DAY-42

-------

PACKAGES

----------------------------

REFER DIAGRAM:

--------------

Package is the collection of related classes and interfaces as a single unit.

packages is a folder contains .class files representing related classes and interface

In java applications packages are able to provide the following advantages:

1. MODULARITY

2. ABSTRATCION

3. SECURITY

4. SHARABILITY

5. REUSABILITY

1. MODULARITY: Dividing the requirment into number of pieces and providing soloution for every individual parts and combining all the solution

to single is called as modularity.

Module is a folder that contains .class files representing the related classes and interface

packages are used to improve the modularity in the application.

2. ABSTRACTION: If we declare classes or interface in a module/package then that classes are not visible in outside the package by default.

hence packages can be used for ABSTRACTION.

3.SECURITY: In java apps packages are able to provide abstraction and encapsulation where both are provide security to the app.

4.SHARABILITY: I fwe declare packages one time tehn we are able to share that packages content to any no of apps or module at a time.

5.REUSABILITY: If we declare a packages one time then we are able to reuse any number of time within a single app or more than one app.

TYPES OF PACKAGES:

--------------------

There are two types of packages:

1. USER DEFINED PACKAGES

2. PREDEFIEND PACKAGES

USER DIFIEND PACKAGES:

------------------------

Synatx for package creation:

-----------------------------

package package\_Name;

Synatax for execution of package:

-----------------------------------

import package\_name.\*;

EXAMPLE: case:1 --> creation of package in current working directory

----------------

package p1;

class B

{

public static void main(String[] args)

{

System.out.println("package creation");

}

}

OUTPUT:

-------

D:\sep\_batch\javaPrograms\DAY-42>javac -d . B.java

D:\sep\_batch\javaPrograms\DAY-42>java p1.B

package creation

EXAMPLE: case:2 --> creation of package in parent directory

----------------

package p1;

class C

{

public static void main(String[] args)

{

System.out.println("package creation 1 ");

}

}

OUTPUT:

--------

D:\sep\_batch\javaPrograms\DAY-42>javac -d .. C.java

D:\sep\_batch\javaPrograms\DAY-42>java p1.c

Error: Could not find or load main class p1.c

D:\sep\_batch\javaPrograms\DAY-42>cd..

D:\sep\_batch\javaPrograms>java p1.C

package creation 1

EXAMPLE: case:3 --> creation of package in random directory

----------------

package p1;

class C

{

public static void main(String[] args)

{

System.out.println("package creation 1 ");

}

}

OUTPUT:

-------

D:\sep\_batch\javaPrograms\DAY-42>javac -d S: C.java

D:\sep\_batch\javaPrograms\DAY-42>S:

S:\>java p1.C

package creation 1

EXAMPLE: case:4 --> creation of SUB-package in current directory

----------------

package p2.p3;

class D

{

public static void main(String[] args)

{

System.out.println("sub-package creation ");

}

}

OUTPUT:

--------

D:\sep\_batch\javaPrograms\DAY-42>java p2.p3.D

sub-package creation

EXAMPLE: case:5 --> DUPLICATE PACKAGES ARE NOT ALLOWED.

----------------

package p1;

class D

{

public static void main(String[] args)

{

System.out.println("sub-package creation ");

}

}

OUTPUT:

-------

D:\sep\_batch\javaPrograms\DAY-42>javac -d . D.java

D:\sep\_batch\javaPrograms\DAY-42>java p1.D

duplicate packages are not permitted

EXAMPLE: CASE:6 --> Multiple package creation within a same class is not allowed

-----------------

package p1;

package p2;

class D

{

public static void main(String[] args)

{

System.out.println("sub-package creation ");

}

}

OUTPUT:

-------

D.java:2: error: class, interface, or enum expected

package p2;

^

case:7

ADDING MULTIPLE CLASSES TO THE SAME PACKAGE:

--------------------------------------------

EXAMPLE:

--------

package Multiple.pack1;

class A

{

public void fun1()

{

System.out.println("inside class A");

}

}

class B

{

public void fun2()

{

System.out.println("inside class B");

}

}

class Demo

{

public static void main(String[] args)

{

System.out.println("creation of multiple classes within a single package");

A a = new A();

a.fun1();

B b = new B();

b.fun2();

}

}

OUTPUT:

-------

D:\sep\_batch\javaPrograms\DAY-42>java Multiple.pack1.Demo

creation of multiple classes within a single package

inside class A

inside class B