## javac

We use javac command to compile single or group of java source file

javac [options] test.java

javac [options] A.java B.java C.java

javac [options] \*.java

-version

-d

-source (compile based on java 1.4 versions)

-cp/ -classpath

-verbose (internally what is happening view)

……

## java

We use java command to run a single class file

java [options] Test A B C

command-line arguments

-version

-cp/ -classpath

-D

-ea / -esa/ -dsa/ -da

…..

Note : at a time compile n number of .java file, but run only one .class file

## Class path

classpath describe the location where required .class files are available.

java-compiler and jvm use classpath to locate required .class files

class Hello{

p s v m(String arg[]){

Sop(“hello”)

}

}

c:\myJava> javac helloworld.java

c:\myJava\Hello.class will be generated

c:\myJava> java Hello

hello

c:\> java Hello

RE: noClassDeffFoundError : Hello

by default jvm always searches in current working directory(c:/) for the required .class file(Hello.class)

If we set class path explicitly

then jvm will search in our specified classpath location

and jvm wont search in current working directory

1. System level: environment variable classpath

It’s permanent

When we’re installing permanent software it’s recommended

1. CMD level: once cmd closed it’s gone

set classpath=c:\myJava

1. Command level: once command execute it’s gone

c:\> java –cp c:\myJava Hello

Once we set class path we can run it from any location

C:\hello\games> java –cp c:\myJava Hello

E:\movies> java –cp c:\myJava Hello

After setting the classpath jvm wont search in current directory and always search specified class path only.

First search in current-work-directory if “Hello” is not there then search for C:\myJava.

E:\movies> java –cp .;C:\myJava Hello

Test class path

> echo %CLASSPATH%

Setting class path at class level is recommended

Because dependent classes varied from command to command

>set path=C:\Program Files\Java\jdk1.8.0\_271\bin

|  |  |
| --- | --- |
| C:  class MyClass{  public void hello(){  Sop(“hello”);  }  } | D:  class Test{  public static void main(String arg[]){  MyClass obj = new MyClass();  obj.hello();  }  } |

C:\> javac MyClass.java

C:\ MyClass.class will be generated

D:\> javac Test.java

CE: cannot find symbol : MyClass

D:\> javac –cp C: Test.java

D:\ Test.class will be generated. Because required MyClass can be found in C:

D:\> java Test

RE: NoClassDefFoundError: MyClass

D:\> java –cp C: Test

RE: NoClassDefFoundError: Test

D:\> java –cp .;C: Test

Output: hello

E:\> java –cp D:;C: Test

Run well. Because in order to run Test it required class files(Test.class, MyClass.class) canbe found in C: and D:

|  |  |  |
| --- | --- | --- |
| C:  package pack1.pack2;  class A{  public void m1(){  Sop(“hello”);  }  } | D:  package pack3.pack4;  import pack1.pack2.A;  class B{  public void m2(){  A a = new A();  a.m1();  }  } | E:  import pack1.pack2.B;  class Test{  p s v main(String arg[]){  B b = new B();  b.m2();  }  } |

C:\> javac –d . A.java

C:

-pack1

-pack2

A.class

D:\> javac –d . B.java

CE: cannot find symbol

symbol: class A

location: pack3.pack4.B

D:\> javac –d . –cp C: B.java

fine

E:\> javac –d . –cp D: Test.java

fine

E:\> java Test

RE: NoClassDefFoundError: pack3.pack4.B

E:\> java –cp .;D: Test

RE: NoClassDefFoundError: pack1.pack2.A

E:\> java –cp .;D:;C: Test

fine

F:\> java –cp E:;D:;C: Test

fine

|  |  |  |
| --- | --- | --- |
| C:  class A{  public void m1(){  Sop(“C: hello”);  }  } | D:  class A{  public void m1(){  Sop(“D: hello”);  }  } | E:  class A{  public void m1(){  Sop(“E: hello”);  }  } |

F:\> java –cp E:;D:;C: Test

output E:hello

In classpath jvm will always consider left to right

F:\> java –cp E:;D:;C:;E:;…. Test

If several dependent classes are there not recommended to set the classpath separately

So we have to group all dependent classes into a zip file (Group of .class files is .jar file) then make this zip file available in the classpath

## JAR

All 3rd party plugins by default available in the form of .jar

Examples:

* To develop a servlet all dependent classes are available in

Servlet-api.jar we have paste this jar file in classpath to compile a servlet program

* To JDBC program all dependent classes are available in

Ojdbc14.jar to run jdbc program we have to place this jar in classpath

* To Use Log4J in our appilacation dependent classes are available

in log4.jar to run Log4J based application can run

### Create jar files

**jar** –cvf myjar.jar Test.class

c: create

v: verbos

f: named file

**jar** –cvf myjar.jar A.class B.class C.class …

c:\file> **jar** –cvf myjar.jar \*.class

Whatever classes are in current working directory (c:\file) with all those classes create jar file

**jar** –cvf myjar.jar \*.\*

with all those create jar file

### Extract jar file (unzip)

**jar** –xvf myjar.jar

x: extract

### Display tableOfContent of jar file

**jar** –tvf myjar.jar

x: table of content

|  |  |
| --- | --- |
| Service provider | Client’s Role |
| class TharsiAlgo{  public static void thasi(int a, int b){  Sop(a+b\*3+32);  }  }  javac TharsiAlgo.java  jar –cvf TharsiAlgo.jar TharsiAlgo.class | Download TharsiAlgo.jar then placed in D:\ of client’s machine  class HelloAi{  p s v main(String arg[]){  TharsiAlgo.thasi(25,10);  }  } |

But c:\project> javac HelloAi.java

CE: cannot find symbol

c:\project> javac –cp D: HelloAi.java

CE: cannot find symbol

but only location is not enough to get that dependent class file

c:\project> javac –cp D:\TharsiAlgo.jar HelloAi.java

c:\project> java HelloAi

RE: noClassDefFondError: TharsiAlgo

c:\project> java –cp D:\TharsiAlgo.jar HelloAi

RE: noClassDefFondError: HelloAi

c:\project> java –cp .;D:\TharsiAlgo.jar HelloAi

### Shortcut way to place jar file in classpath

if we pace jar file in the following location all classes in the jar file will be available to jvm and compiler

C:\Program Files\Java\jdk1.8.0\_271\jre\lib\ext \ \*myjar.jar

### System properties

For every system some persistence information will be maintain in the form of system properties.

import java.util.\*;

class Hello{

public static void main(String arg[]){

Properties p = System.getProperties();

p.list(System.out);

}

}

this will display all properties

To add more properties when run this..

java –Dname = hello Hello

-D[property\_key] = [value]

(space is not allowed after -D)

The main advantage of setting system property is we can customize behavior of our program

class Hello{

public static void main(String arg[]){

String course = System.getProperties(“course”);

if (course.equals(“java”)){ Sop(“java info”); }

else { Sop(“html info”); }

}

}

## jar vs. war vs. ear

#### jar (java archive)

It contains group of .class file

#### war (web archive)

war file represents one web application.

Which contains servlets, jsp, html pages, js, css, xml, .. etc.

So Project Deployment, delivery, and transportation will be easy.

#### ear (enterprise archive)

ear file represents one enterprise application

Which contains servlets, jsps, ejabs, jms components, .. etc.

Note: in general ear file represents a group of war files and jar files

### web application vs. enterprise application

A web application can be developed by only web related technologies like servlets, jsps, + html, css, js ..

example: online library management system

online shopping card

An enterprise application can be developed by any technology from javaEE like servlets, jsps, ejbs, jms components ……

examples: banking applications

telecom based project

### web server vs. application server

Web-server provide environment to run web application

Web-server can provide support for web related technologies like servlets, jsps, html and etc.

example : tomcat

Application-server provide environment to run enterprise application

Application-server can provide support for any technology from javaee. like servlets, jsps, ejbs, jms components ……

example : webLogic, JBoss, webSphere

+

ejbs

jms

..

Servlets

Jsps

HTML

JS

…

Servlets

Jsps

HTML

JS

…

web-server

application-server

Every application server contain in build web server to provide support for web related technology

JEE compatible server is application server

### Executable jar and bat

After crating executable jar just

jar –cvfm manifest.mf myJar.jar Hello.class Hi.class

to run :

java –jar myJar.jar

or just create .bat within that type the above command-line command

If it is aws then double click jar will open the window

## classpath vs. path

classpath: location where required .class files are available

jvm, compiler will use this classpath

if we don’t set classpath then code may not compile or run

path: location where required **bin**ary-executables(bin) [non-text] are available. [jdk/bin]

if we are not set path then javac, and java commands won’t work (javac = javac.exe, java=java.exe)

example : vlc F:\one.mp4

vlc.exe argument1 argument2

## JDK vs. JRE vs. JVM

jdk provide environment to provide environment to **develop and run** java-applications

jre provides environment to **run** java-applications

jvm is responsible to run java program line by line hence it is an interpreter

**Jdk**

**Jre**

Development

tools

jre + develop tools = jdk

jvm + library classes = jre

jvm

So in client machine we have to install jdk

in the developer machine we have to install jdk

#### java vs. javaw vs. javaws keyworld

java : to run java .class file, where sop() will be executed, corresponding output will be displayed to cmd

javaw : to run java .class file, where sop() will be executed,

but corresponding output won’t be displayed to cmd

this use to run GUI applications

javaws (java web start utility)

javaws url

Download the application from web, then execute it

so every end user will get updated version

enhancement will be easy because of centralized control