1. 输入对话框

prompt为用户提供一个输入对话框，用户可以在输入框中输入内容并且点击确定（Enter），程序可以接受输入的内容。



//弹出一个输入框，将用户录入的内容被保存到value变量中告知给计算机。

var value = prompt('提示信息','默认值');

console.log(value);//输入的内容

1. 流程控制
   1. 流程控制简介

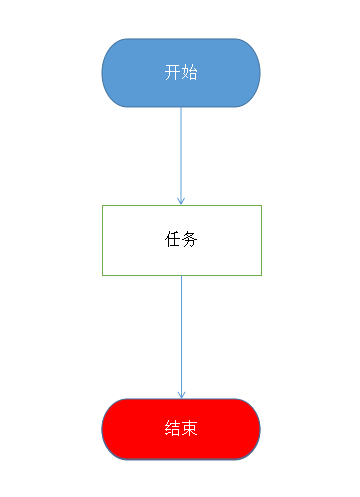
默认情况下，我们编写的每一条JavaScript代码，都会逐行执行，这是最简单的执行流程。但是实际开发中，往往并没有这么简单，我们需要根据某种情况，选择性的执行相应的代码，或者重复执行某段代码，这种情况在现实生活也是存在的。

程序中三种流程控制结构：

默认流程结构：代码逐行执行，最基本的流程。

18岁了；

可以找对象了；



**分支流程结构**：代码执行到某一段的时候，会根据条件判断，如果条件成立则相关执行任务；

现实生活中，如某个一个人做事非常的灵活，如果出现了情况，会进行调整，懂的随机应变。

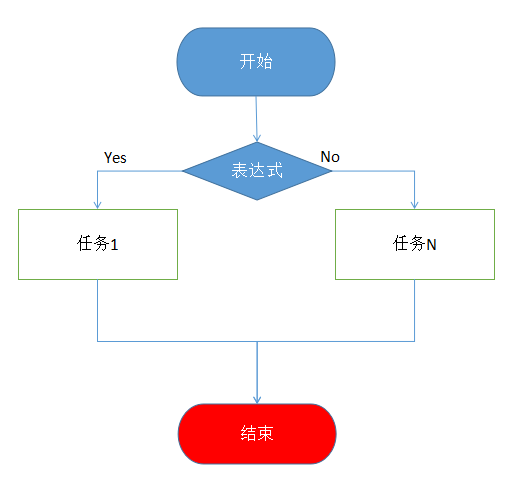
如果

18岁了；

可以找对象了；

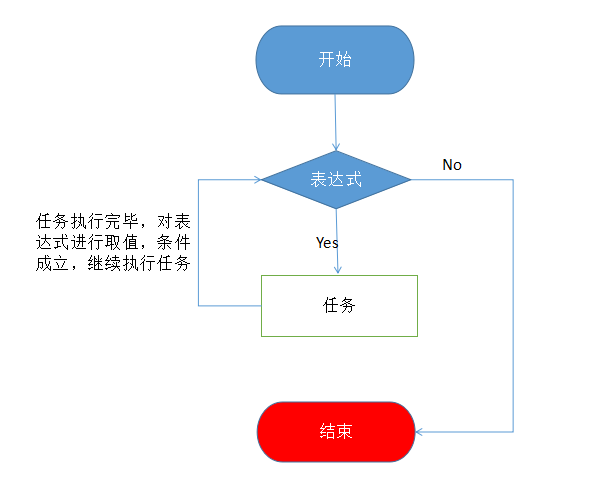
否则

滚粗，小屁孩；



循环流程结构：代码根据条件，重复的执行，但是条件应该保证会执行到某个时间点时不成立（false），或者任务就不会停止。 现实生活中，我们重复所做的事情就就是循环；

重复说1W次“我爱你”；



* 1. 条件语句

条件语句:可以在程序设置一个条件语句，只有满足了这一条件才能让更多的语句得到执行。

* + 1. 条件if语句

语法结构：

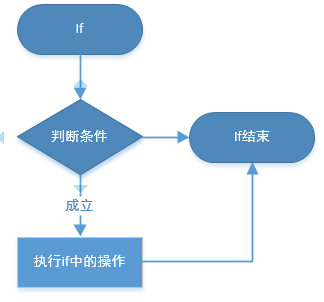
if(条件表达式){

//语句1

//语句....

}

实现生活中，如果你有钱，那么你就可以“开豪车”、“住豪宅”，“开豪车”、“住豪宅”的前提条件是什么，就是有钱，所有我们程序中也是如此，执行某些代码的时候需要有前提条件；



var age = 18;

/\*\*

\*判断是否是一个成年人

\*/

if(age>=18){

console.log(“你是一个成年人”);

}

/\*\*

\* 判断输入数字是否为偶数

\*/

var num = prompt('请输入值：');

if(num%2==0){

console.debug(num+"为偶数");

}

* + 1. 条件分支if else语句

语法结构：

if(条件表达式){

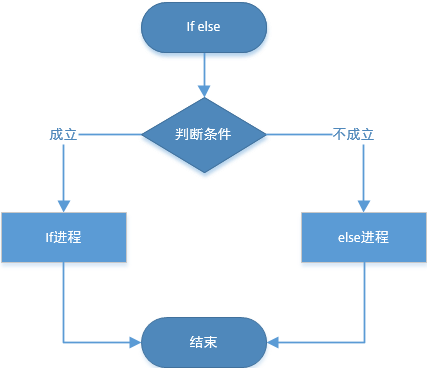
//条件成立的需要执行的语句

}else{

//不成立的需要执行的语句

}

实现生活中，如果你有钱，那么你就可以“开豪车”、“住豪宅”，“开豪车”、“住豪宅”的前提条件是什么，就是有钱，或者你没钱，就只能去“坐公交车”、“租房”；



/\*\*

\* 判断一个数字是否是偶数，否则为奇数。

小学生思维：如果一个数字，不是偶数，那么就是奇数。

\*/

var num = prompt('请输入值：');

if(num%2==0){

document.write(num+"为偶数");

}else{

document.write(num+"为奇数");

}

* + 1. 条件分支if else if else语句

语法结构：

if(条件1){

//语句...

//语句...

}else if(条件2){

//语句...

//语句...

}else if(条件N){

//语句...

//语句...

}else{

//条件都不满足的语句

}

语法结构：

if(条件1){

//语句...

//语句...

}else if(条件2){

//语句...

//语句...

}else if(条件N){

//语句...

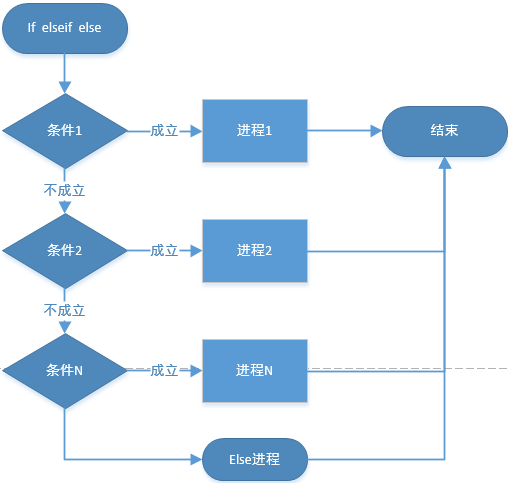
//语句...

}

else 不是必须的，根据功能选择是否使用。

实现生活中，如果你有钱，但钱的多少也一样决定了你能开什么车，住什么房，例如：

你有如果有你20W以上你可以买进口车，如果有你10W以上你可以买合资车，如果你只有5W只可以买国产车，如果你连5W都没有，那就买不了车。



/\*\*

\* 60分及格，70分中等，80分良好，90分优秀

对话框输入一个分数，在页面上输出级别

\*/

var score = prompt('请录入分数');

var result;

if(score>=90){

result = '优';

}else if(score>=80){

result = '良';

}else if(score>=60){

result = '及格';

}else{

result = '不及格';

}

document.write(result);

通过if elseif else完成业务需求时注意条件的顺序.

以下代码虽然语法正确,但是无法完成业务需求

var score = prompt('请录入分数');

var result;

if(score>=60){

result = '及格';

}else if(score>=80){

result = '良';

}else if(score>=90){

result = '优';

}else{

result = '不及格';

}

document.write(result);

**如果条件有重合，最苛刻的条件放在最前面。**

* + 1. 条件分支switch（选择）语句

switch(表达式){

case 表达式1:

语句1;

break;

case 表达式2:

语句2; ...

break;

case 表达式n:

语句n;

break;

default:

语句n+1;

break;

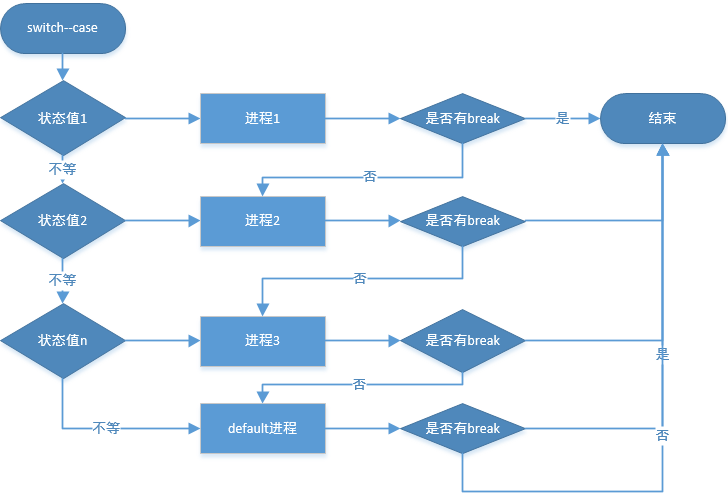
}

switch语句：“如果switch中的表达式等于case 后面的个值(value)，则执行后面的语句，截止break；或switch的右花括号（}）”。

default:默认语句，如果switch中的表达式与case后面的值都不同等，那么默认执行default后面的语句，default不是必须的。

break：关键字会导致代码执行流跳出switch语句。

switch与if类似，但是switch比较适合用于对某个状态值的判断，而if则是针对稍复杂的判断，但是其实switch与if可以相互转换。



/\*\*

\* 输入1--4分别再对应输出 春夏秋冬 ，其他的输出 “未知季节”

if版本

\*/

var num = prompt('请输入:');

if(num==1){

console.log(“春”);

}else if(num==2){

console.log(“夏”);

}else if(num==3){

console.log(“秋”);

}else if(num==4){

console.log(“冬”);

}else{

conosle.log(“未知季节”);

}

/\*\*

\* 输入1--4分别再对应输出 春夏秋冬 ，其他的输出 “未知季节”

switch版本

\*/

var num = prompt('请输入:');

switch (num){

case 1:

document.write('春');

break;

case 2:

document.write('夏');

break;

case 3:

document.write('秋');

break;

case 4:

document.write('冬');

break;

default:

document.write('未知季节');

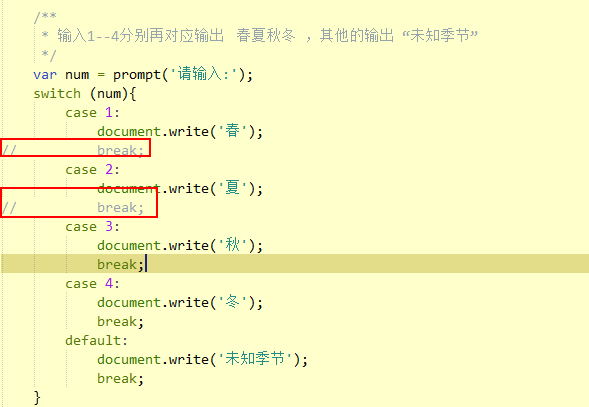
break;

}

* + 1. switch特点(了解)
       1. switch之后必须是case或者是default



* + - 1. break可以不要,但是后面的case就不在判断,直到遇到break或者switch语句执行完毕才结束



* + - 1. default可以出现在switch的任何位置.但还是其他没有匹配上才执行default；



* + 1. if语句和switch语句比较(了解)

switch是状态分支语句,switch中的状态值和case后的值作比较,成立才执行

if是条件分支,值判断if中的条件，条件中可以是范围.

**if中条件可以是一个范围；**

switch条件只能某个值判断；

/\*\*

\* 根据用户输入的月份，告知计算机帮我们输出对应的季节

\* 3,4,5月==> 春季

\* 6,7,8月==> 夏季

\* 9,10,11月==> 秋季

\* 12,1,2月==> 冬季

\* 其他的月份==> 输入错误，请重新输入

\*/

var month = prompt('请输入月份:');

var result;

if(month>=3 && month<=5){

result = '春季'

}else if(month>=6 && month<=8){

result = '夏季'

}else if(month>=9 && month<=11){

result = '秋季'

}else if(month==12 || (month>=1 && month<=2)){

result = '冬季'

}else{

result = "输入错误，请重新输入";

}

document.write(result);

var month = prompt('请输入月份:');

var result;

switch (month){

case 3:

case 4:

case 5:

result = "春季";

break;

case 6:

case 7:

case 8:

result = "夏季";

break;

case 9:

case 10:

case 11:

result = "秋季";

break;

case 12:

case 1:

case 2:

result = "冬季";

break;

default:

result = "输入错误，请重新输入";

break;

}

* + 1. switch和if的选择

switch表示状态分支,与有限个状态值作比较. 可以通过case写出状态值和switch中的值做比较. 状态值太多的话写太多的case不现实.

条件分支表示区间条件,一般判断的是一个范围.

* + 1. 条件分支应用

根据不同的场景选择不同的语句让计算机完成不同的行为。套用上面的语法结构实现即可。

如果你的需求中，存在：“如果什么，就干什么，或者就做什么”，那么就是该使用条件分支的时候；

* 1. 循环语句

你常常希望反复执行同一段代码。我们可以使用循环来完成这个功能，这样就用不着重复地写若干行相同的代码，但是只有满足条件时才反复执行下去。

* + 1. for循环

for(初始化表达式;条件表达式;条件改变表达式){

循环体

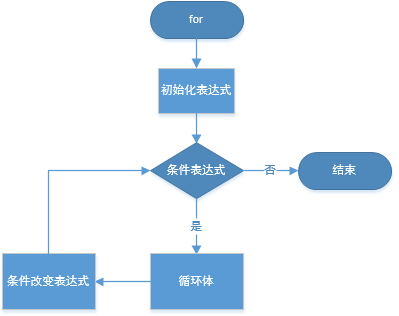
}

初始化表达式：记录循环的次数；

条件表达式：判断是否执行循环，只有条件满足才进行循环；

条件改变表达式：修改条件表达式中的变量；

循环体：需要多次执行的语句；



循环要素：

1. 记录循环次数；var i=0;
2. 循环的条件； i<10;
3. 修改循环条件表达式；i++;
4. 循环的语句；console.log(“i love you”)；

/\*\*

输出：100次“i love you”

\*/

for(var i=0;i<100;i++){

console.log(“i love you”);

}

/\*\*

\* 输出：1-10的数字

\*/

for(var i=1;i<=10;i++){

console.log(i);

}

* + 1. for循环深入

for(初始化表达式A;条件表达式B;条件改变表达式D){

循环体C

}

**初始化表达式A：**

在for语句开始时执行并且执行一次，主要是初始化一些数据，这些数据在后面for循环的其他表达式中要使用。

**条件表达式B：**

条件表达式的结果为boolean类型的值;

true: 执行循环体C

false: for语句执行完毕，不在循环

**循环体C:**

要反复执行的代码。

**条件改变表达式D：**

**循环体C**执行后，该代码执行，通常用来修改变量的值后， **条件表达式B**再判断是否成立。

* + 1. for循环练习
       1. 输出h1到h6的标题
       2. 求解1—100和
       3. 求解 1-100 奇数和
       4. 求解1-100中能被3整除的所有数字的和
       5. 求解1-100 能被3和5同时整除的数字的和
       6. 使用for循环输出2000年到2100年之间的闰年

闰年：400的倍数 或者 是4的倍数不是100的倍数

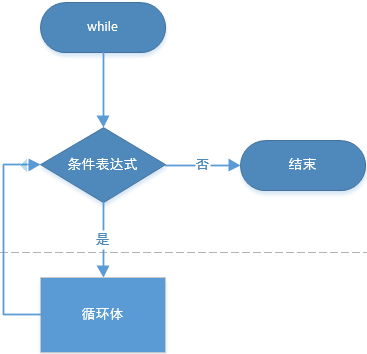
* + 1. while循环

while(条件表达式){

循环体语句

}

while：循环执行代码，如果while()条件表达式为真，将执行while花括号{}中的语句，直到条件不为真时，才会终止while运行。



问题：打印10次“我爱你”；

逗比写法：

console.log(“我爱你”);

console.log(“我爱你”);

console.log(“我爱你”);

console.log(“我爱你”);

console.log(“我爱你”);

console.log(“我爱你”);

console.log(“我爱你”);

console.log(“我爱你”);

console.log(“我爱你”);

console.log(“我爱你”);

懵逼写法：

var count = 0;//记录输出了几次；

while(count<10){//判断输出次数是否小于10次；

console.log(“我爱你”);

}

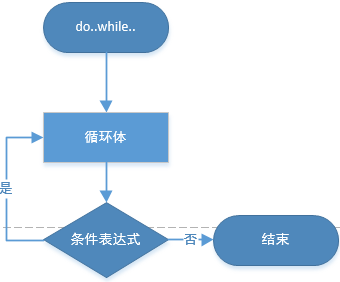
* + 1. do while循环(了解)

do{

循环体

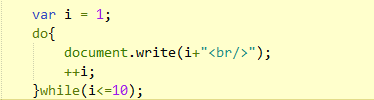
}while(条件表达式);

dowhile循环语句与while类似，在对条件表达式求值之前，循环体内的代码至少会被执行一次。



先执行循环体再做条件判断. 如果条件表达式成立,继续执行循环体.

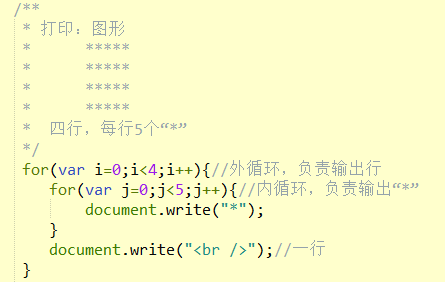
通常是由于条件表达式的判断依赖于循环体的执行才采用do..while

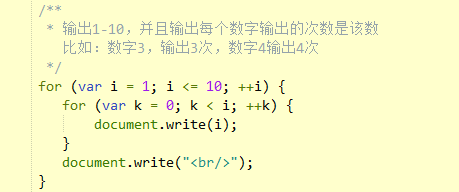


注意do..while()后面一定要加上;号. .

* + 1. 嵌套循环

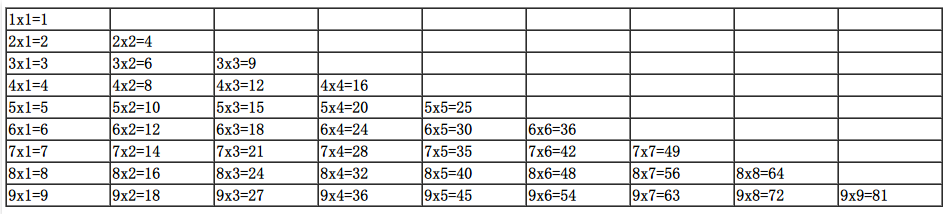
循环中循环

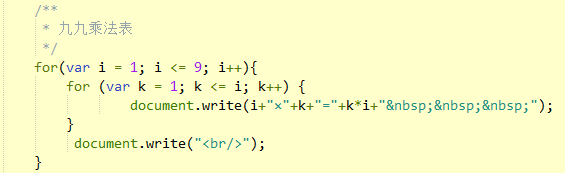




外面的循环每执行一次循环体就会重新执行。 所以说里面的循环要重新全部执行一次。

九九乘法表：





* + 1. 死循环

死循环：循环条件表达式永远为真，循环体语句一直执行，这种情况一定不要出现在你的代码中，后果自负；

while(true){

console.log(“浪”);

}

for(**;;**){

console.log(“浪”);

}

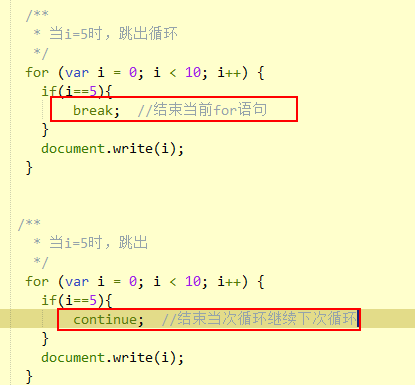
* 1. 终止循环语句break,continue
     1. break,continue用法(掌握)

循环在条件不满足时将会终止.

也可以通过break,continue控制循环.

break: 跳出当前循环

continue: 终止本次循环,继续下次循环.



跳出多层循环:

默认情况下值跳出一层循环。如果需要跳出外层循环，需要在break或者是continue后增加跳出的标记. 跳出指定标记的循环。

