Group U5: REFACTORING

COURSE: SOEN 6441

INSTRUCTOR: AMIN RANJ BAR

TEAM MEMBERS:

● Abhishek Handa (40231719)

● Rajat Sharma (40196467)

● Harman Singh Jolly (40204947)

● Amanpreet Singh (40221947)

● Anurag Teckchandani (40263724)

Potential Refactoring Targets:

1. Take Hardcoded Strings in separate classes.

2. Introduce the State Pattern into the Map editor.

3. Apply the State Pattern to the Startup, Issue, and Order phases of gameplay.

4. Incorporate command syntax validation.

5. Utilize the command pattern for order processing.

6. Transfer the logic from the model to the controller in the "IssueOrder" function.

7. Adjust certain naming conventions for improved clarity.

8. Integrate Exception Handling for scenarios such as adding a country without a continent, missing information, and handling typos when adding neighbors.

9. Create additional test cases for the existing logic.

10. Revise the continent check in Map Validation.

11. Add Javadoc comments for private data members.

12. Implement a restriction that prevents the game from proceeding with fewer than two players.

13. Define a constant for the file path.

14. Implement the Observer pattern for console log updates.

15. Alter the format for saving the map to match the domination map format.

Actual Refactoring Targets:

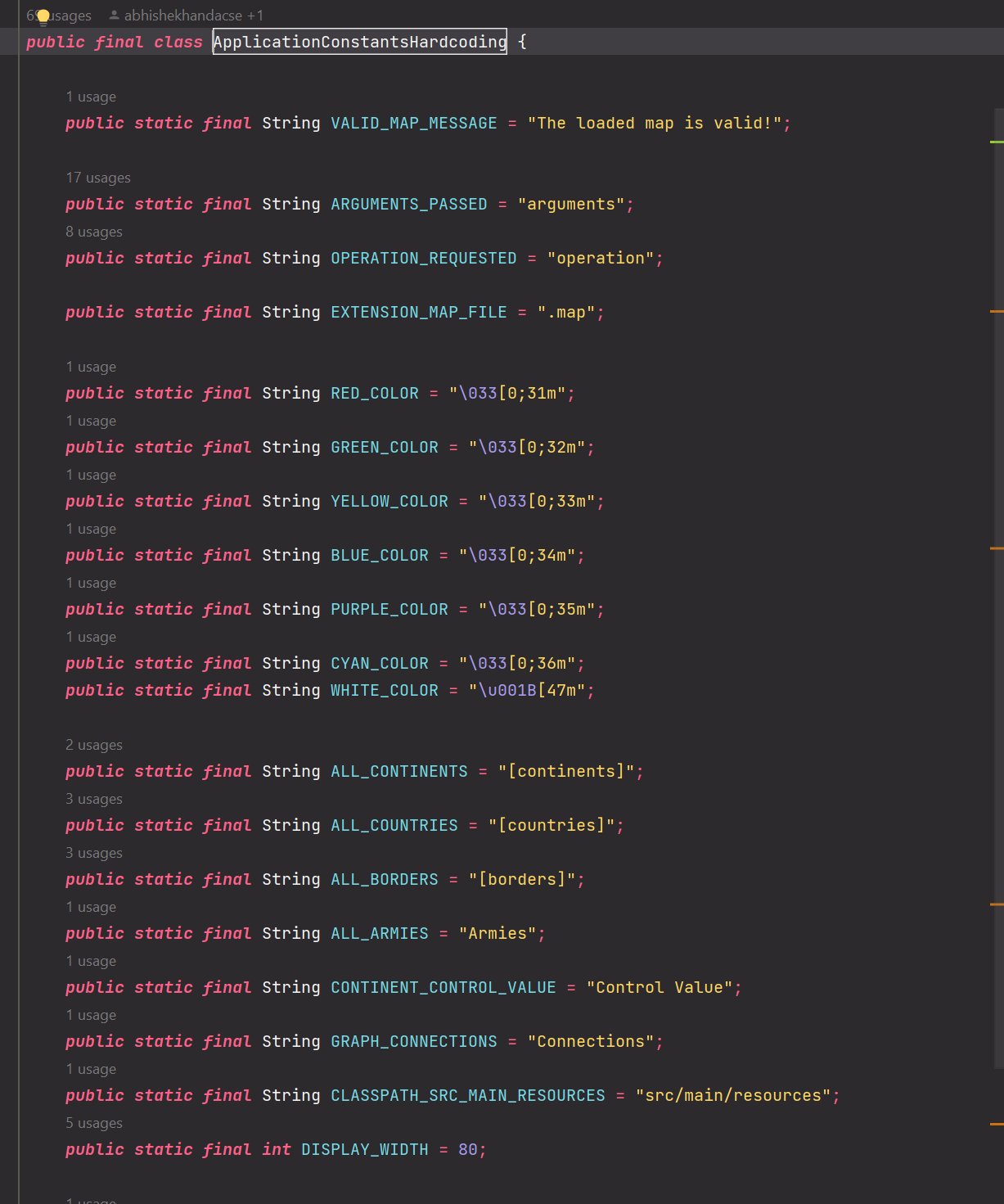
The list of refactoring taken for the above mentioned target list was mainly chosen because of the new requirements established in build 2 and on the greatest pain points and in-consistencies encountered during the development of the first build.

1. Hardcoded strings taken to a separate class called ApplicationConstantsHardcoding. Earlier the hardcodings were in each and every class.

Before Refactoring

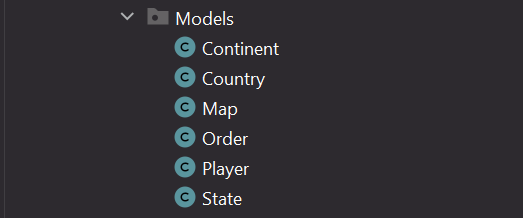


After Refactoring



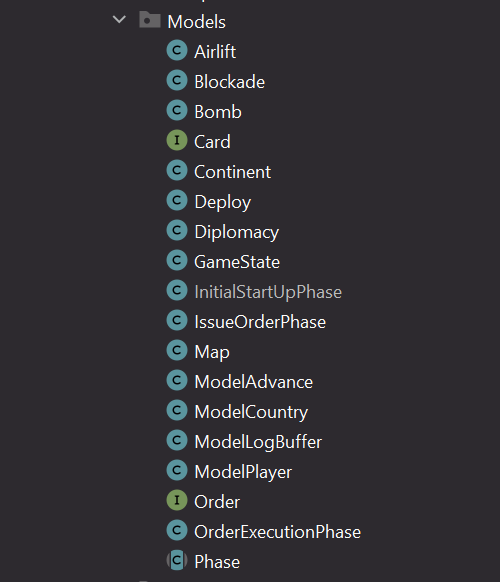
1. Implement State pattern for Phase Change:

Before Refactoring



After Refactoring

State pattern implemented and as asked in the requirements the separate classes created for InitialStartupPhase, IssueOrderPhase and OrderExecutionPhase.



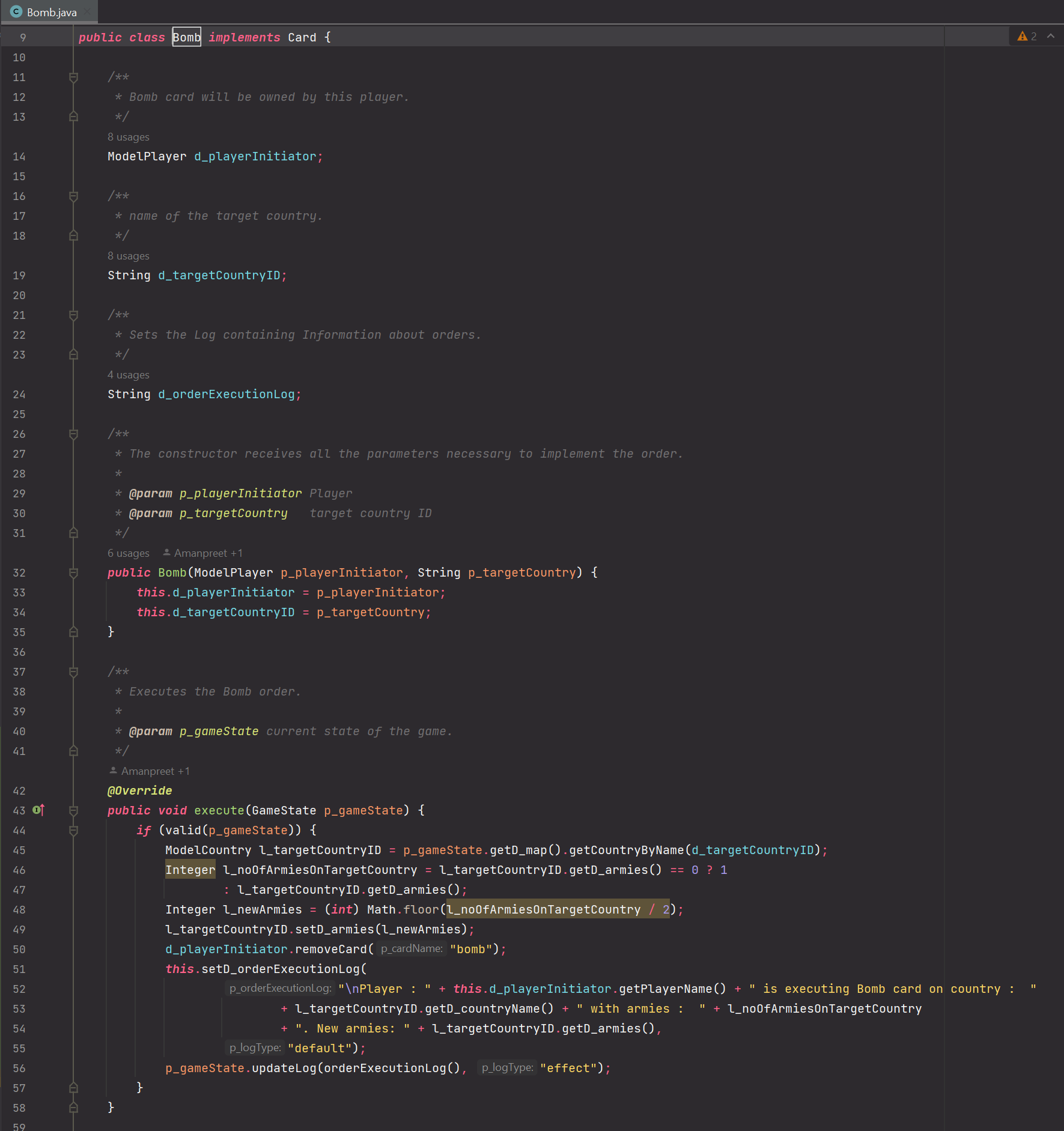
\*\* For detailed class level description, refer the codebase build2 snippet attached.

1. Implement command pattern for processing of orders

Before refactoring

In Build1, the processing of deploy order did not exactly follow the command pattern, there was a common single function execute that had the complete logic. In build 2, all of the Commands -Deploy, Advance, Bomb and Blockade, Airlife implements chain of commands and hence command pattern.

After refactoring

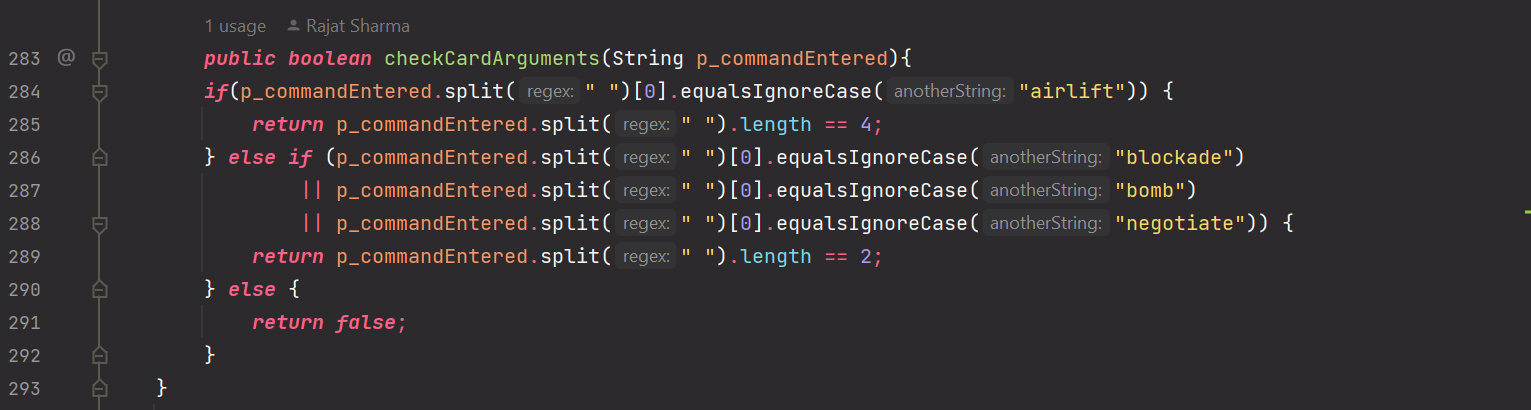


1. Implement Command syntax validation:

Before Refactoring:

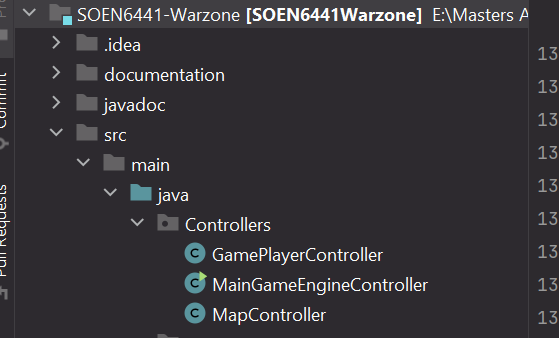
In build 1, the validation for commands was not implemented strictly. In Build 2, we explicitly added the function ValidateCommand to check the validation for all the commands.

After Refactoring:

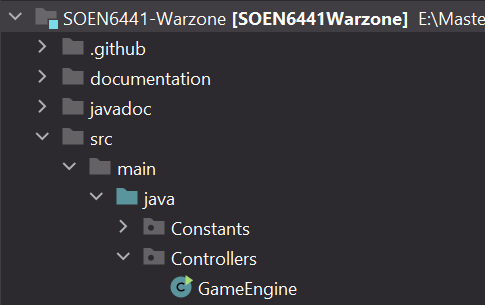


1. Expose only a single Controller rather than having redundant controllers in build 1

Before Refactoring:



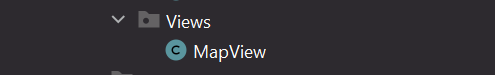
After Refactoring:



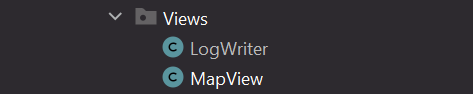
\*\* For detailed class level changes, refer to the code attached.

1. Make separate view for Log Writing to Console

Before refactoring:



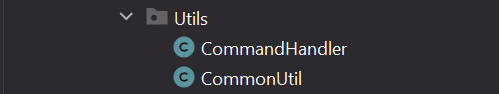
After refactoring:



\*\* For detailed class level changes, refer to the code attached.

1. Introduced LogExceptionHandler as Utility class for logging correctly.

Before refactoring:

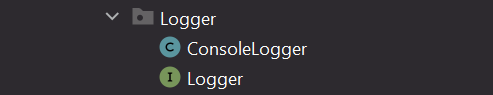


After refactoring:



1. Removed Logger package

Before refactoring:



After refactoring:

No Logger package

