CS 683 Emerging Technologies
Fall Semester, 2004
Doc 23 Messaging & CDC
Contents

References 2
Wireless Messaging 3
  http Example 4
  Messaging 6
    SMSSender 7
    SMSSend 10
    SMSReceive 14
CDC 19
Virtual Machine 19
CDC Class Libraries 19

Copyright ©, All rights reserved. 2004 SDSU & Roger Whitney, 5500 Campanile Drive, San Diego, CA 92182-7700 USA. OpenContent (http://www.opencontent.org/opl.shtml) license defines the copyright on this document.

References

J2ME in a Nutshell, Kim Topley, O'Reilly, 2002, Chapter 6 & 7

Examples in this lecture are based on examples that come with the J2ME Toolkit

Wireless Messaging

- Sockets
  TCP & UDP

- Http
- Messaging
- Web Services

http Example

```
import javax.microedition.lcdui.*;
import javax.microedition.midlet.MIDlet;
import javax.microedition.io.*;
import java.io.*;

public class HttpExample extends MIDlet  {
   public HttpExample() {

      Display display = Display.getDisplay(this);
      String page;
      try {
         page = getViaHttpConnection("http://www.eli.sdsu.edu/index.html");
      }
      catch (IOException error) {
         page = error.getMessage();
      }

      Form form = new Form("Http");
      form.append( page);

      display.setCurrent(form);
   }

   public void startApp() { }
   public void pauseApp() { }
   public void destroyApp( boolean unconditional ) { }
```

HttpExample Continued

```
String getViaHttpConnection(String url)  throws IOException {
   HttpConnection http = null;
   InputStream input = null;
   try {
      http = (HttpConnection)Connector.open(url);
      int responseCode = http.getResponseCode();

      if (responseCode != HttpConnection.HTTP_OK)
          return "HTTP response code: " + responseCode;

       input = http.openInputStream();

       int pageLength = (int)http.getLength();
       byte[] data = null;
       if (pageLength > 0) {
          int actual = 0;
          int bytesread = 0 ;
          data = new byte[pageLength];
          while ((bytesread != pageLength) && (actual != -1)) {
            actual = input.read(data, bytesread, pageLength - bytesread);
            bytesread += actual;
           }
       }
       return new String(data);
   } catch (ClassCastException e)  {
       return "Not an HTTP URL";
    } finally  {
       if (input != null)
          input.close();
       if (http != null)
          http.close();
   }
  }
}
```

Messaging

Example from J2ME Toolkit
Classes

- SMSSender.java
- SMSSend.java
- SMSRecieve.java

SMSSender

```
/*
 * @(#)SMSSender.java   1.5 03/10/29
 *
 * Copyright (c) 1999-2003 Sun Microsystems, Inc. All rights reserved.
 * PROPRIETARY/CONFIDENTIAL
 * Use is subject to license terms
 */

package example.sms;

import javax.microedition.io.*;
import javax.microedition.lcdui.*;
import javax.wireless.messaging.*;

import java.io.IOException;

public class SMSSender
    implements CommandListener, Runnable {

    Command sendCommand = new Command("Send", Command.OK, 1);
    Command backCommand = new Command("Back", Command.BACK, 2);
    Display display;
    String smsPort;
    String destinationAddress;
    TextBox messageBox;
    Displayable backScreen;
    Displayable sendingScreen;
```

SMSSender Continued

```
  public SMSSender(String smsPort, Display display,
        Displayable backScreen, Displayable sendingScreen) {
    this.smsPort = smsPort;
    this.display = display;
    this.destinationAddress = null;
    this.backScreen = backScreen;
    this.sendingScreen = sendingScreen;

    messageBox = new TextBox("Enter Message", null, 65535,
TextField.ANY);
    messageBox.addCommand(backCommand);
    messageBox.addCommand(sendCommand);
    messageBox.setCommandListener(this);
  }

  public void promptAndSend(String destinationAddress)
  {
    this.destinationAddress = destinationAddress;
    display.setCurrent(messageBox);
  }

  public void commandAction(Command c, Displayable s) {
    try {
      if (c == backCommand) {
        display.setCurrent(backScreen);
      } else if (c == sendCommand) {
        display.setCurrent(sendingScreen);
        new Thread(this).start();
      }
    } catch (Exception ex) {
      ex.printStackTrace();
    }
  }
```

SMSSender Continued

```
    public void run() {
        String address = destinationAddress + ":" + smsPort;

        MessageConnection smsconn = null;
        try {
            smsconn = (MessageConnection)Connector.open(address);

            TextMessage txtmessage = (TextMessage)smsconn.newMessage(
                MessageConnection.TEXT_MESSAGE);
            txtmessage.setAddress(address);
            txtmessage.setPayloadText(messageBox.getString());
            smsconn.send(txtmessage);
        } catch (Throwable t) {
            System.out.println("Send caught: ");
            t.printStackTrace();
        }

        if (smsconn != null) {
            try {
                smsconn.close();
            } catch (IOException ioe) {
                System.out.println("Closing connection caught: ");
                ioe.printStackTrace();
            }
        }
    }
}
```

SMSSend

```
/*
* Copyright (c) 1999-2003 Sun Microsystems, Inc. All rights reserved.
 * PROPRIETARY/CONFIDENTIAL
 * Use is subject to license terms
 */

package example.sms;

import javax.microedition.midlet.*;
import javax.microedition.io.*;
import javax.microedition.lcdui.*;
import javax.wireless.messaging.*;

import java.io.IOException;

public class SMSSend extends MIDlet
    implements CommandListener {

    Command exitCommand  = new Command("Exit", Command.EXIT, 2);
    Command okCommand = new Command("OK", Command.OK, 1);
    Display display;
    String smsPort;
    TextBox destinationAddressBox;
    Alert errorMessageAlert;
    Alert sendingMessageAlert;
    SMSSender sender;
    Displayable resumeScreen = null;
```

SMSSend Continued

```
public SMSSend() {
   smsPort = getAppProperty("SMS-Port");

   display = Display.getDisplay(this);

   destinationAddressBox = new TextBox("Destination Address?",
      null, 256, TextField.PHONENUMBER);
   destinationAddressBox.addCommand(exitCommand);
   destinationAddressBox.addCommand(okCommand);
   destinationAddressBox.setCommandListener(this);

   errorMessageAlert = new Alert("SMS", null, null, AlertType.ERROR);
   errorMessageAlert.setTimeout(5000);

   sendingMessageAlert = new Alert("SMS", null, null, AlertType.INFO);
   sendingMessageAlert.setTimeout(5000);
   sendingMessageAlert.setCommandListener(this);

   sender = new SMSSender(smsPort, display, destinationAddressBox,
      sendingMessageAlert);

   resumeScreen = destinationAddressBox;
}

 public void startApp() {
   display.setCurrent(resumeScreen);
}

public void pauseApp() {
   resumeScreen = display.getCurrent();
}

public void destroyApp(boolean unconditional) { }
```

SMSSend Continued

```
public void commandAction(Command c, Displayable s) {
   try {
      if (c == exitCommand || c == Alert.DISMISS_COMMAND) {
         destroyApp(false);
         notifyDestroyed();
      } else if (c == okCommand) {
         promptAndSend();
      }
   } catch (Exception ex) {
      ex.printStackTrace();
   }
}

private void promptAndSend() {
   String address = destinationAddressBox.getString();
   if (!SMSSend.isValidPhoneNumber(address)) {
      errorMessageAlert.setString("Invalid phone number");
      display.setCurrent(errorMessageAlert, destinationAddressBox);
      return;
   }
   String statusMessage = "Sending message to " + address + "...";
   sendingMessageAlert.setString(statusMessage);
   sender.promptAndSend("sms://" + address);
}
```

SMSSend Continued

```
    private static boolean isValidPhoneNumber(String number) {
       char[] chars = number.toCharArray();
       if (chars.length == 0) {
          return false;
       }
       int startPos = 0;
       // initial '+' is OK
       if (chars[0]=='+') {
          startPos = 1;
       }
       for (int i = startPos; i < chars.length; ++i) {
          if (!Character.isDigit(chars[i])) {
             return false;
          }
       }
       return true;
    }
}
```

SMSReceive

```
/*
* Copyright (c) 1999-2003 Sun Microsystems, Inc. All rights reserved.
*/

package example.sms;

import javax.microedition.midlet.*;
import javax.microedition.io.*;
import javax.microedition.lcdui.*;
import javax.wireless.messaging.*;

import java.io.IOException;

public class SMSReceive extends MIDlet
    implements CommandListener, Runnable, MessageListener {

    Command exitCommand  = new Command("Exit", Command.EXIT, 2);
    Command replyCommand = new Command("Reply", Command.OK, 1);
    Alert content;
    Display display;
    Thread thread;
    String[] connections;
    boolean done;
    String smsPort;
    MessageConnection smsconn;
    Message msg;
    String senderAddress;
    Alert sendingMessageAlert;
    SMSSender sender;
    /** The screen to display when we return from being paused */
    Displayable resumeScreen;
```

SMSReceive Continued

```
    public SMSReceive() {
        smsPort = getAppProperty("SMS-Port");

        display = Display.getDisplay(this);

        content = new Alert("SMS Receive");
        content.setTimeout(Alert.FOREVER);
        content.addCommand(exitCommand);
        content.setCommandListener(this);
        content.setString("Receiving...");

        sendingMessageAlert = new Alert("SMS", null, null, AlertType.INFO);
        sendingMessageAlert.setTimeout(5000);
        sendingMessageAlert.setCommandListener(this);

        sender = new SMSSender(smsPort, display, content,
  sendingMessageAlert);

        resumeScreen = content;
    }

    public void pauseApp() {
        done = true;
        thread = null;
        resumeScreen = display.getCurrent();
    }
```

SMSReceive Continued

```
public void destroyApp(boolean unconditional) {
   done = true;
   thread = null;
   if (smsconn != null) {
      try {
         smsconn.close();
      } catch (IOException e) {
         // Ignore any errors on shutdown
      }
   }
}

public void startApp() {
   String smsConnection = "sms://:" + smsPort;
   if (smsconn == null) {
      try {
         smsconn = (MessageConnection) Connector.open(smsConnection);
         smsconn.setMessageListener(this);
      } catch (IOException ioe) {
         ioe.printStackTrace();
      }
   }

   connections = PushRegistry.listConnections(true);
   if (connections == null || connections.length == 0) {
      content.setString("Waiting for SMS on port " + smsPort + "...");
   }
   done = false;
   thread = new Thread(this);
   thread.start();

   display.setCurrent(resumeScreen);
}
```

SMSReceive Continued

```
public void notifyIncomingMessage(MessageConnection conn) {
   if (thread == null) {
      done = false;
      thread = new Thread(this);
      thread.start();
   }
}

public void run() {
   try {
      msg = smsconn.receive();
      if (msg != null) {
         senderAddress = msg.getAddress();
         content.setTitle("From: " + senderAddress);
         if (msg instanceof TextMessage) {
            content.setString(((TextMessage)msg).getPayloadText());
         } else {
            StringBuffer buf = new StringBuffer();
            byte[] data = ((BinaryMessage)msg).getPayloadData();
            for (int i = 0; i < data.length; i++) {
               int intData = (int)data[i] & 0xFF;
               if (intData < 0x10) {
                  buf.append("0");
               }
               buf.append(Integer.toHexString(intData));
               buf.append(' ');
            }
            content.setString(buf.toString());
         }
         content.addCommand(replyCommand);
         display.setCurrent(content);
      }
   } catch (IOException e) {  e.printStackTrace(); }
}
```

SMSReceive Continued

```
    public void commandAction(Command c, Displayable s) {
        try {
            if (c == exitCommand || c == Alert.DISMISS_COMMAND) {
                destroyApp(false);
                notifyDestroyed();
            } else if (c == replyCommand) {
                reply();
            }
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }

    private void reply() {
        // remove the leading "sms://" for diplaying the destination address
        String address = senderAddress.substring(6);
        String statusMessage = "Sending message to " + address + "...";
        sendingMessageAlert.setString(statusMessage);
        sender.promptAndSend(senderAddress);
    }
}
```

CDC

Virtual Machine

Full Java virtual machine specs

No Jit or Hotspot

Regular Java virtual machine does not run on devices

CDC Class Libraries

When includes class from J2SE includes all the class

Except deprecated methods

- java.io
  nearly complete

- java.lang
  Missing compiler, UnknownException

- java.lang.ref
  Complete

- java.lang.reflect
  Complete

- java.math
  Missing BigDecimal

- java.net
  UDP but no TCP

- java.security
  Minimal

- java.security.cert
  Only Certificate

- java.text
  Supports only numbers, dates & error messages

- java.util
  Nearly complete

- java.util.jar
  Reading but no writing

- java.util.zip
  Reading but no writing

- javax.microedition.io