**流程控制语句：**

**3.循环语句:**

需要重复执行统一操作的程序结构称为循环结构

通过判断循环条件反复的执行一段代码多次

创建一个循环，需要三个步骤

1.初始化一个变量

2.在循环中设置一个条件表达式

3.定义一个更新表达式，每次循环结束更新变量

**while语句：**

**语法：**

while(条件表达式){

语句...（循环图）

}

**执行流程：**

先对条件表达式进行求值判断,

如果值为true，则执行循环体，

循环体执行完毕以后，继续对表达式进行判断

如果还为true，则继续执行循环体，以此类推

直到值为false，则终止循环

**do-while语句：**

**语法：**

do{

语句...

}while(条件表达式)

**执行流程：**

循环体执行完毕以后，在对while后的条件表达式进行判断，

如果结果为true，则继续执行循环体，执行完毕继续判断以此类推

如果结果为false，则终止循环

**while和do-while的区别：**

实际上这两个语句功能类似，不同的是while是先判断后执行,

而do. ..while会先执行后判断，

do . . .while可以保证循环体至少执行一次，

而while不能

**for语句：**

for语句，也是一种循环语句，也称为for循环

for的优点：结构清楚

在for循环中，为我们提供了专门的位置用来放三个表达式:

1.初始化表达式

2.条件表达式

3.更新表达式

**语法:**

for(①初始化表达式;②条件表达式;④更新表达式){

③语句.. .

}

**执行流程：**

①执行初始化表达式，初始化变量（初始化表达式只会执行一次）

②执行条件表达式，判断是否执行循环。

如果为true，则执行循环体③

如果为false，终止循环

④执行更新表达式，更新表达式执行完毕继续重复②

**注：for循环的三个表达式都可以省略，也可以写在外面**

如果在for循环中不写任何的表达式，只写两个;

此时循环是一个死循环会一直执行下去，慎用

将条件表达式写死为true的循环，叫做死循环

该循坏不会停止，除非浏览器关闭，死循环在开发中慎用

可以使用break，来终止循环

**注意：循环嵌套循环，尽量短循环在外层，长循环在内层**

**break和continue**

不能在if语句中使用break和continue

**break关键字**

可以用来退出switch或循环语句

break关键字会立即终止/跳出，离他最近的循环体。

ps.可以理解为当 break为某个循环体的子语句（父子关系）时

执行到break时直接跳出父循环体。

可以为break创建一个label，来标识想要终止的循环体

**语法：label：循环语句**

**使用：在break后跟着—个label**

这样break将会结束指定的循环，而不是最近的

eg. outer:

for(var i=e ; i<5 ; i++){

console.log( "@外层循环"+i);

for( var j=e ; j<5; j++){

break outer;

console.log("内层循环:"+j);

}

}

**continue关键字**

用来跳过当次循环 进入下一次循环条件的判断

同样continue只会对离它最近的循环体起作用

**区别：**

**break结束整个循环**

**continue结束当次循环，不会结束整个循环**

**break可以用于分支结构switch中，continue只能用于循环体**

**break和continue关键字后面的代码都不会被执行**

**代码运行计时器**

测试程序的性能

在程序执行前，开启计时器

**console.time("计时器的名字"")**

可以用来开启一个计时器

//它需要一个字符串作为参数。

这个字符串将会作为计时器的标识console.time( "test" );

**console.time()**

console.timeEnd()用来停止一个计时器，需要一个计时器的名字作为参数

eg.console.timeEnd( "test" );

**Unicode编码**

**在字符串中使用转义字符输入Unicode编码**

\u四位编码

console.log("\u2620");

**在网页中使用Unicode编码**

&#编码; //这里的编码需要的是十进制

**语句/代码块**

我们的程序是由一条一条语句构成的

语句是按照自上向下的顺序一条一条执行的

在js中可以使用{ }来为语句进行分组，

同一个{ }得中的语句我们称为是一组语句，

它们要么都执行，要么都不执行，

一个{ }得中的语句我们也称为叫一个代码块

在代码块的后边就不用再编写;了

js中的代码块，只具有分组的的作用，没有其他的用途

代码块内容的内容，在外部是完全可见的