## **函数显式参数与隐藏参数(arguments)**

在先前的教程中，我们已经学习了函数的显式参数:

*functionName*(*parameter1, parameter2, parameter3*) {  
    *code to be executed*  
}

## **默认参数**

如果函数在调用时缺少参数，参数会默认设置为： ****undefined****

有时这是可以接受的，但是建议最好为参数设置一个默认值：

## **实例**

function myFunction(x, y) {  
    if (y === undefined) {  
          y = 0;  
    }   
}

## **Arguments 对象**

JavaScript 函数有个内置的对象 arguments 对象.

argument 对象包含了函数调用的参数数组。

通过这种方式你可以很方便的找到最后一个参数的值：

## **实例**

x = findMax(1, 123, 500, 115, 44, 88);

function findMax() {

    var i, max = arguments[0];

    if(arguments.length < 2){

     return max;

    }

    for (i = 1; i < arguments.length; i++) {

        if (arguments[i] > max) {

            max = arguments[i];

        }

    }

    return max;

}

## **通过值传递参数**

在函数中调用的参数是函数的参数。

如果函数修改参数的值，将不会修改参数的初始值（在函数外定义）。

**实例**

**var** x = 1;// 通过值传递参数**function** **myFunction**(x) {

x++; //修改参数x的值，将不会修改在函数外定义的变量 x

console.log(x);

}

myFunction(x); // 2console.log(x); // 1

## **通过对象传递参数**

在JavaScript中，可以引用对象的值。

因此我们在函数内部修改对象的属性就会修改其初始的值。

修改对象属性可作用于函数外部（全局变量）。

**实例**

**var** obj = {x:1};// 通过对象传递参数

**function** **myFunction**(obj) {

obj.x++; //修改参数对象obj.x的值，函数外定义的obj也将会被修改

console.log(obj.x);

}

myFunction(obj); // 2console.log(obj.x); // 2