**访问控制修饰符**

Java中，可以使用访问控制符来保护对类、变量、方法和构造方法的访问。Java 支持 4 种不同的访问权限。

default (即默认，什么也不写）: 在同一包内可见，不使用任何修饰符。使用对象：类、接口、变量、方法。

private : 在同一类内可见。使用对象：变量、方法。 注意：不能修饰类（外部类）

public : 对所有类可见。使用对象：类、接口、变量、方法

protected : 对同一包内的类和所有子类可见。使用对象：变量、方法。 注意：不能修饰类（外部类）。

我们可以通过以下表来说明访问权限：

修饰符 当前类 同一包内 子孙类(同一包) 子孙类(不同包) 其他包

public Y Y Y Y Y

protected Y Y Y Y/N（说明） N

default Y Y Y N N

private Y N N N N

默认访问修饰符-不使用任何关键字

如果在类、变量、方法或构造函数的定义中没有指定任何访问修饰符，那么它们就默认具有默认访问修饰符。

默认访问修饰符的访问级别是包级别（package-level），即只能被同一包中的其他类访问。

如下例所示，变量和方法的声明可以不使用任何修饰符。

实例

// MyClass.java

class MyClass { // 默认访问修饰符

int x = 10; // 默认访问修饰符

void display() { // 默认访问修饰符

System.out.println("Value of x is: " + x);

}

}

// MyOtherClass.java

class MyOtherClass {

public static void main(String[] args) {

MyClass obj = new MyClass();

obj.display(); // 访问 MyClass 中的默认访问修饰符变量和方法

}

}

以上实例中，MyClass 类和它的成员变量 x 和方法 display() 都使用默认访问修饰符进行了定义。MyOtherClass 类在同一包中，因此可以访问 MyClass 类和它的成员变量和方法。

私有访问修饰符-private

私有访问修饰符是最严格的访问级别，所以被声明为 private 的方法、变量和构造方法只能被所属类访问，并且类和接口不能声明为 private。

声明为私有访问类型的变量只能通过类中公共的 getter 方法被外部类访问。

Private 访问修饰符的使用主要用来隐藏类的实现细节和保护类的数据。

下面的类使用了私有访问修饰符：

public class Logger {

private String format;

public String getFormat() {

return this.format;

}

public void setFormat(String format) {

this.format = format;

}

}

实例中，Logger 类中的 format 变量为私有变量，所以其他类不能直接得到和设置该变量的值。为了使其他类能够操作该变量，定义了两个 public 方法：getFormat() （返回 format的值）和 setFormat(String)（设置 format 的值）

公有访问修饰符-public

被声明为 public 的类、方法、构造方法和接口能够被任何其他类访问。

如果几个相互访问的 public 类分布在不同的包中，则需要导入相应 public 类所在的包。由于类的继承性，类所有的公有方法和变量都能被其子类继承。

以下函数使用了公有访问控制：

public static void main(String[] arguments) {

// ...

}

Java 程序的 main() 方法必须设置成公有的，否则，Java 解释器将不能运行该类。

受保护的访问修饰符-protected

protected 需要从以下两个点来分析说明：

子类与基类在同一包中：被声明为 protected 的变量、方法和构造器能被同一个包中的任何其他类访问；

子类与基类不在同一包中：那么在子类中，子类实例可以访问其从基类继承而来的 protected 方法，而不能访问基类实例的protected方法。

protected 可以修饰数据成员，构造方法，方法成员，不能修饰类（内部类除外）。

子类能访问 protected 修饰符声明的方法和变量，这样就能保护不相关的类使用这些方法和变量。

下面的父类使用了 protected 访问修饰符，子类重写了父类的 openSpeaker() 方法。

class AudioPlayer {

protected boolean openSpeaker(Speaker sp) {

// 实现细节

}

}

class StreamingAudioPlayer extends AudioPlayer {

protected boolean openSpeaker(Speaker sp) {

// 实现细节

}

}

如果把 openSpeaker() 方法声明为 private，那么除了 AudioPlayer 外，其他类将不能访问该方法。

如果把 openSpeaker() 声明为 public，那么所有的类都能够访问该方法。

如果我们只想让该方法对其所在类的子类可见，则将该方法声明为 protected。

访问控制和继承

请注意以下方法继承的规则：

父类中声明为 public 的方法在子类中也必须为 public。

父类中声明为 protected 的方法在子类中要么声明为 protected，要么声明为 public，不能声明为 private。

父类中声明为 private 的方法，不能够被子类继承。