WALA チュートリアル 2
–バイトコード解析–

愛知県立大学 山本研究室
藤浦 祥雅*

はじめに

このチュートリアルでは，WALA[1] を用いた Java バイトコード解析について説明します.

1 前準備

WALA について説明する前に，プロジェクトの準備とテストデータの準備について説明します.

1.1 プロジェクトの準備

サンプルコードを記述するためのプロジェクトを作成します．WALA は Eclipse プラグインに依存しているため，プロジェクトは Java プロジェクトではなく，Plugin プロジェクトを作成します.

1. メニューの [File] → [New] → [Other] を選択
2. (Plug-in 開発内の) Plug-in プロジェクトを選択し，[Next] を選択
3. Project name を入力し，[Next] を選択
4. Option のチェックをすべてはずし，[Finish] を選択

手順 4 で，Activity を作成しないよう設定し，テンプレートを利用せずにプロジェクトを作成します．このチュートリアルでは Project name を `jp.yamamotolab.tutrial.wala` とします.
次に,「Plug-in Dependencies」(プラグイン依存関係) に先ほどインポートした三つのプロジェクト (wala.core, wala.ide, wala.shrike) と，wala.core が依存している org.eclipse.core.runtime プラグインを追加します.

1. <PROJECT_TOP>/META-INF/MANIFEST.MF を開く
2. 画面下にある，[Dependencies] タブを選択
3. 「Required Plug-ins」に，com.ibm.wala.core，com.ibm.wala.shrike，com.ibm.wala.ide と org.eclipse.core.runtime の四つを加える

*mail:[qtutorial@gmail.com]

最後に解析除外対象を指定するファイル (Exclusions.txt) を作成します．場所は作成したプロジェクトの直下に配置します．(jp.yamamotolab.tutrial.wala/Exclusions.txt のように)

リスト 1: Exclusions.txt の例

```
java\/awt\/.*
java\/applet\/.*
java\/beans\/.*
java\/nio\/.*
java\/rmi\/.*
java\/security\/.*
java\/sql\/.*
java\/util\/.*
java\/text\/.*
java\/math\/.*
java\/net\/.*
javax\/.*
com\/sun\/.*
sun\/.*
org\/.*
sunw\/.*
java\/io\/.*
java\/lang\/annotation\/.*
java\/lang\/reflect\/.*
java\/lang\/management\/.*
java\/lang\/instrument\/.*
java\/lang\/ref\/.*
netscape\/.*
```

この Exclusions.txt については，後ほどに説明します．

以上で，プロジェクトの準備は完了です．

1.2 テストデータの準備

このチュートリアルでは，以下の Java コードを解析することとします．

リスト 2: Hello.java

```
public class Hello {
        public void run() {
                String s = "Hello, WALA!!";
                System.out.println(s);
        }

        public static void main(String[] args) {
                Hello h = new Hello();
                h.run();
        }
}
```

今回はバイトコード解析なので，コンパイルを行い，また jar で固めます．Hello.java のあるディレクトリまで移動して

1. javac Hello.java
2. jar cvf Hello.jar *.class

とします．できた Hello.jar は先ほど作成したプロジェクト直下に配置してください．(jp.yamamotolab.tutrial.wala/Hello.ja
のように)

以上で，テストデータの準備は完了です．

2　バイトコード解析をする

準備が整ったところで，本題のバイトコード解析について説明したいと思います.

WALA ではどのファイルを解析するのかなど解析範囲を `AnalysisScope` クラスをとして管理しています．そのため，AnalysisScope オブジェクトを生成することから始めます.

リスト 3: AnalysisScope の生成

```
1  AnalysisScope scope =
2      AnalysisScopeReader.makeJavaBinaryAnalysisScope(
3              "Hello.jar",
4              FileProvider.getFile("Exclusions.txt"));
```

AnalysisScope オブジェクトは `new` でも生成できますが，こまごました設定をしなくていいように，便利なユーティリティが用意されています.

makeJavaBinaryAnalysisScope()

public static AnalysisScope makeJavaBinaryAnalysisScope(java.lang.String classPath, java.io.File exclusionsFile) throws java.io.IOException

classPath File.pathSeparator で分けられた，解析対象のパス

exclusionsFile exclusions が記述されたファイル

これで Hello.jar を解析対象とした AnalysisScope が作成されました．このとき解析対象となるのは Hello クラスだけでなく，Java 標準ライブラリ全体も含まれます．しかし，Java 標準ライブラリすべてを含めると処理も多くなり，無駄に処理時間とメモリ空間が必要となるため，必要の無いパッケージ (やクラス) を除くことができます．その解析対象としないパッケージを Exclusions.txt で指定しています．Exclusions.txt を変更することで無駄な処理を省くことができます.

scope の利用例として，以下にクラス階層の取得方法について説明します.

リスト 4: クラス階層の取得

```
1  IClassHierarchy cha = ClassHierarchy.make(scope);
```

クラス階層は IClassHierarchy クラスとして扱われます．クラス階層を取得するには `ClassHierarchy.make()` メソッドを用います.

クラス階層については，別のチュートリアルにて説明します.

3　おわりに

今回は以下のことを行いました.

- プロジェクトの作成
- テストデータの作成
- AnalysisScope について
- Exclusions.txt について

次のチュートリアルでは，ソースコード解析における AnalysisScope の扱い方について説明します.

参考文献

[1] WALA wiki：*http://wala.sourceforge.net/wiki/*

環境

- Windows XP
- Eclipse 3.5.2
- Java 1.5.22
- WALA 1.3.1M1

変更履歴

2010/11/17：Ver 1.0：公開開始