# Javascript

## JS组成部分

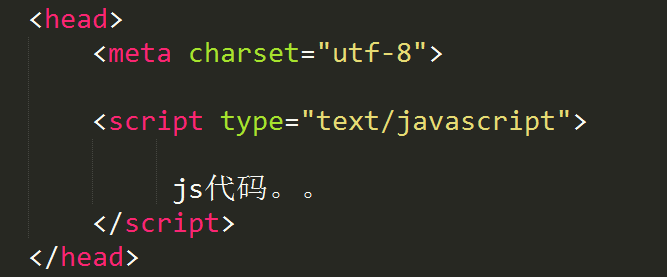
◆ecmascript js标准

◆Dom 通过js操作网页元素

◆bom 通过调用api操作浏览器

## JS书写位置

### 内嵌式写法

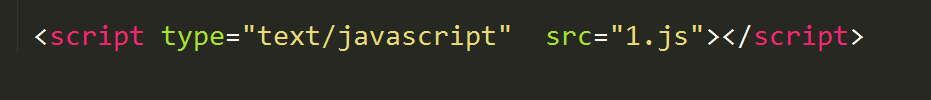


☞ 推荐将script标签放到body标签后面

因为js运行是和代码位置有关。如果js代码在head中，它会先执行js代码，然后显示页面内容。如果js代码过多，则会卡顿的效果。（以下也是可以的）



### 外链式写法



◆新建一个js文件

◆通过script标签中的src属性将外部js引入到html页面中

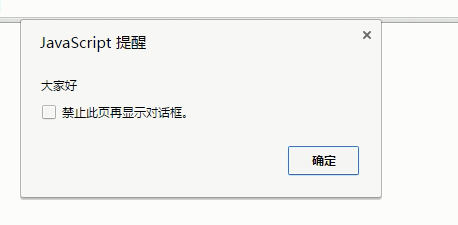
☞ 推荐将多个js文件合并成一个js文件去加载。

## 页面反馈消息（调试的方法）

### 在页面中以弹窗的形式输出消息（确定按钮）

alert(“信息”);

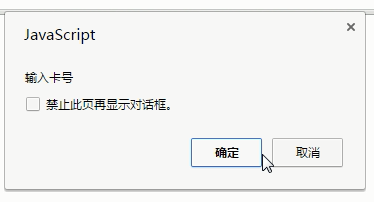
早期用来配合js代码调试。



### 页面输出消息显示 （确定和取消按钮）

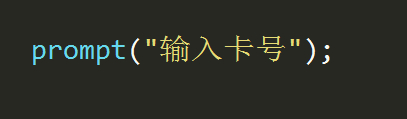
confirm(“消息”);

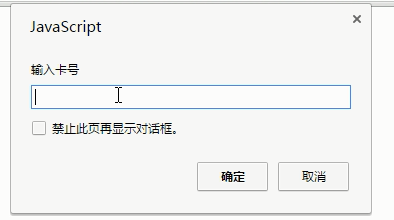
一般和条件判断配合使用



### 接收用户输入信息

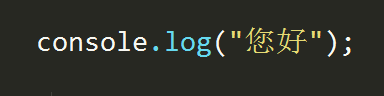
prompt(“信息”);



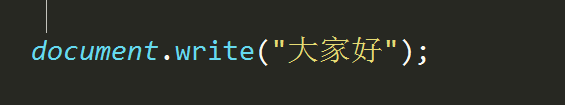


### 控制台显示消息

console.log("您好")



### 在页面中输出消息

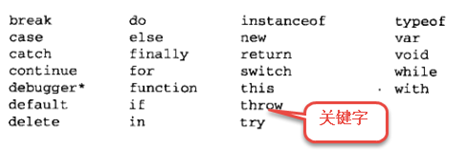
document.write("大家好");

该方法内部可以包含html标签。

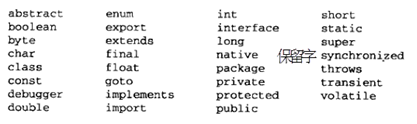
## JS语法要求

* 在js中是严格区分字母大小写
* 在写js代码过程中分号“ ; ”必须的写

### JS关键字



### JS保留字



## JS数据类型

### 简单数据类型

#### number 数字类型

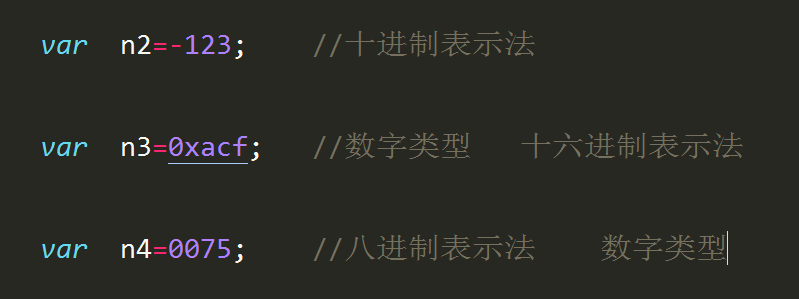
数字类型的表示方式：

◆十进制表示法

◆十六进制表示法

以0x开头 数字0-9 和 a(A)-f(F)

◆八进制表示法

以数字0开头 0-7之间的组合。

#### string 字符串类型

☞ 转义字符

\” 转双引号

\’ 转单引号

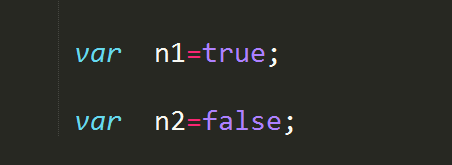
\r 回车

\n 换行

#### boolean 布尔类型

☞ true

☞ false



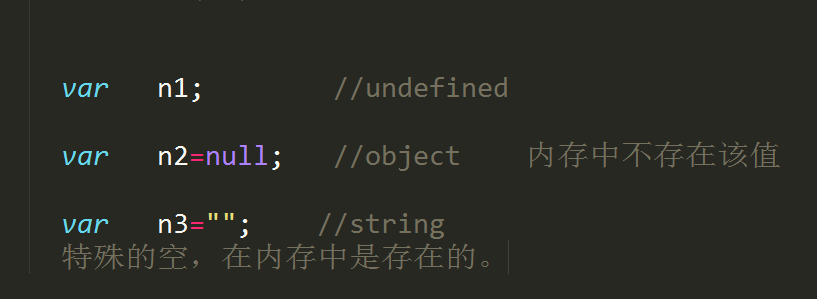
数据类型转换中 数字1：带表true 数字0 代表false

#### undefined 变量未初始化

定义了变量，但是没有过给变量进行赋值。

#### null object

该变量的引用为空

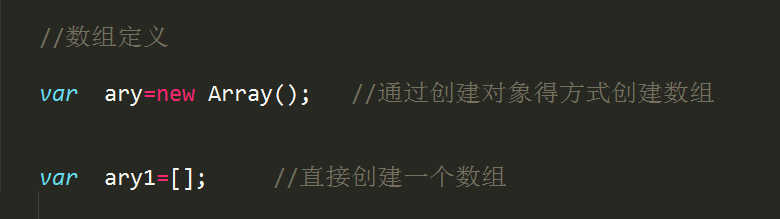


### 复杂数据类型

#### object 对象

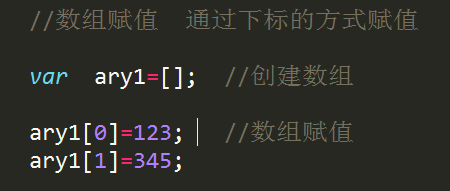
#### Array 数组

定义



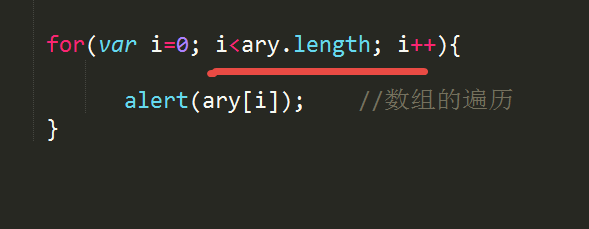
赋值

数组中通过下标的方式进行赋值。下标从0开始。



属性length

length 可以获取数组中的元素个数。



数组合并

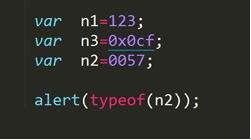


数组返回字符串



### 数据类型判断

typeof(变量)



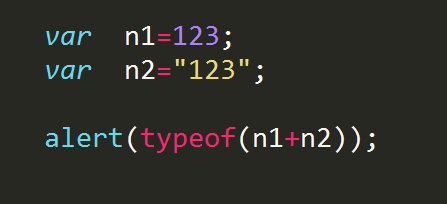
### 数据类型转换

#### 隐式类型转换

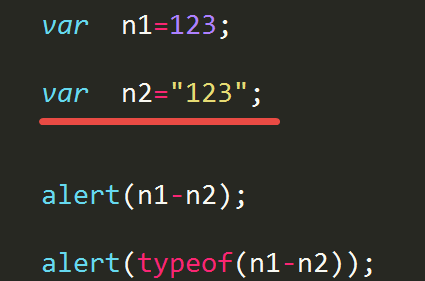
将变量参与到程序运算中，程序将变量进行类型转换

（不是程序员直接进行的类型转换）

* 一个数字类型和一个字符串类型变量相加，最后结果为字符串，加号起的是一个连接的作用



* 如果一个数字类型的变量和一个数字的字符串类型的变量相减，最后结果为数字类型



* 如果一个数字类型的变量和一个非数字类型的变量相减，最后结果为NaN

NaN: not a number 不是一个数字

NaN的数据类型属于number类型。

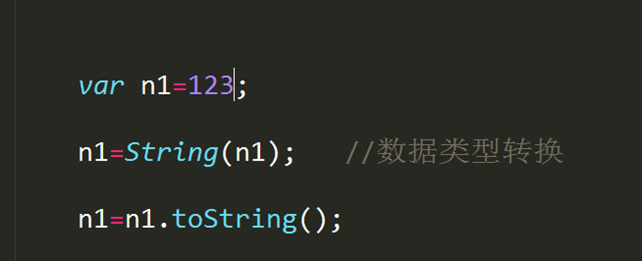
* 如果一个数字类型的变量和一个数字的字符串变量相除，最后结果为数字类型
* 如果一个将数字0作为除数，得出的结果是infinity（无穷大）

infinity的数据类型属于number类型

#### 强制类型转换

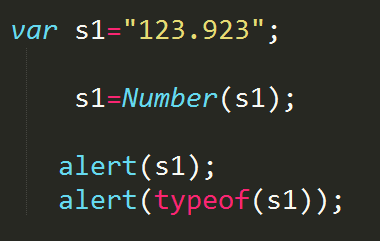
* 将数字转换为字符串

String() .toString();



* 将字符串转换为数字

☞ Number()



非数字类型的字符串最后结果为NaN

☞ parseInt()

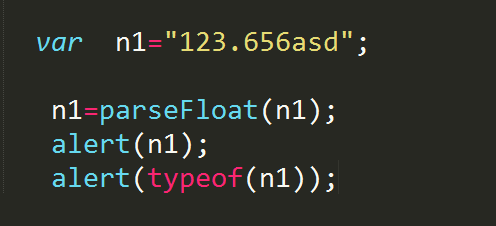
将数字类型的字符串通过该方法转换后，只保留数字的整数部分，不会进行四舍五入运算

非数字类型的字符串转换后的结果为NaN

如果在数字后加上非数字的字符，也可以转换成功

☞ parseFloat()

最后保留的元数据，不会对数字进行四舍五入运算。

如果在数字后加上非数字的字符，也可以转换成功 

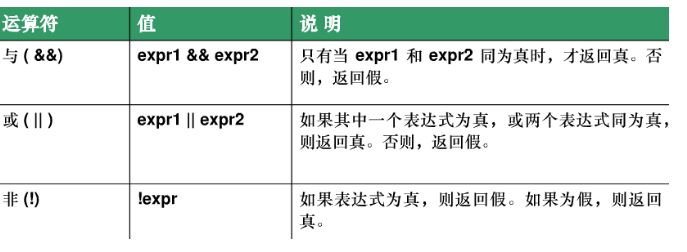
* 将其他类型转换为Boolean类型

可以将数字0 转换为false

其他数据类型通过boolean强制类型转换后结果true

## 运算符

### 逻辑运算符



### 等号运算符

☞ “=” 赋值运算符

☞ “==” 判断是否相同

判断变量是否相同，只判断变量的内容是否相同，与变量的数据 类型无关。

☞ “===” 判断是否相同

判断变量是否相同，除了判断变量的内容还要判断变量的数据类型是否一致。

☞ != 不相等

判断内容是否不相同，不考虑数据类型

☞ !== 不相等

判断的是内容和数据类型是否不相同。

### 逗号运算符



逗号运算符，可以将变量进行集体声明

## 逻辑结构

### switch

Switch(变量n){

case 10:

执行的代码;

break;

case 20:

执行的代码;

break;

default:

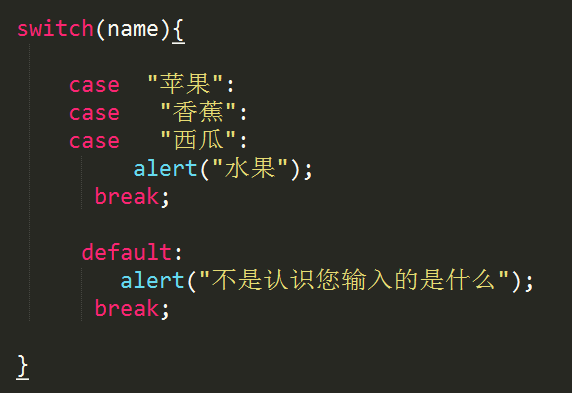
执行的代码;

break;

}

Switch语句后面的变量数据类型必须和case后面的数据类型保持一致。

Switch语句可以对变量进行集体判断：



### 自增自减

var i=123;

var n1=i++; 该段代码的含义是： 先将i的值赋值给变量n1,然后变量i的值再加1。

var n2=++i; 该段代码的含义是： 先将i的值加1,然后变量i加1后的值赋值给变量n2。

### break 与continue

在循环体内，只要代码遇到break,程序立马结束当前循环。

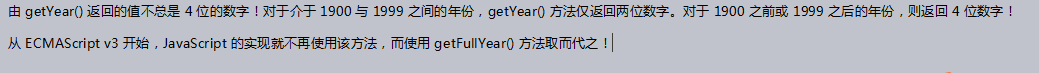
（当前循环指的是break语句所在的循环体。）

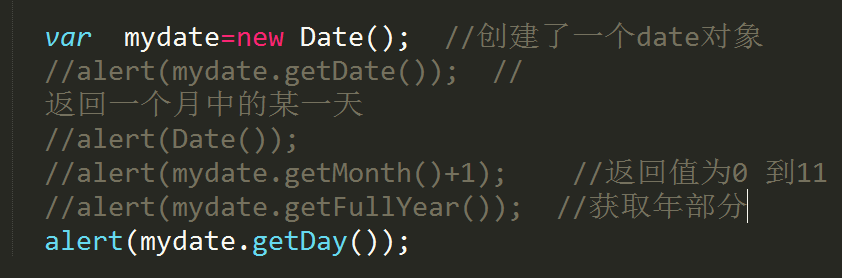
Continue语句指的是跳出本次循环，该语句后面的代码不再执行。

## 相关函数

### Date用法







### Math对象

☞ Math.round(x)

对数字进行四舍五入运算，最后的结果为整数

☞ Math.ceil() 天花板函数

返回一个数字的整数部分

对该数字进行向上舍入

（注意：该方法不会对数字进行四舍五入运算。）

☞ Math.floor() 地板函数

对该数字进行向下舍入

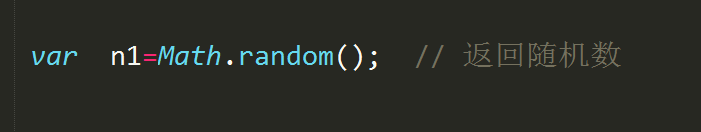
☞ Math.max(x,y)

返回x，y之间的最大值

☞ Math.min(x,y)

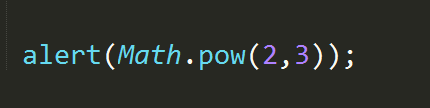
返回x，y之间的最小值

☞ Math.random()



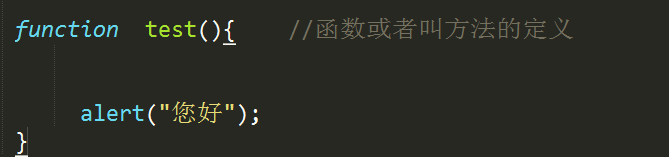
☞ Math.pow(x,y)

返回x值的y次方



### 自定义函数

通过 function关键字 和 自定义方法名 既可定义一个函数。



## JS创建对象

三种方式：

1. var obj = {};

2. var obj1 = new Object();

3. var obj2 = Object.create();

1跟2的区别：

推荐使用第一个方式

第二种方式存在效率问题，因为要new对象，会涉及到原型查找的问题。