Java 简明教程

屈庆磊

quqinglei@icloud.com

2013 年 *10* 月 *9* 日

目录

1 说明

学习一门语言无怪乎，但又不止于，总之它就是一个工具而已，需要做的就是先宏观上理解、知道结构，然后个个击破剩下的就是漫长的时间不段的积累。

- 语法
- 是面向对象还是面向过程，以及概念
- 如何使用外部的库，或者称之为包
- 如何建立自定义库
- 操作系统相关的一些问题：如线程、锁等
- 框架，比如 Java 应该熟悉一种编程框架
- 剩下的都是细节了，学一门语言很简单，而熟悉的使用还是一段比较漫长的路

2 修饰符

- public 的类、类属变量和方法、包内和包外的任何类都可以访问
- protected 的类、类属变量和方法，包内的任何类，以及包外的那些继承了此类的子类才能访问
- private 的类属变量和方法，包内包外的任何类不能访问
- 不以以上三种修饰符修饰的类的方法、类属变量都属于 friendly 类型，包内的任何类都可以访问它，包外的任何类都不能访问（包括包外继承此类的子类）

类

访问修饰符名称	说明	备注
public	可以被所有类访问	public 类必须定义在类和类名相同的文件中
package	可以被同一个包中的类访问	默认的访问权限，可省略此关键字，可以定义在和 public 类的同一文件中

访问修饰符名称	说明
final	使用此修饰符的类不能被继承
abstract	如果要使用 abstract 类，必须首先建立一个继承 abstract 类的新类，新类中实现 abstract 类中的抽象方

3 包

和其他语言一样，在 *Java* 中也有库的概念，在编写程序的时候也需要调用库，我们也可以自定义库，至于如何建立库，如何使用我们可以从例子中得到，再好的描述不如一个简单的例子。

假如我们的当前路径为:/home/java 写一个简单的库文件

```
> mkdir classes # 建立一个路径，自定义的，仅仅作为例子
> cd classes
> mkdir -p com/quqinglei/tools
> cd com/quqinglei/tools
> touch Name.java
> vi Name.java # 内容如下
```

```
// Name.java
package com.quqinglei.tools;
public class Name {
        public Name() {
                System.out.println("quqinglei");
        }
}
```

编译库文件，并打包成 jar 格式的文件，不打包也可以使用

```
// 编译并打包成 jar 文件
> javac Name.java
> cd ~/classes
> jar -cvf lei.jar com
```

写一个小程序调用刚才写的库

```
> cd ~
touch Test.java
```

```
// Test.java
import com.quqinglei.tools.*;
public class Test {
        public static void  main(String[] args) {
                Name n = new Name();
        }
}
```

下面告诉如何编译使用刚才定义的库的小程序，并讲述如何执行

```
# 一下几种方式都可以，如果打成 jar 包，则可以使用 jar 包，否则也可以直接
# 使用路径，可以把 jar 中的路径结构当做路径去理解
# 1
> javac -classpath /home/java/classes/lei.jar:./ Test.java
> java -classpath /home/java/classes/lei.jar:./ Test
quqinglei

# 2
> export CLASSPATH="/home/java/classes/lei.jar:./"
> javac Test.java
> java Test
quqinglei

# 3
> export CLASSPATH="/home/java/classes:./"
> javac Test.java
> java Test
quqinglei

# 4
> javac -classpath /home/java/classes:./ Test.java
```

```
> java -classpath /home/java/classes:./ Test
quqinglei
```

注意：

- 库中的类如果被外部引用，构造函数必须是 `public` 声明的
- 同上，成员函数如果需要被外面的包引用也必须是 `public` 声明的

4 类

4.1 其他

- 子类继承父类的构造函数，并且可以拥有自己的构造函数，两者是叠加，没有冲突
- 子类如果重写了某方法，父类的此方法会被废弃，称之为重写，而非叠加

4.2 初始化和清除

其实本小节讲的就是构造函数，构造函数提供初始化和清除的工作。

- 构造函数可以被重载，非构造函数也是可以重载的
- 数据类型能从一个较为小的类型自动转变为一个较为大的类型，且靠近较大，而不是最大
- 数据类型不能从一个大的匹配到构造函数中的小的类型，如果构造函数没有对比此数据类型大的重载构造函数，则会出现错误，所以尽量不要做一些不匹配的劳什子，精确一点比较好，不要依靠编译器
- 默认构造函数是没有参数的

4.3 抽象类

包含抽象方法的类就叫做抽象类，抽象类是一个模板，或者说是接口，只提供抽象，不提供实现方式，抽象类的子类必须实现抽象类包含的抽象方法。

- 抽象类中不一定包含抽象方法，但包含抽象方法的类一定要被声明成抽象类 `abstract class ClassName`
- 抽象类不能用 `final` 修饰，即一个类不能既是最终类又是抽象类
- `abstract` 不能与 `private/static/final/native` 并列修饰同一个方法
- 抽象类的抽象方法不能拥有代码，它只能是一个声明，如 `abstract foo();`
- 抽象类的子类必须定义抽象类的抽象方法

1
