## **JavaScript 函数定义**

JavaScript 使用关键字 ****function**** 定义函数。

## **函数声明**

在之前的教程中，你已经了解了函数声明的语法 :

function *functionName*(*parameters*) {  
  *执行的代码*  
}

函数声明后不会立即执行，会在我们需要的时候调用到。

## **函数表达式**

JavaScript 函数可以通过一个表达式定义。

函数表达式可以存储在变量中：

## **实例**

var x = function (a, b) {return a \* b};

在函数表达式存储在变量后，变量也可作为一个函数使用：

## **实例**

var x = function (a, b) {return a \* b};  
var z = x(4, 3);

## **Function() 构造函数**

在以上实例中，我们了解到函数通过关键字 ****function**** 定义。

函数同样可以通过内置的 JavaScript 函数构造器（Function()）定义。

## **实例**

var myFunction = new Function("a", "b", "return a \* b");  
  
var x = myFunction(4, 3);

实际上，你不必使用构造函数。上面实例可以写成：

## **实例**

var myFunction = function (a, b) {return a \* b}  
  
var x = myFunction(4, 3);

## **自调用函数**

函数表达式可以 "自调用"。

自调用表达式会自动调用。

如果表达式后面紧跟 () ，则会自动调用。

Y不能自调用声明的函数。

通过添加括号，来说明它是一个函数表达式：

## **实例**

(function () {  
    var x = "Hello!!";      // 我将调用自己  
})();

以上函数实际上是一个 ****匿名自我调用的函数**** (没有函数名)。

## **函数是对象**

在 JavaScript 中使用 ****typeof**** 操作符判断函数类型将返回 "function" 。

但，JavaScript 函数描述为一个对象更加准确。

JavaScript 函数有 ****属性**** 和 ****方法****。

arguments.length 属性返回函数调用过程接收到的参数个数：

## **实例**

function myFunction(a, b) {  
    return arguments.length;  
}