# クラスファイルの構造解析(2)

2003年6月23日

海谷 治彦

# 目次

- コンスタントプールの解決とダイナミックリンク
- バイトコードの直接解析の手順
  - フィールド，メソッド，クラス属性等について.

# コンスタントプール リゾリューション

- Constant Pool Resolution
- CP中の，クラス，メソッド，フィールドの参照を，JVM中の実際のメモリ空間のアドレスに置きかえること.
- JVMでのダイナミックリンクの実現手段
  - リンクについては講義第2回資料参照.
- 教科書 p.115~

先週の資料の再掲載

# 簡単な例題 ./cpr/ の下

```
class Go{
static ClassA a;

    public static void main(String[] args){
        a=new ClassA();
    }
}
```

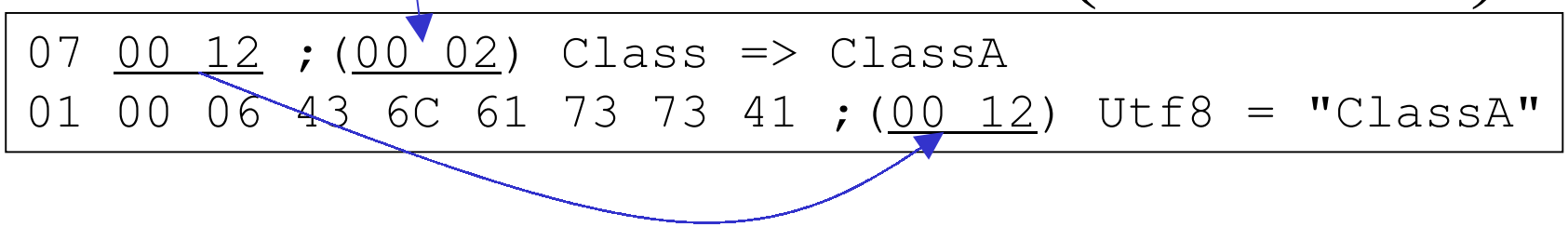
```
class ClassA{
private int a;
}
```

# ロードとリンクの実際 1/3

```
BB 00 02 ; new ClassA
59       ; dup
B7 00 03 ; invokespecial ClassA/<init>()V
B3 00 04 ; putstatic Go/a LClassA;
B1       ; return
```

## CPの抜粋 (Go.class)

```
07 00 12 ;(00 02) Class => ClassA
01 00 06 43 6C 61 73 73 41 ;(00 12) Utf8 = "ClassA"
```



ロード時には，CP#0002は，CP#0012の文字列を指してる.

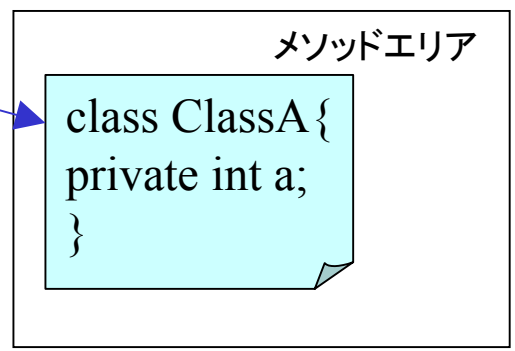
# ロードとリンクの実際 2/3

```
BB 00 02 ; new ClassA
59       ; dup
B7 00 03 ; invokespecial ClassA/<init>()V
B3 00 04 ; putstatic Go/a LClassA;
B1       ; return
```

## CPの抜粋 (Go.class)

```
07 00 12 ;(00 02) Class => ClassA
01 00 06 43 6C 61 73 73 41 ;(00 12) Utf8 = "ClassA"
```

しかし，new ClassA が実行されると，ClassAがロードされ，#0012は，実際にClassAが配置されたメモリにおきかわる



# ロードとリンクの実際 3/3

- 教科書p.116に書いてある例を，より簡単な形で図示した.
- jdbでの観察例も, ./cpr/ 下においた.
- これにより，CPエントリ#0002番，ClassAをさすCPは，以降，単なる文字列でなく，実際のClassAを指すようになる.
- CPリゾリューションが起こらない限り，クラスはロードされない.

# CPリゾリューションのまとめ

- CPエントリ内の，Fieldref, Methodref, InterfaceMethodref, Class の4つは，必要に応じて，メソッドエリア内のアドレスを指すようになる.
- 上記のようなことをCPリゾリューション(解決)と呼ぶ.
- ただし，その参照が利用されない限り，解決は行われない.

# Javaが遅いと感じるのは

- このCPリゾリューションを一発目にやってるので，走り出しが遅く感じる.
- 加えて，後述のバイトコード検証(p.117〜)もあるので，なおさら走り出しが遅い.

# 用語の確認

- ダイナミックリンク: もともとはばらばらだったクラスをCPを通して結び付け，相互に利用できるようにすること.
- オンデマンドなロード: CP解決される際，必要なクラスをロードする．例えプログラム中に宣言されていても，利用されない限り，そのクラスはロードしない.

# バイトコード解析の手順

- javacでクラスファイルを得る.
- bin2hex でダンプをとる.
- クラスファイルの構造を参考に区切りをつける.
  - CPのエントリを分割.
  - インストラクションの認識.
- かなり非人道的な作業です(涙)

# クラスファイルの構造概要

- 教科書 p.36 図1-13 である.
- 大雑把にいって,
  1. クラスファイル自体の情報
  2. コンスタントプールのリスト
  3. クラス自体の情報
  4. 実装しているインタフェースのリスト
  5. フィールドのリスト
  6. メソッドのリスト
  7. クラスの付加的情報のリスト(Innerクラスもこれに含まれる)

の7つのパートからなる.

再録

# 例題

```
public class Byte implements Runnable{
private int s;
    public void run(){
        s++;
    }
}
```

- インタフェース 1つ
- フィールド 1つ
- メソッド 2つ (暗黙のコンストラクタを含める)

# とりあえずバイト列を見る

```
CA FE BA BE 00 03 00 2D 00 14 0A 00 04 00 0F 09 ; .......-........
00 03 00 10 07 00 11 07 00 12 07 00 13 01 00 01 ; ................
73 01 00 01 49 01 00 06 3C 69 6E 69 74 3E 01 00 ; s...I...<init>..
03 28 29 56 01 00 04 43 6F 64 65 01 00 0F 4C 69 ; .()V...Code...Li
6E 65 4E 75 6D 62 65 72 54 61 62 6C 65 01 00 03 ; neNumberTable...
72 75 6E 01 00 0A 53 6F 75 72 63 65 46 69 6C 65 ; run...SourceFile
01 00 09 42 79 74 65 2E 6A 61 76 61 0C 00 08 00 ; ...Byte.java....
09 0C 00 06 00 07 01 00 04 42 79 74 65 01 00 10 ; .........Byte...
6A 61 76 61 2F 6C 61 6E 67 2F 4F 62 6A 65 63 74 ; java/lang/Object
01 00 12 6A 61 76 61 2F 6C 61 6E 67 2F 52 75 6E ; ...java/lang/Run
6E 61 62 6C 65 00 21 00 03 00 04 00 01 00 05 00 ; nable.!.........
01 00 02 00 06 00 07 00 00 00 02 00 01 00 08 00 ; ................
09 00 01 00 0A 00 00 00 1D 00 01 00 01 00 00 00 ; ................
05 2A B7 00 01 B1 00 00 00 01 00 0B 00 00 00 06 ; .*..............
00 01 00 00 00 01 00 01 00 0C 00 09 00 01 00 0A ; ................
00 00 00 27 00 03 00 01 00 00 00 0B 2A 59 B4 00 ; ...'........*Y..
02 04 60 B5 00 02 B1 00 00 00 01 00 0B 00 00 00 ; ..`.............
0A 00 02 00 00 00 04 00 0A 00 05 00 01 00 0D 00 ; ................
00 00 02 00 0E                                  ; .....
```

# bin2hex –q Byte.class の結果

```
CA FE BA BE 00 03 00 2D 00 14 0A 00 04 00 0F 09
00 03 00 10 07 00 11 07 00 12 07 00 13 01 00 01
73 01 00 01 49 01 00 06 3C 69 6E 69 74 3E 01 00
03 28 29 56 01 00 04 43 6F 64 65 01 00 0F 4C 69
6E 65 4E 75 6D 62 65 72 54 61 62 6C 65 01 00 03
72 75 6E 01 00 0A 53 6F 75 72 63 65 46 69 6C 65
01 00 09 42 79 74 65 2E 6A 61 76 61 0C 00 08 00
09 0C 00 06 00 07 01 00 04 42 79 74 65 01 00 10
6A 61 76 61 2F 6C 61 6E 67 2F 4F 62 6A 65 63 74
01 00 12 6A 61 76 61 2F 6C 61 6E 67 2F 52 75 6E
6E 61 62 6C 65 00 21 00 03 00 04 00 01 00 05 00
01 00 02 00 06 00 07 00 00 00 02 00 01 00 08 00
09 00 01 00 0A 00 00 00 1D 00 01 00 01 00 00 00
05 2A B7 00 01 B1 00 00 00 01 00 0B 00 00 00 06
00 01 00 00 00 01 00 01 00 0C 00 09 00 01 00 0A
00 00 00 27 00 03 00 01 00 00 00 0B 2A 59 B4 00
02 04 60 B5 00 02 B1 00 00 00 01 00 0B 00 00 00
0A 00 02 00 00 00 04 00 0A 00 05 00 01 00 0D 00
00 00 02 00 0E
```

# 最初の8バイトはプリアンブルとバージョン番号

```
CA FE BA BE 00 03 00 2D 00 14 0A 00 04 00 0F 09
00 03 00 10 07 00 11 07 00 12 07 00 13 01 00 01
73 01 00 01 49 01 00 06 3C 69 6E 69 74 3E 01 00
03 28 29 56 01 00 04 43 6F 64 65 01 00 0F 4C 69
6E 65 4E 75 6D 62 65 72 54 61 62 6C 65 01 00 03
72 75 6E 01 00 0A 53 6F 75 72 63 65 46 69 6C 65
01 00 09 42 79 74 65 2E 6A 61 76 61 0C 00 08 00
09 0C 00 06 00 07 01 00 04 42 79 74 65 01 00 10
6A 61 76 61 2F 6C 61 6E 67 2F 4F 62 6A 65 63 74
01 00 12 6A 61 76 61 2F 6C 61 6E 67 2F 52 75 6E
6E 61 62 6C 65
```

以下，省略.

# CPの数は0x0014個-1

```
CA FE BA BE ; プリアンブル
00 03 00 2D ; マイナーバージョン, メジャーバージョン
00 14 ; 0x0014-1 = 16+4-1 = 19個 (10進法)
0A 00 04 00 0F 09
00 03 00 10 07 00 11 07 00 12 07 00 13 01 00 01
73 01 00 01 49 01 00 06 3C 69 6E 69 74 3E 01 00
03 28 29 56 01 00 04 43 6F 64 65 01 00 0F 4C 69
6E 65 4E 75 6D 62 65 72 54 61 62 6C 65 01 00 03
72 75 6E 01 00 0A 53 6F 75 72 63 65 46 69 6C 65
01 00 09 42 79 74 65 2E 6A 61 76 61 0C 00 08 00
09 0C 00 06 00 07 01 00 04 42 79 74 65 01 00 10
6A 61 76 61 2F 6C 61 6E 67 2F 4F 62 6A 65 63 74
01 00 12 6A 61 76 61 2F 6C 61 6E 67 2F 52 75 6E
6E 61 62 6C 65
```

以下, 省略.

# 19個のCPを分割 p.51, 189~ 等参照

```
CA FE BA BE ; プリアンブル
00 03 00 2D ; マイナーバージョン, メジャーバージョン
00 14 ; 0x0014-1 = 16+4-1 = 19個 (10進法)
0A 00 04 00 0F 09
00 03 00 10 07 00 11 07 00 12 07 00 13 01 00 01
73 01 00 01 49 01 00 06 3C 69 6E 69 74 3E 01 00
03 28 29 56 01 00 04 43 6F 64 65 01 00 0F 4C 69
6E 65 4E 75 6D 62 65 72 54 61 62 6C 65 01 00 03
72 75 6E 01 00 0A 53 6F 75 72 63 65 46 69 6C 65
01 00 09 42 79 74 65 2E 6A 61 76 61 0C 00 08 00
09 0C 00 06 00 07 01 00 04 42 79 74 65 01 00 10
6A 61 76 61 2F 6C 61 6E 67 2F 4F 62 6A 65 63 74
01 00 12 6A 61 76 61 2F 6C 61 6E 67 2F 52 75 6E
6E 61 62 6C 65
```

以下, 省略.

# CPエントリ毎のサイズ

- 1 Utf8 = 1B + 2B + 上記2Bで示されたサイズ
- 2 Integer = 1B + 4B
- 4 Float = 1B + 4B
- 5 long = 1B + 8B
- 6 double = 1B + 8B
- 7 class = 1B + 2B
- 8 string = 1B + 2B
- 9 fieldref = 1B + 2B + 2B
- A methodref = 1B + 2B + 2B
- B interfaceMethodref = 1B + 2B + 2B
- C NameAndType = 1B + 2B + 2B

```
CA FE BA BE ; プリアンブル
00 03 00 2D ; マイナーバージョン, メジャーバージョン
00 14 ; 0x0014-1 = 16+4-1 = 19個 (10進法)
0A 00 04 00 0F ; 0001
09 00 03 00 10 ; 0002
07 00 11 ; 0003
07 00 12 ; 0004
07 00 13 ; 0005
01 00 01 73 ; 0006
01 00 01 49 ; 0007
01 00 06 3C 69 6E 69 74 3E ; 0008
01 00 03 28 29 56 ; 0009
01 00 04 43 6F 64 65 ; 000a
01 00 0F 4C 69 6E 65 4E 75 6D 62 65 72 54 61 62 6C 65 ; 000b
01 00 03 72 75 6E ; 000c
01 00 0A 53 6F 75 72 63 65 46 69 6C 65 ; 000d
01 00 09 42 79 74 65 2E 6A 61 76 61 ; 000e
0C 00 08 00 09 ; 000f
0C 00 06 00 07 ; 0010
01 00 04 42 79 74 65 ; 0011
01 00 10 6A 61 76 61 2F 6C 61 6E 67 2F 4F 62 6A 65 63 74 ; 0012
01 00 12 6A 61 76 61 2F 6C 61 6E 67 2F 52 75 6E 6E 61 62 6C 65 ; 0013
```

以下, 省略.

# 19個のCPを分割

# 残りのバイト 6Bはクラス情報

```
00 21 ; アクセスフラフ PUBLIC + SUPER
00 03 ; このクラスのCP “Byte”
00 04 ; スーパークラスのCP “java/lang/Object”
00 01 00 05 00
01 00 02 00 06 00 07 00 00 00 02 00 01 00 08 00
09 00 01 00 0A 00 00 00 1D 00 01 00 01 00 00 00
05 2A B7 00 01 B1 00 00 00 01 00 0B 00 00 00 06
00 01 00 00 00 01 00 01 00 0C 00 09 00 01 00 0A
00 00 00 27 00 03 00 01 00 00 00 0B 2A 59 B4 00
02 04 60 B5 00 02 B1 00 00 00 01 00 0B 00 00 00
0A 00 02 00 00 00 04 00 0A 00 05 00 01 00 0D 00
00 00 02 00 0E
```

# 実装インタフェースの数

```
00 21 ; アクセスフラフ PUBLIC + SUPER
00 03 ; このクラスのCP “Byte”
00 04 ; スーパークラスのCP “java/lang/Object”
00 01 ; インタフェース 0x0001個 要は一個
00 05 ; そのCPのリスト 一個だけど. CP05=“java/lang/Runnable”
00
01 00 02 00 06 00 07 00 00 00 02 00 01 00 08 00
09 00 01 00 0A 00 00 00 1D 00 01 00 01 00 00 00
05 2A B7 00 01 B1 00 00 00 01 00 0B 00 00 00 06
00 01 00 00 00 01 00 01 00 0C 00 09 00 01 00 0A
00 00 00 27 00 03 00 01 00 00 00 0B 2A 59 B4 00
02 04 60 B5 00 02 B1 00 00 00 01 00 0B 00 00 00
0A 00 02 00 00 00 04 00 0A 00 05 00 01 00 0D 00
00 00 02 00 0E
```

# フィールドの数

```
00 01 ; インタフェース 0x0001個 要は一個
00 05 ; CP05="java/lang/Runnable"
00 01 ; フィールドの数 0x1個
00 02 ; 一個目のフィールドのフラグ
00 06 ; フィールドの名前のCP "s"
00 07 ; フィールドの型へのCP "I"
00 00 ; このフィールドの属性は1つもない
00 02 00 01 00 08 00
09 00 01 00 0A 00 00 00 1D 00 01 00 01 00 00 00
05 2A B7 00 01 B1 00 00 00 01 00 0B 00 00 00 06
00 01 00 00 00 01 00 01 00 0C 00 09 00 01 00 0A
00 00 00 27 00 03 00 01 00 00 00 0B 2A 59 B4 00
02 04 60 B5 00 02 B1 00 00 00 01 00 0B 00 00 00
0A 00 02 00 00 00 04 00 0A 00 05 00 01 00 0D 00
00 00 02 00 0E
```

# メソッド数

```
00 01 ; インタフェース 0x0001個 要は一個
00 05 ; CP05="java/lang/Runnable"
00 01 ; フィールドの数 0x1個
00 02 ; 一個目のフィールドのフラグ
00 06 ; フィールドの名前のCP "s"
00 07 ; フィールドの型へのCP "I"
00 00 ; このフィールドの属性は1つもない
00 02 ; メソッドは二つ
00 01 00 08 00
09 00 01 00 0A 00 00 00 1D 00 01 00 01 00 00 00
05 2A B7 00 01 B1 00 00 00 01 00 0B 00 00 00 06
00 01 00 00 00 01
以下省略
```

# メソッドその1

```
00 02 ; メソッドは二つ
00 01 ; アクセスフラグ PUBLIC
00 08 ; メソッド名へのCP “<init>”
00 09 ; メソッド型へのCP “()V”
00 01 ; メソッド属性の数 1個
00 0A ; この属性の種類のCP “Code”
00 00 00 1D ; 属性の長さ 0x0000001Dバイト=29B 残り29Bがメソッド
00 01 ; stack limit
00 01 ; locals limit
00 00 00 05 ; コードの長さ 5B
2A          ; aload_0
B7 00 01 ; invokespecial java/lang/Object/<init>()V
B1          ; return
00 00 ; 例外ハンドラの数 = 0
00 01 ; このコードのコード属性の数
00 0B ; 属性の種類のCP “LineNumberTable”
00 00 00 06 ; 属性の長さ 6B
00 01 ; 行番号は1つ
00 00 00 01 ; コード先頭から0B目が, .line 1
以下省略
```

```
00 01 ; アクセスフラグ PUBLIC
00 0C ; メソッド名のCP “run”
00 09 ; 型のCP “()V”
00 01 ; 属性数 1
00 0A ; 属性の種類のCP “Code”
00 00 00 27 ; 属性の長さ 0x27= 39B, 残り39Bがこのメソッド
00 03 ; stack limit=3
00 01 ; locals limit=1
00 00 00 0B ; コードの長さ 0xB=11バイト
2A        ; aload_0
59        ; dup
B4 00 02 ; getfield Byte/s I
04        ; iconst_1
60        ; iadd
B5 00 02 ; putfield Byte/s I
B1        ; return
00 00 ; 例外ハンドラ数=0
00 01 ; コード属性の属性数=1
00 0B ; 属性の種類のCP “LineNumberTable”
00 00 00 0A ; 属性の長さAバイト=10バイト
00 02 ; 2行分
00 00 00 04 ; .line 4
00 0A 00 05 ; .line 5
```

# メソッド2

# クラス属性

```
00 01 ; num. of class attribute(s)= 1(d)
00 0D ; class's attr.1(d) name="SourceFile"
00 00 00 02 ; length of attr.=2(d)
00 0E ; filenameのCP "Byte.java"
```

# 感想

- 疲れた.

- あまり人のやることじゃないな.
- でも，マイコンの授業だし，一生一度の記念にやってみてください，来週の演習にて.

今日はおしまい