

JavaScript 条件判断与循环语句

一、🌟 条件判断 (if / if-else / switch)

✅ 1. 条件判断的概念

条件判断让程序“**根据情况选择不同的执行路径**”。

就像我们生活中的判断逻辑：

如果今天下雨了，就带伞；否则就不带伞。

✅ 2. if 语句

📌 语法结构：

```
if (条件) {  
    // 条件为 true 时执行的代码  
}
```

✅ 示例：



```
let age = 18;
if (age >= 18) {
  console.log("你已成年");
}
```

✓ 3. if-else 语句



```
let score = 65;

if (score >= 60) {
  console.log("及格");
} else {
  console.log("不及格");
}
```

✓ 4. if-else if-else 多条件判断



```
let temperature = 35;

if (temperature > 37) {
  console.log("发烧了");
} else if (temperature > 30) {
  console.log("有点热");
} else {
  console.log("正常");
}
```

✅ 5. switch 语句（适合判断固定值）

```
let fruit = "香蕉";

switch (fruit) {
  case "苹果":
    console.log("你选的是苹果");
    break;
  case "香蕉":
    console.log("你选的是香蕉");
    break;
  default:
    console.log("没有这种水果");
}
```

⚠️ 注意：

- 每个 case 后面必须加 **break**，否则会继续执行下面的 case。
- **default** 是所有条件都不满足时的执行结果。

🌀 二、循环语句（for / while / do-while）

✅ 1. 什么是循环？

循环就是“重复做某件事”，直到某个条件不成立为止。

生活中的例子：

洗碗时，每次洗一只，直到碗全部洗完。

✅ 2. for 循环（最常用）

标准的 `for` 循环（经典三段式）



```
for (初始值; 条件; 每次循环后的操作) {  
    // 循环体  
}
```

示例：打印 1 到 5



```
for (let i = 1; i <= 5; i++) {  
    console.log(i);  
}
```

`for...in` 循环（用于对象）

🌟 用于遍历对象的 **键名（属性名）**。



js

复制编辑

```
let person = { name: 'Jett', age: 18 };  
  
for (let key in person) {  
    console.log(key + ': ' + person[key]);  
}
```

输出：



makefile

复制编辑

name: Jett

age: 18

注意：

- `for...in` 会遍历对象 **可枚举属性**，包括继承的属性；
- 不推荐用它遍历数组。

`for...of` 循环（用于数组和其他可迭代对象）

🌟 用于遍历数组、字符串、Set、Map 等“可迭代对象”的 **值**。



js

复制编辑

```
let arr = ['apple', 'banana', 'cherry'];
```

```
for (let fruit of arr) {  
  console.log(fruit);  
}
```

输出：



nginx

复制编辑

apple

banana

cherry

嵌套 **for** 的结构和流程



js

复制编辑

```
for (let i = 0; i < 外层循环次数; i++) {  
  // 每执行一次 i 的循环, 就会执行一次完整的 j 的循环  
  for (let j = 0; j < 内层循环次数; j++) {  
    // 循环体: 执行的代码  
  }  
}
```

✅ 流程解释:

1. 外层循环开始 (**i=0**)
2. 内层循环完整执行一轮 (**j=0~n**)
3. 外层循环继续下一轮 (**i=1**), 再执行一轮完整的内层循环
4. 一直重复, 直到外层结束

✨ 输出一个 4x5 的星号表格 (4行5列)



js

复制编辑

```
for (let i = 0; i < 4; i++) {           // 控制行数
  let line = '';
  for (let j = 0; j < 5; j++) {         // 每行5个星号
    line += '★ ';
  }
  console.log(line);
}
```

✓ 输出：



复制编辑

```
★ ★ ★ ★ ★
★ ★ ★ ★ ★
★ ★ ★ ★ ★
★ ★ ★ ★ ★
```

✓ 解释：

- 外层控制 **有多少行**
- 内层控制 **每行输出多少个“★”**
- `line += '★ '` 是把每个星号接到一起
- 每打印一行，就输出一次整行的内容

✓ 3. while 循环

语法结构：



```
while (条件) {  
    // 条件为 true 时一直执行  
}
```

示例：打印 1 到 5



```
let i = 1;  
while (i <= 5) {  
    console.log(i);  
    i++;  
}
```

4. do...while 循环（先执行一次，再判断）

语法结构：



```
do {  
    // 先执行一次  
} while (条件);
```

示例：



```
let i = 1;
do {
  console.log(i);
  i++;
} while (i <= 5);
```



三、循环控制语句（break 和 continue）



1. break: 跳出整个循环



```
for (let i = 1; i <= 10; i++) {
  if (i === 5) {
    break; // 到 5 停止
  }
  console.log(i);
}
// 输出: 1 2 3 4
```



2. continue: 跳过当前循环，进入下一次



```
for (let i = 1; i <= 5; i++) {
  if (i === 3) {
    continue; // 跳过 3
  }
  console.log(i);
}
// 输出: 1 2 4 5
```



小结对比表

类型	适用场景	特点
<code>if/else</code>	复杂判断分支	条件灵活，支持区间判断
<code>switch</code>	精确值匹配	多个固定值判断更清晰
<code>for</code>	明确次数的循环	最常见，适合计数
<code>while</code>	不确定次数的循环	条件为真就一直循环
<code>do...while</code>	至少执行一次的循环	哪怕条件为假，也会先执行一次
<code>break</code>	提前退出循环	可用于 for / while / switch 中
<code>continue</code>	跳过本轮循环	不退出循环，只跳过当前这一次