ESModule 使用

- 静态引入
- 外链
- 引入第三方链接
- 动态引入

```
// utils.js
export const add = (a, b) => a + b;
export const minus = (a, b) => a - b;
export const multiply = (a, b) => a * b;
export const divide = (a, b) => a / b;
```

```
// console.js
console.log("console");
```

```
<!-- 静态引入 -->
<script type="module">
    import { add, minus, multiply, divide } from "./utils.js";
    const a = 30, b = 20;
    console.log("a + b = ", add(a, b));  // 50
    console.log("a - b = ", minus(a, b)); // 10
    console.log("a * b = ", multiply(a, b)); // 600
    console.log("a / b = ", divide(a, b)); // 1.5
</script>
```

```
<!-- 外链 -->
<script type="module" src="./console.js">
    // 定义在这里的代码不会执行, console 是 console.js 文件返回的
</script>
```

```
<!-- 引入第三方链接 -->
<script type="module">
    import { onBeforeMount } from
"https://cdn.bootcdn.net/ajax/libs/vue/3.3.4/vue.esm-browser.min.js";
    console.log("onBeforeMount", onBeforeMount);
    // onBeforeMount (t,e=currentInstance)=>
(!isInSSRComponentSetup||"sp"===n)&&injectHook(n,(...e)=>t(...e),e)
</script>
```

```html
<!-- 动态引入 -->
<button id="btn1">btn1</button>
<button id="btn2">btn2</button>
<script type="module">
    const btn1 = document.getElementById("btn1");
    const btn2 = document.getElementById("btn2");

    btn1.addEventListener("click", async () => {
        let res1 = await import("./utils.js");
        const a = 30, b = 20;
        console.log("a + b = ", res1.add(a, b)); // 50
        console.log("a - b = ", res1.minus(a, b)); // 10
        console.log("a * b = ", res1.multiply(a, b)); // 600
        console.log("a / b = ", res1.divide(a, b)); // 1.5
    });

    btn2.addEventListener("click", async () => {
        let res2 = await import("./utils.js");
        const a = 60, b = 20;
        console.log("a + b = ", res2.add(a, b)); // 80
        console.log("a - b = ", res2.minus(a, b)); // 40
        console.log("a * b = ", res2.multiply(a, b)); // 1200
        console.log("a / b = ", res2.divide(a, b)); // 3
    });
</script>
```