

Java Verified™ Program

Technical Topics

Topic 2: Internationalization and Testing

To reach a much wider market, J2ME MIDP developers should consider making their applications multi-lingual. Many operators are multi-national and would naturally look for applications that could operate in the countries of their markets. But how should a multi-lingual application be built? Should there be one version for each language? And how should it be tested, especially if the only difference is perhaps just the text that is displayed? Once? Or as many times as there are languages? Each test takes time and the more time spent testing, the more the testing will cost. This topic discusses one technical approach to the Internationalization (I18N) of a MIDP application. These should make a developer's application more attractive to distributors and it should result in lower testing costs.

Internationalization (I18N)

From its beginning, the Java language was designed so that programs could easily work with different languages, alphabets, date formats, etc. In addition to language support, a set of classes were developed to allow a programmer to make his/her program multi-lingual - without requiring a rewrite of the code for each language. These internationalization classes help J2SE and J2EE developers. However many of the classes have never been part of the J2ME platform, especially the combination of MIDP using the CLDC.

There is an I18 technique used by J2SE and J2EE programmers that MIDP developers can follow: it is to place all locale-specific text in a separate file – usually called a resource file. That is, rather than having literal strings in your code, they are placed in the resource file and accessed by name. To change languages, for instance, your application needs only to access a different resource file. Importantly, the code does not change. In the section below on testing this can result in lower testing costs. Also, please, see the pointer below for a discussion on making applications "world aware".

JSR 238

There could be more class support for the internationalization of MIDP code. To this end, JSR 238 - Mobile Internationalization API - proposes a new package: the Mobile Internationalization API. This optional package would be used with MIDP running on CLDC. As of November 2004, JSR 238 is in public review. The reader is urged to study the JSR, make comments and begin understanding what APIs will be available for writing multi-locality applications. The JSR specifies new classes that are similar to those in J2SE but designed with the limited resources of MIDP devices in mind.

One requirement of the JSR is the use of a ResourceManger. This class must be used to accessed device- or application-specific resources. All resources will be stored in binary resource files. These files must be stored in the application's JAR file. There is also, a specific format format for the resource files. This is, of course, similar to the resource file technique described above.

Testing Multi-lingual Applications

When an application has been tested by one of the authorized Java Verified Program testers and it passes, it is signed using a Public Key Infrastructure (PKI) certificate. The signature is based on the contents – every byte – of the JAR file. So, if there are two versions of an application – one French and the other German – then the testing houses must treat these two as completely different applications. This is done to make sure that the application has not changed – other than what is expected in different language versions. So the testing of two versions of the application could be almost twice the cost of testing one version.

If, however, the class bytes are identical in the two versions and the only differences are between the resource files then the testing house can reduce the number of testing criteria checks: less testing means a lower cost. Below is a list of the testing criteria that need to be checked for each language version. For a description of each criterion, please refer to the latest [Unified Testing Criteria document](#).

| <i>Testing Category</i> | <i>Tests</i> |
|-------------------------|------------------------------|
| Application Launch | AL1, AL2, AL3 |
| User Interface | UI1, UI2, UI3, UI4, UI6, UI9 |
| Localization | LO1, LO2 |

Summary

In summary, developers should use resource files, when possible. Literals (text) should never be hardcoded in their applications. The literals should be defined once, given an identifier and referenced by that name in the code. These literal definitions should be placed in a separate resource file and placed in the JAR. If these techniques are followed it should

- increase the application's marketability by its having the ability to switch languages easily and
- reduce the overall cost of testing a multi-lingual application.

It is important to note that each Testing House sets its own policy and pricing on language version testing.

Resources

1. <http://developers.sun.com/techttopics/mobility/midp/ttips/worldaware/> Writing World-Aware J2ME Applications.
2. <http://www.jcp.org/en/jsr/detail?id=238> JSR 238 – Mobile Internationalization API. The JSR is in public review
3. A global search using "midp resource file" will return a number of articles on how to utilize resource files.