

**Sun Certified Mobile Application Developer (311-110)**

**Sun Microsystems, Inc.**

**Exam Notes**

**Sathya Srinivasan**

**02/16/2004**

## Table of Contents

Browsing MIDlet suites.....	3
Transferring MIDlet suites.....	3
Using HTTP.....	3
Push Registries.....	3
Basic Authentication.....	3
Installing MIDlet suites.....	3
Updating MIDlet suites.....	3
Deleting MIDlet suites.....	4
MIDP execution environment.....	5
MIDlet suites.....	5
MIDlet suite packaging.....	5
Device Service Discovery.....	5
Version Discovery.....	5
Purpose of MIDlet class.....	6
MIDlet communication with AMS.....	6
Platform request API.....	6
Valid MIDlet states and transitions.....	7
MIDlet Lifecycle behavior.....	7
Mandatory attributes.....	8
Optional attributes (Typical).....	8
Optional Attributes (Non-Typical).....	8
Description.....	9
Timer and TimerTask.....	10
Method Arguments.....	10
Push Registry activities.....	10

## Chapter 5 Application lifecycle provisioning

---

Explain the specification guarantees for: browsing for MIDlet suites, transferring MIDlet suites, using HTTP, push registries, basic authentication, installing and updating MIDlet suites, invoking MIDlet suites, and deleting MIDlet suites.

### Browsing MIDlet suites

- The Application Management Software must provide facilities for the user to browse the installed MIDlets.

### Transferring MIDlet suites

- User can cancel a MIDlet suite installation at any time without affecting anything.
- Appropriate permissions will be asked from the user if not defined in the descriptor.

### Using HTTP

- HTTP is used as a protocol for transferring the JAD and JAR files to the device.
- Appropriate header values should be set, like the content-type as described in other sections.

### Push Registries

- Static push registries required by the MIDlet suite should be defined using the `MIDlet-Push-<n>` attributes in the JAD file.
- If the requested resources for the push registry attributes are not available, the application **will not** be installed.

### Basic Authentication

- If the application requires basic authentication (401 - Unauthorized, 407 - Proxy authentication required HTTP responses during installation), the device should ask for approval from user.

### Installing MIDlet suites

- See other sections.

### Updating MIDlet suites

- If an application is already installed, then attempting to install the application is treated as an update the application.
- The MIDP implementation must notify if the installation is an older, newer, or same version as the suite already installed in the device and must get confirmation from user before installing.
- If the signer for the new and existing MIDlet suites are identical or if the JAD file or the JAR file of the new installation have the exact same path (URL) as that of the old set, then the RMS data must be retained. If not, the user must be given an option to retain the RMS data.
- An untrusted MIDlet suite cannot be an updated to a trusted MIDlet suite.

### **Deleting MIDlet suites**

- When a MIDlet suite is deleted, all MIDlets and RMS data associated with this suite will be deleted.
- User will be prompted for confirmation.
- Single MIDlet cannot be deleted. The most atomic part is a MIDlet suite.

Identify correct and incorrect statements or examples about the MIDP application model, including: the MIDP execution environment, MIDlet suites, MIDlet suite packaging (including the manifest and the application descriptor), discovering available services on the device, discovering which version of MIDP/CLDC is on the device.

### MIDP execution environment

- MIDlet suites must not contain classes contained in the CLDC or MIDP specification.
- Installed application cannot override system classes/packages.
- Class files cannot be read as resources or extracted for reuse.
- JAD properties can be read using the `MIDlet.getAppProperty()` method.
- Non-class resources in the JAR can be read using the `Class.getResourceAsStream()` method.

### MIDlet suites

- A MIDlet suite MUST contain at least one class that extends `MIDlet`.
- The MIDlet class MUST have a no-args constructor.
- A class should not contain a standard `main()` method. If present, the method must be ignored as the starting point for the application. The MIDlet class must be used instead. This is because the `main()` method is the starting point for a CLDC application, but is overridden by MIDP specifications.

### MIDlet suite packaging

- Each MIDlet suite is packaged into a SINGLE JAR file.
- One or more MIDlets may be present in a JAR file.
- The manifest file (`/META-INF/manifest.mf`) file inside the JAR file contains attributes used during the installation and execution of the suite.
- Sharing resources is managed at the API level and not at a suite level.
- A package contains a JAD file and a JAR file.

### Device Service Discovery

- **Details needed.**

### Version Discovery

- The MIDP specification can be obtained by calling the `System.getProperty("microedition.profiles")` or `MIDlet.getAppProperty("MicroEdition-Profile")` method if a JAD file is present.
- The CLDC specification can be obtained by calling the `System.getProperty("microedition.configuration")` or `MIDlet.getAppProperty("MicroEdition-Configuration")` method. This would typically be "CLDC-1.0" or "CLDC-1.1".

Develop applications that correctly reflect a MIDlet's application lifecycle, including: the purpose of the MIDlet class, communication with the application management software, platform request API, valid MIDlet states and transitions, and the behavior that should and should NOT be implemented within different lifecycle methods (including the constructor).

## Purpose of MIDlet class

- MIDlet class is the starting point for running a MIDlet suite.
- The class provides methods to the AMS for managing the lifecycle of the suite. This will be used by the AMS to create an instance of this MIDlet.

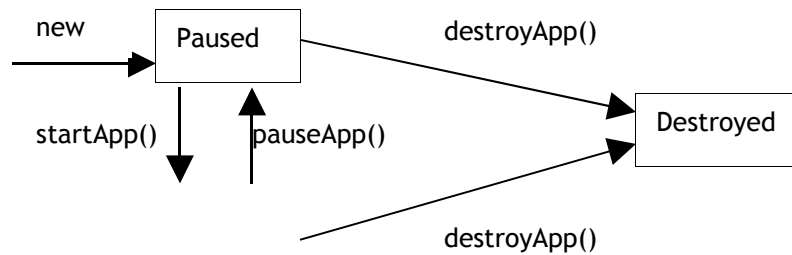
## MIDlet communication with AMS

Method	Description
<code>notifyPaused()</code>	The MIDlet is ready to be paused.
<code>notifyDestroyed()</code>	The MIDlet is ready to be destroyed.
<code>resumeRequest()</code>	The MIDlet is ready to run, typically because of a Push Registry or alarm request.

## Platform request API

- This is the method `MIDlet.platformRequest(String url):boolean` method.
- This indicates that the requested URL should be handled.
  - An application in the URL may need to be installed or data displayed.
  - The MIDlet maybe in background while this is being done. If this is not possible, the device may wait till the MIDlet exits and might then handle the device.
- This is a non-blocking method.
- On platforms where the device has to exit to honor this request, if multiple requests are made, only last request will be handled. If not, multiple requests must be handled concurrently.
- A very typical use for this request is to install an update of the running application.
  - If the requested URL refers to an update to this MIDlet, then this MIDlet must be stopped to install the requested MIDlet suite.
  - If the requested resource is a MIDlet but not an update, it should be handled like a new install. The requesting MIDlet may or may not be stopped based on the device capability.
- Another use is to make a phone call.
  - In this case, the URL should be of the form `"tel://msisdn-number"`
  - The request should be passed to the default phone application of the device.
  - Appropriate permissions must be requested by the user first.
- Other URLs may be supported later.

## Valid MIDlet states and transitions



- `MIDlet.resumeRequest()` method can be used by the application (usually because of a notification from Push Registry) to the AMS that it is ready to run again.

## MIDlet Lifecycle behavior

- `MIDlet` class must have a no-args constructor.
- `MIDlet` (or any other class in the suite) should not have a `static public void main(String [] args)` method. If present, the method will be ignored by the AMS.
- `MIDlet` must not call the `System.exit()` method. If called, a `SecurityException` will be thrown.
- Call the `MIDlet.notifyDestroyed()` after calling the `MIDlet.destroyApp()` method if you want to close the application from inside the application. This is the one that notifies that the `MIDlet` will be ready to be destroyed.

**Deploy a MIDP 2.0 application with the correct use of JAD files and manifests.**

### Mandatory attributes

- A JAD file does not have to be present to install a MIDlet. In such cases, the MIDlet is treated as untrusted.

Attribute	Description	JAR Manifest	JAD
MIDlet-Name	Name of the MIDlet suite.	Optional	Mandatory
MIDlet-Version	Version for this MIDlet suite.	Optional	Mandatory
MIDlet-Jar-URL	URL for the JAR file	N/A	Mandatory
MIDlet-Jar-Size	Size of the attached JAR file	N/A	Mandatory
MIDlet-Vendor	The name of the vendor providing the MIDlet.	Optional	Mandatory
MIDlet-<n>	Description for each MIDlet in the form MIDlet Name, {Icon-URL}, MIDlet class name	Mandatory	Optional
MicroEdition-Profile	The minimum MIDP configuration needed. Typically MIDP-2.0	Mandatory	Optional
MicroEdition-Configuration	The minimum CLDC configuration needed. Typically CLDC-1.0 or CLDC-1.1	Mandatory	Optional

### Optional attributes (Typical)

Attribute	Description	Where?
MIDlet-Description	A detailed description for the MIDlet suite.	Manifest or JAD
MIDlet-Icon	Optional icon for the MIDlet suite.	Manifest or JAD
MIDlet-Info-URL	URL where further information can be obtained about the midlet suite	Manifest or JAD
MIDlet-Data-Size	The size of the associated JAR file	Manifest or JAD
MIDlet-Permissions	Permissions for the MIDlet suite	Manifest or JAD
MIDlet-Push-<n>	Push Registry entries. Format is similar to MIDlet-<n> attribute.	Manifest or JAD

### Optional Attributes (Non-Typical)

Attribute	Description	Where?
MIDlet-Permissions-Opt	Optional permissions for the MIDlet suite	Manifest or JAD
MIDlet-Install-Notify	URL where the success of installation is to be notified. Uses POST method. HTTP protocol must be supported.	Manifest or JAD
MIDlet-Delete-Notify	URL where the success of deletion is to be notified. Uses POST method. HTTP protocol must be supported.	Manifest or JAD
MIDlet-Delete-Confirm	URL where the confirmation of deletion is to be notified. Uses POST method. HTTP protocol must be supported.	Manifest or JAD

Given an installation failure, analyze the problem, and develop possible resolutions.

Code	Message	Description
900	Success	MIDlet suite was installed successfully.
901	Insufficient memory	Not enough memory to install the MIDlet suite. <ul style="list-style-type: none"> <li>You can try reducing the MIDlet-Data-Size requirement if high.</li> </ul>
902	User cancelled	User cancelled the installation. <ul style="list-style-type: none"> <li>Don't cancel next time!</li> </ul>
903	Loss of service	Service for data transfer was lost. <ul style="list-style-type: none"> <li>Maybe cell reception was not good. Try again.</li> </ul>
904	JAR size mismatch	Size of JAR does not match size mentioned in the JAD file. <ul style="list-style-type: none"> <li>Correct the attribute value.</li> </ul>
905	Attribute mismatch	Attribute in JAR manifest does not match with JAD attribute. <ul style="list-style-type: none"> <li>Correct the attribute value mismatch (esp. MIDlet-Vendor, MIDlet-Name, and MIDlet-Version).</li> <li>If MIDlet is trusted, all MIDlet-* must match between JAR manifest and JAD entries.</li> </ul>
906	Invalid descriptor	Descriptor file was invalid. Bad entries, etc. <ul style="list-style-type: none"> <li>Correct the values.</li> <li>Maybe some mandatory values are missing.</li> </ul>
907	Invalid JAR	JAR file is invalid. <ul style="list-style-type: none"> <li>Manifest cannot be extracted from JAR.</li> <li>JAR manifest does not have correct syntax or mandatory attributes are missing.</li> </ul>
908	Incompatible configuration or Profile	The Configuration value (MicroEdition-Configuration attribute) or Profile value (MicroEdition-Profile) is invalid.
909	Application authentication failure	Application could not be authenticated properly.
910	Application authorization failure	The installation is not authorized. <ul style="list-style-type: none"> <li>Attempt was made to install an untrusted version of an installed, trusted suite.</li> <li>Permissions mentioned in the MIDlet-Permissions attribute cannot be satisfied.</li> </ul>
911	Push registration failure	The static push registry entries could not be honored.
912	Deletion Notification	Deletion notifications could not be sent successfully.

Given a set of requirements, develop applications that correctly implement MIDP 2.0 support for delayed or scheduled activities using timers and background threads.

## Timer and TimerTask

- These classes function roughly in the same manner as that of their counterpart classes in J2SE.
- The `TimerTask` is a `Runnable` class that can be used to do something upon being triggered by the `Timer` class.
  - The `scheduledExecutionTime()` returns the scheduled time of the last run of this task or the currently running task.
  - The `cancel()` method cancels this task if it has not run yet. If it is a repeating task, this will not run again.
- At least 5 `Timer` threads must be supported by a MIDP 2.0 compliant device.
- The `Timer` class is thread-safe.
- The `Timer.schedule()` methods can be used to schedule tasks.

Method Arguments	Description
<code>TimerTask task, Date time</code>	Schedules the task to be run at the specified time.
<code>TimerTask task, Date firstTime, long period</code>	Schedules the task to be run repetitively.
<code>TimerTask task, long delay</code>	Schedules the task to be run after a specified delay from the time of the call (in milliseconds)
<code>TimerTask task, long delay, long period</code>	Schedules the task to be run after a specified delay repetitively from the time of the call (in milliseconds)

- The `Timer.scheduleAtFixedRate()` methods can be used to schedule tasks at a fixed rate.
  - This is same as the `schedule()` method except that if one of the runs took a longer time, the successiver runs will be squished such that on an average, the correct number of runs happen (roughly 2 runs for a (2\*period) time and so on).

## Push Registry activities

- An application could be started because of a notification from the push registry. In such cases, upon notification, the application should call the `MIDlet.resumeRequest()` method to notify the AMS that it is ready to run.