学号 E21714049 专业 计算机科学与技术 姓名 梅世祺
实验日期 2019.10.09 教师签字 成绩

实验报告

【实验名称】接口、内部类、匿名类和异常类

【实验目的】

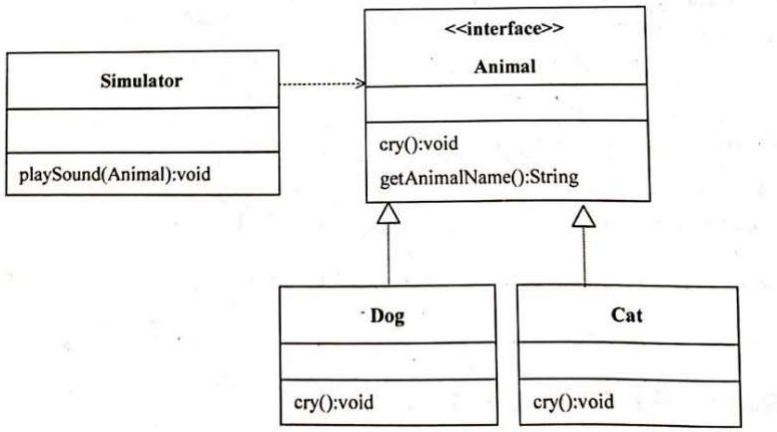
学习 Java 中的接口基本概念，接口回调，了解内部类、匿名类和异常类的基本概念。

【实验原理】

- 设计一个动物声音“模拟器”,希望模拟器可以模拟许多动物的叫声,要求如下。编写接口 Animal。接口 Animal 有两个抽象方法 cry()和 getAnimalName(),即要求各种具体的动物给出自己的叫声和种类名称
- 编写模拟器类 Simulator。该类有一个 playsound(Animalanimal)方法,该方法的数是 Animal 类型。即参数 animal 可以调用 Animal 的子类重写的 cry()方法播放

具体动物的声音,调用子类重写的 getAnimalName()方法显示动物种类的名称。

- 编写 Animal 类的子类:Dog 和 Cat 类。



【实验内容】

首先编写接口 Animal:

```java
interface Animal {
    void cry();
    String getAnimalName();
}
```

接着分别编写 Cat 类和 Dog 类来实现接口 Animal 中的抽象方法:

```java
public class Cat implements Animal {
    public void cry() {
        System.out.println("Miao...Miao...");
    }

    public String getAnimalName() {
        return "Cat";
    }
}
```

```java
public class Dog implements Animal {
    public void cry() {
        System.out.println("Wang...Wang...");
    }

    public String getAnimalName() {
        return "Dog";
    }
}
```

然后编写 Simulator 类，实现 playSound()实例方法：

```java
public class Simulator {
    public void playSound(Animal animal) {
        System.out.println(animal.getAnimalName());
        animal.cry();
    }
}
```

最后编写 Application 主类测试代码：

```java
public class Application {
    Run | Debug
    public static void main(String[] args) {
        Simulator simulator = new Simulator();
        simulator.playSound(new Dog());
        simulator.playSound(new Cat());
    }
}
```

运行结果：

```
/usr/lib/jvm/java-11-openjdk-amd64/bin/java -Dfile.encoding=UTF-8 -cp /home/lolimay/.vscode-server/data/User/w
orkspaceStorage/e905fefc39f8d668c5396f9d79d25c5b/redhat.java/jdt_ws/jdt.ls-java-project/bin Application
$ /usr/lib/jvm/java-11-openjdk-amd64/bin/java -Dfile.encoding=UTF-8 -cp /home/lolimay/.vscode-server/data/User
/workspaceStorage/e905fefc39f8d668c5396f9d79d25c5b/redhat.java/jdt_ws/jdt.ls-java-project/bin Application
Dog
Wang...Wang...
Cat
Miao...Miao...
$
```

【小结或讨论】

1. 抽象类和抽象方法

抽象类不能直接通过 new 关键字实例化对象，因为它包含的抽象方法只有声明没有实现。必须要在其子类中实现所有抽象方法后，通过子类实例化。

2. 抽象类与接口的区别

接口是对行为的抽象。

- 属性被隐式指定为 public static final，即接口中的成员变量全都是全局常量；
- 方法被隐式指定为 public abstract，即接口中的方法全都是抽象方法。

抽象类是对类（一类事物）的抽象。

被 abstract 关键字修饰的类称为抽象类，抽象类一般包含至少一个抽象方法，抽象方法必须要被子类实现。抽象类可以包含普通实例方法。

3. 什么时候用抽象类 什么时候用接口？

门与警报的例子：

App.java > Alarm

```java
interface Alarm {
    void alarm();
}
abstract class Door {
    abstract void open();
    abstract void close();
}
class AlarmDoor extends Door implements Alarm {
    @Override
    void open() {
        System.out.println("open the door");
    }
    @Override
    void close() {
        System.out.println("close the door");
    }
    public void alarm() {
        System.out.println("alarm!");
    }
}
public class App {
    Run | Debug
    public static void main(String[] args) {
        AlarmDoor alarmDoor = new AlarmDoor();
        alarmDoor.open();
        alarmDoor.close();
        alarmDoor.alarm();
    }
}
```

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL 2:

```
open the door
close the door
alarm!
$
```

如上所示，应该把 alarm() 放在接口中，把 open() 和 close() 放在抽象类 Door 中。这样我们就可以通过接口 Alarm 和抽象类 Door 来定义警报门类（AlarmDoor） 。

4. 内部类和匿名类？

Java 支持在一个类中声明另一个类，这样的类称为内部类；

和某接口有关的匿名类就是实现该接口的一个类，该子类没有明显的用类声明来定义，所以称作匿名类。

5. 异常类

Java 使用 try-catch 来处理异常，我们用 Exception 异常类及其子类，用来表示 java 中可能出现的异常，并处理这些异常。