

実行形式の作成と構成管理

2012年7月8日

海谷 治彦

動機付け

- 大多数の人はプログラムは実行するだけ。
 - プログラム単体で実行できないと困る。
 - 毎回, Eclipseを立ち上げていては困る.
- 実行可能ファイルの作成
 - Javaに限らず通常, 多数のソースファイル, 外部ライブラリから実行ファイルは構成される。
 - 修正毎に手作業でコンパイルするのはつらい.
- 開発時と実行時の環境の違い
 - 特に外部ライブラリの配置は開発者マシンと利用者のマシンとでは異なる場合が多い.

Javaでの解法

- Jarファイル
 - 単体ファイルで実行可能なJavaプログラム
 - 気分的にはWindowsの exe ファイルと同じ.
- antプログラム
 - 実行ファイルの生成, 文書の生成等を自動化するための「プログラム開発手順」のプログラム
- jarのManifestファイル
 - 利用者の実行時のライブラリ配置等を指定できる.
 - 開発時点での配置と異なっても良い, 同じでもよいけど.

Jarファイルの作り方

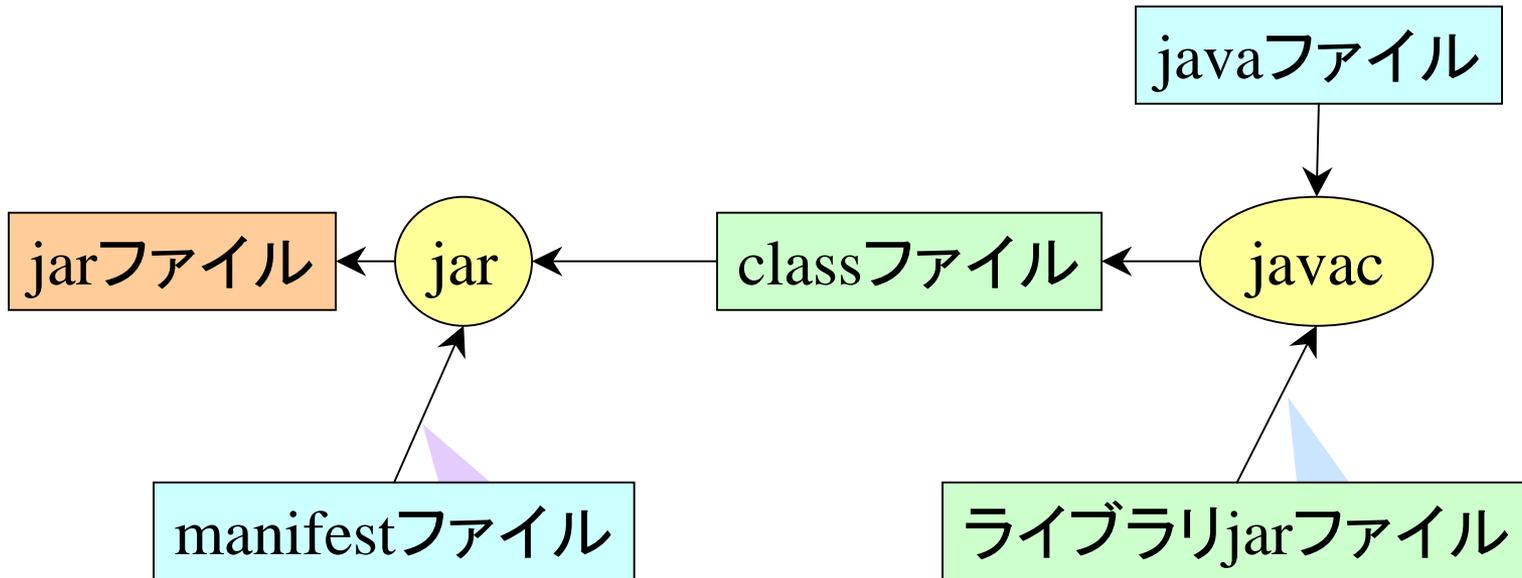
詳細はHP参照

- Eclipseから, File → Export で作る.
- jarコマンドで作る.
- antで作る.

- 多分, zip とかでも作れると思うけど, 未確認. (やめたほうがよい)

開発時(jar作成時)の流れ

開発者のマシンの中(Eclipseの中といっても良い)



- どこにmainメソッドがあるか
- どこにライブラリがあるか
- 開発時の場所と違ってよい.

- javaファイル内で使われる型や引数が出てくるかのチェック.

実行時(jar利用時)の流れ

ユーザーのマシンの中

ライブラリjarファイル

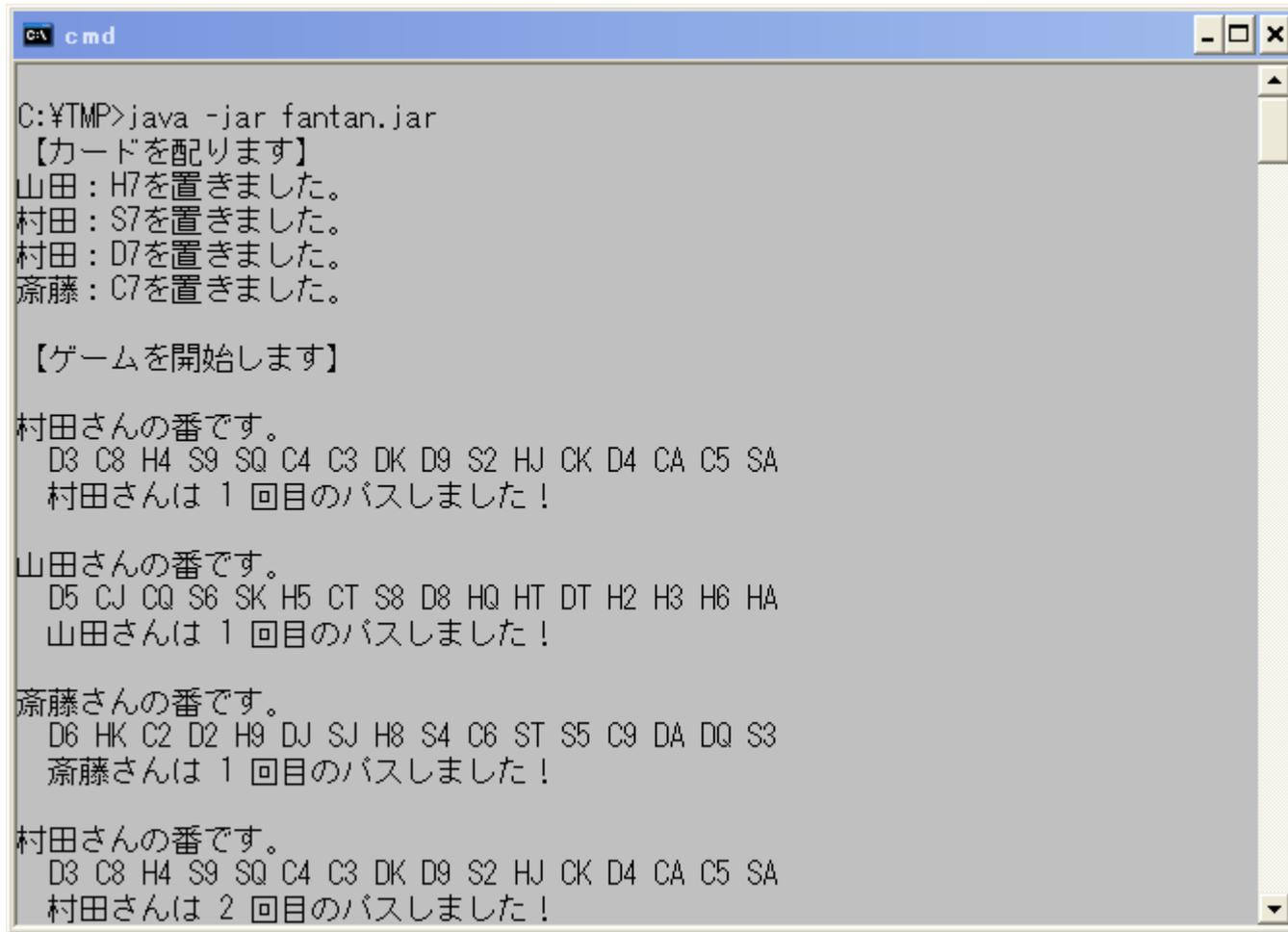
java

実行jarファイル

•ライブラリjarファイルの場所はココに埋め込まれている.

- ライブラリjarファイルは, 実行jarファイルの指定に従い配置されてないとマズい.
- 実はユーザーマシン内に無くても良い, http等でアクセスできる外部位置でもOK.

jarの実行例



```
C:\>cmd
C:¥TMP>java -jar fantan.jar
【カードを配ります】
山田：H7を置きました。
村田：S7を置きました。
村田：D7を置きました。
斎藤：C7を置きました。

【ゲームを開始します】

村田さんの番です。
D3 C8 H4 S9 SQ C4 C3 DK D9 S2 HJ CK D4 CA C5 SA
村田さんは 1 回目のパスしました！

山田さんの番です。
D5 CJ CQ S6 SK H5 CT S8 D8 HQ HT DT H2 H3 H6 HA
山田さんは 1 回目のパスしました！

斎藤さんの番です。
D6 HK C2 D2 H9 DJ SJ H8 S4 C6 ST S5 C9 DA DQ S3
斎藤さんは 1 回目のパスしました！

村田さんの番です。
D3 C8 H4 S9 SQ C4 C3 DK D9 S2 HJ CK D4 CA C5 SA
村田さんは 2 回目のパスしました！
```

jarの内部構造

Manifest-Version: 1.0

Ant-Version: Apache Ant 1.8.2

Created-By: 1.7.0_05-b05 (Oracle Corporation)

Main-Class: fantan.Fantan

Class-Path: /Program% 20Files/Java/extJar/trumpFramework.jar

```
C:¥TMP>jar tvf fantan.jar
```

```
 0 Sun Jul 08 16:22:10 JST 2012 META-INF/  
191 Sun Jul 08 16:22:08 JST 2012 META-INF/MANIFEST.MF  
 0 Wed Jul 04 10:42:48 JST 2012 fantan/  
1429 Wed Jul 04 10:42:48 JST 2012 fantan/Fantan.class  
1160 Wed Jul 04 10:42:48 JST 2012 fantan/FantanMaster.class  
2282 Wed Jul 04 10:42:48 JST 2012 fantan/FantanPlayer.class  
1386 Wed Jul 04 10:42:48 JST 2012 fantan/FantanRule.class  
1416 Wed Jul 04 10:42:48 JST 2012 fantan/FantanTable.class
```

開発時の作業自動化 ant

- antツールを用いると開発時の種々の作業(タスクと呼ばれる)の実行手順を予め記述して, 自動, 繰り返し実行できます.
- タスクの例
 - jarを作る
 - コンパイルする
 - javadocでマニュアルを生成
 - junitのテスト
- Eclipseとは独立したコマンドですが, Eclipseからも使えます.
- Eclipseのメニューから呼び出すよりもきめ細かい設定ができます.

antへの入力ファイルの例

```
<?xml version="1.0" ?>

<project name="bulid" default="jar" basedir="." >
<property name="jar_name" value="fantan.jar"/>
<property name="usrclasspath"
           value="/Program%20Files/Java/extJar/trumpFramework.jar"/>
<property name="devclasspath" value="trumpFramework.jar"/>

<target name="jar" depends="compile" >
<jar jarfile="${jar_name}" basedir="bin" includes="**/*.class" >
<manifest>
<attribute name="Main-Class" value="fantan.Fantan"/>
<attribute name="Class-Path" value="${usrclasspath}"/>
</manifest>
</jar>
</target>

<target name="compile">
<javac srcdir="src" destdir="bin" classpath="${devclasspath}"/>
</target>

</project>
```

記述の特徴

- 見ての通り XML
- `<project> </project>`で全体を被う.
- `<property>`は記述内の変数と考え良い. 複数回利用する値や変更をよくする値には便利.
- `<target>`がタスクに相当する.

manifestを 別ファイルに書いても良い

```
<target name="jar" depends="compile">  
  <jar jarfile="${jar_name}" basedir="bin" includes="*.class"  
    manifest="manifest.txt"/>  
</target>
```

```
Main-Class: MyExtractText  
Class-Path: . /Program%20Files/Java/extJar/commons-logging-1.1.1.jar Program%20
```

※ クラスパスが長い...

外部ファイルが複数ある例

```
<?xml version="1.0" ?>

<project name="bulid" default="jar" basedir=".">
  <!-- name of jar file -->
  <property name="jar_name" value="pdf2txt.jar"/>
  <property name="common_logging" value="commons-logging-1.1.1.jar"/>
  <property name="fontbox" value="fontbox-1.1.0.jar"/>
  <property name="pdfbox" value="pdfbox-1.1.0.jar"/>
  <property name="devpath" value="C:¥¥Program Files¥¥Java¥¥extJAR¥¥"/>
  <property name="usrpath" value="/Program%20Files/Java/extJar/" />
  <property name="netpath" value="http://cai.cs.shinshu-u.ac.jp/~kaiya/" />

  <target name="jar" depends="compile">
    <jar jarfile="${jar_name}" basedir="bin" includes="**/*.class">
      <manifest>
        <attribute name="Main-Class" value="MyExtractText"/>
        <attribute name="Class-Path" value=". ${usrpath}${common_logging} ${usrpath}${fontbox} ${usrpath}${pdfbox}"/>
      </manifest>
    </jar>
  </target>

  <target name="compile">
    <javac srcdir="src" destdir="bin">
      <classpath>
        <pathelement location="${devpath}${common_logging}"/>
        <pathelement location="${devpath}${fontbox}"/>
        <pathelement location="${devpath}${pdfbox}"/>
      </classpath>
    </javac>
  </target>

</project>
```

前頁のTIPS

- 開発マシンはWin, 利用は汎用を想定しているなので, パスの書き方が微妙に面倒.
- この例では使っていないが, 利用の際にはネット上に配置されたjarをライブラリとして利用可能.