1.Tujuan

- Mendeskripsikan Generic Connection Framework, dan bagaimana ia dapat digunakan untuk mendukung method koneksi yang berbeda-beda.
- Menspesifikasikan parameter-parameter koneksi dengan menggunakan format pengalamatan GCF URL
- Membuat koneksi HTTP/HTTPS
- Menciptakan MIDlet dengan menggunakan TCP sockets dan UDP datagram

2. Latar Belakang

Generic Connection Framework mendukung koneksi packet (socket) dan stream (datagram). Sesuai dengan namanya, framework ini menyediakan API dasar bagi koneksi di CLDC. Framework ini menyediakan pondasi umum dari berbagai koneksi seperti HTTP, socket, dan datagram. Walaupun Bluetooth dan serial I/O termasuk kedalam API ini, GCF menyediakan satu set API yang lebih generic dan mendasar yang menjadi abstraksi dari berbagai tipe koneksi. Harus dicatat, bahwa tidak semua tipe koneksi dibutuhkan bagi implementasi sebuah MIDP device.

3. Percobaan

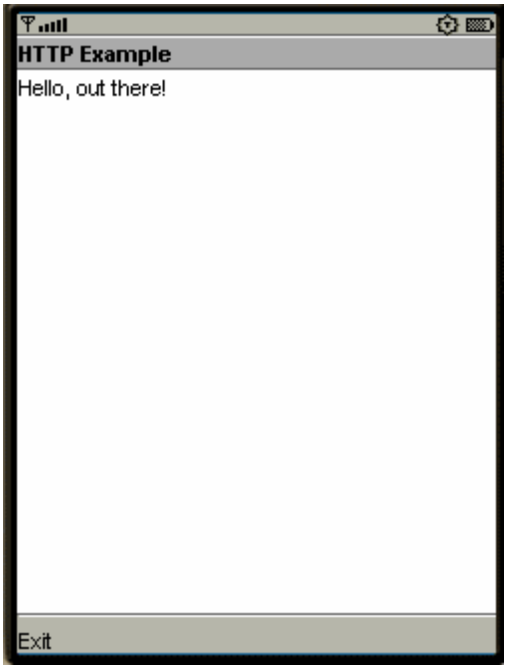
Percobaan 1: Koneksi HTTP

```
/*****************************************
 *    Percobaan 1 : Mencoba Koneksi HTTP   *
 *        Tested On : Motorola C380        *
 *       Test Result : Work Properly       *
 *****************************************/
import javax.microedition.io.*;
import java.io.*;
import javax.microedition.lcdui.*;
import javax.microedition.midlet.*;

public class HttpExample extends MIDlet implements CommandListener{
      Display display;
      Form formHttp;
      Command exitCommand = new Command("Exit", Command.EXIT, 0);
      HttpConnection connection = null;
```

```
      InputStream iStream = null;
      byte[] data = null;

      public HttpExample(){
            formHttp = new Form("HTTP Example");
            formHttp.addCommand(exitCommand);
            formHttp.setCommandListener(this);
            try {
                  connection = (HttpConnection) Connector.open("http://www.ba-
stuttgart.de/~weghorn/j2me.txt");
                  int code = connection.getResponseCode();
                  switch (code){
                  case HttpConnection.HTTP_OK:
                        iStream = connection.openInputStream();
                        int length = (int) connection.getLength();
                        if (length > 0){
                              data = new byte[length];
                              int totalBytes = 0;
                              int bytesRead = 0;
                              while ((totalBytes < length)) {
                                    bytesRead = iStream.read(
                                    data, totalBytes, length - totalBytes);
                                    if (bytesRead > 0){
                                          totalBytes += bytesRead;
                                    }else{
                                          break;
                                    }
                              }
                              formHttp.append(new String(data));
                        } else {
                              //panjang tidak diketahui, baca tiap karakter
                        }
                        break;
                  default:
                        break;
                  }
            } catch (Exception e){
            }
      }
      public void startApp(){
            if (display == null){
                  display = Display.getDisplay(this);
                  display.setCurrent(formHttp);
            }
      }
```

```
      public void pauseApp(){
      }
      public void destroyApp(boolean d){
      }
      public void commandAction(Command c, Displayable d){
            if (c == exitCommand){
                  destroyApp(true);
                  notifyDestroyed(); // Exit
            }
      }
}
```

Hasil :



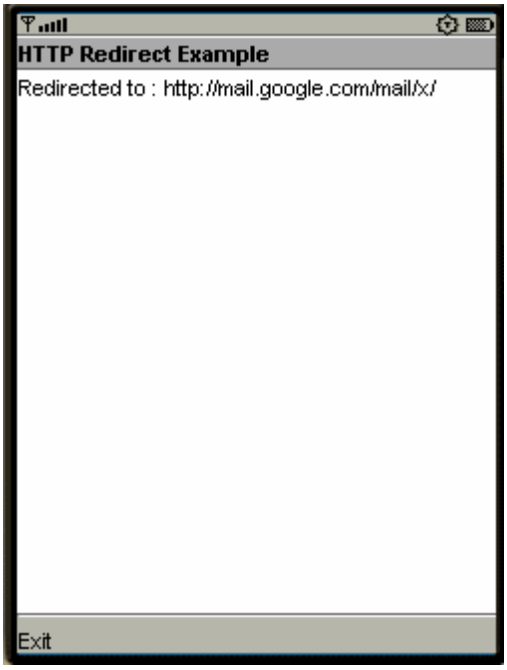
Percobaan 2 : Handling Redirect

```
/*****************************************
 *     Percobaan 2 : Handling Redirect     *
 *        Tested On : Motorola C380        *
 *       Test Result : Work Properly       *
 *****************************************/
```

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

public class HandlingRedirectExample extends MIDlet implements CommandListener{
      Display display;
      Form formHttp;
      Command exitCommand = new Command("Exit", Command.EXIT, 0);
      HttpConnection connection = null;
      InputStream iStream = null;
      byte[] data = null;

      public HandlingRedirectExample(){
            formHttp = new Form("HTTP Redirect Example");
            formHttp.addCommand(exitCommand);
            formHttp.setCommandListener(this);
            try {
                  connection = (HttpConnection)
Connector.open("http://m.gmail.com");
                  int code = connection.getResponseCode();
                  switch (code){
                        case HttpConnection.HTTP_MOVED_PERM:
                        case HttpConnection.HTTP_MOVED_TEMP:
                        case HttpConnection.HTTP_SEE_OTHER:
                        case HttpConnection.HTTP_TEMP_REDIRECT:
                              String newUrl =
connection.getHeaderField("Location");
                              formHttp.append("Redirected to : "+newUrl);
                        break;
                  default:
                        break;
                  }
                  connection.close();
            } catch (Exception e){
```

```
                  formHttp.append(e.toString());
            }
      }
      public void startApp(){
            if (display == null){
                  display = Display.getDisplay(this);
                  display.setCurrent(formHttp);
            }
      }
      public void pauseApp(){
      }
      public void destroyApp(boolean d){
      }

      public void commandAction(Command c, Displayable d){
            if (c == exitCommand){
                  destroyApp(true);
                  notifyDestroyed(); // Exit
            }
      }
}
```

Hasil :



Percobaan 3 : Koneksi HTTPS

```
/*****************************************
 *   Percobaan 3 : Mencoba Koneksi HTTPS   *
 *        Tested On : Motorola C380        *
 *       Test Result : Work Properly       *
 *****************************************/
```

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

public class HttpsExample extends MIDlet implements CommandListener{
      Display display;
      Form formHttps;
      Command exitCommand = new Command("Exit", Command.EXIT, 0);
      HttpsConnection connection = null;
      InputStream iStream = null;
```

```
      byte[] data = null;

      public HttpsExample(){
            formHttps = new Form("HTTPS Example");
            formHttps.addCommand(exitCommand);
            formHttps.setCommandListener(this);
            try {
                  connection = (HttpsConnection) Connector.open("https://msp.f-
secure.com/web-test/");
                  int code = connection.getResponseCode();
                  switch (code){
                  case HttpConnection.HTTP_OK:
                        iStream = connection.openInputStream();
                        int length = (int) connection.getLength();
                        if (length > 0){
                              data = new byte[length];
                              int totalBytes = 0;
                              int bytesRead = 0;
                              while ((totalBytes < length)) {
                                    bytesRead = iStream.read(
                                    data, totalBytes, length - totalBytes);
                                    if (bytesRead > 0){
                                          totalBytes += bytesRead;
                                    }else{
                                          break;
                                    }
                              }
                              formHttps.append(new String(data));
                        } else {
                              //panjang tidak diketahui, baca tiap karakter
                        }
                        break;
                  default:
                        break;
                  }
                  connection.close();
            } catch (Exception e){
                  formHttps.append(e.toString());
            }
      }
      public void startApp(){
            if (display == null){
                  display = Display.getDisplay(this);
                  display.setCurrent(formHttps);
            }
```

```
      }
      public void pauseApp(){
      }
      public void destroyApp(boolean d){
      }

      public void commandAction(Command c, Displayable d){
            if (c == exitCommand){
                  destroyApp(true);
                  notifyDestroyed(); // Exit
            }
      }
}
```

Hasil :

Percobaan 4 : Socket

```
/*****************************************
 * Percobaan 4 : Mencoba Socket          *
 * Tested On : Emulator Only             *
 * Test Result : Work Properly on Localhost *
 *****************************************/
```

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

public class TCPSocketExample extends MIDlet implements CommandListener{
      Display display;
      Form formSocket;
      Command exitCommand = new Command("Exit", Command.EXIT, 0);
      SocketConnection connection = null;
      InputStream iStream = null;
      OutputStream oStream = null;

      public TCPSocketExample(){
            formSocket = new Form("TCP Socket");
            formSocket.addCommand(exitCommand);
            formSocket.setCommandListener(this);
            try {
                  connection = (SocketConnection)
Connector.open("socket://localhost:80");
                  connection.setSocketOption(connection.DELAY, 0);
                  iStream = connection.openInputStream();
                  oStream = connection.openOutputStream();

                  oStream.write("GET /test.txt HTTP/1.0\n\n".getBytes());
                  int c = 0;
                  String data = "";
                  while((c = iStream.read()) != -1) {
                        data += (char)c;
                  }
                  formSocket.append(data);
                  oStream.close();
```

```
                iStream.close();
                connection.close();
            } catch (Exception e){
                formSocket.append(e.toString());
            }
      }
      public void startApp(){
            if (display == null){
                display = Display.getDisplay(this);
                display.setCurrent(formSocket);
            }
      }
      public void pauseApp(){
      }
      public void destroyApp(boolean d){
      }

      public void commandAction(Command c, Displayable d){
            if (c == exitCommand){
                destroyApp(true);
                notifyDestroyed(); // Exit
            }
      }
}
```

Hasil :

Percobaan 5 : Client – Server Socket

```
/*****************************************
 * Percobaan 5 : Server Socket           *
 * Tested On : Emulator Only             *
 * Test Result : Work Properly on Localhost *
 *****************************************/
```

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

public class ServerSocketExample extends MIDlet implements Runnable,CommandListener{
       Display display;
       Form formServer;
       Command exitCommand = new Command("Exit", Command.EXIT, 0);
       ServerSocketConnection connection = null;
```

```
    SocketConnection conn = null;
    InputStream iStream = null;

    public ServerSocketExample(){
           formServer = new Form("Server Socket");
           formServer.addCommand(exitCommand);
           formServer.setCommandListener(this);
    }
    public void startApp(){
           if (display == null){
                  display = Display.getDisplay(this);
                  display.setCurrent(formServer);
           }
           Thread t= new Thread(this);
           t.start();
    }
    public void pauseApp(){
    }
    public void destroyApp(boolean d){
    }

    public void commandAction(Command c, Displayable d){
           if (c == exitCommand){
                  try{
                         if (connection != null)
                                connection.close();
                         if (conn != null)
                                conn.close();
                         if (iStream != null)
                                iStream.close();
                  }catch (Exception e){
                  }
                  destroyApp(true);
                  notifyDestroyed(); // Exit
           }
    }
    public void start(){
    }
    public void run(){
           try {
                  connection = (ServerSocketConnection) Connector.open("socket://:3887");
                  conn = (SocketConnection) connection.acceptAndOpen();
                  conn.setSocketOption(conn.DELAY, 0);
                  iStream = conn.openInputStream();

                  int c = 0;
                  String data = "";
                  while((c = iStream.read()) != -1) {
                         data += (char)c;
                  }
```

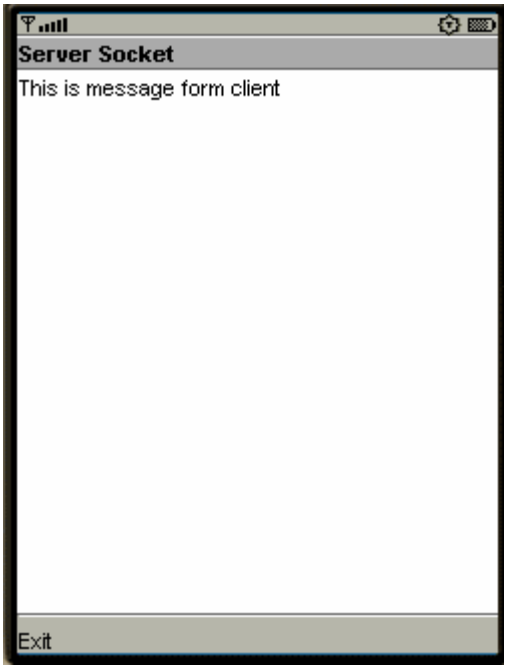
```
                formServer.append(data);
            } catch (Exception e){
                formServer.append(e.toString());
            }
      }
      public void stop(){
      }
}
```

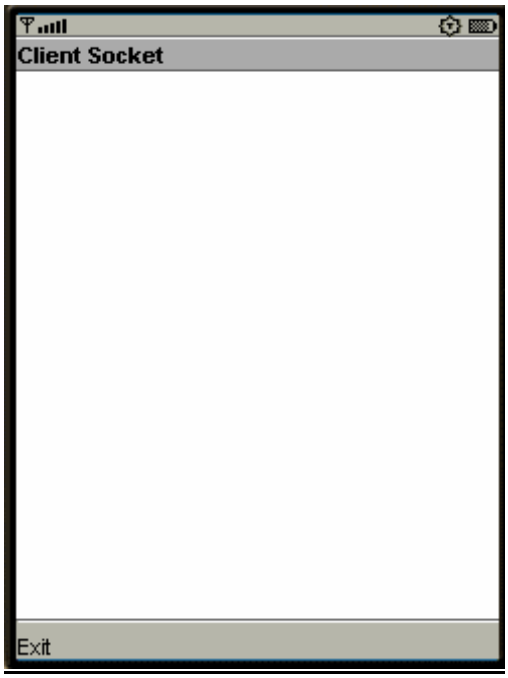
```
/*****************************************
 * Percobaan 5 : Client Socket       *
 * Tested On : Emulator Only               *
 * Test Result : Work Properly on Localhost *
 *****************************************/
```

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

public class ClientSocket extends MIDlet implements CommandListener{
      Display display;
      Form formClient;
      Command exitCommand = new Command("Exit", Command.EXIT, 0);
      SocketConnection connection = null;
      OutputStream oStream = null;

      public ClientSocket(){
            formClient = new Form("Client Socket");
            formClient.addCommand(exitCommand);
            formClient.setCommandListener(this);
            try {
                  connection = (SocketConnection)
Connector.open("socket://localhost:3887");
                  connection.setSocketOption(connection.DELAY, 0);
                  oStream = connection.openOutputStream();

                  oStream.write("This is message form client\n".getBytes());
                  oStream.close();
                  connection.close();
            } catch (Exception e){
                  formClient.append(e.toString());
            }
      }
      public void startApp(){
            if (display == null){
                  display = Display.getDisplay(this);
                  display.setCurrent(formClient);
            }
      }
```

```
	public void pauseApp(){
	}
	public void destroyApp(boolean d){
	}
	public void commandAction(Command c, Displayable d){
		if (c == exitCommand){
			try{
				if (connection != null)
					connection.close();
				if (oStream != null)
					oStream.close();
			}catch (Exception e){
			}
			destroyApp(true);
			notifyDestroyed(); // Exit
		}
	}
}
```

Hasil :





Percobaan 6 : Datagram Client – Server

```
/*****************************************
 * Percobaan 6 : Server Datagram          *
 * Tested On : Emulator Only              *
 * Test Result : Work Properly on Localhost *
 *****************************************/
```

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

public class DatagramServerExample extends MIDlet implements Runnable,CommandListener{
             Display display;
             Form formServer;
Command exitCommand = new Command("Exit", Command.EXIT, 0);
       DatagramConnection dc = null;
           Datagram dgram = null;
         InputStream iStream = null;
      public DatagramServerExample(){
                                          formServer = new Form("Datagram Server");
                                             formServer.addCommand(exitCommand);
                                            formServer.setCommandListener(this);
                     }
          public void startApp(){
                                                  if (display == null){
                                                              display =
Display.getDisplay(this);

      display.setCurrent(formServer);
                                                                    }
                                                Thread t= new Thread(this);
                                                        t.start();
                     }
          public void pauseApp(){
                     }
     public void destroyApp(boolean d){
                     }

public void commandAction(Command c, Displayable d){
                                                    if (c == exitCommand){
                                                              destroyApp(true);
                                                              notifyDestroyed(); // Exit
                                                           }
```

```
                    }
            public void start(){
                    }
             public void run(){
                                                        try {
                                                             dc = (DatagramConnection)
Connector.open("datagram://:3887");
                                                             while (true) {
                                                                    Datagram dgram =
dc.newDatagram(128);
                                                                    dc.receive(dgram);
                                                                    if
(dgram.getLength() > 0){
                                                                           String mesg =
                                                                                  new
String(dgram.getData(), 0, dgram.getLength());

          formServer.append(mesg);
                                                                    }
                                                             }
                                                    } catch (Exception e){

      formServer.append(e.toString());
                                                       }
                    }
public void stop(){
                    }
}
```

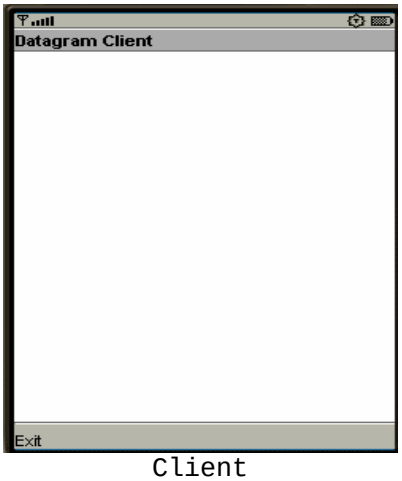
```
/*****************************************
 * Percobaan 6 : Client Datagram          *
 * Tested On : Emulator Only              *
 * Test Result : Work Properly on Localhost *

*****************************************/
```

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

public class DatagramClientExample extends MIDlet implements CommandListener{
              Display display;
              Form formClient;
Command exitCommand = new Command("Exit", Command.EXIT, 0);
       DatagramConnection dc = null;
           Datagram dgram = null;

      public DatagramClientExample(){
```

```
                                        formClient = new Form("Datagram Client");
                                           formClient.addCommand(exitCommand);
                                          formClient.setCommandListener(this);
                                        String url = "datagram://localhost:3887";
                                                        try {
                                                             dc = (DatagramConnection)
Connector.open(url);
                                                             byte[] msg = "This is
message form client\n".getBytes();
                                                             dgram =
dc.newDatagram(msg,msg.length,url);
                                                             dc.send(dgram);
                                                    } catch (Exception e){

      formClient.append(e.toString());
                                                                 }
                        }
public void startApp(){
                                                     if (display == null){
                                                             display =
Display.getDisplay(this);

       display.setCurrent(formClient);
                                                                 }
                        }
           public void pauseApp(){
                        }
     public void destroyApp(boolean d){
                        }

public void commandAction(Command c, Displayable d){
                                                    if (c == exitCommand){
                                                             destroyApp(true);
                                                             notifyDestroyed(); // Exit
                                                                 }
                        }
}
```

Hasil :

Server



Client

4. Latihan

6.7.1 Mendapatkan URL

Buatlah sebuah MIDlet yang mendapatkan HTTP URL. Aplikasi tersebut akan mendapatkan URL dengan method GET dan menampilkan jenis koneksi/ content properties (jika tersedia): Reponse Code, Response Message, Length, Type, Encoding, Expiration dan Last Modified Date.

Jawaban:

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

public class Exercise61 extends MIDlet implements CommandListener, Runnable {
    private Display display;
    private Command exitCommand = new Command("Exit", Command.EXIT, 1);
    private Command okCommand = new Command("Fetch", Command.OK, 1);
    private TextField urlField = new TextField(
            "URL",
            "http://localhost:8084/Chapter07/",
            64, TextField.URL);
    private StringItem message = new StringItem("URL Fetcher", "");
    HttpConnection connection = null;

    public void startApp() {
        display = Display.getDisplay( this );
        display.setCurrent(new MainForm());
    }

    public void pauseApp() {
    }

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

```
class MainForm extends Form {
    MainForm(){
        // Setup the Form
        super("Exercise 6.1");
        addCommand(exitCommand);
        addCommand(okCommand);
        append(urlField);
        append(message);
        setCommandListener(Exercise61.this);
    }
}

public void commandAction(Command c, Displayable d){
    if(c == exitCommand){
        notifyDestroyed();
    }
    if (c == okCommand){
        try {
            Thread t = new Thread(this);
            t.start();
        } catch (Exception e){}
    }
}

public void run(){
    try {
        String url = urlField.getString();
        connection = (HttpConnection)
        Connector.open(url);

        StringBuffer buff = new StringBuffer(1024);
        buff.append("Response Code: ");
        buff.append(connection.getResponseCode());
        buff.append("\n");

        buff.append("Response Mesg: ");
        buff.append(connection.getResponseMessage());
        buff.append("\n");

        buff.append("Length: ");
        buff.append(connection.getLength());
        buff.append("\n");

        buff.append("Content Type: ");
        buff.append(connection.getType());
        buff.append("\n");

        buff.append("getEncoding: ");
        buff.append(connection.getEncoding());
```

```
            buff.append("\n");

            buff.append("Date: ");
            buff.append(connection.getDate());
            buff.append("\n");

            buff.append("Expiration: ");
            buff.append(connection.getExpiration());
            buff.append("\n");

            buff.append("Last Modified: ");
            buff.append(connection.getLastModified());
            buff.append("\n");

            buff.append("URL: ");
            buff.append(connection.getURL());
            buff.append("\n");

            buff.append("Request Method: ");
            buff.append(connection.getRequestMethod());
            buff.append("\n");

            message.setText(buff.toString());
        } catch (Exception ex){

        }
    }
}
```