DesPaD
(Design Pattern Detector)
Manual

Murat Oruc

1. INTRODUCTION

DesPaD (Design Pattern Detector) tool is made for finding GoF design patterns from a given source code which is in Java. It uses Subdue, sub-graph mining tool during the detection process. DesPaD only runs on Linux-based operating systems. We define the steps of using DesPaD in the next section.

2. IMPLEMENTATION

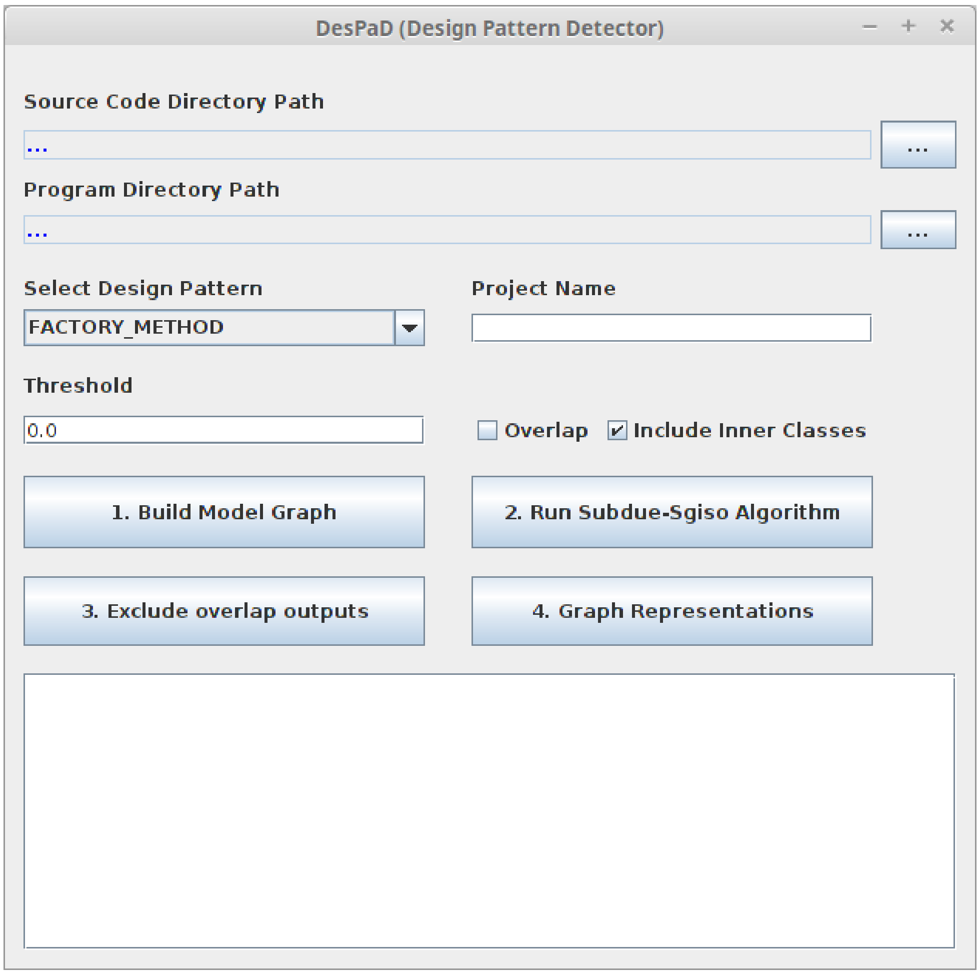
After unzipping the DesPaD file, you will see the shortcut icon in DesPaD folder. (See Figure 1)

Figure 1. DesPaD shortcut.

Next install the "Graphviz" application because DesPaD uses it in its source code. Run the code below in terminal.

- **sudo apt-get install graphviz**

Clicking the shortcut you will see the user interface. (See Figure 2)


DesPaD (Design Pattern Detector)

Source Code Directory Path: ...

Program Directory Path: ...

Select Design Pattern: FACTORY_METHOD

Project Name:

Threshold: 0.0

☐ Overlap ☑ Include Inner Classes

Figure 2. DesPaD user interface.

In “Source Code Directory Path” section, you will select a root directory of a software projects which is written in Java programming language. DesPaD will find all java files in choosen directory recursively and parse them in the background. (See Figure 3)

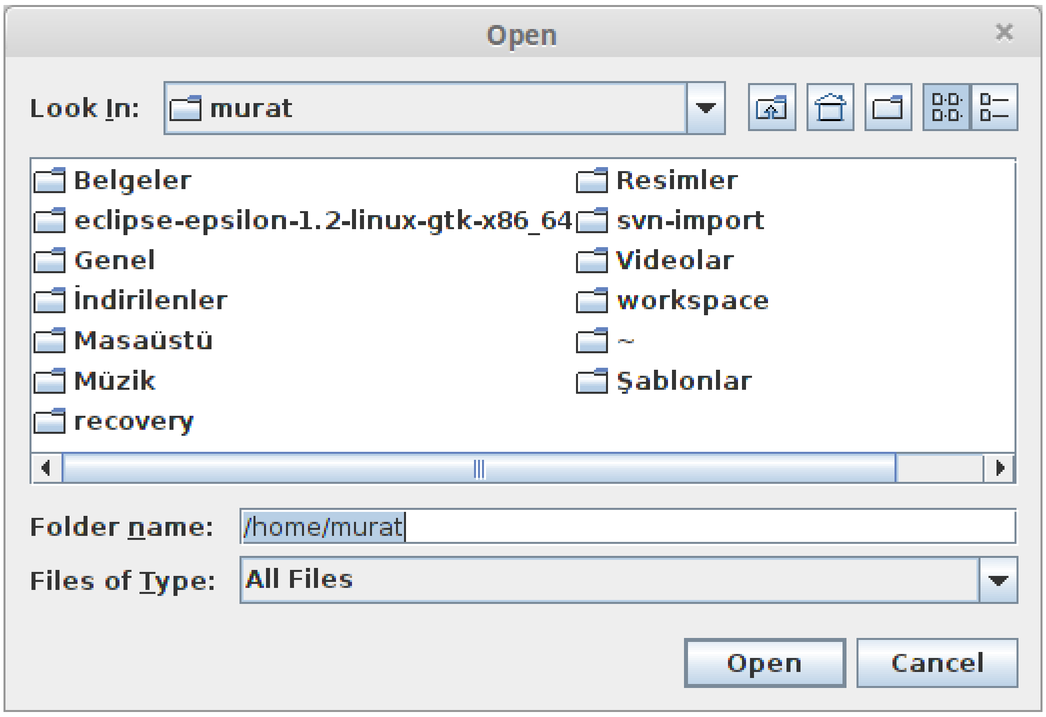


Figure 3. Selection of a directory interface.

In “Program Directory Path” section, you will select the directory where the “DesignPatternDetection. jar” file exists. DesPaD use this directory for running shell script file in Subdue and creating some output files.

In “Select Design Pattern Name” section, you will see 23 GoF design pattern names. You have to select one of them for searching it in the given source code. Some of the patterns have more than one templates, this is because they cannot be defined by only one template.

In “Project Name” section, you will write the directory name which outputs of the algortihm are saved into. Due to finding the outputs properly, it is better to give a name, similar to the software project.(See Figure 4)

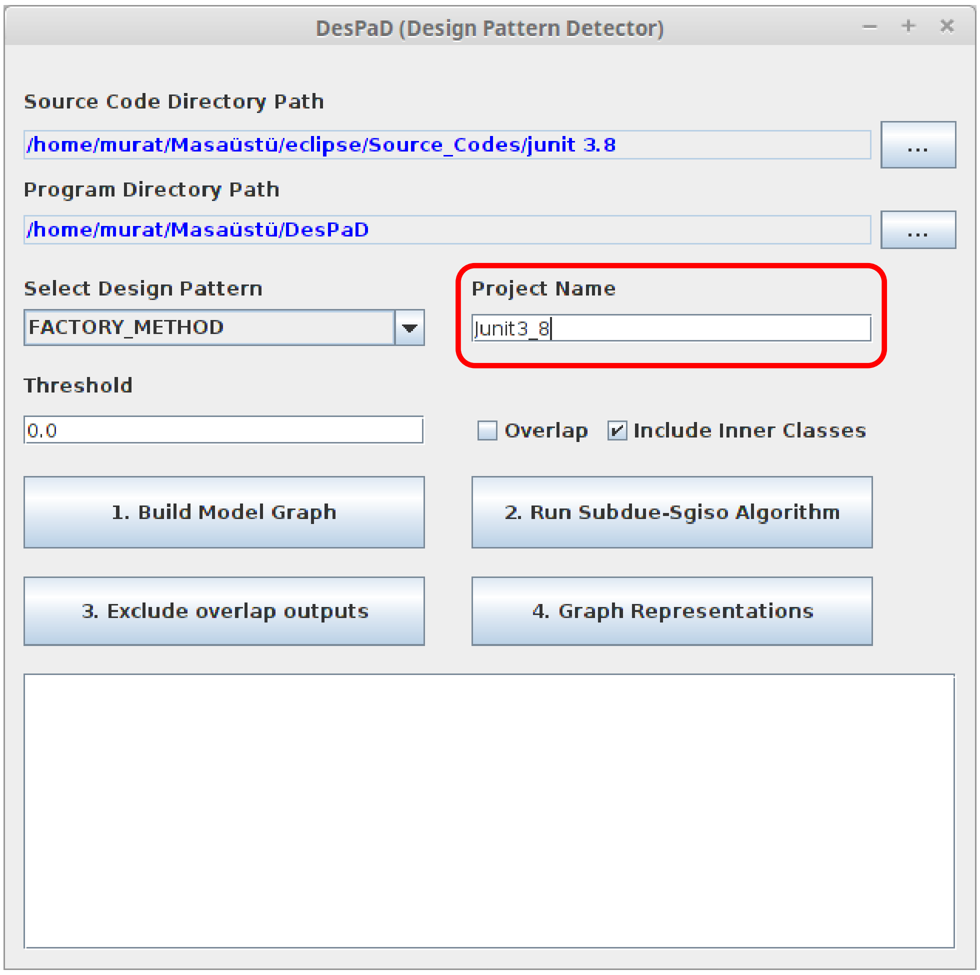

DesPaD (Design Pattern Detector)
Source Code Directory Path
/home/murat/Masaüstü/eclipse/Source_Codes/junit 3.8
...
Program Directory Path
/home/murat/Masaüstü/DesPaD
...
Select Design Pattern
FACTORY_METHOD
Project Name
Junit3_8
Threshold
0.0
Overlap
Include Inner Classes
1. Build Model Graph
2. Run Subdue-Sgiso Algorithm
3. Exclude overlap outputs
4. Graph Representations

Figure 4. Project Name section.

In “Threshold” section, default value is 0.0. This means the detection algorithm will run as exact match. If you want to find similar patterns, this value may be entered as 0.1 through 1.0. As the threshold value rises, the similarity of the found instances descreases.

In “Overlap” section, if you click overlap checkbox the detection algorithm will search as overlapped instances.

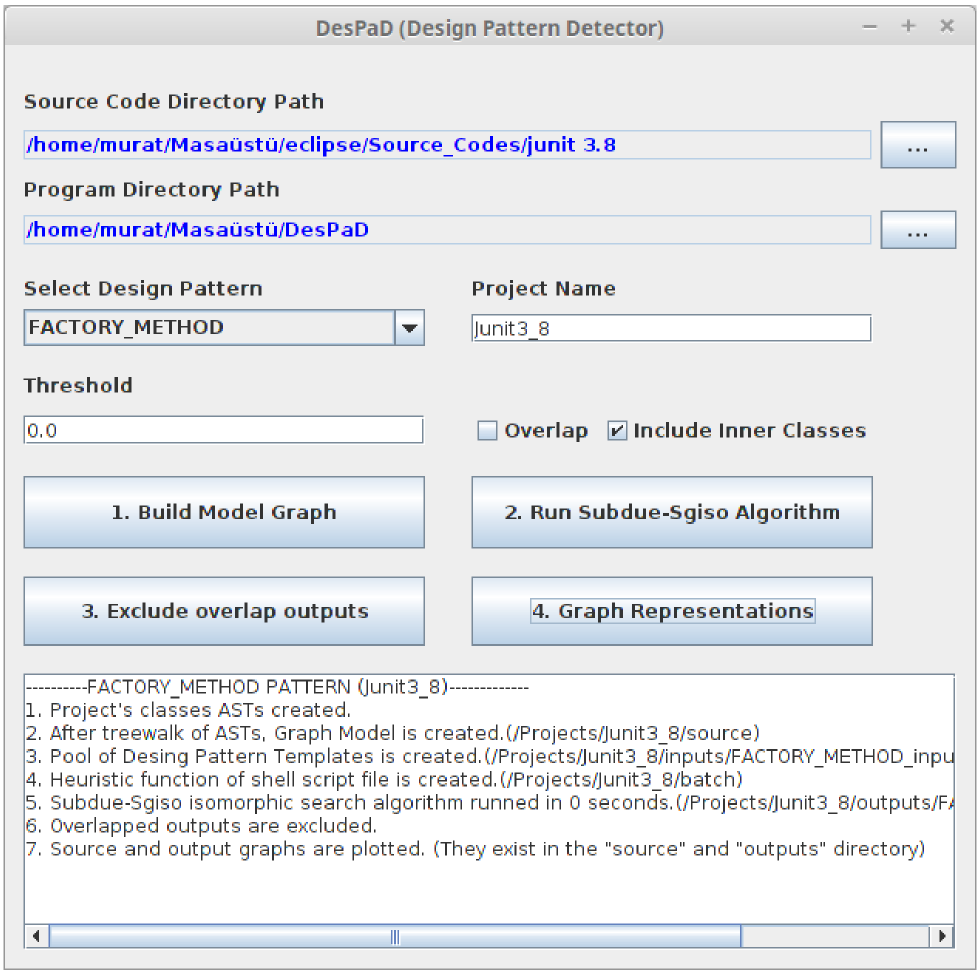
In “Include Inner Classes” section, defalut value is true. It is for using Java’s internal classes or not.

In “1. Build Model Graph” button, the parsing of the given source code and forming its model graph process is executed. (See Figure 5)

In “2. Run Subdue-Sgiso Algorithm” button, sub-graph mining algorithm is run. Isomorphic sub-graph mining search is applied in sgiso methodin Subdue. The output files are saved automatically in the “Projects” folder in DesPaD directory. (See Figure 5)

In “3. Exclude overlap outputs” button, the tool finds the overlap instances and excludes them automatically. (See Figure 5)

In “4. Graph Representations” button, the tool plots the model graph of the given source code and the detected design patterns’ outputs as graphs. It uses Graphviz open-sourced application for drawing. (See Figure 5) The graphs are in DesPaD, Projects, outputs and source directory. (See Figure 6)


DesPaD (Design Pattern Detector)
...
...
1. Build Model Graph
2. Run Subdue-Sgiso Algorithm
3. Exclude overlap outputs
4. Graph Representations

Source Code Directory Path: /home/murat/Masaüstü/eclipse/Source_Codes/junit 3.8

Program Directory Path: /home/murat/Masaüstü/DesPaD

Select Design Pattern: FACTORY_METHOD

Project Name: Junit3_8

Threshold: 0.0

☐ Overlap ☑ Include Inner Classes

```
----------FACTORY_METHOD PATTERN (Junit3_8)-------------
1. Project's classes ASTs created.
2. After treewalk of ASTs, Graph Model is created.(/Projects/Junit3_8/source)
3. Pool of Desing Pattern Templates is created.(/Projects/Junit3_8/inputs/FACTORY_METHOD_inpu
4. Heuristic function of shell script file is created.(/Projects/Junit3_8/batch)
5. Subdue-Sgiso isomorphic search algorithm runned in 0 seconds.(/Projects/Junit3_8/outputs/F
6. Overlapped outputs are excluded.
7. Source and output graphs are plotted. (They exist in the "source" and "outputs" directory)
```

Figure 5. The user interface after program runned.

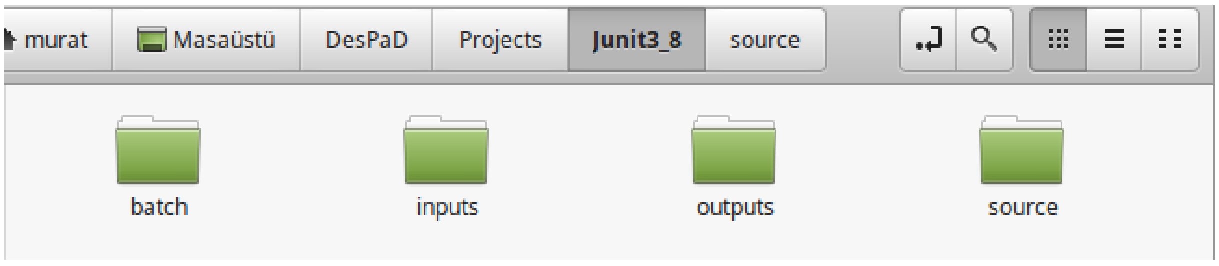


Figure 6. The directories created by DesPaD.