MIDlet プログラミング入門

荒川靖弘

2002 年 6 月 30 日

目次

1　はじめに

本テキストは携帯端末に組み込むことのできる Java アプリケーション「MIDlet」について解説します。

前半では主に営業や SE 向けに組み込み機器用の Java2 プラットフォームである J2ME とその上位クラスライブラリ群である CLDC/MIDP について紹介します。後半ではやや技術者向けに CLDC/MIDP 上のアプリケーション実装である「MIDlet」のプログラミングについて触りの部分を解説していきます。

また本テキストでは様々な用語・略語が出てきます。付録 A 節に用語・略語の一覧を用意しました。参考にしてください。

2　携帯端末と J2ME

J2ME は Sun Microsystems が 1999 年に発表したコンピュータ電子機器および組み込み型装置を対象とした Java2 プラットフォームです。従来の Java の設計思想を継承しつつ組み込み用途向けにコンパクトにまとめられているのが特徴です。図 1 に従来の Java2 プラットフォームとの比較を挙げます。

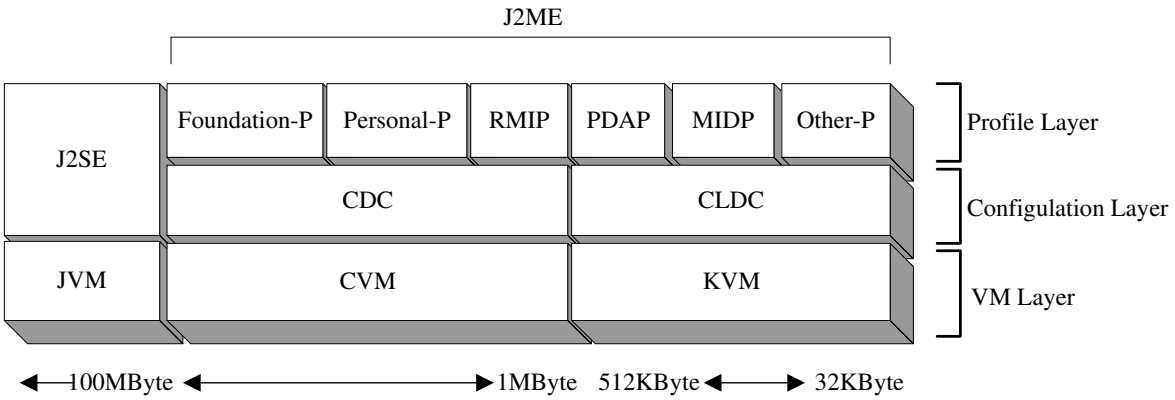

図 1　Java2 プラットフォーム

このように J2ME は主に 3 つのレイヤに分かれています。

仮想マシンレイヤ　Java 仮想マシンを実装したレイヤです。ターゲットとなるハードウェア（主に CPU パワーとメモリ容量）によって JVM，CVM，KVM の 3 つの実装があります。

コンフィギュレーションレイヤ　コンフィギュレーションレイヤでは各仮想マシンに対応した基本クラスライブラリを提供します。J2ME では CDC および CLDC の 2 つが定義されています。

プロファイルレイヤ　コンフィギュレーションレイヤの上位に位置し，各市場（コンシューマ機，PDA，携帯電話など）からの具体的な要求を実装したクラスライブラリ群です。MIDP は PDA や携帯電話などの「携帯端末」用に考えられたプロファイルです。

2.1 KVM/CLDC

KVM はメモリ容量や CPU パワーの少ない携帯端末などに組み込むことができるようにチューニングされた仮想マシンです。名前が示すように[*1]メモリ容量が KByte 単位で設計されているのが特徴です。また CLDC は KVM 用に調整された基本クラスライブラリです。

KVM/CLDC が動作するためのハードウェア制約は以下のとおりになっています。

- Java プラットフォーム用に利用可能な合計メモリ容量が 126~512KByte であること
- 25MHz 以上の 16/32bit CPU であること
- 省電力であること
- 何らかのネットワーク接続機能を備えていること
- KVM/CLDC に利用可能な 128KByte 以上の不揮発性メモリを備えていること
- Java ランタイムおよびオブジェクトメモリに利用可能な 32KByte 以上の揮発性メモリを備えていること

2.2 MIDP と MIDlet

MIDP は KVM/CLDC 用のプロファイルです。MIDP のハードウェア制約は以下の通りになっています。

ディスプレイ 少なくとも以下の要件を満たすことが必要です

- 画面サイズ：96×54 Pixel
- 画面深度：1bit
- アスペクト比：ほぼ 1 対 1

入力 以下に挙げたユーザ入力方式のいずれかひとつ，もしくは複数をサポートしていることが必要です

- 片手キーボード
- 両手キーボード
- タッチスクリーン

メモリ 少なくとも以下の要件を満たすことが必要です

- MIDP コンポーネント用に 128KByte の不揮発性メモリ
- アプリケーションが作成した固定データ用に 8KByte の不揮発性メモリ
- Java ランタイム用に 32KByte の不揮発性メモリ

ネットワーク 少なくとも以下の要件を満たすことが必要です

- 限定された帯域幅で，双方向の一時的なワイヤレス通信機能

MIDP 上のアプリケーションは「MIDlet[*2]」と呼ばれています。MIDlet は Java アプレットとよく似た構造をしています。MIDlet は CLDC/MIDP 端末であればどの端末でも同じように動作します。

しかし実は携帯端末についてはプロファイルレイヤの互換性がないのが実情です。図 2 に日本の大手ベンダで採用している MIDlet の構成を示します。J-PHONE および au では MIDP の他に機能を拡張するための拡張

[*1] KVM の「K」はキロを表しています。
[*2]「ミッドレット」と発音するそうです。

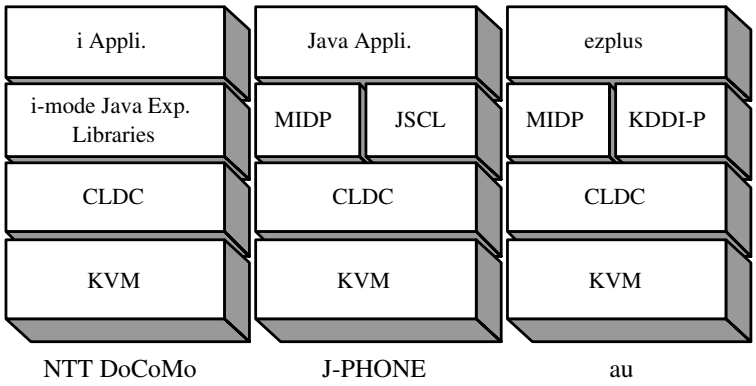

図 2　MIDlet 構成

プロファイルを用意しています。従って拡張機能を使わず CLDC/MIDP のみで実装されたアプリケーションであれば互換性を確保することができます[*3]。一方 NTT DoCoMo では，独自のプロファイルによる「i アプリ」のみ対応しています。したがって他社用の MIDP 端末との互換性はありません。

i アプリは KVM/CLDC 環境用のアプリケーションとしてはかなり早い（まだ MIDP が整備されていない）時期に登場しており，アプリケーション構成としてはかなり異質なものとなっています。今後様々な携帯端末が登場すると予想されますが，KVM ベースの Java アプリケーションは MIDP をプロファイルの中心に据えた（J-PHONE や au のような）構成になると思われます。

2.3　MIDlet の配布

携帯端末への MIDlet の配布方法としては今のところ以下の 2 通りが考えられます。

1. 専用同期ソフトなどを使ったシリアル回線によるインストール
2. WWW 等によるネットワーク経由でのインストール

特定業務向け用途であれば最初の方法が最も適しています。不特定のユーザへ配布する場合には 2 番目の方法がよいでしょう。ただし，2 番目の方法はベンダによって手順が異なります。J-PHONE の Java アプリの場合は「コンテンツアグリゲータ」と呼ばれる J-PHONE 認定企業のサイトにアップロードし，審査を受ける必要があります。またインターネット上の Java アプリの管理も「コンテンツアグリゲータ」が行っています。一方 au ではそのような審査・管理機構はなく自由にアプリケーションを公開できます。ただし配布パッケージは「KJX ファイル」と呼ばれる独自の形式を使う必要があります。またダウンロード用の専用 CGI が用意されており，その CGI 経由でなければ配布できないようになっています。（つまり CGI の使えないサイトでは ezplus アプリケーションの配布はできないことになります）

携帯端末では MIDlet の管理を行う AMS（または JAM）と呼ばれる管理ソフトウェアが，MIDlet の制御を行っています。（図 3 参照）

AMS の基本的な機能は以下のとおりです。

[*3] ただし au の ezplus アプリケーションは配布パッケージに独自の形式（KJX）を採用しているため，au 用のパッケージがそのまま J-PHONE 端末に使えるわけではありません。

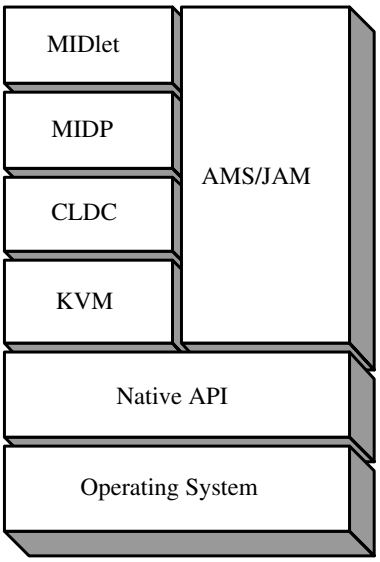

図 3　MIDlet と AMS

- ユーザ操作により，PC に接続したシリアルケーブルを介して，またはネットワーク経由で MIDlet をインストール・アンインストールする。
- MIDlet のランタイム環境を提供する。
- 上記の動作中にエラーによりシステムがダウンしないようエラー状態等を適切にハンドリングする

3　CLDC/MIDP ライブラリ

この節では CLDC/MIDP クラスライブラリの内容についてかんたんに紹介します。基本的には Java の標準ライブラリと同等のものがパッケージ化されていますが，ハードウェア等の制約によりサポートされていないものもあります。ここでは，それらの差異を中心に述べていくことにします。

3.1　KVM の制約事項

CLDC/MIDP ライブラリについて述べる前に，仮想マシンである KVM の制約事項について言及しておきます。

KVM は機能をなるべく簡略化するために CLDC でサポートされていない機能（3.2 節参照）は実装されていません。また J2ME は Java セキュリティモデルの限定されたバージョンを提供しているため，セキュリティホールを生み出すようなある特定の機能も KVM から省かれています。以降ではこのように省かれた機能について説明します。

3.1.1　JNI の非サポート

KVM では以下の理由により JNI をサポートしていません。

- CLDC のセキュリティモデルによると[*4]，アプリケーションプログラマはネイティブ機能を備えた新しいライブラリをダウンロードできないし Java ライブラリの一部でないネイティブ機能にはアクセスできないため。
- JNI は装置のリソースを大量に消費してしまうため。

3.1.2 ユーザ定義クラスローダの非サポート

ユーザは KVM に内蔵されたクラスローダを上書きしたり置き換えたり，あるいは設定し直したりもできません。

3.1.3 リフレクションの非サポート

リフレクションとはクラスやオブジェクトやメソッドなどの内容をチェックする機能です。KVM はリフレクションを実装していません。またリフレクションに依存した機能（RMI およびオブジェクトシリアライズ）もサポートされていません。

3.1.4 スレッドグループとデーモンスレッドの非サポート

KVM はマルチスレッド型のアプリケーションをフルサポートしていますが，スレッドグループとデーモンスレッドはサポートしていません。グループ操作を行うためには，明示的にコレクションオブジェクトを使う必要があります。

3.1.5 ウィークリファレンスの非サポート

ウィークリファレンスを利用すると，オブジェクトが再利用の対象になったとコレクションが判断した際に，プログラムは通知を受け取れるようになります。KVM はウィークリファレンスをサポートしていません。

3.2 CLDC ライブラリ

CLDC クラスライブラリは以下のとおりです。

- `java.lang`
- `java.util`
- `java.io`
- `javax.microedition.io`

このうち最初の 3 つ（`java.lang`，`java.util`，`java.io`）は J2SE のサブセットで，J2SE と上位互換性があります。これらのクラスについての説明は割愛します。

最後の `javax.microedition.io` は CLDC 固有のクラスで，J2SE との互換性はありません。このクラスは J2ME 端末用にネットワーク通信の抽象的なフレームワーク（Generic Connection framework）を提供しています。Generic Connection framework では，ソケット，データグラム，シリアル，HTTP などのプロトコルのインタフェースを用意していますが，実際にこれら全てが実装されるわけではなく通常はこれらプロトコル

[*4] J2ME のセキュリティモデルは J2SE のセキュリティモデル「sandbox」を応用したものです。J2ME の sandbox ではアプリケーションは閉じた環境でのみ動作しなければなりません。

の一部をサポートしたサブセットが使われます。ただし HttpConnection だけは必須となっているため，通信機能を実装する場合には HTTP を使用した方が汎用性が高いといえます。

3.2.1 データタイプ

表 1 に示すように，CLDC ではプリミティブなデータタイプだけをサポートしています。

表 1　CLDC でサポートされるデータタイプ

データタイプ	ラッパークラス
byte	java.lang.Byte
short	java.lang.Short
int	java.lang.Integer
long	java.lang.Long
char	java.lang.Character
boolean	java.lang.Boolean

float，double といった浮動小数点のデータタイプはコストが高くつきすぎるためサポートされていません[*5]。

3.2.2 ファイナライズ

ファイナライズ（finalize()）はガベージコレクションを行う前にオブジェクト自身で明示的にクリーンアップ処理を行うためのコールバックの一種です。CDLC では（KVM のガベージコレクション機能を簡略化するため）この機能はサポートされていません。

3.2.3 国際化

CLDC では java.io.InputStreamReader と java.io.OutputStreamWriter のみ国際化（l18n）機能をサポートしています。従ってアプリケーション内部でロケールを切り替えるのは非常に手間のかかる難しい処理であるといえます。むしろ各ロケールごとに専用の MIDlet を用意し，配布時にターゲット装置に合わせた MIDlet をインストールした方が低コストで安全と思われます。

システムのロケールとデフォルト文字エンコーディングは，CLDC にある microedition.locale および microedition.encoding プロパティで定義されています。なお，日本の大手ベンダの携帯端末のデフォルト文字エンコーディングはシフト JIS になっているそうです。

3.2.4 例外処理

CLDC では java.lang.VirtualMachineError と java.lang.OutOfMemoryError の（Exception クラスから派生した）2 つのエラークラスしかサポートしていません。これは以下の理由によります。

- ふつう組込型システムではエラー状況から復旧する処理はかなり装置固有であるため。アプリケーションプログラマは，これらの装置固有のエラーを操作できると考えるべきではない。
- （Exception ではなく）Error の例外処理は一般に復旧不可能であるため。

[*5] KVM を実装する装置は，一般にプロセッサパワーが貧弱で浮動小数点演算命令をサポートしていないことも多いため，浮動小数点演算による（主に時間）コストが非常に高くなります。

- フル機能のエラーハンドリングを実装することはリソースの制限された CLDC 装置には大きな負荷となるあため。

付録 A　用語・略語集

本テキストでは様々な用語・略語が出てきます。これらの略語について表 2 に一覧を挙げておきます。参考にしてください。

表 2　略語一覧

略語	正式名
J2SE	Java 2 Standard Edition
J2ME	Java 2 Micro Edition
JVM	Java Virtual Machine
KVM	K Virtual Machine
CVM	C Virtual Machine
CLDC	Connected Limited Device Configuration
CDC	Connected Device Configuration
MIDP	Mobile Information Device Profile
RMI	Remote Method Invocation
RMIP	RMI Profile
KDDI-P	KDDI Profile
JSCL	J-PHONE Specific Class Libraries
AMS	Application Managemanet Software
JAM	Java Application Manager
JNI	Java Native Interface

参考文献

[1] Yu Feng and Dr. Jun Zhu.　J2ME ワイヤレス Java プログラミング.　（株）アスキー, 2002. http://www.ascii.co.jp/pb/ant/j2me/.

[2] 市嶋洋平, 大森敏行, 堀内かほり. 携帯電話の未来. 日経バイト, Vol. 229, pp. 68–87, June 2002.

[3] 布留川英一.　MIDP Java ゲームプログラミング.　（株）毎日コミュニケーションズ, 2002. http://www.people.or.jp/~npaka/midpbook/.