Lab9

TA:

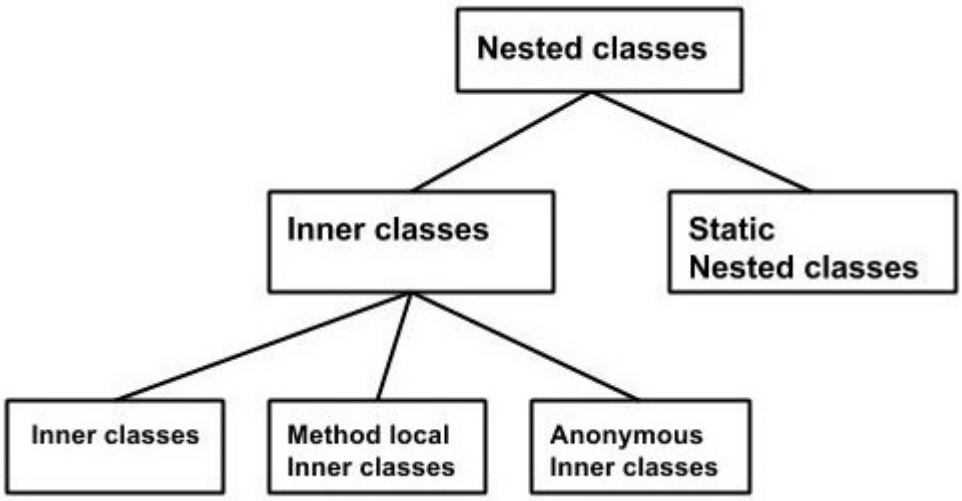
- 14302010005@fudan.edu.cn 姜卓立
- 14302010040@fudan.edu.cn 武多才
- 14302010042@fudan.edu.cn 何培剑

1 摘要

1. 学习并熟练掌握内部类的使用
2. 使用 Intellij 导出java工程制品：可运行jar包
3. 根据自己对面向对象的理解，使用面向对象的方法实现PJ基本功能。

2 内部类

简单的说定义在类内部的类即为内部类。严格一点我们需要区分嵌套类（Nested Class）和 内部类 （Inner Class）。其关系如下图所示，Inner Class 是 Nest Class 的一部分。



从上图可以看出内部类总共有以下三种形式：

1. Inner class, 内部类
2. Method local Inner class，局部内部类
3. Anonymous Inner class，匿名内部类

此外还有一种 Static Nested class （静态嵌套类）的形式。

2.1 内部类和静态嵌套类

查看下面代码，注重三点：

1. seeOuter 方法对外部类属性的可见性。
2. TestHowToUseInnerClass 中对不同内部类的使用方法。

3. 需要总结出static修饰符对内部类修饰的语义和对类变量修饰的语义是一样的。感兴趣可以尝试一下可见性修饰符和final修饰符。

```
/**
 * Top level class definition
 */
class MyOuterClassDemo {
    private static int myStaticVar = 2;
    private int myVar = 1;

    static class MyStaticNestedClassDemo {
        public void seeOuter() {
            // error, non-static variable myVar can not be referenced from a static (Nested)
class
            // System.out.println("Value of myVar is :" + myVar);
            System.out.println("Value of myStaticVar is :" + myStaticVar);
        }
    }

    // inner class definition
    class MyInnerClassDemo {
        public void seeOuter() {
            System.out.println("Value of myVar is :" + myVar);
            System.out.println("Value of myStaticVar is :" + myStaticVar);
        }
    }
}

class TestHowToUseInnerClass {
    TestHowToUseInnerClass() {
        // 1. How to reference a Inner Class
        MyOuterClassDemo outer = new MyOuterClassDemo();
        // use instance of MyOuterClassDemo: outer
        MyOuterClassDemo.MyInnerClassDemo inner = outer.new MyInnerClassDemo();
        inner.seeOuter();

        // 2. How to reference a static nested class
        // use class MyOuterClassDemo directly
        MyOuterClassDemo.MyStaticNestedClassDemo staticNested = new
MyOuterClassDemo.MyStaticNestedClassDemo();
        staticNested.seeOuter();

        // consider what cause this difference
    }

    public static void main(String[] args) {
        new TestHowToUseInnerClass();
    }
}
```

2.2 局部内部类

局部内部类被定义在外部类的方法当中。

1. 如果你想使用内部类，必须同一方法中实例化内部类
2. 只有 abstract 和 final 这两个修饰符被允许修饰局部内部类
3. 只有在**方法的局部变量**被标记为 final 或 局部变量是 effectively final （初始化后没被改过）的， 内部类才能使用它们。consider why???

```
class MyOuterClassDemo {
    private int x = 1;

    void doThings() {
        String name = "local variable"; // name is effectively final
        x += 1;
        int age = 10;
        age += 1; // age is not effectively final
        // inner class defined inside a method of outer class
        class MyInnerClassDemo {
            void seeOuter() {
                // ok, x is not method local variable
                System.out.println("Outer Value of x is :" + x);
                System.out.println("Value of name is :" + name);
                // error: age is not effectively final
                //System.out.println("Value of age is :" + age);
            }
        }
        MyInnerClassDemo inner = new MyInnerClassDemo();
        inner.seeOuter();
    }
}
```

2.3 匿名内部类

匿名内部类涉及到接口，这里为了内部类系统的完整性，先提一下，不要求掌握。匿名内部类有以下特点。

1. 没有名字
2. 只能被实例化一次
3. 通常被声明在方法或代码块的内部，以一个带有分号的花括号结尾
4. 因为没有名字，所以没有构造函数
5. 不能是静态的（static)

看下面这个例子，这个例子很好解释了匿名内部类的语法。

```
abstract class Animal {
    abstract void play();
}

class Person{
    public static void main(String[] args){
        showAnimalPlay(new Animal() {
            @Override
            void play() {
                System.out.println("I play like a dog.");
            }
        });
    }
    private static void showAnimalPlay(Animal a) {
        a.play();
    }
}
```

匿名内部类的最大优点就是简化代码，从上面的例子可以看到我们不需要为Animal写一个狗的实现类，就可以使用相同的功能。

匿名内部类最常用于事件驱动的设计中，PJ2中会经常用到，java8以后只有一个函数的匿名内部类已经可以用Lambda表达式取代了，其形式更加简介优雅，可喜可贺。

3 使用 Intellij 导出java工程制品

本部分内同要求同学们带好自己的电脑，助教于课堂上演示。参考链接
https://www.jetbrains.com/help/idea/java-se.html#package_app

PS：请认真掌握，本次lab需要提交可直接运行制品。

4 作业：迷宫游戏

4.1 要求

本次迷宫需要实现迷宫游戏所有的基本功能，实现时结合自己所学的面向对象的知识以及自己对面向对象的理解，将自己的设计尽量面向对象。

4.2 评分标准

说明	分值
地图设计	20（现在一层迷宫即可）
人物能够移动（wasd）	10
人物碰到怪物死亡	5
怪物移动	6
人物拾取宝物	8
攻击怪物	8
实现足迹	10
实现后退（b）	8
进入下一层迷宫以及判断胜利	5
查看玩家信息	5
查看帮助信息	5
实现沙盒模式	10

5 提交

1. 提交地址：**ftp://10.132.141.33/classes/17/171 程序设计A(戴开宇)/WORK_UPLOAD/lab9/**
2. 提交物：可运行制品，项目源码； 命名格式为**lab9_[学号]**， 例如：**lab9_17302010001.zip**
3. Deadline: 2017年11月26日23:59:59

6 声明

任何形式的作业都欢迎同学们相互讨论，但抄袭是严格禁止的。一旦发现抄袭行为，抄袭者和被抄袭者都以0分处理

7 参考链接

献上我参考的文档的链接，文档内容都很不错，有疑惑的同学可以看一下这些文档。

匿名内部类：

1. https://docs.oracle.com/javase/tutorial/java/javaOO/innerclasses.html
2. http://liuzxc.github.io/blog/java-advance-02/,
3. https://docs.oracle.com/javase/tutorial/java/javaOO/nested.html
4. https://www.tutorialspoint.com/java/java_innerclasses.htm

Intellij Tutorial:

1. https://www.jetbrains.com/help/idea/java-se.html#package_app