



黑马程序员™  
www.itheima.com

传智播客旗下  
高端IT教育品牌

# JavaScript 流程控制-循环

# 目录 Contents

- ◆ 循环
- ◆ for 循环
- ◆ 双重 for 循环
- ◆ while 循环
- ◆ do while 循环
- ◆ continue break

# 1. 循环

## 循环目的

- 在实际问题中，有许多具有规律性的重复操作，因此在程序中要完成这类操作就需要重复执行某些语句

# 1. 循环

## 3. JS 中的循环

在Js 中，主要有三种类型的循环语句：

- for 循环
- while 循环
- do...while 循环

# 目录

# Contents

- ◆ 循环
- ◆ for 循环
- ◆ 双重 for 循环
- ◆ while 循环
- ◆ do while 循环
- ◆ continue break

## 2. for 循环

在程序中，一组被重复执行的语句被称之为**循环体**，能否继续重复执行，取决于循环的**终止条件**。由循环体及循环的终止条件组成的语句，被称之为**循环语句**

### 2.1 语法结构

for 循环主要用于把某些代码循环若干次，通常跟计数有关系。其语法结构如下：

```
for (初始化变量; 条件表达式; 操作表达式 ) {  
    //循环体  
}
```

- **初始化变量**：通常被用于初始化一个计数器，该表达式可以使用 var 关键字声明新的变量，这个变量帮我们来记录次数。
- **条件表达式**：用于确定每一次循环是否能被执行。如果结果是 true 就继续循环，否则退出循环。
- **操作表达式**：每次循环的最后都要执行的表达式。通常被用于更新或者递增计数器变量。当然，递减变量也是可以的。

## 2. for 循环

### 2.1 语法结构

```
for( 初始化变量; 条件表达式; 操作表达式 ){  
    //循环体语句  
}
```

#### 执行过程:

1. 初始化变量, **初始化操作在整个 for 循环只会执行一次。**
2. 执行条件表达式, 如果为true, 则执行循环体语句, 否则退出循环, 循环结束。
3. 执行操作表达式, 此时第一轮结束。
4. 第二轮开始, 直接去执行条件表达式 (不再初始化变量), 如果为 true, 则去执行循环体语句, 否则退出循环。
5. 继续执行操作表达式, 第二轮结束。
6. 后续跟第二轮一致, 直至条件表达式为假, 结束整个 for 循环。

## 2. for 循环

### 2.1 语法结构

#### 断点调试:

断点调试是指自己在程序的某一行设置一个断点，调试时，程序运行到这一行就会停住，然后你可以一步一步往下调试，调试过程中可以看各个变量当前的值，出错的话，调试到出错的代码行即显示错误，停下。

#### 断点调试可以帮助我们观察程序的运行过程

浏览器中按 F12--> sources --> 找到需要调试的文件--> 在程序的某一行设置断点

Watch: 监视，通过watch可以监视变量的值的变化，非常的常用。

F11: 程序单步执行，让程序一行一行的执行，这个时候，观察watch中变量的值的变化。

代码调试的能力非常重要，只有学会了代码调试，才能学会自己解决bug的能力。初学者不要觉得调试代码麻烦就不去调试，知识点花点功夫肯定学的会，但是代码调试这个东西，自己不去练，永远都学不会。

今天学的代码调试非常的简单，只要求同学们记住代码调试的这几个按钮的作用即可，后面还会学到很多的代码调试技巧。



## 2. for 循环

### 2.2 for 循环重复相同的代码

for循环可以重复相同的代码，比如我们要输出10句“媳妇我错了”

```
// 基本写法
for(var i = 1; i <= 10; i++){
    console.log('媳妇我错了~');
}

// 用户输入次数
var num = prompt('请输入次数:');
for ( var i = 1 ; i <= num; i++) {
    console.log('媳妇我错了~');
}
```

## 2. for 循环

### 2.3 for 循环重复不相同的代码

for 循环还可以重复不同的代码，这主要是因为使用了计数器，计数器在每次循环过程中都会有变化。

例如，求输出一个人1到100岁：

```
// 基本写法
for (var i = 1; i <= 100; i++) {
    console.log('这个人今年' + i + '岁了');
}
```

## 2. for 循环

### 2.3 for 循环重复不相同的代码

for 循环还可以重复不同的代码，这主要是因为使用了计数器，计数器在每次循环过程中都会有变化。

例如，求输出一个人1到100岁：

```
// for 里面是可以添加其他语句的
for (var i = 1; i <= 100; i++) {
    if (i == 1) {
        console.log('这个人今年1岁了， 它出生了');
    } else if (i == 100) {
        console.log('这个人今年100岁了，它死了');
    } else {
        console.log('这个人今年' + i + '岁了');
    }
}
```

## ■ 2. for 循环

### 2.4 for 循环重复某些相同操作

for 循环因为有了计数器的存在，我们还可以重复的执行某些操作，比如做一些算术运算。

## 2. for 循环

### 课堂案例1：求1-100之间所有整数的累加和

案例分析：

- ① 需要循环100次，我们需要一个计数器 `i`
- ② 我们需要一个存储结果的变量 `sum`，但是初始值一定是 0
- ③ 核心算法： $1 + 2 + 3 + 4 \dots$ ，`sum = sum + i;`

## 2. for 循环



### 课堂案例 1：求1-100之间所有整数的累加和

实现代码：

```
var sum = 0;
for(var i = 1;i <= 100; i++){
    sumNum += i;
}
console.log('1-100之间整数的和 = ' + sum);
```

## 2. for 循环



### 课堂练习

- ① 求1-100之间所有数的平均值
- ② 求1-100之间所有偶数和奇数的和
- ③ 求1-100之间所有能被3整除的数字的和

## 2. for 循环



### 课堂案例 2：求学生成绩

要求用户输入班级人数，之后依次输入每个学生的成绩，最后打印出该班级总的成绩以及平均成绩。

此网页显示

请输入班级总的人数:

确定 取消



## 2. for 循环



### 案例分析

- ① 弹出输入框输入总的班级人数 ( num )
- ② 依次输入学生的成绩 (保存起来 score) , 此时我们需要用到 for 循环, 弹出的次数跟班级总人数有关系 条件表达式  $i \leq \text{num}$
- ③ 进行业务处理: 计算成绩。先求总成绩 (sum) , 之后求平均成绩 (average)
- ④ 弹出结果

## 2. for 循环



### 实现代码

```
var num = prompt('请输入班级总的人数:'); // num 班级总的人数
var sum = 0; // 总成绩
var average = 0; // 平均成绩
for (var i = 1; i <= num; i++) {
    var score = prompt('请输入第' + i + '个学生的成绩');
    sum = sum + parseFloat(score);
}
average = sum / num;
alert('班级总的成绩是: ' + sum);
alert('班级总的平均成绩是: ' + average);
```

## 2. for 循环

### 一行打印五个星星



我们采取追加字符串的方式，这样可以打印到控制台上。

```
var star = '';  
for (var i = 1; i <= 5; i++) {  
    star += '☆'  
}  
console.log(star);
```

## 2. for 循环

### 思考



# 目录 Contents

- ◆ 循环
- ◆ for 循环
- ◆ 双重 for 循环
- ◆ while 循环
- ◆ do while 循环
- ◆ continue break



## 3. 双重 for 循环

### 3.1 双重 for 循环概述

很多情况下，单层 for 循环并不能满足我们的需求，比如我们要打印一个 5 行 5 列的图形、打印一个倒直角三角形等，此时就可以通过循环嵌套来实现。



**循环嵌套**是指在一个循环语句中再定义一个循环语句的语法结构，例如在for循环语句中，可以再嵌套一个for 循环，这样的 for 循环语句我们称之为**双重for循环**。



## 3. 双重 for 循环

### 3.2 双重 for 循环语法

```
for (外循环的初始; 外循环的条件; 外循环的操作表达式) {  
    for (内循环的初始; 内循环的条件; 内循环的操作表达式) {  
        需执行的代码;  
    }  
}
```

- 内层循环可以看做外层循环的语句
- 内层循环执行的顺序也要遵循 for 循环的执行顺序
- 外层循环执行一次，内层循环要执行全部次数

## 3. 双重 for 循环

### 3.4 打印五行五列星星



核心:

1. 内层循环负责一行打印五个星星
2. 外层循环负责打印五行





## 3. 双重 for 循环

### 3.4 打印五行五列星星

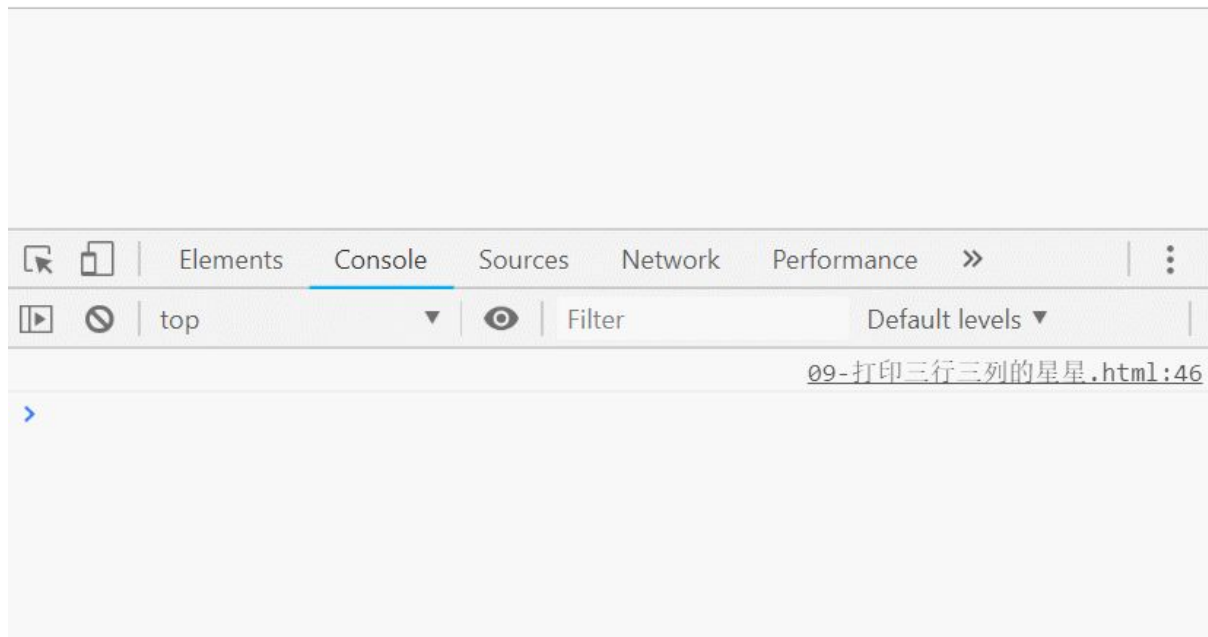
```
var star = '';
for (var j = 1; j <= 3; j++) {
    for (var i = 1; i <= 3; i++) {
        star += '☆'
    }
    // 每次满 5个星星 就 加一次换行
    star += '\n'
}
console.log(star);
```

## 3. 双重 for 循环



### 课堂案例 1：打印 n 行 n 列的星星

要求用户输入行数和列数，之后在控制台打印出用户输入行数和列数的星星。





## 3. 双重 for 循环



### 实现代码

```
var row = prompt('请输入您打印几行星星:');  
var col = prompt('请输入您打印几列星星:');  
var str = '';  
for (var i = 1; i <= row; i++) {  
    for (j = 1; j <= col; j++) {  
        str += '☆';  
    }  
    str += '\n';  
}  
console.log(str);
```

## 3. 双重 for 循环



### 课堂案例 2：打印倒三角形

```
☆☆☆☆☆☆☆☆☆☆
☆☆☆☆☆☆☆☆
☆☆☆☆☆☆☆☆
☆☆☆☆☆☆☆☆
☆☆☆☆☆☆
☆☆☆☆
☆☆☆☆
☆☆☆☆
☆☆
☆☆
☆☆
```

---



## 3. 双重 for 循环



### 案例分析

- ① 一共有10行，但是每行的星星个数不一样，因此需要用到双重 for 循环
- ② 外层的 for 控制行数 i，循环10次可以打印10行
- ③ 内层的 for 控制每行的星星个数 j
- ④ 核心算法：每一行星星的个数  $j = i$ ；  $j \leq 10$ ；  $j++$
- ⑤ 每行打印完毕后，都需要重新换一行



## 3. 双重 for 循环



### 实现代码

```
var row = prompt('请输入您打印几行星星:');  
var col = prompt('请输入您打印几列星星:');  
var str = '';  
for (var i = 1; i <= row; i++) {  
    for (j = 1; j <= col; j++) {  
        str += '☆';  
    }  
    str += '\n';  
}  
console.log(str);
```

### 3. 双重 for 循环



思考：打印正三角形

```
☆
☆☆
☆☆☆
☆☆☆☆
☆☆☆☆☆
☆☆☆☆☆☆
☆☆☆☆☆☆☆
☆☆☆☆☆☆☆☆
☆☆☆☆☆☆☆☆☆
☆☆☆☆☆☆☆☆☆☆
```

### 3. 双重 for 循环



#### 课堂案例3：打印九九乘法表

九九乘法表

|       |        |        |        |        |        |        |        |        |  |
|-------|--------|--------|--------|--------|--------|--------|--------|--------|--|
| 1x1=1 |        |        |        |        |        |        |        |        |  |
| 1x2=2 | 2x2=4  |        |        |        |        |        |        |        |  |
| 1x3=3 | 2x3=6  | 3x3=9  |        |        |        |        |        |        |  |
| 1x4=4 | 2x4=8  | 3x4=12 | 4x4=16 |        |        |        |        |        |  |
| 1x5=5 | 2x5=10 | 3x5=15 | 4x5=20 | 5x5=25 |        |        |        |        |  |
| 1x6=6 | 2x6=12 | 3x6=18 | 4x6=24 | 5x6=30 | 6x6=36 |        |        |        |  |
| 1x7=7 | 2x7=14 | 3x7=21 | 4x7=28 | 5x7=35 | 6x7=42 | 7x7=49 |        |        |  |
| 1x8=8 | 2x8=16 | 3x8=24 | 4x8=32 | 5x8=40 | 6x8=48 | 7x8=56 | 8x8=64 |        |  |
| 1x9=9 | 2x9=18 | 3x9=27 | 4x9=36 | 5x9=45 | 6x9=54 | 7x9=63 | 8x9=72 | 9x9=81 |  |





## 3. 双重 for 循环



### 案例分析

- ① 一共有9行，但是每行的个数不一样，因此需要用到双重 for 循环
- ② 外层的 for 循环控制行数  $i$ ，循环9次，可以打印 9 行
- ③ 内层的 for 循环控制每行公式  $j$
- ④ 核心算法：每一行 公式的个数正好和行数一致， $j \leq i$ ;
- ⑤ 每行打印完毕，都需要重新换一行
- ⑥ 把公式用  $i$  和  $j$  替换

## 3. 双重 for 循环



### 实现代码

```
var str = ''
for (var i = 1; i <= 9; i++) { // 外层for控制 行数 9行
    for (var j = 1; j <= i; j++) { // j 控制列数    列数和行数是一样的    j <= i
        str += j + " x " + i + " = " + i * j + '\t';
    }
    str += '\n';
}
console.log(str);
```



## 3. 双重 for 循环

### 3.5 for 循环小结

- for 循环可以重复执行某些相同代码
- for 循环可以重复执行些许不同的代码，因为我们有计数器
- for 循环可以重复执行某些操作，比如算术运算符加法操作
- 随着需求增加，双重for循环可以做更多、更好看的效果
- 双重 for 循环，外层循环一次，内层 for 循环全部执行
- for 循环是循环条件和数字直接相关的循环
- 分析要比写代码更重要
- 一些核心算法想不到，但是要学会，分析它执行过程
- 举一反三，自己经常总结，做一些相似的案例

# 目录 Contents

- ◆ 循环
- ◆ for 循环
- ◆ 双重 for 循环
- ◆ while 循环
- ◆ do while 循环
- ◆ continue break

## 4. while 循环

while 语句可以在条件表达式为真的前提下，循环执行指定的一段代码，直到表达式不为真时结束循环。

while语句的语法结构如下：

```
while (条件表达式) {  
    // 循环体代码  
}
```

### 执行思路：

- ① 先执行条件表达式，如果结果为 true，则执行循环体代码；如果为 false，则退出循环，执行后面代码
- ② 执行循环体代码
- ③ 循环体代码执行完毕后，程序会继续判断执行条件表达式，如条件仍为true，则会继续执行循环体，直到循环条件为 false 时，整个循环过程才会结束

## 4. while 循环

while 语句可以在条件表达式为真的前提下，循环执行指定的一段代码，直到表达式不为真时结束循环。

while语句的语法结构如下：

```
while (条件表达式) {  
    // 循环体代码  
}
```

### 注意：

- ① 使用 while 循环时一定要注意，它必须要有退出条件，否则会成为死循环
- ② while 循环和 for 循环的不同之处在于 while 循环可以做较为复杂的条件判断，比如判断用户名和密码

## 4. while 循环



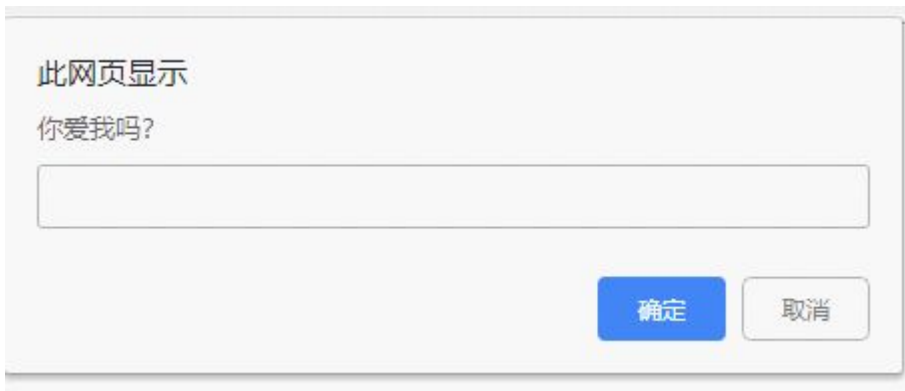
### 课堂案例 1:

- ① 打印人的一生, 从1岁到100岁
- ② 计算 1 ~ 100 之间所有整数的和

## 5. while 循环

### 课堂案例 2：询问你爱我吗

弹出一个提示框，你爱我吗？如果输入我爱你，就提示结束，否则，一直询问。



此网页显示

你爱我吗?

确定 取消



## 5. do while 循环



### 案例分析

- ① 弹出输入框，要求用户输入。
- ② 判断条件比较复杂我们使用 while 循环。
- ③ while 循环语句中的条件表达式只要输入的不是 我爱你，就一直循环。

# 目录 Contents

- ◆ 循环
- ◆ for 循环
- ◆ 双重 for 循环
- ◆ while 循环
- ◆ do while 循环
- ◆ continue break



## 5. do while 循环

do... while 语句其实是 while 语句的一个变体。该循环会先执行一次代码块，然后对条件表达式进行判断，如果条件为真，就会重复执行循环体，否则退出循环。

do... while 语句的语法结构如下：

```
do {  
    // 循环体代码 - 条件表达式为 true 时重复执行循环体代码  
} while (条件表达式);
```

### 执行思路：

- ① 先执行一次循环体代码
- ② 再执行条件表达式，如果结果为 true，则继续执行循环体代码，如果为 false，则退出循环，继续执行后面代码

**注意：**先再执行循环体，再判断，我们会发现 do...while 循环语句**至少会执行一次循环体代码**

## 5. do while 循环



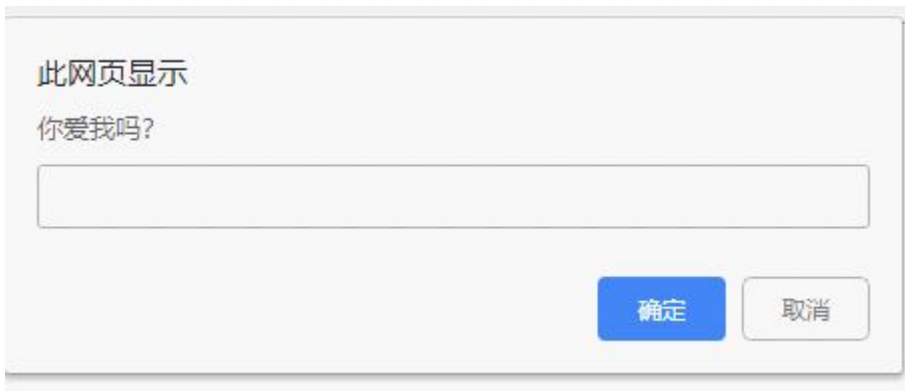
### 课堂案例 1:

- ① 打印人的一生, 从1岁到100岁
- ② 计算 1 ~ 100 之间所有整数的和

## 5. do while 循环

### 课堂案例 2：询问你爱我吗

弹出一个提示框，你爱我吗？如果输入我爱你，就提示结束，否则，一直询问。



此网页显示

你爱我吗?

确定 取消

## 5. do while 循环



### 案例分析

- ① 弹出输入框，要求用户输入。
- ② 判断条件我们使用 do...while 循环。
- ③ do... while 循环语句中的条件表达式只要输入的不是我爱你，就一直循环。

## 5. do while 循环



### 实现代码

```
do {  
    var love = prompt('你爱我吗? ');  
} while (love != '我爱你')  
alert('登录成功');
```



## 5. do while 循环

### 循环小结

- JS 中循环有 for 、while 、 do while
- 三个循环很多情况下都可以相互替代使用
- 如果是用来计次数，跟数字相关的，三者使用基本相同，但是我们更喜欢用 for
- while 和 do...while 可以做更复杂的判断条件，比 for 循环灵活一些
- while 和 do...while 执行顺序不一样，while 先判断后执行，do...while 先执行一次，再判断执行
- while 和 do...while 执行次数不一样，do...while 至少会执行一次循环体，而 while 可能一次也不执行
- 实际工作中，**我们更常用for 循环语句**，它写法更简洁直观，所以这个要重点学习



# 目录 Contents

- ◆ 循环
- ◆ for 循环
- ◆ 双重 for 循环
- ◆ while 循环
- ◆ do while 循环
- ◆ continue break

## 6. continue break

### 6.1 continue 关键字

**continue 关键字**用于立即**跳出本次循环，继续下一次循环**（本次循环体中 continue 之后的代码就会少执行一次）。

例如，吃5个包子，第3个有虫子，就扔掉第3个，继续吃第4个第5个包子，其代码实现如下：

```
for (var i = 1; i <= 5; i++) {  
    if (i == 3) {  
        console.log('这个包子有虫子，扔掉');  
        continue; // 跳出本次循环，跳出的是第3次循环  
    }  
    console.log('我正在吃第' + i + '个包子呢');  
}
```



## 6. continue break

### 6.2 break 关键字

**break 关键字**用于立即跳出整个循环（循环结束）。

例如，吃5个包子，吃到第3个发现里面有半个虫子，其余的不吃了，其代码实现如下：

```
for (var i = 1; i <= 5; i++) {  
    if (i == 3) {  
        break; // 直接退出整个for 循环，跳到整个for下面的语句  
    }  
    console.log('我正在吃第' + i + '个包子呢');  
}
```



传智播客旗下高端IT教育品牌