## **JavaScript 函数调用**

JavaScript 函数有 4 种调用方式。

每种方式的不同方式在于 ****this**** 的初始化。

## *****this*** 关键字**

一般而言，在Javascript中，this指向函数执行时的当前对象。

## **作为一个函数调用**

## **实例**

function myFunction(a, b) {  
    return a \* b;  
}  
myFunction(10, 2);           // myFunction(10, 2) 返回 20

在浏览器中的页面对象是浏览器窗口(window 对象)。以上函数会自动变为 window 对象的函数。

myFunction() 和 window.myFunction() 是一样的：

## **实例**

function myFunction(a, b) {  
    return a \* b;  
}  
window.myFunction(10, 2);    // window.myFunction(10, 2) 返回 20

## **全局对象**

当函数没有被自身的对象调用时， ****this**** 的值就会变成全局对象。

在 web 浏览器中全局对象是浏览器窗口（window 对象）。

该实例返回 ****this**** 的值是 window 对象:

## **实例**

function myFunction() {  
    return this;  
}  
myFunction();                // 返回 window 对象

## **函数作为方法调用**

在 JavaScript 中你可以将函数定义为对象的方法。

以下实例创建了一个对象 (****myObject****), 对象有两个属性 (****firstName**** 和 ****lastName****), 及一个方法 (****fullName****):

## **实例**

var myObject = {  
    firstName:"John",  
    lastName: "Doe",  
    fullName: function () {  
        return this.firstName + " " + this.lastName;  
    }  
}  
myObject.fullName();         // 返回 "John Doe"

## **使用构造函数调用函数**

如果函数调用前使用了 ****new**** 关键字, 则是调用了构造函数。

这看起来就像创建了新的函数，但实际上 JavaScript 函数是重新创建的对象：

// 构造函数:  
function myFunction(arg1, arg2) {  
    this.firstName = arg1;  
    this.lastName  = arg2;  
}  
  
// This creates a new object  
var x = new myFunction("John","Doe");  
x.firstName;                             // 返回 "John"