



```

public class Connection implements Runnable {
    public static Connection _instance;
    private ServerSocketConnection _server;
    private InputStream _in;
    private SocketConnection _client;
    public String _status = "not connected";
    public boolean _isConnected = false;
    private OutputStream _out;

    public static Connection getInstance() {
        if(_instance == null) {
            _instance = new Connection();
        }

        return _instance;
    }

    public Connection() {
        _status = "connecting...";
        new Thread(this).start();
    }

    public void write(String aData) throws Exception {
        _status = "sending data...";
        aData += '\0';

        synchronized (this) {
            if(_client != null) {
                _out.write(aData.getBytes());
                _out.flush();
            }
        }
    }

    public String read() {
        StringBuffer recv = new StringBuffer();

        synchronized (this) {
            if(_client != null) {
                try {
                    int ch;
                    if(_in.available() > 0) {
                        _status = "reading data...";
                        while(( ch = _in.read()) != '\0') {
                            recv.append( (char) ch );
                        }
                    }
                } catch (Exception e) {
                    _status = "read: " + e.toString();
                }
            }
        }

        return recv.toString();
    }

    public void close() {
        try {
            _server.close();
            _client.close();
        } catch (IOException e) {

```

## Flash\_Liteã Java\_MEã æ ¥ç¶ã ã ã æ 1æ³

```
        _status = "close: " + e.toString();
    }
}

public void run() {
    try {
        _status = "connecting...";
        _server = (ServerSocketConnection) Connector.open("socket://:9002");
        _client = null;
    } catch (Exception e) {
        _status = "Run (1): " + e.toString();
    }

    try {
        _status = "waiting for incoming connection...";
        _client = (SocketConnection) _server.acceptAndOpen();
        _client.setSocketOption(SocketConnection.DELAY, 0);
        _client.setSocketOption(SocketConnection.KEEPALIVE, 2);

        _out = _client.openOutputStream();
        _in = _client.openInputStream();

        _status = "client connected...";
    } catch (Exception e) {
        _status = "Run (2): " + e.toString();
    }
}
}
```

## MIDlet??? (Java ME)

????????Flash Lite??

```
/* Jarpa 1.05 - Java Packaging for Flash Lite Developers
 *
 * @author Felipe Andrade <felipe.andrade@i2tecnologia.com.br>
 *
 * http://www.i2tecnologia.com.br/jarpa/
 * http://blog.felipeandrade.org/
 * http://www.biskero.org
 */

package com.i2tecnologia.jarpa;

import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;
//import javax.microedition.location.*;

public class Jarpa extends MIDlet implements CommandListener, Runnable {
    private Form _form;
    private Command _cmdExit;
    private StringItem _status;
    private StringItem _statusRead;

    // values short than 20 sometimes close the midlet
    private static int MAX_INTERVAL = 20;

    // target folder
    private String _targetFolder = "file:///E:/Others/Jarpa_105.swf";
```

## Flash\_Liteã Java\_MEã æ ¥ç¶ã ã ã æ 1æ³

```
// sample increment
private int _inc = 0;

public void startApp() {
    _form = new Form("Jarpa Debug");
    _cmdExit = new Command("Exit", Command.EXIT, 1);
    _status = new StringItem("Teste STR", "");
    _statusRead = new StringItem("", "");

    _form.append(_status);
    _form.append(_statusRead);

    _form.addCommand(_cmdExit);
    _form.setCommandListener(this);
    Display.getDisplay(this).setCurrent(_form);

    new Thread(this).start();
}

public void pauseApp() {}
public void destroyApp(boolean aCond) {}

public void run() {
    try {
        Resources.getInstance().copyResources(_targetFolder);
        platformRequest(_targetFolder);
    } catch (Exception e) {
        _status.setText("Error: " + e.toString());
    }

    while(true) {
        _status.setText("Status: " + Connection.getInstance()._status);

        try {
            String sent = "" + (++_inc);
            String rcv = "";

            try {
                Connection.getInstance().write(sent);
                rcv = Connection.getInstance().read();
            } catch (Exception e) {
                destroyApp(true);
                notifyDestroyed();
            }

            _statusRead.setText("Write: " + sent + "\nRead: " + rcv + " / len: " + rcv.length);
            Thread.sleep(MAX_INTERVAL);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}

public void commandAction(Command aCmd, Displayable aDisplay) {
    destroyApp(true);
    notifyDestroyed();
}
}
```

## Resources???(Java ME)

Resources????JAR??SWF????????????????????????????????

```

/* Jarpa 1.05 - Java Packaging for Flash Lite Developers
 *
 * @author Felipe Andrade <felipe.andrade@i2tecnologia.com.br>
 *
 * http://www.i2tecnologia.com.br/jarpa/
 * http://blog.felipeandrade.org/
 * http://www.biskero.org
 */

package com.i2tecnologia.jarpa;

import java.io.DataOutputStream;
import java.io.InputStream;
import javax.microedition.io.Connector;
import javax.microedition.io.file.FileConnection;

public class Resources {
    public static Resources _instance;

    // data from the swf
    private InputStream _incomingData;

    private String _contentFolder = "";
    private String _toFolder = "";
    private String _fromFolder = "";

    public static Resources getInstance() {
        if(_instance == null)
            _instance = new Resources();

        return _instance;
    }

    public void copyResources(String aFolder) {
        _toFolder = aFolder;
        int index = aFolder.lastIndexOf('/');
        String fileName = aFolder.substring(index + 1, aFolder.length());

        _contentFolder = aFolder.substring(0, index);
        _fromFolder = "/res/" + (fileName);

        try {
            // get resources from the jar
            this._incomingData = this.getClass().getResourceAsStream(this._fromFolder);

            // open a connection (sign problems ... on N95 only working in the memory card)
            FileConnection fConn = (FileConnection)Connector.open(this._toFolder);
            FileConnection fContent = (FileConnection)Connector.open(this._contentFolder);

            if(!fContent.exists()){
                fContent.mkdir();
            }

            fContent.close();

            // if not exists creates the file
            if (!fConn.exists()) {

```

## Flash\_Liteã ¨Java\_MEã æ ¥ç¶ã ã ã æ 1æ³

```
fConn.create();

int ch = 0;
DataOutputStream dataStorage = fConn.openDataOutputStream();

// write to the file
while ((ch = this._incomingData.read()) > -1) {
    dataStorage.write(ch);
}

// closes
dataStorage.flush();
dataStorage.close();

if(this._incomingData != null)
    this._incomingData.close();
}

fConn.close();
} catch (Exception e) {
}
}
}
```

## Jarpa???? (Flash Lite)

??ActionScript??????Flash Lite????Java ME??

```
/**
    Jarpa - Jarpa Framework
    Copyright (C) 2008 i2tecnologia
*/

import mx.utils.Delegate;
import mx.events.EventDispatcher;

/**
    Jarpa
    @version 1.05
    @author: Felipe Andrade
*/
class com.i2tecnologia.jarpa.Jarpa extends EventDispatcher {
private var host_str:String;
private var port_num:Number;
private var jarpa_xmls:XMLSocket;
private var isConnected_bool:Boolean = false;

/**
        Jarpa(host_str:String, port_num:Number)
        @host_str: server host
        @port_num: port number ( > 1024 )

        Constructor
    */
public function Jarpa(host_str:String, port_num:Number) {
this.host_str = host_str;
this.port_num = port_num;
```

## Flash Liteã Java\_MEã æ ¥ç¶ã ã ã æ 1æ³

```
this.jarpa_xmls = new XMLSocket();
}

/**
     *
     * connect():Void
     *
     * Connect to local server
     */
public function connect():Void {
this.jarpa_xmls.connect(this.host_str, this.port_num);
this.jarpa_xmls.onConnect = Delegate.create(this, onConnect);
this.jarpa_xmls.onClose = Delegate.create(this, onClose);
}

/**
     *
     * onConnect(success_bool:Boolean)
     * @success_bool: connected or not
     *
     * Connection handler
     */
public function onConnect(success_bool:Boolean):Void {
    (true);setStatus
        ({type:"onConnect",
status:success_bool});
        (); stateReading
}

/**
     *
     * write(data_str:String)
     * @data_str: write a message to socket
     *
     * Write a message to socket
     */
public function write(data_str:String):Void {
this.jarpa_xmls.send(data_str);
}

/**
     *
     * onReadData(data_str:Boolean)
     * @data_str: receive a message from socket
     *
     * Asynchronous handler
     */
public function onReadData(data_str:String):Void {
    ({type:"onReadData",
data:data_str});
}

/**
     *
     * onClose():Void
     *
     * Invoked when the server is closed
     */
public function onClose():Void {
    (false);setStatus
        ({type:"onClose"});
}

/**
     *
     * stateReading():Void
     *
     * Start handling messages

```

## Flash\_Liteã Java\_MEã æ ¥ç¶ã ã ã æ 1æ³

```
*/
public function stateReading():Void {
this.jarpa_xmls.onData = Delegate.create(this, onReadData);
}

/**
        setStatus(status_bool:Boolean):Void
        @status: boolean value that indicates the status of application

        Set status of application
*/
public function setStatus(status_bool:Boolean):Void {
this.isConnected_bool = status_bool;
}

/**
        getStatus():Boolean

        Get status of application
*/
public function getStatus():Boolean {
return this.isConnected_bool;
}
}
```

## ??????????? (Flash Lite)

???Flash Lite???(.fla)????????????????????

```
stop();
fscommand2("FullScreen", true);
fscommand2("SetSoftkeys", "left", "right");
import com.i2tecnologia.jarpa.Jarpa;

var connection:Jarpa = new Jarpa('', 9002);

/* *****
        Jarpa
***** */
var handler_obj:Object = new Object();
var id:Number;

handler_obj.onJarpaConnect = function(evt_obj:Object):Void {
    preload_time = false;

    if(evt_obj.status) {
        text stãConnected.";
    } else {
        text stãConnection failed.";
        setInterval(function() {
            connect();
            _visible = true;
            text = "Reconnecting.";
            clearInterval(id);
        }, 4000);
    }
}

handler_obj.onJarpaClose = function(evt_obj:Object):Void {
```

???????????? (Flash Lite)

## Flash Lite - Java ME - Example 3

```
status_txt.setText("Connection closed.");
}

handler_obj.onReadData = function(evt_obj:Object):Void {
    status_txt.setText(evt_obj.data);
}

connection.addEventListener("onJarpaConnect", handler_obj);
connection.addEventListener("onJarpaClose", handler_obj);
connection.addEventListener("onReadData", handler_obj);

connection.connect();
status_txt.text = "Connecting to Java ME.";

/* *****
           Event keys
***** */
var handlerKey_obj:Object = new Object();
handlerKey_obj.onKeyDown = function() {
    switch(Key.getCode()) {
    case ExtendedKey.SOFT1:
        write("JARPA" + connection);
    break;
    case ExtendedKey.SOFT2:
        ("Quit");      fscommand2
    break;
    }
}

Key.addListener(handlerKey_obj);
```