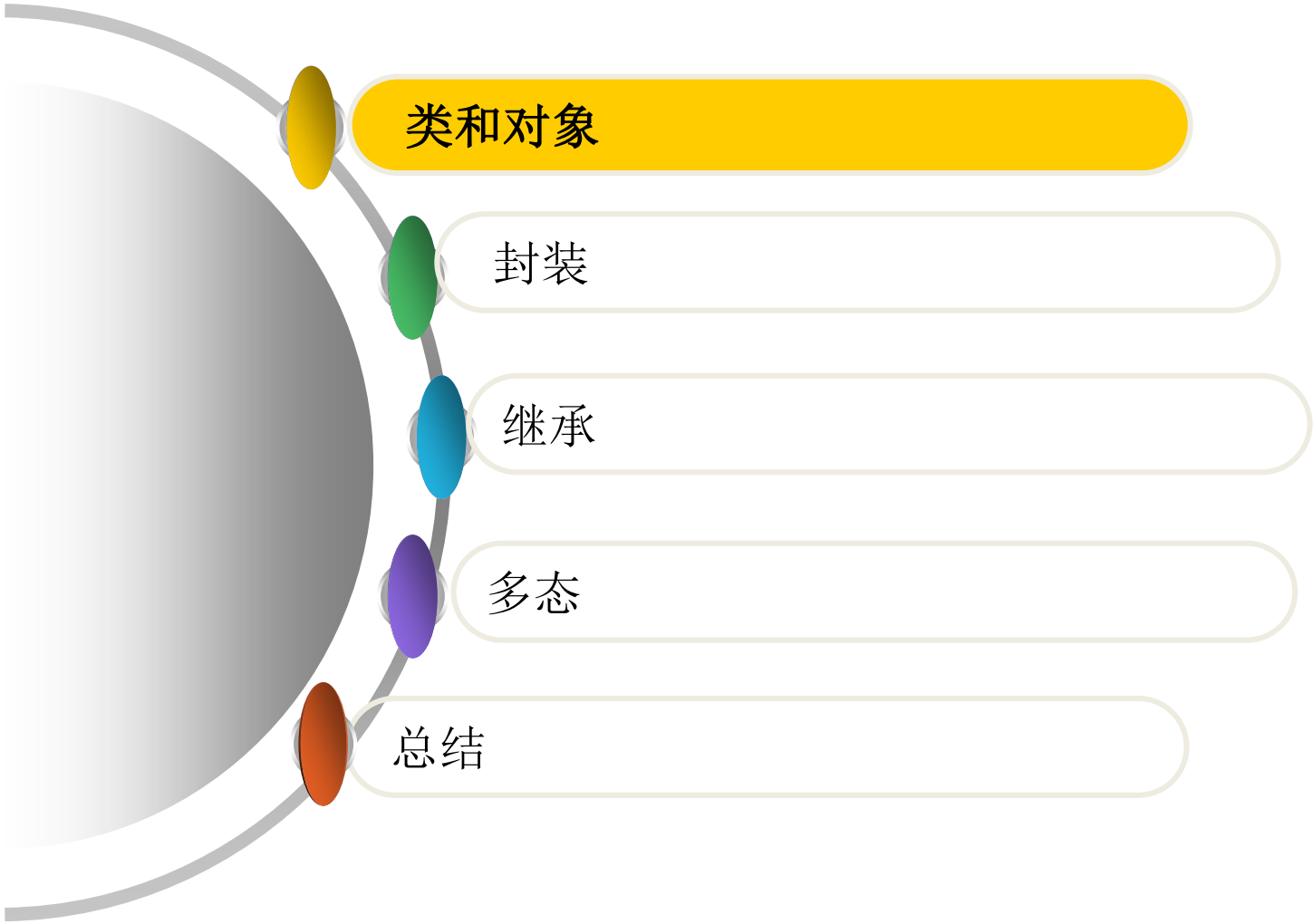
第二讲 Java面向对象

初版时间: 2015年5月1日
初版制作者: 林焕彬
教材版本号：szlanyou-V1.1

修订时间:
修订者:
教材版本号：szlanyou-V1.1

目录

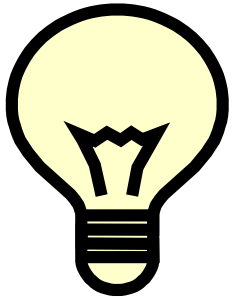


一、Java类和对象 1-1

提问

说一说教室里的对象，描述它们的属性和方法是什么？

万物皆对象



一、Java类和对象 1-2

学生对象
姓名—张浩
年龄—20
体重—60kg

对象：用来描述客观事物的一个实体
属性：对象具有的各种特征
方法：对象执行的操作

类：具有相同属性和方法的一组对象的集合
类是对象的抽象，对象是类的具体

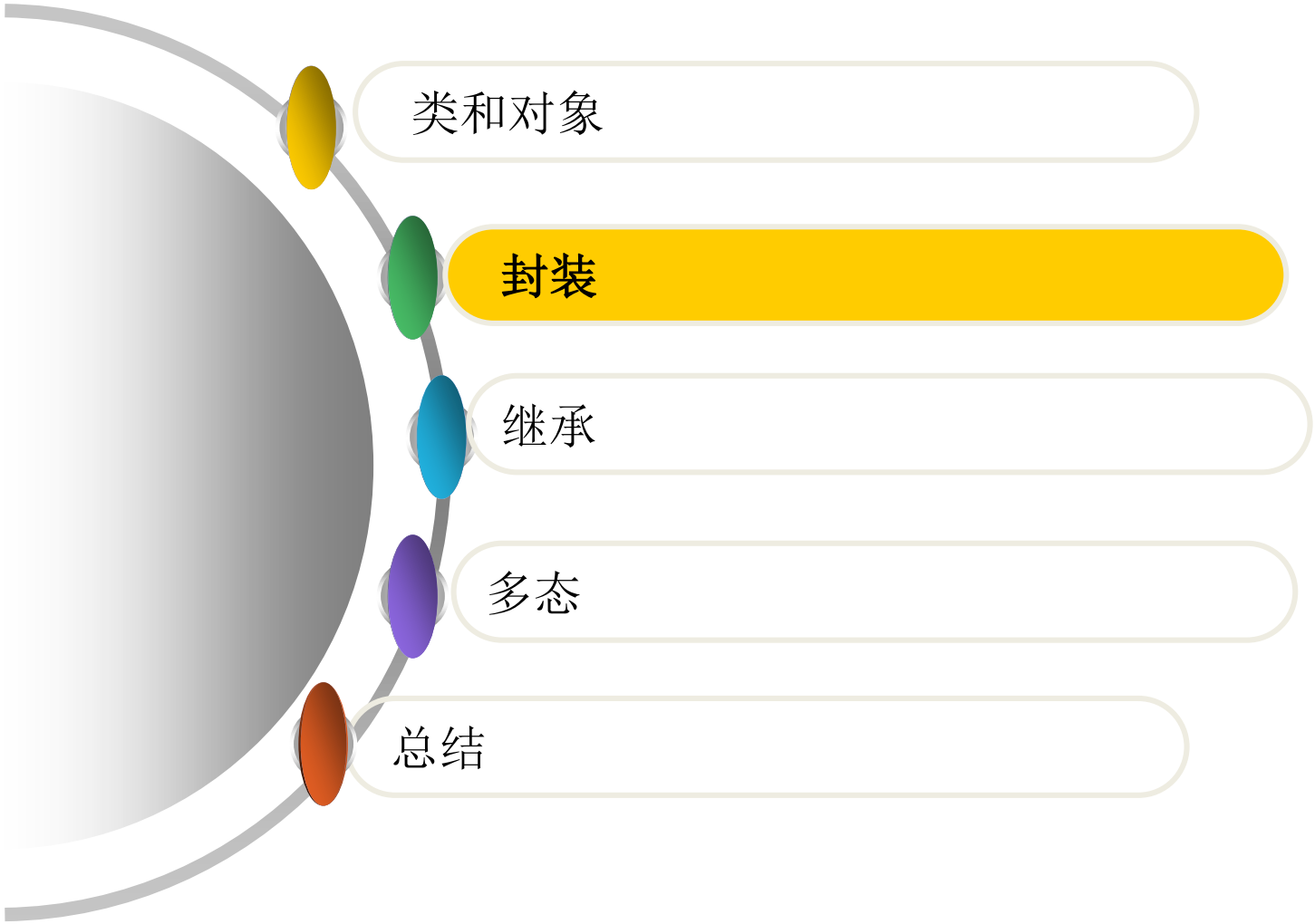
员变量）

员方法）

体重—62kg

操作：
学习
……

目录



二、封装 2-1

提问

对象数组中定义Student对象的两种方式，出现如下问题，如何避免？

```
Student stu1=new Student();
stu1.age=-20;
……
```

```
Student stu1=new Student("张浩",-20,60);
……
```

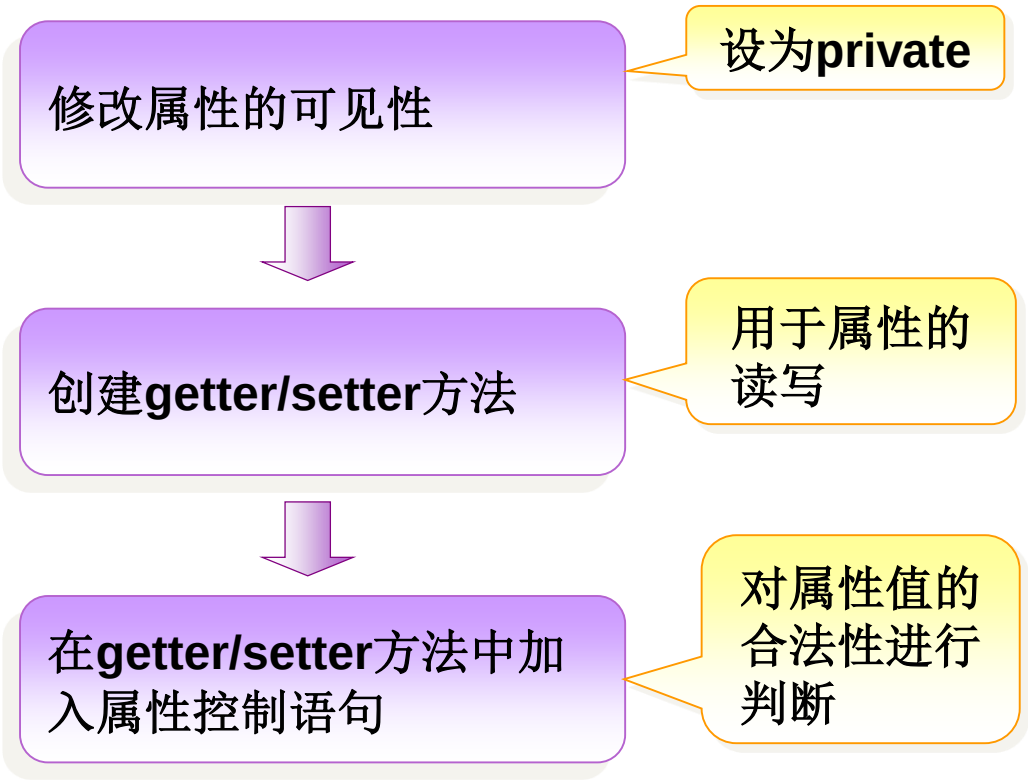
面向对象的三大特征：封装、继承和多态
封装：将类的某些信息隐藏在类内部，不允许外部程序直接访问，而是通过该类提供的方法来实现对隐藏信息的操作和访问

二、封装的好处 2-2

封装的好处：将类的某些信息隐藏在类内部，不允许外部程序直接访问，而是通过该类提供的方法来实现对隐藏信息的操作和访问

二、如何进行封装 2-3

- 封装的步骤



二、访问修饰符 2-4

作用域 修饰符	同一个类中	同一个包中	子类中	任何地方
private	可以	不可以	不可以	不可以
默认修饰符	可以	可以	不可以	不可以
protected	可以	可以	可以	不可以
public	可以	可以	可以	可以

二、封装的实现 2-5

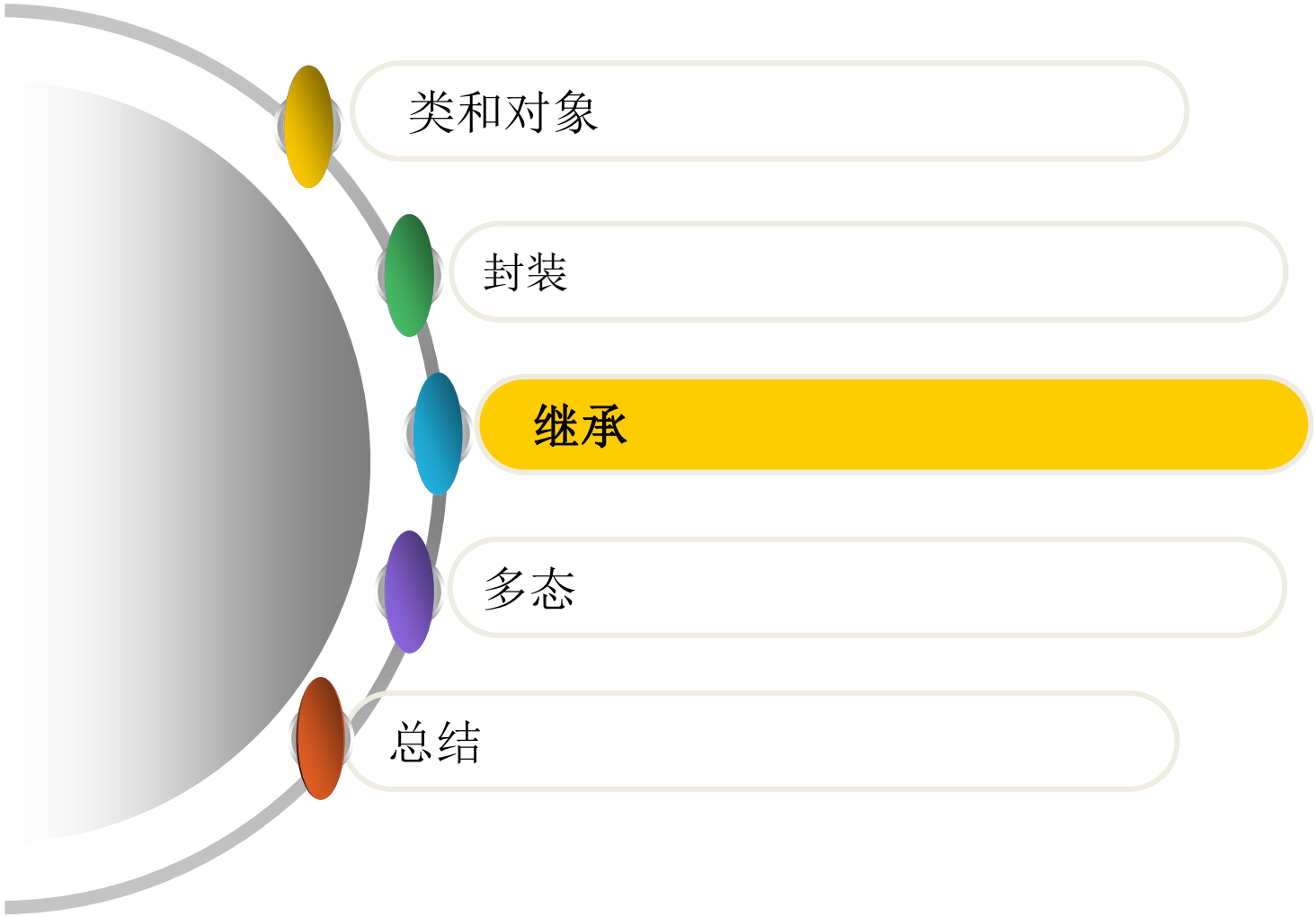
访问修饰符

```
private int age;
public getAge(){
    return age;
}
public setAge(int age){
    if (age < 0 || age >150) {
    ...省略...
    }
    this.age = age;
}
```

设置setter/getter方法

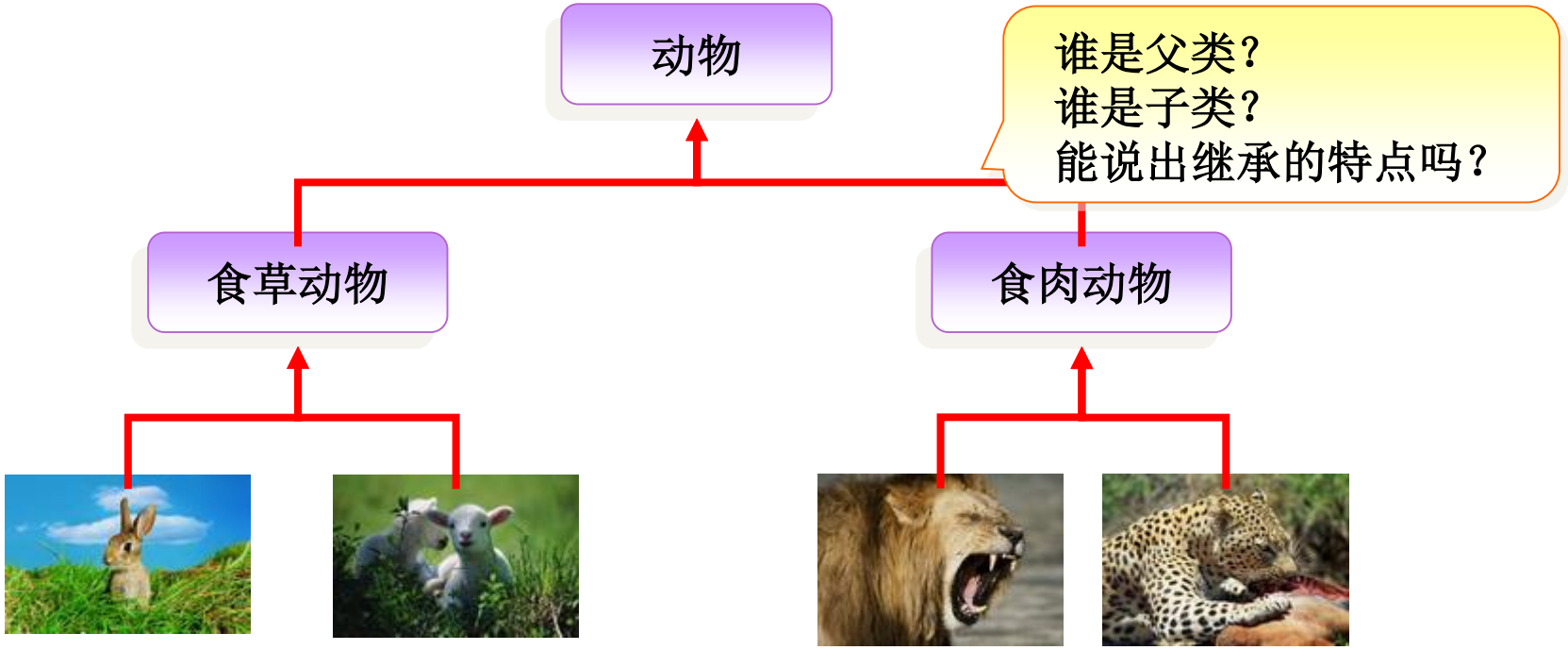
对属性值的合法性进行判断

目录



三、继承 3-1

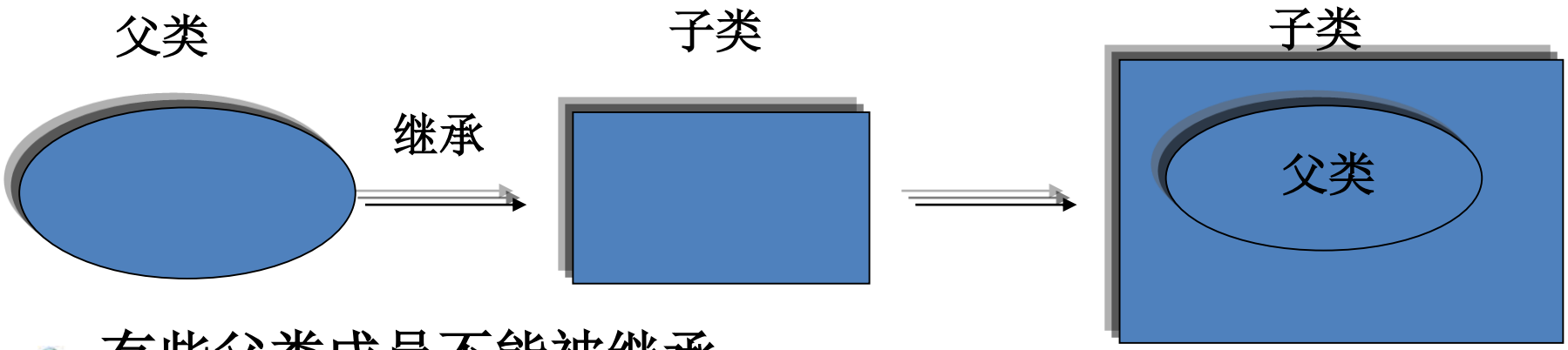
- 生活中，继承的例子随处可见



继承需要符合的关系：is-a，父类更通用、子类更具体

三、继承 3-2

- Java继承是使用已存在的类的定义作为基础建立新类的技术：



- 有些父类成员不能被继承：

1）private成员；2）子类与父类不在同包，使用默认访问权限的成员；

3）构造方法

- 单根继承，即一个类只能有一个直接父类

三、为什么需要继承 3-3

- 开发教员类，其中教员分为Java教员以及.NET教员，各自的要求如下：
  - Java教员
    - 属性：姓名、所属中心
    - 方法：授课（步骤：打开Eclipse、实施理论课授课）、自我介绍
  - .NET教员
    - 属性：姓名、所属中心
    - 方法：授课（步骤：启动Visual Studio .NET、实施理论课授课）、自我介绍

三、为什么需要继承 3-4

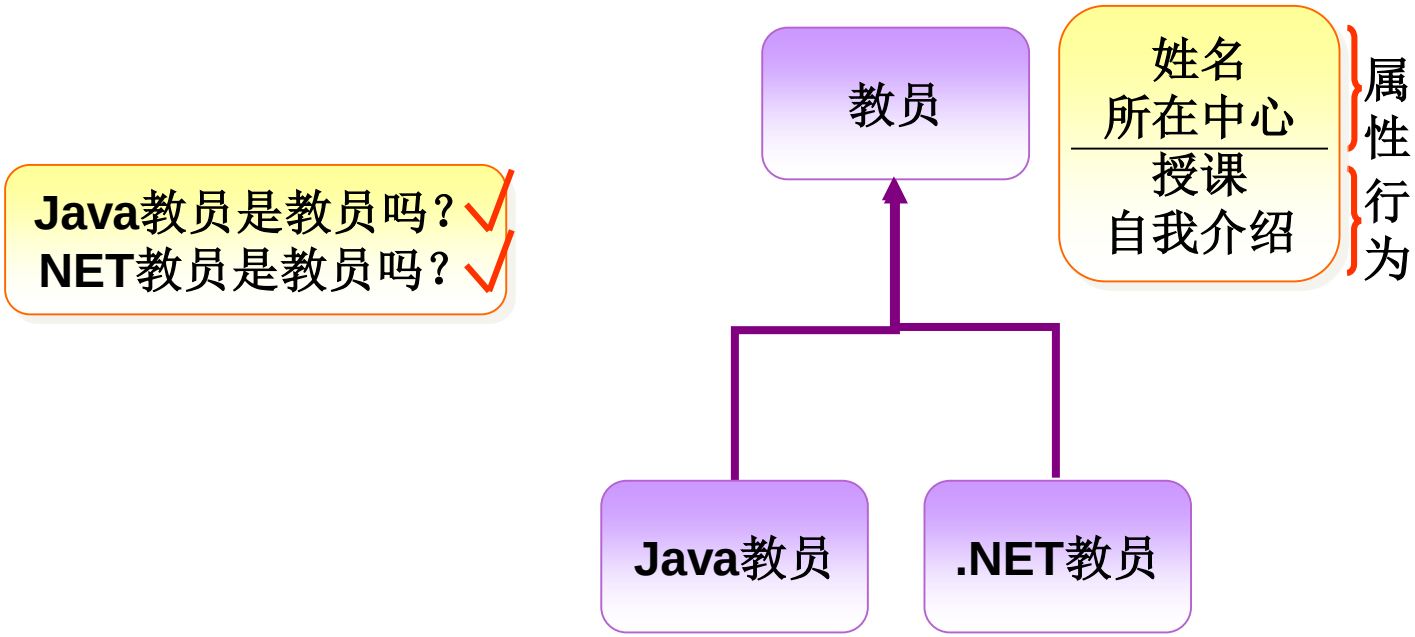
```
public class NETTeacher {
    private String name; // 教员姓
    private String school; // 所在中
    public NETTeacher (String my
    String mySchool) {
        name = myName;
        school = mySchool;
    }
    public void giveLession(){
        System.out.println("启动 V
        Studio .NET");
        System.out.println("知识点
        System.out.println("总结提
    }
     public void introduction() {
        System.out.println("大家好
        + school + "的" + name +
     }
}
```

```
public class JavaTeacher {
    private String name; // 教员姓名
    private String school; // 所在中心
    public JavaTeacher(String myName,
    String mySchool) {
        name = myName;
        school = mySchool;
    }
    public void giveLession(){
        System.out.println("启动 Eclipse");
        System.out.println("知识点讲解");
        System.out.println("总结提问");
    }
    public void introduction() {
        System.out.println("大家好！ 我是"
        + school + "的" + name + ".");
    }
```

代码存在重复，违背了“write once, only once”的原则

三、如何实现继承 3-5

- 如何改进？有没有可能建立继承关系，让子类自动继承父类的属性和方法？



三、如何实现继承 3-6

- 在父类中只定义一些通用的属性与方法，例如：

```
public class Teacher {
        private String name;   // 教员姓名
        private String school; // 所在中心
        public Teacher(String myName, String mySchool) {
              //初始化属性值
        }
        public void giveLesson() {  //授课方法的具体实现  }
        public void introduction() { //自我介绍方法的具体实现  }
}
```

- 在Java语言中，用extends关键字来表示一个类继承了另一个类，例如：

```
public class JavaTeacher2  extends  Teacher {
      //其余代码省略
}
```

三、如何实现继承 3-7

- 子类自动继承父类的属性和方法，子类中可以定义特定的属性和方法

```
public class Teacher {
    private String name;  // 教员姓名
    private String school;// 所在中心
    public Teacher (String
        myName, String mySchool) {
        //初始化属性值
    }
    public void giveLesson() {
        //授课方法的具体实现
    }
    public void introduction() {
        //自我介绍方法的具体实现
    }
}
```

父类中的属性和方法可以被子类继承

```
public class JavaTeacher2 extends
Teacher {
    public JavaTeacher2(String myName,
    String mySchool) {
        super(myName, mySchool);
    }
    public void giveLesson(){
        System.out.println("启动Eclipse");
        super.giveLesson();
    }
}
```

三、如何实现继承 3-8

- 方法重写：子类和父类的方法具有相同的名称、参数列表、返回类型

```
public class Teacher {
    public void giveLesson(){
        System.out.println("知识点讲解");
        System.out.println("总结提问");
    }
}
```

```
public class JavaTeacher2 extends Teacher {
    public void giveLesson(){
        System.out.println("启动 Eclipse");
        super.giveLesson();
    }
}
```

三、如何实现继承 3-9

- 子类的构造方法中，通过super关键字调用父类的构造方法

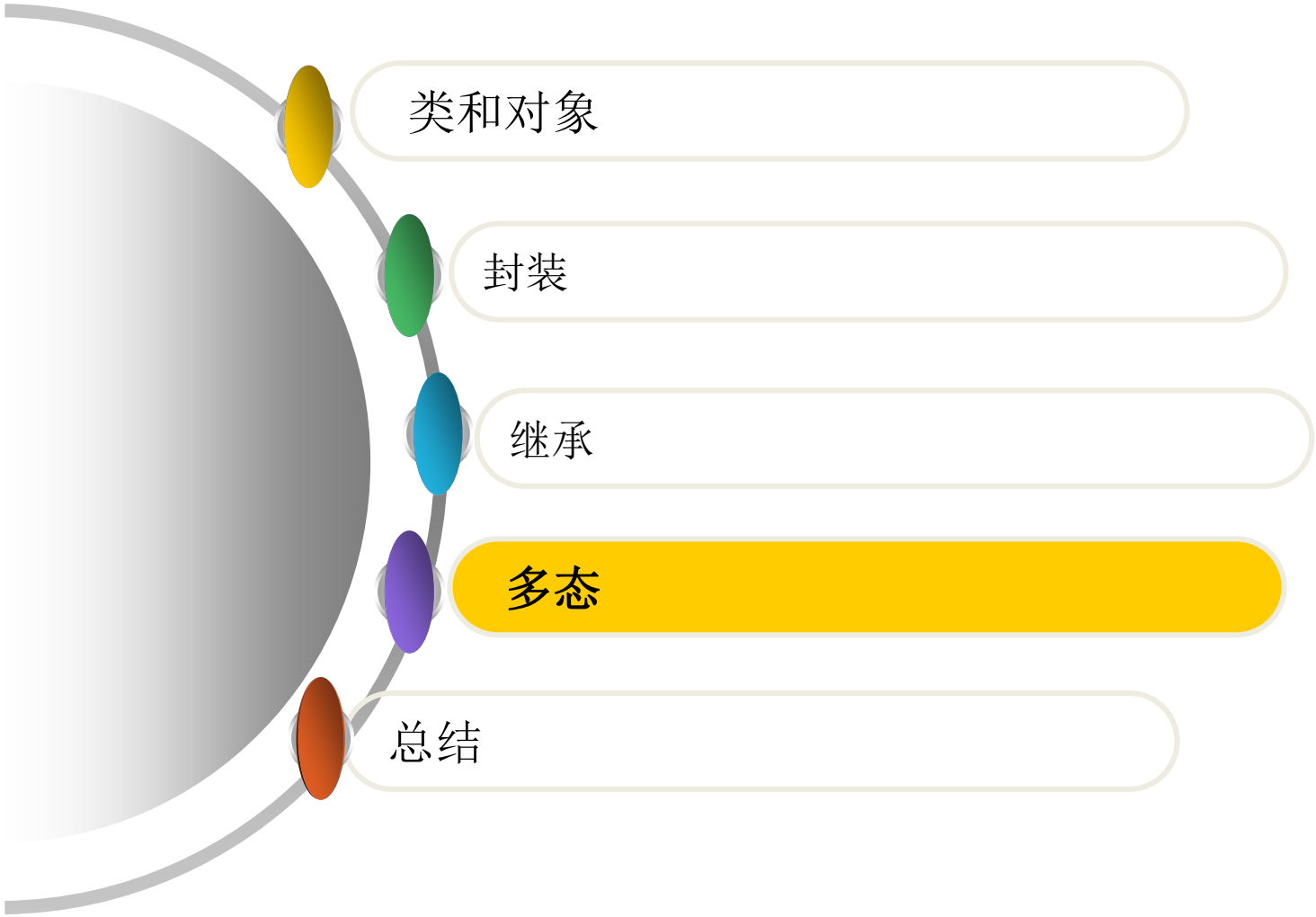
```
public class JavaTeacher2 extends Teacher {
    public JavaTeacher2(String myName, String mySchool) {
        super(myName, mySchool);
    }
}
```

通过调用父类的构造方法，完成对属性值的初始化

- 方法重写后，通过super关键字调用父类的方法

```
public class JavaTeacher2 extends Teacher {
    public void giveLesson(){
        System.out.println("启动 Eclipse");
        super.giveLesson();
    }
}
```

目录

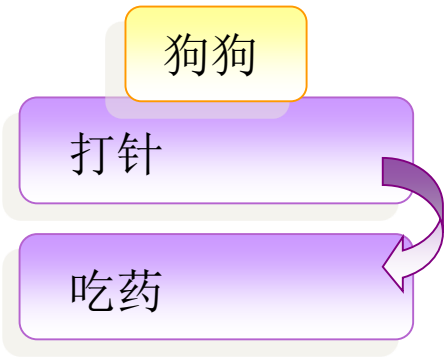


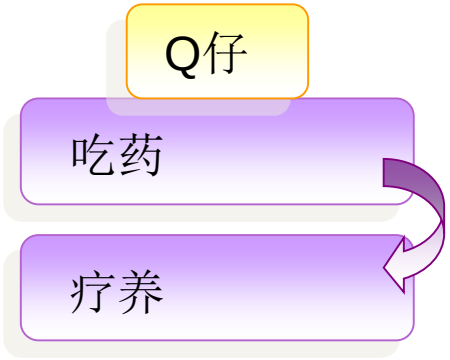
四、为什么需要多态 4-1

- 宠物生病了，需要主人给宠物看病

1）不同宠物看病过程不一样

2）不同宠物恢复后体力值不一样





四、为什么需要多态 4-2

提问

如下主人类中给宠物看病的方法，如果又需要给XXX宠物看病，怎么办？

```
//给狗看病
public void cure(Dog dog) {
        if (dog.getHealth() < 50) {
            dog.setHealth(60);
            System.out.println("打针、吃药");
        }
    }
```

```
//给企鹅看病
public void cure(Penguin penguin){
        if (penguin.getHealth() < 50) {
            penguin.setHealth(70);
            System.out.println("吃药、疗养");
        }
    }
```

添加XXX类，继承Pet类
修改主人类，添加给XXX看病的方法

频繁修改代码，代码可扩展性、可维护性差。使用多态优化

四、多态的实现 4-3

使用多态优化后的代码

提示：
父类Pet可以声明为抽象类；
toHospital()可以声明为抽象方法

Dog类

```
public class Dog extends Pet {
    public
        this
        Sys
    }
}
```

3

```
(Pet pet) {
th() < 50)
tal();
```

又要给XXX看病时，只需：
1. 编写XXX类继承Pet类（旧方案也需要）
2. 创建XXX类对象（旧方案也需要）
3. 其他代码不变（不用修改Master类）

多态：同一个引用类型，使用不同的实例而执行不同操作。方法重写是实现多态的基础

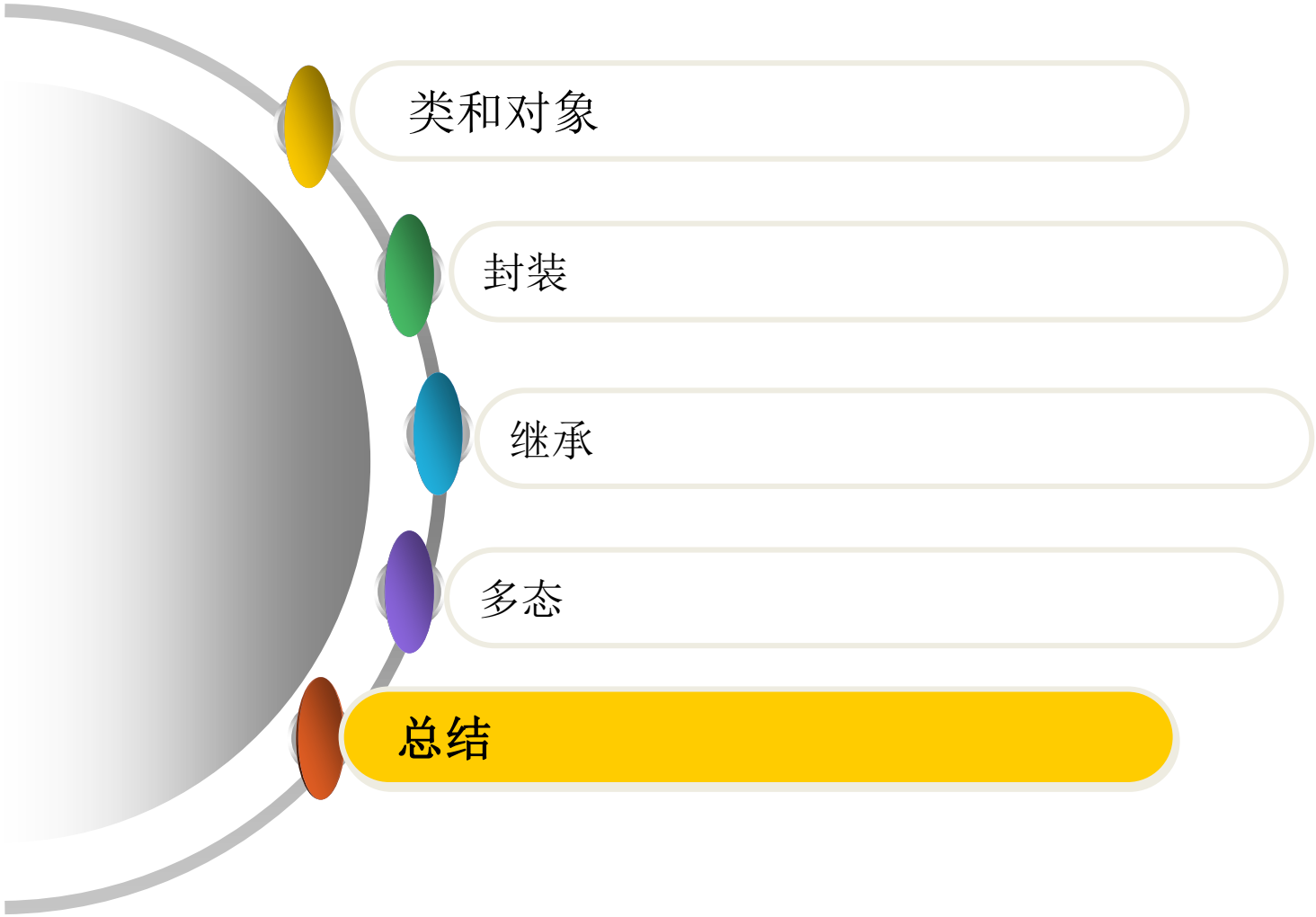
Penguin

```
public class
    public void toHospital() {
        this.setHealth(70);
        System.out.println("吃药、疗养");
    }
}
```

4

```
Pet pet = new Dog();
Master master = new Master();
master.cure(pet);
… …
```

目录



五、总结

提问

类是具有相同属性和方法的一组对象的集合。

封装是将类的某些信息隐藏在类内部，不允许外部程序直接访问。

Java继承是使用已存在的类的定义作为基础建立新类的技术。

多态是同一个引用类型，使用不同的实例而执行不同操作。

五、上机练习

练习

- *需求说明*
  - 编写宠物类Pet及其子类Dog（狗）、Penguin（企鹅），Cat（猫）等，其中宠物类定义了看病的方法，子类分别重写了看病的方法。请编写测试方法分别实例化各种具体的宠物，调用看病的方法。

完成时间：10分钟

共性问题集中讲解

Thank you

Q&A