ANEXO C

Tutorial Invocación de Servicios Web en Teléfonos Móviles con el JSR 172.

Introducción

Los dispositivos móviles, a diferencia de equipos de escritorio o servidores, no cuentan con suficiente capacidad de procesamiento para realizar operaciones de negocios o mantener en ellos extensas bases de datos. Sin embargo con las tecnologías de movilidad existentes, como GPRS, los dispositivos móviles pueden establecer comunicación a través de la web.

Objetivos

1. Adquirir conocimientos básicos acerca de los Servicios Web en Teléfonos Móviles.
2. Conocer la API para servicios Web (JSR 172) y desarrollar un cliente para consumir el servicio Web creado en el Tutorial 1.

Prerrequisitos

1. Conocimiento básico del lenguaje Java.
2. Conocimiento básico de J2ME.
3. Tutorial 1 Servicios Web.

Base Teórica

LA API J2ME Web Services (WSA 1.0 o jsr 172) extiende a la plataforma de servicios web para incluir clientes J2ME.

El jsr 172 se desarrolló para proveer una infraestructura con dos paquetes opcionales de J2ME para:

1. Proveer la capacidades básicas de procesamiento XML
2. Reutilizar conceptos de servicios Web cuando se diseña clientes J2ME para servicios empresariales.
3. Proveer APIs y convenciones para programar clientes J2ME de servicios empresariales.
4. Permitir la interoperabilidad de clientes J2ME con servicios Web.
5. Proveer un modelo programático para la comunicación de clientes J2ME con servicios Web, coherentes con otros clientes Java tales como J2SE.

Desarrollo del Tutorial

Software a utilizar.

- IDE Eclipse 3.3. Europa.
- Plug-In de eclipse Eclipse ME 1.76.que se puede encontrar en la página web http://eclipseme.org/updates
- Servidor de Aplicaciones Sun Java System Application Server.
- Sun Java Wireless Toolkit 2.5.1.

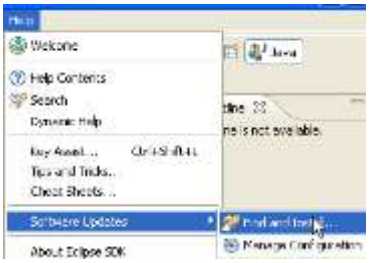
Nota: en este tutorial se parte del Tutorial 1, y se supone que el servicio web ConversorWS, creado en ese tutorial, está ejecutándose y su WSDL se encuentra en la dirección. http://localhost:8080/ConversorWebApp/ConversorWSService?WSDL

Instalación y Configuración de eclipse y del Plugin Eclipse ME

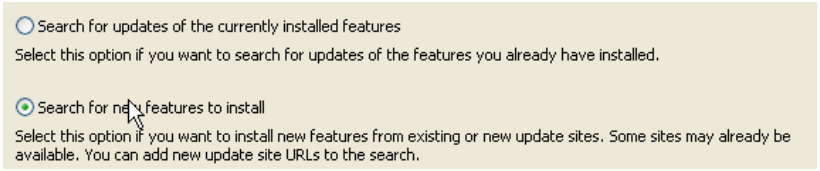
Paso 1: Ubicar la carpeta de instalación del WTK o instalarlo en una ubicación conveniente, por ejemplo C:\ WTK2.5.1

Paso 2: Instalar el plugin EclipseME utilizando la característica de Software Updates de Eclipse o usando el archivo eclipseme.feature_1.7.6_site.zip. Para esto:

1. Seleccione en Eclipse el menú Help, luego Software Updates y Find and Install.



2. En la ventana emergente Install/Update seleccione la opción “Search for new Updates to install”.



3. Haga clic en el botón “New Archived Site”. Busque el archivo eclipseme.feature_1.7.6_site.zip y haga clic en Abrir y luego en “OK”.

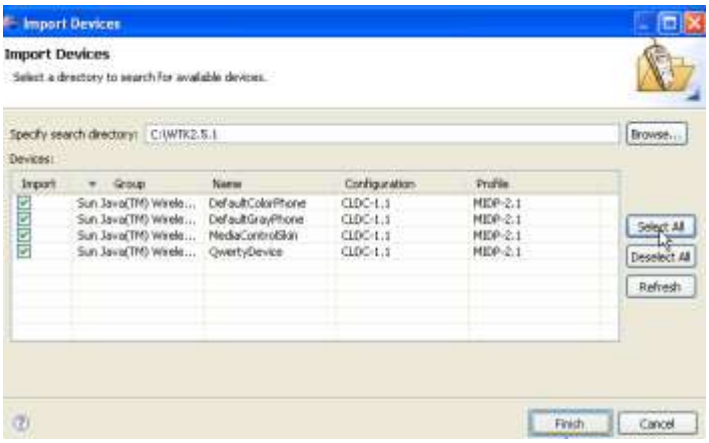
4. Haga clic en “Finish” y el proceso de instalación del plug-in lo guiará hasta finalizar.

Paso 3: Configurar los Simuladores J2ME del WTK dentro de Eclipse

1. En el menú Window de Eclipse seleccione Preferences, expanda la opción J2ME y seleccione Device Management. Usando el botón “Import” se lanza un diálogo donde se debe introducir la ruta de instalación del WTK. Posteriormente dar clic en el botón Refresh para que se carguen las referencias de los simuladores disponibles.

2. Haga clic en el botón “Browse” y seleccione la carpeta donde instaló el Wireless Toolkit (En este caso en C:\WTK2.1.5) y haga clic en el botón Aceptar.

3. Haga clic en el botón Refresh y observe que aparecen los emuladores que el wireless toolkit ofrece. Haga clic en Finish.



4. Finalmente seleccionar Default Color Phone como simulador por defecto.

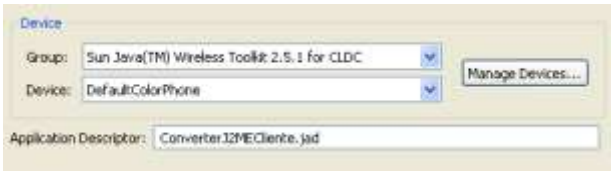
Default	Group	Name	Configuration	Profile
☐	Sun Java(TM) Wirele...	MediaControlSkin	CLDC-1.1	MIDP-2.1
☐	Sun Java(TM) Wirele...	DefaultGrayPhone	CLDC-1.1	MIDP-2.1
☑	Sun Java(TM) Wirele...	DefaultColorPhone	CLDC-1.1	MIDP-2.1
☐	Sun Java(TM) Wirele...	QwertyDevice	CLDC-1.1	MIDP-2.1

Creando el cliente J2ME

1. Haga clic en File luego New, a continuación seleccione Other, expanda la carpeta J2ME y seleccione J2ME MIDlet Suite y haga clic en Finish.

2. En la siguiente ventana en Project Name escriba “ConverterJ2MECliente”, haga clic en Next.

3. En la siguiente ventana, en la categoría Group seleccione Sun Java Wireless Toolkit 2.5.1 for CLDC, en la categoría Device seleccione DefaultColorPhone y haga clic en Next.



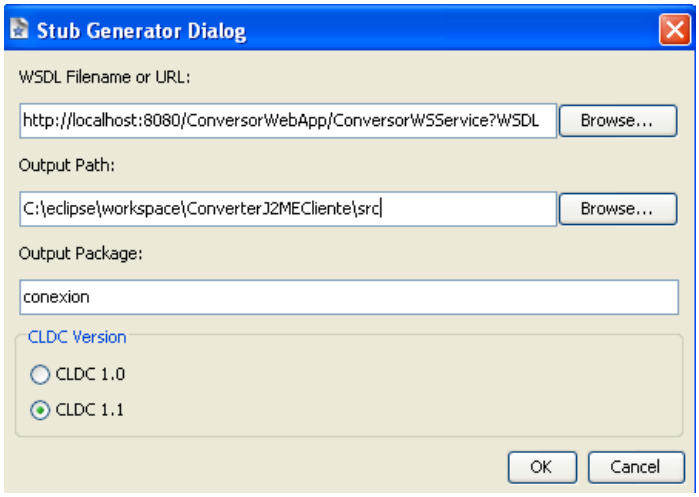
4. Haga clic en Finish.

5. Dentro del proyecto cree tres paquetes con los siguientes nombres: control, conexión y visual.

6. A continuación generaremos los Stubs utilizando la herramienta WTK 2.5.

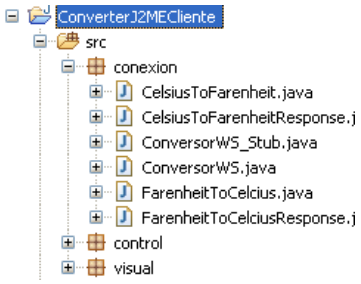
    a. Utilizando la opción Stub Generator de la aplicación Utilities del WTK (C:\WTK2.5.1\bin\utils.exe) vamos a generar las clases Stub que van a ser incluidas en la aplicación móvil. Seleccione la opción StubGenerator.

b. Introducir el URL del documento WSDL que describe el servicio a invocar, el paquete Java donde estas clases estarán dentro del proyecto, para este caso el paquete conexion, y la ruta en el disco duro donde se copiarán estas clases generadas. Seleccionar CLDC 1.1.



c. Haga clic en OK

7. Regresamos a eclipse y hacemos clic derecho sobre el proyecto y luego en Refresh, observamos que el paquete conexión ahora tiene las clases generadas por el Stub Generator Dialog.



8. En el paquete control creamos una clase llamada ConverterMID y agregamos el siguiente código:

```
package control;
import javax.microedition.lcdui.Display;
import javax.microedition.midlet.MIDlet;
import javax.microedition.midlet.MIDletStateChangeException;
import visual.ConverterForm;
public class ConverterMID extends MIDlet {
        private ConverterForm conversorForm;
        private Display display;
        public ConverterMID() {
                display = Display.getDisplay(this);
                conversorForm = new ConverterForm(this);
        }
        protected void startApp() throws
        MIDletStateChangeException {
                display.setCurrent(conversorForm);
        }

        protected void destroyApp(boolean arg0) throws
        MIDletStateChangeException {
                notifyDestroyed();
        }

        protected void pauseApp() {

        }
        public void salir() {
                try {
                        destroyApp(true);
                } catch (MIDletStateChangeException e) {
                        e.printStackTrace();
                }
        }
}
```

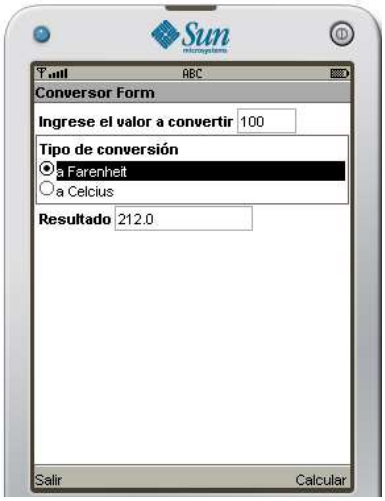
9. En el paquete visual crear una clase llamada ConverterForm

   package visual;

```
package visual;
import java.rmi.RemoteException;
import javax.microedition.lcdui.Choice;
import javax.microedition.lcdui.ChoiceGroup;
import javax.microedition.lcdui.Command;
import javax.microedition.lcdui.CommandListener;
import javax.microedition.lcdui.Displayable;
import javax.microedition.lcdui.Form;
import javax.microedition.lcdui.TextField;
import conexion.ConversorWS_Stub;
import control.ConverterMID;
public class ConverterForm extends Form implements CommandListener, Runnable{
          private TextField tfConversor;
          private ChoiceGroup selector;
          private Command salir;
          private Command calcular;
          private TextField tfResultado;
          private ConverterMID mid;
          private int index;
          public ConverterForm(ConverterMID mid) {
                    super("Conversor Form");
                    this.mid = mid;
                    salir = new Command("Salir", Command.EXIT, 0);
                    calcular = new Command("Calcular", Command.OK, 0);
                    tfResultado = new TextField("Resultado", "", 10, TextField.ANY);
                    tfConversor = new TextField("Ingrese el valor a convertir", "", 4,
                    TextField.ANY);
                    selector = new ChoiceGroup("Tipo de conversión", Choice.EXCLUSIVE, new
                    String[]{"a Farenheit", "a Celcius"}, null );
                    selector.setSelectedIndex(0, true);
                    append(tfConversor);
                    append(selector);
                    append(tfResultado);
                    addCommand(salir);
                    addCommand(calcular);
                    setCommandListener(this);    }
          public void commandAction(Command c, Displayable d) {
                    if(c==calcular){
                              Thread t = new Thread(this);
                              t.start();}
                    else if(c==salir){
                              mid.salir();}}
          public void run() {
                    index = selector.getSelectedIndex();
                    ConversorWS_Stub conversor = new ConversorWS_Stub();
                    float valor = Float.valueOf(tfConversor.getString()).floatValue();
                       tfConversor.setString("");
                       switch (index) {
                       case 0://Seleccion de celcius a Fahrenheit
try {System.out.println((Float.toString(conversor.celsiusToFarenheit(valor))));
tfResultado.setString((Float.toString(conversor.celsiusToFarenheit(valor))));
                              } catch (RemoteException e) {
                               break;
                       case 1://Seleccion de fahrenheit a Celcius
                              try {
tfResultado.setString((Float.toString(conversor.farenheitToCelcius((valor)))));
                              } catch (RemoteException e) {
                                     e.printStackTrace();
                              }
                              break;
                       default:
                              break;
                       }
               }
       }
```

10. Finalmente guarde el proyecto y haga clic derecho sobre la clase ConverterMID.java y a continuación clic en Run As y finalmente seleccione Emulated J2ME MIDlet.

11. En el Emulador ingrese valores para convertir y a continuación haga clic en calcular.



Referencias:

- En internet consulte: “JSR 172: J2ME™ Web Services Specification”:

http://jcp.org/en/jsr/detail?id=172

- En internet consulte: J2ME Web Services APIs (WSA), JSR 172:

http://java.sun.com/products/wsa/