## 两种加载方式

|  |  |  |  |
| --- | --- | --- | --- |
| 加载方式 | 规范 | 命令 | 特点 |
| 运行时加载 | CommonJS/AMD | require | 社区方案，提供了服务器/浏览器的模块加载方案  非语言层面的标准  只能在运行时确定模块的依赖关系及输入/输出的变量，无法进行静态优化。 |
| 编译时加载 | ESMAScript6+ | import | 语言规格层面支持模块功能  支持编译时静态分析，便于JS引入宏和类型检验  动态绑定 |

## export命令

### 定义

export命令用于规定模块的对外接口

### 输出变量

1. 单个输出

// profile.js

export var firstName = 'Michael';

export var lastName = 'Jackson';

export var year = 1958;

1. 批量输出

// profile.js

var firstName = 'Michael';

var lastName = 'Jackson';

var year = 1958;

export {firstName, lastName, year};

1. 重命名

var n = 1;

export {n as m};

1. 动态绑定

export语句输出的接口，与其对应的值是动态绑定关系，即通过该接口，可以取到模块内部实时的值。

export var foo = 'bar';

setTimeout(() => foo = 'baz', 500);

上面代码输出变量foo，值为bar，500毫秒之后变成baz。

### 输出函数

1. 单个输出
   1. 方式一

export function multiply(x, y) {

return x \* y;

};

* 1. 方式二

function f() {}

export {f};

1. 批量输出

function v1() { ... }

function v2() { ... }

export {v1,v2}

1. 重命名

function v1() { ... }

function v2() { ... }

export {

v1 as streamV1,

v2 as streamV2,

v2 as streamLatestVersion

};

重命名后，v2可以用不同的名字输出两次。

### 输出类

### export default

## import命令

### 定义

使用export命令定义了模块的对外接口以后，其他JS文件就可以通过import命令加载这个模块（文件）。

### 导入变量

1. 批量导入

import命令接受一个对象（用大括号表示），里面指定要从其他模块导入的变量名。大括号里面的变量名，必须与被导入模块（profile.js）对外接口的名称相同。

import {firstName, lastName, year} from './profile';

1. 重命名

import { lastName as surname } from './profile';

### 提升

import命令具有提升效果，会提升到整个模块的头部，首先执行

foo();

import { foo } from 'my\_module';

上面的代码不会报错，因为import的执行早于foo的调用。

### 执行

import语句会执行所加载的模块，因此可以有下面的写法。

import 'lodash';

上面代码仅仅执行lodash模块，但是不输入任何值

## 小结

1. import命令导入的对象可能是一个变量，也可能是一个函数，或者一个类，要视具体情况而定。
2. import与require的差异较大，最大不要混用。要尽可能的使用import.