Conversion of Java Applets Animations to Java MIDlets

Kapil Kadam, Sameer Sahasrabudhe

Abstract—Java applets are used in e-learning, mainly for creating animations. Many leading educational and professional organizations have large repositories of animations, created using Java applets which are mostly for desktops. Use of mobile phones is increasing especially amongst students, and creating new e-learning content for mobile phones is resource intensive. The Java applets available are not compatible with the mobile technology. It is desirable to have a methodology to adapt existing Java animations on mobile phones. This paper presents a methodology (APP2MID) which converts the Java applet into Java MIDlet that can run on mobile phones. The conversion methodology is divided in three logical phases and will help the user to create MIDlets from the available Java applets maintaining the original programming logic. We illustrate this methodology using a case study.

I. INTRODUCTION

Java applets have been a one of the popular content creation format for eLearning. Many leading organizations in educational and professional domains have created large repositories of educational animations created using JAVA [4].

Concurrently, mobiles are becoming even more common and have been having lesser dependencies. The portability and the reach of mobiles is quite more than that of the computers and therefore it is not at all surprising that mobiles are preferred as a medium for educational content dissemination [1], [2].

Looking at the large number of Open Source JAVA applets available for the desktops, it is desirable to have a methodology to convert those for mobiles in a seamless manner. In this paper we present one such methodology: APP2MID.

Java SE (Java standard edition) is used for creating applets especially for desktops. These applets cannot run on mobile phones because the APIs used are not supported by mobile phones. For creating applications for mobile phones and other handheld devices J2ME (Java 2 Micro edition) technology is used [6]. J2ME represents a simpler version of Java SE, with a reduced feature set, for the devices with limited memory, small screen sizes, alternative input methods, and slow processors [7].

TABLE I

<table>
<thead>
<tr>
<th>Package</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>java.lang</td>
<td>Boolean, Byte, Character, Class, Integer, Long, Double, Float, Math, Object, Runnable, Runtime, Short, String, StringBuffer, System, Thread, Throwable</td>
</tr>
<tr>
<td>java.io</td>
<td>ByteArrayInputStream, ByteArrayOutputStream, DataInputStream, DataOutput, DataOutputStream, DataOutputStream, InputStream, OutputStream, InputStreamReader, OutputStreamWriter, PrintStream, Reader, Writer</td>
</tr>
<tr>
<td>java.util</td>
<td>Calendar, Date, Enumeration, Hashtable, Random, Stack, TimeZone, Vector</td>
</tr>
</tbody>
</table>

From the earlier discussion following points are evident:

1. Available Java applets animations are huge in number.
2. J2ME does not support all the APIs that are used in Java applet animations creation
3. Re-creation of same animations for mobiles would be resource intensive.
4. Maintaining the original programming logic of the animation is critical.

Seamless conversion of applets to MIDlets is certainly desirable. In this paper we present one such methodology. A case study is used to illustrate the methodology in detail.

II. CONVERSION PROCESS: APP2MID

The conversion process APP2MID has three logical phases. These phases convert the original Java applet into various formats in a cascading manner to achieve the final goal of creating a MIDlet. The Java applet is not converted directly into MIDlet, as it may create problems for code extraction.

The first phase converts the Java applet to a XML file for better representation and mapping. The second phase transforms this applet XML to a MIDlet XML. The third phase converts the MIDlet XML to a MIDlet. Finally testing is done, to check the quality of the converted MIDlet. Refer...
Fig. 1: Conversion Phases

A. Conversion Phase-I

This phase has Java applet as the input. This applet is converted to XML using the available Java to XML converter (Java2XML) [9]. After conversion the applet XML is validated by using a browser (Internet Explorer) [3].

Java and XML both are platform independent and XML is fully compatible with Java. Standard Java class structure can be well represented in the form of XML tags, so that Java classes, methods, interfaces, packages are processed easily by maintaining the Java programming standards [5].

The produced representation allows easy extraction of the semantics of source code, its analysis and transformation. The output of this phase is a valid XML of the applet. Schematic representation of this phase is shown in Fig. 2.

B. Conversion Phase-II

The second phase has the output of the Phase-I as the input. The motivation of this phase is to transform the applet XML into MIDlet XML. This enable mapping of applet XML into another XML to obtain the desired MIDlet structure.

In order to address this we wrote a Java program (APP2MIDconverter.java), containing class “XML Traverser” which does the XML to XML transformation. This created Package.xml file which contained XML representation of basic MIDlet program structure. The transformed XML is validated in a similar way as mentioned in the Phase-I.

C. Conversion Phase-III

The third phase has the output of the Phase-II as the input. The motivation of this phase is to convert MIDlet XML into Java MIDlet which is the desired final output.

In order to achieve this we wrote another class “Converter” in the “APP2MIDconverter.java” program mentioned in Phase-II. This class “Converter” does the MIDlet XML to Java MIDlet conversion. The output of this phase is a desired Java MIDlet. Schematic representation of this phase is shown in Fig. 4.

D. Testing Phase

Testing is the most important way of assuring (or controlling) the quality of software or application developed. The third phase delivers the Java MIDlet as final output.

The resultant Java MIDlet is tested with Sun’s Wireless toolkit emulator [10] by selecting MIDP2.0 & CLDC1.1 APIs, that gives the desired output, the actual results are presented in further sections. An error free result leads towards the actual device testing. Mobile phones like Nokia N72 [12] & Sony Ericsson W58i [13] were used for testing as they are supported by MIDP 2.0 & CLDC1.1 APIs.

III. Case Study

CONVERSION OF ANIMATIONS IN PROJECT OSCAR

Project OSCAR stands for Open Source Courseware Animations Repository [11]. This is a project of IIT Bombay and hosts over hundred animations, which covers different subjects from standard 7th to standard 12th like Mathematics, Physics, Biology and Chemistry. It also covers Engineering subjects for Undergraduate and Post Graduate studies like Networking, Mechanical engineering, etc.

Most of the animations from the repository are Java applets, which is suitable for the conversion methodology presented in this paper.

We have selected Basic Angles animation from Mathematics repository as a test case [11]. BasicAngles.java is a Java applet program which contains the code for the basic angles animation. Fig. 5 shows the output of BasicAngles.java applet when executed in applet viewer and Fig. 6 shows the small portion of the code.
A. Phase-I Results

As mentioned in the conversion phase-I the input for this phase is Java applet program i.e. BasicAngles.java (refer Fig. 6) which is converted into applet XML. The output of conversion phase-I is shown in Fig. 7 which shows the small portion of the XML applet.

As mentioned in the conversion phase-I about standard Java class structure & XML tags, it can be seen from the Fig. 7 that the standard class structure of Java applet (refer Fig. 6) is well represented in the form of XML. For example class “BasicAngles” with visibility attribute “public” (Fig 6) is represented as a XML tag

```xml
<class name="BasicAngles" visibility="public">
```

in Fig 7. Other code of Java applet was also converted in the form of XML tags. That leads to easy extraction of the semantics of source code, its analysis and transformation as said earlier.

B. Phase-II Results

As said earlier the input of the second phase is the output of the Phase-I, which is the applet XML (Figure7). After processing through conversion phase-II we got the MIDlet XML (Fig 8), which shows the portion of MIDlet XML. At the end of conversion phase-II we get the XML representation of MIDlet.

The desired output at this phase is XML representation of MIDlet. The transformation adds packages required for MIDlet. For example package “javax.microedition.midlet” represented as XML tag

```xml
<import module="javax.microedition.midlet"/>
```

in the MIDLXML (Fig. 8).

C. Phase-III Results

The output of phase-II i.e. MIDlet XML is the input for phase-III. The XML representation of desired MIDlet intends to convert this into Java MIDlet. Every tag of this XML is processed and converted into Java MIDlet format maintaining its standards.

For example XML tag

```xml
<import module="javax.microedition.midlet"/>
```

from Fig. 8 is converted to following Java statement of MIDlet.

```
import javax.microedition.midlet;
```

as shown in Fig. 9.
D. Testing Phase Results

We tested the resultant Java MIDlet on Sun’s wireless toolkit emulator, which is mainly used to test J2ME applications. The MIDlet worked without any errors. The results are shown in Fig. 10.

![Java MIDlet Animation on Sun’s wireless toolkit emulator](image-url)

IV. CONCLUSION

The outputs of the case study of the Basic Angles animation applet, confirms the success of the conversion methodology: APP2MID. Some more applets were also tried and were converted successfully. The creators of Project OSCAR appreciated the conversion, and are planning to implement it for expanding the scope of their project to mobile phones.

On critical review of the converted applets we have found out the following aspects, which can be addressed in the future:

- It is necessary to try and convert all possible Java SE classes used in animation.

- The MIDlet created using this methodology was of the same screen size of the applet. This feature made it mandatory to the user to use scrolling to view the animation on the smaller screen size of mobile phones.

- The MIDlet created using this methodology will be compatible only with the mobile handsets with API support of MIDP 2.0, and CLDC 1.1.

REFERENCES


[10] Harsh Jain, “Java to Xml, URL:https://java2xml.dev.java.net

