# 12章、javascript

## 1、什么是JavaScript

JavaScript简称"JS", 是-种运行于JS解释器或执行引擎中的解释性脚本语言

## 2、JS发展史

1、1992年Nombas公司开发了一款脚本语言，能够运行在网页中，名称CM(C--),更名为ScriptEase

2、1995年Netscape(网景)为自己的浏览器

Navigator2.0开发了另一款脚本语言-LiveScript,后更名为JavaScript

3、1996年Microsoft， 为自己的浏览器IE3.0开发了-款JavaScript的克隆版本JScript

4、1997年Netscape找到了ECMA(欧洲计算机制造商联合会)。Netscape将JS的核心交给了ECMA，从此JS的核心被更名为ECMA Script,简称ES

## 3、js组成

完整的Js组成:

### 1、核心- ECMAScript

包含了Js中最基本的语法规范

### 2、浏览器对象模型-BOM模型

Browser object Model

提供了一系列的可操作方法和属性用于和浏览器进行交互

### 3、文档对象模型-DOM模型

Document object Model

提供了一系列的可操作方法和属性用于和文档进行交互

## 4、js的使用方式

### 1、在元素事件中编写Js代码

事件:用户在元素上所激发的行为操作

onclick:当用户单击元素时激发的操作

语法:

<标记onclick="JS执行代码">

JS可执行代码:

console.log();

JS可执行代码:

console.log("向控制台中输出一句话");

console.log('向控制台中输出一句话');

### 2、将代码嵌入在网页的<script></script>里

语法:

在网页的任意位置处，均可以嵌入

<script>

可执行的s代码

</script>

特点:

网页加载时就执行

document.write("向网页中输出的一句话!");

注意:将document.write编写在按钮中的话，在执行时，会刷新网页中的内容

练习:

在一个按钮中,单击时,执行document.write()

### 3、将JS代码编写在外部的JS文件中

1、创建一个js文件(xxx.js)，编写Js代码

2、在使用的网页中对js文件进行引入

<script src="JS文件路径"></script>

注意:在引入的JS标记中是不能编写JS脚本的

练习:

1. 使用元素事件方式，向控制台中输出一句话

"我的第一个JS练习"

1. 使用<script></script>，向网页中输出一句话

"我的第一个JS练习"

1. 使用外部JS文件的方式，在网页中弹出一句话

"我的第一个JS练习”

## 5、JS的基础语法

### 1、JS都是由语句来组成的

1、由运算符，关键字和表达式构成的

2、Js中严格区分大小写

console.log("...");正确

Console.log( "...");错误

3、每条语句必须以;表示结束

4、所有的标识符，符号必须都是英文的console.log();

### 2、JS中的注释

单行://

//这是一段有注释的内容，其中文字包含比较多，自动换行了

多行:/\* \*/

## 6、JS中的变量和常量

### 1、变量

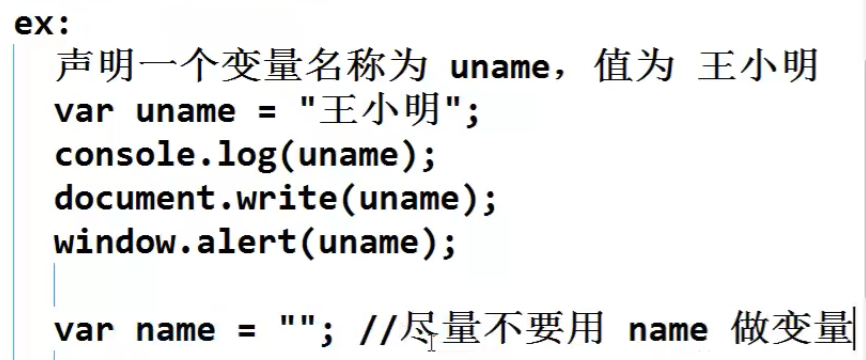
#### 1、声明变量

声明: var 变量名;

赋值:变量名=值;

声明并赋值:

var 变量名=值;

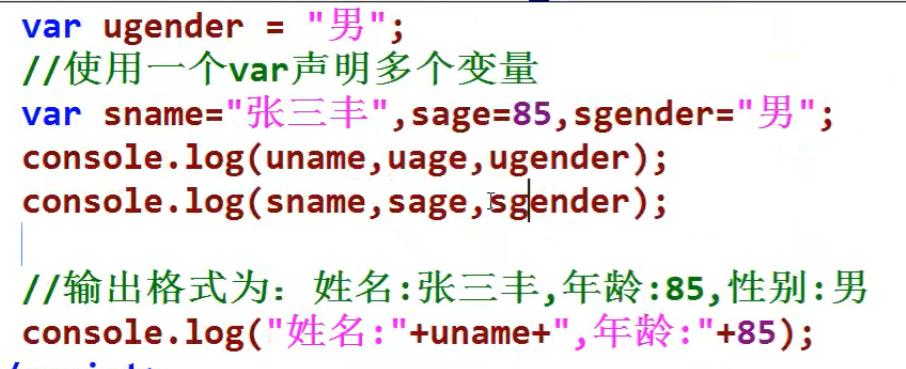


#### 2、注意

1、在声明变量时，尽量使用var 关键字，如果省略var的话，也可以，但容易出问题。

2、声明变量如果未赋值的话，默认值为

undefined



#### 3、变量名的命名规范

1、不能是JS中的关键字和保留关键字

2、由字母，数字，下划线(\_ )和$组成

3、数字不能开头

4、尽量不要重复

5、尽量要见名知意

6、如无特殊需求，尽量使用小驼峰命名法

#### 4、变量的使用方式

##### 1、为变量赋值

只要变量出现在赋值(=)符号的左边,一律都是赋值操作

var age = 35;

var newAge = age;

##### 2、获取变量的值

只要变量没出现在赋值(=)符号的左边，一律都是取值操作

console.1og(uname) ;

var age = 35;//赋值

age = age + 35;

赋值操作:

1、先运算赋值符号右面表达式或值,再赋值给左边的变量

### 2、常量

#### 1、什么是常量

经声明就不允许被修改的数据就是常量

#### 2、语法

const常量名=值;

注意:

1、常量声明好之后是不允许被修改的，所以一定要赋值

2、常量名一般采用全大写形式

### 3、数据类型

JS中的数据类型分为以下两大类:

#### 1、基本数据类型

##### 1、number类型

数字类型，可以表示32位(4字节)的整数以及64位(8字节)的浮点数

整数:

表示十进制，八进制，十六进制

十进制:var num=10;

八进制: var num = 0o10;

十六进制: var num = 0x10;

小数:

小数点计数法: var num = 123.456;

指数计数法: var num = 1.8e2;

##### 2、string类型

字符串类型

由Unicode字符，数字，标点组成

注意:字符串在使用时必须用""或'引起来

###### 1、查看字符的unicode码

var str ="张";

var ucode=str.charCodeAt().toString(16);

###### 2、如何将Unicode码转换成对应的字符?

已知Unicode码: 5f20

var str = "\u5f20";

console.log(str);

###### 3、中文范围

"\u4e00”~ "\u9fa5"

###### 4、转义字符

\n :换行

\t : 一个制表符

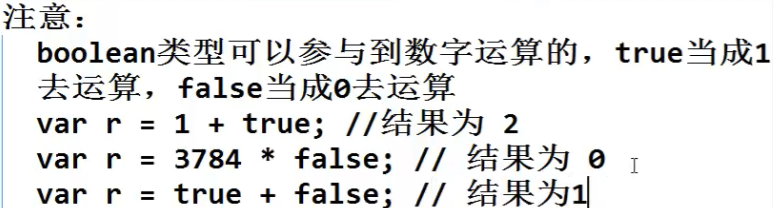
\":"

\’:'

\\:\\

##### 3、boolean类型

布尔类型，只用于表示真(true)或假(false)通常会表示条件的结果



#### 2、查看数据类型

使用typeof() 或typeof查看变量的数据类型

#### 3、数据类型转换

##### 1、隐式转换

大部分加法运算时，如果数据类型不一致的话可以

进行隐式转换

1、数字+字符串:将数字转换为字符串

var num = 15; //number

var str = "18"; //string

varr=num+str;//1518

2、数字＋布尔:将布尔类型转换为number

var r = 35 + true ; //36

1. 字符串+布尔:将布尔转换为字符串

var result ="你好"+ true;

结果为:你好true

4、布尔+布尔:将布尔转换成数字

var r = true + true;

结果:2

注意:

任何数据类型与字符串做"+"运算时，都是先转换成字符串再做拼接

##### 强制转换（显示转换/转换函数）

###### 1、tostring()

将任意类型的数据转换成字符串,并返回转换后的结果

var num = 15;

var r =num.toString();

var num = 15;

var r = num + "";

###### 2、p**a**rseInt()

作用:将任意类型的数据尽量转换成整数，如果实在无法转换的话,则返回NaN(Not a Number)

1、

var r = parseInt( "13");

r : 13

2、

var r = parseInt( "13.5");

r : 13

3、

var r = parseInt( "13你好");

r : 13

4、

var r = parseInt( "你好13");

r : NaN

###### 3、parseFloat()

作用:尽量将任意类型的数据转换为小数

1、

var num = parseF1oat( "35.07");

num : 35.07

2、

var num = parseFloat( "35.7你好");

num : 35.7

3、

var num =.parseFloat( "你好35.7");

num : NaN

###### 4、Number()

作用:将指定的数据转换成数字，只要包含非法字符的话,结果就为NaN

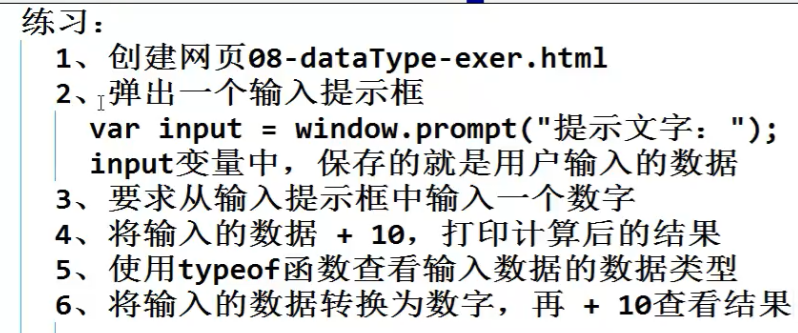
1、

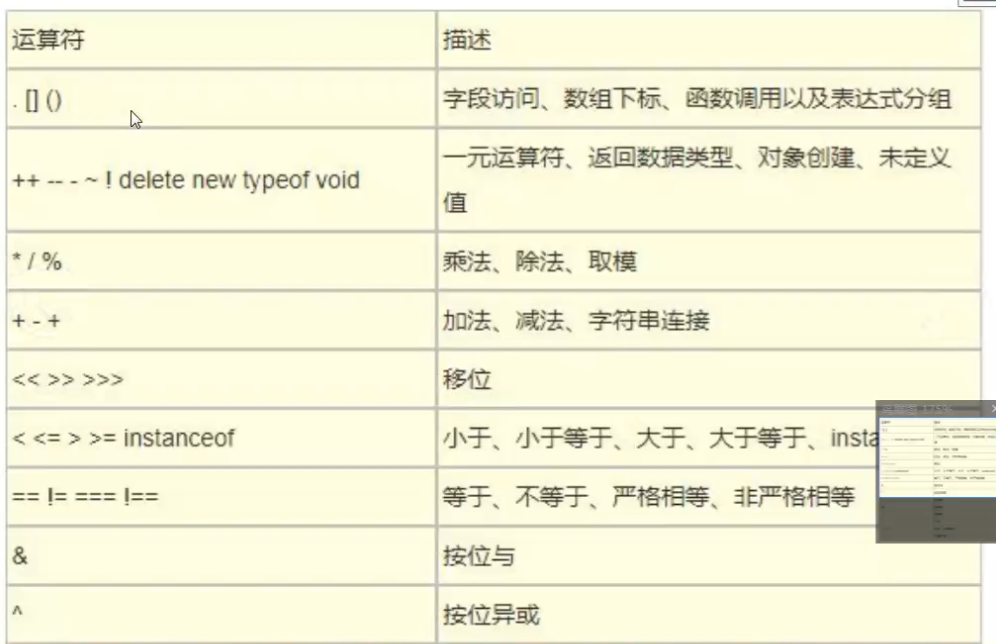
var r = Number( "35.5");

r : 35.5

var r =Number( "35你好");

r : NaN





## 7、流程结构

### 1、分支结构.

#### 1、作用

根据条件，选择某一段代码去执行

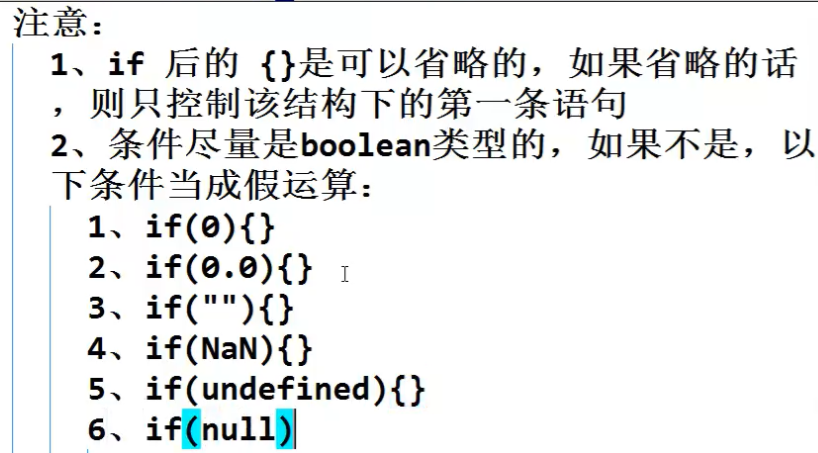
#### 2、if结构

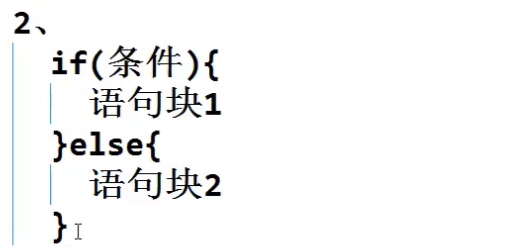
1、

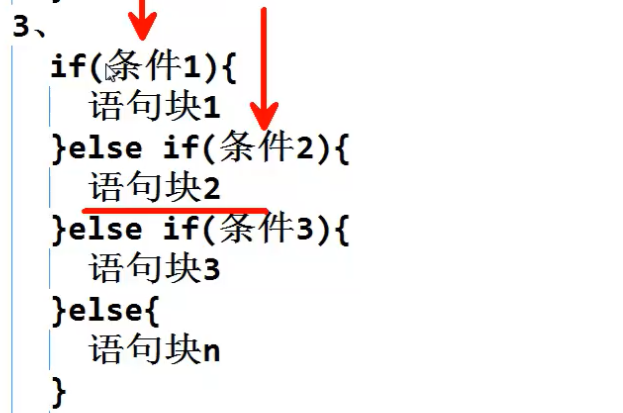
if(条件){

满足条件要 执行的代码块

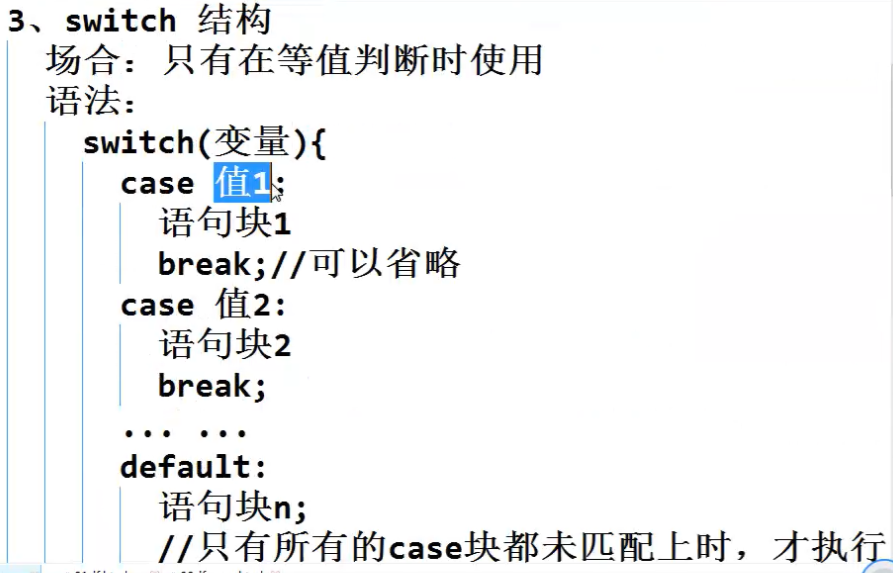
}

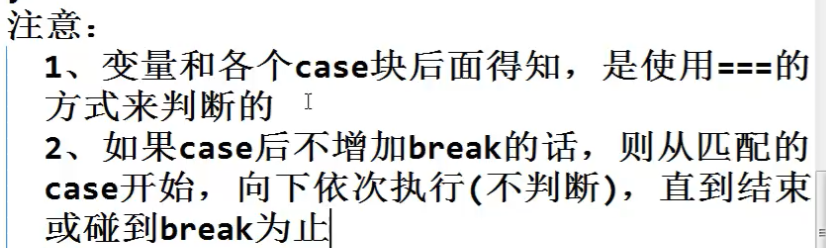


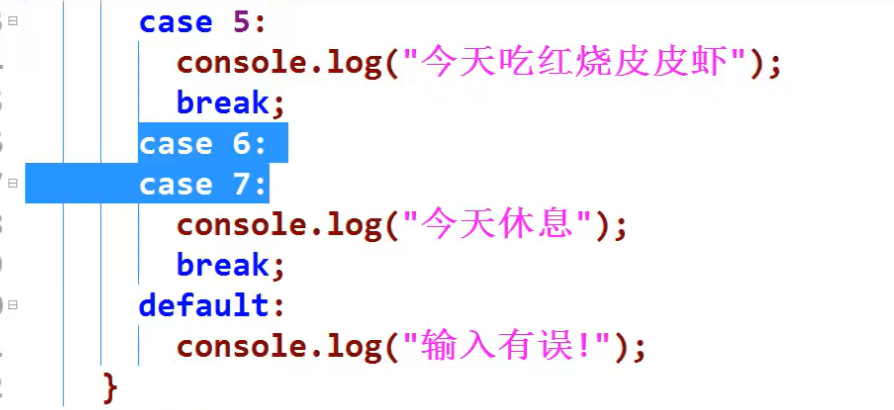




#### 3、switch结构







### 2、循环结构

#### 1、循环的作用

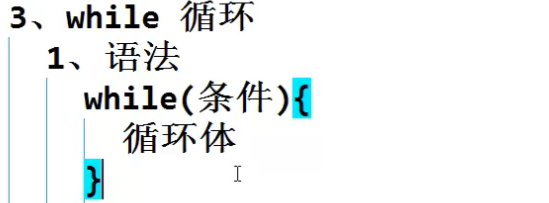
重复的执行相同或相似的代码

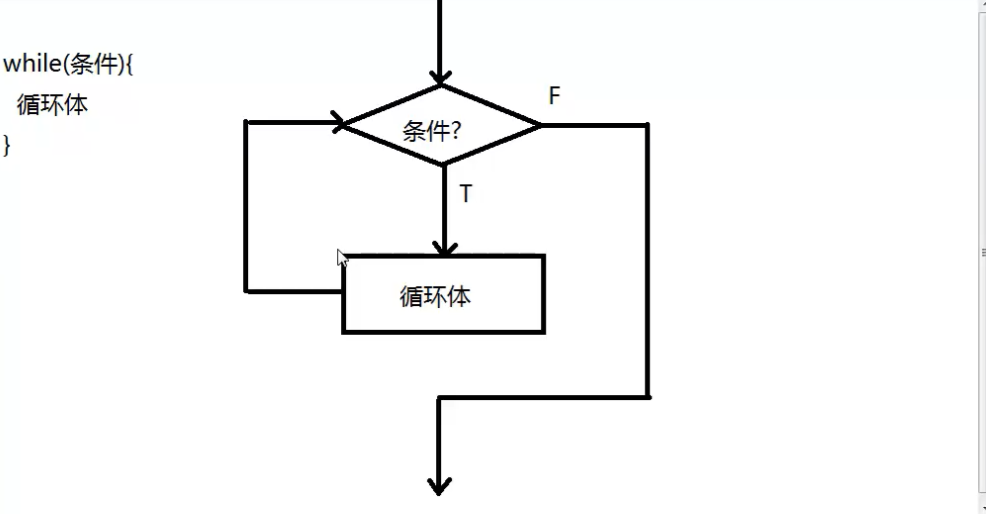
#### 2、循环二要素

1、循环条件

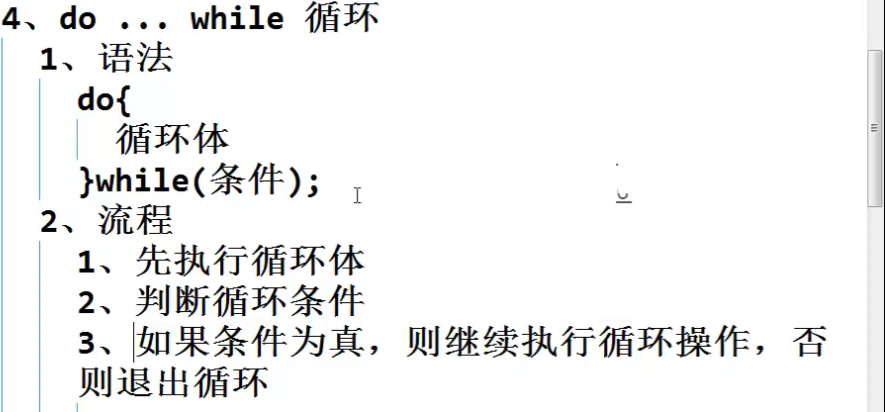
2、循环操作(循环体)

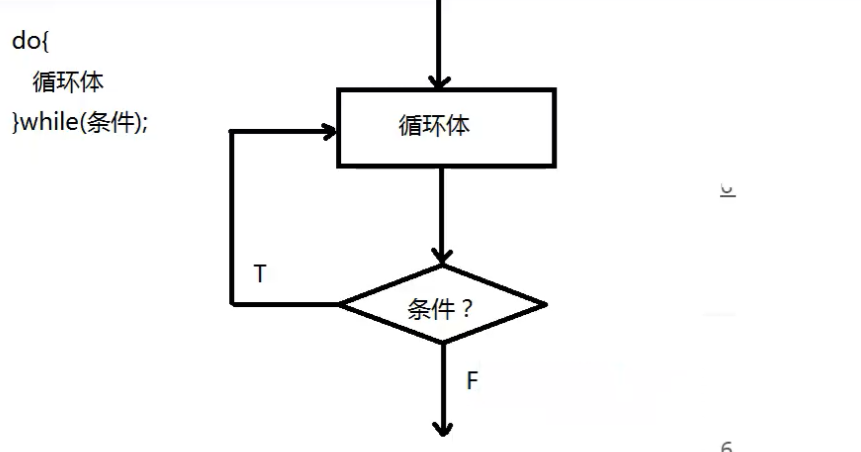
#### 3、while循环

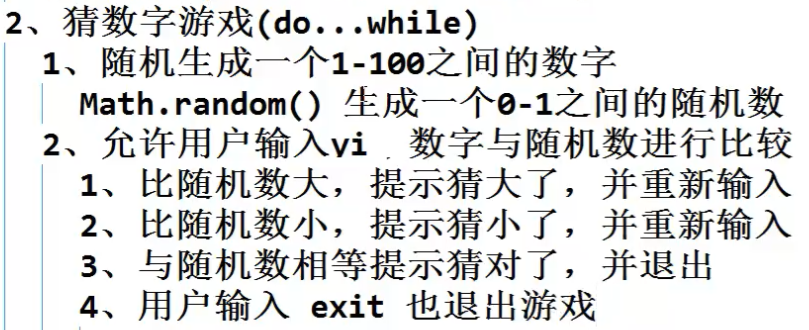




#### 4、do…while循环







#### 5、循环的流程控制

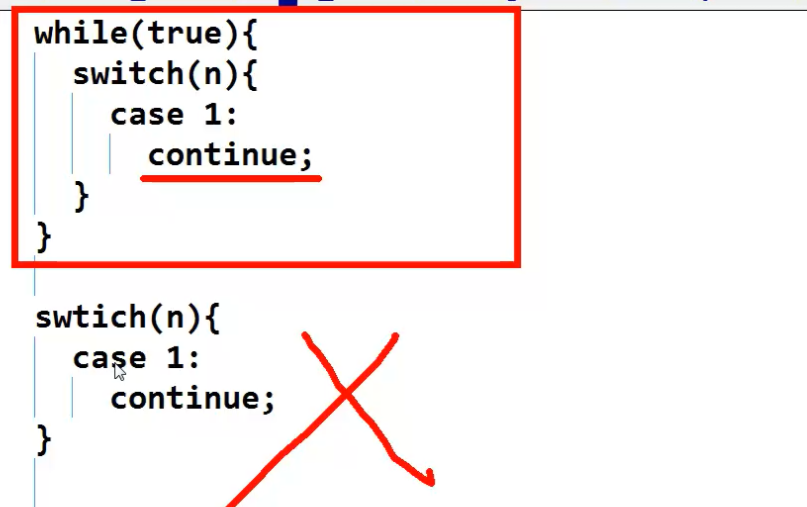
1、break

用在循环体内，用于跳出整个循环结构

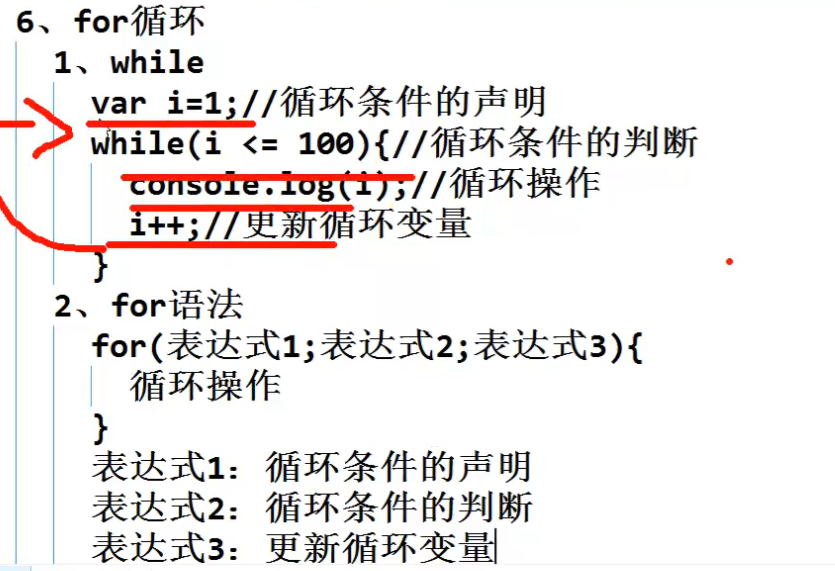
2、continue

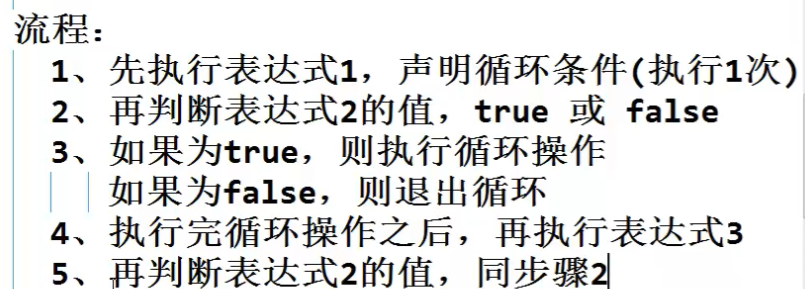
用在循环体内，用于跳出本次循环，继续执行

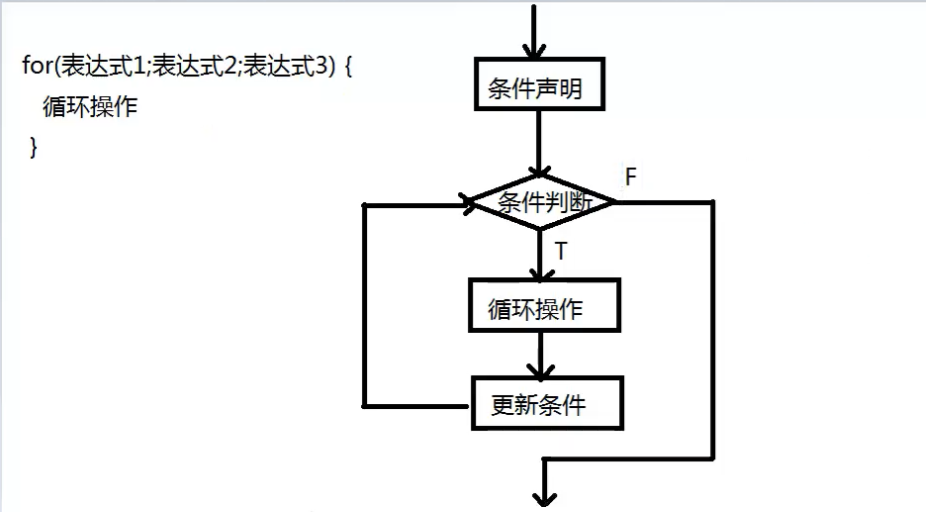
下次循环

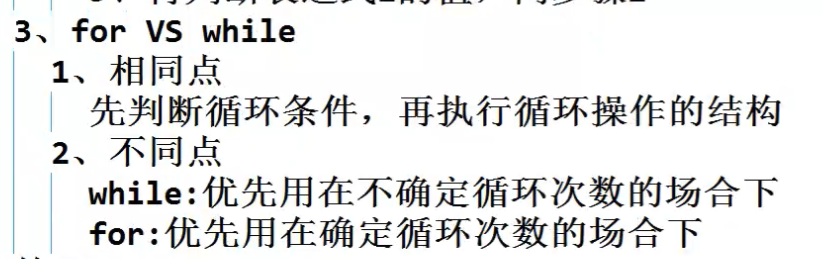


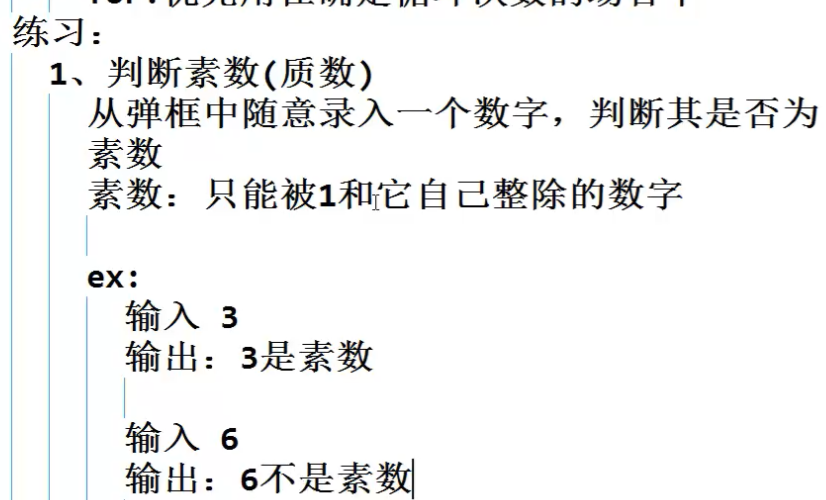
#### 6、for循环

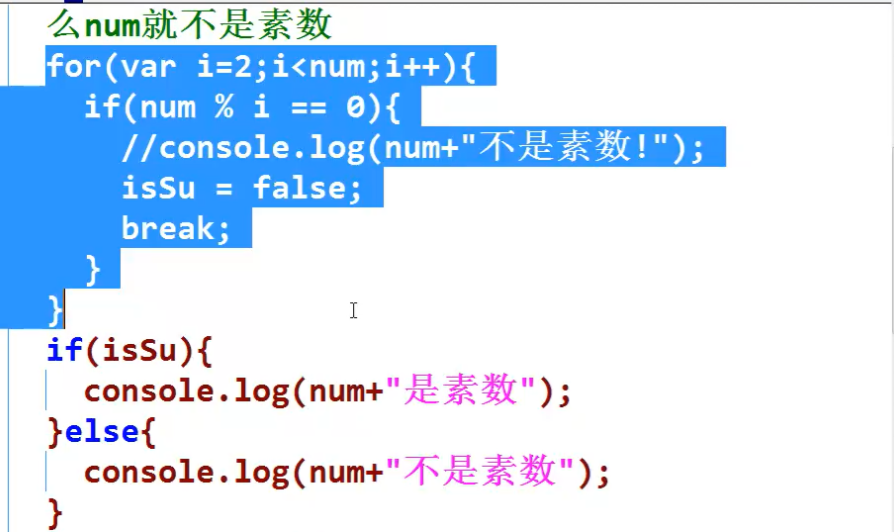












### 3、函数

#### 1、什么是函数

函数，即function, 是一段预定义好，可以独立

执行并包含多条执行语句的代码块

优点:

1、多条执行语句的封装

2、实现了代码的复用

#### 2、在JS中创建函数

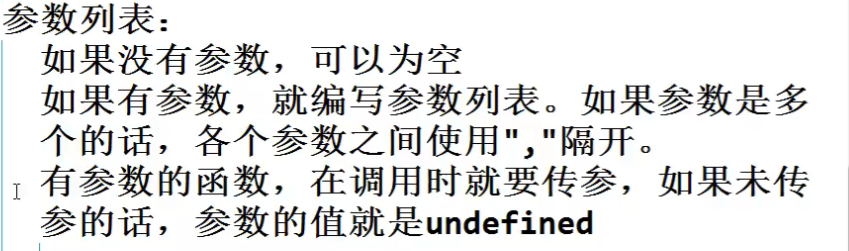
function函数名(参数列表){

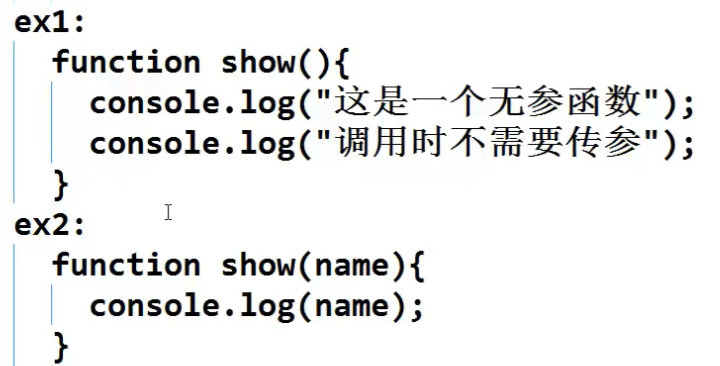
函数体

[返回值]

}

##### 参数列表





##### 返回值

在函数体内，经过运算后,需要传递给函数调用者的一个值,就是返回值。

返回值是可选的,有返回值的话,需要使用return进行返回,最多只能返回一个值

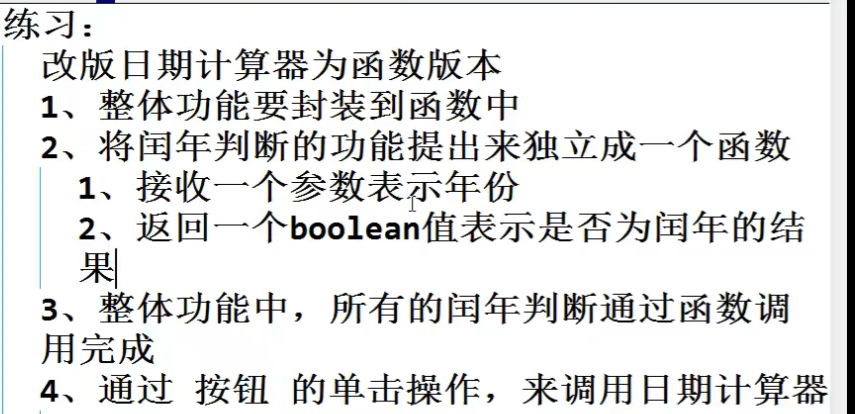
#### 3、函数的调用

在任意的JS合法的位置处，都允许做函数的调用

var ret =函数名(参数列表);

有参数，则传参，否则，为空

有返回值，可以接收,否则不接收



#### 4、变量的作用域

##### 1、什么是变量的作用域

|变量的作用域指的是变量的可访问范围

##### 2、作用域的分类

###### 1、局部变量

使用var关键字声明在某个函数内的变量，都是局部变量

局部变量只能在声明的函数内使用,出了函数就无法使用

###### 2、全局变量

1、在function之外声明的变量都是全局变量

2、声明变量不使用var关键字的时候，都是全局变量

全局变量可以应用在各个函数中以及各个位置处

注意:全局变量，推荐放在所有的function之外，使用var关键字去声明