Java应用：编写高级JavaScript代码Java认证考试 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/644/2021_2022_Java_E5_BA_94_E7_94_A8_c104_644513.htm 1、创建高级对象 使用构造函数来创建对象 构造函数是一个函数，调用它来例示并初始化特殊类型的对象。可以使用 new 关键字来调用一个构造函数。下面给出了使用构造函数的新示例。 var myObject = new Object(). // 创建没有属性的通用对象。 var myBirthday = new Date(1961, 5, 10). // 创建一个 Date 对象。 var myCar = new Car(). // 创建一个用户定义的对象，并初始化其属性。 通过构造函数将一个参数作为特定的 this 关键字的值传递给新创建的空对象。然后构造函数负责为新对象执行适应的初始化（创建属性并给出其初始值）。完成后，构造函数返回它所构造的对象的一个参数。 编写构造函数 可以使用 new 运算符结合像 Object()、Date() 和 Function() 这样的预定义的构造函数来创建对象并对其初始化。面向对象的编程其强有力的特征是定义自定义构造函数以创建脚本中使用的自定义对象的能力。创建了自定义的构造函数，这样就可以创建具有已定义属性的对象。下面是自定义函数的示例（注意 this 关键字的使用）。 function Circle (xPoint, yPoint, radius) { this.x = xPoint. // 圆心的 x 坐标。 this.y = yPoint. // 圆心的 y 坐标。 this.r = radius. // 圆的半径。 } 调用 Circle 构造函数时，给出圆心点的值和圆的半径（所有这些元素是完全定义一个独特的圆对象所必需的）。结束时 Circle 对象包含三个属性。下面是如何例示 Circle 对象。 var aCirle = new Circle(5, 11, 99). 使用原型来创建对象 在编写构造函数时，可以使用原型对象（它

本身是所有构造函数的一个属性）的属性来创建继承属性和共享方法。原型属性和方法将按引用复制给类中的每个对象，因此它们都具有相同的值。可以在一个对象中 更改原型属性的值，新的值将覆盖默认值，但仅在该实例中有效。属于这个类的其他对象不受此更改的影响。下面给出了使用自定义构造函数的示例，Circle（注意 this 关键字的使用）。 Circle.prototype.pi = Math.PI. function ACirclesArea () { return this.pi * this.r * this.r. // 计算圆面积的公式为 ?r2。 } Circle.prototype.area = ACirclesArea. // 计算圆面积的函数现在是 Circle Prototype 对象的一个方法。 var a = ACircle.area(). // 此为如何在 Circle 对象上调用面积函数。 使用这个原则，可以给预定义的构造函数（都具有原型对象）定义附加属性。例如，如果想要能够删除字符串的前后空格（与 VBScript 的 Trim 函数类似），就可以给 String 原型对象创建自己的方法。 // 增加一个名为 trim 的函数作为 // String 构造函数的原型对象的一个方法。 String.prototype.trim = function() { // 用正则表达式将前后空格 // 用空字符串替代。 return this.replace(/(^\s*) (\s*$)/g, ""). } // 有空格的字符串 var s = " leading and trailing spaces ". // 显示 " leading and trailing spaces (35)" window.alert(s " (" s.length ")"). // 删除前后空格 s = s.trim(). // 显示"leading and trailing spaces (27)" window.alert(s " (" s.length ")"). 2.递归 递归是一种重要的编程技术。该方法用于让一个函数从其内部调用其自身。一个示例就是计算阶乘。0 的阶乘被特别地定义为 1。 更大数的阶乘是通过计算 1 * 2 * ...来求得的，每次增加 1，直至达到要计算其阶乘的那个数。 下面的段落是用文字定义的计算阶乘的一个函数。 “ 如果这个数

小于零，则拒绝接收。如果不是一个整数，则将其向下舍入为相邻的整数。如果这个数为 0，则其阶乘为 1。如果这个数大于 0，则将其与相邻较小的数的阶乘相乘。 ”要计算任何大于 0 的数的阶乘，至少需要计算一个其他数的阶乘。用来实现这个功能的函数就是已经位于其中的函数；该函数在执行当前的这个数之前，必须调用它本身来计算相邻的较小数的阶乘。这就是一个递归示例。 递归和迭代（循环）是密切相关的？能用递归处理的算法也都可以采用迭代，反之亦然。确定的算法通常可以用几种方法实现，您只需选择最自然贴切的方法，或者您觉得用起来最轻松的一种即可。 显然，这样有可能会出现问题。可以很容易地创建一个递归函数，但该函数不能得到一个确定的结果，并且不能达到一个终点。这样的递归将导致计算机执行一个“ 无限 ”循环。下面就是一个示例：在计算阶乘的文字描述中遗漏了第一条规则（对负数的处理），并试图计算任何负数的阶乘。这将导致失败，因为按顺序计算 -24 的阶乘时，首先不得不计算 -25 的阶乘；然而这样又不得不计算 -26 的阶乘；如此继续。很明显，这样永远也不会到达一个终止点。 因此在设计递归函数时应特别仔细。如果怀疑其中存在着无限递归的可能，则可以让该函数记录它调用自身的次数。如果该函数调用自身的次数太多，即使您已决定了它应调用多少次，就自动退出。 下面仍然是阶乘函数，这次是用 JScript 代码编写的。 // 计算阶乘的函数。如果传递了 // 无效的数值（例如小于零）， // 将返回 -1，表明发生了错误。若数值有效， // 把数值转换为最相近的整数，并 // 返回阶乘。 function factorial(aNumber) { aNumber = Math.floor(aNumber). // 如果这个数不是一个整数

，则向下舍入。 if (aNumber 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com