iアプリ Java
ゲームプログラミング

○　第２回　プロジェクトの作成　○

Java

Javaとは、Sun Microsystems社が開発したオブジェクト指向プログラミング言語で、その実行環境も含めてJavaと呼びます。
Javaは、どのような環境でも実行できることを目標としており、仮想マシンが存在すれば、OSに依存することなくコードを実行できることが特徴です。OSに依存しないコードが書けることから、サーバー用のアプリケーションや、組み込みソフトの実行環境として携帯電話に普及しています。

Javaの特徴

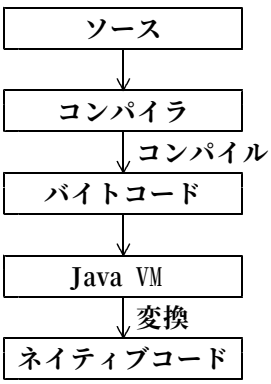
Javaは、C++の優れた点だけを引き継ぎ、オブジェクト指向に不要なものや危険性の高いものを削り、シンプルで安全性の高い言語を目指しています。たいていの言語が文字とファイルの入出力しか言語仕様に含んでいないのに対し、Javaでは画像、GUI、ネットワークなどの膨大なクラスライブラリを言語仕様として標準サポートしています。

▲C++で作成されたプログラム
(構造化＋オブジェクト指向プログラミング)

▲Javaで作成されたプログラム
(完全なオブジェクト指向プログラミング)

Javaの仕組み

Javaで記述されたプログラムは、コンパイラによりバイトコードにコンパイルされます。バイトコードは、JavaVM(Java Virtual Machine)という仮想マシン用のコードです。JavaVMは、バイトコードを１つ１つ環境に合うように解釈しながら実行します(バイトコードインタプリタ方式)。直接実行できるネイティブコードに比べ実行速度は劣りますが、Windows, Macintosh, Linuxなどさまざまな環境のものが提供されており、同じJavaプログラムをOSやCPU、表示機能、ネットワークなどの違いを超えて、異なる環境で動かすことができるようになっています。



iアプリ

ｉアプリは、NTTドコモのｉモード端末機でダウンロードして利用できるアプリケーションです。ゲームや地図情報、株価情報などのコンテンツが携帯電話で楽しめます。
携帯電話は、PCと比較してハードウェアの性能上、厳しい制約があり、通常のJavaより機能は削られています。

Eclipse

Eclipseは、統合ソフトウェア開発環境(IDE)の一つで、Java開発者を中心に急速に普及しています。ソフトウェア開発ツールの共通プラットフォームの標準になるといわれています。ソースコードが公開されており、無償で入手・改変・再配布できるようになっています。大手ソフトウェアベンダの中には、自社の開発ツール製品にEclipseを組み込み、Eclipseに追加する形で自社独自部分を提供するという形の製品を発売するところも現れています。

携帯電話の仕様

携帯電話の各機種の仕様は、以下のようになっています。

機種	ﾌﾟﾛﾌｧｲﾙ	JAR容量	ｽｸﾗｯﾁﾊﾟｯﾄﾞ	Canvasｸﾗｽ解像度	浮動小数点対応
505i	DoJa3.0	30KB	200KB	240×240	×
506i	DoJa3.0	30KB	200KB	240×240	×
700i	DoJa4.0LE	30KB	200KB	240×240	×
900i	DoJa3.5	100KB	400KB	240×240	×
901i	DoJa4.0	100KB	400KB	240×240	○

容量と解像度は、２０年前のファミコン程度です。実行速度に関しても、PCとは比べものにならないぐらい非力なCPUを搭載しています(Pentium4の数十分の一程度)。さらに、その上でJavaVMを稼働し、バイトコードを解釈しながら実行するので、とても処理が重くなります。PC上で動かす場合はPCのCPUが使用されるので快適に動きますが、実機では遅くて使い物にならない、ということはよくあります。

軽量化と高速化

サイズ縮小や高速化を行うには、無駄なコードを省き、Javaコンパイラがより高速なコードを出力するようにプログラムを作成していきます。しかし、突き詰め過ぎると、ソースコードが読みづらくなり保守管理が大変になってしまいます。

最適化を簡単に行いたい場合、オブファスケーターと呼ばれるツールを用います。このツールは、Javaのソースコードを解析し、約１５～２０%のサイズ縮小と高速化を行います。たとえば、ProGuard(http://proguard.sourceforge.net/)では、以下のような処理を行っています。

- フィールド変数、クラス、メソッド、クラス変数の名前を短いアルファベットに置き換える
- デバッグ情報の削除
- コンスタントプールの再評価
- 実行されないコードの削除
- 使われていないクラスの削除
- 使われていないフィールド変数の削除
- 呼ばれていないメソッドの削除
- 不必要な分岐の削除
- 不必要な比較の削除
- 書き込みのみのフィールドの削除
- PUSH/POPのような単純な命令の最適化
- 可能な場合、クラスの属性をfinalにする
- 可能な場合、単純なset～/get～メソッドのインライン化
- 単一のインプリメンテーションを行っているインタフェースの交換
- logging codeの削除

オブファスケーターを用いる以外にも、以下のような方法を用いることにより、サイズを縮小および高速化を行えます。

- 7-zipを使ってより圧縮率の高いJARファイルにする
  JARファイルは、ZIPで圧縮されています。そこで、ZIPと互換性があり、より高い圧縮をする7-Zip(http://www.7-zip.org/)というソフトを使います。１～２%ほど圧縮率が向上します。

- クラス数を少なくする
  プロジェクトで作成するクラス数を２つ程度に抑えます。ゲームでは、IApplicationを継承したクラスと、Cavasを継承したクラスのみが理想です。また、属性はfinalを付けると、５バイト程度小さくなります。１つクラス増やすと２キロバイトほど増えるといわれているので、なるべくクラスを作らないようにしましょう。

- メソッド数を少なくする
  メソッドを増やすごとに３０バイトほど増えるといわれているので、極力作らないようにします。しかし、同じような処理が沢山ある場合はメソッドを作った方が効率が良くなります。また、メソッドをpublicで宣言することで、５バイトほど削減できます。
- 変数を少なくする
  変数は可能な限り少なくします。クラス変数はなるべく定義をしないようにします。フィールド変

数の場合もpublic staticとし、なるべく定義しないようにします。
また、１つの変数の複数の目的に再利用をするとサイズ縮小になりますが、可読性が損なわれてバグの温床になるので注意しましょう。
ほかに、例として0～255までの整数値をとる４つの変数は、まとめて１つのint型変数でビットシフト演算子や論理演算子を用いて展開するといった方法があります。

・定数は直で打つ
static finalの定数を定義せず数字で直に打つことにより、サイズ縮小に貢献します。ただし、値に変更があった場合に書き換えるのが大変面倒になります。C／C++のようなプリプロセッサを実現するツールもあります(PPP(http://www.and.or.jp/~pamulow/))。

・１次元配列と外部ファイルを使う
変数を１次元配列にまとめることで、容量が少なくなります。多次元配列も１次元に展開します。
配列は１つずつ初期値を入れず、プログラム内で展開するようにします。
例えば、「int[] array = new int[5];」という配列がある場合、

```
array[0] = 0;
array[1] = 1;
array[2] = 2;
array[3] = 3;
array[4] = 4;
```

とするよりも

```
for (int i = array.length - 1; i >= 0; i--)
  array[i] = i;
```

とした方がサイズは少なくなります。非常に大きな配列やプログラム内で展開できないような不規則な値の配列の場合は、外部ファイルとして数値データを持って、読み込むプログラムを書いた方が容量が少なくなります。文字列についても同様に、外部ファイルにすると容量が少なくなります。

・ifのみ使う
if-else文をif-if文に置き換えた方が容量が少なくなります。また、switch文よりもifのほうが高速です。また、for文も大きな配列を使う場合などを除き、if文を使った方が高速です。ただし、可読性が大幅に損なわれてしまいます。

・try-catchを一つにまとめる
例外を受け取るとき、tryをネストせずに１つだけ定義し、その中に詰め込みます。エラーメッセージはtoStringメソッドを使って取り出します。なお、実機で例外が発生したときは、プログラムを停止しなければならない場合がほとんどなので、例外からの復帰や、高度な処理を記述する必要はありません。

・比較は０と行う
０との比較はバイトコードで専用の命令が用意されているために、速くなり、サイズも軽くなります。例えば、

```
for(int i = 0; i < 256; i++)
  array[i] = i;
```

という文は

```
for(int i = 255; i > 0; i--)
  array[i] = i;
```

と書きます。

・TimerおよびTimerTaskクラスは使わない
定期的にイベントをおこしたいのであれば、Threadクラス、System.currentTimeMillisメソッド、Thread.sleepメソッドを組み合わせた方が、よりサイズが小さく、速くなります。

・リソース読み込みや破棄をした後にガベージコレクタを使用する
メモリを大量に使った後などは、System.gcメソッドを利用することで、空きメモリを最大限活用することができます。例えば、画像を破棄した後や、大きな配列を使った後などに使用します。
これは軽量化、高速化するわけでなく、逆にサイズも大きくなり、処理も遅くなりますが、適度にガベージコレクトをすることは、例外MemoryOutofErrorが起こりにくくなります。

・ネットワークからリソースをダウンロードし、スクラッチパッドに保存する
容量を最大限に活用するために、JARにおさまりきらない部分は、ネットワーク(HTTP)からデータをダウンロードし、スクラッチパッドに保存する方法を取ります。
ダウンロードと保存用のコードを書く手間が増えますが、この処理は初回起動時だけでいいので、有効な手段だといえます。

プロジェクトの作成

Eclipseでは、プロジェクトという単位でプログラムを管理します。iアプリでは、ひとつのiアプリに対してひとつのプロジェクトを作成します。

プロジェクトは、Javaソースファイルやリソースと呼ばれる各種画像や音楽ファイルなどをまとめて管理するものです。プロジェクトの実体は、プロジェクトフォルダ以下に作成される各種フォルダです。フォルダの構成は、以下のようになっています。

フォルダ	用途
bin	実行ファイル(.jar)が生成されるフォルダ
classes	コンパイルしたファイルが保存されるフォルダ
res	画像や音楽などのリソースを置くフォルダ
sp	iアプリでデータを保存するときに用いるスクラッチパッドの内容が保存されるフォルダ
src	ソースファイルを保存するフォルダ

課　題

Eclipseを起動し、iアプリ用のプロジェクトを作成しましょう。

①デスクトップにあるEclipseのアイコンをダブルクリックして起動し、メニューから「ファイル」→「新規」→「プロジェクト」を選択します。

②「新規プロジェクト」の画面が表示されます。iアプリを作成する場合は、左側のツリーで「Java」を選んで、右側のプロジェクト一覧からは「DoJa-4.0プロジェクト」を選びます。そして「次へ」をクリックします。

③プロジェクト名を入力する画面が表示されるので、プロジェクトの名前「JavaGame」を入力して「終了」をクリックします。ここで入力した名前のフォルダが作成されます。