Creating applications for OpenStage 60/80 – OpenStageSDK intro

1 This document in a nutshell

OpenStage phone has several APIs (application programming interfaces) for: creating GUI (graphical user interface), direct key input, audio streaming and push. OpenStageSDK makes it easier to use these APIs, because it unifies them into a single library. Moreover, this SDK is written in such a way that only elementary programming knowledge is needed to start development. It is free and open source software and was made using such software. To show capabilities of the SDK, the example instant messaging application *Open IM* (also open source) was written. There are also several very simple examples available.

2 OpenStage as a client

Introduction

OpenStage phones can become clients in many client-server applications. Client-server is a model of application, in which the data is processed and stored on the remote machine, which communicates with many clients (other machines) via a network.

Even if the clients do not communicate with each other directly, they can easily send information to the server, which can then send this information to some other client. Main role of the server is to store and process data, listen for requests coming from clients and respond to them. Main role of the client is to receive the data, display it, and send back data modified by the user. E-mail and instant messaging are very good examples of client-server systems.

Receiving

Phone behaves somewhat like a web browser, which can display web pages that were sent to it. OpenStage will interpret and display not HTML, but XML documents. In syntax they are very similar to an ordinary web page, but because they define specific user interface elements, there is a separate specification for them.

OpenStage can also receive HTTP POST requests, which can be sent by the server to launch a specific client application. The requests can be customized to for example not to launch application if it is already running, or to force the change the phone's active application to the one desired by the server.

Sending

The phone can send HTTP GET requests to any remote machine it is connected to, provided it is given its address. Such requests are expected to deliver a XML document in return. GET requests are used to launch an application, and they are used to interact with the server while the application is running. Example of such request:

```
http://www.site.com/bin/app?phonenumber=48224113595&ipaddress=192.168.53.1
```

GET request is also what the web browser is sending to the servers to receive a web page. In that manner, OpenStage phones are very similar to the web browsers, because they also simply request a page that the user wants to display.

Of course, GET requests may contain variables, like, in the example above:

```
phonenumber=48224113595
```

Such variables may be used by the server to create the page dynamically, or to even contact a database before creating a page, which will for example list recently received messages for phone number 48224113595.

3 Required features of the server

Basic requirement

HTTP (optionally HTTPS) server is needed to communicate with OpenStage on the application level.

Receiving

Machine that wishes to become a server for an application compatible with OpenStage phone, should be capable of listening for GET requests, and for every such request, it should generate a XML document, which is valid according to the specification.

```
http://www.site.com/bin/PhoneCheck?phonenumber=48224113595
```

For the request above to complete successfully, the script called PhoneCheck must under any configuration of variables generate a valid XML document.

Sending

The server can define what will be displayed on the screen of the phone. List of elements includes: static text, forms with editable fields, alerts, images, tickers, buttons, media player/recorder and commands to navigate between various user interface elements, or send data that was entered by the user back to the server.

The client application can not only be launched from OpenStage applications menu, but also remotely from the server, because the phone has the capability to listen for requests from the server and react to them (it is called push). Application, while launched, can for example turn on the LED of the phone or play an audio file to inform the user about that fact.

Before the application can be launched, it has to be defined on the phone, and for that configuration the location of the server side of the application is needed. There are specific constrains on the kind of content that the OpenStage phone can receive from, and send to the server.

4 Introduction to OpenStageSDK

What is SDK

SDK (software development kit) is a piece of software that makes creating other software easier. OpenStageSDK is aimed to make creating XML applications easier. It is written in Java – one of the most cross-platform programming languages. It has all functions mentioned in the official specification of OpenStage's XML, as of September 2011.

Advantages of SDK

The first advantage of OpenStageSDK over manually created XML files is: the SDK creates the XML files dynamically. Second is that while using OpenStageSDK, programmer may use any other Java library. These two together enable creation of large, complex applications that use databases, are spread among several servers, or simply must be very flexible with the content.

The XML documents sent to OpenStage device need to be valid, otherwise error message will be displayed instead of what was sent, the communication between server and client will be interrupted, and the application will need to be restarted. OpenStageSDK implements handling of the most common and recoverable errors, and provides detailed feedback in case the validation error could not be automatically corrected.

Moreover, SDK includes easy-to-use push request constructor and sender, with which you can issue POST requests to OpenStage devices without knowledge of POST request syntax. As a bonus, several examples of simple applications are included with the SDK.

Documentation and licensing

OpenStageSDK is an open source project, which comes with source code and complete documentation. The documentation system called Javadoc is an integral part of Java language. Javadoc is in fact a group of HTML pages generated directly from the source code.

SDK is published under Apache License 2.0, which basically means that you may do whatever you want with it without any costs, as long as you leave the original copyright notice intact. However, before using it for commercial purposes, please consult the Apache Foundation web page.

5 What is needed to use the SDK

Any environment that is capable of running Java, is an environment that can use OpenStage SDK. Below, the key elements of such environment are described.

Java

Sun/Oracle JRE (Java run-time environment) will be enough, however JDK (Java development kit) is strongly recommended, as it is a JRE that comes with documentation and other useful tools. OpenStage SDK was tested on JDK 1.6 and 1.7. These are also called Java SE SDK (Java standard edition software development kit) . Therefore, this or a later version is recommended.

To create basic XML applications for OpenStage phones, basic knowledge of Java programming language is required. Only your skill in Java programming can limit your ability to develop applications for OpenStage phones.

No knowledge of JRE or JDK is required, as these tools do not to be configured at all, but knowledge of how they function, what is the difference between them, and what they are needed for, is recommended.

Web server

The server that is capable of running Java HTTP Servlets (for example Tomcat) is needed. Java servlet is the simplest solution that allows attaching a Java library (namely the SDK) to an entity that is capable of listening and broadcasting HTTP GET and POST requests.

6 Complete environment example – Tomcat, Eclipse & MySQL

The below-mentioned tools are recommended by the author of SDK, as they are available for both Windows and Linux systems, and it is probably the most straightforward set of programs, which has many useful features while being very simple to configure. However, if you prefer other tools, you are free to choose something else.

Tomcat

The Apache Tomcat web server is required to make the XML applications accessible from another machines, including VoIP phones. The OpenStage SDK was tested on version 7, therefore this, or a later version is recommended.

Only basic knowledge of Tomcat functions is required to make even quite complicated applications for OpenStage phone. Also, no or very little configuration is required to start the server.

Eclipse

Eclipse for Java EE (enterprise edition) developers is required. It is needed because this version can be integrated with a web server, and that is needed to publish web applications. Also, it supports creation of HTTP (hypertext transfer protocol) Servlets, and that is also needed to create an XML application that can communicate with OpenStage phone through HTTP.

Eclipse is also recommended because of its good support of Javadoc (Java documentation) – a special documentation format that is an integral part of Java. It also allows one to access documentation from

source editor without losing context of what one is doing. OpenStage SDK comes with complete documentation in Javadoc.

Moreover, Eclipse can integrate with a Tomcat server in a few clicks, and then there is no configuration needed to deploy application manually

Basic knowledge how to use Eclipse is required to create basic XML applications. However, some knowledge about the most useful and advanced features of Eclipse makes it a lot easier (and faster) to program in Java.

MySQL

MySQL is a database system. Database is an optional element, however it is a necessary for every larger application. MySQL was chosen from many available database systems because of ease of use.

Quick step-by-step guide for launching Open IM (or any application available as a ".war" file), for Linux and Windows:

1. Download and install Sun/Oracle Java SE SDK, or JRE, version 6.
2. Download, install and launch MySQL server, version 5.1 or later:
   a) Linux:
      1. Using repository system for your distribution, install mysql-server package, or if your distribution does not have it in the repositories, download source from dev.mysql.com then compile & install it manually.
      2. Ensure that mysql processes are running, by entering **`ps -e|grep mysqld`** to console. There should be at least two processes. If there are less than two, use **`mysql`** command to launch the processes.
      3. Find and launch script **`mysql_secure_installation`** – it may be for example in */usr/bin* or in */usr/local/bin*
      4. Use command **`mysql -u root -p`** to launch mysql manager.
   b) Windows: use installer, then launch server manager to gain access to be able to enter SQL query.
3. Create a database on your server, with SQL query:

   ```
   CREATE DATABASE database_name;
   ```

4. Download and extract Apache Tomcat version 7. You may need to configure it so that it knows where you've installed JRE/JDK.
5. Place OpenStageSDK.jar and MySQL connector (available for example at http://dev.mysql.com/downloads/connector/j/) in *lib* sub-directory of Tomcat folder.
6. Place OpenIM.war in *webapps* directory inside Tomcat.
7. Go to Tomcat main directory and launch it:
   a) Linux: *bin/startup.sh*
   b) Windows: *bin/startup.bat*
8. In your web browser, go to this page:

   ```
   http://localhost:8080/OpenIM/Inbox
   ```

   You should substitute localhost with your computer's IP (actual internal/external IP). If the page displays, proceed. If it doesn't, check if the Tomcat process is running, and check under which port it is running, as it may not run on the default 8080 port.

7. Configure the OpenStage phone accordingly. From the example above, set *HTTP Server address* to the computer's IP, *HTTP Server port* to "8080", and *Program name on server* to

"OpenIM/Inbox".

8. Launch application from your phone - it should be displayed. If not, check if there really is a connection between your machine and the phone, also check if relevant port (8080 in the example) is not blocked. Start with entering the internal IP of your computer if you are having difficulties with the external.
9. Open IM will request you to configure a database, as well as register as the IM user after the database is properly configured.

Quick step-by-step guide for a new project, for Linux and Windows:

1. Download and install Sun/Oracle Java SE SDK version 6.
2. Download and extract Eclipse for Java EE developers.
3. Download and extract Apache Tomcat version 7.
4. Place OpenStageSDK.jar in *lib* sub-directory of Tomcat folder.
5. When in directory, to which Eclipse was extracted, launch Eclipse.
   a) Linux: *./eclipse*
   b) Windows: *eclipse.exe*
6. Create new dynamic web project (Alt+Shift+N -> Dynamic Web Project).
   1. Inside the creator window, click <New runtime>, then select Tomcat v7.0, then in the next step, <Browse> to the directory to which you have extracted Tomcat.
   2. Other fields may be left with default values.
7. Add a servlet to your project (Ctrl+N -> Web -> Servlet). Enter a class name and click Finish.
8. At the top of your code, near other imports, add:

```
import java.io.PrintWriter;
import pl.mbdev.openstage.*;
```

9. In the *doGet(...)* method of the servlet, place the following code:

```
// get the writer to which the XML data will be written
PrintWriter out = response.getWriter();
// create a screen with ID = 1
IppScreen s = new IppScreen(1);
// then, add a text box to the screen
s.add(new IppTextBox("Hello world!", "Indeed, hello hello.", "localhost", "text"));
// send the screen to OpenStage device
s.sendTo(out);
// data was sent, close the output
out.close();
```

10. Run your project (Ctrl + F11). In the wizard window, select your Tomcat server and Finish.
11. In your web browser, you should see the generated XML content. To access this application from your phone, you need to add new XML application to your phone. If what you see in the browser is:

    `http://localhost:8080/YourProject/ServletClass`

    You should set *HTTP Server address* to your computer's IP (not localhost, but actual internal/external IP), *HTTP Server port* to "8080", and *Program name on server* to "YourProject/ServletClass".

12. Launch application from your phone - it should be displayed. If not, check if there really is a connection between your machine and the phone, also check if relevant port (8080 in the example) is not blocked. Start with entering the internal IP of your computer if you are having difficulties with the external.

7 Alternative environments

Below are some examples of other environments. The list is of course not complete.

Java EE SDK with Glassfish

Glassfish is a server that is shipped in Java EE package. It is not present in Java SE environment. It is possible to integrate Eclipse with Glassfish server, if one needs the sophisticated enterprise libraries that come with Java EE run-time. Since the setup is not totally obvious, it is a not recommended tool unless you know that your project needs Java Enterprise Edition.

JDeveloper Studio, with WebLogic server

The only program needed in this solution is a huge application JDeveloper Studio, version 11g or later. JDeveloper is a freeware IDE designed for creating Java EE applications. It is generally not recommended as a server for OpenStage applications, because it takes too much HDD space and other important system resources like processor time and memory, and moreover response time of the environment is quite long. Nevertheless, below there is a:

Step-by-step guide for creating new OpenStage application in JDeveloper:

1. Download and install JDeveloper Studio from the Oracle web page. Make sure that among the installed elements there is „Web Logic server".
2. Launch JDeveloper. In the window named „select role" choose „all features".
3. Create new application of type Java EE (Java Enterprise Edition) including the HTTP Servlet:
    1. Click „New Application", select „Java EE Web Application".
    2. Notice, that the list on the right contains two projects, „Project 1" and „Project 2" respectively. The first one may be left with default settings, the second one should be given a name you wish for your project.
    3. After creation, click with RMB (right mouse button) on the name of your project, and select „New..."
    4. Go to category named „Web Tier", subcategory „Servlets" and select „HTTP servlet".
    5. In step 1. in the field „Generated Content Type" choose XML.
    6. In step 2. in the field „Name" enter name of your program/service. It is also good to enter a meaningful value to the field „URL Pattern".
4. Attach OpenStage SDK to your project.
    1. Extract OpenStageSDK.jar to the place of your choosing.
    2. Click with RMB on the name of your project, the one for which a Servlet was created. Select „Project Properties".
    3. From the list choose „Libraries and Classpath". Click „Add Library". For „Location", choose „User". For all 3 fields: „Class Path", „Source Path" and „Doc Path" choose „Add Entry" and find OpenStageSDK.jar. Hint: it has a folder icon on the list of files.
    4. Select „Deployed by Default".

8 References

Complete specification for XML applications:
"OpenStage 60/80 - XML Applications, Developer's Guide"

written by Mateusz Bysiek, mb@mbdev.pl, http://mbdev.pl/