河南中医药大学智能医学工程专业《计算机程序设计》课程

第10章：继承与多态

黄子杰

河南中医药大学信息技术学院（智能医疗行业学院）与河南方和信息科技有限公司 联合建设

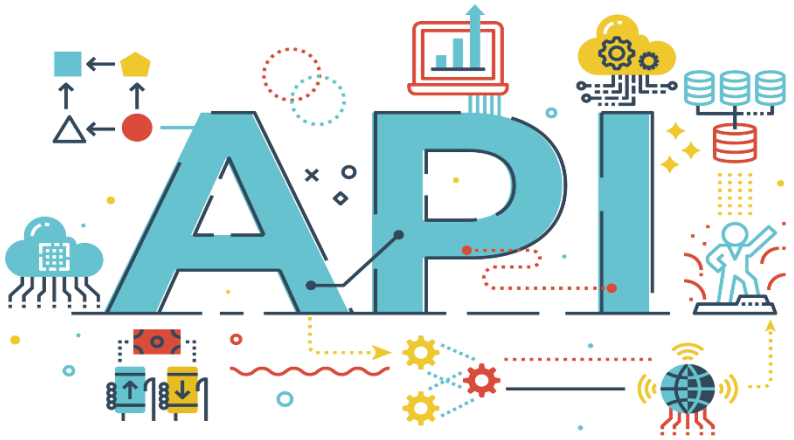
河南中医药大学信息技术学院互联网技术教学团队

https://webdev.hactcm.edu.cn

2024/9/13

本章概要

在面向对象编程（OOP）中，继承和多态是两大核心概念



10.1 继承

继承是面向对象编程中的一个基本特征，它允许我们定义一个类（子类或派生类）来继承另一个类（父类或基类）的属性和方法。这样，子类就拥有了父类的所有功能，并且还可以在此基础上添加新的功能或修改已有的功能。继承提高了代码的复用性，也使得类的结构更加清晰。

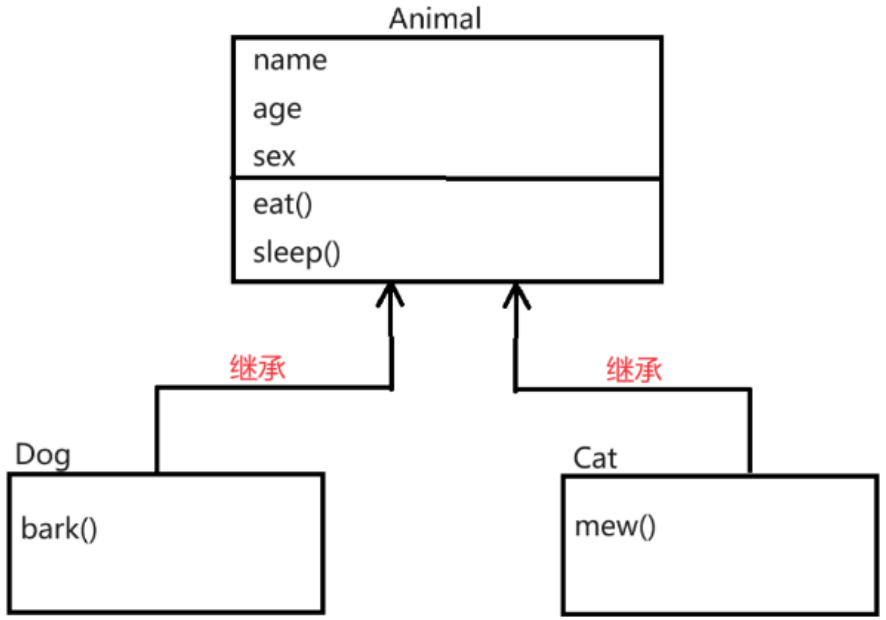
10.1.1 Java 继承的实现

- 详细说明：

关键字：在Java中，使用extends关键字来实现继承。

继承关系：子类继承父类的所有非私有属性和方法。子类可以添加新的属性和方法，也可以重写父类的方法。

Java 继承的实现



父类/基类/超类

子类/派生类

上述图示中，Dog和Cat都继承了Animal类，其中：Animal类称为父类/基类/超类，Dog和Cat可以称为Animal的子类/派生类，继承之后，子类可以复用父类中成员，子类在实现时只需关心自己新增加的成员即可。从继承概念中可以看出继承最大的作用就是：实现代码复用（父类代码一次实现，子类多次使用），还有就是来实现多态。

10.1.2 构造方法在类继承中的作用

- 概述

构造方法在类的继承中扮演重要角色，子类的构造方法可以调用父类的构造方法，从而初始化从父类继承的属性。在Java继承体系中，子类的构造方法可以通过`super`关键字调用父类的构造方法，确保从父类继承的属性得到正确的初始化，是实现继承的关键步骤。

10.1.2 构造方法在类继承中的作用

- 详细说明：

- super关键字：在子类构造方法中，可以使用super关键字调用父类的构造方法。如果不显式调用，Java会自动调用父类的无参构造方法。

- 构造方法链：继承链上的每个构造方法都会被调用，确保所有属性正确初始化。

10.1.2 构造方法在类继承中的作用

- 生活举例：

构造方法：像家族传承的财富，子辈可以在继承家族财富的基础上再添加自己的积蓄。

- 代码示例：

eg.10.1.2.txt

10.2 多态性

多态指的是同一操作作用于不同的对象，可以有不同的解释，产生不同的执行结果。多态性是通过方法的重写（Override）和重载（Overload）来实现的。在Java等语言中，多态性通常与接口和抽象类一起使用，以实现更灵活的设计。

10.2.1 方法的重载

- 概述：

方法的重载（Overloading）是指在同一个类中可以定义多个方法，它们具有相同的名字，但参数不同（类型或数量不同），实现同一操作名在不同情况下执行不同功能。

10.2.1 方法的重载

- 详细说明：

多态性的一种：重载是多态性的一种表现形式，允许同一方法名在不同场景下执行不同操作，增加了代码的灵活性和可读性。

编译期多态：方法的重载在编译期间由编译器决定调用哪个方法，在方法重载中，编译器负责在编译期间根据提供的参数决定调用哪个具体的方法，确保了正确性和效率，避免了运行时错误。

10.2.1 方法的重载

- 生活举例：

方法重载：就像一个人有多个电话号码，朋友打手机，公司打座机。

- 代码示例：

eg.10.2.1.txt

10.2.2 方法的覆盖

- 概述：

方法的覆盖（Overriding）是指子类重写从父类继承的方法，提供自己的实现。它是多态性的重要表现之一，允许不同类的对象对同一消息做出不同的响应。

10.2.2 方法的覆盖

- 详细说明：

动态绑定：确保在运行时根据对象的实际类型调用正确的方法。

@Override注解：在Java中，使用@Override注解可以明确指出一个方法是有意覆盖父类(超类)中的方法。这要求子类必须提供该方法的新实现，增强了代码的可读性和可维护性。

10.2.2 方法的覆盖

- 生活举例：

方法覆盖：就像孩子学会了父母的技能，但根据自己的风格来表现。

- 代码示例：

eg.10.2.2.txt

10.3 几个特殊类

- Object类
- Class类

10.3.1 Object类

- 概述

Object类是Java中的根类，所有类都直接或间接继承自Object类。
它提供了一些基本方法，如equals()、hashCode()、toString()等。

10.3.1 Object类

- 生活举例：

Object类：就像所有物品的基本模具，不同的物品都源于这个模具。

代码示例：

eg.10.3.1.txt

10.3.2 Class类

- 概述：

Class类是Java反射机制的核心，用于获取类的元数据，。它代表类和接口在运行时的实体。通过Class对象，可以获取类的相关信息，如构造函数、方法、字段等。

10.3.2 Class类

- 详细说明：

- 常用方法：
- getName()：获取类的全限定名，提供了一种识别和引用特定类的方式。
- getMethods()：获取类的所有方法，通过该方法可以查看一个类的所有方法，包括公共、保护、默认（包）访问和私有方法，从而全面了解类的行为。
- getFields()：获取类的所有字段。

10.3.2 Class类

- 生活举例：

Class类：就像一本说明书，告诉你一个物品的所有特征和功能。

代码示例：

eg.10.3.2.txt

10.4 对象引用转换和访问继承成员

在Java中，对象的引用转换和访问继承成员遵循以下规则：

向上转型（Upcasting）：

是安全的，因为子类包含了父类的所有成员。

可以直接进行，无需显式转换。

子类实例可以被赋予父类类型的引用。

向下转型（Downcasting）：

不安全，可能导致ClassCastException。

需要显式转换，使用(类型)对象引用进行转换。

转换之前应该使用instanceof检查确保安全。

举例说明：

eg10.1-14.txt

10.4.1 对象引用转换

- 概述：

对象引用转换是指将一个子类对象的引用赋给父类变量，或将父类引用转换为子类引用。

10.4.1 对象引用转换

- 详细说明：

向上转型（Upcasting）：它允许将子类对象自动转换为父类引用，这种转换是安全的，因为它保证了子类拥有父类的所有特性和行为。

向下转型（Downcasting）：需要显式转换，将父类引用转换为子类引用，这需要使用强制类型转换，并且仅在引用的对象确实是子类实例时才安全，否则会引发ClassCastException异常。

10.4.1 对象引用转换

- 生活举例：

引用转换：就像员工调职到不同部门，可能需要适应新环境。

- 代码示例：

eg.10.4.1.txt

10.4.2 访问继承成员

- 概述

子类可以访问继承自父类的非私有成员，且可以通过super关键字访问被覆盖的方法和属性。

10.4.2 访问继承成员

- 详细说明：

super关键字：用于在子类中调用父类的方法或访问父类的属性。

访问顺序：子类优先访问自身成员，如果没有再查找父类成员。

10.4.2 访问继承成员

- 生活举例：

访问继承成员：就像孩子在需要时可以依靠父母的经验和资源。

- 代码示例：

eg.10.4.2.txt

10.5 访问控制符

- 概述：

访问控制符用于控制类、方法和变量的可见性和访问权限。Java中有四种主要的访问控制符：public、protected、default（无修饰符）和private。

10.5 访问控制符

- 详细说明：

public：所有类都可以访问。
protected：子类和同一包中的类可以访问。
default（无修饰符）：同一包中的类可以访问。
private：只有类内部可以访问。

10.5 访问控制符

- 生活举例：

访问控制：像公司内的权限控制，有的文件是公开的（public），有的只对团队开放（protected），有的仅限部门内部查看（default），有的则是完全私密的（private）。

代码示例：

eg.10.5.txt

10.6 final修饰符的使用

- 概述：

final修饰符可以用来修饰类、方法和变量，表明它们不能被修改或继承。

10.6 final修饰符的使用

- 详细说明：

final类：使用final关键字修饰的类不能被其他类继承，这确保了类的独特性和不可更改性，常用于设计不希望被扩展的基础类或核心组件。

final方法：当一个方法被声明为final时，它不能被子类覆盖，这保证了方法的行为在继承链中保持一致，防止子类修改其行为。

final变量：final变量一旦赋值后就不能更改，适用于那些需要保持常量的值，如配置参数或不变的状态信息，确保数据的不变性和安全性。

10.6 final修饰符的使用

- 生活举例：

final修饰符：像一块不可动摇的基石，一旦确立就不能再修改。

- 代码示例：

eg.10.6.txt

本章重点

- Java 继承的实现
- 构造方法在类继承中的作用
- 多态性
- 方法的重载
- 方法的覆盖
- 对象引用转换和访问继承成员
- 访问继承成员

本章作业

作业一：医院系统类设计与继承实现

目的：通过医院系统的类设计，练习Java的继承、构造方法、多态性、访问控制符以及final修饰符的使用。

要求：

1 设计一个Person类作为基类，包含姓名、年龄等基础属性及相应的构造方法和getter/setter方法。

2 创建Patient（患者）类继承自Person类，并添加特定的属性如病历号、诊断结果等，以及对应的构造方法和getter/setter方法。注意，患者类中的某些信息可能需要被设为私有（private），并通过公共（public）方法访问，以练习访问控制符的使用。

3 创建一个Doctor（医生）类，它不需要直接继承自Person类，但可以设计为包含对Patient对象的引用和操作，如查看病历、开具诊断等。这里可以展示多态性的应用，比如医生可以处理不同类型的患者（虽然在这个场景中患者类只有一个，但可以设想为多个子类的情况）。

4 在Doctor类中，使用final修饰符定义一个或多个方法，表示这些方法是最终方法，不可被子类覆盖。

5 编写一个简单的测试类，创建医生和患者的对象，演示医生如何查看患者的病历信息，并验证final修饰符的使用。

本章作业

作业二：医院挂号系统的方法重载与覆盖

目的：通过医院挂号系统的实现，练习Java的方法重载、方法覆盖以及对象引用转换。

要求：

1 设计一个RegistrationService（挂号服务）接口，定义一个或多个抽象方法用于挂号，比如register(Patient patient)。

2 实现RegistrationService接口的多个子类，分别代表不同的挂号方式，如OnlineRegistration（在线挂号）和CounterRegistration（窗口挂号）。在这些子类中，实现register方法的具体逻辑，并考虑方法3 3 重载的情况，比如CounterRegistration类可以提供一个额外的register(Patient patient, String doctorName)方法，允许在挂号时指定医生。

4 编写一个测试类，模拟患者使用不同的挂号方式进行挂号，展示方法重载和接口实现（即多态性）的应用。

5 在测试类中，尝试将RegistrationService接口的对象引用指向其子类的对象，并调用register方法，观察对象引用转换的效果。

计算机程序设计课程学习平台

面向河南中医药大学智能医学工程专业使用

河南中医药大学信息技术学院（智能医疗行业学院）与河南方和信息科技有限公司 联合建设

河南中医药大学信息技术学院互联网技术教学团队