Package in Java

What is Package?

A package is nothing but a physical folder structure (directories) that contains a group of related classes, interfaces, and sub-packages according to their functionality.

It provides a convenient way to organize your work. The Java language has various in-built packages.

Example: java.lang, java.util, java.io, and java.net.

All these packages are defined as a very clear and systematic packaging mechanism for categorizing and managing.

Advantage of using packages

1. **Maintenance:** Java packages are used for proper maintenance. If any developer newly joined a company, he can easily reach to files needed.

2. **Reusability:** We can place the common code in a common folder so that everybody can check that folder and use it whenever needed.

3. **Name conflict:** Packages help to resolve the naming conflict between the two classes with the same name. Assume that there are two classes with the same name Student.java.

4. Each class will be stored in its own packages such as stdPack1 and stdPack2 without having any **conflict** of names.

5. **Data Encapsulation:** They provide a way to hide classes, preventing other programs from accessing classes that are meant for internal use only

6. **Organized:** It also helps in organizing the files within our project.

7. **Access Protection:** A package provides access protection. It can be used to provide visibility control. The members of the class can be defined in such a manner that they will be visible only to elements of that package.

Types of Packages:

There are two types package in java as

1) Predefined Package
2) Userdefined Package

Predefined Package:

Predefined packages in java are those which are developed by Sun Microsystem / Oracle and supplied as a part of JDK to simplify the task of java programmer. They are also called built-in packages in java.

These packages consist of a large number of predefined classes, interfaces, and methods that are used by the programmer to perform any task in his programs.

Some of the commonly used built-in packages are java.lang, java.io, java.util, java.applet, etc.

Java APIs contains the following predefined packages as

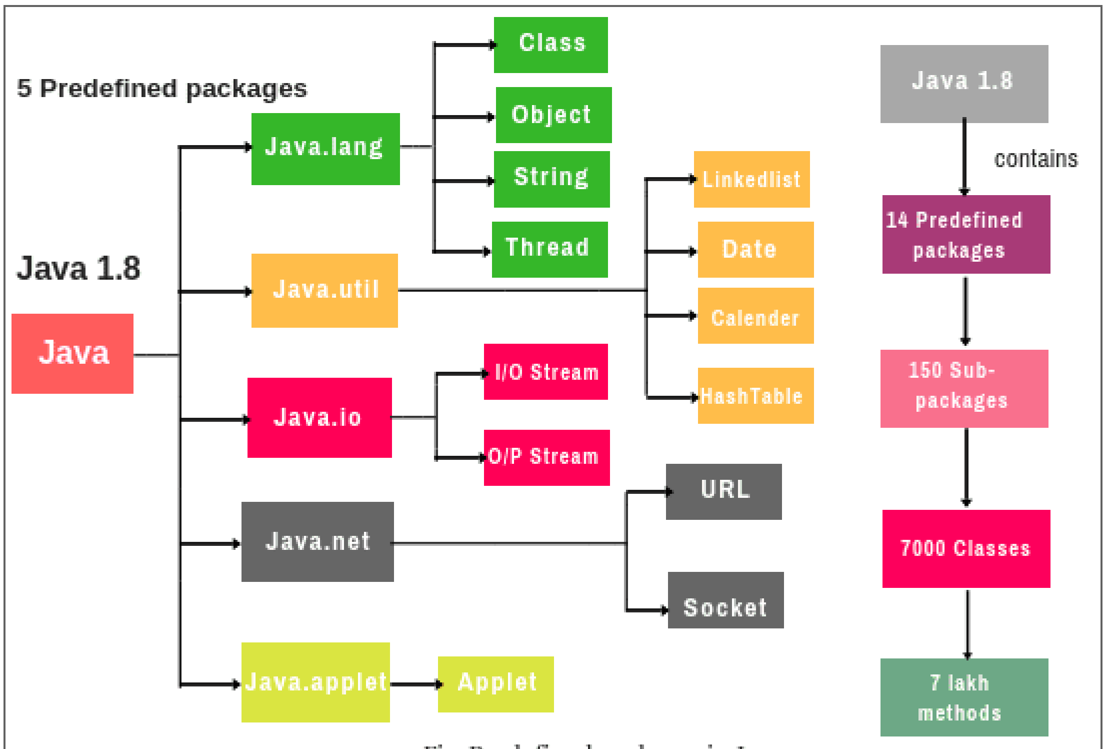


Fig: Predefined packages in Java

Core packages:

1) **Java.lang:** lang stands for language. The Java language package consists of java classes and interfaces that form the core of the Java language and the JVM. It is a fundamental package that is useful for writing and executing all Java programs.
   Examples are classes, objects, String, Thread, predefined data types, etc. It is imported automatically into the Java programs.

2) **Java.io:** io stands for input and output. It provides a set of I/O streams that are used to read and write data to files. A stream represents a flow of data from one place to another place.

3) **Java util:** util stands for utility. It contains a collection of useful utility classes and related interfaces that implement data structures like LinkedList, Dictionary, HashTable, stack, vector, Calender, data utility, etc.

4) **Java.net:** net stands for network. It contains networking classes and interfaces for networking operations. The programming related to client-server can be done by using this package.

Note:

1) Java contains 14 predefined packages which are main packages. These 14 predefined packages contain nearly 150 sub-packages that consist of a minimum of 7 thousand classes. These 7 thousand classes contain approx 7 lakhs methods.
2) Up to Java 1.7 version contains 13 predefined packages. From Java 1.8 version onwards, one new package is introduced called java.time.
3) java.lang package is automatically imported in every Java class.
4) You can import all classes and sub-packages from a package by using the wildcard * e.g. import com.abc.* will import all classes on package com.abc as well as classes on any subpackage.

Example 1:

```
import java.util.Scanner; // java.util is predefined package and we can use scanner class
public class Predefined_Package_Example {
        public static void main(String[] args) {
                float r;
                Scanner sc = new Scanner(System.in);
                System.out.println("Enter Radius:");
                r = sc.nextFloat();

                System.out.println("Area of Circle:"+Math.PI*r*r);

                sc.close();
        }
}
```

Userdefined Package:

The package which is defined by the user is called a User-defined package. It contains user-defined classes and interfaces.

Creating package in Java

Java supports a keyword called “package” which is used to create user-defined packages in java programming.

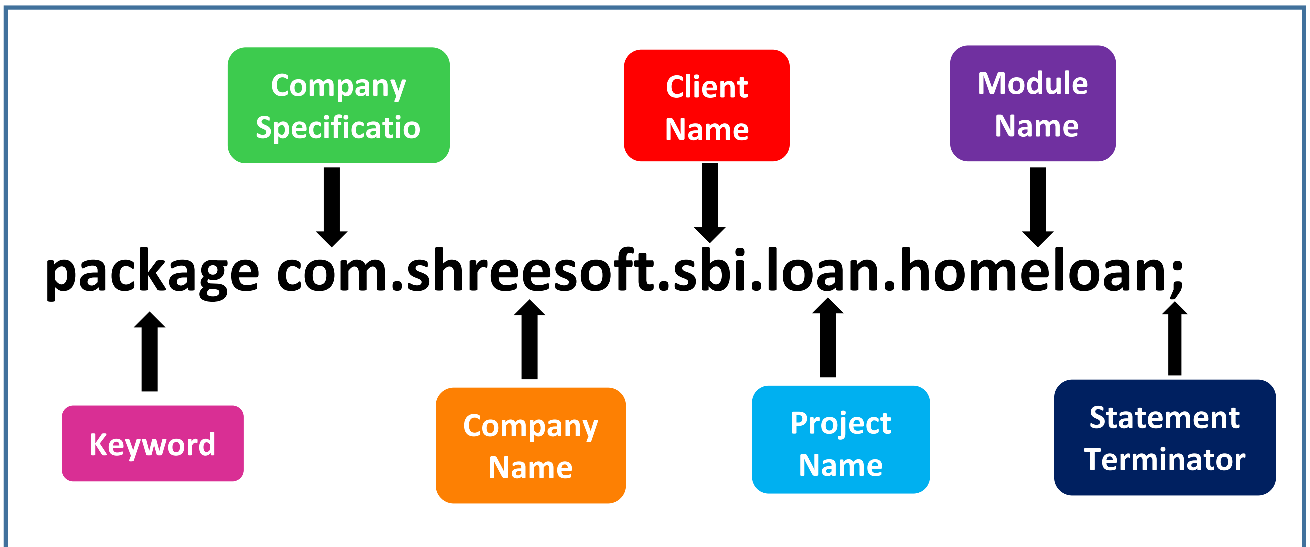
Syntax:

```
package packageName1[.pkg2[.pkg3]];
```

Example:

```
package myPackage;
public class A {
    // class body
}
```

Naming Convention to declare User-defined Package in Real-time Project



Complete Package Structure of Project

Note: while declaring the package name, every character should be lowercase.

1. Suppose you are working in ShreeSoft and the domain name of shreesoft is www.shreesoft.com. You can declare the package by reversing the domain like this:

   package com.shreesoft;

   where,
   com → It is generally company specification name and the folder starts with com which is called root folder.
   shreesoft → Company name where the product is developed. It is the subfolder.

2. **sbi** → Client name for which we are developing our product or working for the project.

3. **loan** → Name of the project.

4. **homeloan** → It is the name of the modules of the loan project. There are a number of modules in the loan project like a Home loan, Car loan, or Personal loan. Suppose you are working for Home loan module.

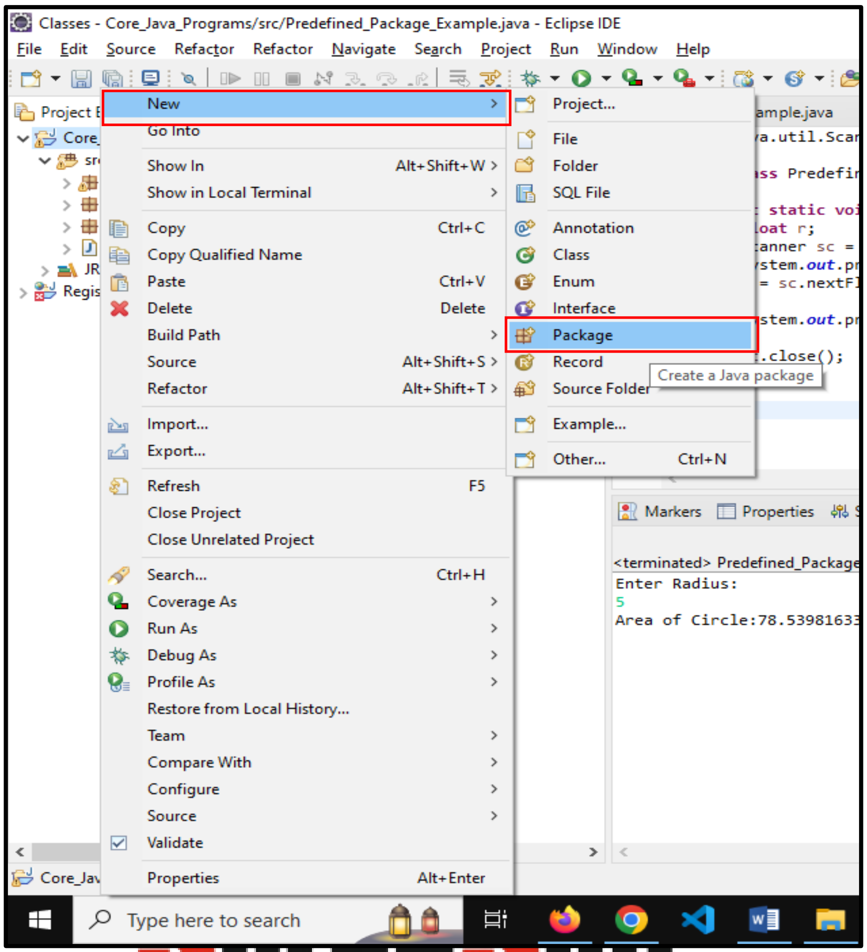
Note: Keep in mind Root folder should be always the same for all the classes.

How to create package using Eclipse IDE?

Follow below steps to create package using IDE as

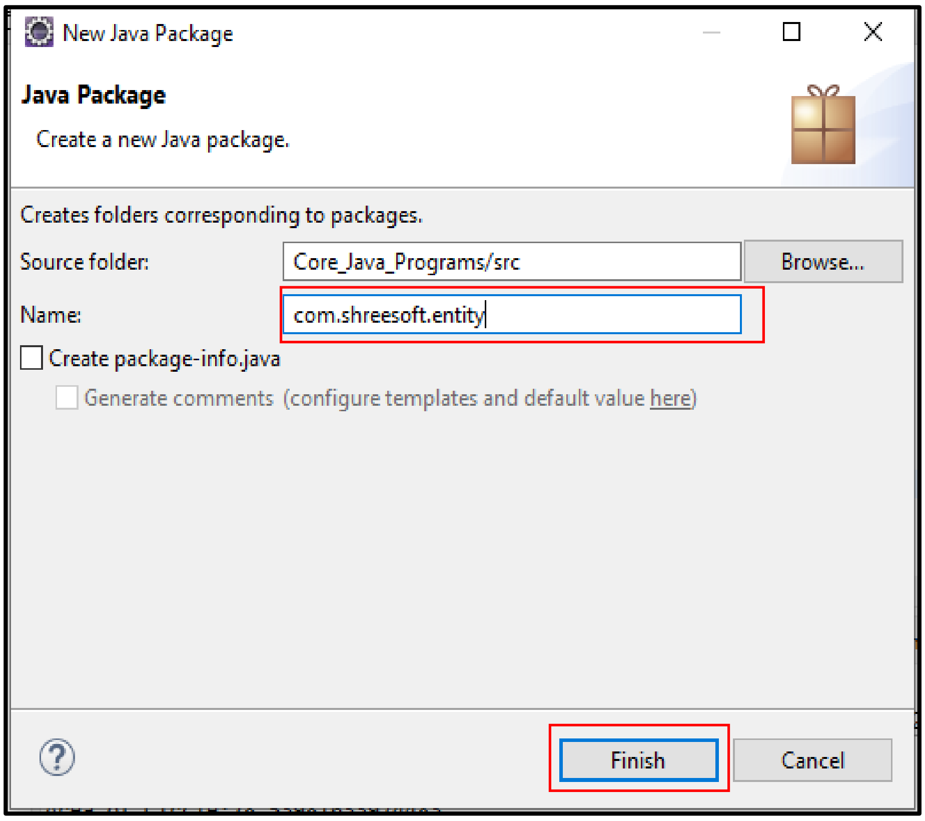
Step 1: To create project, if project is not available

Step 2: Select File→ New→ Package, or right-click the Package Explorer in the Java perspective and select New→ Package. Alternatively, you can specify the package when you create a new class.



```
<terminated> Predefined_Package
Enter Radius:
5
Area of Circle:78.53981633
```

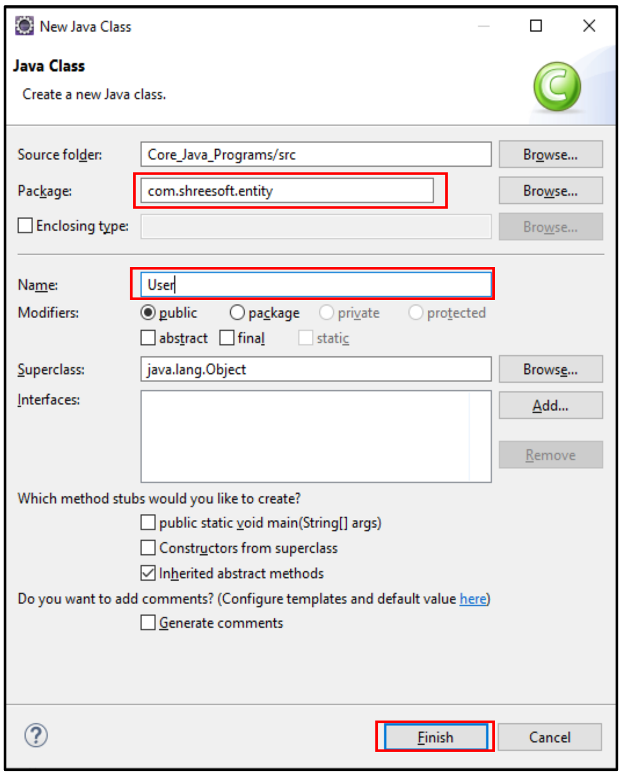
Step 3: Click on Package and fill the package name.


New Java Package
Java Package
Create a new Java package.
Creates folders corresponding to packages.
Browse...
Create package-info.java
Generate comments (configure templates and default value here)
Finish
Cancel

Source folder: Core_Java_Programs/src

Name: com.shreesoft.entity

Step 4: When the new package appears in the project, you can add classes to it by right-clicking it and selecting New→ Class.


New Java Class

Java Class

Create a new Java class.

Source folder: Core_Java_Programs/src

Package: com.shreesoft.entity

☐ Enclosing type:

Name: User

Modifiers: ● public ○ package ○ private ○ protected

☐ abstract ☐ final ☐ static

Superclass: java.lang.Object

Interfaces:

Which method stubs would you like to create?

☐ public static void main(String[] args)

☐ Constructors from superclass

☑ Inherited abstract methods

Do you want to add comments? (Configure templates and default value here)

☐ Generate comments

Example:

```
package com.shreesoft.demo; // This is user defined package
import java.util.Scanner;

//This is class which we can add in userdefined package
public class Userdefined_Package_Example1 {
	public static void main(String[] args) {
		int x, y;
		Scanner scan = new Scanner(System.in);
		System.out.println("Enter 2 Number:");
		x = scan.nextInt();
		y = scan.nextInt();

		System.out.println("Addition:"+(x+y));
		System.out.println("Subtraction:"+(x-y));
		System.out.println("Multiplication:"+(x*y));
		System.out.println("Division:"+((float)x/y));

		scan.close();
	}
}
```

How to compile package in Java?

If you are not using any Eclipse IDE, you follow the syntax given below:

Syntax:

To Compile the application: `javac -d directory javafilename`

1. Here, javac means java compiler.

2. -d means directory. It creates the folder structure.

3. .(dot) means the current directory. It places the folder structure in the current working directory.

For example: javac –d . Userdefined_Package_Example.java

Here, Userdefined_Package_Example.java is the file name. So in this way, you must compile application if the application contains a package statement.

After the compilation, you can see the folder structure in your system like this:

```
com
  |—> shreesoft
     |——> demo
       |——> Userdefined_Package_Example1.class
```

How to run Java package program?

You have to use the fully qualified name to execute java code. The fully qualified name means class name with a complete package structure.

Syntax:

```
java completePackageName.className
```

Example:

```
java com.shreesoft.demo.Userdefined_Package_Example1
```

How to import package in java?

- An import is a keyword that is used to make the classes and interfaces of other packages accessible to the current package.
- If we use package.*, all the classes and interfaces of this package can be accessed (imported) from outside the packages.
- If you import packageName.className, you can only access the declared class of this package. Let's understand it by an example program.

Example:

```java
package com.shreesoft.main;

public class MathLib {

        public String isEven(int n) {
                return n%2==0?"Even":"Odd";
        }

        public double areaOfCircle(double r) {
                return Math.PI*r*r;
        }
}
```

```java
package com.shreesoft.demo;

import java.util.Scanner;
import com.shreesoft.main.MathLib;

public class Userdefined_Package_Example2 {

	public static void main(String[] args) {
		double r;
		int n;

		MathLib mathLib = new MathLib();

		Scanner scan = new Scanner(System.in);

		System.out.println("Enter Radius:");
		r = scan.nextDouble();

		System.out.println("Area of Circle:"+mathLib.areaOfCircle(r));

		System.out.println("Enter Number:");
		n = scan.nextInt();

		System.out.println("Number is:"+mathLib.isEven(n));

		scan.close();
	}
}
```