Department of Management Science and Technology

Software Comprehension and Maintenance

Wireless IRC project
Adding a new feature

Tzavala Polina
8020149

A. WRLIrc's Presentation

1.Project's Brief Presentation

WLIrc is an IRC Client for Java cell phones or any other device who support java MIDP 1.0.It can be downloaded to almost every mobile device from the following wap adress
http://thesverre.servebeer.com/wap/WLIrc.jad
Its main characteristics are the following:

- Development Status: 4-Beta
- License: GNU
- Programming Language: Java
- Topic: Internet Relay Chat
- Translations: English
- Activity Percentile (last week): 90.21%
- Administrator and developper: Sverre Kristian valskrå

2.Irc's Full description

WLIrc is based on IRC. Internet Relay Chat (IRC) is one of the most popular and most interactive services on the Internet that lets people all over the world participate in real-time conversations. It is based on a client program that helps us to connect to an IRC server.

An IRC server can have thousands of chat channels open simultaneously. Each channel has a specific topic. And many people can simultaneously participate in discussions over this topic.

When the user runs a "client" program a connection to a "server" in an IRC network opens. All servers are interconnected and pass messages from user to user over the IRC network using the ctcp protocol. One server can be connected to several other servers and up to hundreds of clients. Several larger and smaller IRC networks exist. The largest one, called EFnet (Eris Free net), usually serves over 15000 users at any given moment

3.WRLirc's Full description

WLIrc is Wireless Internet Relay Chat client meant for use with mobile phones supporting GPRS and MIDP 1.0 compliant Java programs.

GPRS is usually much cheaper and faster than text messages. Therefore, WLIrc can be a much cheaper substitute for SMS in the future. (There is an option for chatting in private with a specific person).
The project's main distinctive features are the following:

- WLIrc runs on any J2ME device.
- WLIrc supports multiple windows.

- WLIrc supports connection throught an http gateway for devices who don't support sockets.
- WLIrc saves configuration on the phone for later use.

4.WLIrc's History

- WLIrc was initially released in September 2003.
- Eversince 7 new versions have been issued fixing bugs and adding new commands .
- The last version was released on 7 January and made some ctcp and operator commands avalaible.

5.Adding   a new feature to WrLIrc

My contribution to wrlirc will be a new MyFavourites command to the main menu. The command will be similar to the Myfavourites command of the internet explorer.

The main functionality of this command will be saving a certain number of messages that need to be reused easily when chatting.
Every time the user sends a new message ,he will be asked to save it in his favourites. Then he will need only a click to resend it.The saved messages will be organised in a list .The user will be able to search for certain messages in the list and to delete certain messages from the list. Every time he looks for a message he will not have to scroll down the whole list. The message will be shown when typing its first letters.

B. Design of a new feature: WRLIrc's Architecture,Compiling and Code

1.An overview of Midlets

WRLIrc is a Midlet Suite. Midlets are small programs that are written in order to be used in mobile Devices. Midlets are build by JVM and are run and controlled by the Application Management Software . The application management software is part of the mobile device's software that deals with creating, starting, pausing and deleting Midlets and allocating resources among the Midlets. It is responsible for managing the activities of multiple Midlets within a runtime Environment.
What makes Midlets different is the fact that they can exploit efficiently and share the limited resources of a mobile device and that they can conform to the security framework of the device. Midlets are grouped together in packages called Midlet Suites.

2.Midlet Suites' architecture

A Midlet Suite includes:
• A JAR File that may include one or more Midlets , some other classes ,the resources (text, image ,properties ,database and utf files) and a manifest File .
• A JAD (Java Application Descriptor) File that includes all the information needed by the Application Management Software to identify, retrieve and install midlets.(Such information may be the name and the version of the application, the name of the vendor and a digital signature) .The descriptor allows the application management software on the device to verify that the MIDlet is suited to the device before loading the full JAR file of the MIDlet suite.

3.Compiling:J2ME Wireless Toolkit and Net and NetBeans Mobile

•In order to run a Midlet in a computer we need an emulator. An emulator is software that simulates the execution of the application on various mobile devices.
•J2ME Wireless Toolkit includes an emulator and a Development Environment that can be used to prepare class and jar Files and directories and to run class Files. As result, it simulates the whole procedure of executing Over The Air an application from a server to a device

4. The Code

WRLIrc Packages
The basic packages imported in WRLIrc are:
•javax.microedition.midlet
•javax.microedition.lcdui
•javax.microedition.io

javax.microedition.midlet

A MIDlet is an abstract class. Every application must extend this class to allow the application management software to control it and to be able to retrieve properties from the application descriptor and notify and request state changes.The methods of this class allow the application management software to create, start, pause, and destroy a MIDlet.

Specifically a Midlet contains methods that:
•Signal the MIDlet to enter a specific state (start,pause,destroy) or notify the application software that the Midlet has entered a state.
•Provide the MIDlet with a mechanism to retrieve named properties from the application management software.
•Requests that the device displays or installs the indicated URL.

javax.microedition.lcdui

This package manages the user iterface. It includes:
•A Displayable object that shows one or more components in the screen.Only one Displayable may be visible at a time, and the user can see and interact with only contents of that Displayable
•The class Display that acts as the display manager that is instantiated for each active MIDlet and provides methods to retrieve information about the device's display capabilities. A Displayable is made visible by calling the setCurrent() method of Display.
•A variety of components that can be displayed
–List is used when the user should select from a predefined set of choices.
–TextBox is used when asking textual input.
–Alert is used to display temporary messages containing text and images.
–A special class Form is defined for cases where screens with a predefined structure are not sufficient.

WRLIrc's classes

•Class Hierarchy
•java.lang.Object
–Database
–javax.microedition.lcdui.Displayable
•javax.microedition.lcdui.Canvas
–DisplayCanvas–Encoding
–Listener (implements java.lang.Runnable)
–Message
–javax.microedition.midlet.MIDlet
•WLIrc (implements javax.microedition.lcdui.CommandListener)
–PollHttpIrc (implements Irc)
–ScreenOutput (implements javax.microedition.lcdui.CommandListener)
•Channel
•Console
•Private–ScreenOutput.ListListener
–ScreenOutput.TextboxListener
–SocketIrc (implements Irc)
–Utils

5.How does WRLirc work

•WRLIrc's main classes are WRLIrc.java and Screenoutput.java.
When the Midlet is launched WRLIrc.java is the first class to be executed .It displays on the screen the name of the project and a menu with the commands (connect,config,advanced,font, exit) and defines the results of the interaction with the user. If the user selects the connect command the class PollHttpIrc will open a new connection.If the user selects the config command the class WRLIrc will display a new form with some Textfields that are used for entering data.
Class database is used to save all the data that must be reused when the application is closed and then reopened.It uses the class RecordStore. A record store consists of a collection of records which will remain persistent

across multiple invocations of the MIDlet. The platform is responsible for making its best effort to maintain the integrity of the MIDlet's record stores throughout the normal use of the platform, including reboots, battery changes, etc. Record stores are created in platform-dependent locations, which are not exposed to the MIDlets. Records are saved in properties files that can be found in the resources directory.

6.Design of adding the autocomplete command

•A vector called messages will be used to store the users' favourite messages.

•The menu displayed under the message textbox will have three more commands:
–save,
–delete,
–autocomplete

•The command save will add the content of the textbox as a new element to vector messages and will call the Database save method. This method replaces the third record of the Record Store with the vector messages. Before this happens the vector messages should be converted into a byte[]array.

•The autocomplete command will compare all the elements of the vector messages to the string entered into the textbox. Then it will display a list with all the messages that start with this String.

7.Design problems and solutions

I have tried to save messages in two different ways: by adding a new record to the RecordStore and by adding the vector to the existing record. When I tried to close and reopen the device ,all the records (even the configuration data such as the nickname) where destroyed .I have tried to delete the database file where the records are stored. Then the autocomplete command worked correctly. Every time the number or the content of the Record store changes the database file must be deleted before recompiling the program. Otherwise recordstore cannot display the contents of the database file. This problem is related to the way in which the database file is written.

C.Communication with the developper

1.On Apr 3, 2005 7:07 PM, polina <polina@users.sourceforge.net> wrote:
Mr Sverre,

I am writting regarding to the WRIrc program that I downloaded from sourceforge.net.

I am an undergraduate student from the department of Management and Technology(http://www.dmst.aueb.gr/) of the University of Economics and Business of Athens of Greece.

In the context of the course Software Comprehension and Maintenance (http://www.dmst.aueb.gr/dds/ismr/index.htm) , I am studying the WRIrc program. One of my assignments is adding a certain feature to the program.I have been looking for> a to do list but I have not found anything yet.

Therefore,I have made an attempt to add an autocomplete , a save and a delete command under the msg command of the menu.This way the user will have the choice to save his messages (in a vector stored in the second Record of the RecordStore) and then reuse them by typing their first letters.The autocomplete command finds the saved message that starts with these letters and prints it out.

I would really like to send you my WLIrc.jad and src files but I have not found your e-mail yet and I am still trying to use CVS.

I have tried to make the fewest changes to the source code and it works.I would like to ask you if you could possibly examine my implementation for any improvements and if you could make any suggestions for adding other features to your program,
Thanks for your time,
Yours sincerely,
Tzavala Polina

2. On Apr 5, 2005 6:00 PM

Hi

My email address is thesverre@gmail.com, I think it is a link to it on wlirc's homepage http://wirelessirc.sf.net

The autocomplete sounds like a good idea

I have not started to use CVS yet, because I have been the only developer to this project. But I will probably add my code to CVS later.

I will be happy if you send me your source files.

Yours sincerely
Sverre

3.

Thank you very much for answering my mail.
Please find attached the WRLIrc file with the changes I have made. All my changes are the following:
ScreenOutput.java: lines 222-327
Database.java: lines 138,171-185,281
Utils.java: line 77

I have created a public Vector called messages. Its bytes are saved in the same byte[] in which the bytes of the array rs are saved. This way I did not have to create a new record (I have tried to do the least possible changes) but I had to create one more method (SplitVectorToString).

I would like to ask you to make to me any recommendations and suggestions about the implementation as this is the main subject of my assignement.
Yours sincerely,
Tzavala Polina

4.The developper's answer was as follows:

I have now added my project to CVS, and added your user "polina" to the project. You are available to read and write from the cvs repository. I have also merged your code to my code. Was a little bit difficult because I had also made some changes to the code.

The autocomplete function looks OK, but I think it would be better if you made a list of all entries. And when you write some text, the list derease to these entries that match your text.

I made also some small changes to the code, especially in the database.java file. Look at the repository for a update.

Other things to do:

- make a historylist, like mirc's arrow up, arrow down... (this is only saved in buffer)
- timestamps
- ignore list that could be comma separated
- make the codebase smaller. Remove the textboxlistener class
- implement flood control
- nick list,(move back, next button to a command instead of an element)

D.Implementation

The following classes were slightly altered:

1. **public class Database**

```
/*
Copyright (C) 2003  Sverre Kristian Valskr?

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2
of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA  02111-1307, USA.
*/



import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.util.Random;
import java.util.Vector;

import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Font;
import javax.microedition.rms.RecordStore;


public class Database
{
	protected Vector messages= new Vector();
	protected String host =  "thesverre.servebeer.com"; // my server
	protected String gateway =
"http://thesverre.servebeer.com/wap/httpgateway";
	protected String nick = "";
	protected String channels ="#WLIrc";
	protected int  port =6667;
```

```
        public final static int LITTLE_INFO = 0;
        public final static int NORMAL_INFO = 1;
        protected int debug =NORMAL_INFO;
        protected int connection =CON_SOCKET;
        protected int polltime = 30; // in seconds
        protected int font_face = 0;
        protected int font_style = 0;
        protected int font_size = 2;
        protected boolean waitafterdata = true;
        protected String startupScript = "";
        protected String password = "";
        protected String encoding ="";
        protected int maxLinesInWindow=200;
        protected String userkey ="";
        protected String realname ="WLirc user";
        protected String friends ="";
        protected boolean fastconnect = false;
        protected boolean notifier_PRIV = false;
        protected boolean notifier_CHAN = true;
        protected boolean notifier_FRIEND = false;
        protected String notifier_TYPE= "S"; // [S,A,SA] Sound or alert or Both
        private Display display;
        protected boolean timestamp=true;

        public final static int CON_SOCKET = 0;
        public final static int CON_HTTPPOLL = 1;
        private final static int NUMREC = 21;
        private final static String NAMEOFSTORE = "config";



        public Database(Display display)
        {
                this.display = display;
        }
        /**
         * returns
         */
        public boolean load()
        {
                boolean firsttime = true;
                try{

                RecordStore recstore =
RecordStore.openRecordStore(NAMEOFSTORE,
                                        true);
                int size = recstore.getNumRecords();
        if (size !=3 || !new
String(recstore.getRecord(1)).equals(WLIrc.VERSION) )
                {
                        recstore .closeRecordStore();
                        RecordStore.deleteRecordStore(NAMEOFSTORE);
                        recstore  =
RecordStore.openRecordStore(NAMEOFSTORE,
                                    true);
          byte[] bv = WLIrc.VERSION.getBytes();
          recstore.addRecord(bv,0,bv.length);

                        String[] rs = new String[NUMREC];


                        encoding = System.getProperty("microedition.encoding");
                        if ( encoding == null || encoding.equals(""))
                                encoding = "ISO-8859-1";

                        // generates an unique ID, the id will be used by a identd
                        userkey ="";
                        Random r = new Random();
                        for (int i=0;i<2;i++)
                        {
                                int j = Math.abs(r.nextInt() % 25) +65;
                                userkey += String.valueOf((char)j);
                        }

                        rs[0] = host;
                        rs[1] = gateway;
                        rs[2] = nick;
                        rs[3] = channels;
                        rs[4] = new Integer(debug).toString();
                        rs[5] = new Integer(connection).toString();
                        rs[6] = new Integer(polltime).toString();
                        rs[7] =  new Integer(font_face).toString();
                        rs[8] =new Integer(font_style).toString();
                        rs[9] =new Integer(font_size).toString();
                        rs[10] =waitafterdata ? "1" : "0";
                        rs[11] =startupScript;
                        rs[12] =password;
                        rs[13] =new Integer(port).toString();
                        rs[14] =encoding;
                        rs[15] =new Integer(maxLinesInWindow).toString();
                        rs[16] =userkey;
                        rs[17] =realname;
                        rs[18] =friends;
                        rs[19] = fastconnect ? "1" : "0";
                        rs[20]=timestamp ? "1" : "0";
```

```
			byte[] b = getByteArray(rs);
			byte[] bm = getByteArray(messages);
			recstore.addRecord(b,0,b.length);
			recstore.addRecord(bm,0,bm.length);
		}
		else
		{
			String[] rs = new String[NUMREC];
			firsttime = false;
			String  b = new String(recstore.getRecord(2));
			String  bm = "";
			if (recstore.getRecord(3) != null)
				bm = new String(recstore.getRecord(3));
			rs = Utils.splitString(b,String.valueOf((char)0));
			host = getRecordString(rs,0);
			gateway =getRecordString(rs,1);
			nick = getRecordString(rs,2);
			channels = getRecordString(rs,3);
			debug = getRecordInt(rs,4);
			connection = getRecordInt(rs,5);
			polltime = getRecordInt(rs,6);
			font_face = getRecordInt(rs,7);
			font_style = getRecordInt(rs,8);
			font_size = getRecordInt(rs,9);
			waitafterdata = getRecordBoolean(rs,10);
			startupScript = getRecordString(rs,11);
			password = getRecordString(rs,12);
			port = getRecordInt(rs,13);
			encoding = getRecordString(rs,14);
			maxLinesInWindow= getRecordInt(rs,15);
			userkey = getRecordString(rs,16);
			realname = getRecordString(rs,17);
			friends= getRecordString(rs,18);
			fastconnect= getRecordBoolean(rs,19);
			timestamp= getRecordBoolean(rs,20);


	messages=Utils.splitStringToVector(bm,String.valueOf((char)0));
		}

		recstore.closeRecordStore();
		recstore = null;
		}
		catch(Exception e)
		{
			e.printStackTrace();
			System.err.println("ERROR DATABASE");
			WLIrc.writeError(e,display,WLIrc.mainForm);
		}
		return firsttime;
	}
	private byte[] getByteArray(String[] rs)
		throws IOException
	{
		ByteArrayOutputStream ba = new ByteArrayOutputStream();
		for (int i = 0; i < rs.length; i++) {
			String t = rs[i] + String.valueOf((char)0);
			ba.write(t.getBytes());
		}
		return ba.toByteArray();
	}

	private byte[] getByteArray(Vector messages)
		throws IOException
	{
		ByteArrayOutputStream ba = new ByteArrayOutputStream();
		for (int i = 0; i < messages.size(); i++) {
			String t = messages.elementAt(i).toString() +
String.valueOf((char)0);
			ba.write(t.getBytes());
		}

		return ba.toByteArray();
	}

	private String getRecordString(String[] rs,int nr)
		throws Exception
	{
		String b = rs[nr];
		return (b != null) ? b.trim() : "";
	}
	private boolean getRecordBoolean(String[] rs,int nr)
		throws Exception
	{
		String b  = rs[nr];
		return (b != null && new String(b).equals("1"));
	}
	private int getRecordInt(String[] rs,int nr)
		throws Exception
	{
		String b  = rs[nr];
		return (b != null) ? Integer.parseInt(new String(b)) : 0;
	}
	public String[] getChannels()
	{
		return Utils.hasNoValue(channels) ? null
:Utils.splitString(channels," ");
	}

	public String[] getFriends()
	{
		return Utils.hasNoValue(friends) ? null :Utils.splitString(friends,"
");
	}


	public Font getFont()
	{
		int style = font_style;
		if (style == 0)
			style = Font.STYLE_PLAIN;
		else if (style == 1)
			style = Font.STYLE_BOLD;
		else
			style = Font.STYLE_ITALIC;

		int size = font_size;
		if (size == 0)
			size = Font.SIZE_LARGE;
		else if (size == 1)
			size = Font.SIZE_MEDIUM;
		else
			size = Font.SIZE_SMALL;

		int face = font_face;
		if (face == 0)
			face = Font.FACE_MONOSPACE;
		else if (face == 1)
			face = Font.FACE_PROPORTIONAL;
		else
			face = Font.FACE_SYSTEM;
		return Font.getFont(face,style,size);
	}

	public void save()
```

```
        {

                try{
                String[] rs = new String[NUMREC];
                rs[0]= host;
                rs[1]= gateway;
                rs[2]= nick;
                rs[3]= channels;
                rs[4]= new Integer(debug).toString();
                rs[5]= new Integer(connection).toString();
                rs[6]= new Integer(polltime).toString();
                rs[7]= new Integer(font_face).toString();
                rs[8]= new Integer(font_style).toString();
                rs[9]= new Integer(font_size).toString();
                rs[10]= waitafterdata ? "1" : "0";
                rs[11]= startupScript;
                rs[12]= password;
                rs[13]=new Integer(port).toString();
                rs[14]=encoding;
                rs[15]= new Integer(maxLinesInWindow).toString();
                rs[16] = userkey;
                rs[17]= realname;
                rs[18]= friends;
                rs[19]= fastconnect ? "1" : "0";
                rs[20]=timestamp? "1" : "0";


                RecordStore recstore =
RecordStore.openRecordStore(NAMEOFSTORE,

          true);
                byte[] b = getByteArray(rs);

                recstore.setRecord(2,b,0,b.length);
                byte[] bm = getByteArray(messages);

                recstore.setRecord(3,bm,0,bm.length);

                recstore.closeRecordStore();
                recstore = null;
                }
                catch(Exception e)
                {
                        WLIrc.writeError(e,display,WLIrc.mainForm);
                }
        }
}
```

2. public class ScreenOutput

```
/*
Copyright (C) 2003  Sverre Kristian Valskr?

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2
of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA  02111-1307, USA.
*/

import java.io.IOException;
import java.util.*;

import javax.microedition.lcdui.*;
import javax.microedition.lcdui.Command;
import javax.microedition.lcdui.CommandListener;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Displayable;
import javax.microedition.lcdui.List;
import javax.microedition.lcdui.TextBox;
import javax.microedition.lcdui.TextField;
import javax.microedition.lcdui.Alert;

public class ScreenOutput implements CommandListener {
	protected Display display;
	protected Irc irc;
	protected Database db;
   private static final String MY_FAVOURITES = "MY FAVOURITE
MESSAGES";
	private static final String CHANGE_NICK = "Change nick";
	private static final String JOIN_CHANNEL = "Join channel";
	private static final String QUERY = "Query";
	protected TextBox textbox;
	protected Alert savedAlert=new Alert("Saved");
	protected Alert deletedAlert=new Alert("Deleted");
	protected Command searchCommand;
	protected Command msgCommand;
	protected Command actionCommand;
	protected Command sendActionCommand;
   protected Command saveCommand;
   protected Command autoCompleteCommand;
   protected Command MyFavouritesCommand;
   protected Command deleteCommand;
        protected Command sendCommand;
        protected Command cancelCommand;
        protected Command okCommand;
        protected Command colorCommand;
        protected Command boldCommand;
        protected Command underlineCommand;
        protected Command backCommand;

        private Command disconnectCommand;
        private Command windowsCommand;
        protected DisplayCanvas canvas;
        private String name;
   private List dynlist;
   private List exclusiveList;
   protected  boolean listflag=false;

   protected Command nickCommand;
        protected Command joinCommand;
        protected Command queryCommand;
        protected String changenick = null;
        protected Command timestampCommand;
        protected String TimeStampString;


        public ScreenOutput(
                Display display,
                Database db,
                Irc irc,
                String name,
                int headercolor) {
                if (!(db.debug == Database.LITTLE_INFO && this instanceof
Console))
                        WLIrc.allwindows.push(this);

                this.display = display;
                this.name = name;
                this.db = db;
                if(db.timestamp==true)TimeStampString="TimeStampOn";
                else TimeStampString="TimeStampOff";
                this.irc = irc;
            backCommand=new Command("Back", Command.OK, 1);
                autoCompleteCommand=new Command("AutoComplete",
Command.OK, 1);
                MyFavouritesCommand=new Command("MyFavourites",
Command.OK, 80);
                saveCommand=new Command("Save", Command.OK, 1);
                deleteCommand=new Command("Delete", Command.OK, 2);
                sendCommand = new Command("Send", Command.OK, 1);
                cancelCommand = new Command("Cancel", Command.CANCEL, 2);
                okCommand = new Command("Ok", Command.OK, 1);
                colorCommand = new Command("Insert color char",
Command.ITEM, 3);
                boldCommand = new Command("Insert bold char", Command.ITEM,
3);
                underlineCommand =
                        new Command("Insert Underline char", Command.ITEM, 3);
                sendActionCommand = new Command("Send", Command.OK, 1);
        searchCommand = new Command("Search", Command.OK, 1);

                msgCommand = new Command("Msg", Command.OK, 10);

                windowsCommand = new Command("Windows",
Command.SCREEN, 20);
                joinCommand = new Command("Join", Command.SCREEN, 30);
                queryCommand = new Command("Query", Command.SCREEN, 40);
                nickCommand = new Command("Change nick", Command.SCREEN,
50);
                disconnectCommand = new Command("Disconnect",
Command.SCREEN, 70);
                timestampCommand=new Command
(TimeStampString,Command.SCREEN, 80);

        canvas = new DisplayCanvas(name, db);
                canvas.headerColor = headercolor;
        canvas.addCommand(msgCommand);
        canvas.addCommand(MyFavouritesCommand);
                canvas.addCommand(windowsCommand);
                canvas.addCommand(disconnectCommand);
                canvas.addCommand(joinCommand);
                canvas.addCommand(nickCommand);
                canvas.addCommand(queryCommand);
                canvas.addCommand(timestampCommand);
                canvas.setCommandListener(this);
        }

        public void commandAction(Command c, Displayable s) {


        try {
             if (c == autoCompleteCommand) {
                        textbox = new TextBox("Search for", "", 1000,
TextField.ANY);
                        textbox.setCommandListener(this);
                        textbox.addCommand(searchCommand);
                        display.setCurrent(textbox);
                }
                else if (c == searchCommand){
```

```
                                        searchData();

                                }

                                else if (c == saveCommand) {

                                        saveData();

                                }
                                else if (c == deleteCommand) {

                                        deleteData();

                                }
                                else if (c == sendCommand) {
                                sendData();

                                } else if (c == cancelCommand) {
                                        show();
                                        textbox = null;
                                } else if (c == sendActionCommand) {
                                        String str = textbox.getString();
                                        show();
                                        textbox = null;
                                        irc.writeLine(
                                                "PRIVMSG "
                                                        + getName()
                                                        + " :"
                                                        + Listener.CTCP
                                                        + "ACTION "
                                                        + str
                                                        + Listener.CTCP);
                                        write("* " + db.nick + " " + str, true);
                                } else if (c == okCommand) {
                                        if (textbox.getTitle().equals(JOIN_CHANNEL))
        {
                                                String str = textbox.getString();
                                                if (!str.startsWith("#"))
                                                        str = "#" + str;
                                                show();
                                                irc.writeLine("JOIN " + str);
                                                textbox = null;
                                        } else if (

            textbox.getTitle().equals(CHANGE_NICK)) // change nick
                                                {
                                                String str = textbox.getString();
                                                irc.writeLine("NICK " + str);
                                                changenick = str;
                                                show();
                                                textbox = null;

                                        } else {
                                                String str = textbox.getString();
                                                WLIrc.getPrivate(str, display, db, irc);
                                                textbox = null;

                                        }
                                } else if (c == colorCommand) {
                                        textbox.setString(
                                                textbox.getString() +
        Utils.toString(Listener.COLOR));
                                } else if (c == underlineCommand) {
                                        textbox.setString(
                                                textbox.getString() +
        Utils.toString(Listener.UNDERLINE));
                                } else if (c == boldCommand) {
                                        textbox.setString(
                                                textbox.getString() +
        Utils.toString(Listener.BOLD));
                                }
                        } catch (IOException e) {
                                WLIrc.writeError(e, display, WLIrc.mainForm);
                        }

                        if(c==timestampCommand){
                                canvas.removeCommand(timestampCommand);
                                db.timestamp=!db.timestamp;
                                db.save();
                                db.load();
                                if(db.timestamp==true)
                                TimeStampString="TimeStampOn";
                                else TimeStampString="TimeStampOff";
                                timestampCommand=new Command
        (TimeStampString,Command.SCREEN, 80);
                       canvas.addCommand(timestampCommand);
                     }
                        else if (c == joinCommand) {
                        textbox = new TextBox(JOIN_CHANNEL, "", 30,
        TextField.ANY);
                        textbox.setCommandListener(this);
                        textbox.addCommand(okCommand);
                        textbox.addCommand(cancelCommand);
                        display.setCurrent(textbox);
                } else if (c == nickCommand) {
                        textbox = new TextBox(CHANGE_NICK, "", 30,
        TextField.ANY);
                        textbox.setCommandListener(this);
                        textbox.addCommand(okCommand);
                        textbox.addCommand(cancelCommand);
                        display.setCurrent(textbox);
                } else if (c == queryCommand) {
                        textbox = new TextBox(QUERY, "", 30, TextField.ANY);
                        textbox.setCommandListener(this);
                        textbox.addCommand(okCommand);
                        textbox.addCommand(cancelCommand);
                        display.setCurrent(textbox);
                } else if (c == disconnectCommand) {
                        // disconn
                        irc.setConnected(false);
                        display.setCurrent(WLIrc.mainForm);
                        WLIrc.cleanup();
                } else if (c == msgCommand) {
                        addTextBox();
                        listflag=false;
                }
                else if(c == MyFavouritesCommand){

            exclusiveList = new List("Matching messages", Choice.EXCLUSIVE);
            for (int j=0;j<db.messages.size();j++){
                exclusiveList.append(db.messages.elementAt(j).toString(),null);
            }
            listflag=true;
            exclusiveList.addCommand(sendCommand);
            exclusiveList.addCommand(deleteCommand);
            exclusiveList.addCommand(cancelCommand);
            exclusiveList.addCommand(autoCompleteCommand);
            exclusiveList.setCommandListener(this);
            display.setCurrent(exclusiveList);


                                        }

                        else if (c == actionCommand) {
                        textbox = new TextBox("Write Action", "", 1000,
        TextField.ANY);
                        textbox.setCommandListener(this);
                        textbox.addCommand(sendActionCommand);
                        textbox.addCommand(cancelCommand);
                        display.setCurrent(textbox);
                } else if (c == windowsCommand) {

                        dynlist = new List("Windows", List.IMPLICIT);

                        Enumeration e = WLIrc.allwindows.elements();
                        while (e.hasMoreElements()) {
                                ScreenOutput so = (ScreenOutput) e.nextElement();
                                dynlist.append(so.getName(), null);
                        }
```

```
			dynlist.setCommandListener(this);
			display.setCurrent(dynlist);

		} else if (dynlist != null && c == List.SELECT_COMMAND) {

			String name = dynlist.getString(dynlist.getSelectedIndex());
			Enumeration e = WLIrc.allwindows.elements();
			while (e.hasMoreElements()) {
				ScreenOutput so = (ScreenOutput) e.nextElement();
				if (name.equals(so.getName())) {
					so.show();
				}
			}

		}

	}

	public void write(String str, boolean action) {
		write(new Message(str, action ? Message.ACTION :
Message.NORMAL));
	}
	public void write(Message m) {
		if (getDisplayable().isShown())
			canvas.windowstatus = 0;
		else {
			if (m.getType() == Message.NORMAL || m.getType() ==
Message.ACTION)
				canvas.windowstatus = Message.NORMAL;
			else
				canvas.windowstatus = Message.INFO;
		}
		canvas.addText(m);

		Displayable disp = display.getCurrent();
		if (disp instanceof DisplayCanvas) {
			((DisplayCanvas) disp).repaint();
		}

	}

	public void writeInfo(String str) {
		String time="";
		if (db.timestamp) {
			Calendar cal = Calendar.getInstance();
			time = "[" + cal.get(Calendar.HOUR_OF_DAY) + ":" +
				 (cal.get(Calendar.MINUTE)<10?"0":"") +
cal.get(Calendar.MINUTE) + "]";
			}
		write(new Message("**** " + time+str, Message.INFO));
	}

	public Displayable getDisplayable() {
		return canvas;
	}
	public String getName() {
		return name;
	}

	public void show() {
		display.setCurrent(canvas);
	}

	public void close() {
		WLIrc.allwindows.removeElement(this);
		if (WLIrc.allwindows.size() > 0) {
			ScreenOutput so = (ScreenOutput)
WLIrc.allwindows.lastElement();
			display.setCurrent(so.canvas);
		} else
			display.setCurrent(WLIrc.getConsole().canvas);
	}
	public void clear() {
		canvas.v.removeAllElements();
	}

	protected void addTextBox() {
		textbox = new TextBox("Write text", "", 1000, TextField.ANY);
		textbox.setCommandListener(this);
		textbox.addCommand(autoCompleteCommand);
		textbox.addCommand(saveCommand);
      textbox.addCommand(deleteCommand);
      textbox.addCommand(sendCommand);
		textbox.addCommand(cancelCommand);
		textbox.addCommand(colorCommand);
		textbox.addCommand(boldCommand);
		textbox.addCommand(underlineCommand);
		display.setCurrent(textbox);
	}
protected void searchData(){
	String[] stringArray = new String [10];
	      Image[] imageArray = null;
					String str = textbox.getString();
					exclusiveList = new List("Matching messages",
Choice.EXCLUSIVE);
					for (int j=0;j<db.messages.size();j++){

	if(db.messages.elementAt(j).toString().startsWith(str))
```

```
        exclusiveList.append(db.messages.elementAt(j).toString(),null);
                                    }

      listflag=true;
      exclusiveList.addCommand(sendCommand);
      exclusiveList.addCommand(cancelCommand);
      exclusiveList.setCommandListener(this);
      display.setCurrent(exclusiveList);
        }
        protected void sendData() throws IOException {
        String str;
        if(listflag)
                str = exclusiveList.getString(exclusiveList.getSelectedIndex());
          else  str = textbox.getString();


                show();
                textbox = null;
                display.setCurrent(textbox);


                if (str != null && str.length() > 0) {
                        if (str.charAt(0) == '/') {
                                if (str.toUpperCase().startsWith("/MSG")) {
                                        String[] s = Utils.splitString(str, " ");
                                        if (s.length > 1) {
                                                String rest = "";
                                                for (int i = 2; i < s.length; i++)
                                                        rest += s[i] + " ";
                                                irc.writeLine("PRIVMSG " + s[1] + " :"
rest);
                                                write(
                                                        new Message(
                                                                "Message to " + s[1] + ":
" + rest,
                                                                Message.INFO));
                                        }
                                } else if (str.toUpperCase().startsWith("/ME")) {
                                        irc.writeLine(
                                                "PRIVMSG "
                                                        + getName()
                                                        + " :"
                                                        + Listener.CTCP
                                                        + "ACTION "
                                                        + str
                                                        + Listener.CTCP);
                                        write("* " + db.nick + " " + str, true);
                                } else if (str.toUpperCase().startsWith("/CLEAR")) {
                                        clear();
                                        write(new Message("window cleared!",
Message.INFO));
                                } else {
                                        irc.writeLine(str.substring(1));
                                        new Message("RAW MESSAGE: " + str,
Message.INFO);
                                }
                        } else {
                                irc.writeLine("PRIVMSG " + getName() + " :" + str);
                                write(db.nick + "> " + str, false);
                        }
                }
        }

        protected void saveData()  {
                String str = textbox.getString();
                db.messages.addElement(str);
                db.save();

                savedAlert.setString("Message saved");
                display.setCurrent(savedAlert);
                textbox.setString(str);
                display.setCurrent(textbox);
        }
        protected void deleteData()  {
                String str;
        if(listflag)
                str = exclusiveList.getString(exclusiveList.getSelectedIndex());
          else  str = textbox.getString();

                        for (int j=0;j<db.messages.size();j++){
                        if(db.messages.elementAt(j).toString().startsWith(str)){

        if(!listflag)textbox.setString(db.messages.elementAt(j).toString());
                                db.messages.removeElementAt(j);
                                        db.save();
                                }
                        }
                        deletedAlert.setString("Message deleted");
                        display.setCurrent(deletedAlert);
                    textbox.setString(str);
                    display.setCurrent(textbox);
    }

}
```

2. public class Utils

```
    /*
Copyright (C) 2003  Sverre Kristian Valskrε

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2
of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA  02111-1307, USA.
*/

import java.util.Enumeration;
import java.util.Vector;

import javax.microedition.lcdui.Alert;
import javax.microedition.lcdui.AlertType;
import javax.microedition.lcdui.Display;

public class Utils {

	public static boolean exits(String[] s, String match) {
		if (s != null) {
			for (int i = 0; i < s.length; i++) {
				if (s[i].toUpperCase().equals(match.toUpperCase()))
					return true;

			}
		}
		return false;
	}

	public static String argsfrom(String[] s, int arg) {

			String r = "";
			for (int i = arg; i < s.length; i++) {
				r += s[i] + " ";
			}
			return r.trim();
		}

	public static Object getElement(Vector v, String s) {
		if (v != null) {
			Enumeration e = v.elements();
			while (e.hasMoreElements()) {
				String el = (String) e.nextElement();
				if (el.toUpperCase().equals(s.toUpperCase()))
					return el;
			}
		}
		return null;
	}

	public static String[] splitString(String str, String delims) {
		if (str == null
			|| str.equals("")
			|| delims == null
			|| delims.length() == 0)
			return null;
		String nstr = new String(str);
		Vector v = new Vector();
		int pos = 0;
		int newpos = 0;
		int count = 0;
		while (newpos != -1) {
			newpos = nstr.indexOf(delims, pos);
			if (newpos != -1) {

				v.addElement(nstr.substring(pos, newpos));
				count++;
				nstr =
					nstr.substring(newpos + (delims.length()),
nstr.length());
			}
		}

		v.addElement(nstr);

		String[] s = new String[v.size()];
		for (int i = 0; i < s.length; i++) {
			s[i] = (String) v.elementAt(i);
		}
		return s;
	}

	public static Object popFifo(Vector v) {
		if (!v.isEmpty()) {
			Object o = (Object) v.firstElement();
```

```
                v.removeElementAt(0);
                return o;
            }
            return null;
        }

        public static String URLencode(String s) {
            if (s != null) {
                StringBuffer tmp = new StringBuffer();
                int i = 0;
                try {
                    while (true) {
                        int b = (int) s.charAt(i++);
                        if ((b >= 0x30 && b <= 0x39)
                            || (b >= 0x41 && b <= 0x5A)
                            || (b >= 0x61 && b <= 0x7A)) {
                            tmp.append((char) b);
                        } else {
                            tmp.append("%");
                            if (b <= 0xf)
                                tmp.append("0");
                            tmp.append(Integer.toHexString(b));
                        }
                    }
                } catch (Exception e) {
                }
                return tmp.toString();
            }
            return null;
        }

        public static boolean hasNoValue(String s) {
            return (s == null || s.equals("") || s.getBytes().length == 0);
        }

        public static String toString(char b) {
            char[] ba = new char[1];
            ba[0] = b;
            return new String(ba);
        }

        public static String trimName(String name) {
            if (name.charAt(0) == '@' || name.charAt(0) == '+')
                return name.substring(1).trim();
            else
                return name.trim();
        }

        public static void notifier(String header,String text,Database db,Display
display)
```

```
        {
                Alert a = new Alert(header,text,null,AlertType.INFO );
                if (db.notifier_TYPE.indexOf("S") != -1)
                        a.getType().playSound(display);
                if (db.notifier_TYPE.indexOf("A") != -1)
                {
                        a.setTimeout(Alert.FOREVER);
                        display.setCurrent(a,display.getCurrent());
                }

        }

        public static Vector splitStringToVector(String str, String delims) {
                String s[] = splitString(str,delims);
                Vector v = new Vector();
                if (s != null)
                {

                        for (int i=0;i<s.length;i++)
                        {
                                v.addElement(s[i]);
                        }

                }

        return v;
        }



}
```

E.Integrating the changes into the program Looking for a CVS program

Using Tortoise CVS

CVS is a program that stores all the versions of a file in a single file in a clever way that only stores the differences between versions.

TortoiseCVS is a front-end client to make using CVS easier and more intuitive. It allows developers to work with files controlled by CVS directly from Windows Explorer™.

A particular benefit is that TortoiseCVS has an SSH client built into it, without special setup. Anyone can easily commit code to SourceForge.net projects using TortoiseCVS.

Indeed Tortoise is a user friendly program.All the actions needed to do a commit were the following:

Firsly, I made an account on SourceForge.net.The username I have chosen is my developper name. Then I filled in the following configuration form of The Tortoise

•Protocol: Secure Shell (:ext)
•Server: cvs.sourceforge.net
•Directory: /cvsroot/wirelessirc
•Username:polina

And finally I chose the commit command from the windows explorer.

F.Experimenting with the To Do List

Trying to implement the timestamp:

The timestamp is the time indication that appears before each message we receive:

```
if(c==timestampCommand){
        canvas.removeCommand(timestampCommand);
        db.timestamp=!db.timestamp;
        db.save();
        if(db.timestamp==true)
        TimeStampString="TimeStampOn";
        else TimeStampString="TimeStampOff";
        timestampCommand=new Command
        (TimeStampString,Command.SCREEN, 80);
          canvas.addCommand(timestampCommand);
                    }
public void writeInfo(String str) {
                String time="";
                if (db.timestamp) {
                        Calendar cal = Calendar.getInstance();
                        time = "[" + cal.get(Calendar.HOUR_OF_DAY) + ":" +

                                (cal.get(Calendar.MINUTE)<10?"0":"")+cal.get(Cal
                                endar.MINUTE) + "]";
                }
                write(new Message("**** " + time+str, Message.INFO));
}
```

<u>Trying to remove the back command from the namelist and add it as a menu command:</u>

```
/**namelist.append(BACK, null);**/
    namelist.addCommand(backCommand);if(c == backCommand){
                if(display.getCurrent().equals(namelist))
                listnames(c, s);
                else if(display.getCurrent()==nameslist)
                show();
                else
                  display.setCurrent(namelist);
 }
```

G. Working on WRLIrc :Advantages and Disadvantages

The initial reason why I have chosen WRLIrc is the fact that it can be installed and compiled very easily and very quickly .In fact it was the only java program among 7 programs that I downloaded that I managed to compile. This is due to the fact that Wireless Toolkit is a user friendly runtime Environment.

What's more WRLIrc is a very small program. Consequently I had the opportunity to examine its whole structure and understand how it works instead of studying only the function of the new added feature.

But WRLIrc has some disadvantages as well: It has no documentation and the code is not very well organised (One of the To Do features is removing a whole surplus class).

Nevertheless,the most important thing is that working with WRLIrc was a trigger to reading the documentation of java MicroEdition ,to looking for some more information about mobile applications and to using Wireless Toolkit.

Finally, the suggested To Do List is very interesting and I will try to go on with implementing its items.

All in all, working with WRLIrc was a really rewarding experience.