JavaScript编程语句

管理科学与工程学科
耿方方

主要内容

- 条件语句
- 循环语句

- 观察一个流程图，除了最基本的脚本之外，还能看到脚本有多个不同的路径，这就意味着浏览器在不同的情况下会执行不同的代码。
- 主要根据三种情况来执行代码：

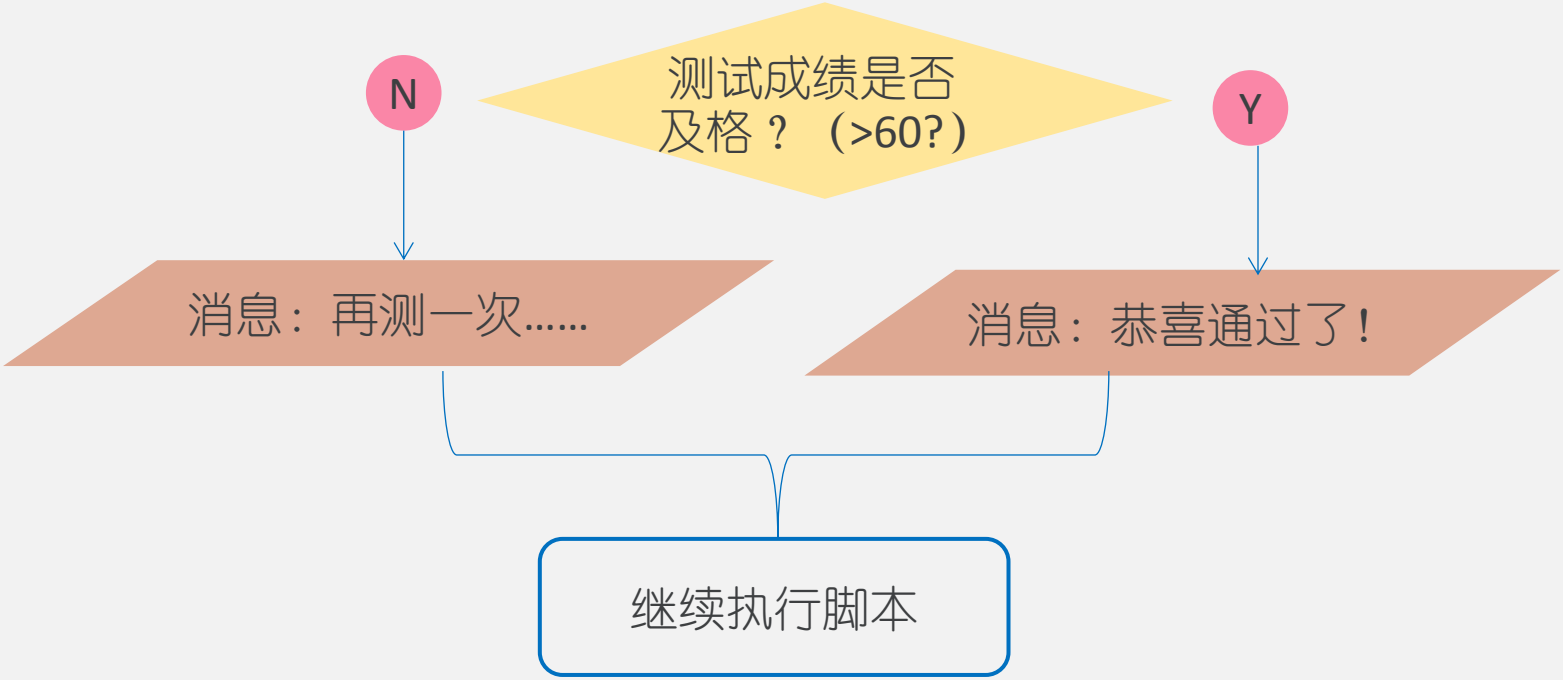
判断：对值进行分析，判断他们是否符合预期结果。

决策：根据评估结果，可以决定脚本接下来要采取哪种路径。

循环：需要重复执行同样的一组步骤。

条件语句

- 在一段程序中，通常有好几处需要进行判断，以决定接下来要运行哪些代码，使用流程图可以有效规划行为：



条件语句

- 判断由两部分组成：

1：进行判断的表达式，它会返回一个值。

2：一个条件语句，用于说明在某种特定情况下应该执行什么操作。

```
if (score>50) {
   document.write( ‘you passed!’ );
  }else{
    document.write( ‘try again…’ );
```

条件语句

- 在编写代码时，经常需要根据不同的条件完成不同的行为。可以在代码中使用条件语句来完成这个任务。
- 在 JavaScript 中，我们可以使用下面几种条件语句：
- (1) if 语句

  在一个指定的条件成立时执行代码。
- (2) if...else 语句

  在指定的条件成立时执行代码，当条件不成立时执行另外的代码。
- (3) if...else if....else 语句

  使用这个语句可以选择执行若干块代码中的一个。

条件语句

- (1) if 语句

if语句是最常见的条件语句，基本语法如下所示。条件必须放在if后面的圆括号中。条件的求值结果永远是一个布尔值，即只能为true或false。花括号中的语句不管内容有多少条，只有在给定条件的求值结果为true的情况下才会执行。

```
if(condition){
    //执行语句内容
}
```

```
if(1+1=3){
    alert("It's wrong");
}
```

条件语句

if语句

- (1) if 语句

  实例1:

```
<html>
    <head>
        <title>if语句</title>
        <meta charset="UTF-8">
</head>
    <body>
        <section id="page1">
            <h1>if语句</h1>
            <section id="answer"></section>
        </section>
    <script src="demo3-01-1.js"></script>
    </body>
</html>
```

```
var score = 75;     // Score
var msg;            // Message

if (score >= 50) {
  msg = 'Congratulations!';
  msg += ' Proceed to the next round.';
}

var el =
document.getElementById('answer');
el.textContent = msg;
```

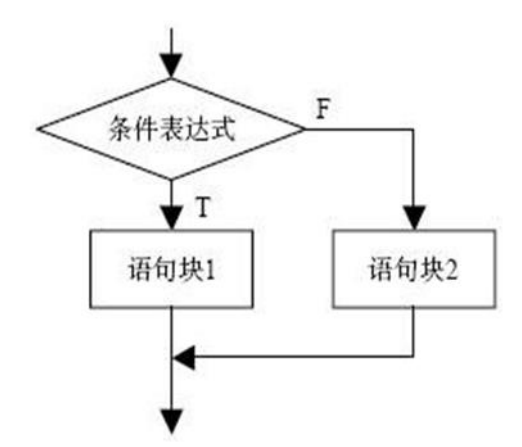
条件语句

- (2) if....else语句

如果希望条件成立时执行一段代码，而条件不成立时执行另一段代码，那么可以使用 if....else 语句。

语法:

```
if (条件)
{
条件成立时执行此代码
}
else
{
条件不成立时执行此代码
}
```



条件语句

if...else语句

- (2) if•••else 语句

实例2:

```
<html>
    <head>
        <title>if-else语句</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <section id="page1">
            <h1>if-else语句</h1>
            <section id="answer"></section>
        </section>
    <script src="demo3-02-1.js"></script>
    </body>
</html>
```

```
var pass = 50;      // Pass mark
var score = 75;    // Current score
var msg;           // Message
if (score > pass) {
  msg = 'Congratulations, you
passed!';
} else {
  msg = 'Have another go!';
}

var el =
document.getElementById('answer');
el.textContent = msg;
```

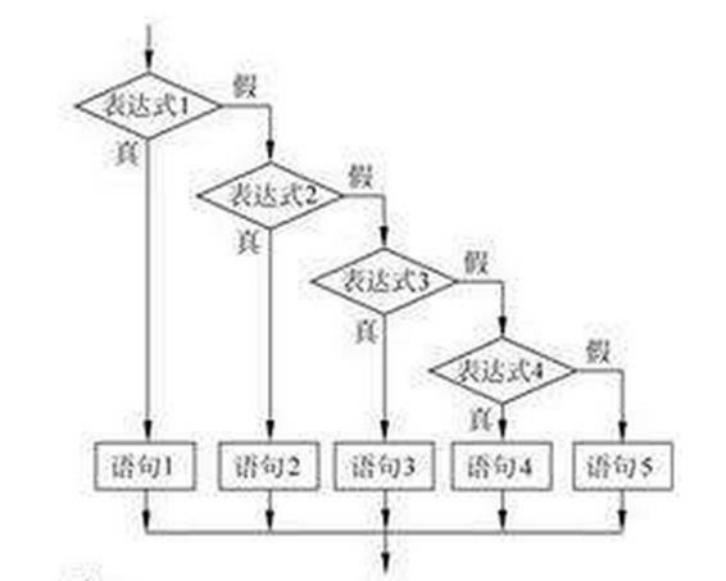
条件语句

- (3) if...else if....else语句

当需要选择多套代码中的一套来运行时，请使用 if....else if...else 语句。

语法:

```
if（条件1）
{
条件1成立时执行代码
}
else if（条件2）
{
条件2成立时执行代码
}
else
{
条件1和条件2均不成立时执行代码
}
```



条件语句

if...else if....else语句

- (3) if...else if....else语句

实例3:

```
<html>
    <head>
        <title>if-else if-else</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <h1>if-else if-else</h1>
        <script type="text/javascript">
            var d = new Date()
            var time = d.getHours()
            if (time<10) {
                 document.write("<b>Good morning</b>")}
             else if (time>10 && time<16){
                 document.write("<b>Good day</b>")}
            else {
                 document.write("<b>Hello World!</b>")}
          </script>
    </body>
</html>
```

条件语句

if语句的嵌套

- If语句不但可以单独使用，而且可以嵌套应用，即在if语句的从句部分嵌套另外一个完整的if语句。在if语句中嵌套使用if语句，其外层if语句的从句部分的大括号“{}”可以省略。
- 在使用应用嵌套的if语句时，最好使用大括号“{}”来确定相互之间的层次关系。否则由于大括号“{}”使用位置不同，可能导致程序代码的含义完全不同，从而输出不同的内容。

条件语句

if语句的嵌套

- 例如:
- var m=16;n=9;
- If(m<1){
- if(n==1)
- alert(“判断结果: M小于1, N等于1”);
- else
- alert(“判断结果: M小于1, N不等于1”);
- }else if(m>10){
- if(n==1)
- alert(“判断结果: M大于10, N等于1”);
- else
- alert(“判断结果: M大于10, N不等于1”);
- }

条件语句

if语句的嵌套

- 例如:
- var m=16;n=9;
- If(m<1){
- if(n==1)
- alert(“判断结果: M小于1, N等于1”);
- else
- alert(“判断结果: M小于1, N不等于1”);
- }else if(m>10){
- if(n==1)
- alert(“判断结果: M大于10, N等于1”);
  }else
- alert(“判断结果: M大于10, N不等于1”);

条件语句

- switch语句
- 一个if语句会在程序的执行流程中产生一个分支，但是当程序含有多个分支，并且所有的分支都依赖于一个变量的值时，多个if语句重复性的检测同一个变量的值将会产生资源浪费。而switch语句正是用来处理这种情况的，它比重复使用if语句要高效的多，其语法结构如下所示：

```
switch(expression){
    //执行代码内容
}
```

条件语句

- switch语句
- 在执行代码内容中，不同的位置要使用case关键字后加一个值和一个冒号来标记。当执行一个switch语句时，它先计算expression的值，然后查找与这个值匹配的case标签，找到相应的case标签，就开始执行case标签后的代码块语句，如果没有相匹配的内容，就开始执行标签default后的语句，如果没有default标签，就跳过所有的代码块。

```
switch(n)
    {
    case 1:
      执行代码块 1
      break
    case 2:
      执行代码块 2
      break
    default:
    如果n即不是1也不是2，则执行此代码
    break
    }
```

条件语句

- Switch语句
- 案例4：
- <html>
- <head>
- <title>switch语句</title>
- <meta charset="UTF-8">
- </head>
- <body>
- <section id="page1">
- <h1>switch语句</h1>
- <section id="answer"></section>
- </section>
- <script src="demo3-04-1.js"></script>
- </body>
- </html>

switch语句

```
var msg;
level = 2;
switch (level) {
case 1:
    msg = 'Good luck on the first test';
    break;
case 2:
    msg = 'Second of three - keep
going!';
    break;
case 3:
    msg = 'Final round, almost there!';
    break;
default:
    msg = 'Good luck!';
    break;
}
var el =
document.getElementById('answer');
el.textContent = msg;
```

循环语句

- 循环
- 循环会检查一个条件，如果这个条件返回true,那么会执行一段相应的代码。然后这个条件会再被检查，如果依然返回true，那么这段代码会再被执行。这个过程一直重复，直到这个条件返回为false为止。
- 循环共有三种类型：

for：如果需要将一段代码运行特定的次数，那么可以使用for循环。在for循环中，检查的条件通常是一个计数器，这个计数器用来计算循环需要运行多少次。

while：如果不确定代码究竟要被执行多少次，可以使用while循环。这里的判断条件也可以使用计算器之外的形式，只要条件返回true，对应的代码就会一直执行。

do…while：与while非常相似，只有一处关键的区别：在do…while循环中，即使条件返回false，被包裹在花括号之中的语句也至少会执行一次。

循环语句

- for循环
- 在JavaScript中使用for循环来执行一些代码十分方便，其语法结构如下所示：

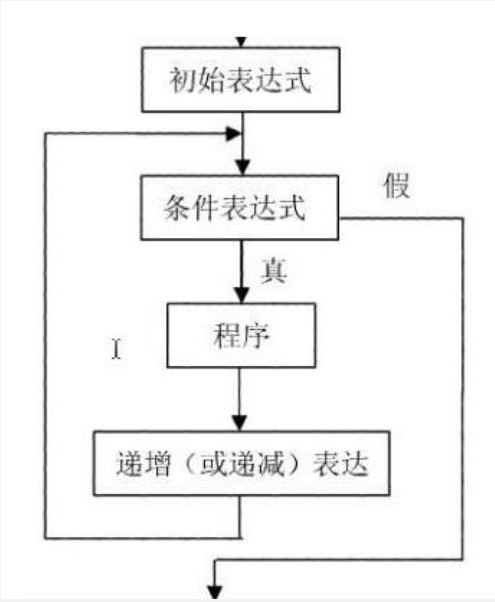
```
for(initial condition;test condition;alter condition){
    //执行语句内容
}
```

- For循环使用计数器作为条件，这种方式让代码执行制定的次数。在这里可以看到这个条件是由三条语句组成的。
- 初始化：创建一个变量，然后赋值为0。这个变量通常命名为i，它起到计数器的作用
- 条件：循环一直重复运行下去，直到计数器达到特定的数值。
- 更新：每次循环执行完花括号内部的语句之后，计数器都会加1。

循环语句

- for循环
- 例如：

```
for(var count=1;count<11;count++){
    alert(count);
}
```



- 用for循环来重复执行一些代码的好处是循环控制结构更加清晰。与循环有关的所有内容都包含在for语句的圆括号里面。

循环语句

- for循环
- 案例5:
- <html>
- <head>
- <title>for循环</title>
- <meta charset="UTF-8">
- </head>
- <body>
- <section id="page1">
- <h1>for循环</h1>
- <section id="answer"></section>
- </section>
- <script src="demo3-05-1.js"></script>
- </body>
- </html>

```
var scores = [24, 32, 17];
var arrayLength = scores.length
var roundNumber = 0
var msg = ‘ ';
for (var i = 0; i < arrayLength; i++) {
roundNumber = (i + 1);
msg += 'Round ' + roundNumber + ': ';
msg += scores[i] + '<br />';
}

document.getElementById('answer').i
nnerHTML = msg;
```

循环语句

- For•••in循环
- For...in 声明用于对数组或者对象的属性进行循环操作。for ... in 循环中的代码每执行一次，就会对数组的元素或者对象的属性进行一次操作。
- 语法:

for（变量 in 对象）
{　　在此执行代码
}

```
for(variable in object){
    //执行语句内容
}
```

“变量”用来指定变量，指定的变量可以是数组元素，也可以是对象的属性。

循环语句

- for循环
- 案例6:
- <html>
- <head>
- <title>for...in</title>
- <meta charset="UTF-8">
- </head>
- <body>
- <script type="text/javascript">
- var x
- var mycars = new Array()
- mycars[0] = "Saab"
- mycars[1] = "Volvo"
- mycars[2] = "BMW"
- for (x in mycars)
- {
- document.write(mycars[x] + "
")
- }
- </script>
- </body>
- </html>

循环语句

- while循环
- while循环语句与if语句十分相似，它们的语法几乎一样，其语法结构如下所示：

```
while(expression){
    //执行语句内容
}
```

- 例如：

```
var count=1;
while(count<11){
    alert(count);
    count++;
}
```

- while循环语句与if语句唯一的区别是：只要给定条件的求值结果是true，包含在花括号里的代码就将反复执行下去。

循环语句

- while循环
- 案例7:
- <html>
- <head>
- <title>while语句</title>
- <meta charset="UTF-8">
- </head>
- <body>
- <section id="page1">
- <h1>while语句</h1>
- <section id="answer"></section>
- </section>
- <script src="demo3-07-1.js"></script>
- </body>
- </html>

```
var i = 1;       // Set counter to 1
var msg = '';    // Message

while (i < 10) {
  msg += i + ' x 5 = ' + (i * 5) + '<br />';
  i++;
}

document.getElementById('answer').in
nerHTML = msg;
```

循环语句

- Do…while循环

在某些场合，我们希望那些包含在循环语句内部的代码至少执行一次，这时我们便需要使用do循环了，其语法结构如下所示：

```
do{
    //执行语句内容
}while(condition)
```

- 例如：

```
var count=1;
do{
    alert(count);
    count++;
}while(count<11)
```

- do循环与while循环最大的区别便是：对循环控制条件的求值发生在每次循环结束之后。因此，即使循环控制条件的首次求值结果为false，花括号里的语句也至少会被执行一次。

循环语句

- Do•••while循环
- 案例8:
- <html>
- <head>
- <title>do...while循环</title>
- <meta charset="UTF-8">
- </head>
- <body>
- <section id="page1">
- <h1>do...while循环</h1>
- <section id="answer"></section>
- </section>
- <script src="demo3-08-1.js"></script>
- </body>
- </html>

```
var i = 1;       // Set counter to 1
var msg = '';    // Message
do {
  msg += i + ' x 5 = ' + (i * 5) + '<br />';
  i++;
} while (i < 1);

document.getElementById('answer').in
nerHTML = msg;
```

循环语句

- 在JavaScript中break语句会使运行的程序立刻退出包含在最内层的循环或者退出一个switch语句，其语法结构如下所示：

```
break;
```

- 由于其用来退出循环或者switch语句，因此只有当它出现在这些语句当中时，这种形式的break语句才能被解析。

- JavaScript允许关键字break后跟一个标签名，当break和标签一起使用时，它将跳转到这个带有标签的语句的尾部，或者禁止这个语句。该语句可以是任何用括号括起来的语句，它不一定是循环语句或者switch语句。

循环语句

例如：

```
<script>
    var now=new Date();  //获取日期和时间
    var day=now.getDay();
    var week;
    switch (day){
      case1:
      week=“星期一”;
      break;
      case2:
      week=“星期二”;
      break;
      case3:
      week=“星期三”;
      break;
      case4:
      week=“星期四”;
      break;
      case5:
      week=“星期五”;
      break;
      case6:
      week=“星期六”;
      break;
    default:
      week=“星期日”;
      break;
    }
    Document.write(“今天是”+week);
</script>
```

循环语句

countinue语句

- continue语句与break语句相似，不同的是它不是退出一个循环而是开始循环的一次新迭代，其语法结构如下所示：

```
continue;
```

- continue语句只能在while语句、do/while语句、for语句或者for/in语句的循环体中使用，在其它地方使用将不会被解析。

循环语句

countinue语句

- 执行continue语句时，封闭循环的当前迭代就会被终止，开始执行下一次迭代，这对不同类型的循环语句来说含义是不同的：
- 在while循环语句中，会再次检测循环开头的expression，如果值为true，将从头开始执行循环内容；
- 在do/while循环中，会跳到循环的底部，在顶部开始下一次循环之前会在此先检测循环条件；
- 在for循环中，先计算increment表达式，然后再检测test表达式以确定是否应该执行下一次迭代；
- 在for/in循环中，将以下一个赋给循环变量的属性名开始新的迭代。
- 在while循环和for循环中continue语句行为的不同之处在于，while循环是直接跳到循环条件处，而在for循环中则要先计算increment表达式，然后再跳转到循环条件处。

循环语句

countinue语句

- 例如:

```
<script>
      var sum=0;
      for(i=0;i<100;i++){
        if(i%2==0){
          sum=sum+i;}
         else{
            continue;
               }
         }
     alert(“100以内所有偶数的和为:”+sum);
</script>
```

实例

- 实例9:
- <html>
- <head>
- <title>条件循环实例</title>
- <meta charset="UTF-8">
- </head>
- <body>
- <section id="page2">
- <h1>条件循环实例</h1>
- <section id="blackboard"></section>
- </section>
- <script src="demo3-09-1.js"></script>
- </body>
- </html>

```
var table = 3;            // Unit of table
var operator = 'addition';
var i = 1;
var msg = '';             // Message
if (operator === 'addition') {
while (i < 11) {
   msg += i + ' + ' + table + ' = ' + (i + table)
+ '<br />';
   i++;
 }
} else {
while (i < 11) {
   msg += i + ' x ' + table + ' = ' + (i * table)
+ '<br />';
   i++;
 }
}
var el =
document.getElementById('blackboard');
el.innerHTML = msg;
```

想一想

1、判断用户是否输入了用户名和密码？

2、用for循环语句开发一个乘法口诀表，并将算式以及计算结构打印在特定的表格中。