

本章学习要点

- ◆ 面向对象程序设计的一些基本概念
- ◆ 面向对象程序设计的基本特征
- ◆ 继承的一般实现方法
- ◆ 多态的一般实现方法

3.1 基本知识点

Java 语言的程序设计机制全面、系统地实现面向对象的程序设计。有关面向对象程序设计的一些基本概念主要如下。

- 类：它是每个面向对象的程序的基本结构,包含数据域和操作这些数据。类提供了创建软件对象的模板。
- 实例：实例是以类为模板创建的对象,一个类可以用来生成任意多个实例。
- 封装性：允许或者禁止访问类或对象的数据和成员方法的能力。
- 重载性：允许一个成员方法有多重定义,根据不同场合选用不同的定义。
- 继承性：获得相关类已经具备的一些特征能力。
- 多态性：处理基于公共特征的多个相关类之间的能力,可根据不同的环境调用不同类的方法。

1. 类

在软件设计中,虽然是“面向对象”,即关注的焦点是对象,但对象是依靠类来描述的,类实际上是对某种具有共同特征类型的一类对象的定义,即类定义了一类对象的类型,属于该类型的所有对象都具有相同的变量和方法。用户可以利用相同类型的对象模型创建一个类,表达同一种对象的共同的属性和行为。所以类是对象的软件模型,是一个抽象定义、一个模板,也有人把类称为抽象数据类型。

2. 对象

在面向对象的程序设计方法中,对象是一些相关的变量和方法的软件集,是可以保存状态(信息)和一组操作(行为)的整体。对象用于模仿现实世界中的一些对象,例如桌子、电视、自行车等。现实世界中的对象有两个共同特征,即状态和行为。例如,自行车的状态有车的样式、颜色、当前档位、两个轮子的大小等,其行为有刹车以及改变档位加速、减速等。

在面向对象设计的过程中,既可以利用对象来代表现实世界中的实物对象(例如可用一个动画程序来代表现实世界中的动物),也可以使用软件对象来模拟抽象的概念,例如事件是一个 GUI(图形用户界面)窗口系统的对象,它可以代表用户按下鼠标按钮或者键盘上的

按键所产生的事件。

3. 封装性

具体来说,封装就是指利用抽象数据类型将数据和基于数据的操作结合在一起,数据被保护在抽象数据类型的内部,系统的其他部分只有通过包裹在数据之外被授权的操作才能够与这个抽象数据类型进行交互。对象的封装性消除了传统结构方法中数据与操作分离所带来的种种问题,降低了维护数据与操作之间的相容性的负担;而且把对象的私有数据和公共数据分离开,保护了私有数据,减少了模块间可能产生的干扰,达到降低程序复杂性、提高可控性的目的,提高了程序的可重用性和可维护性。

4. 继承性

继承是面向对象的程序中两个类之间的一种关系,是一个类可以从另一个类(即它的父类)继承状态和行为。被继承的类也可称为超类,继承父类的类称为子类。

继承性为组织和构造软件提供了一个强大且自然的机理。一个父类可以同时拥有多个子类,这时这个父类实际上是所有子类的公共变量和方法的集合,每一个子类从父类中继承了这些变量和方法。

继承的优越性在于:通过使用继承,程序员可以在不同的子类多次重新使用在父类中的代码,使程序结构清晰,而子类又可以提供一些特殊的行为,这些特殊的行为在父类中是没有的。

采用继承机制来组织、设计系统中的类,可以提高程序的抽象程度,使之更接近于人们的思维方式,同时也可以提高程序的可重用性,从而提高程序的开发效率,降低维护成本。

5. 多态性

多态是指同名的多个方法共存于同一个程序中的情况。在软件设计过程中,有时候需要利用这种“重名”现象来提高程序的抽象性和简洁性。

对象的动态行为是由对象间相互通信而发生的。多态的含义是一个消息可以与不同的对象结合,产生不同的行为,而且这些对象属于相同的类,也可以属于不同的类。同一消息可以用不同方法解释,方法的解释依赖于接收消息的类,而不依赖于发送消息的实例。多态通常是一个消息在不同的类中用不同方法实现的。

多态的实现是由消息的接收者确定一个消息应如何解释,而不是由消息的发送者确定,消息的发送者只需知道另外的实例可以执行一种特定操作即可,这一特性对于可扩充系统的开发是特别有用的。按这种方法可开发出易于维护、可塑性好的系统。

3.2 教材习题与解答

1. 什么是面向对象技术?

答:面向对象技术包括面向对象分析(Object-Oriented Analysis, OOA)、面向对象设计(Object-Oriented Design, OOD)及面向对象程序设计(Object-Oriented Programming, OOP)3部分内容。

(1) 面向对象分析指软件需求分析的一种带有约束性的方法,用于软件开发过程中的问题定义阶段。其主要活动是对问题进行抽象建模,包括实例建模、类和对象建模、组件建模和分布建模等,产生一种描述系统功能和问题领域的基本特征的综合文档。

(2) 面向对象设计是将面向对象分析所创建的分析模型转变作为软件构造蓝图的设计模型。面向对象设计的独特性在于其具有基于抽象、信息隐蔽、功能独立性和模块性建造系统 4 个重要软件设计概念的能力。

(3) 面向对象程序设计是指使用类和对象以及面向对象特有的概念进行编程,在结构化程序设计的基础上,于 20 世纪 80 年代初涌现出来的一种程序设计方法。

2. 什么是面向对象程序设计? 面向对象程序设计的优点有哪些?

答: 面向对象程序设计(Object-Oriented Programming, OOP)是一种程序设计范型,同时也是一种程序开发的方法。一般定义为按人们认识客观世界的系统思维方式,采用基于对象的概念建立模型,模拟客观世界分析、设计、实现软件的办法。通过面向对象的思想编程使计算机软件系统能与现实世界中的系统一一对应。

使用面向对象程序设计有以下优点。

(1) 易维护: 采用面向对象思想设计的结构可读性高,由于继承的存在,即使改变需求,维护也只是在局部模块,所以维护起来非常方便且成本较低。

(2) 质量高: 在设计时可重用现有的,在以前项目的领域中已被测试过的类,使系统满足业务需求并具有较高的质量。

(3) 效率高: 在软件开发时,根据设计的需要对现实世界中的事物进行抽象,产生类。使用这样的方法解决问题接近于人们日常生活和自然的思考方式,势必会提高软件开发的效率和质量。

(4) 易扩展: 由于继承、封装、多态的特性,自然设计出高内聚、低耦合的系统结构,使得系统更灵活、更容易扩展,而且成本较低。

3. 简述抽象与封装、继承与多态性等的概念。

答: 抽象就是忽略一个主题中与当前目标无关的那些方面,以便更充分地注意与当前目标有关的方面。抽象并不打算了解全部问题,而只是选择其中的一部分,暂时不用部分细节。抽象包括两个方面,一是过程抽象,二是数据抽象。

继承是一种连接类的层次模型,并且允许和鼓励用户对类的重用,它提供了一种明确表述共性的方法。对象的一个新类可以从现有的类中派生,这个过程称为类继承。新类继承了原始类的特性,新类称为原始类的派生类(子类),原始类称为新类的基类(父类)。派生类可以从它的基类那里继承方法和实例变量,并且类可以修改或增加新的方法,使之更适合特殊的需要。

封装是把过程和数据包围起来,对数据的访问只能通过已定义的界面。面向对象计算始于这个基本概念,即现实世界可以被描绘成一系列完全自治、封装的对象,这些对象通过一个受保护的接口访问其他对象。

多态性是指允许不同类的对象对同一消息做出响应。多态性包括参数化多态性和包含多态性。多态性语言具有灵活、抽象、行为共享、代码共享的优势,很好地解决了应用程序函数同名的问题。

4. 什么是对象的自治性?

答: 对象的自治性是指对象作为独立的整体可以通过自身定义的操作来管理自己。一个对象的操作可以完成两类功能: 一是修改自身的状态; 二是向外界发布消息。当一个对象想影响其他对象时,需要调用那个对象的方法,而不是直接去改变那个对象。对象的这种

自治性能够使得所有修改对象的操作都可以以对象自身所具有的一种行为的形式存在于对象整体之中,从而维护了对象的完整性,有利于对象在不同环境下的重用、扩充和维护。

5. 什么是多态? 面向对象程序设计为什么要引入多态的特性? 使用多态有什么优点?

答: 多态性是对象自动根据实际情况调用不同类层上适当的同名方法的一种现象。

无论在什么时候创建有层次关系的一些类,如果这些类有可能共享一个方法声明,就应当考虑分别在每个类中实现不同的方法。在超类中使用抽象(或普通)的方法,然后在子类中实现(或覆盖)该方法,这样就能产生多态性。

多态性能使用户很容易地在从超类继承来的新类上添加新的功能。

6. 什么是消息?

答: 消息是对象间进行联系或者交互的手段,是一个对象向其他对象发请求执行某个操作的信号或命令。发送消息,也就是对另一个对象的方法的调用。单一的对象通常用处不是很大,所以在一个应用中通常包含许多对象,并且通过消息使对象交互作用,从而表现更为复杂的行为或获得更强的功能。

3.3 补充习题与解答

1. 面向对象的主要特征有哪些方面?

答:

(1) 抽象: 抽象就是忽略一个主题中与当前目标无关的那些方面,以便更充分地注意与当前目标有关的方面。抽象并不打算了解全部问题,而只是选择其中的一部分,暂时不用部分细节。抽象包括两个方面,一是过程抽象,二是数据抽象。

(2) 继承: 继承是一种连接类的层次模型,并且允许和鼓励用户对类的重用,它提供了一种明确表述共性的方法。对象的一个新类可以从现有的类中派生,这个过程称为类继承。新类继承了原始类的特性,新类称为原始类的派生类(子类),原始类称为新类的基类(父类)。派生类可以从它的基类那里继承方法和实例变量,并且类可以修改或增加新的方法,使之更适合特殊的需要。

(3) 封装: 封装是把过程和数据包围起来,对数据的访问只能通过已定义的界面。面向对象计算始于这个基本概念,即现实世界可以被描绘成一系列完全自治、封装的对象,这些对象通过一个受保护的接口访问其他对象。

(4) 多态性: 多态性是指允许不同类的对象对同一消息做出响应。多态性包括参数化多态性和包含多态性。多态性语言具有灵活、抽象、行为共享、代码共享的优势,很好地解决了应用程序函数同名的问题。

2. 为什么说 Java 语言是纯面向对象的语言?

答: Java 是一种面向对象的语言,因为它具有面向对象语言的基本特征,例如抽象与封装、继承与多态等。Java 的程序设计集中于对象及其接口,Java 提供了简单的类机制以及动态的接口模型,实现了模块化和信息封装。Java 类可提供一类对象的原型,再通过继承机制,实现了代码的重用。

3. 什么是对象的安全性?

答: 对象的安全性是指对象具有通过一定的接口和相应的消息机制与外界相联系的特

性,该特性与对象的封装性结合在一起,较好地实现了信息隐藏。这样使得对象成为一个使用方便的“黑匣子”,其中隐藏了私有数据和内部运行机制。在使用对象时用户只需要了解其接口提供的功能和操作,而不必了解对象内部的数据描述和具体的功能实现。

4. 为什么对象具有可扩展性?

答:继承是面向对象的另一个重要特性,通过继承可以很方便地实现应用的扩展和已有代码的重复使用,在保证质量的前提下提高开发效率。

面向对象程序设计最主要的特征是对各对象之间的消息传递和类之间的继承。类的继承用于描述对象、建立问题领域模型以及构造软件系统,为代码重用提供了一种有效的途径。

5. 在定义类中要定义哪些成员? 这些成员又可以分为几类?

答:在定义一个类的时候用变量来表示类的属性,也就是所谓的成员变量,而表现一个类的行为的过程或函数称为成员方法。类的成员变量和成员方法构成类的成员。

在定义一个类的时候实际上存在两种类型,即实例成员和类成员。

6. 为什么要在面向对象技术中使用抽象?

答:在面向对象技术中使用抽象,一方面可以去掉与核心问题无关的细节,使开发工作可以集中在比较关键和主要的部分;另一方面,在数据抽象过程中对数据和分析、辨别与定义可以帮助开发人员对整个问题的认识。最后抽象形成的抽象数据类型是进行进一步设计和实现的基础与依据。

抽象可以帮助人们明确工作的重点,抓住问题的本质,理清问题的脉络。面向对象的软件开发方法之所以能够处理大规模、高复杂度的系统,抽象手段发挥了重要作用。