**Packages In Java**

Package in Java is a mechanism to encapsulate a group of classes, sub packages and interfaces. Packages are used for:

Preventing naming conflicts. For example there can be two classes with name Employee in two packages, college.staff.cse.Employee and college.staff.ee.Employee.

Making searching/locating and usage of classes, interfaces, enumerations and annotations easier

Packages can be considered as data encapsulation (or data-hiding).

All we need to do is put related classes into packages. After that, we can simply write an import class from existing packages and use it in our program. A package is a container of a group of related classes where some of the classes are accessible are exposed and others are kept for internal purpose.

We can reuse existing classes from the packages as many time as we need it in our program.

**How packages work?**

Package names and directory structure are closely related. For example if a package name is college.staff.cse, then there are three directories, college, staff and cse such that cse is present in staff and staff is present college. Also, the directory college is accessible through CLASSPATH variable, i.e., path of parent directory of college is present in CLASSPATH. The idea is to make sure that classes are easy to locate.

Package naming conventions : Packages are named in reverse order of domain names, i.e., org.geeksforgeeks.practice. For example, in a college, the recommended convention is college.tech.cse, college.tech.ee, college.art.history, etc.

**Adding a class to a Package** : We can add more classes to a created package by using package name at the top of the program and saving it in the package directory. We need a new java file to define a public class, otherwise we can add the new class to an existing .java file and recompile it.

**Subpackages:** Packages that are inside another package are the subpackages. These are not imported by default, they have to imported explicitly. Also, members of a subpackage have no access privileges, i.e., they are considered as different package for protected and default access specifiers.

**Types of packages:**

packages

**Built-in Packages**

These packages consist of a large number of classes which are a part of Java API.Some of the commonly used built-in packages are:

1) java.lang: Contains language support classes(e.g classed which defines primitive data types, math operations). This package is automatically imported.

2) java.io: Contains classed for supporting input / output operations.

3) java.util: Contains utility classes which implement data structures like Linked List, Dictionary and support ; for Date / Time operations.

4) java.applet: Contains classes for creating Applets.

5) java.awt: Contain classes for implementing the components for graphical user interfaces (like button , ;menus etc).

6) java.net: Contain classes for supporting networking operations.

**User-defined packages**

These are the packages that are defined by the user. First we create a directory myPackage (name should be same as the name of the package). Then create the MyClass inside the directory with the first statement being the package names.

// Name of the package must be same as the directory

// under which this file is saved

package myPackage;

public class MyClass

{

public void getNames(String s)

{

System.out.println(s);

}

}

Now we can use the MyClass class in our program.

/\* import 'MyClass' class from 'names' myPackage \*/

import myPackage.MyClass;

public class PrintName

{

public static void main(String args[])

{

// Initializing the String variable

// with a value

String name = "KING";

// Creating an instance of class MyClass in

// the package.

MyClass obj = new MyClass();

obj.getNames(name);

}

}

Note : MyClass.java must be saved inside the myPackage directory since it is a part of the package.

**Using Static Import**

Static import is a feature introduced in Java programming language ( versions 5 and above ) that allows members ( fields and methods ) defined in a class as public static to be used in Java code without specifying the class in which the field is defined.

// Note static keyword after import.

import static java.lang.System.\*;

class StaticImportDemo

{

public static void main(String args[])

{

// We don't need to use 'System.out'

// as imported using static.

out.println("GeeksforGeeks");

}

}

Output:

GeeksforGeeks