文章编号：2096-1472(2017)-04-09-03

Java语言中对象的理解与应用

邢如意
（江苏联合职业技术学院徐州财经分院，江苏 徐州 221008）

摘 要：面向对象是当前主流的程序设计方法，是软件开发过程中重要的理论支撑。正确理解对象与类的概念，以及正确的使用对象对于学习面向对象编程具有重要作用。文中以面向对象程序设计中“对象”作为主体贯穿全文，分别从对象的理解、Java语言中对象管理、对象如何正确应用三个方面进行阐述。文章内容精炼、源于实战，对于帮助理解和掌握Java面向对象程序设计具有较好的参考价值。

关键词：Java；对象；面向对象；OOP
中图分类号：TP312 **文献标识码**：A

Understanding and Application of the Object in Java Programming Language

XING Ruyi
(*Xuzhou Branch of Finance & Economics,Jiangsu Union Technical Institute,Xuzhou* 221008,*China*)

Abstract: As the current mainstream in programming,the object-oriented method is important theoretical support in the process of software development.It is important to thoroughly understand the concepts of objects and classes,and properly apply objects in programming.Focusing on the object in the object-oriented programming,the paper elaborates on the understanding of the object,the object management in the Java programming language,and how to apply the object correctly. With the efficient language and practical experience,the paper is of good reference to the understanding and application of Java object-oriented programming.

Keywords:java;object;oject oriented;Object-Oriented Programming

1 引言(Introduction)

面向对象是当前主流编程语言的共同的特征，如Java、C＃语言。面向对象涉及到软件开发的各个阶段，具体包括面向对象分析OOA、面向对象设计OOD、面向对象编程OOP，形成了完整的面向对象的软件工程理论、方法和工具[1]。

在学习面向对象编程语言时，首先学习的就是对象的概念，能够正确的理解对象和使用对象对于之后的面向对象特性学习具有很大帮助。本文从对象与类的理解、对象的原理和对象的使用三个方面循序渐进阐述，较全面的介绍了关于对象概念、对象存储原理及在编程中正确使用对象的技巧等内容，内容对于理解面向对象思想和学习面向对象编程具有较大帮助。

2 理解对象与类(Understanding objects and classes)

掌握面向对象程序设计首先要正确理解面向对象思想，面向对象思想的核心是正确理解“万物皆对象”这句话。在编程时运用“万物皆对象”的思想是指：通过使用面向对象的方式，将现实世界中的实体看作为对象，对这些实体的特性进行描述，并分析出实体具有的功能或职责。描述实体特性的过程即为提取对象属性的过程，分析实体的功能或职责的过程即为提取对象方法的过程，完整描述一个对象即从属性和方法两个层面，提取对象属性和方法的过程即为使用面向对象思想进行分析和抽象的过程。

在把客观实体抽象为对象后，还要继续分析各对象之间的关系，最后将对象的抽象结果描述出来形成一段文本，此文本即为面向对象思想中的“类class”。从以上过程可以发现，类是对象的属性、方法、对象之间关系的描述。对象是现实世界中可以描述的实体，是具体的；类是对象的描述，是不具体的，在编程时类体现为一段代码文本。

以下以驾驶员和汽车为例阐述类与对象的关系与区别。现实世界中汽车与驾驶员皆为生活中常见的实体。当我们看到一辆汽车时会自然联想到汽车的品牌、型号等属性信息，分析其职责(功能)可以得出汽车可以行驶、停止。但汽车不会自己行驶或停止，它需要驾驶员执行启动、挂挡等操作，可以看出汽车与驾驶员二者是存在依赖关系的。分析驾驶员对象可以得出驾驶员具有驾驶证照号码、性别、年龄等属性信息，驾驶员具有“驾驶”能力。分析汽车对象可以得出汽车具有品牌、型号、价格等属性，汽车具有行驶和停止的功能。经过上述分析，可以抽象出汽车类和驾驶员类，以及两

个类间的依赖关系，其类图如图1所示。

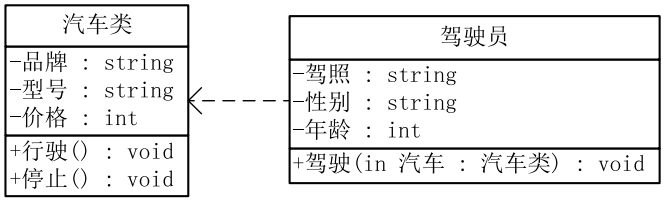

图1 汽车类与驾驶员类图

Fig.1 Class diagram of car class and driver class

经过对汽车与驾驶员两个对象的分析，继而再抽象出两个类可以看出，面向对象思想与人类的思维方式极为相似，对于初学者而言并不难入门，但对于一些较为抽象的场景而言初学者通常会感觉稍有压力。下面以开发一款简单人机猜数字游戏为例继续理解对象与类的概念。人机猜数字游戏的规则为：机器随机生成一个数值，玩家输入猜的数字且每一局最多猜三次，若三次都未猜中则本局失败。游戏根据猜的数字位数分为三个级别，1位数为初级、2位数为中级、3位数为高级。根据游戏规则的描述可快速分析出本案例至少包括人类玩家对象和机器玩家对象。人类玩家对象具有玩家名称、当前游戏级别属性，功能职责为输入猜的数字。机器玩家对象职责为根据当前人类玩家的级别生成相应位数的随机数字。根据上述分析可以看到只有人类玩家和机器玩家对象还无法实现游戏功能，因为目前还没有控制游戏进度的对象和描述游戏级别信息的对象。因此，在进一步分析后抽象出游戏对象和级别对象。游戏对象相当于裁判，可控制游戏的启动及停止、判断所猜数字是否正确和设置当前人类玩家的级别。游戏级别对象则用于保存级别的参数信息，包括级别名称、级别对应数字的位数。综上所述，本案例最后抽取出游戏类、机器玩家类、人类玩家类、游戏级别类。其中游戏类与机器玩家类、人类玩家类具有关联关系，机器玩家类与人类玩家类具有依赖关系，人类玩家类与游戏等级类具有关联性，其类图如图2所示。

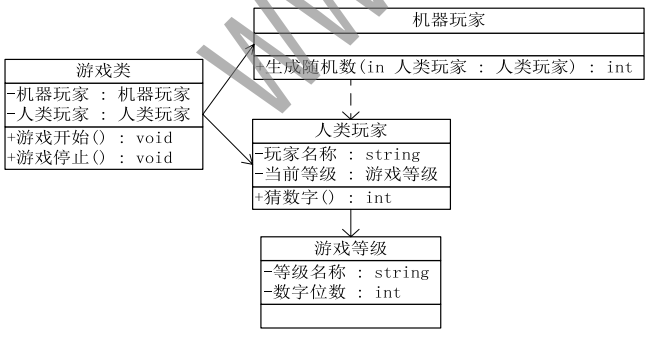

图2 人机猜数字游戏类图

Fig.2 Class diagram of man-machine guessing game

3 Java对象管理(Java object management)

Java语言编写的程序运行在Java虚拟机之上，基于此特点才实现了Java程序的平台无关性和良好的可移植性[2,3]。基于虚拟机平台运行的另一个好处是在编写程序时不再需要考虑C或C++语言中的手动内存管理问题，Java虚拟机会自动进行内存的申请和释放。在Java虚拟机中，存放对象的区域是内存占用比例最高的，这个区域称为堆heap[4,5]。因此能够理解对象在堆中的存储原理对编写高效的Java程序具有很大帮助。

学习Java对象管理的主要内容是学习对象的生命周期，即对象的创建、使用和销毁三个阶段。

3.1 对象创建

创建一个Java对象需要经过三个过程分别为加载类文件、分配内存、对象初始化[5,6]。

(1)加载类文件：类是对对象的描述，在使用new关键字创建对象时，虚拟机会先检查对应的类文件是否已经被加载，若未加载则会先加载类文件到虚拟机中。Java语言提供了多种类加载器classloader用于加载类文件(.class文件)到虚拟机中[6,7]，分别包括：

a.启动加载器：最顶层的类加载器，用于加载核心类库，如java语言提供的rt.jar等；

b.扩展类加载器：用于加载Java目录下ext目录中的所有类库文件；

c.系统类加载器：负责加载应用程序classpath目录下所有类文件，此目录下类文件即为开发者自主编写的程序文件[8]。

(2)分配内存：当类文件加载完后，对象在创建时需使用的初始内存大小就确定了。虚拟机在堆空间中划分出相应大小的存储空间。当前虚拟机在内存分配方法上采用指针碰撞法或空闲列表法。“指针碰撞”法假设Java堆中的内存是绝对规整的，所有用过的内存都放在一边，空闲的内存放在另一边，中间放一个指针作为分界点。当需要分配内存时只需要把指针向空闲内存方向移动对象大小相等的距离即可。如果Java堆中的内存并不规整，那么虚拟机需要维护一个列表用来记录哪些内存块可用。当需要分配内存时从列表中找出一个足够大的空间划分给对象实例，这就是“空闲列表”。

(3)对象初始化：虚拟机在对象内存分配完成后会给对象内部的属性分配初始值，此初始值如果没有手动设置则会被设置为数据类型默认初始值(例如int数据类型的初始值为0)。最后调用类的构造方法完成对象初始化操作。

3.2 对象使用

当创建好对象后，需要通过引用方式来访问使用对象，常见的有两种方式，第一种是句柄，第二种是直接指针。

(1)句柄

Java堆中专门划分一部分内存作为句柄池，Java栈中的引用存储的是对象的句柄地址，而句柄地址存储了对象实例数据与数据类型各自的具体地址信息。

使用句柄的好处就是引用中存储的是固定的句柄地址，在对象被移动(垃圾收集时移动对象)时只会改变句柄中实例数据指针，而引用不需要更改。

(2)直接指针

直接指针就是Java栈中的引用直接存储对象的内存地址。使用直接指针最大的好处就是访问速度快，它节省了一次指针定位的时间开销。Sun的Hot Spot虚拟机使用的直接指针访问对象。

(3)对象销毁

对象的销毁回收由虚拟机负责执行，虚拟机判断一个对象是否可以被回收的算法包括引用计数法和根集算法。引用计数法为虚拟机内部为每个对象保存一个引用数量，当一个对象的引用数量为0时，则虚拟机会在下次垃圾回收时将此对象回收。由于引用计数算法无法解决对象之间引用闭环问题，因此出现了根集算法。根集算法原理是从GCRoot(如一个静态变量)开始遍历引用关系，最后对于无法被遍历到的对象则会被回收[9,10]。

4 对象的正确使用(Proper use of object)

掌握Java虚拟机内部原理可有效帮助开发人员编写出高效的Java程序，但虚拟机内部原理的复杂性导致学习成本较高、入门较难。因此对于初学者而言可以边学习边参考一些最佳实践，从而写出高质量Java代码。在编码时正确使用对象主要包括以下几点：

4.1 尽量复用对象，不重复创建

重复创建对象将导致内存占用增大，浪费内存空间。解决方法是重用已创建的对象和通过代码方式限制避免创建重复对象。

(1)使用单例模式。单例模式的作用是确保一个类只能创建出一个对象。实现上主要为：a.显示声明私有构造方法，禁止使用者使用new关键字创建对象；b.在类中创建一个自身类型的对象，并将其定义为私有的和静态的，此变量即为供外部使用的唯一的对象；c.定义一个全局方法，此方法向外部提供唯一的实例变量。

(2)重用对象。适当使用new关键字，在创建完一个对象后，后期考虑不再创建新对象，而是将之前创建对象内部存储数据更新以供下次使用。例如字符串变量，虚拟机对于字符串单独设置了常量池进行存储，因此对于代码String s=hello”和String s=new String(“hello”)而言，前者的效率更高。

4.2 及时清空过期对象

虽然Java虚拟机提供了自动化内存管理，实现了自动垃圾回收，但是为了提高程序运行效率，当在代码中某对象不再使用时，应手动设置对象值为null，这样在下次垃圾回收时此对象即可被回收。例如在当编写数据访问代码时，对于Connection对象的释放即应在使用后立即清除引用，正确写法如下。

```
Connection conn=null；//创建对象，默认值为空
try{
  conn=DriverManager.getConnection(url,username,pwd)；//初始化conn对象
   // ...
 }finally {
 if(null!=conn) {
        try{conn.close()；conn=null；}catch(Exception e) {；}  //使用后立即清除引用
    }
 }
```

5 结论(Conclusion)

理解面向对象思想的核心是正确理解对象概念，从对象到类，从类再到接口等抽象过程。面向对象思想三大特性包括封装、继承和多态，这三大特性在学习和开发过程中是密不可分的，在分析和设计对象与类时以高内聚、低耦合为原则，以提高代码的复用性。Java程序在运行时直接关系到性能的是虚拟机内部对象的管理，正确理解对象的生命周期和在代码中正确使用对象对于编程高效的Java程序具有极大帮助。因此从理解对象的概念、学习对象的正确使用入手，继而进入面向对象分析、设计和编程领域，在开发中才能更好更快地提高对于系统的分析和设计能力。

参考文献(References)

[1] Adam Drozdek.Object-Oriented Programming and Representation of Objects[J].Studies in Logic,Grammar and Rhetoric,2015,40(1):293-302.

[2] Savrun-Yeni,et al.Efficient Hosted Interpreters on the JVM[J].Acm Transactions on Architecture & Code Optimization,2014,11(1):9.

[3] Maplesden,et al.Performance Analysis for Object-Oriented Software:A Systematic Mapping[J].IEEE Transactions on Software Engineering,2015,41(7):691-710.

[4] 严仲兴.Java面向对象程序设计[M].北京:高等教育出版社,2005.

[5] 黄俊爽,等.浅谈Java面向对象程序设计[J].科技信息,2010,13:463;465.

[6] 李永远.JAVA虚拟机相关技术研究与实践[J].信息通信,2015,05:120.

[7] 冯宇.分析Java平台的核心——虚拟机[J].网络安全技术与应用,2015(05):134;138.

[8] 崔行臣,赵佟.Java动态类加载机制分析及其应用[J].计算机系统应用,2013(07):187-191.

[9] 杜天宇,景慎艳.Java虚拟机的系统优化研究[J].电脑知识与技术,2016(01):72-73.

[10] 任嘉光.Java性能优化技术综述[J].信息化建设,2016(06):121.

作者简介：

邢如意(1982-)，男，硕士，讲师，系统分析师.研究领域：软件工程，分布式系统设计与开发.