

Java ME for leJOS project

Versión 0.1

Juan Antonio Breña Moral

7-ene-09

Index

1.- Introduction	4
1.1.- Goals.....	4
1.2.- leJOS project	4
1.2.1.- leJOS technologies	4
1.2.2.- Java leJOS NXJ	5
1.3.- About the author	5
2.- Getting Started	6
2.1.- Introduction.....	6
2.2.- Install SDKs	6
2.2.1.- Installing J2SE SDK.....	6
2.2.2.- Checking your J2SE Installation	6
2.2.3.- Installing Java ME Wireless Toolkit	6
2.3.- Install IDE.....	6
2.3.1.- Installing Eclipse IDE.....	6
2.3.2.- Installing Eclipse ME Plugin	6
2.4.- Install Optional Software	6
2.4.1.- Proguard	6
2.4.2.- Antenna.....	6
3.- Creating your first Java ME project	6
3.1.- Introduction.....	6
3.2.- Creating a new Java ME Project.....	6
3.3.- Java ME Settings.....	6
3.4.- Add a HelloWorld Middlet.....	6
3.5.- Test your Middlet with a simulator.....	6
3.6.- Package your application	6
3.7.- Deliver your Middlet into a Mobile Phone	6
3.7.1.- Deliver Java ME with Nokia PC Suite	6

Revision History

Name	Date	Reason For Changes	Version
Juan Antonio Breña Moral	03/01/2009	Initial Version	0.1

1.- Introduction

1.1.- Goals

Many developers around the world choose leJOS, Java for Lego Mindstorm, as the main platform to develop robots with NXT Lego Mindstorm. I consider that this eBook will help leJOS community, Lego Mindstorm community, Robot's developers and Java fans to develop better software.

Robotics will be very important for the humanity in the next 10 years and this eBook is an effort to help in this way.

Many people spend several hours in their robotics projects with problems with wires & electronics, protocols and problems with programming languages, Lego Mindstorm is easy and Java/leJOS is an excellent platform to demonstrate your software engineering skills to develop better robots. NXT Brick is the easiest way to enter in the robotics world and leJOS the best platform in the moment to use software engineering ideas.

Enjoy, Learn, Contact with me to improve the eBook and share your ideas.

Juan Antonio Breña Moral.
www.juanantonio.info

1.2.- leJOS project

leJOS project is Open Source project designed to develop a JVM for Lego Mindstorms. A long the time the project has grown, developing others subprojects as: PC API, Mobile API & iCommand.

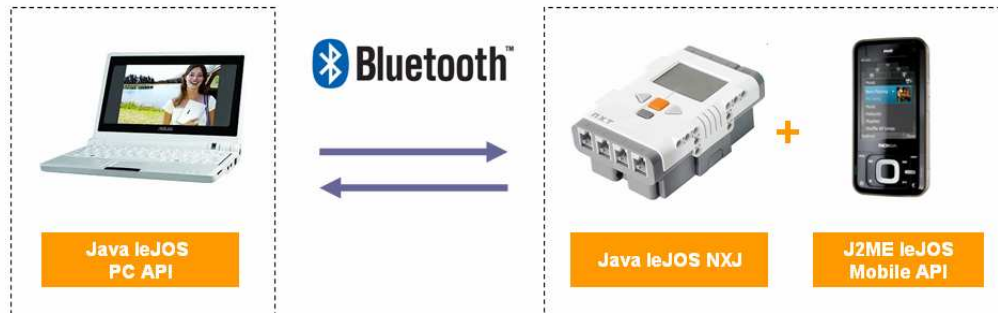
1.2.1.- leJOS technologies

leJOS project for Lego Mindstorms NXT develop and maintain the following subprojects:

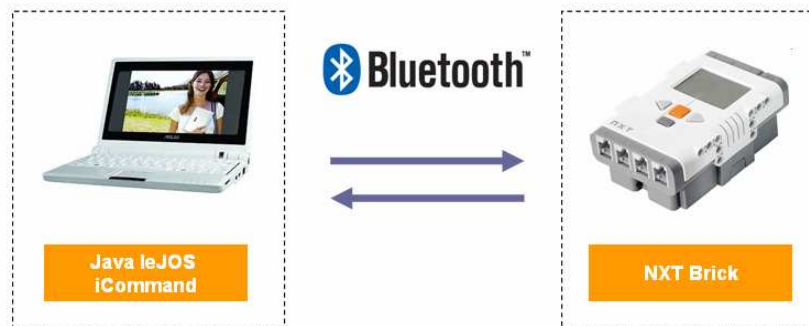
- Java leJOS NXJ
- Java leJOS PC API
- Java leJOS Mobile API
- Java leJOS iCommand

In leJOS project exists 2 approaches about developing LEGO robots with Java leJOS.

1. Develop your software in your NXT brick and collaborate with others environments (PC & Mobiles Devices)
 - a. NXJ
 - b. PC API
 - c. Mobile API
2. Develop your software in your PC or Mobile Device
 - a. iCommand
 - b. Mobile API



Approach 1



Approach 2

1.2.2.- Java leJOS NXJ

leJOS NXJ is a Java programming environment for the LEGO MINDSTORMS NXT ®. It allows you to program LEGO ® robots in Java.

It consists of:

- Replacement firmware for the NXT that includes a Java Virtual Machine.
- A library of Java classes (classes.jar) that implement the leJOS NXJ Application Programming Interface (API).
- A linker for linking user Java classes with classes.jar to form a .binary file that can be uploaded and run on the NXT.
- PC tools for flashing the firmware, uploading programs, debugging, and many other functions.
- A PC API for writing PC programs that communicate with leJOS NXJ programs using Java streams over Bluetooth or USB, or using the LEGO Communications Protocol.
- Many sample programs

1.3.- About the author



Juan Antonio Breña Moral collaborates in leJOS Research team since 2006. He works in Europe leading Marketing, Engineering and IT projects for middle and large customers in several markets as Defence, Telecommunications, Pharmaceuticals, Energy, Automobile, Construction, Insurance and Internet.

Further information:

www.juanantonio.info
www.esmeta.es

2.- Getting Started

2.1.- Introduction

If you need to develop Java software to be used on a mobile device, you need to install the following software on your PC Computer:

- SDK
 - J2SE SDK
 - Java ME Wireless Toolkit
- IDE
 - Eclipse
 - Eclipseme plugin
- Optional software
 - Proguard
 - Antenna
- Software to deliver Java ME software
 - Nokia PC Suite (If you use Nokia Mobile Phones)

2.2.- Install SDKs

2.2.1.- Installing J2SE SDK

If you are going to develop Java ME Software, it is necessary to download latest Java SDK in your computer. Download latest Java SDK from Sun Microsystems's website: <http://java.sun.com/javase/downloads/index.jsp> Once you have downloaded your latest Java SDK use the assistant to install Java SDK.

NOTE:

When I wrote this document, the installer used was: jdk-6u3-windows-i586-p.exe

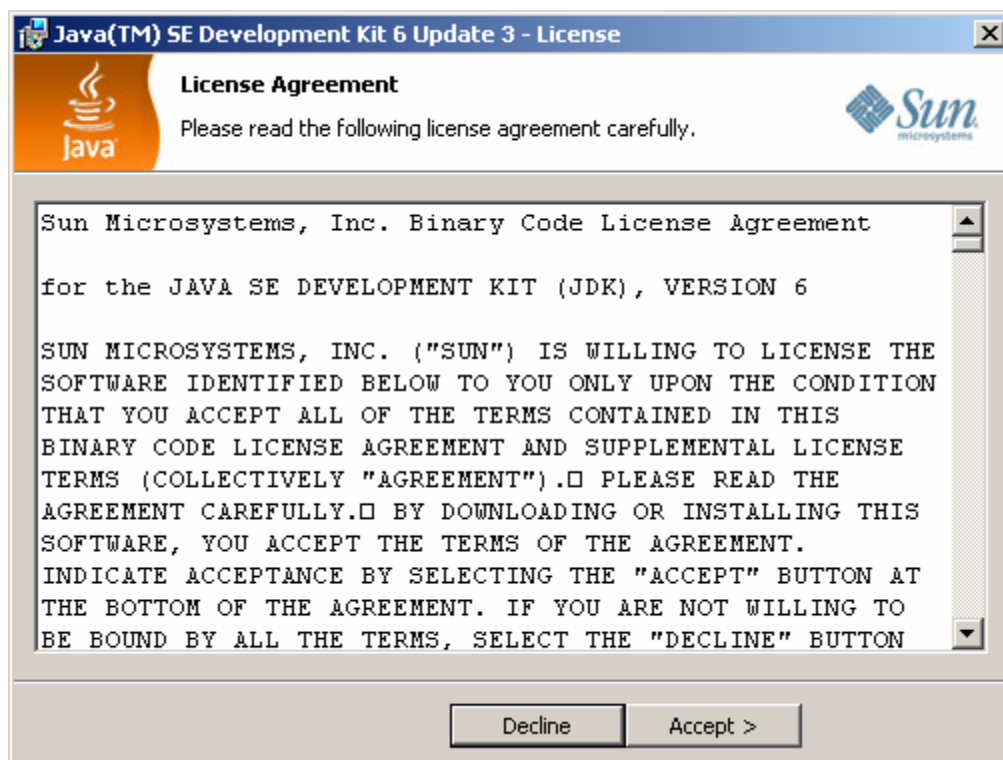
The steps to install SDK are the following:

2.2.1.1.- Licence agreement

When you begin the installer, you have to wait until the button next is enable to click and show a window when you have to accept the licence agreement:

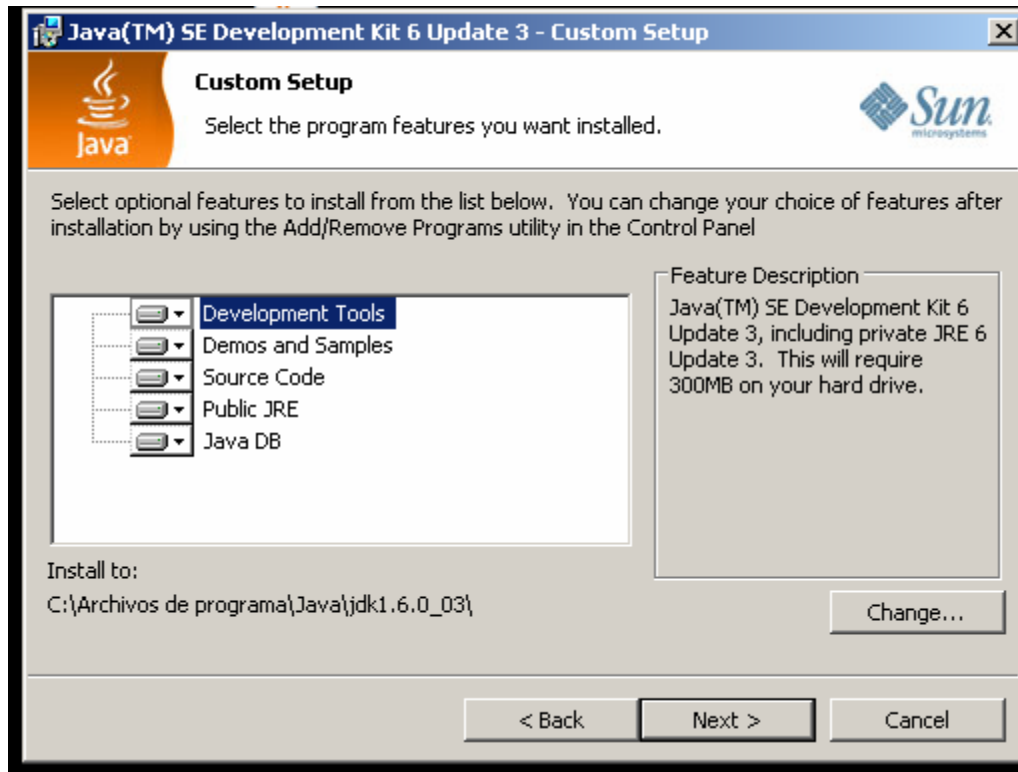


Click in next button when the button is enabled.

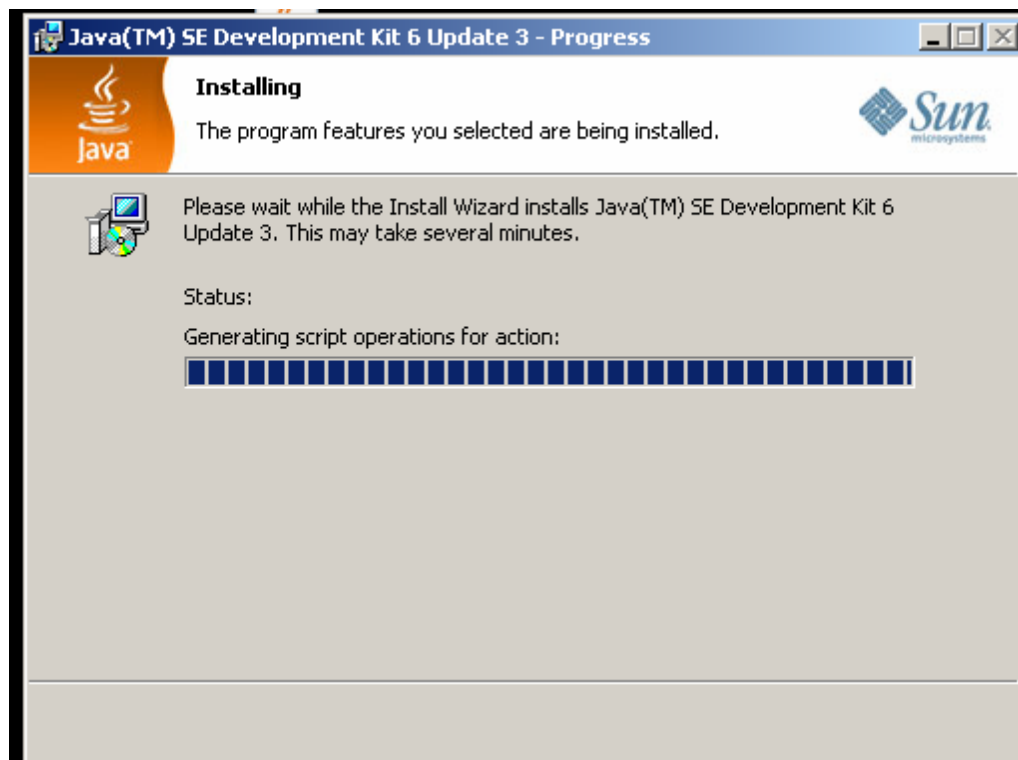


2.2.1.2.- Component Selection

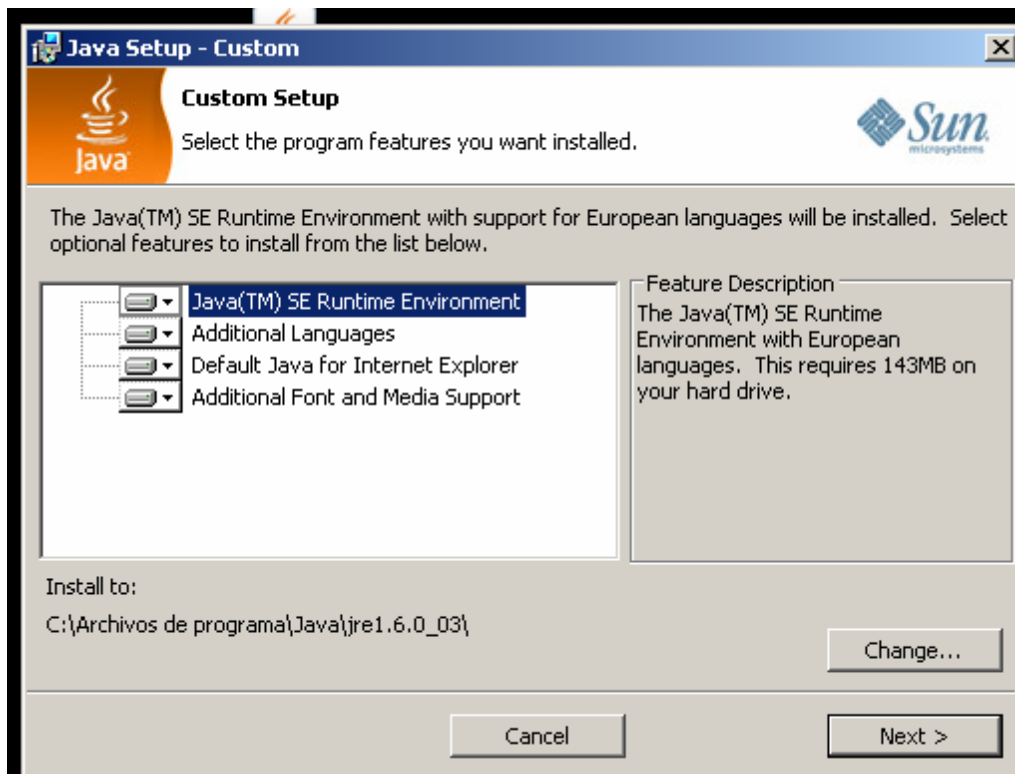
Choose all components to install and click in next button



Once you have chosen the components, the assistant will install them into your computer.



When the installer has finished with the component's installations, this one makes another question about optional features.



Select all components and click in next button to install the optional features into your computer.



When the installer finish this process, Java SDK will be installed on your computer.



Note: In this point of the installation, your computer is able to create any Java Program, test it.

2.2.2.- Checking your J2SE Installation

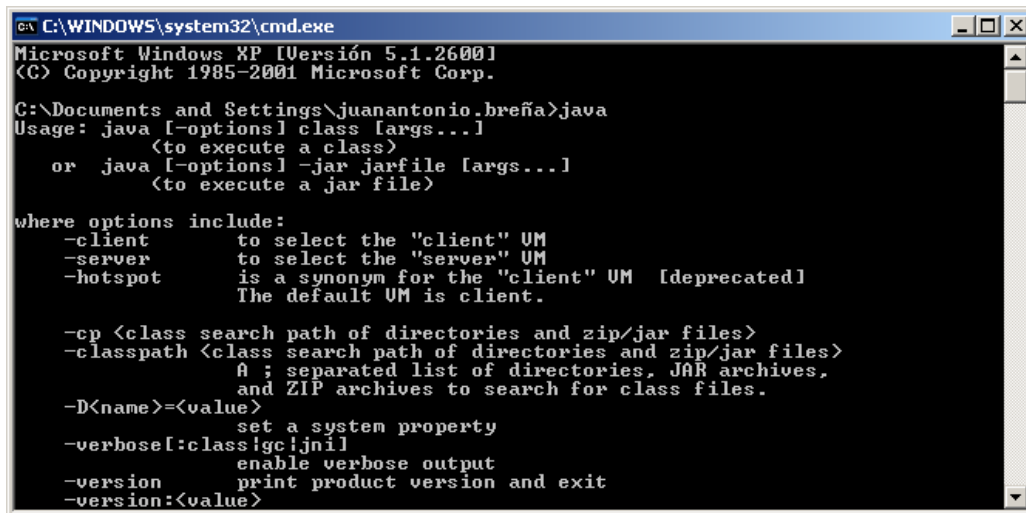
Once you have installed J2SE SDK on your computer, it is necessary to check that you can compile and execute Java programs.

Open a MS-DOS console on your computer and type the command Java:

- Java: Java command used to execute java programs.
- Javac: Java command used to compile a Java program

2.2.2.1.- Test1: Command java

The reason to make the first test is due to we need to check that your operating system recognize the command java which is used to execute java programs. If the console returns the options to use the command then the test is a success.



```

C:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [Versión 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\juanantonio.breña>java
Usage: java [-options] class [args...]
           (to execute a class)
    or java [-options] -jar jarfile [args...]
           (to execute a jar file)

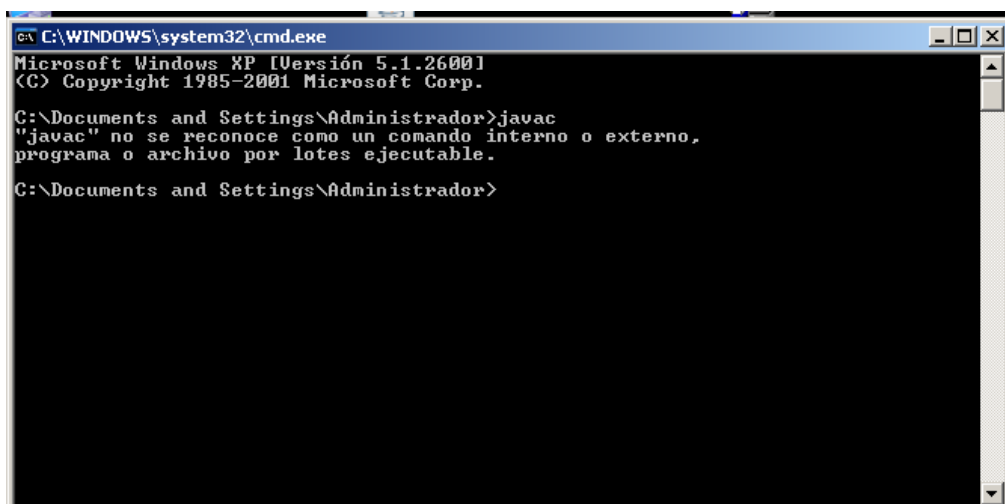
where options include:
    -client          to select the "client" VM
    -server          to select the "server" VM
    -hotspot         is a synonym for the "client" VM [deprecated]
                    The default VM is client.

    -cp <class search path of directories and zip/jar files>
    -classpath <class search path of directories and zip/jar files>
                A ; separated list of directories, JAR archives,
                and ZIP archives to search for class files.
    -D<name>=<value> set a system property
    -verbose[:class[:gc[:jni]] enable verbose output
    -version          print product version and exit
    -version:<value>

```

2.2.2.2.- Test2: Command javac

The second test is necessary to know if your operating system recognizes the command `javac` which is used to compile your programs. Type `javac` on your console and see the message.



```

C:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [Versión 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

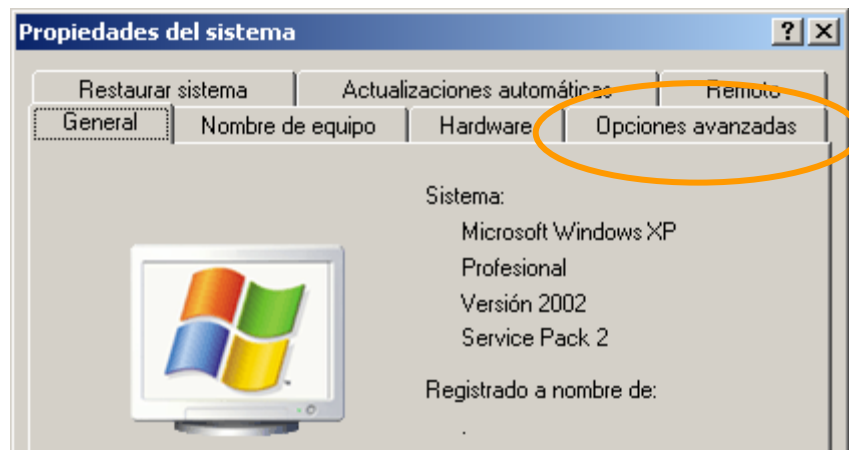
C:\Documents and Settings\Administrador>javac
"javac" no se reconoce como un comando interno o externo,
programa o archivo por lotes ejecutable.

C:\Documents and Settings\Administrador>

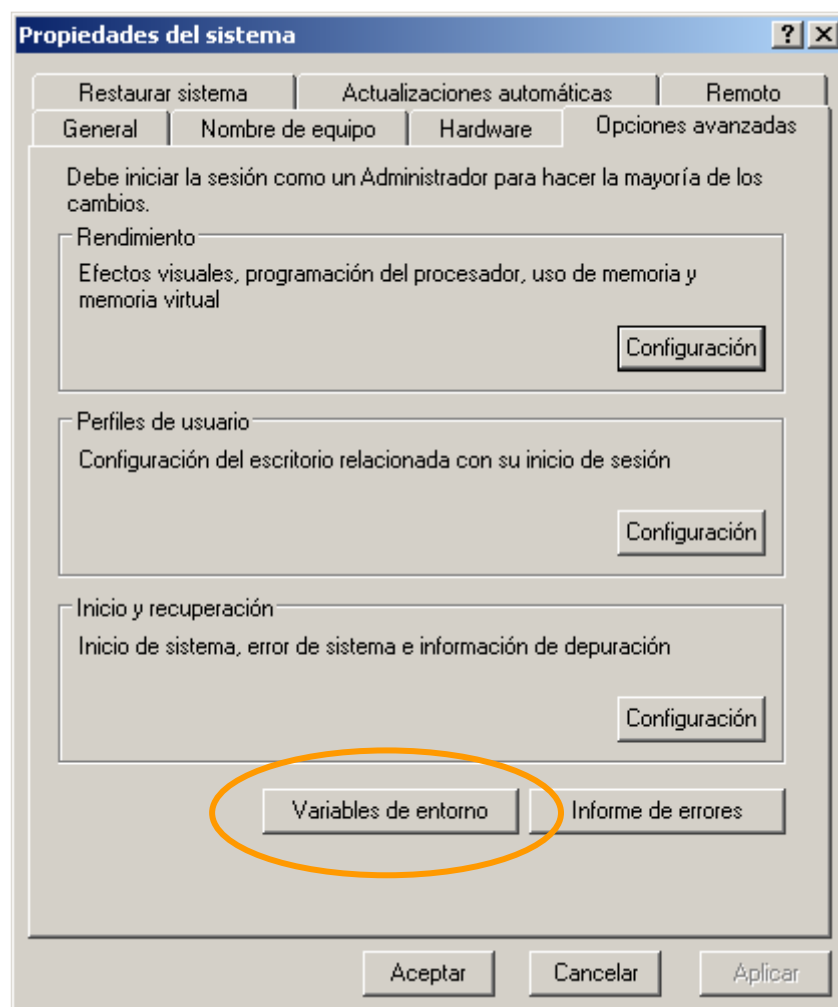
```

If your system says that your command doesn't understand the command, then you have to update environment variables in your system.

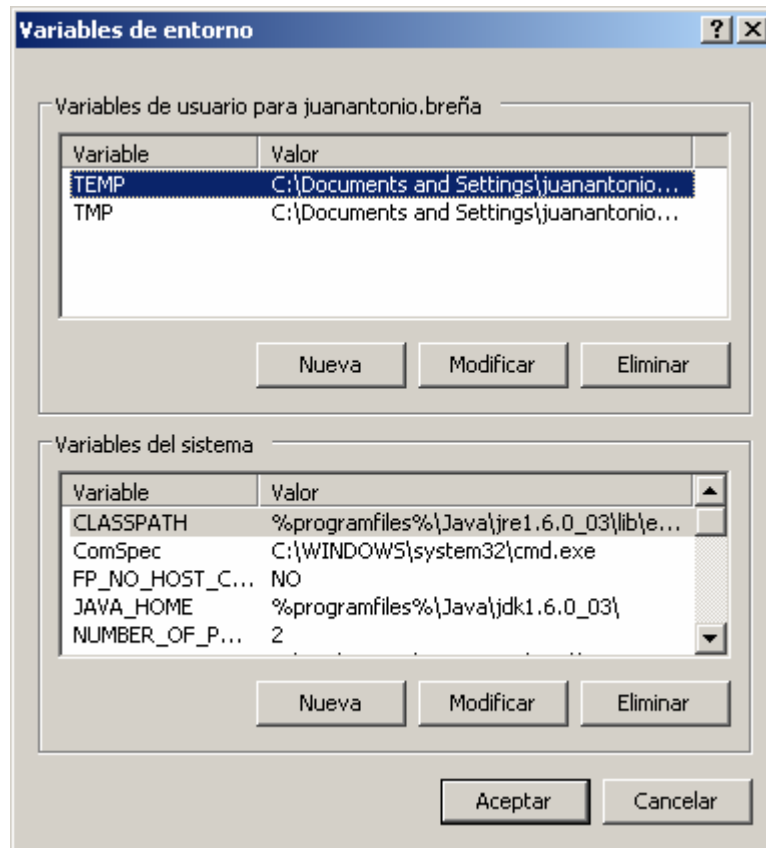
Right click in your PC Icon and select properties. Click in the tab "advanced options"



On the tab "Advanced options" click in the button "Environment variables"



When you click on this window you will see a new window where you will have to update the variable path. This path is really critic because you say windows what command you could execute directly from a Shell console.

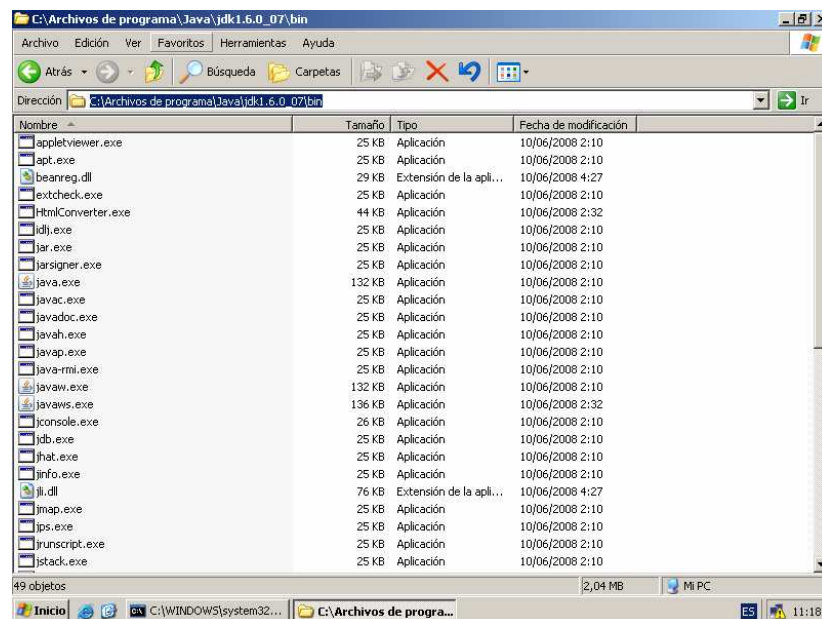


Path variable is located on System variables area. Find the variable path and click and update button. Path variable could have many statements due to it is used by several applications. In a new clean system, you should have the following content:

path:

`%SystemRoot%\system32;%SystemRoot%;%SystemRoot%\System32\Wbem`

To update path variable, find on your computer where is located J2SE SDK.



In this case, the path is:

`C:\Archivos de programa\Java\jdk1.6.0_07\bin\;`

Once you know the path, add the path at the end of the content of the system variable path:

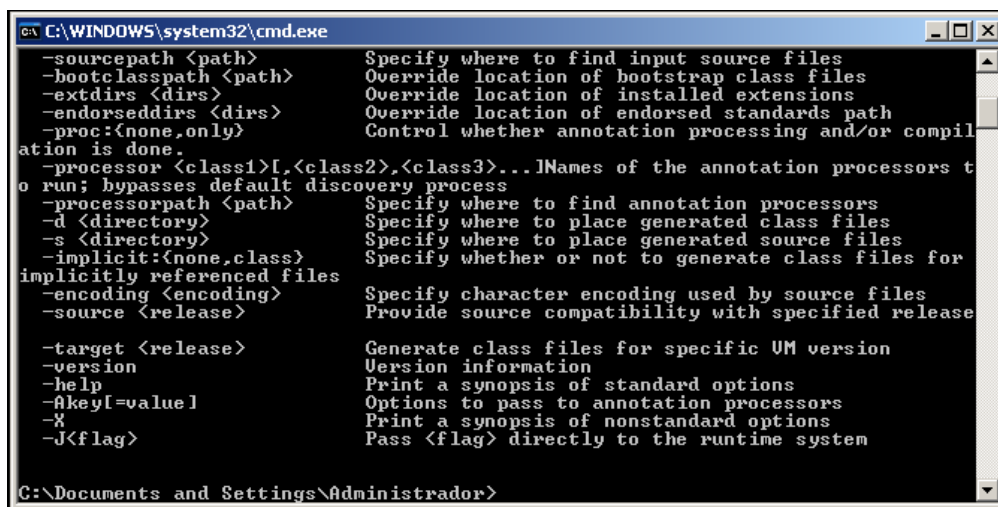
`%programfiles%\Java\jdk1.6.0_07\bin\;`

And the result is:

New path:

`%SystemRoot%\system32;%SystemRoot%;%SystemRoot%\System32\Wbem;
%programfiles%\Java\jdk1.6.0_07\bin\;`

Once you have made the changes, **reboot** the system and check again the command `javac`.



```

C:\WINDOWS\system32\cmd.exe
-sourcepath <path>          Specify where to find input source files
-bootclasspath <path>      Override location of bootstrap class files
-extdirs <dirs>            Override location of installed extensions
-endorseddirs <dirs>      Override location of endorsed standards path
-processor <none,only>     Control whether annotation processing and/or compilation is done.
-processor <class1>[,<class2>,<class3>...]Names of the annotation processors to run; bypasses default discovery process
-processorpath <path>      Specify where to find annotation processors
-d <directory>            Specify where to place generated class files
-s <directory>            Specify where to place generated source files
-implicit:<none,class>    Specify whether or not to generate class files for implicitly referenced files
-encoding <encoding>      Specify character encoding used by source files
-source <release>         Provide source compatibility with specified release
-target <release>         Generate class files for specific VM version
-version                  Version information
-help                    Print a synopsis of standard options
-Akey[=value]            Options to pass to annotation processors
-X                        Print a synopsis of nonstandard options
-J<flag>                 Pass <flag> directly to the runtime system

C:\Documents and Settings\Administrador>

```

If you notice that you see the options for the command `javac`, then the test was a success and you have finished the installation. Now your computer is able to develop Java Software.

2.2.3.- Installing Java ME Wireless Toolkit

Java ME Wireless Toolkit is used to develop Java ME Software. Download latest release from: <http://java.sun.com/javame/downloads/index.jsp>

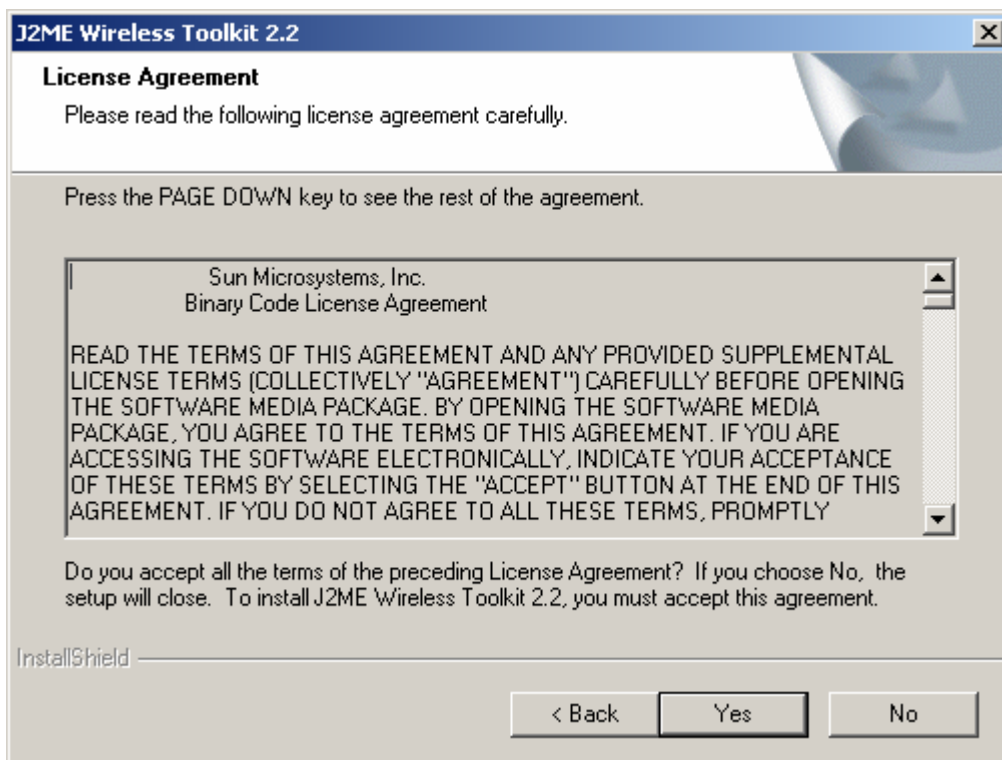
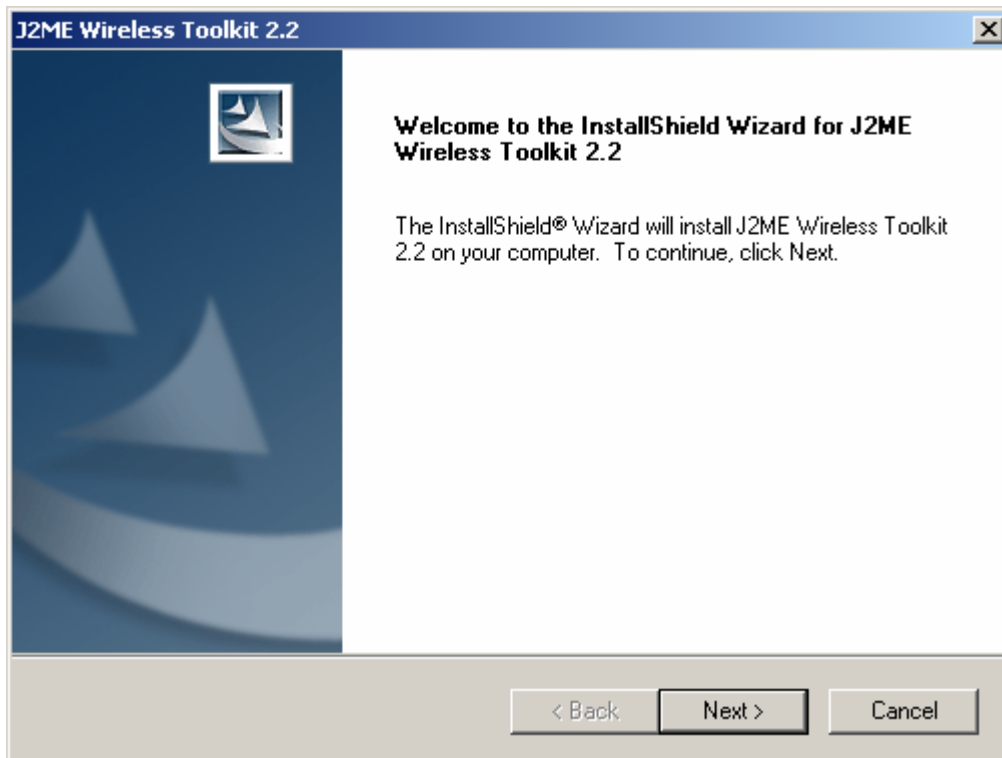
Note:

When I wrote this document I used: `j2me_wireless_toolkit-2_2-windows.exe`

Once you have saved the toolkit execute the installer and follow the steps.

2.2.3.1.- Licence agreement

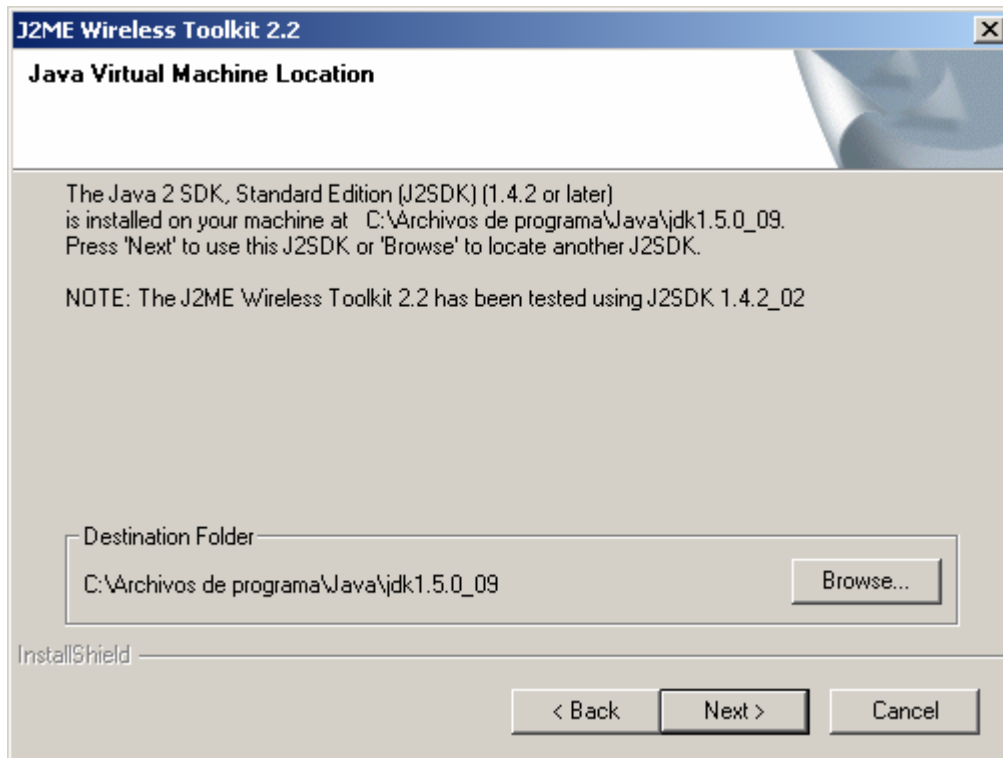
When you init the installer, click in next button to show the licence agreement.



Accept the agreement and click in yes button.

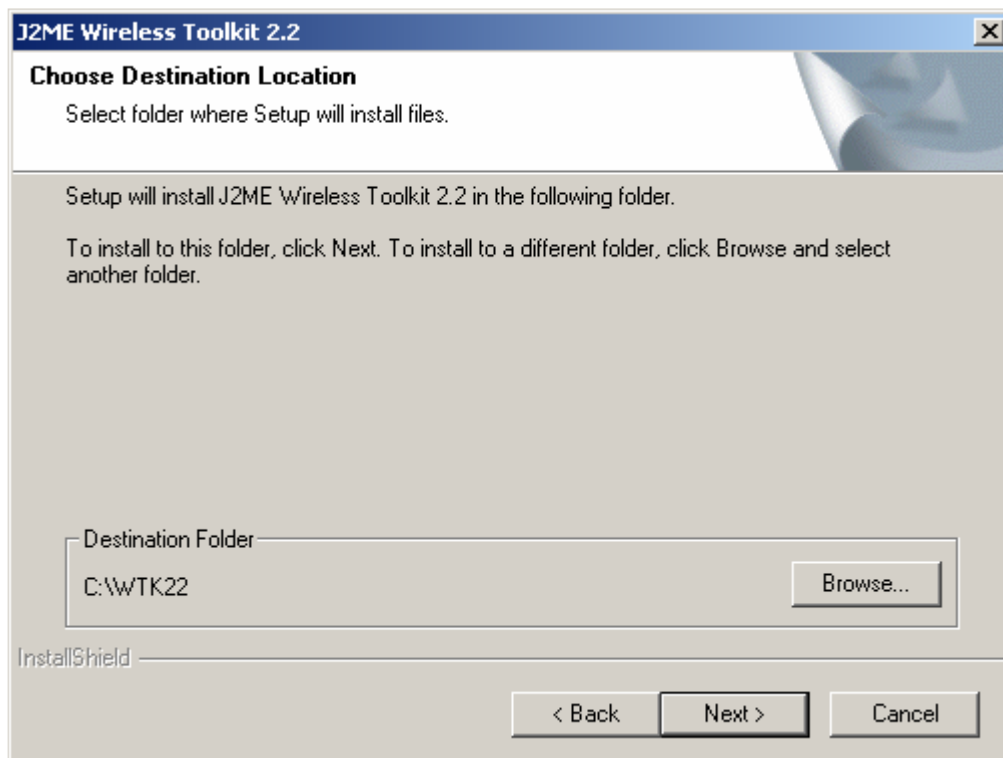
2.2.3.2.- JDK Detection

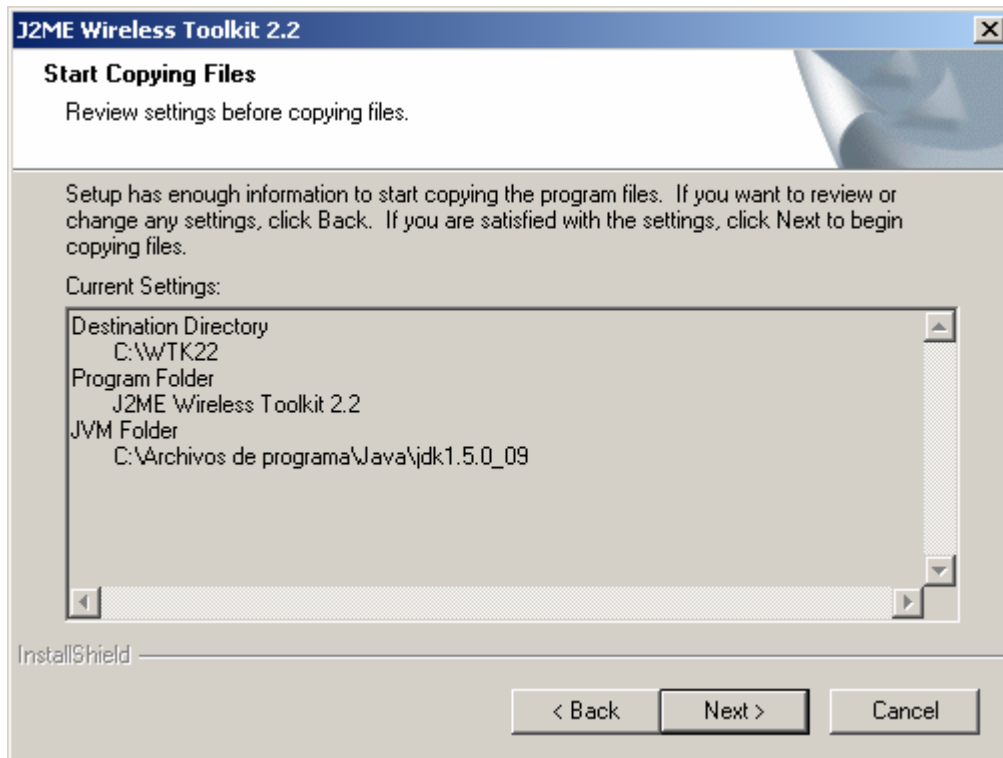
In some computers, it is possible that you have installed several J2SE SDK, then you have to select the SDK which you will link with the toolkit.



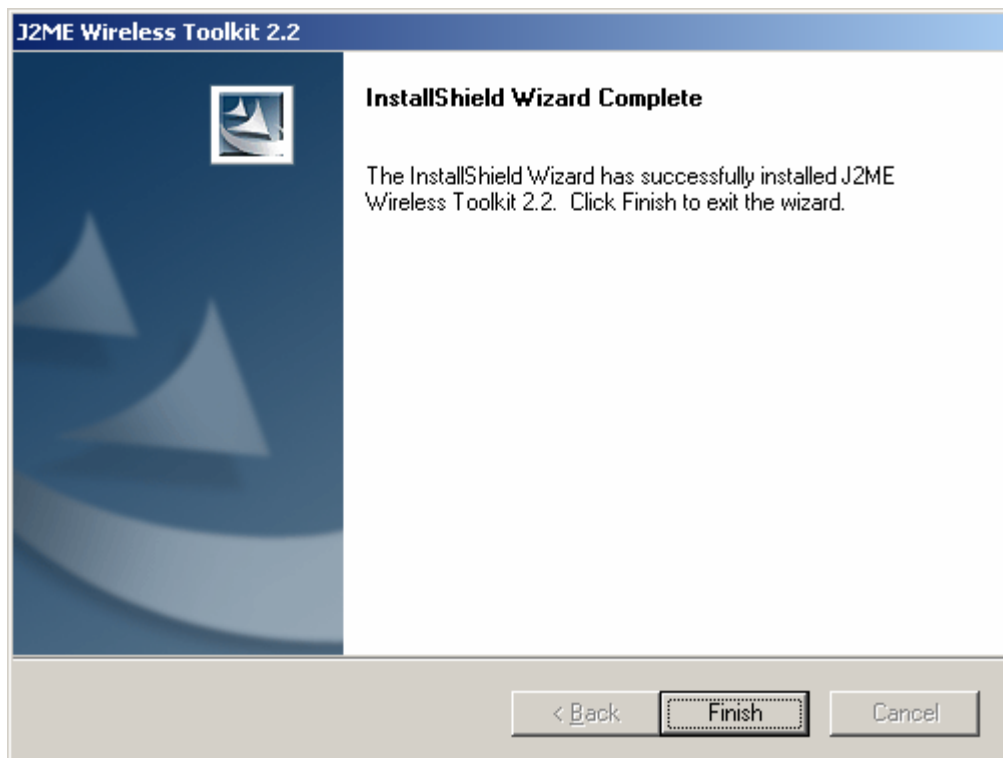
2.2.3.3.- Choose where you want to install the toolkit

Choose the place where you are going to install the toolkit. The installer gives you a recommendation if you have any suggestion, install the toolkit in the default path.





Check all parameters and click in next button to install the toolkit.



Now you have installed the Java ME Toolkit. In following points you will learn how to install a IDE to develop easily your Java ME Software

2.3.- Install IDE

2.3.1.- Installing Eclipse IDE

2.3.1.1.- Introduction

Eclipse is an extensible, open source IDE (integrated development environment). The project was originally launched in November 2001, when IBM donated \$40 million worth of source code from Websphere Studio Workbench and formed the Eclipse Consortium to manage the continued development of the tool.

The stated goals of Eclipse are "to develop a robust, full-featured, commercial-quality industry platform for the development of highly integrated tools." To that end, the Eclipse Consortium has been focused on three major projects:

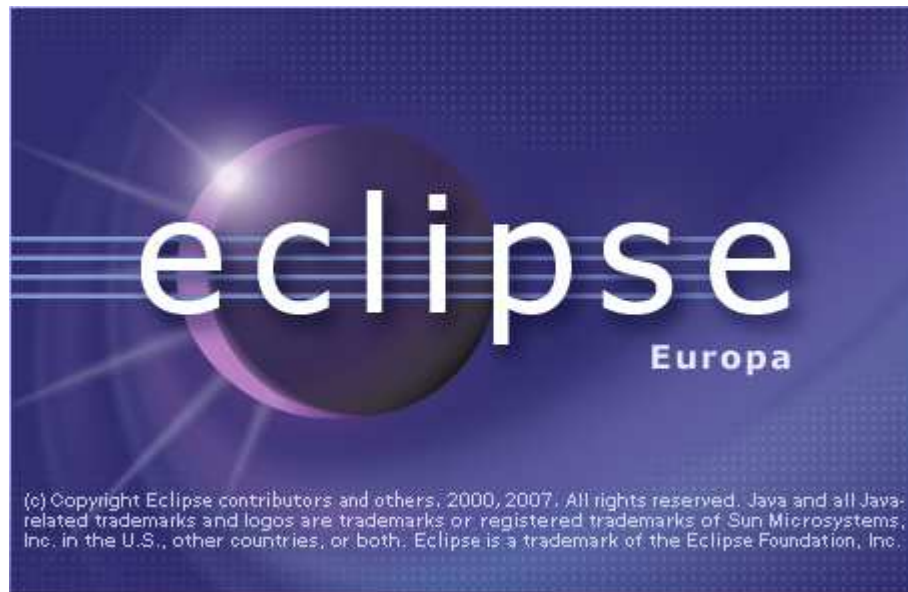
1. The Eclipse Project is responsible for developing the Eclipse IDE workbench (the "platform" for hosting Eclipse tools), the Java Development Tools (JDT), and the Plug-In Development Environment (PDE) used to extend the platform.
2. The Eclipse Tools Project is focused on creating best-of-breed tools for the Eclipse platform. Current subprojects include a Cobol IDE, a C/C++ IDE, and an EMF modeling tool.
3. The Eclipse Technology Project focuses on technology research, incubation, and education using the Eclipse platform.

Download Eclipse Europa 3.3 Classic from eclipse's website. Use the following URL: <http://www.eclipse.org/downloads/>

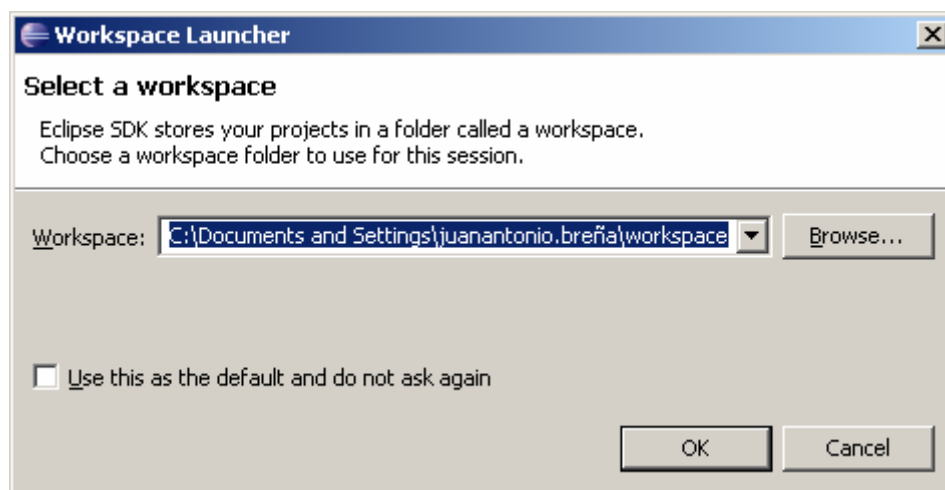
This software doesn't have any installer. Unzip latest release and execute eclipse.exe to run Eclipse IDE.

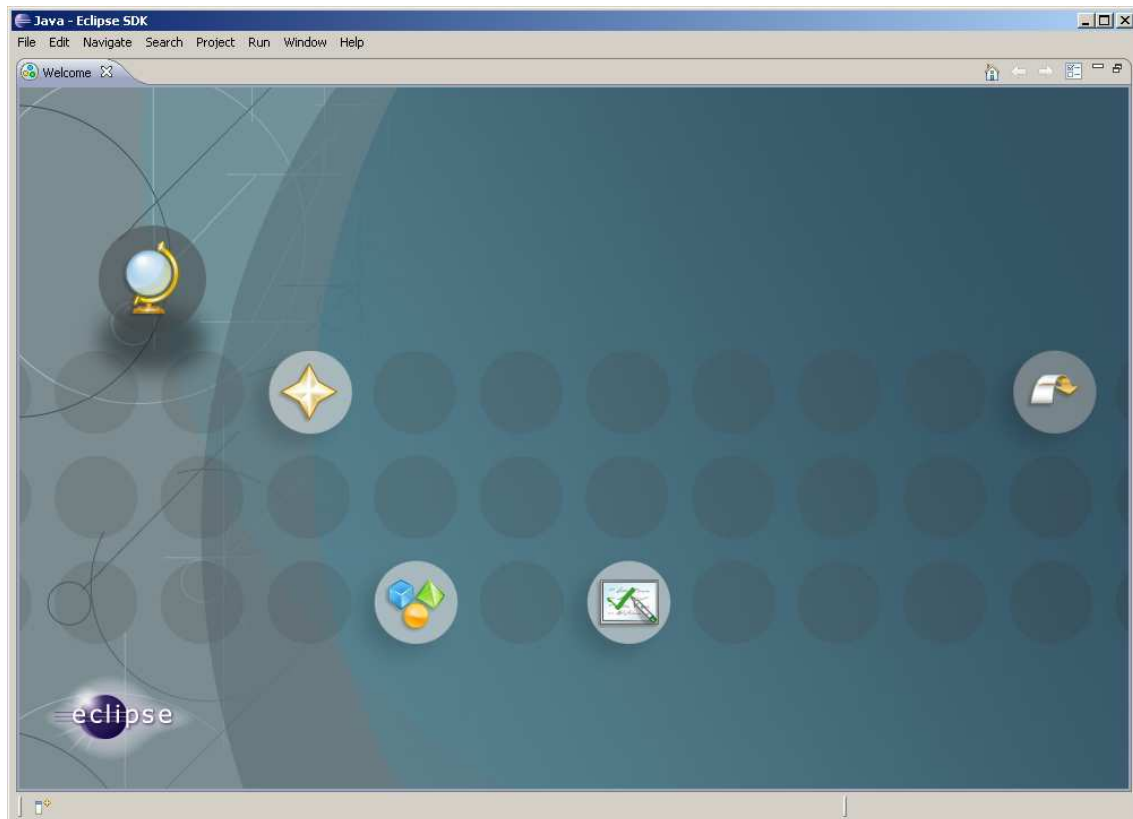


Every time that you execute Eclipse IDE, you will see the following window:



The first time that you use Eclipse, you have to determinate your workspace directory:





2.3.2.- Installing Eclipse ME Plugin

EclipseME is an Eclipse plugin to help develop J2ME MIDlets. EclipseME does the "grunt work" of connecting Wireless Toolkits to the Eclipse development environment, allowing you to focus on developing your application, rather than worrying about the special needs of J2ME development.

Note: This document used the following release:

<http://download.eclipse.org/dsdp/mtj/updates/0.9/stable/>

If you need to install the plugin Eclipse ME, one way to install is using the update manager from eclipse.

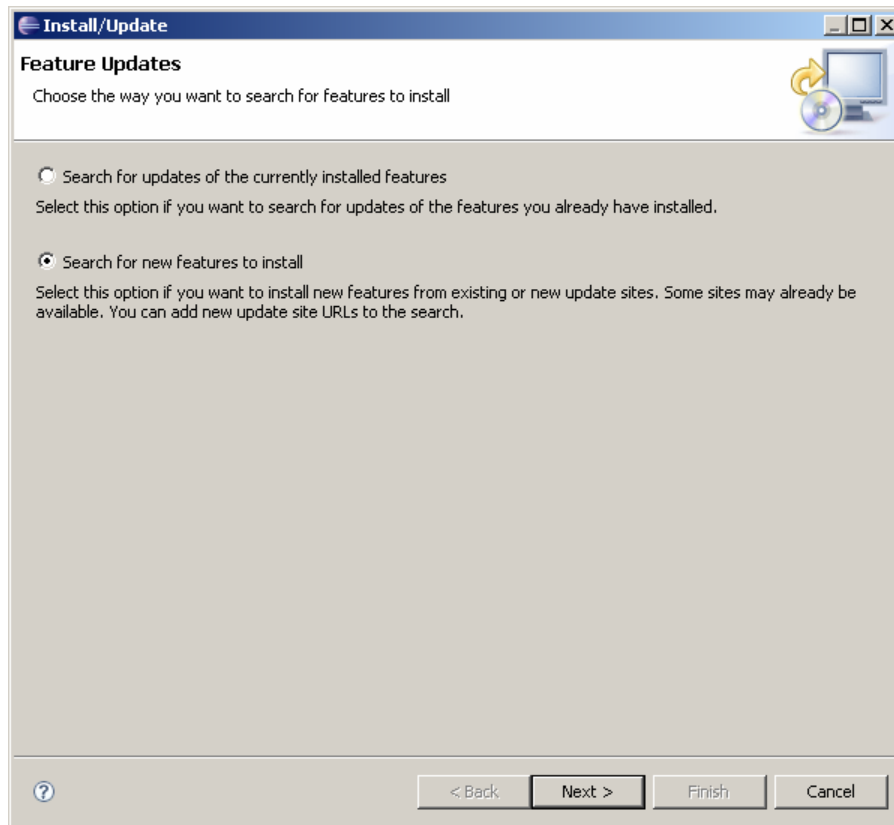
2.3.2.1.- Installation Process with Update Manager

Open Eclipse IDE and click in the main menu on the option:

Help > Software Updates > Find and Install



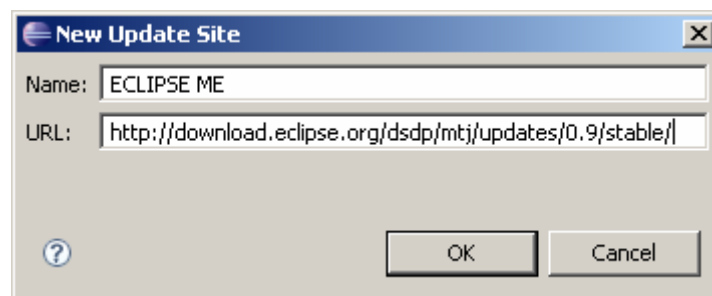
Then you will see the following assistant which will help you if you need to increase Eclipse's features. In this case you will use this assistant to install Eclipse ME Plugin.



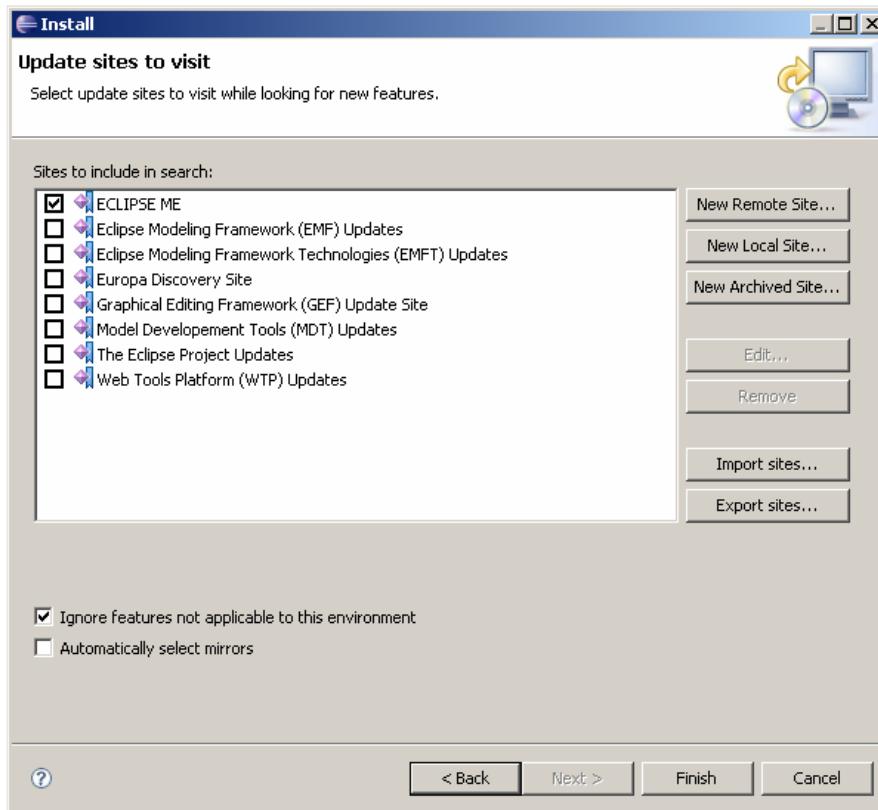
Select the option "Search for new features to install" and click in the button Next.

The URL where is located the latest release is:

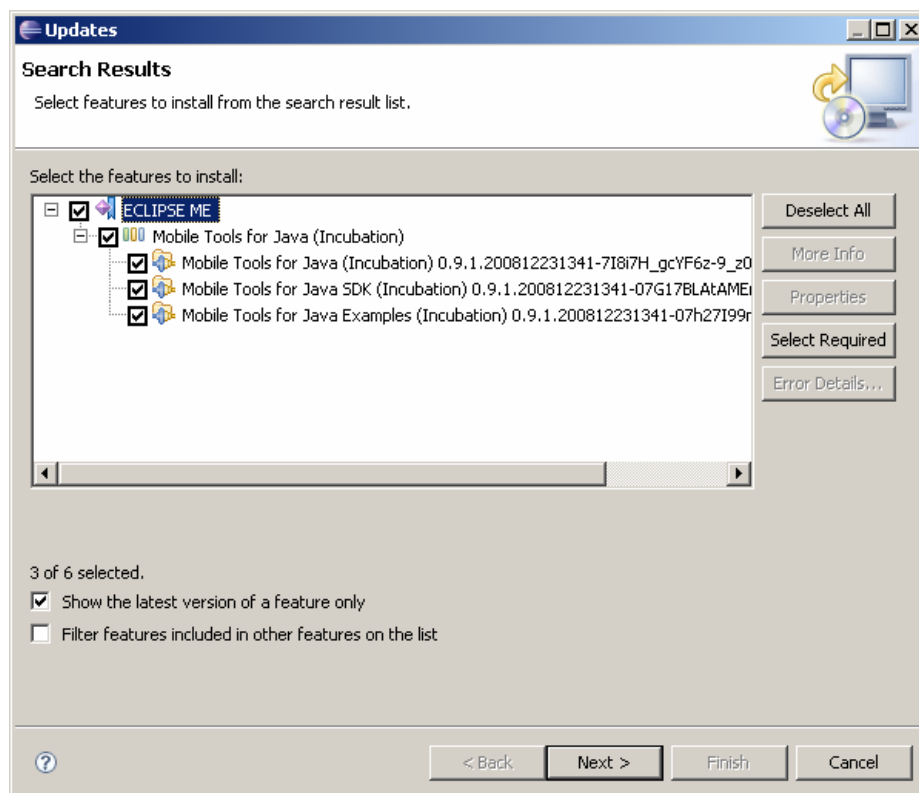
<http://download.eclipse.org/dsdp/mtj/updates/0.9/stable/>



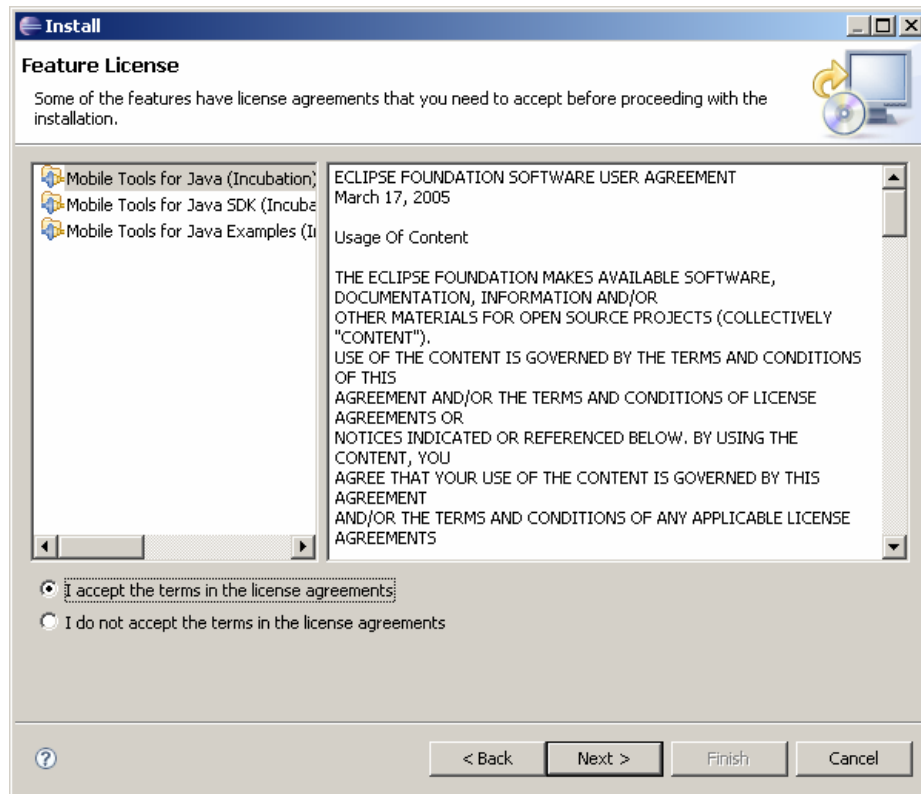
Once you have typed the Name and URL, click in the button Ok to show a new window with places where you can search to find plugins for Eclipse:



Click in the button Finish. In this moment Eclipse search in the selected options, in this case in the option Eclipse ME and Eclipse will find the following software to install:



Select all options and click in the button Next.



Accept the licence to install the plugin and click in the button Next. In this moment, Eclipse will install the plugin. Reboot eclipse to take effect the changes.

Now you have a Computer where you can develop Java ME Software.

2.4.- Install Optional Software

When you develop Java ME software, it is necessary to optimize the code and make Unit tests. Proguard and Antenna are a Open Source projects developed to cover this needs.

2.4.1.- Proguard

ProGuard is a free Java class file shrinker, optimizer, obfuscator, and preverifier. It detects and removes unused classes, fields, methods, and attributes. It optimizes bytecode and removes unused instructions. It renames the remaining classes, fields, and methods using short meaningless names. Finally, it preverifies the processed code for Java 6 or for Java Micro Edition.

Some uses of ProGuard are:

- Creating more compact code, for smaller code archives, faster transfer across networks, faster loading, and smaller memory footprints.
- Making programs and libraries harder to reverse-engineer.
- Listing dead code, so it can be removed from the source code.
- Retargeting and preverifying existing class files for Java 6, to take full advantage of Java 6's faster class loading.

You can visit the project in the following URL to download latest release
<http://proguard.sourceforge.net/>

http://sourceforge.net/project/showfiles.php?group_id=54750

Note:

This document used Proguard 4.3

Store this libraries in a unique location, for example:

D:\DATA\PROJECTS\ROBOTICS\J2ME\proguard4.3

When we create a new Java ME project, we will use Proguard.

2.4.2.- Antenna

Antenna provides a set of Ant tasks suitable for developing wireless Java applications targeted at the Mobile Information Device Profile (MIDP). With Antenna, you can compile, preverify, package, obfuscate, and run your MIDP applications (aka MIDlets), manipulate Java Application Descriptor (JAD) files, as well as convert JAR files to PRC files designed to run on the MIDP for PalmOS implementations from Sun and IBM. Deployment is supported via a deployment task and a corresponding HTTP servlet for Over-the-Air (OTA) provisioning. A small preprocessor allows to generate different variants of a MIDlet from a single source

You can visit the project in the following URL to download latest release

<http://antenna.sourceforge.net/>

http://sourceforge.net/project/showfiles.php?group_id=67420

Note:

This document used Antenna1.1

Store this libraries in a unique location, for example:

D:\DATA\PROJECTS\ROBOTICS\J2ME

When we create a new Java ME project, we will use Antenna.

3.- Creating your first Java ME project

3.1.- Introduction

In this chapter we will create our first Java ME project. We learn the following topics:

- Create a Java ME project with Eclipse
- Test the software with a simulator
- Deliver your project on your mobile

3.2.- Creating a new Java ME Project

Open Eclipse and click in the option:

File > New > Midlet Project



Then you will see an assistant to define some parameters:

New MIDlet Project

Create a MIDlet Project
Create a MIDlet project in the workspace or in an external location.

Project name: HelloWorld

Application Descriptor
Name to be used for the jad file, generated during the "Create Package" process:
☒ Use project name as filename
☐ Use custom jad file name
 Jad filename:
 HelloWorld.jad

Contents
☐ Create new project in workspace
☒ Create project from existing source
 Directory: D:\DATA\PROJECTS\ROBOTICS\J2ME\ECLIPSEME\HelloWorld Browse...

Configurations
You can add more configurations here:

active	Configuration
<input checked="" type="checkbox"/>	DefaultColorPhone
<input type="checkbox"/>	
<input type="checkbox"/>	
<input type="checkbox"/>	
<input type="checkbox"/>	
<input type="checkbox"/>	
<input type="checkbox"/>	
<input type="checkbox"/>	

Add... Edit... Remove

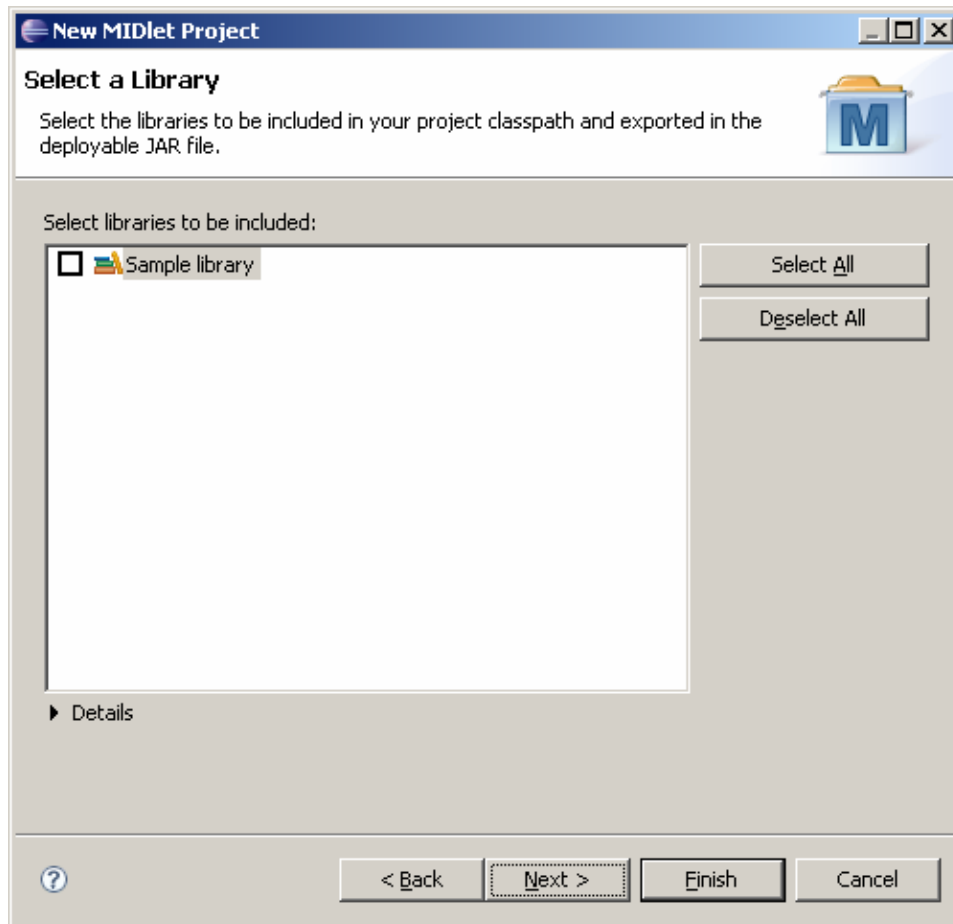
Preprocessor
☐ Enable Preprocessing Support

< Back Next > Finish Cancel

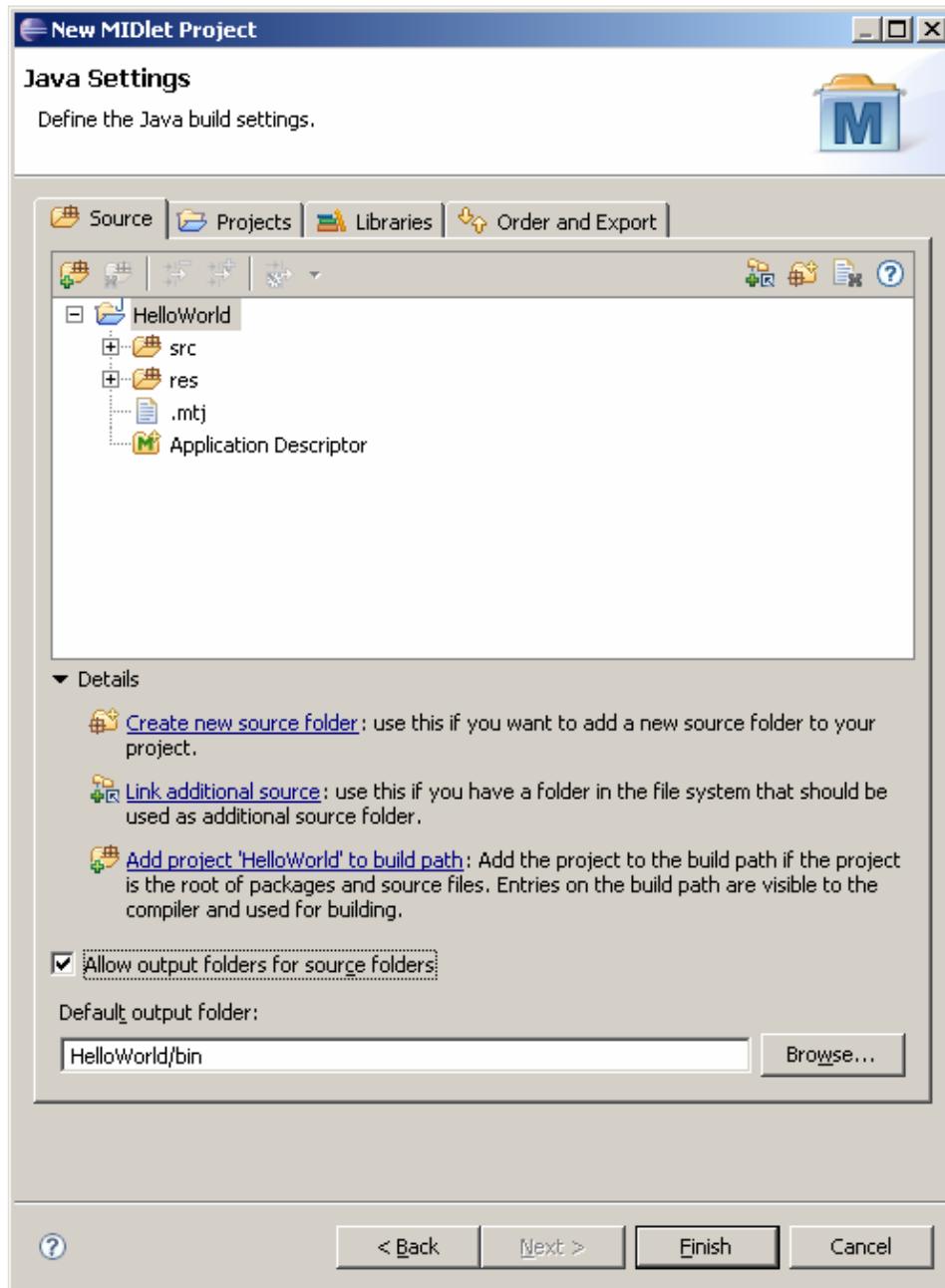
The parameters to define are:

- Project name
- Place where you are going to store the source code and bytecodes to send to your mobile phone

Click in the button Next to select a librarie used in the new project. In this example we will not include any library.



Click in next button to define others parameters:



Click in the option "Allow output folders for source folders" and click in the button Finish.

3.3.- Java ME Settings

Eclipse ME Plugin offers a control panel to define many parameters. This panel will save many hours when you have to debug your projects on your real mobile phones.

Required Information

This section describes required information about this application.

MIDlet Jar URL:

MIDlet Name:

MIDlet Vendor:

MIDlet Version:

Microedition Configuration:

Microedition Profile:

Running

Run your application within a Java ME device:

Launch as emulated Java ME MIDlet

Launch as emulated Java ME JAD

Debugging

Debug your application within a Java ME device:

Launch as emulated Java ME MIDlet in Debug mode

Launch as emulated Java ME JAD in Debug mode

Packaging

Package your application:

Create package

Create obfuscated package

Exporting

Generate your Buildfiles based on the configuration of the MIDlet Project:

Export Antenna Buildfiles

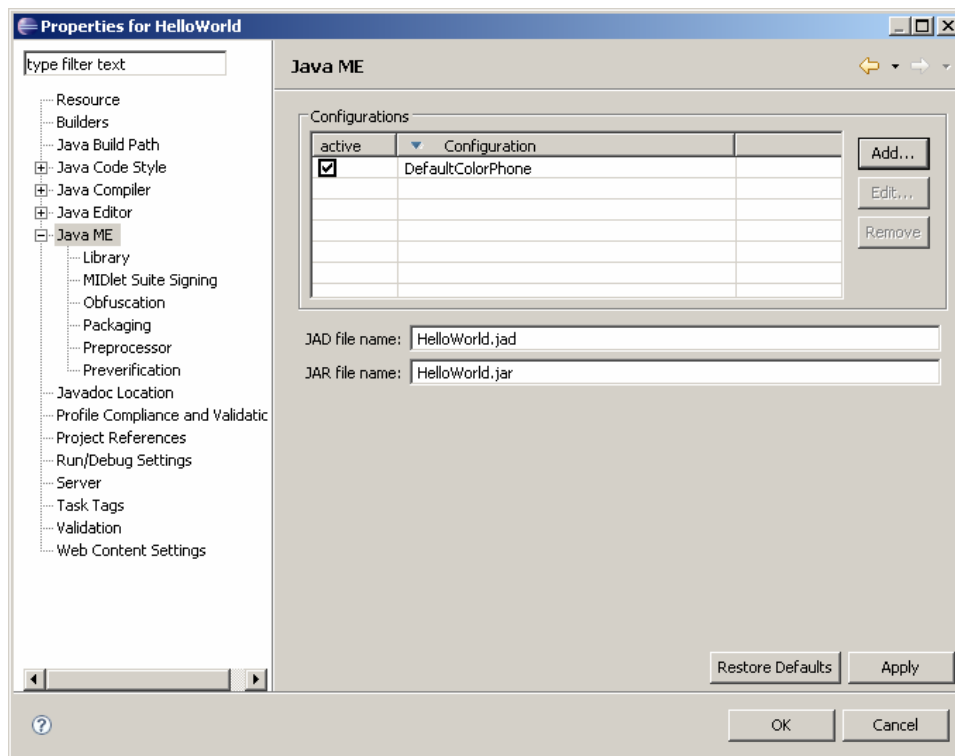
Runtime

Specify the execution environments to run this Project.

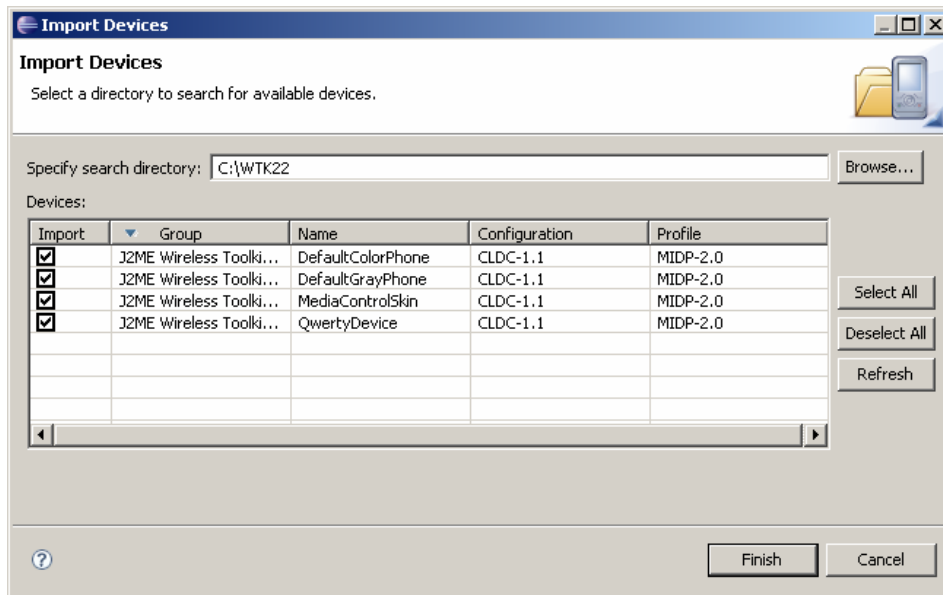
active	Configuration
<input checked="" type="checkbox"/>	DefaultColorPhone
<input type="checkbox"/>	
<input type="checkbox"/>	
<input type="checkbox"/>	
<input type="checkbox"/>	
<input type="checkbox"/>	
<input type="checkbox"/>	
<input type="checkbox"/>	

In general, you can configure Java ME Settings if you click:

Project > Properties

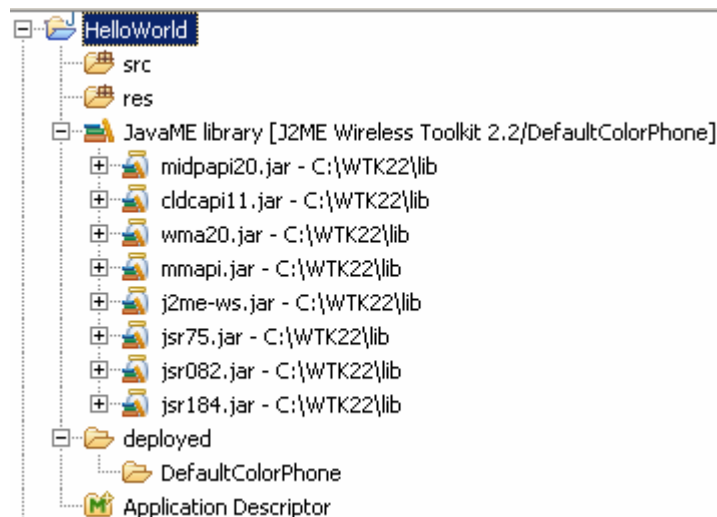


If you need to add Mobile profiles, click in Add to import profiles from Wireless Toolkit:



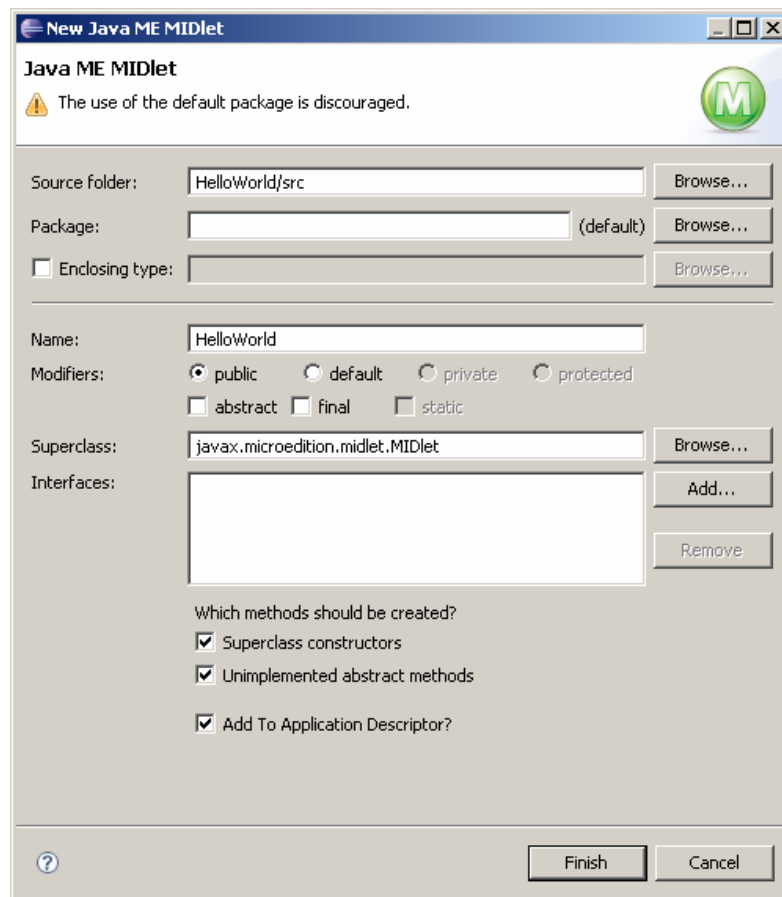
3.4.- Add a HelloWorld Middlet

For the moment, your project is empty. Add the middlet HelloWorld in the project.



select the folder src and right click to select:

new > new > Java ME Middlet



Click in the button Finish to create a new Middle in the project.

HelloWorld.java

```
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;

/**
 *
 */

public class HelloWorld extends MIDlet implements CommandListener {

    private Command exitCommand; // The exit command
    private Display display;      // The display for this MIDlet

    public HelloWorld() {
        display = Display.getDisplay(this);
        exitCommand = new Command("Exit", Command.EXIT, 0);
    }

    public void startApp() {
        TextBox t = new TextBox("Hello", "Hello, World!", 256, 0);
        t.addCommand(exitCommand);
        t.setCommandListener(this);
        display.setCurrent(t);
    }

    public void pauseApp() {
    }
}
```

```

public void destroyApp(boolean unconditional) {
}

public void commandAction(Command c, Displayable s) {
    if (c == exitCommand) {
        destroyApp(false);
        notifyDestroyed();
    }
}
}

```

3.5.- Test your Middlet with a simulator

Once you have code your first Middlet, you need to test in your computer. When you defined the project, you indicated that you designed a Java ME solution to a Color Mobile phone but you can add more Devices.

Required Information

This section describes required information about this application.

MIDlet Jar URL: HelloWorld.jar

MIDlet Name: HelloWorld MIDlet Suite

MIDlet Vendor: MIDlet Suite Vendor

MIDlet Version: 1.0.0

Microedition Configuration: Connected Limited Device Configuration (1.1)

Microedition Profile: Mobile Information Device Profile (2.0)

Packaging

Package your application:

Create package

Create obfuscated package

Exporting

Generate your Buildfiles based on the configuration of the MIDlet Project:

Export Antenna Buildfiles

Running

Run your application within a Java ME device:

Launch as emulated Java ME MIDlet

Launch as emulated Java ME JAD

Debugging

Debug your application within a Java ME device:

Launch as emulated Java ME MIDlet in Debug mode

Launch as emulated Java ME JAD in Debug mode

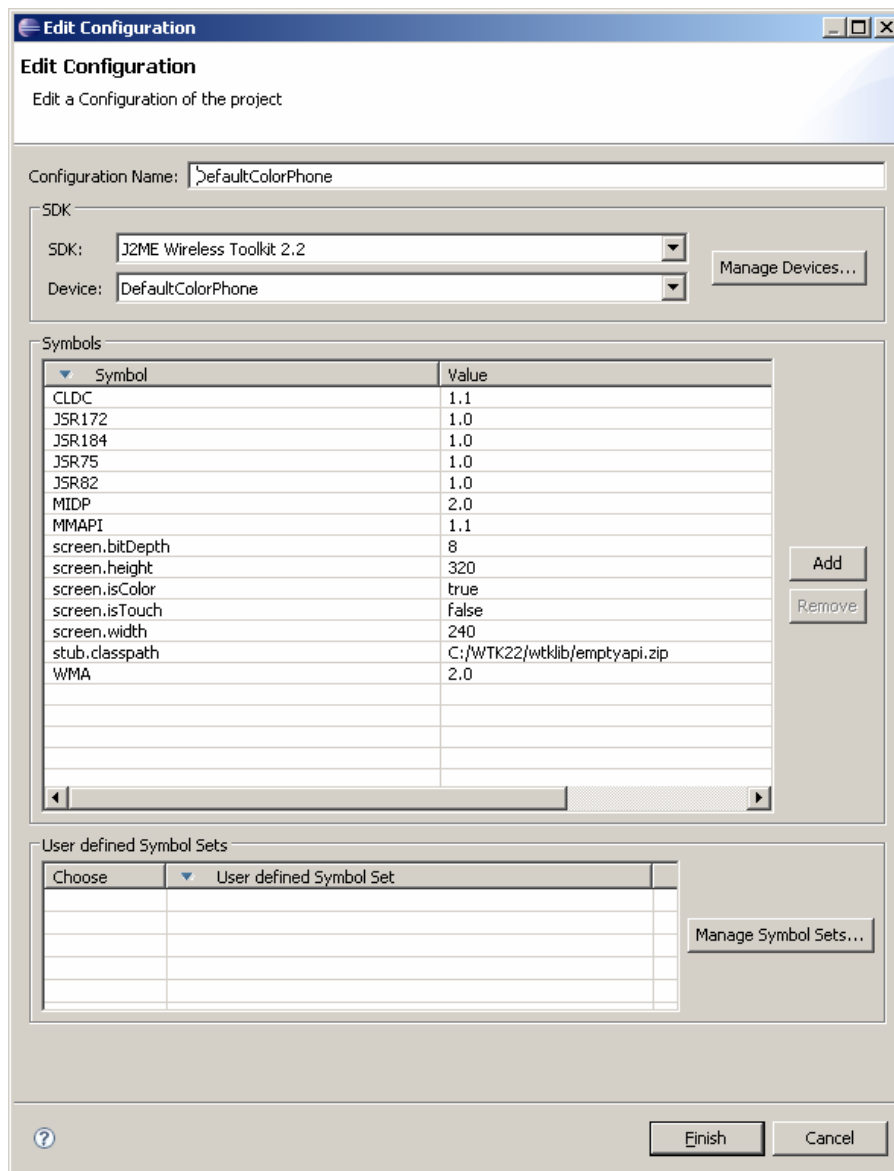
Runtime

Specify the execution environments to run this Project.

active	Configuration	
<input checked="" type="checkbox"/>	DefaultColorPhone	
<input type="checkbox"/>		
<input type="checkbox"/>		
<input type="checkbox"/>		
<input type="checkbox"/>		
<input type="checkbox"/>		

Add... Edit... Remove

If you select the Device, you can see the details:



To test your first Java ME project, go to the Control panel and click in the Link "Launch as emulated Java ME JAD"



Once you have clicked on the link then EclipseME will show a emulator running your program.



3.6.- Package your application

Once you consider that your Java ME application is finished, come back to the control panel and click in the link: Create package to create a package

Required Information

This section describes required information about this application.

MIDlet Jar URL: HelloWorld.jar

MIDlet Name: HelloWorld MIDlet Suite

MIDlet Vendor: MIDlet Suite Vendor

MIDlet Version: 1.0.0

Microedition Configuration: Connected Limited Device Configuration (1.1)

Microedition Profile: Mobile Information Device Profile (2.0)

Packaging

Package your application:

☒ Create package

☐ Create obfuscated package

Exporting

Generate your Buildfiles based on the configuration of the MIDlet Project:

☒ Export Antenna Buildfiles

Running

Run your application within a Java ME device:

☒ Launch as emulated Java ME MIDlet

☒ Launch as emulated Java ME JAD

Debugging

Debug your application within a Java ME device:

☒ Launch as emulated Java ME MIDlet in Debug mode

☒ Launch as emulated Java ME JAD in Debug mode

Runtime

Specify the execution environments to run this Project.

active	Configuration	
<input checked="" type="checkbox"/>	DefaultColorPhone	
<input type="checkbox"/>		
<input type="checkbox"/>		
<input type="checkbox"/>		
<input type="checkbox"/>		
<input type="checkbox"/>		

Add... Edit... Remove

Eclipse ME create a package and descriptor file (.JAD)

3.7.- Deliver your Middlet into a Mobile Phone

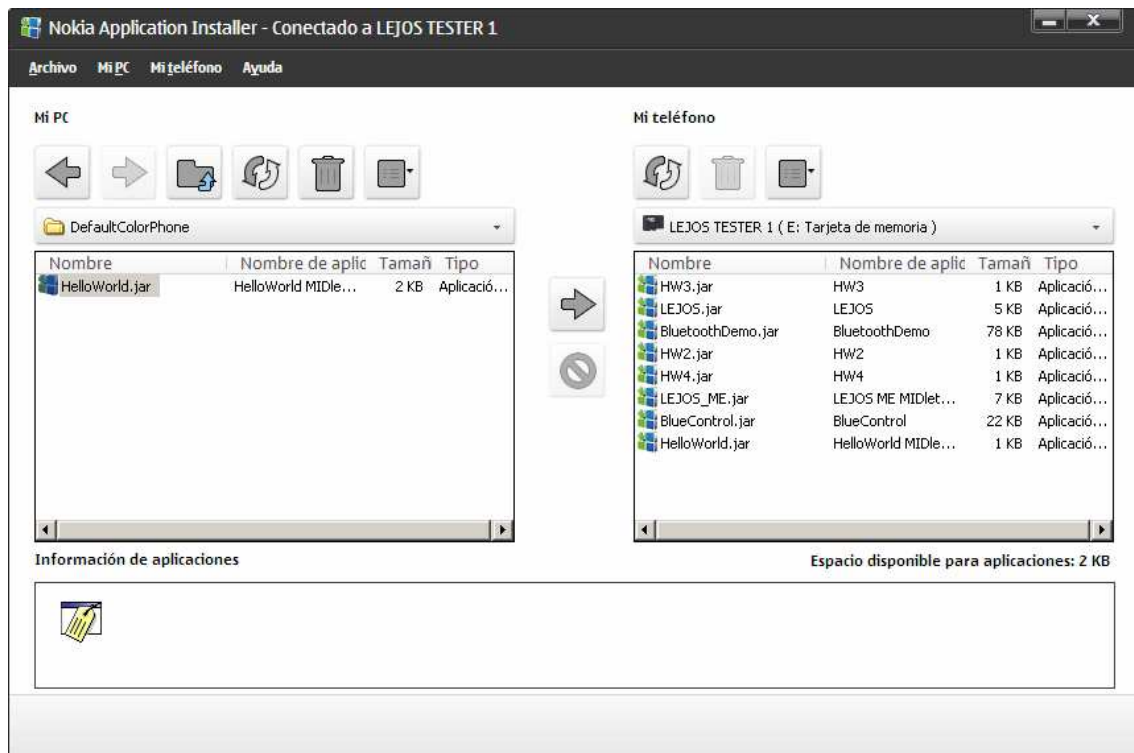
Once you have developed your Middlet application and you made several tests then you can deliver your software into a Mobile Phone.

3.7.1.- Deliver Java ME with Nokia PC Suite

If you have a Nokia Mobile Phone, Nokia has software to manage their mobiles. You can use PC Suite to send by Bluetooth your Java ME software. Open Nokia PC Suite. When PC Suite and click in the launcher for Nokia Application Installer.



Once you have launched Nokia Application Manager, you will find your application and send into your mobile phone.



3.7.1.1.- Problems with the delivery

Along the time, Mobile phones has evolved then some mobile phones will not run with your software if this one use latest Java ME features. When you deliver a mobile application, check the following parameters from the Control Panel:

Required Information
This section describes required information about this application.

MIDlet Jar URL: HelloWorld.jar
 MIDlet Name: HelloWorld MIDlet Suite
 MIDlet Vendor: MIDlet Suite Vendor
 MIDlet Version: 1.0.0
 Microedition Configuration: Connected Limited Device Configuration (1.1)
 Microedition Profile: Mobile Information Device Profile (2.0)

Running
Run your application within a Java ME device:
 Launch as emulated Java ME MIDlet
 Launch as emulated Java ME JAD

Debugging
Debug your application within a Java ME device:
 Launch as emulated Java ME MIDlet in Debug mode
 Launch as emulated Java ME JAD in Debug mode

Packaging
Package your application:
 Create package
 Create obfuscated package

Exporting
Generate your Buildfiles based on the configuration of the MIDlet Project:
 Export Antenna Buildfiles

Runtime
Specify the execution environments to run this Project.

active	Configuration
<input checked="" type="checkbox"/>	DefaultColorPhone
<input type="checkbox"/>	
<input type="checkbox"/>	
<input type="checkbox"/>	
<input type="checkbox"/>	

Add... Edit... Remove

- Microedition Configuration:
 - CLDC 1.0
 - CLDC 1.1
- Microedition Profile:
 - MIDP 1.0
 - MIDP 2.0
 - MIDP 2.1
 - IMP 1.0
 - IMP NG

Check the technical specifications for your mobile phone on Internet.