自然言語に基づく変数の命名方法の提案

70711041
鳥居克哉

目次1

- 研究の背景
  - 識別子とは
  - コンソールに"Hello Worldと表示するjavaプログラムの例"
  - 識別子の命名にはいくつかの制約がある
  - 識別子の命名は重要である
  - 識別子の命名に関する暗黙知の存在
  - 識別子の可読性に自然言語の品詞が影響している可能性

目次2

- 仮説
- 研究の目的
- 研究、調査の手法
  - javaについて
  - OSSについて
  - ASTによるソースコード解析
  - JavaParserの採用について
- 進捗

研究の背景

識別子とは

プログラム言語における変数名や関数名を表す名前[^1]

javaでは

- クラス名
- メソッド名
- 変数名
- パッケージ名 ...etc

例としてJavaを挙げた理由は後述する

コンソールに"Hello World"と表示するjavaプログラムの例

```java
package sample

public class Main {
  public static void main(String[] args) {
    String text = "Hello World";
    System.out.println(text);
  }
}
```

このソースコードに含まれる識別子

sample, Main, main, String, args, text, System, out, println

識別子の命名にはいくつかの制約がある

Java SE 8の場合[^2]

- 識別子の長さは無制限
- 識別子の命名にはUnicodeの「A-Z, a-z, 0-9, _, $」が使用可能
- 識別子の大文字小文字は区別される
- 識別子の最初の文字は`Character.isJavaIdentifier(int)`がtrueを返す文字で始まる
- 識別子は予約後と重複してはいけない

制限はあるが、ある程度自由に識別子は命名することが可能と考えられる

識別子の命名は重要である

- ソフトウェア保守の時間の多くはソースコードを理解することに費やされる
- 識別子に単語を用いることでプログラムの理解速度が19%向上[^3]

識別子の命名によっては保守にかかるコストの削減が期待される

識別子の命名に関する暗黙知の存在を示唆

- 一般に命名の重要性は広く認識されている
  - コーディングスタイルの存在
  - コーディングスタイルは命名の本質的な手法を定めていない[^2]
- 命名によってソースコードの理解度が左右される

命名に関する暗黙知が開発者の中に存在する可能性が考えられる

識別子の可読性に自然言語の品詞が影響している可能性

- 人間は自然言語の使い手である
- 人間がプログラムを書く時、識別子に自然言語で命名する
- 自然言語には「品詞」が存在する
- 品詞に基づくコーディングスタイルも存在する
  (メソッド名は動詞か動詞句になるよう命名する[^4])

仮説

品詞が識別子の命名に影響を与えているのではないか

研究の目的

- 暗黙知であったプログラム中の識別子の命名と品詞の関係を明らかにする
- 品詞の組み合わせに基づく命名規則の提案

研究、調査の手法

1. JavaプログラムをJavaParserを使ってAST解析し、識別子を抽出する
2. 抽出した識別子を単語ごとに分解する
3. 2で分解した単語がどの品詞になるのかを判定する
4. 3で得られた結果をもとに統計的手法を用いて相関を求める
5. 4の分析結果から命名の手法を提案する

javaについて

- オブジェクト指向言語、静的型付け言語
- 1995年にSun Microsystemsにより初めてリリースされたプログラミング言語
- 歴史が長い
- エコシステムが充実している
    - 複数のIDEの存在
      (Eclipse, Intellij, NetBeans)
    - OSSリポジトリの数
      (2020-10月時点で1,622,484のリポジトリが存在[^5])

OSSについて

OSS: オープンソースソフトウェア(Open Source Software)

- 無償で公開されているソフトウェア
- 利用、改変、再配布が自由
- 様々な人がメンテナンスをしている
  (facebook/reactでは1500人を超える[^6])
    - 可読性の高さが期待される

OSS=暗黙知の集積場

ASTによるソースコード解析

AST: (Abstract Syntax Tree)抽象構文木

- 中間表現のひとつ
  (プログラミング言語 → 中間言語 →アセンブリ言語)
- プログラムをツリー構造で保持
- ソースコードの情報を得るために有効

JavaParserの採用について

- AST解析ツールの選択肢は主に以下二つ
  - `JDT`
  - `JavaParser`
- ドキュメントが充実している[^7]
- JavaParserはEclipseに依存していない

進捗

ツールはおおきく以下の二つ

1. 解析作業自動化＆AST解析
2. 識別子を単語単位で分解＆品詞を判定

1のAST解析部以外は完成
2はこれから

参考文献

[^1]: オーム社(2001)『情報技術用語大辞典』相磯秀夫監修, オーム社, p288, l60 ~ l63
[^2]: James Gosling・Bill Joy・Guy Steele・Gilad Bracha・Alex Buckley(2015)『The Java® Language Specification:Java SE 8 Edition』
https://docs.oracle.com/javase/specs/jls/se8/jls8.pdf (参照2020-10-31)p22-24
[^3]: Johannes Hofmeister・Janet Siegmund・Daniel V. Holt(2017)「Shorter Identifier Names Take Longer to Comprehend」『2017 IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER)』
IEEEhttps://www.infosun.fim.uni-passau.de/publications/docs/HoSeHo17.pdf(参照2020-10-25)
[^4]:Google(発行年不明)「Google Java Style Guide」
https://google.github.io/styleguide/javaguide.html(参照2020-10-10)

[^5]: Github 「Search・java」https://github.com/search?q=java(参照2020-10-20)

[^6]: Github「facebook/react」https://github.com/facebook/react(参照2020-10-20)

[^7]: Nicholas Smith, Danny van Bruggen and Federico Tomassetti(2019)「JavaParser: Visited:JavaParser: Visited」Leanpubhttps://www.livingtech.com.cn/media/javaparservisited.pdf(参照2020-10-21)