export资料
https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Reference/Statements/export

export 定义

在创建JavaScript模块时，**export** 语句用于从模块中导出实时绑定的函数、对象或原始值，其他程序可以使用**import**进行导入。import导入的时候，这些绑定值只能被导入模块所读取。export导出模块中对这些绑定值进行修改，所修改的值也会实时的更新。
export 模块都处于严格模式，export语句不能嵌入在js语句中，只能是顶级作用域。如果在其他作用域会报错。

export 两种导出方式

1. **命名导出**
2. **默认导出**

```
// 导出单个特性
export let name1, name2, …, nameN; // also var, const
export let name1 = …, name2 = …, …, nameN; // also var, const
export function FunctionName(){...}
export class ClassName {...}

// 导出列表
export { name1, name2, …, nameN };

// 重命名导出
export { variable1 as name1, variable2 as name2, …, nameN };

// 解构导出并重命名
export const { name1, name2: bar } = o;

// 默认导出
export default expression;
export default function (…) { … } // also class, function*
export default function name1(…) { … } // also class, function*
export { name1 as default, … };

// 导出模块合集
export * from …; // does not set the default export
```

```
export * as name1 from …; // Draft ECMAScript® 2021
export { name1, name2, …, nameN } from …;//导出另外模块下某些函数方法
export { import1 as name1, import2 as name2, …, nameN } from …;
export { default } from …;
```

注意：

命名导出 export，一定需要以import {} 方式导入且 {}导入的名称与导出的名称保持一致。

默认导出 export ，可以任意使用一个别名来接收，不需要使用{}的方式，直接使用别名的方式接收。

Import 定义

静态的import 语句用于导入由另一个模块导出的绑定。无论是否声明了 strict mode ，导入的模块都运行在严格模式下。在浏览器中，import 语句只能在声明了 type="module" 的 script 的标签中使用。

动态import() 不需要依赖 `type="module"` 的script标签，在 script 标签中使用 nomodule 属性，可以确保向后兼容。

静态导入和动态导入

```
// 静态导入
import './a.js';

import b from './b.js';
b();

import {c} from './c.js';
c();

// 动态导入，import()返回是一个promise，提供Promise的API
import('./a.js').then(()=>{
 console.log('a.js is loaded dynamically');
});

import('./b.js').then((module)=>{
 const b = module.default;
 b('isDynamic');
});

import('./c.js').then(({c})=>{
```

```
 c('isDynamic');
});
```

动态import()导入可以结合 async await来使用

```
// STATIC
import {test} from './utils_en.js'; // no dynamic locale
test();

// DYNAMIC
(async () => {
 const locale = 'en';

 const {test} = await import(`./utils_${locale}.js`);
 test('isDynamic');
})();
```

相关语法:

```
import defaultExport from "module-name";
import * as name from "module-name";
import { export } from "module-name";
import { export as alias } from "module-name";
import { export1 , export2 } from "module-name";
import { foo , bar } from "module-name/path/to/specific/un-exported/file";
import { export1 , export2 as alias2 , [...] } from "module-name";
import defaultExport, { export [ , [...] ] } from "module-name";
import defaultExport, * as name from "module-name";
import "module-name";
var promise = import("module-name");//这是一个处于第三阶段的提案。
```

如果export default 默认导出话(import命令可以为该匿名函数指定任意名字),import命令接受一对大括号，里面指定要从其他模块导入的变量名，**必须与被导入模块（profile.js）对外接口的名称相同。**和上面的注意一个意思。

按需加载

```
//根据情况动态import 相关js
button.addEventListener('click', event => {
 import('./dialogBox.js')
 .then(dialogBox => {
```

```
  dialogBox.open();
  })
  .catch(error => {
  /* Error handling */
  })
});
```

条件加载

```
//根据不同的条件，加载不同的模块
if (condition) {
 import('moduleA').then(...);
} else {
 import('moduleB').then(...);
}
```