Java 允许使用**包**将类组织在一个集合中，借助包可以方便地组织自己的代码，并将自己的代码与别人提供的代码库分开管理。

1 包名

使用包的主要原因是确保类名的唯一性。如果不同的程序员建立了同名的类，只要将这些类放在不同的包中，就不会产生冲突。为了保证包名的绝对唯一性，要用一个因特网域名以逆序的形式作为包名。例如域名`horstmann.com`，以逆序来写就得到了包名`com.horstmann`，然后可以追加一个工程名，如`com.horstmann.corejava`。如果再把`Employee`类放在这个包里，那么这个类的完全限定名就是`com.horstmann.corejava.Employee`。

包之间可以互相嵌套，即可以将一个包放在另一个包里。从编译器的角度来看，嵌套的包之间没有任何关系，每一个包都是一个独立的类的集合。

2 类的导入

一个类可以使用所属包中的所有类，以及其他包中的公共类。

访问其他包中的公共类有两种方式。第一种方式是使用完全限定名，也就是包名后面跟着类名，这种方式很繁琐。第二种方式是使用`import`语句。

`import`语句是一种引用包中各个类的简洁方式，可以在使用类时不必写出类的全名。`import`语句应该位于源文件的顶部，但位于`package`语句的后面。

可以使用`import`语句导入一个特定的类或者整个包。例如，可以使用下面这条语句导入`java.time`包中的所有类：

```
import java.time.*;
```

还可以导入一个包中的特定类：

```
import java.time.LocateDate;
```

只能用星号 `*` 导入一个包，而不能使用 `import java.*` 或 `import java.*.*` 导入以 java 为前缀的所有包。

如果导入的若干个包中存在重名的类，要使用这样的类时就要在类名的前面加上完整的包名。

3 静态导入

静态导入允许导入静态方法和静态字段。例如：

```
import static java.lang.System.*; // 导入 System 类的静态字段和静态方法

// 直接使用 System 类的静态字段和静态方法，而不必加类名前缀
out.println("Goodbye, World!"); // 静态字段 System.out
exit(0); // 静态方法 System.exit()
```

还可以导入特定的方法或字段。例如：

```
import static java.lang.System.out;
```

4 在包中增加类

要想将类放入包中，就必须将包名放在源文件的开头。例如：

```
package com.horstmann.corejava;

public class Employee
{
    ...
}
```

如果没有在源文件中放置 `package` 语句，这个源文件中的类就属于无名包。

源文件要放到与完整包名匹配的子目录中。例如，com.horstmann.corejava 包中的所有源文件应该放置在子目录 com/horstmann/corejava 中（Windows 中则是 com\horstmann\corejava）。编译器将类文件也放在相同的目录结构中。

5 包访问

对于类的实例字段和方法，标记为`public`的部分可以由任意类使用，标记为`private`的部分只能由定义它们的类使用。如果没有指定`public`或`private`，这个部分（类、方法或变量）可以被同一个包中的所有方法访问。

对于类来说，不指定访问权限将默认在同一个包中的方法可以访问。但是对于变量，如果不指定访问权限，将被包中所有方法访问，破坏了封装性。因此变量必须显式地标记为`private`。