JPCERT CC®

JEB Plugin 開発チュートリアル 第4回

– JEB PluginからASTを扱う –

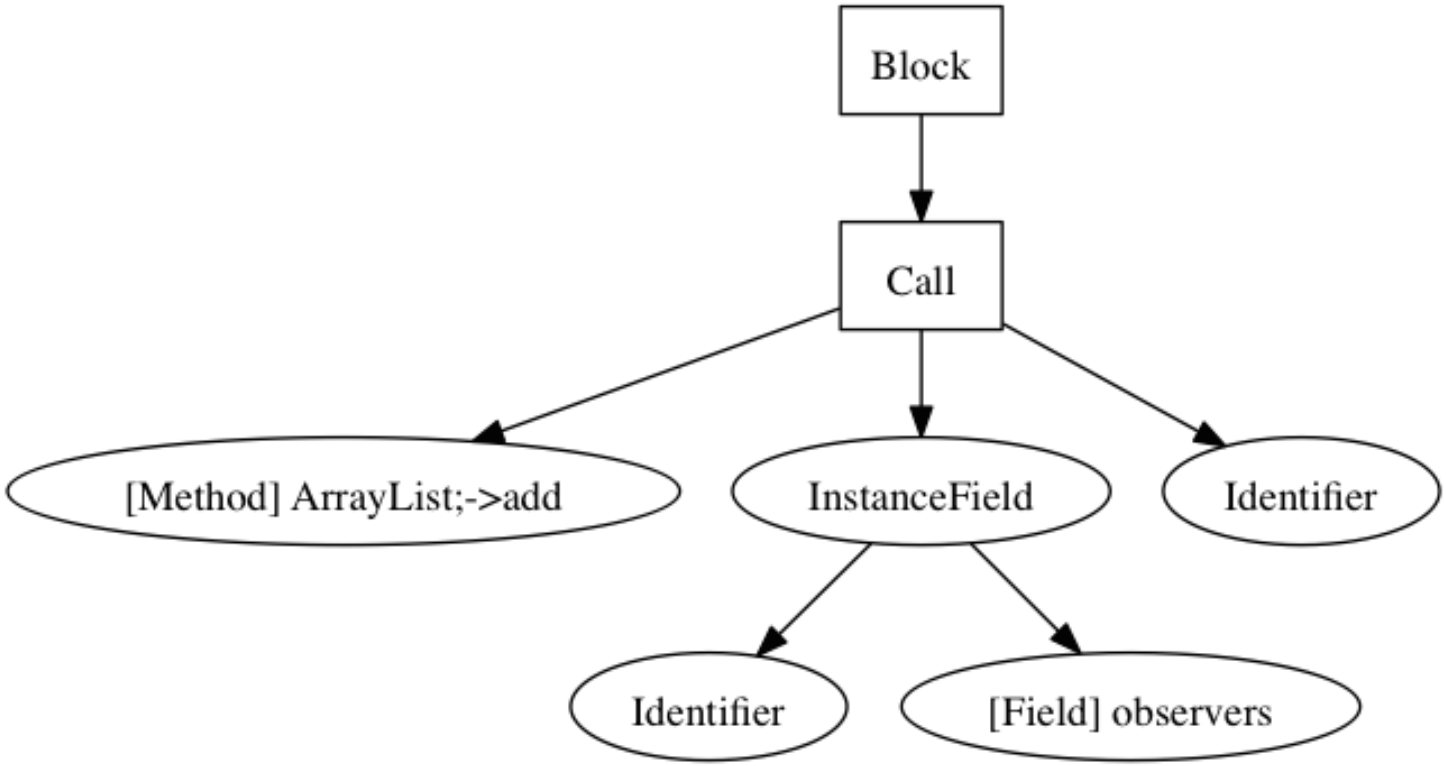
一般社団法人JPCERTコーディネーションセンター

目次

- 第0回 JEBとは？
- 第1回 JEB Pluginとは
  - 1. JEB Pluginの使い方
  - 2. JEB Pluginの構造
  - 3. JEBのUIを利用するためのAPI
  - 4. ViewとSignature
- 第2回 DEXファイルの構造を理解する
  - 1. DEXファイルの構造
  - 2. jeb.api.dex
  - 3. クロスリファレンス
- 第3回 バイトコードについての理解
  - 1. CodeItem
- **第4回 JEB PluginからASTを扱う**

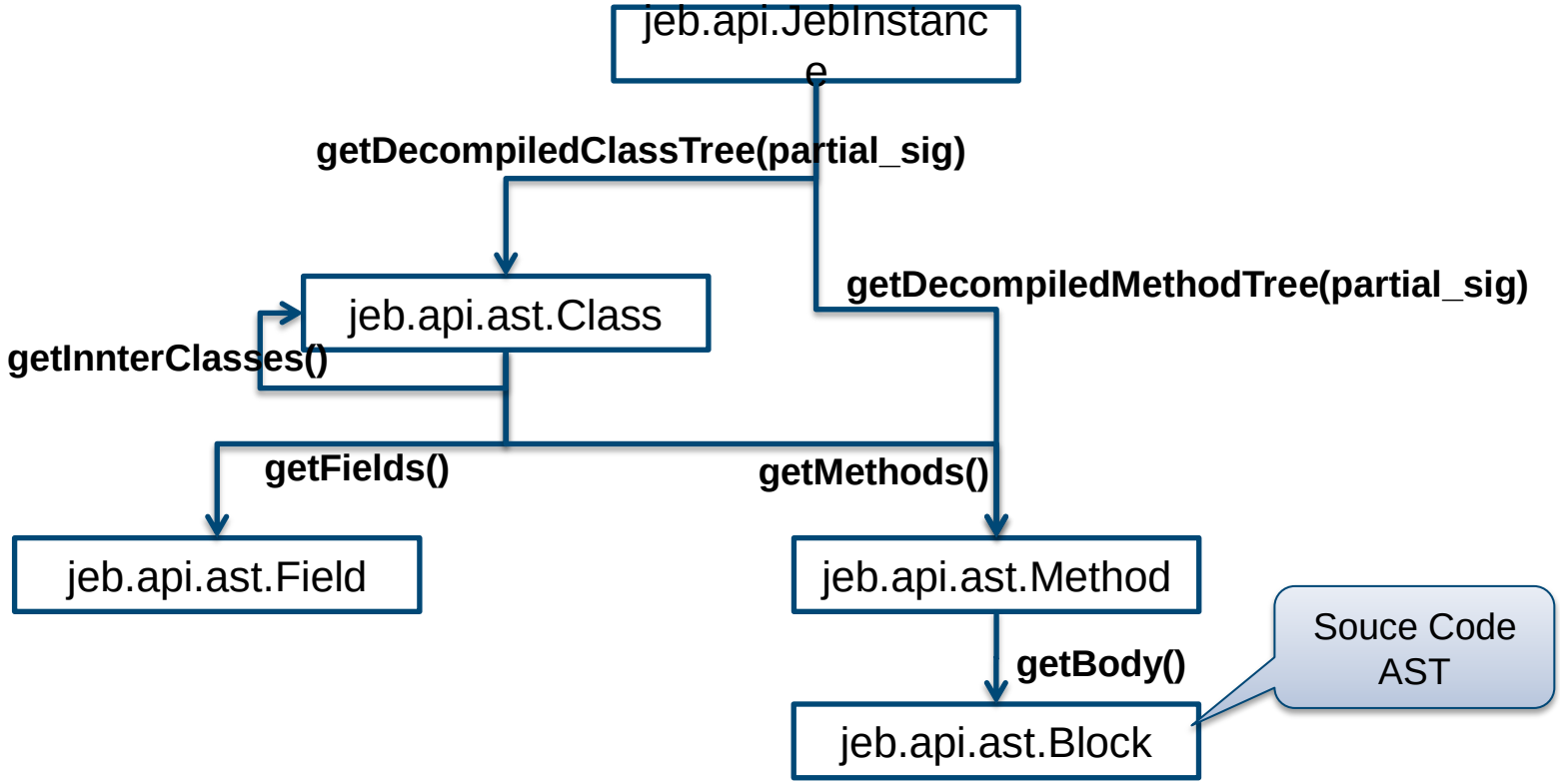
ASTとは

- 抽象構文木
  - —Abstract Syntax Tree
  - —言語(プログラム)の意味に関係のないものを取り除きツリー構造で表したもの



JEBとAST

- JEBでは、DEXをデコンパイルして得られるJavaコードが、ASTとして内部的に管理されている
- JEB PluginからASTにアクセスするためのAPIが用意されている



ASTの取得

- JEB PluginからASTを取得するには、事前にJavaのコードにデコンパイルしておく必要がある
  - JebUI.decompileClass()
- メソッドの取得
  - JebInstance.getDecompiledMethod(sig)
  - jeb.api.ast.Class.getMethods()
    - Methodのリストを取得
- クラスの取得
  - JebInstance.getDecompiledClassTree(sig)

例題1

- AST クラス/メソッドを取得しよう
  —jeb.api.ast.Classの使い方を理解する

[例題1] AST クラス/メソッドの取得

- 課題
  - Assembly Viewのキャレット位置のクラス(jeb.api.ast.Class)を取得し、内包するフィールドとメソッドの一覧を表示する
- 期待する出力結果

```
Landroid/support/v4/app/BackStackState;
target class: Landroid/support/v4/app/BackStackState;
[Method]
    Landroid/support/v4/app/BackStackState;-><clinit>()V
    Landroid/support/v4/app/BackStackState;-><init>(Landroid/os/Parcel;)V
    Landroid/support/v4/app/BackStackState;-><init>(Landroid/support/v4/app
    Landroid/support/v4/app/BackStackState;->describeContents()I
    Landroid/support/v4/app/BackStackState;->instantiate(Landroid/support/v
    Landroid/support/v4/app/BackStackState;->writeToParcel(Landroid/os/Parc

[Field]
    Landroid/support/v4/app/BackStackState;->CREATOR:Landroid/os/Parcelable
    Landroid/support/v4/app/BackStackState;->mBreadCrumbShortTitleRes:I
    Landroid/support/v4/app/BackStackState;->mBreadCrumbShortTitleText:Ljav
```

- ヒント
  - デコンパイルの実行
    - decompileClass()
  - キャレット位置のSignatureの取得
    - getCodePosition().getSignature()
  - jeb.api.ast.Classの取得
    - getDecompiledClassTree()
  - メソッド/フィールド一覧の取得

例題1の解答例

- DisplayAstClass.py

[解説] AST クラス/メソッドの取得

```
sig = view.getCodePosition().getSignature()
print sig

csig = re.sub(r'->.*$', '', sig)
ui.decompileClass(csig, False)

cls = jeb.getDecompiledClassTree(csig)
print "target class: " + cls.getType()

print "[Method]"
for m in cls.getMethods():
    print "¥t" + m.getSignature()
print ""

print "[Field]"
for f in cls.getFields():
    print "¥t" + f.getSignature()
```

キャレット位置のSignatureを取得する

取得したSignatureをClass Signatureに変換し、クラスをデコンパイルする

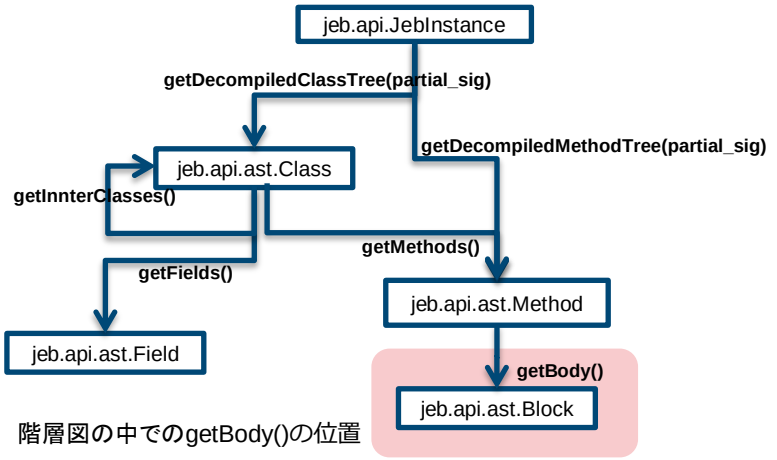
Class Signatureを使用して、クラスを取得する

メソッドを取得

フィールドを取得

ASTのコードの取得

- jeb.api.ast.Method.getBody()
  —このメソッドを使用することでBlockを取得できる



階層図の中でのgetBody()の位置

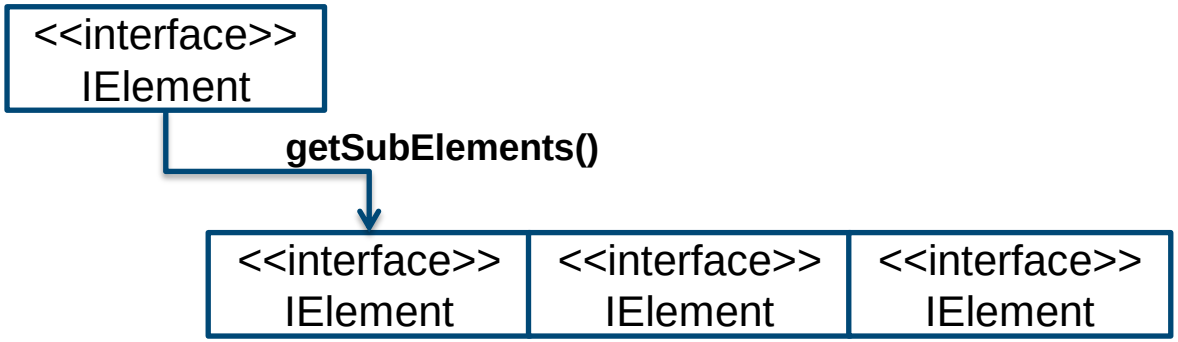
メソッドのコード全体をBlockとして取得する

```
public void doWhileSimpleLoop() {
    int v0 = 0;
    do {
        ++v0;
        Log.i(Loop.TAG, String.format("do-while loop:%d", Integer.valueOf(v0)));
    }
    while(v0 < 1024);
}
```

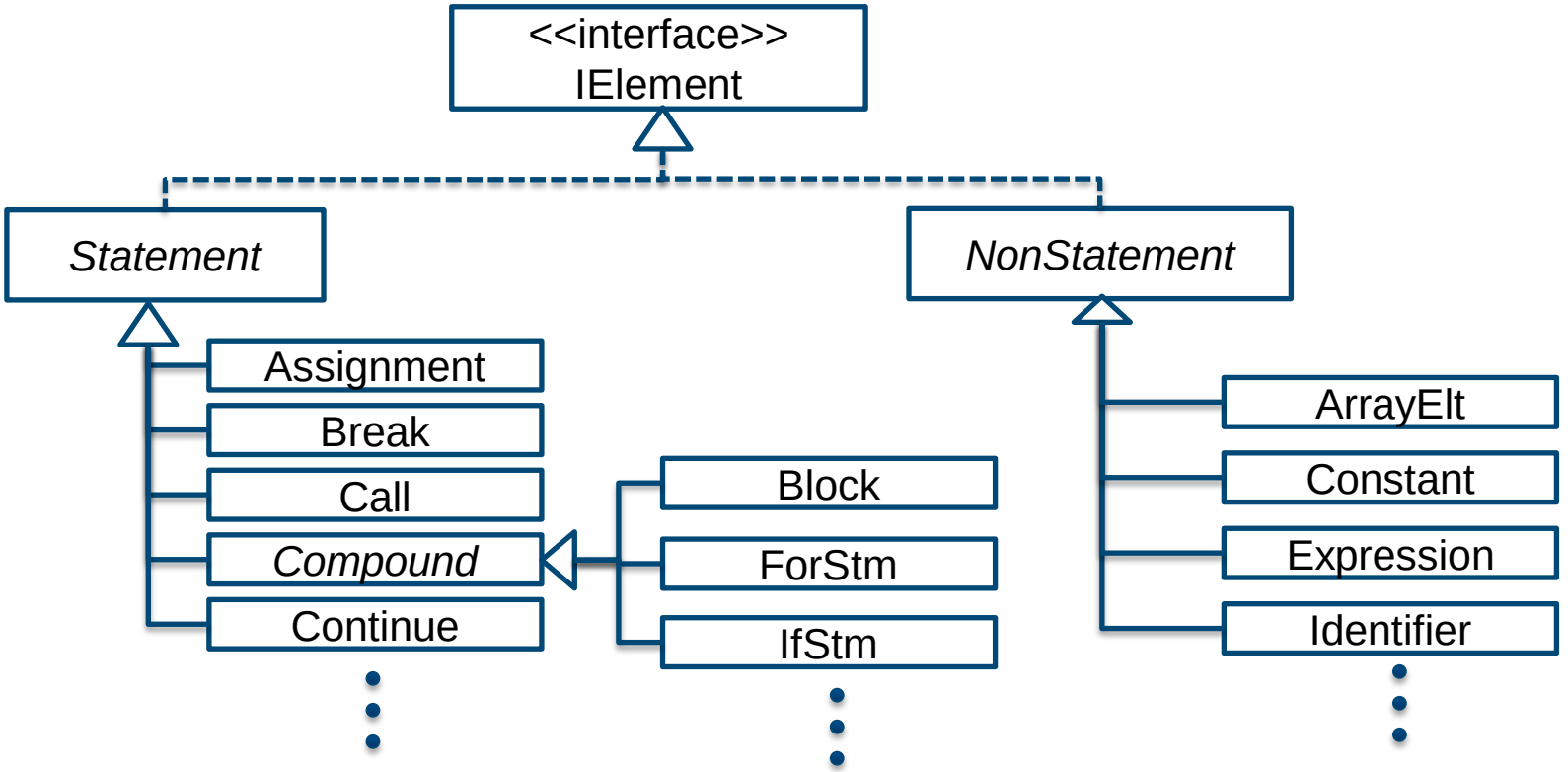
IElement

- すべてのASTノードのベースとなるインターフェイス

- IElement.getSubElements()
  —ASTの子ノードのリスト取得
- IElement.replaceSubElements ()
  —ASTの子ノードの置き換え



ASTのクラス構造

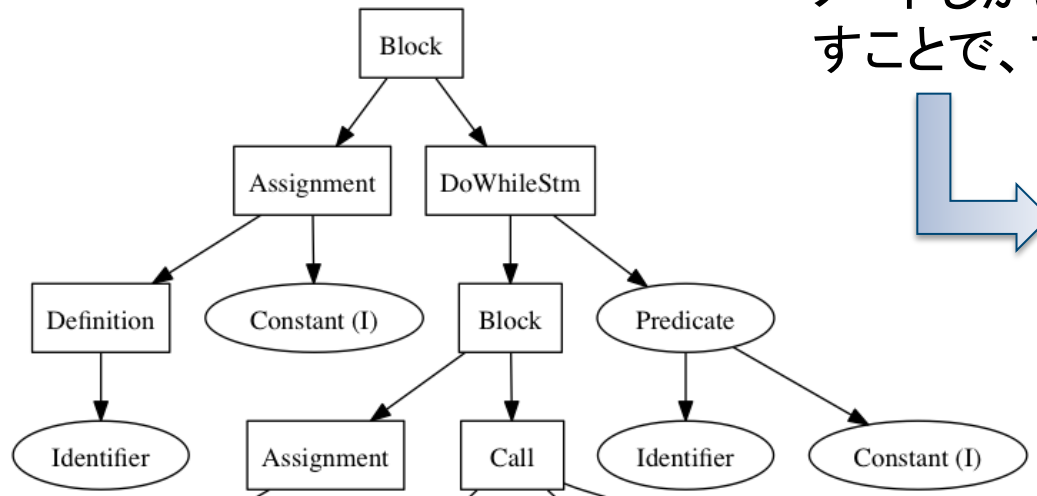


すべてのノードがStatement(文ノード)のNotStatement(非文ノード)のサブクラスで構成

- Statement 文ノード
  - Assignment: 値の代入 (例 str = “abc”;)
  - Call: メソッド呼び出し (例 str.startsWiths(“a”);)
- NonStatement 非文ノード
  - Expression: 式 (例: a+b)

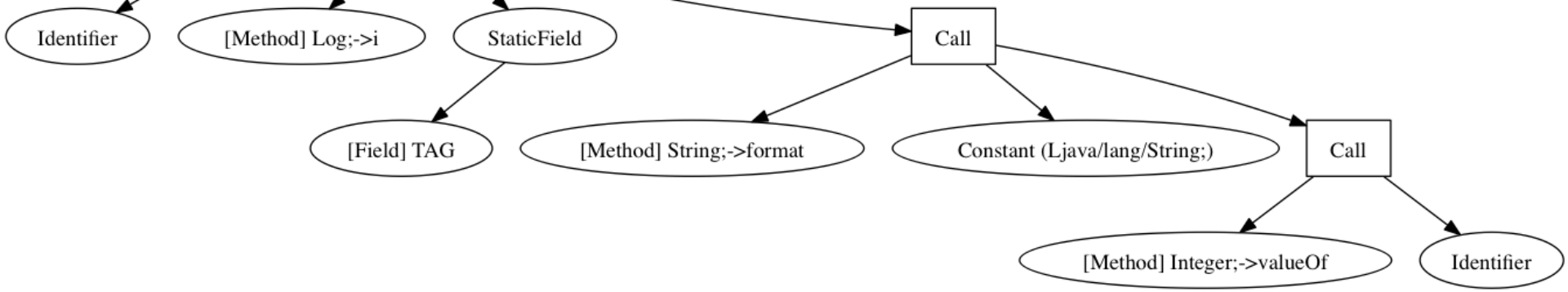
ASTの要素を辿る

IElement.getSubElements()では直下のノードしか取得できないが、再帰的に呼び出すことで、すべてのノードを取得できる




```
def traverse(elem):
    print elem.__class__.__name__

    for e in elem.getSubElements():
        traverse(e)
```



例題2

- ASTのtraverse
  - —IElement.getSubElements()の使い方を理解する

[例題2] ASTのtraverse

- 課題
  - Java Viewでキャレット位置のメソッドのASTをtraverseする
  - ASTの各ノードのクラス名を表示する
- 期待する出力結果

```
[0]Block
 [1]Call
  [2]Method
  [2]New
   [3]Method
    [4]Definition
     [5]Identifier
    [4]Definition
     [5]Identifier
    [4]Definition
     [5]Identifier
```

- ヒント
  - コードブロックの取得: jeb.api.ast.Method.getBody()
  - IElement.getSubElements()
  - elementを受け取って、名前を表示するメソッドを作る
    - def traverse(self, elem, depth=0):
    - getSubElements() → 再帰呼び出し
    - 再帰呼び出しでdepthを+1する
      - print("[%d] %s" % (depth, class_name))
  - クラス名の取得: obj.__class__.__name__

例題2の解答例

- TraverseAst.py

[解説] ASTのtraverse

```
sig = view.getCodePosition().getSignature()
if sig in dex.getMethodSignatures(False):
    method = jeb.getDecompiledMethodTree(sig)
    self.traverse(method.getBody())
else:
    print "caret is not in method.: " + sig
```

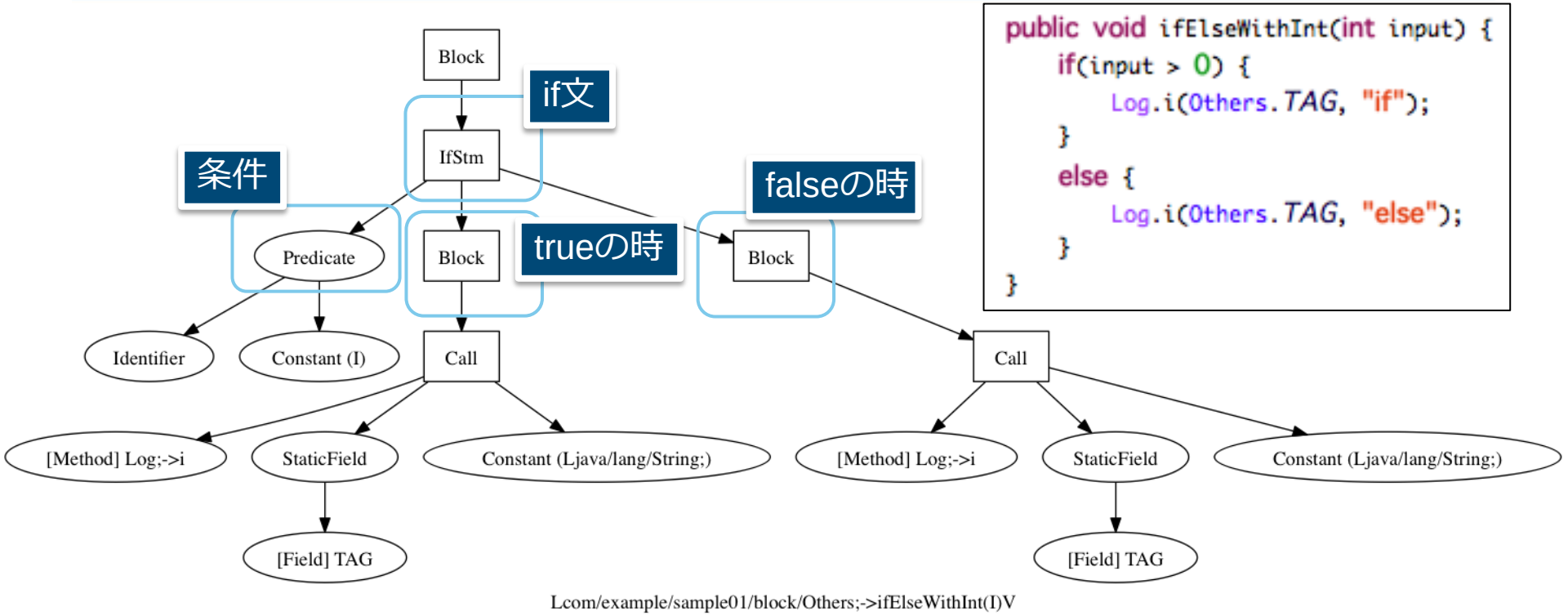
getSubElements() を再帰的に呼び出すメソッドに処理を投げる

```
def traverse(self, elem, depth=0):
    print("%s[%d]%s" % (' '*depth, depth, elem.__class__.__name__))

    for e in elem.getSubElements():
        self.traverse(e, depth+1)
```

getSubElements()は直下のノードしか取得できないので、これを再帰的に呼び出すメソッドを作る

IElementをTraverseする限界




```
public void ifElseWithInt(int input) {
    if(input > 0) {
        Log.i(Others.TAG, "if");
    }
    else {
        Log.i(Others.TAG, "else");
    }
}
```

- IElement.getSubElements()でTraverseするだけでは限界がある
  - 子ノードのリストが取得できるだけでそれぞれが何を表しているかまでは分からない。

各クラスに実装されているメソッドで情報を取得する

ノード固有の情報へのアクセス方法

- 例: IfStm(if文)の場合
  - —条件の取得　※条件は複数存在する可能性があるのでindex指定する
    - IfStm.getBranchPredicate(index)※
  - —条件一致した時の処理の取得
    - IfStm.getBranchBody(index)※
  - —else節の取得
    - IfStm.getDefaultBlock()
- 例: Call(メソッド呼び出しの場合)
  - —呼び出すメソッドの取得
    - Call.getMethod()
  - —引数の取得
    - Call.getArguments()

```
if(p) {
  b
 } else if(p1) {
  b1
 } else {
  b2
}
```

例題3

- MethodノードのSignatureの出力
  - —jeb.api.ast.Method.getSignature()の使い方を理解する

[例題3] MethodノードのSignatureの出力

- 課題
  - Java Viewでキャレット位置のメソッドのASTをtraverseする
  - ASTの各ノードのクラス名を表示する
  - 但し、Traverseしているノードがの場合は、”[METHOD] Signature”のように表示する
- 期待する出力結果

```
[0]Block
 [1]Call
  [2][METHOD] Lcom/example/contentprovider/MainActivity$RssTask;->execute([Ljava/lang/Object;)Landroid/os/AsyncTask;
  [2]New
   [3][METHOD] Lcom/example/contentprovider/MainActivity$RssTask;-><init>(Lcom/example/contentprovider/MainActivity;Landroid/app/Activity;)V
    [4]Definition
     [5]Identifier
    [4]Definition
     [5]Identifier
    [4]Definition
     [5]Identifier
    [4]Block
```

- ヒント
  - Traverse自体は先ほどと同じ
  - オブジェクトのクラス判定
    - if isinstance(obj, jeb.api.ast.Method):
  - メソッドのSignatureの取得
    - jeb.api.ast.Method.getSignature()

例題3の解答例

- TraverseAst2.py

[解説] MethodノードのSignatureの出力

```
sig = view.getCodePosition().getSignature()
if sig in dex.getMethodSignatures(False):
    method = jeb.getDecompiledMethodTree(sig)
    self.traverse(method.getBody())
else:
    print "caret is not in method.: " + sig

def traverse(self, elem, depth=0):
    if isinstance(elem, jeb.api.ast.Method):
        name = "[METHOD] %s" % elem.getSignature()
    else:
        name = elem.__class__.__name__
    print("%s[%d]%s" % (' '*depth, depth, name))
    for e in elem.getSubElements():
        self.traverse(e, depth+1)
```

取得したノードがjeb.api.ast.Method
のインスタンスかどうかチェックする

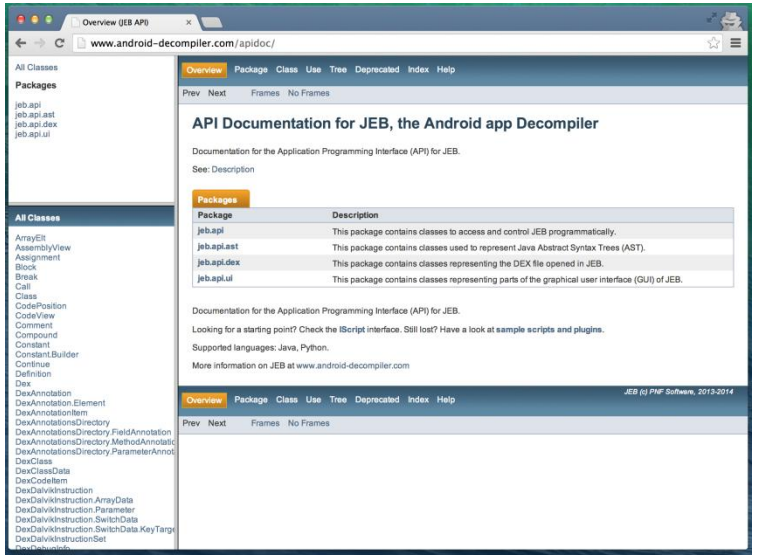
Methodノードの
Signatureを出力する

再帰的に呼び出す

[参考] APIリファレンス

- APIのリファレンス
  - —http://www.android-decompiler.com/apidoc/

- jeb.api
  - — PluginからJEBにアクセスするためのクラスが含まれているパッケージ
- jeb.api.ast
  - — ASTを使用するためのクラスが含まれているパッケージ
- jeb.api.dex
  - — JEBで開いたDEXファイルを使用するためのクラスが含まれているパッケージ
- jeb.api.ui
  - — JEBのUIを操作するためのクラスが含まれているパッケージ



[参考] jeb.api.ast.Statement継承クラス

Class	Desc.	Example
Assignment	値の設定	str = “abc”;
Break	ブレイク文	break;
Call	メソッド呼び出し	str.equals(“abc”);
Compound	ブロック	
Continue		continue;
Definition	変数定義	int a;
Goto		goto label;
Label		labelXX:
Monitor	syncronize	__monitor_enter(lock); __monitor_exit(lock);
New	インスタンス生成	new Exception(“aaa”);
NewArray		int[] a = {1,2,3};
Return		return true;
Throw	例外投入	throw e;

[参考] Compound (Statement)

Class	Example

Block	{
	…
	}

DoWhileStm	do { … } while(true);

ForStm	for(i=0; i<10; i++) {
	…
	}

IfStm	if (xxx) {
	…
	} else {
	}

SwitchStm	switch(xxx) {
	case 0:
	..
	}

TryStm	try {
	} catch (Exception e) {
	}

WhileStm	while (true) { }

[参考] jeb.api.ast.NonStatement継承クラス

Class	Desc.	Example
ArrayElt	配列要素へのアクセス	array[index]
Class	クラス	
Constant	定数	
Expression	式	a + 1 a * ((int)b – foo())
Field	フィールド	
Identifier	変数	
InstanceField	インスタンスフィールド	
Method	メソッド	
StaticField	スタティックフィールド	
TypeReference	型参照(オブジェクトのクラス確認)	if (pet instanceof Dog) { … }