)
3. CSS渐变的两种基本用法(30283)
4. 在vue项目中封装echarts的正确姿势(29519)
5. Canvas的drawImage方法使用(21176)
6. 利用正则表达式去掉字符串的前后空格(16519)
7. switch case加范围判断(16010)
8. CSS有哪些属性是可以继承的？(15627)
9. 配置Gitlab pages和Gitlab CI(15156)

```
import A from "./A";

console.log(A); // undefined
```

如果我们要使用它，只能使用一种 {} 的形式来决定取哪些模块，但是正如上面所说，**这里的大括号取法虽然很像解构，但其实它不是解构，只是看起来像而已。**

```
// 一般写法
import { year } from "./A";
console.log(year); // 1958

// 重命名写法
import { year as thatYear } from ".A"; // 重命名变量
console.log(year); // 1958

// 导入全部模块并存在一个对象里
import * as Info from "./A";
console.log(Info); // Module {firstName: "Michael", lastName: "Jackson", yea
r: 1958}
```

默认导出

除了单独导出，还有一个用得很多的写法，那就是默认导出。

所谓默认导出，也就是 export default ，它本质上就是导出一个叫 default 的变量，后面可以跟变量名、值。

```
// A文件
var firstName = "Michael";
var lastName = "Jackson";
var year = 1958;

export default { nn: firstName, lastName, year }; // 现在就不报错了，因为后面
跟的真的是一个对象
```

这个时候,外面用的时候就要把这个导出的变量赋值给另外一个变量了，而不能直接解构。

```
import JacksonInfo from "./A";
console.log(JacksonInfo); // {firstName: "Michael", lastName: "Jackson", yea
r: 1958}
```

简单总结一下：

- 单独导出。大括号选择性导出，大括号选择性导入
- 默认导出。导出一个变量，导入一个变量

冷门的转发导出

还有一种是转发导出，比如：

10. 在Vue-cli3.x中引入element-ui的新方式(14583)

```
export { year, firstName } from "./A";

// 上面这种写法基本等同于下面这两句，不过上面的写法有个特别之处，那就是转发导出的模块，是没有导入到当前模块的，所以才说和下面的写法“基本相同”
import { year, firstName } from "./A";
export { year, firstName };
```

好了好了别秀了，如果要了解更多连招操作，可以到阮一峰ES6文档去看。
传送门

CommonJS 和 ES6 模块的区别

还是来看看区别吧，这个更加重要，使用方法的不同，只是一些花招罢了。

就目前来说，这两者主要有以下两个区别：

1. ES6模块是静态导入，编译时加载的；而CommonJS是动态导入，运行时加载的；
2. **ES6模块不使用默认导出时**，即使导出原始数据类型，导出的仍然是其引用。而CommonJS导出原始数据类型则只是导出一个值而已；

第一个区别很好理解，我们知道ES6的静态加载确实有点厉害，它能在编译时，也就是代码运行之前就确定好各模块的导入导出关系，这样非常容易做静态优化（比如多次import某个模块，合并为一次import）。另外这种静态加载的特点也决定 `import ABC fom "xxx"` 这种语句不能放在类似条件语句这种代码块中，而是要放在代码顶层，因为代码块需要运行才知道结果，而由于 `import ABC fom "xxx"` 的优先级是高于代码运行的，所以是即使被放在代码底部也是没问题的，同样会有命令提升的效果。下面这种操作不会报错：

```
console.log(Info);

import Info from "./A";
```

那第二个区别怎么理解呢？简单来讲，**ES6模块不使用默认导出时**，导出任何东西都是导出其引用（无论是基本类型还是引用类型），各个模块import这个值的时候，实际上是去同一个地方（源模块）取值。或者这么说：*ES6模块不用默认导出时，导出的都是活指针。*

而CommonJS导出基本类型的时候，就是导出一个值而已，与源模块没有连接关系。

```
// CommonJS中
var num = 0

module.exports = { num }; // 就是导出 { num: 0 }这个对象而已
```

```
// ES6模块中
var num = 0

export { num }; // 导出当前num变量的指针，取值的时候都是跑回当前这个源模块来取
```

```
// ES6模块默认导出
var num = 0

export default num ; // 这样导出的东西又变成值了，而不是活指针
```

分类: JavaScript

« 上一篇: [清代八股文]Promise如何实现串行执行

» 下一篇: [React Hooks长文总结系列一]初出茅庐，状态与副作用

posted @ 2021-03-26 20:37 陌上兮月 阅读(442) 评论(0) 编辑 收藏 举报