Compiler Course Project Documentation

Mohammad Mahdi Abdollahpour

Introdcution

The goal of this project is to implement a software to mimic the functionality of Contain and ContainIn query in the Scitools Understand software analysis toolkit. I used ANTLRv4 in C# runtime along with SQLite databse to save the results. To find more information about Understand, the database schemas and the terminology visit here.

Contain and ContainIn

In Understand, Contain and ContainIn queries are inverse of each other. I will just explain Contain functionality. This query calculates all the references between the following entities and their containing package:

- Java Class
- Java Interface
- Java Enum
- Java Annotation Type

To be more specific, Understand searches all .java files and finds type declarations for the above entities and foreach of the declaration, it defines a Contain reference with Id of the declared entity (_ent_id in the database) and Id of the package (_scope_id in the database).

In addition to the user defined entities, Understand defines Contain references based on import statements. For statements like java.x.y.C, it defines C as an unknown class entity and java.x.y as an unknown package and defines a Contain reference between them. The rationale is that it assumes that the java.x.y package should contain the definition for C. For statements like java.x.y.*, it searches the whole source file to find any usage of any unkown class. However Understand can not successfully define an entity with a meaningful long name (package name + class/interface name) for such cases, thus it fills the reference table and entity table with a lot of useless orphan entities and references. Because of this, I ignored the second type of import statements in my implementation.

Implementation

JavaExplorer: It is a high-level class that explores .java files in the source directory and calls JavaPackageVisitor on each of them. It saves extracted entities and references using the visitor. It also has a method to export the extracted data into the SQLite databse using the Dapper ORM.

JavaPackageVisitor: It is an ANTLR visitor. It's main responsibility is to find entities (classes, interfaces, enums, annotation types, and packages) and Contain references between them.

EntityKindUtils: It contains tools to extract exact entity kind names according to the predefined OpenUnderstand database. It uses the information from the ANTLR context (mainly the modifiers) to construct the kind name string. JavaExplorer uses these names to find the correct entity id from the database.

Usage

In order to use this project, you have to follow these steps:

1. Clone the OpenUnderstand project.
2. Modify openunderstand/ound.py
    a. Change dbname and project directory based on your need
3. Build and run the C# project using two command line arguments seperated with space:
    a. java project directory path
    b. SQLite .db file path

Example: There are some java projects in the benchmark directory at the OpenUnderstand repo. To run on benchmark/jhotdraw-develop we can do:

1. git clone https://github.com/m-zakeri/OpenUnderstand/
2. cd OpenUnderstand/openunderstand
3. Modify openunderstand/ound.py
    a. dbname=database_ound.db
    b. project_dir=Path('../benchmark/jhotdraw-develop').resolve().as_posix()
4. python ound.py
5. cd ../../UndContain/UndContain
6. dotnet run ../../OpenUnderstand/benchmark/jhotdraw-develop ../../OpenUnderstand/openunderstand/database_ound.db