Packages

A package is a directory that contains related classes, interfaces & sub-directories.

Packages are used to avoid naming collisions.

With packages, we can have same name to files in different packages.

Types of packages :-

There are two types of packages : 1) Pre-defined packages (or) built-in packages

2) User-defined packages.

Built-in packages :-

These packages are already available in java language.

They provide necessary classes, interfaces & methods, for developing the programs to provide solutions to the real-time problems.

The following are some of important built-in packages.

i) java.lang - is a language package which contains wrapper classes, used for converting object type into primitive type & vice versa

ii) java.util - utility package. This package contains the classes like stack, linked list, hashtable, arrays, list. (All these are called "collection frameworks")

iii) java.io – this package contains all the classes & interfaces to perform input & output operations
(like stream classes)

iv) java.net – networking package.
This package is used to develop client server programming
This package has classes & interfaces to provide support for net package has classes & interfaces to provide support for socket programming

v) java.sql – This package is used to connect database with the program.

vi) java.awt – abstract window Toolkit
This ~~too~~ package contains classes & interface to develop GUI programs.
java.awt.event is a subpackage of java.awt

vii) javax.swing – is a extension package of awt used for developing GUI programs.

viii) java.applet – This package is used to develop the applet program which comes from server to client & gets executed at client side.

ix) java.txt – This package contains important classes for formatting date, time and numbers.

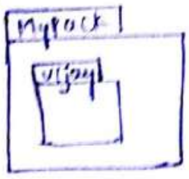
Creating user-defined packages :-

→ To create a package use the following statement as first line in java source file.

$$\boxed{\text{package packagename1.packagename2.......;}}$$

Eg :- package MyPack;

package MyPack.Vijay ; // vijay is a subpackage of MyPack



```
package p1;

public class B{
    public static void main(String args[]){
        System.out.println("Inside package p1");
        }
}
```

mpile :-
```
javac -d . B.java
```

compiler will create directory/folder with the name MyPack specified in program with package statement, in current directory where java source file is available

Run :-
```
d>java p1.B
```

```
Inside package p1
```

Write a java program to create a package CSEC & store addition class in it

```
package CSEC;
public class Addition
{
    private double a;
    private double b;
public Addition ( double x, double y)
    {
        a = x;
        b = y;
    }
public void Add ()
    {
        system.out.println ("~~Addition~~ sum of two numbers is:" + (a+b));
    }
}
```

Importing packages

The existing packages can be used or accessed using 2 diff. ways:

1) using import statement
2) By using fully qualified name

To import packages, the keyword 'import' is used as follows

① import packagename.subpackagename;

② import packagename.classname;

③ import packagename.*;

Eg :- i) import java.util.*;

ii) import java.awt.event.*;

iii) import java.util.scanner;

iv) import java.awt.*;

NOTE :-
'*' imports only classes & interfaces, but not sub-packages

To import packages, use fully qualified name as follows

i) class A extends java.util.Scanner
```
{
    ===
}
```

ii) class A
```
{
        java.util.Scanner obj = new java.util.Scanner(System.in);
}
```

Write a java program which depicts how to use addition class of package CSEC

```
class ADD
{
    public static void main(string args[])
    {
        CSEC.Addition obj = new CSEC.Addition(12.00, 13.00);
        obj.sum();
    }
}
```

compile :- javac ADD.java

Run :- java ADD

o/p :- sum of two numbers is ; 25.0

Write a java program which uses the import statement to import CSEC package & its classes into a program

```
import ~~#~~ CSEC.Addition;
class Adduse
{
    public static void main(string args[])
    {
        Addition obj = new Addition(12.00, 13.00);
    }   obj.sum();
}
```

Write a program to add another class ~~&~~ subtraction to the same package csec.

```
package csec;
public class Subtraction
{
    public double sub (double d1, double d2)
    {
        return (d1-d2);
    }
}
```

save :- Subtraction.java

compile :- javac -d. Subtraction.java // csec package is not created again as it is already existing

---

classpath environment variables & finding packages

The java run-time system knows about the packages in 3 ways.

By default the java run-time system uses the current working directory as its starting point.

A directory path(s) (i.e package path) can be specified by setting the CLASSPATH environment variable

By using hyphenclasspath option with java & javac to specify the path to your classes.

CLASSPATH is an environment variable that tells the java compiler where to look for class files to import.

<u>Access Protection</u>

	private	default	protected	public
same class	**yes**	**yes**	**yes**	**yes**
same package subclass	**no**	**yes**	**yes**	**yes**
same package non-sub-class	**no**	**yes**	**yes**	**yes**
different package sub-class	**no**	**no**	**yes**	**yes**
different package non-sub-class	**no**	**no**	**no**	**yes**

- Any member declared public can be accessed from anywhere.
- Any member declared private cannot be seen outside its class.
- When a member does not have any access specification (default access), it is visible to all classes within the same package.
- To make a member visible outside the current package, but only to subclasses of the current class, declare this member protected.