Archived:Como conectar o Flash Lite e o Java ME

Aquivado: Este artigo foi arquivado, pois o conteúdo não é mais considerado relevante para se criar soluções comerciais atuais. Se você achar que este artigo ainda é importante, inclua o template {{ForArchiveReview|*escreva a sua justificativa*}}.

Não recomendamos o desenvolvimento em Flash Lite para aparelhos atuais da Nokia. Todos os artigos relacionados a Flash Lite foram arquivados. A plataforma Nokia Asha e os aparelhos recentes S40 não possuem mais o Flash Lite. O uso de Flash Lite em Symbian é limitado. Por exemplo, informações relativas à plataforma Nokia Belle podem ser encontradas neste link (em Inglês). Informações específicas de Flash Lite para aparelhos S40 ANTIGOS e Symbian podem ser encontradas na Flash Lite Developers Library (em Inglês).

Recentemente, uma nova geração de telefones móveis com suporte para Java ME e Flash Lite tem sido desenvolvido, o que tem proporcionado uma ampla aceitação destes dispositivos pelo mercado, ampliando a concepção e o desenvolvimento de aplicações híbridas. Neste contexto, o projeto Jarpa foi criado, permitindo que desenvolvedores de aplicações baseadas em Flash Lite criem e disseminem os seus negócios com aplicações Java ME.

Criado por Felipe Andrade e Alessandro Pace, o projeto Jarpa é baseado em software livre sob licença Apache.

Download

Faça o download dos exemplos dos códigos em: Códigos Google.http://www.i2tecnologia.com.br/jarpa/

Classe Socket (Java ME)

Cria uma conexão de soquete para ler e gravar dados.

```
 /*Jarpa 1,05 - Java Embalagens para desenvolvedores Flash Lite
  *
  *@Author Felipe Andrade <felipe.andrade@i2tecnologia.com.br> Traduzido por <Maicon
Herverton maiconherverton@yahoo.com.br>
  *
  *Http://www.i2tecnologia.com.br/jarpa/
  *Http://blog.felipeandrade.org/
  *Http://www.biskero.org
  */

package com.i2tecnologia.jarpa;

import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import javax.microedition.io.Connector;
import javax.microedition.io.ServerSocketConnection;
import javax.microedition.io.SocketConnection;

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...";
```


(C) Copyright Nokia 2014. All rights reserved.

```
        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) {
            _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) {
```


(C) Copyright Nokia 2014. All rights reserved.

```
            _status = "Run (2): " + e.toString();
        }
    }
}
```

Classe MIDlet (Java ME)

Responsável por enviar os dados personalizados ao cliente flash lite. Ela permite o acesso direto a serviços como Local.

```
/*Jarpa 1,05 - Java Embalagens para desenvolvedores Flash Lite
  *
  *@Author Felipe Andrade <felipe.andrade@i2tecnologia.com.br> Traduzido por <Maicon
Herverton maiconherverton@yahoo.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;

    //Valores menores que 20, fecham o MIDlet
    private static int MAX_INTERVAL = 20;

    //Pasta de destino
    private String _targetFolder = "file:///E:/Others/Jarpa_105.swf";

    //Amostra
    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());
```


(C) Copyright Nokia 2014. All rights reserved.

```
        }

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

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

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

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

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

Classe de recursos (Java ME)

A classe de recursos é utilizada para copiar o arquivo SWF com o JAR para a pasta do dispositivo.

```
/* Jarpa 1,05 - Java Embalagens para desenvolvedores Flash Lite
  *
  * @Author Felipe Andrade <felipe.andrade@i2tecnologia.com.br> Traduzido por <Maicon
Herverton maiconherverton@yahoo.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;

    // Dados do swf
    private InputStream _incomingData;

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

    public static Resources getInstance() {
        if(_instance == null)
```


(C) Copyright Nokia 2014. All rights reserved.

```
            _instance = new Resources();

        return _instance;
    }

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

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

        try {
            // Obtém recursos do jar
            this._incomingData = this.getClass().getResourceAsStream(this._fromFolder);

            // Abre uma conexão (problemas de sinal ... no N95 só funciona no cartão de
memória)
            FileConnection fConn = (FileConnection)Connector.open(this._toFolder);
            FileConnection fContent =
(FileConnection)Connector.open(this._contentFolder);

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

            fContent.close();

            // Se não existir cria o arquivo
            if (!fConn.exists()) {
                fConn.create();

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

                // Escreve no arquivo
                while ((ch = this._incomingData.read()) > -1) {
                    dataStorage.write(ch);
               }

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

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

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

        }
    }
}
```

Jarpa Connector (Flash Lite)

Esta classe ActionScript permite que os desenvolvedores Flash Lite conectem e manipulem dados do aplicativo Java ME.

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


(C) Copyright Nokia 2014. All rights reserved.

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

/**
  Jarpa
  @version 1.05
  @author: Felipe Andrade Tradução: Maicon Herverton
*/
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 )

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

  this.jarpa_xmls = new XMLSocket();
 }

 /**
    connect():Void

  Conecta-se ao servidor local
 */
 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

  Conexão
 */
 public function onConnect(success_bool:Boolean):Void {
  setStatus(true);
  dispatchEvent({type: "onJarpaConnect",
         status:success_bool});
  stateReading();
 }

 /**
    write(data_str:String)
    @data_str: write a message to socket

  Escreve a mensagem no 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
```


(C) Copyright Nokia 2014. All rights reserved.

```
  Manipulador assincrônico
 */
 public function onReadData(data_str:String):Void {
  dispatchEvent({type: "onReadData",
        data:data_str});
 }

 /**
    onClose():Void

    Chamado quando o servidor está fechado
 */
 public function onClose():Void {
  setStatus(false);
  dispatchEvent({type: "onJarpaClose"})
 }

 /**
    stateReading():Void

  Inicia o manipulador de mensagens
 */
 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

  Modifica os status da aplicação
 */
 public function setStatus(status_bool:Boolean):Void {
  this.isConnected_bool = status_bool;
 }

 /**
    getStatus():Boolean

  Obtém os status da aplicação
 */
 public function getStatus():Boolean {
  return this.isConnected_bool;
 }
}
```

Conteúdo do conector (Flash Lite)

Você tem que usar o seguinte código em um novo arquivo 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 {
 preloader_mc._visible = false;
```


(C) Copyright Nokia 2014. All rights reserved.

```
 if(evt_obj.status) {
  status_txt.text = "Connected.";
 } else {
  status_txt.text = "Connection failed.";
  id = setInterval(function() {
       connection.connect();
       preloader_mc._visible = true;
       status_txt.text = "Reconnecting.";
       clearInterval(id);
       }, 4000);
 }
}

handler_obj.onJarpaClose = function(evt_obj:Object):Void {
 status_txt.text = "Connection closed.";
}

handler_obj.onReadData = function(evt_obj:Object):Void {
 status_txt.text = 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.";

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

Key.addListener(handlerKey_obj);
```

Autor

FelipeAndrade 10:39, 25 May 2008 Traduzido por: maiconherverton 2009

(C) Copyright Nokia 2014. All rights reserved.

(C) Copyright Nokia 2014. All rights reserved.