Packages in Java

What are Pacakges in Java ?

- It is a mechanism to encapsulate a group of classes, sub packages and interfaces.

Use of Packages :

- Preventing naming conflicts.
  - *Example:* there can be two classes with name Employee in two packages, college.staff.cse.Employee and college.staff.ee.Employee.
- Makes searching/locating and usage of classes, interfaces, enumerations and annotations easier.
- **Provides controlled access :**
  - protected and default have package level access control.
  - A protected member is accessible by classes in the same package and its subclasses.
  - A default member (without any access specifier) is accessible by classes in the same package only.
- Packages can be considered as ***data encapsulation*** (or data-hiding).

How to create a package ?

- 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.
- It 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.
- They are considered as different package for protected and default access specifiers.

```
import java.util.*;
// util is a subpackage created inside java package.
```

Accessing classes inside a package

```
// import the Vector class from util package.
import java.util.vector;

// import all the classes from util package
import java.util.*;
```

- First Statement is used to import **Vector** class from **util** package which is contained inside **java**.
- Second statement imports all the classes from **util** package.

```
// All the classes and interfaces of this package will be accessible but not subpackages.
import package.*;

// Only mentioned class of this package will be accessible.
import package.classname;

// Class name is generally used when two packages have the same class name.
// For example in below code both packages have date class.
// So we need to use a fully qualified name to avoid conflict
import java.util.Date;
import my.packag.Date;
```

Types of Packages

1. 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:
  i. **java.lang:** Contains language support classes(e.g classed which defines primitive data types, math operations). This package is automatically imported.
  ii. **java.io:** Contains classed for supporting input / output operations.
  iii. **java.util:** Contains utility classes which implement data structures like Linked List, Dictionary and support ; for Date / Time operations.
  iv. **java.applet:** Contains classes for creating Applets.
  v. **java.awt:** Contain classes for implementing the components for graphical user interfaces (like button , menus etc).
  vi. **java.net:** Contain classes for supporting networking operations.

2. 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 myPackage**.

Using Static Import

- Static import is a feature introduced in **Java** programming language 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
```