Mobile Information Device Programming (3)

Lecturer: Alireza Mousavi
School of Engineering & Design
www.brunel.ac.uk/~emstaam

How to Get Started

- Installation of J2ME suite on your Personal Computer
  - Downloading the Software
  - Installing the suite
  - Java development Kit
  - Testing your installation
  - Troubleshooting

Software Download

The minimum requirements for running applications are:

1. Release 1.4 of the Java 2 Platform, Standard Edition

   http://java.sun.com/j2se/1.4/index.jsp

2. Connected Device Limited Configuration (CLDC)
3. Mobile Information Device Profile (MIDP)
4. J2ME Wireless Toolkit

Items 2-4: http://java.sun.com/products/j2mewtoolkit/download-2_1.html

1. Install J2SDK

- Install the package as instructed

- Update the PATH environment variable (this allows you to run the Java compiler from any directory without specifying the full path each time) – example for *Windows 2000*

  - *Control Panel choose System*
  - *Click Environment (Advanced/Environment) – Find the PATH entry and at the end add the location of the\bin directory in your J2SDK install path – add the following to the end of the path:*

    c:\JDK1.4.2\bin (pending the version of the software)

2. Install CLDC

- Create a directory called *J2ME* on your hard-drive e.g. *c* or *d*

- Download the CLDC zip files and extract them into *C:\J2ME*

- Update the PATH Environment variable – *c:\J2ME\J2ME_CLDC/bin*

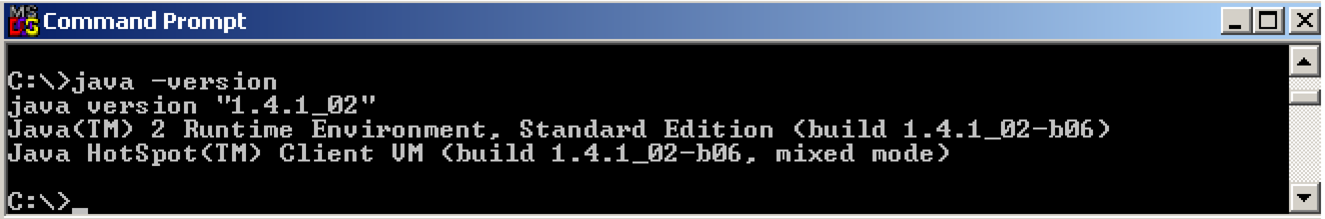
Install MIDP

- Download MIDP latest version
- Save the zip file into previously created *J2ME* (*c:|J2ME*) and extract
- Update the PATH Environment Variable – *c:|j2me|MIDP2.0fcs|bin*
- Add/Create the CLASSPATH Environment Variable – *CLASSPATH=c:|J2ME|MIdp2.0fcs|classes;.*

  Note: the "." at the end of command represents the current working directory
- Add/Create MIDP_HOME Environment Variable – *MIDP_HOME=c:|j2me|midp2.0fcs*

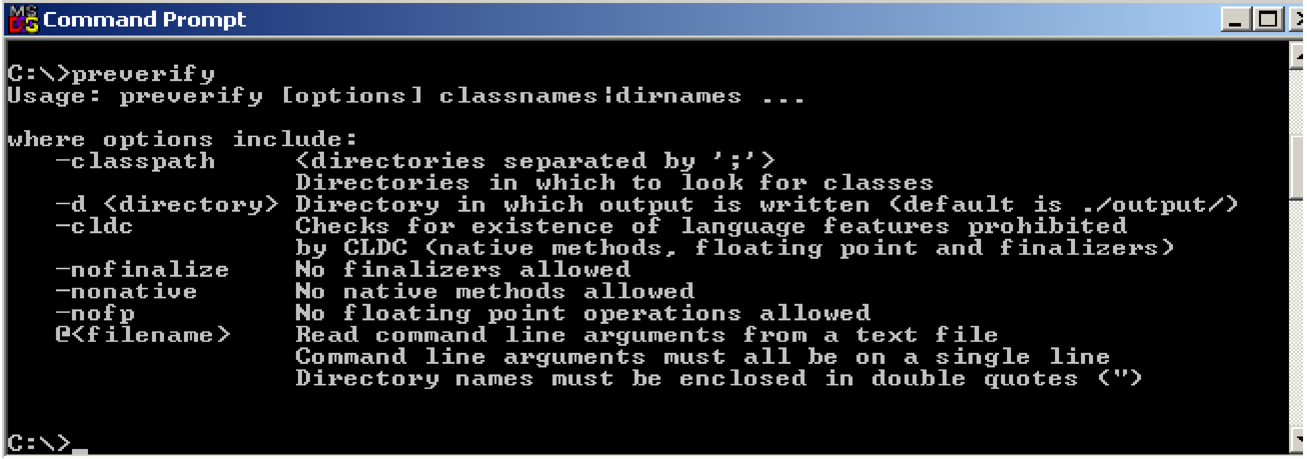
Installation Test

1. Go to the command prompt.
2. Test your JDK installation by typing "*java –version*"



```
C:\>java -version
java version "1.4.1_02"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.4.1_02-b06)
Java HotSpot(TM) Client VM (build 1.4.1_02-b06, mixed mode)

C:\>_
```

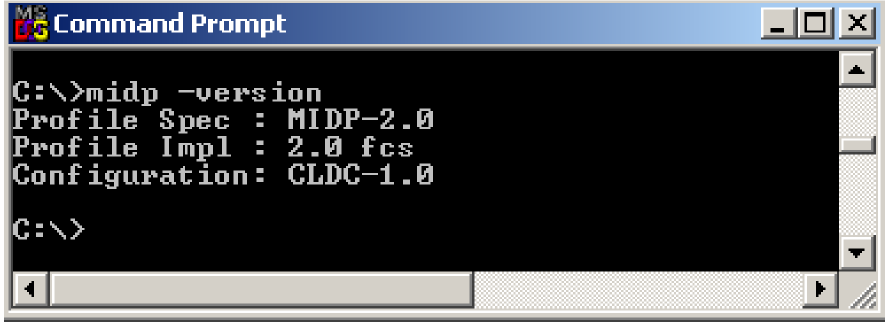
3. Test the CLDC installation by typing "*preverify*"



```
C:\>preverify
Usage: preverify [options] classnames|dirnames ...

where options include:
   -classpath     <directories separated by ';'>
                  Directories in which to look for classes
   -d <directory> Directory in which output is written (default is ./output/)
   -cldc          Checks for existence of language features prohibited
                  by CLDC (native methods, floating point and finalizers)
   -nofinalize    No finalizers allowed
   -nonative      No native methods allowed
   -nofp          No floating point operations allowed
   @<filename>    Read command line arguments from a text file
                  Command line arguments must all be on a single line
                  Directory names must be enclosed in double quotes (")


C:\>_
```

Test Installation cont.

4. Check the MIDP installation by typing “*midp –version*”



```
C:\>midp -version
Profile Spec : MIDP-2.0
Profile Impl : 2.0 fcs
Configuration: CLDC-1.0

C:\>
```

Troubleshooting

In case any of the programs fail to run:

- Make sure the installations are correct – for example run *midp.exe* in the *c:|j2me|midp2.0fcs|bin*

- Make sure the PATH settings are correct

- Make sure that the reference to the current working directory (“.”) as part of the CLASSPATH environment variable

- Verify the directory paths in the PATH environment variable

The MIDP Application Development Process

MIDP application development is similar to other Java Application developments

According to MIDP 2.0 specification, a MIDlet is an application that is developed for mobile devices and can only use MIDP and CLDC APIs

The steps for developing and testing MIDlets

1. Write the program in Java for the MIDlet
2. Compile the MIDlet source code
3. Pre-verify the MIDlet compiled code – ensure that the code executes safely – it verifies codes before downloading on device
4. Provide attributes that are used during the application installation and execution (create the “manifest.mf” file)
5. Package the MIDlet by creating the *JAR* file containing the application resources and the *JAD* file containing application information. It helps to keep applications organised
6. Execute the application using the emulator

Write Java Code

1. Create a file called midlets e.g. c:\midlets
2. Upload JBuilder, Create a new Project, Open a new Java File
3. Write the code
4. Save file e.g. c:\midlets\Welcome\Welcome.java

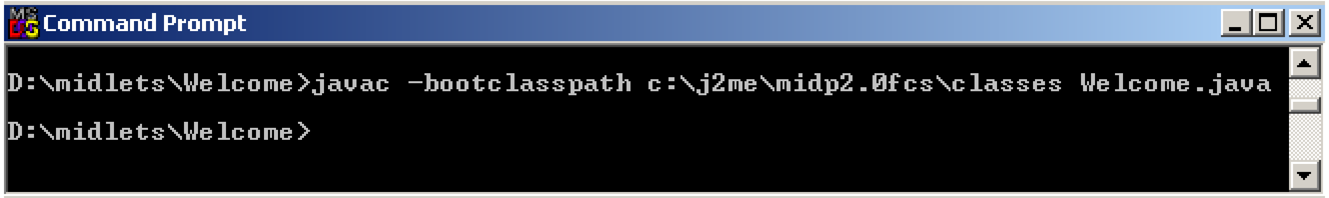
Compile and Pre-verify MIDlets

Two ways:

1. Use of Command Line
2. Use of J2ME Wireless Toolkit

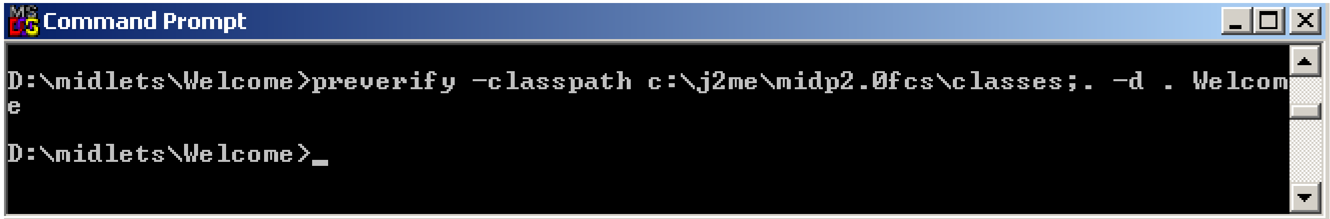
Compile and Pre-verify and Run MIDlets using command prompt

1. Go to command prompt
2. Compile the java file



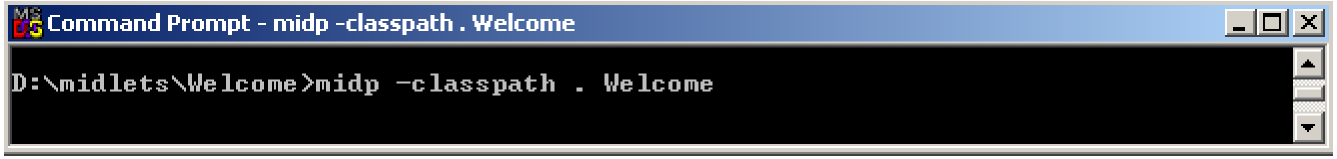
```
D:\midlets\Welcome>javac -bootclasspath c:\j2me\midp2.0fcs\classes Welcome.java

D:\midlets\Welcome>
```

3. Pre-verify the class file



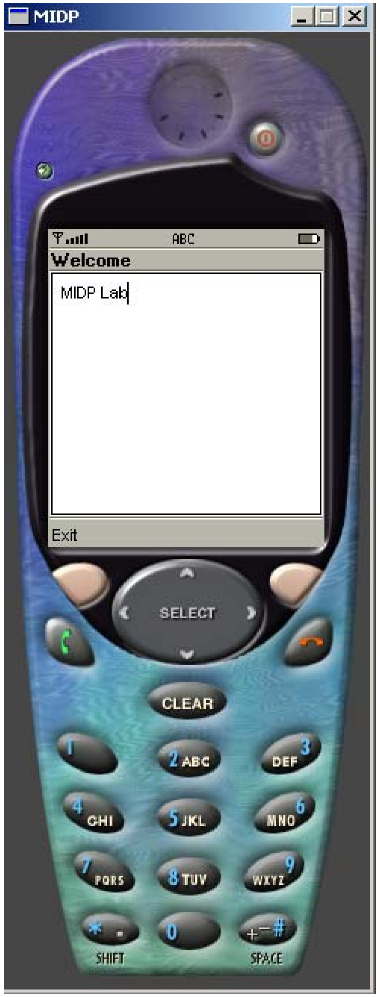
```
D:\midlets\Welcome>preverify -classpath c:\j2me\midp2.0fcs\classes;. -d . Welcom
e

D:\midlets\Welcome>_
```

4. Run the MIDlet



```
D:\midlets\Welcome>midp -classpath . Welcome
```

The Result



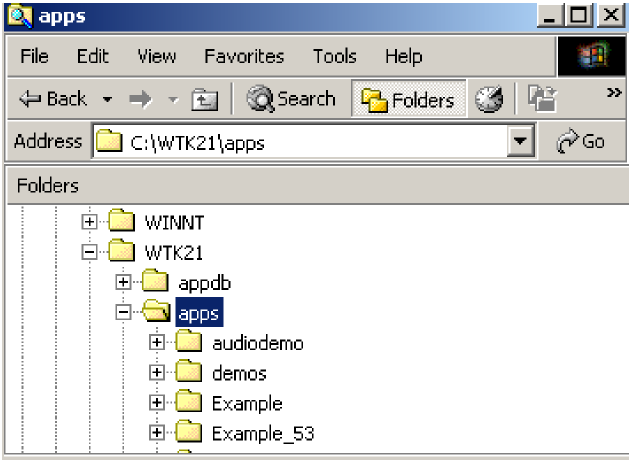
Welcome

MIDP Lab

Exit

Compile and Pre-verify and Run MIDlets using J2ME Wireless ToolKit

1. Once you have created you Java File in your Project transfer your files under "*apps*" directory – e.g. c:\Wtk21\apps



Address C:\WTK21\apps

- WINNT
- WTK21
  - appdb
  - apps
    - audiodemo
    - demos
    - Example
    - Example_53

2. Create a directory called "*src*" and copy the Java File to this directory

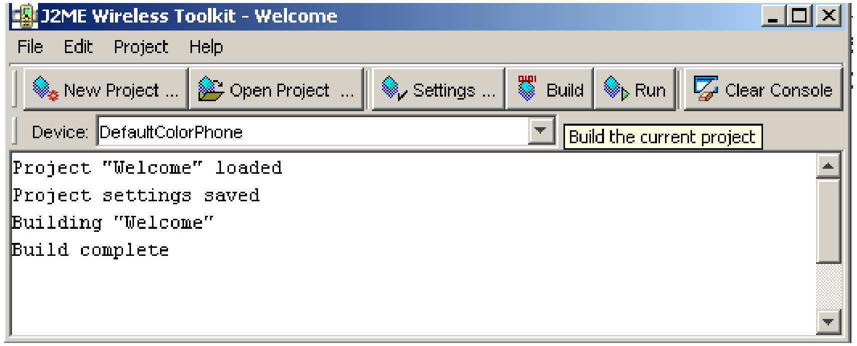
Compile and Pre-verify and Run MIDlets using J2ME Wireless ToolKit cont.

3. Launch the Wireless Toolkit software and create the project with the name of the source code.

   “You may get a message saying that it already exists ! Why? – If so just overwrite. The other option is to create the project in the Wireless Toolkit in advance and copy the Java file into a “src” directory within the project.”

4. Load and Build the Project

   “Note the files and directories created see the \bin directory JAR & JAD created automatically

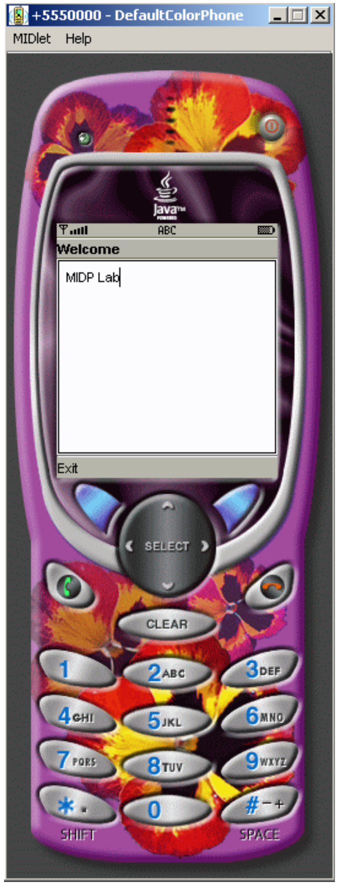


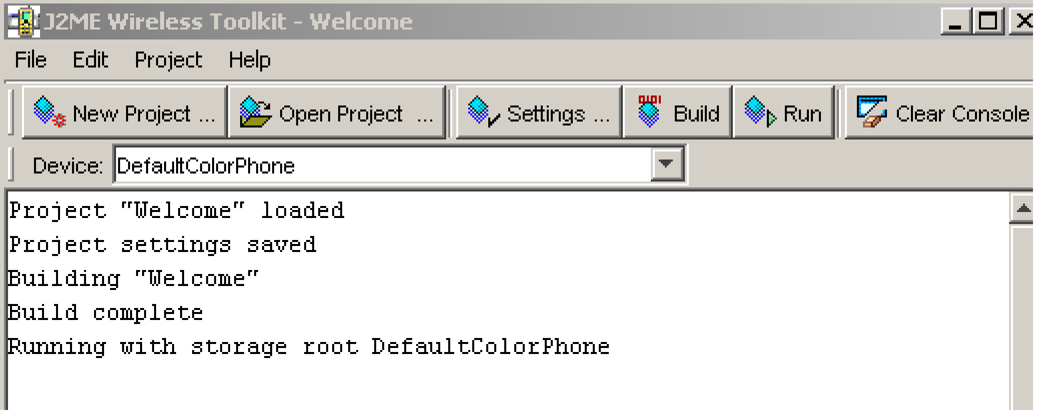
Device: DefaultColorPhone

```
Project "Welcome" loaded
Project settings saved
Building "Welcome"
Build complete
```

Compile and Pre-verify and Run MIDlets using J2ME Wireless ToolKit cont.

5. Run the project





Device: DefaultColorPhone

```
Project "Welcome" loaded
Project settings saved
Building "Welcome"
Build complete
Running with storage root DefaultColorPhone
```

Packaging the MIDlet

- Here we use an example:

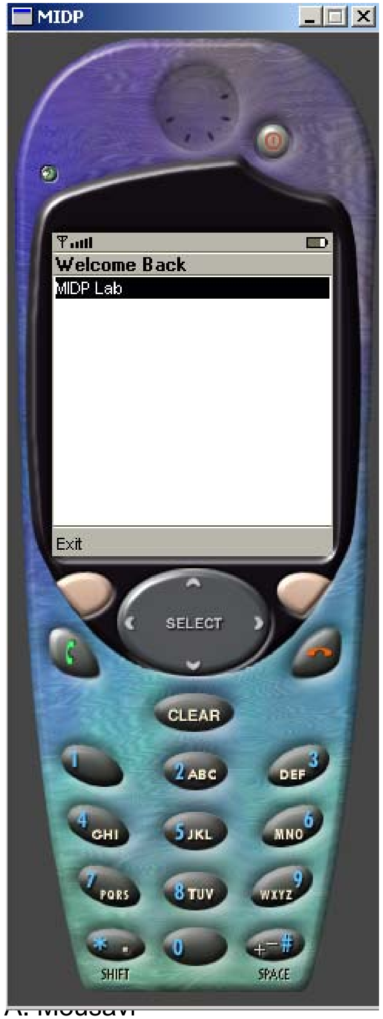
1. Create another project under the c:\midlets directory and call it WelcomeBack

2. Create the WelcomeBack java File

3. Save the file

4. Compile, Pre-verify and run using the command prompt

Result



Welcome Back

MIDP Lab

Exit

Packaging the MIDlet cont.

5. Create the *JAR* file

   5.1 Create a new directory called WelcomeJar where both examples Welcome and WelcomeBack were created i.e. c:\midlets

   5.2 Copy both class files(Welcome.class and WelcomeBack.class) into the directory

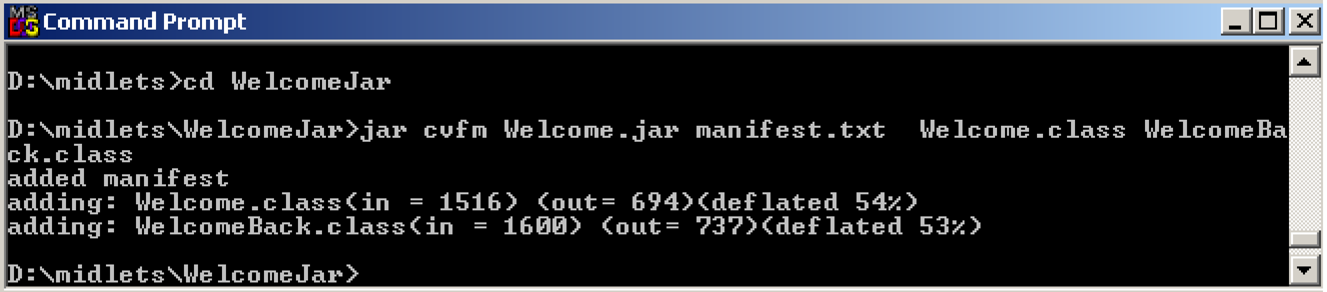
   5.3 Create a new file called mainfest.txt and type the Jar file attributes

Manifest.txt

```
MIDlet-Name: Welcome Example
MIDlet-Version: 1.0
MIDlet-Vendor: Brunel Lab
MIDlet-1: Welcome, Welcome.png, Welcome
MIDlet-2: WelcomeBack, WelcomeBack.png, WelcomeBack
MicroEdition-Profile: MIDP-2.0
MicroEdition-Configuration: CLDC-1.0
```

Packaging the MIDlet cont.

5.4 Create the JAR file using the command prompt (make sure you change the directory to WelcomeJar)



```
D:\midlets>cd WelcomeJar

D:\midlets\WelcomeJar>jar cvfm Welcome.jar manifest.txt  Welcome.class WelcomeBa
ck.class
added manifest
adding: Welcome.class(in = 1516) (out= 694)(deflated 54%)
adding: WelcomeBack.class(in = 1600) (out= 737)(deflated 53%)

D:\midlets\WelcomeJar>
```

6. Create the *JAD* file

Welcome.jad

```
MIDlet-Name: Welcome Example
MIDlet-Version: 1.0
MIDlet-Vendor: Brunel Lab
MIDlet-Description: Two Midlets in one Package
MIDlet-Jar-URL: Welcome.jar
MIDlet-Jar-Size: 2114
MIDlet-1: Welcome, Welcome.png, Welcome
MIDlet-2: WelcomeBack, WelcomeBack.png, WelcomeBack
MicroEdition-Profile: MIDP-2.0
MicroEdition-Configuration: CLDC-1.0
```

Packaging the MIDlet cont.

7. Run the MIDlet suite using:

   midp –classpath . –Xdescriptor Welcome.jad

8. Now you have three Directories Welcome, WelcomeBack and WelcomeJar, yet not a package.

Lets continue !

Packaging the MIDlet cont.

8.1 Make a directory for the new project e.g. WelcomePackage under c:\midlets

8.2 Create three subdirectories:

- jclasses → output from java compiler
- pclasses → output from preverifier
- resources → resource files (images, and so on)

8.3 Copy the Welcome.java and WelcomeBack.java into the WelcomePackage directory.

8.4 Open the files and add the following statement and save:

“*package FirstPackage*”

Packaging the MIDlet cont.

8.5 Copy and create the JAR and JAD file. Make sure you update the paths

Manifest.txt

```
MIDlet-Name: Welcome Example
MIDlet-Version: 1.0
MIDlet-Vendor: Brunel Lab
MIDlet-1: Welcome, /resources/circle.png, FirstPackage.Welcome
MIDlet-2: WelcomeBack, /resources/circle.png, FirstPackage.WelcomeBack
MicroEdition-Profile: MIDP-2.0
MicroEdition-Configuration: CLDC-1.0
```

Welcome.jad

```
MIDlet-Name: Welcome Example
MIDlet-Version: 1.0
MIDlet-Vendor: Brunel Lab
MIDlet-Description: Two Midlets in one Package
MIDlet-Jar-URL: Welcome.jar
MIDlet-Jar-Size: 2114
MIDlet-1: Welcome, Welcome.png, Welcome
MIDlet-2: WelcomeBack, WelcomeBack.png, WelcomeBack
MicroEdition-Profile: MIDP-2.0
MicroEdition-Configuration: CLDC-1.0
```

Packaging the MIDlet cont.

8.6 Compile and preverify

```
Javac –bootclasspath c:\j2me\midp2.0fcs\classes –d jclasses *.java
```

"-d": option tells the compiler to write the classes files into jclasses

```
preverify –classpath c:\j2me\midp2.0fcs\classes; –d .  pclasses  jclasses
```

"-d" . : option tells the pre-verifier to store the verified classes in the pclasses directory. "jclasses" at the end tells the pre-verifier where to look for the classes to verify

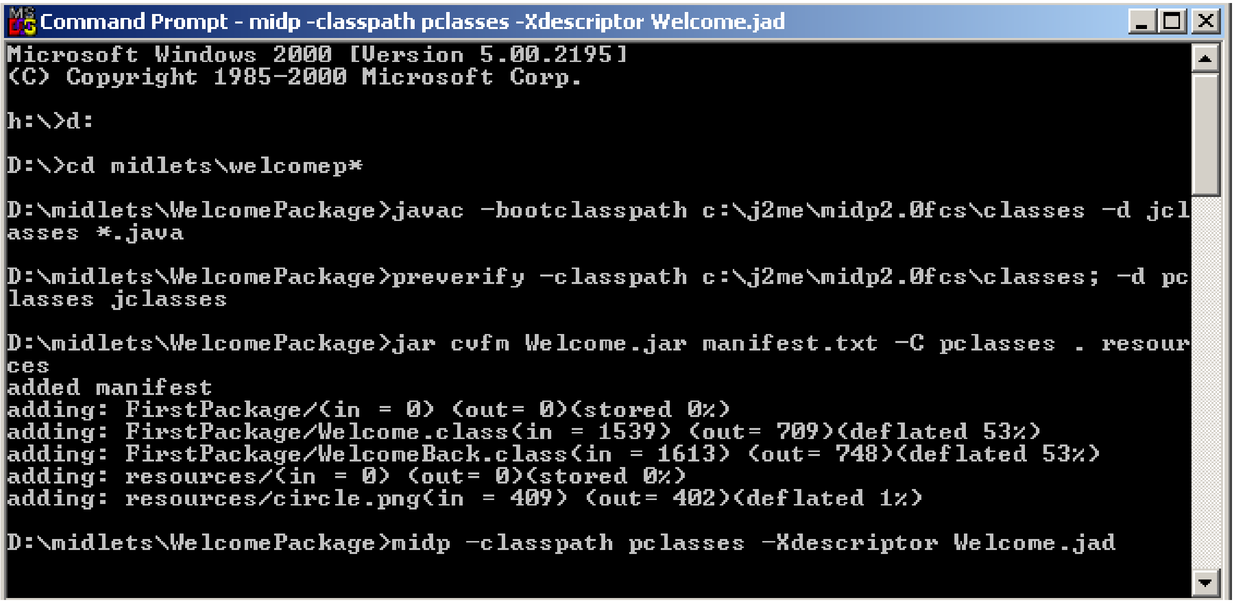
8.7 Run the jar executable to create the JAR file

```
jar cvfm Welcome.jar manifest.txt –C  pclasses . Resources
```

8.8 Run the MIDlets

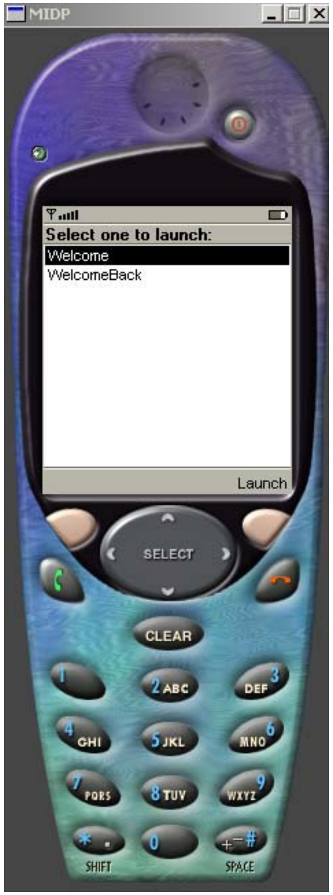
```
midp  -classpath pclasses –Xdescriptor Welcome.jad
```

Result



Command Prompt - midp -classpath pclasses -Xdescriptor Welcome.jad

```
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

h:\>d:

D:\>cd midlets\welcomep*

D:\midlets\WelcomePackage>javac -bootclasspath c:\j2me\midp2.0fcs\classes -d jcl
asses *.java

D:\midlets\WelcomePackage>preverify -classpath c:\j2me\midp2.0fcs\classes; -d pc
lasses jclasses

D:\midlets\WelcomePackage>jar cvfm Welcome.jar manifest.txt -C pclasses . resour
ces
added manifest
adding: FirstPackage/(in = 0) (out= 0)(stored 0%)
adding: FirstPackage/Welcome.class(in = 1539) (out= 709)(deflated 53%)
adding: FirstPackage/WelcomeBack.class(in = 1613) (out= 748)(deflated 53%)
adding: resources/(in = 0) (out= 0)(stored 0%)
adding: resources/circle.png(in = 409) (out= 402)(deflated 1%)

D:\midlets\WelcomePackage>midp -classpath pclasses -Xdescriptor Welcome.jad
```



Select one to launch:

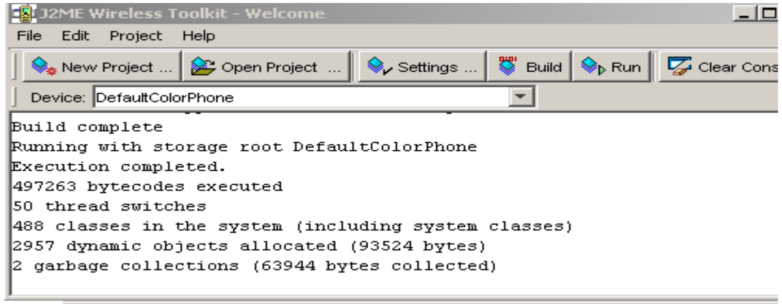
Welcome

WelcomeBack

Launch

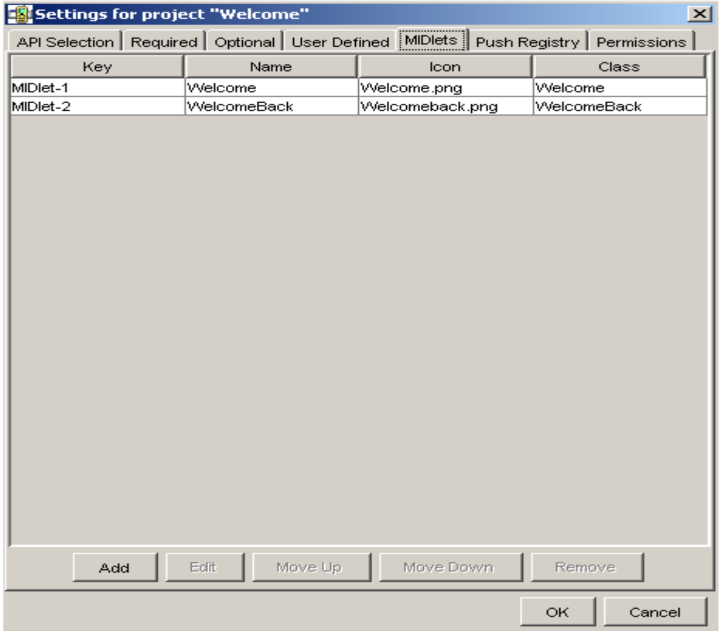
Create the Package using Wireless Toolkit

1. Copy the WelcomeBack.java into the “c:\wtk21\Welcome\src”
2. Update the configuration of manifest file – settings<<MIDlets <<Add
3. Build project
4. Run
5. Package – Project<<Package



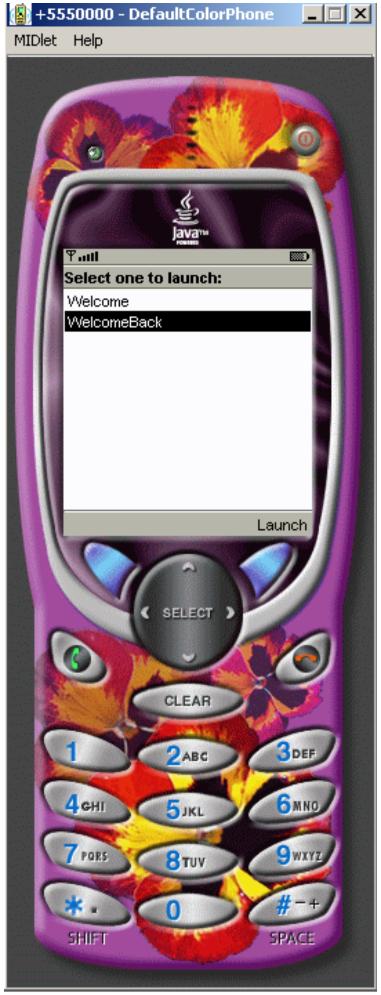
Device: DefaultColorPhone

```
Build complete
Running with storage root DefaultColorPhone
Execution completed.
497263 bytecodes executed
50 thread switches
488 classes in the system (including system classes)
2957 dynamic objects allocated (93524 bytes)
2 garbage collections (63944 bytes collected)
```



Key	Name	Icon	Class
MIDlet-1	Welcome	Welcome.png	Welcome
MIDlet-2	WelcomeBack	Welcomeback.png	WelcomeBack

Result



Select one to launch:

- Welcome
- WelcomeBack

Launch