**学习目标**

1、掌握函数的作用

2、掌握函数的定义

3、掌握函数的调用

4、掌握函数的返回值

5、掌握arguments

**函数的作用**

作用：可以封装任意多条语句，而且可以在任何地方、任何时候调用执行。

发散：我们可以把反复执行的脚本，封装在一个脚本里面，这样我们可以在我们想要调用的时候调用它。

**函数的定义**

函数使用function声明，后跟一组参数一级函数体，语法如下：

function functionName([arg0,arg1,…argn]){

statements

}

说明：

1、functionName是要定义的函数名，属于标识符 （标识符规则）

2、 []中的arg0,arg1,…argn为函数的参数，不是必需的

3、[]只说明里面的内容不是必须的，它不是语法

4、函数定义好了，必须调用才能执行，否则是不会执行的。

**函数的调用**

语法：

函数名([arg1,arg2,…argn])

例一：

// 声明一个没有参数的函数

function myFun(){

alert("我是一个函数");

}

myFun();// 函数的调用

例二

// 声明一个带有参数的函数，做相加

function add(num1,num2){

var sum=num1+num2;

alert(num1+'和'+ num2 +'的和是'+sum);

}

add(3,5);

add(99,798);

add(-1,-9);

**函数的返回值**

任何函数在任何时候都可以通过return语句后跟要返回的值来实现返回值。

说明：

1、函数会在执行完return语句之后停止并立即退出。

2、return语句也可以不带有任何返回值，一般用于需要提前停止函数执行而又不需要返回值的情况下。

3、函数的返回值即函数调用的结果。可以把函数调用赋给某个值，这个值即函数调用的结果。

例：

function add(num1, num2) {

var sum = num1 + num2;

return sum;

}

console.log(add(3, 5));

alert(add(99, 798));

var s = add(-1, -9);

document.write(s);

例

function myFunction(arg){

// 如果arg是非数字，不做任何操作，否则返回arg的2倍

if(isNaN(arg)) return;

return arg\*2;

}

console.log(myFunction('abc'));

console.log(myFunction(40));

**arguments**

函数体内的参数可以通过arguments对象来访问这个类数组参数。

说明：

1、arguments对象只是与数组类似，但它并不是Array的实例。

2、可以使用方括号语法访问它的每一个元素。

3、使用length属性来确定传递进来的参数个数。

特别说明：第9单元好像还差一个讲arguments的视频。