JClassDiagram 迭代三设计文档

第31组

团队介绍

姓名	学号	QQ号	职责
刘畅	221900071	1053432766	组长
梅天豪	221900056	2242974635	组员
潘智杰	221900313	3160400570	组员
庞鸿博	221900314	3231417980	组员

项目地址

> https://git.nju.edu.cn/software_group_31

项目目标

在迭代三的开发周期中，我们将重点扩展JClassDiagram的系统功能，使其具备多文件解析与处理能力，以有效适配实际Java项目开发需求。我们将构建具备参数配置功能的命令行交互界面并集成设计模式识别模块，重点针对单例模式（Singleton Pattern）和策略模式（Strategy Pattern）进行特征提取与模式验证，同时开发支持用户自定义配置的分析器组件，允许开发人员根据具体需求调整模式检测规则。通过本阶段的升级，系统将形成完整的项目级代码分析能力，并建立可扩展的基于类图的程序分析框架。

系统设计

总体设计

```
./src/main
└─java
    |  JClassDiagram.java
    |  Main.java
    |
    ├─command
    |      ClassOp.java
    |      CommandLineTool.java
    |      FieldOp.java
    |      FunctionOp.java
    |      Query.java
    |
    ├─diagram
    |      ClassDiagram.java
    |      ClassDiagramGenerator.java
    |      ClassParser.java
    |      SmellAnalyzer.java
    |
    ├─graph
    |      ClassMap.java
    |      Graph.java
    |
    ├─model
    |      AbstractClassModel.java
    |      BaseModel.java
    |      ClassModel.java
    |      EnumModel.java
    |      FieldModel.java
    |      InterfaceModel.java
    |      MethodModel.java
    |
    └─utils
```

```
            AnalyzerUtil.java
            CommonUtil.java
            Factory.java
            GlobalVar.java
```

类设计

> 1. 绘制当前系统（三次迭代）的 **UML 类图**。由于大的类图往往非常杂乱，建议你手动删除一些不重要的关系边、不重要的类使类图更加清晰。

迭代一类图

迭代二类图

迭代三类图

设计模式

命令模式：CommandLineTool 作为命令解析器，根据用户输入决定调用哪个操作。

中介者模式：CommandLineTool使用ClassDiagram作为中介者，有效降低了Command类内部的耦合

弹性设计

1. **高内聚、低耦合的模块划分**

- 每种操作都集中在对应类中：`ClassOp`，`FieldOp`，`FunctionOp`，`Query`。

- 方便以后新增其他操作类型，

  2.**灵活的指令解析器**

- `CommandLineTool` 的指令设计非常灵活，支持组合参数、可选项（，利于快速新增或修改命令。

重要类：

1.**CommandLineTool**

属性

类型	名称	描述
`ClassDiagram`	`diagram`	承载所有类结构及图关系的主类图对象

方法

返回类型	方法名	描述
`String`	`execute(String command)`	解析并执行一条命令，返回输出结果（若有）
`String`	`getArg(String[] parts, String key)`	获取指定命令行参数的值
`String`	`getOptionalArg(String[] parts, String key, String defaultValue)`	获取可选参数的值，若不存在则返回默认值

2.**ClassOp**

方法

返回类型	方法名	描述
`void`	`addClass(String name, boolean isAbstract, List<AbstractClassModel> class_list)`	添加一个类对象到类图列表中
`void`	`addInterface(String name, List<AbstractClassModel> class_list)`	添加一个接口到类图
`void`	`addEnum(String name, String valueList, List<AbstractClassModel> class_list)`	添加一个枚举类型及其常量列表
`void`	`deleteClassInterfaceEnum(String name, List<AbstractClassModel> class_list, Graph graph)`	删除类/接口/枚举，并删除图中相关联的关系

3.**FieldOp**

方法

返回类型	方法名	描述
`void`	`addField(String target, String fieldName, String type, String access, boolean isStatic, List<AbstractClassModel> class_list)`	向指定类添加一个字段
`void`	`deleteField(String target, String fieldName, List<AbstractClassModel> class_list)`	删除类中的指定字段
`void`	`modifyField(String target, String fieldName, String newName, String newType, String newAccess, Boolean isStatic, List<AbstractClassModel> class_list)`	修改字段的属性（通过删除原字段再添加新字段实现）

4.**FunctionOp**

方法

返回类型	方法名	描述
`void`	`addFunction(String target, String functionName, String returnType, String params, String access, boolean isStatic, boolean isAbstract, List<AbstractClassModel> class_list)`	向类中添加一个方法（支持参数、返回类型、static/abstract）
`void`	`deleteFunction(String target, String functionName, List<AbstractClassModel> class_list)`	删除类中指定方法
`void`	`modifyFunction(String target, String functionName, String newName, String newParams, String newReturn, String newAccess, Boolean isStatic, Boolean isAbstract, List<AbstractClassModel> class_list)`	修改方法的名称、参数、返回类型等属性（通过重建实现）

5.**ClassDiagramGenerator**

方法

返回类型	方法名	描述
`ClassDiagram`	`parse(Path sourcePath)`	主方法：解析传入的 `.java` 文件路径或文件夹路径，返回类图对象
`void`	`collectJavaFiles(Path directory, List<Path> javaFiles)`	递归收集目录及子目录下所有 `.java` 文件，存入 `javaFiles` 列表
`ClassDiagram`	`createClassDiagramFromFiles(List<Path> javaFiles)`	根据收集到的 `.java` 文件路径列表生成 `ClassDiagram` 对象

数据结构与算法设计

命令行实现：CommandLineTool通过统一命令语法解析类/字段/函数，实现了对他们的操作以及Undo 撤销机制,查询与坏味道检测算法

实际工作安排

>

姓名	任务
刘畅	测试与分析
庞鸿博	设计与文档
梅天豪	编码
潘智杰	编码

运行结果

- 测试用例中 `lab3/monopoly`

- 我们的JClassDiagram Java Project