

ashah148 /
CSE-464-2025-ashah148

<> Code

Issues

Pull requests

Actions

Projects

Security

Insights



main

CSE-464-2025-ashah148 / README.md



ashah148 Create README.md

8edb1e9 · 3 minutes ago



87 lines (74 loc) · 2.22 KB

Preview

Code

Blame

Raw



Graph Processing Project

Overview

This project implements a directed graph using JGraphT and Graphviz. It supports:

- Parsing a DOT file to create a graph.
- Adding nodes and edges dynamically.
- Outputting the graph to a DOT file and a PNG image.
- Unit tests for all features.

Installation & Dependencies

Required Libraries:

- JGraphT
- Graphviz-Java
- JUnit 5
- Apache Maven

Ensure you have **Maven** installed. If not, install it using:

```
sudo apt install maven # Linux
brew install maven     # macOS
choco install maven    # Windows
```



Running the Program

1. Compile the Project

```
mvn clean package
```



2. Run the Program

```
mvn exec:java -Dexec.mainClass="com.graph.GraphHandler" -Dexec.args='
```



This will:

- Parse test.dot
- Generate output.dot
- Generate graph.png

Running Unit Tests

To run unit tests:

```
mvn test
```



Ensure all tests pass successfully.

Example Input (test.dot)

```
digraph G {  
    A -> B;  
    B -> C;  
    C -> A;  
}
```



Expected Output (expected.txt)

```
digraph G {  
    A;  
    B;  
    C;  
    A -> B;  
    B -> C;
```



```
C -> A;  
}
```

Generated Output Files

- `output.dot` (Graph representation in DOT format)
- `graph.png` (Graph visualization as an image)

Screenshots:

- Graph and output

The screenshot displays an IDE interface for a project named 'CSE-464-2025-ashah148'. The left sidebar shows the project structure with files like `graph.png`, `output.dot`, `pom.xml`, `README`, and `test.dot`. The main editor area is split into two panes. The top pane shows a graph visualization with nodes A, B, C, X, and Y. Nodes A, B, and C are arranged vertically on the left, with directed edges from A to B and B to C. Nodes X and Y are arranged vertically on the right, with directed edges from X to Y and Y to X. The bottom pane shows the DOT output file `output.dot` with the following content:

```
1 digraph G {
2   A;
3   B;
4   C;
5   X;
6   Y;
7   A -> B;
8   B -> C;
9   C -> A;
10  X -> Y;
11  Y -> X;
12 }
```

The terminal at the bottom shows the output of the build process, indicating that all tests completed successfully.

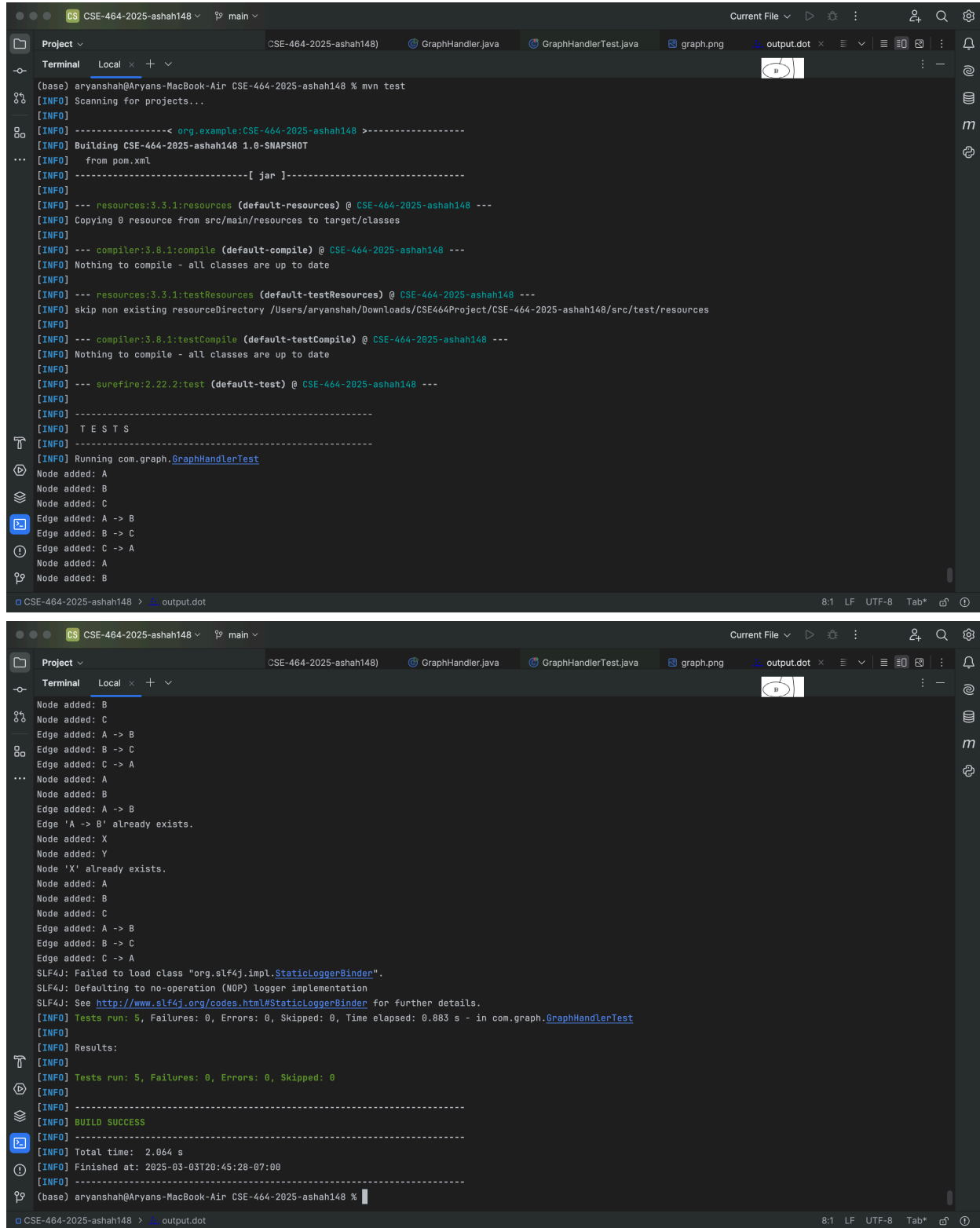
- Graph and output from unit tests

The screenshot displays an IDE interface for a project named 'CSE-464-2025-ashah148'. The left sidebar shows the project structure with files like .gitignore, expected.txt, graph.png, output.dot, pom.xml, README, and test.dot. The main editor area shows the 'graph.png' file, which is a directed graph with three nodes: A, B, and C. Node A is at the top, B is in the middle, and C is at the bottom. There are directed edges from A to B, B to C, and C to A, forming a cycle. The terminal at the bottom shows the output of running tests, indicating that all tests passed successfully. The status bar at the bottom indicates the file encoding is UTF-8 and the tab is active.

```
graph TD; A((A)) --> B((B)); B --> C((C)); C --> A;
```

```
digraph G {
  A;
  B;
  C;
  A --> B;
  B --> C;
  C --> A;
}
```

- Successfully passing the test cases



```
(base) aryanshah@Aryans-MacBook-Air CSE-464-2025-ashah148 % mvn test
[INFO] Scanning for projects...
[INFO]
[INFO] -----< org.example:CSE-464-2025-ashah148 >-----
[INFO] Building CSE-464-2025-ashah148 1.0-SNAPSHOT
[INFO] from pom.xml
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- resources:3.3.1:resources (default-resources) @ CSE-464-2025-ashah148 ---
[INFO] Copying 0 resource from src/main/resources to target/classes
[INFO]
[INFO] --- compiler:3.8.1:compile (default-compile) @ CSE-464-2025-ashah148 ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- resources:3.3.1:testResources (default-testResources) @ CSE-464-2025-ashah148 ---
[INFO] skip non existing resourceDirectory /Users/aryanshah/Downloads/CSE464Project/CSE-464-2025-ashah148/src/test/resources
[INFO]
[INFO] --- compiler:3.8.1:testCompile (default-testCompile) @ CSE-464-2025-ashah148 ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- surefire:2.22.2:test (default-test) @ CSE-464-2025-ashah148 ---
[INFO]
[INFO] -----
[INFO] T E S T S
[INFO] -----
[INFO] Running com.graph.GraphHandlerTest
Node added: A
Node added: B
Node added: C
Edge added: A -> B
Edge added: B -> C
Edge added: C -> A
Node added: A
Node added: B
CSE-464-2025-ashah148 > .\output.dot

CSE-464-2025-ashah148 % mvn test
Node added: B
Node added: C
Edge added: A -> B
Edge added: B -> C
Edge added: C -> A
Node added: A
Node added: B
Edge added: A -> B
Edge 'A -> B' already exists.
Node added: X
Node added: Y
Node 'X' already exists.
Node added: A
Node added: B
Node added: C
Edge added: A -> B
Edge added: B -> C
Edge added: C -> A
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
[INFO] Tests run: 5, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.883 s - in com.graph.GraphHandlerTest
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 5, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 2.064 s
[INFO] Finished at: 2025-03-03T20:45:28-07:00
[INFO]
(base) aryanshah@Aryans-MacBook-Air CSE-464-2025-ashah148 %
```

Author: Aryan Shah Course: CSE-464 Submission Date: 03/03/2025