博学谷
www.boxuegu.com

第十六章 Module模块

一样的在线教育，不一样的教学品质

目录 Contents

什么是模块化？

开发中将JavaScript代码封装到多个js文件中，项目的各个页面去引用，导致需要引入更多的js文件。

```
<script src="zepto.js"></script>
<script src="jhash.js"></script>
<script src="fastClick.js"></script>
<script src="iScroll.js"></script>
<script src="underscore.js"></script>
<script src="handlebar.js"></script>
<script src="datacenter.js"></script>
<script src="util/wxbridge.js"></script>
<script src="util/login.js"></script>
<script src="util/base.js"></script>
```

js文件引入造成如下问题：

1.全局变量污染：各个文件的变量都是挂载到window对象上，污染全局变量。

2.变量重名：不同文件中的变量如果重名，后面的会覆盖前面的，造成程序运行错误。

3.文件依赖顺序：多个文件之间存在依赖关系，需要保证一定加载顺序。

什么是模块化？

模块化其实是一种规范，一种约束，这种约束会大大提升开发效率；将每个js文件看作是一个模块，每个模块通过固定的方式引入，并且通过固定的方式向外暴露指定的内容 。

目录 Contents

export命令的使用

一个模块是一个独立的文件；该文件内部所有变量外部无法获取。若希望外部能够获取模块内部的变量，就必须使用export命令输出该变量。

```
export let userName = 'zs';
export let userAge = 18;
```

export的写法，还可采用如下写法：

```
let userName = 'zs';
let userAge = 18;
export { userName, userAge }
```

以上两种写法是等价的关系，但是建议采用第二种写法：**其可以看出整个模块导出的变量。**

export命令除了可以输出变量以外，也可以**输出函数或者是类(Class）。**

```
export function add(num1, num2) {
    return num1 + num2
}
```

目录 Contents

import命令的使用

使用export命令导出模块中的变量及函数，若其它文件需要变量或函数，可通过import命令进行导入。

```
<script  type='module'>
        import {
            userName,
            userAge,
            add
        } from "./student.js";
        console.log(userName, userAge);
        console.log(add(3, 6));

    </script>
```

- **import命令用于加载student.js文件，并获取该文件中导出的变量和函数。**
- **import命令接受一个对象，在大括号中指定的名字一定要和模块中导出的变量，函数名字要相同。**
- **若想要更换名字，可在import命令中使用as关键字，将变量名或者是函数名进行重新命名。**

目录 Contents

模块的整体加载

除了指定加载某个输出的值，还可以使用整体加载，使用星号指定一个对象，所有输出值都加载到该对象上。

```
<script type="module">
        import * as stu from './student.js';
        console.log(stu.userName);
        console.log(stu.userAge);
        console.log(stu.add(3, 6));
    </script>
```

目录 Contents

export default命令

使用import命令时，需要知道模块所导出的变量名和函数名的方式较为麻烦。

用户在使用某个模块时，还需去查看文档了解该模块导出的变量和函数。

```
export default function add(num1, num2) {
    return num1 + num2;
}
```

使用 export default命令简化：

```
<script type='module'>
        import showAdd from './student.js'
        console.log(showAdd(3, 6));
    </script>
```

import命令后的名字可以是任意的名字，不需要和模块中导出的函数名字保持一致，且import命令后不使用大括号。

export default命令

export default命令也可使用如下写法：

```
function add(num1, num2) {
    return num1 + num2;
}
export default add;
```

问题：

导出的模块文件中，同时使用了export default命令与export命令，那么如何使用import命令进行导入呢？

```
let userName = 'zs';
let userAge = 18;
export { userName, userAge }

function add(num1, num2) {
    return num1 + num2;
}
export default add;
```

```
import showAdd, {
            userName,
            userAge
        } from './student.js';
        console.log(showAdd(3, 6));
        console.log(userName,
userAge);
```

export default命令

export default命令可导出类：

◆ **导出类代码：**

```
export default class Person {
    constructor(num1, num2) {
        this.num1 = num1;
        this.num2 = num2;
    }
    getShow() {
        return this.num1 + this.num2;
    }
}
```

◆ **导入代码**

```
import Student from './student.js';
    let stu=new Student(8,9);
    console.log(stu.getShow());
```

目录 Contents

ES6模块加载的本质

通过如下代码体会模块加载的实质：

◆ **模块中导出的代码**

```
export let count = 3;
export function addCount() {
    count++;
}
```

◆ **导入的代码**

```
import {
            count,
            addCount
        } from './moduleDemo.js';
        console.log(count);
        addCount();
        console.log(count);
```

模块中的值，发生变化，对应导入的值也会发生变化，说明ES6中模块是动态引用的，并不会缓存值。

所以，**ES6模块输出的是值的引用**。

一样的在线教育，不一样的教学品质