Chapter 8 Packages & Inheritance

Based on the course literature:

Java: A beginner's guide

Fifth Edition

Herbert Schildt

What we'll cover

- Use packages
- Understand how packages affect access
- Apply protected access modifier
- Import packages
- Know Java's standard packages
- Understand interface fundamentals
- Implement an interface
- Apply interface references
- Extend interfaces

Packages and interfaces

- Packages and interfaces help to organise our code.

Packages

- Packages are namespaces
- Normally a package will contain a number of related classes.
- Classes in packages have their own access settings.
- Every class belongs to a package
- Default has no name.

Packages

- packages pkg;
- packages pkg1.pkg2.pkg3;

- packages are mirrored by directories
- pkg1/pkg2/pkg3

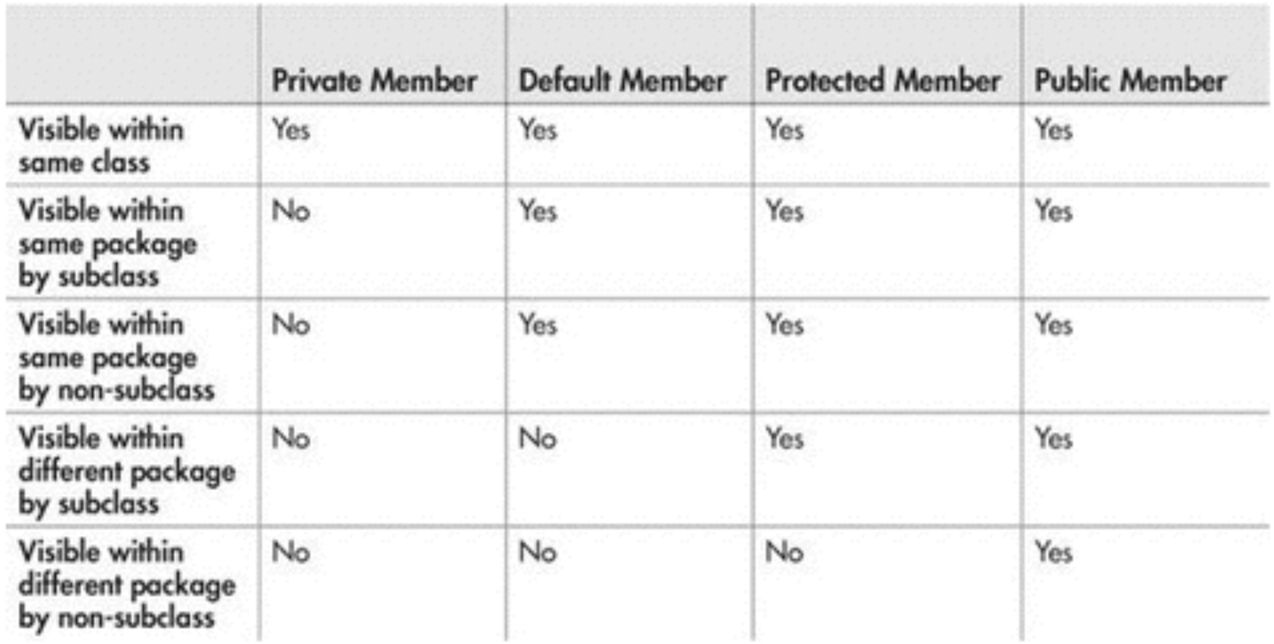
Finding packages

- Current working directory
- –classpath option with java or javac command
- CLASSPATH environmental variable

Demo 1

ARCTIC TIGER

Access modifiers

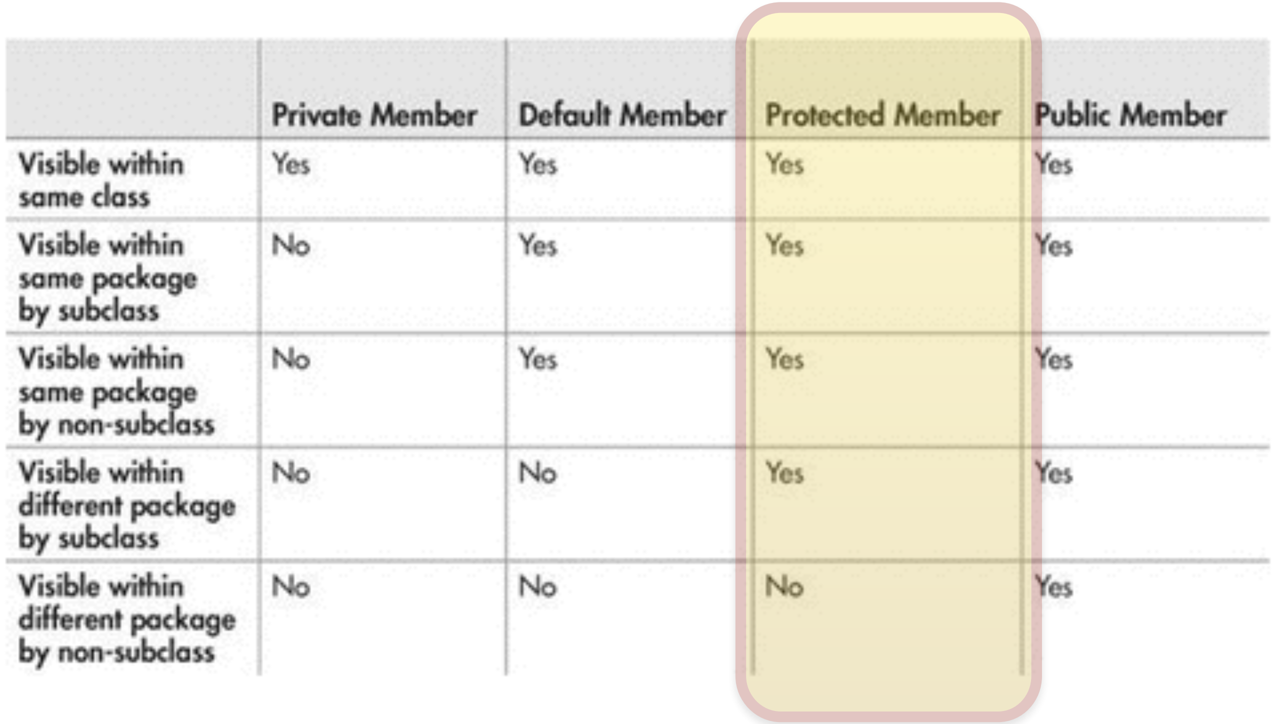


	Private Member	Default Member	Protected Member	Public Member
Visible within same class	Yes	Yes	Yes	Yes
Visible within same package by subclass	No	Yes	Yes	Yes
Visible within same package by non-subclass	No	Yes	Yes	Yes
Visible within different package by subclass	No	No	Yes	Yes
Visible within different package by non-subclass	No	No	No	Yes

Table 8-1 Class Member Access

Demo 2

protected



	Private Member	Default Member	Protected Member	Public Member
Visible within same class	Yes	Yes	Yes	Yes
Visible within same package by subclass	No	Yes	Yes	Yes
Visible within same package by non-subclass	No	Yes	Yes	Yes
Visible within different package by subclass	No	No	Yes	Yes
Visible within different package by non-subclass	No	No	No	Yes

Table 8-1 Class Member Access

ARCTICTIGER

Working with packages

- When in another package:
  - Reference the namespace and the class:

    ```
    bookpack.Book book = new bookrack.Book();
    ```

  - Add an import above the class;

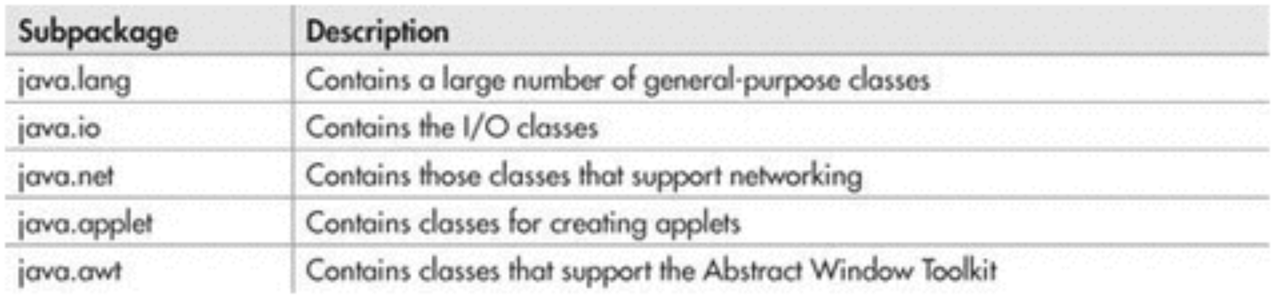
    ```
    import bookpack.Book;
    ```

  - Add an import with a wildcard

    ```
    import bookpack.*;
    ```

Demo 3

Some key packages



Subpackage	Description
java.lang	Contains a large number of general-purpose classes
java.io	Contains the I/O classes
java.net	Contains those classes that support networking
java.applet	Contains classes for creating applets
java.awt	Contains classes that support the Abstract Window Toolkit

Interfaces

- Interfaces can be used as a contract for what public or default methods a class must implement.
- They are similar to abstract classes, but more extreme.
- They are extremely useful when you want code to be dynamic.
- Interface methods contain no body.

```
access interface name {
        type var1 = value;
        return-type method-name(param-list);
}
```

Interface basics

- no method can have a body
- variables declared in an interface are constants, public final static.
- multiple classes can implement an interface.
- a class can implement multiple interfaces.

Implementing an interface

```
class classname extends superclass implements interface1, interface2 {
        // class-body
}
```

Implementing an interface

- A class must implement all the methods specified in the interface.
- The methods should all be implemented as public.
- The methods should have exactly the same signature and return values as the interface version.

ARCTICTIGER

Demo 4

Working with an objects interface

```
SomeInterface obj  = new Car();
obj.interfaceMethod();
```