Packages In Java

Demonstrated By,
Prashant Srivastava
Asst. Prof.
Department of Computer Science & Engg.
Shambhunath Group Of Institutions-Allahabad

Introduction

- The main feature of OOP is its ability to support the reuse of code:

- Extending the classes (via inheritance)
- Extending interfaces

- The features in basic form limited to reusing the classes within a program.
- What if we need to use classes from other programs without physically copying them into the program under development ?
- In Java, this is achieved by using what is known as “packages”, a concept similar to “class libraries” in other languages.

Packages

- Packages are Java's way of grouping a number of related classes and/or interfaces together into a single unit. That means, packages act as "containers" for classes.

- The benefits of organising classes into packages are:

- The classes contained in the packages of other programs/applications can be reused.
- In packages classes can be unique compared with classes in other packages. That two classes in two different packages can have the same name. If there is a naming clash, then classes can be accessed with their fully qualified name.
- Classes in packages can be hidden if we don't want other packages to access them.
- Packages also provide a way for separating "design" from coding.

Types of package:

1) User defined package: The package we create is called user-defined package.

2) Built-in package: The already defined package like java.io.*, java.lang.* etc are known as built-in packages.

Java Foundation Packages

Java provides a large number of classes groped into different packages based on their functionality.

The six foundation Java packages are:

- java.lang
  - *Contains classes for primitive types, strings, math functions, threads, and exception*
- java.util
  - *Contains classes such as vectors, hash tables, date etc.*
- java.io
  - *Stream classes for I/O*
- java.awt
  - *Classes for implementing GUI – windows, buttons, menus etc.*
- java.net
  - *Classes for networking*
- java.applet
  - *Classes for creating and implementing applets*

Accessing Classes from Packages

There are two ways of accessing the classes stored in packages:

- Using fully qualified class name
  - *java.lang.Math.sqrt(x);*
- Import package and use class name directly.
  - *import java.lang.Math*
  - *Math.sqrt(x);*

Selected or all classes in packages can be imported:

```
import package.class;
import package.*;
```

Implicit in all programs: import java.lang.*;

package statement(s) must appear first

Creating Packages

Java supports a keyword called “package” for creating user-defined packages. The package statement must be the first statement in a Java source file (except comments and white spaces) followed by one or more classes.

```
package myPackage;
public class ClassA {
        // class body
}
class ClassB {
 // class body
}
```

```
package mypack;
public class Simple{
 public static void main(String args[]){
    System.out.println("Welcome to package");
   }
}
```

How to compile & Run java package

javac -d directory javafilename

javac -d . Simple.java

To Run: java mypack.Simple

Sub Packages

We can also put a package inside an another package. The packages that comes lower in the naming hierarchy are called "subpackage" of the corresponding package higher in the hierarchy i.e. the package that we are putting into another package is called "sub package".

Example of Sub Package

```
package importpackage.subpackage;

public class HelloWorld {
  public void show(){
  System.out.println("This is the function of the class
HelloWorld!!");
  }
}
```

```
import importpackage.subpackage.*;
class CallPackage{
  public static void main(String[] args){
  HelloWorld h2=new HelloWorld();
  h2.show();
  }
}
```

Excercise

```
package Geometry;

public class Square {
    public double side;

    public double calculateArea()
    {
            return side * side;
    }
}
```

```
import Geometry;
public class Exercise {
    public static void main(String[] args) {
            Square sqr = new  Square();
            sqr.side = 32.40;
          System.out.println("\nSquare
Characteristics");
        System.out.printf("Side:   ", sqr.side);
System.out.printf(“Area:   ", sqr. calculateArea());

    }
}
```