对象是包含一组键值对的实例。 值可以是标量、函数、数组、对象等，如下实例：

var object\_name = {

    key1: "value1", // 标量

    key2: "value",

    key3: function () {

        // 函数

    },

    key4: ["content1", "content2"] //集合

}

**鸭子类型(Duck Typing)**

鸭子类型（英语：duck typing）是动态类型的一种风格，是多态(polymorphism)的一种形式。

*可以这样表述：*

*"当看到一只鸟走起来像鸭子、游泳起来像鸭子、叫起来也像鸭子，那么这只鸟就可以被称为鸭子。"*

在鸭子类型中，关注点在于对象的行为，能作什么；而不是关注对象所属的类型。例如，在不使用鸭子类型的语言中，我们可以编写一个函数，它接受一个类型为"鸭子"的对象，并调用它的"走"和"叫"方法。在使用鸭子类型的语言中，这样的一个函数可以接受一个任意类型的对象，并调用它的"走"和"叫"方法。如果这些需要被调用的方法不存在，那么将引发一个运行时错误。任何拥有这样的正确的"走"和"叫"方法的对象都可被函数接受的这种行为引出了以上表述，这种决定类型的方式因此得名。

interface IPoint {

    x: number

    y: number

}

function addPoints(p1: IPoint, p2: IPoint): IPoint {

    var x = p1.x + p2.x

    var y = p1.y + p2.y

    return { x: x, y: y }

}

// 正确

var newPoint = addPoints({ x: 3, y: 4 }, { x: 5, y: 1 })

// 错误

var newPoint2 = addPoints({ x: 1 }, { x: 4, y: 3 })