Mobile Information Device Profile (MIDP)

Mobile Information Device Profile (MIDP) is a specification published for the use of Java on embedded devices such as mobile phones and PDAs. MIDP is part of the Java Platform, Micro Edition (Java ME) framework and sits on top of Connected Limited Device Configuration, a set of lower level programming interfaces. MIDP was developed under the Java Community Process as JSR 37 (MIDP 1.0) and JSR 118 (MIDP 2.0). As of 2007, MIDP 3.0 is being developed under JSR 271. The first MIDP devices were models i80s and i50sx from Motorola, launched in April 2001.

Noteworthy Limitations

- MIDP 1.0 has no active rendering APIs
- MIDP 1.0 has no support for direct access to image pixels (RGB data)
- MIDP 1.0 has no support for full screen mode
- MIDP 1.0 has no support for audio.
- MIDP 1.0 only requires HTTP support.
- MIDP 1.0 cannot query key status (although key events are supported)
- The specifications are not always clear, leading to differences in implementations.
- Some limitations may be avoided by using a vendor-specific API or MIDP 2.0, which obviously reduces the portability of the application.

General APIs

The core APIs are defined by the underlying Connected Limited Device Configuration.

javax.microedition.io

Contains the Java ME-specific classes used for I/O operations.

javax.microedition.lcdui

Contains the Java ME-specific classes used for the GUI.

LCDUI has a simple screen based approach where a single Displayable is always active at a time in the application user interface. LCDUI API provides a small set of displayables common in mobile device user interfaces: List, Alert, TextBox, Form and Canvas. For all displayables the device MIDP implementation has control over the presentation and layout of the displayable. Canvas is a low-level graphics surface for which an application has full control over what is rendered to it, although normally some space is reserved for system areas like screen title and indicators common in mobile device UIs. Since MIDP 2.0, Canvas also supports a full-screen mode that allows to make full screen graphics, which is especially useful for games.

LCDUI also has quite unique approach of abstract operations, called Commands. The placement of commands added to a displayable is completely up to the device implementation of this toolkit. The application programmer uses API specified command types to indicate the usage or purpose of the command in application user interface. Common types are BACK, EXIT, ITEM, SCREEN. The idea of the command abstraction is to make applications more portable between various different mobile device. Application developers should use the command types properly to indicate the purpose of an operation, and device implementation then places the operation to the common location for a given type in device's specific user interface style. This may be e.g. a specific key, like "a back navigation key" for BACK commands or button on screen.

The term LCDUI was actually a joke in JCP Expert Group that created it. It has not been opened up in the MIDP specifications but stands for Limited Capability Device User Interface. The joke was that no-one else really knows what it stands for. Then later the Programming Wireless Devices with the Java 2 Platform, Micro Edition book gave this term out.

_ Other common definitions have appeared. "LCD UI" would reflect the fact that mobile phones normally use LCD displays; however, the API is not specifically tailored to this particular display technology. It is also said that "LCD UI" stands for "lowest common denominator" due to the fact the specific UI has simplest possible design.

javax.microedition.rms

Main article: Record Management System

Provides a form of persistent storage for Java ME.

javax.microedition.midlet

Contains the base classes for Java ME applications.

Specialized APIs added in MIDP 2.0

MIDP 2.0 saw the introduction of gaming and multimedia APIs and some optional packages.

(C) Copyright Nokia 2014. All rights reserved.

javax.microedition.media

Contains the base classes of the multimedia playback. These are approximately a subset of the JSR 135 Java Mobile Media API.

javax.microedition.lcdui.game

A gaming API aimed at simple 2D sprite based games.

javax.microedition.pki

Authenticate APIs for secure connections.

javax.microedition.messaging

Wireless messaging API (optional), sending SMS and MMS messages.

javax.microedition.pim

Personal information management API (optional), access the device's address book.

javax.microedition.io.file

The File Connection Optional Package (FCOP) is one of two optional packages defined by JSR 75 through the Java Community Process. The FileConnection API specified in JSR 75 gives access to the local file systems on devices like PDA. In order to overcome security issues MIDlet needs to include requested file permission in its JAD file under MIDLet-Permission property.

Development Tools

There are several different ways to create MIDP applications: Code can be written in a plain text editor, or you can use a more advanced IDE such as NetBeans or Eclipse (with the appropriate plugins) which has a user interface for graphically laying out any forms you create, as well as providing many other advanced features not available in a simple text editor.

(C) Copyright Nokia 2014. All rights reserved.