**Creating jar file and executing:**

Step1: compile .java file ---> in order to create a .class file.

Step 2: jar -cvf <MyJarFile.jar> <classfilename> ----> to create a jar file

Step 3: java -cp <MyJarFile.jar> <classfilename> -----> to execute class file.

Storing the class file in a jar file is that we can execute class from any location on the file system.

Or to execute from any where

* java -cp ./path to folder/<MyJarFile.jar> <classFileName>

**Creating war file:**

Step1: compile .java file ---> in order to create a .class file.

Step 2: jar -cvf <MyJarFile.war> <classfilename> ----> to create a jar file

Step 3: java -cp <MyJarFile.war> <classfilename> -----> to execute class file.

Storing the class file in a jar file is that we can execute class from any location on the file system.

Or to execute from any where

* java -cp ./path to folder/<MyJarFile.war> <classFileName>

**Creating ear file:**

Step1: compile .java file ---> in order to create a .class file.

Step 2: jar -cvf <MyJarFile.ear> <classfilename> ----> to create a jar file

Step 3: java -cp <MyJarFile.ear> <classfilename> -----> to execute class file.

Storing the class file in a jar file is that we can execute class from any location on the file system.

Or to execute from any where

* java -cp ./path to folder/<MyJarFile.war> <classFileName>

**For maven project:**

* mvn package ---> will create the {artifactid}-{version}.war file under ${basedir}/target directoy. If the packaging POM element is set to war in the pom.xml file.

Note: If no packing value has been specified, it will default to jar file.

**Classpath:** is a parameter in the Java Virtual Machine or Java compiler that specifies the location of user-defined classes and packages.

The parameter may be set either on command-line or through environment variable.

* When we execute java programs, the JVM finds and load classes.
* The classpath tells where to look in the file system of the classes.

JVM searched for and loads classes in order:

1. Bootstrap classes: the classes(bytecode) that are fundamental to the Java platform (public classes of Java library and private classes that are necessary for this library) .
2. Extension classes: packages that are in ext directory of JDK or JRE ( jre/lib/ext).
3. User-defined packages and libraries.

**Setting the path to execute Java programs:**

D:\myprogram\

|

---> org\

|

---> mypackage\

|

---> HelloWorld.class

---> SupportClass.class

---> UtilClass.class

Windows:

* Java -classpath D:\myprogram org.mypackage.HellowWord

Java -classpath <path to the package> <pkgName.classFileName>

Linux:

* Java -cp D:\myprogram org.mypackage.HellowWord

Java -cp <path to the package> pkgName.classFileName

**Setting the path through an environment variable:**

The environment variable named **CLASSPATH** may be alternatively used to set classpath.

* Set CLASSPATH=D:\myprogram

>java org.mypackage.Helloworld

set CLASSPATH=<path of the directory>

[ if none specified then the Current working directory is used as classpath (i.e. “.” ].

**Setting the path of a jar file:**

if a program uses a supporting library enclosed in a jar file physically located in a directory.

D:\myprogram\

|

---> lib\

|

---> supportLib.jar

|

---> org\

|

--> mypackage\

|

---> HelloWorld.class

---> SupportClass.class

---> UtilClass.class

* java -classpath D:\myprogram;D:\myprogram\lib\supportLib.jar org.mypackage.HelloWorld

java -classpath <path to the package<;<path including .jar file> <pkgName.classFileName>

or Alternatively.

* set CLASSPATH=<path to the package<;<path including .jar file> <pkgName.classFileName>

**Adding all jar file in a directory:**

* **java -classpath “.;<path to the directory>\\*” <directoryName>**

@@ important topic:

**Setting the path in a manifest file:** if the program has been enclosed in a jar file, located directly in D

D:\myprogram\

|

---> helloWorld.jar

|

---> lib\

|

---> supportLib.jar

The manifest file defined in helloWorld.jar has this definition.

Main-Class: org.mypackage.HelloWord

Class-Path: lib/supportLib.jar

The program is launched with following command.

* java -jar D:\myprogram\helloWorld.jar [app arguments]