



Risk Game (Build-1)

Coding Standards

Advanced Programming Practices

SOEN 6441
Fall-2019

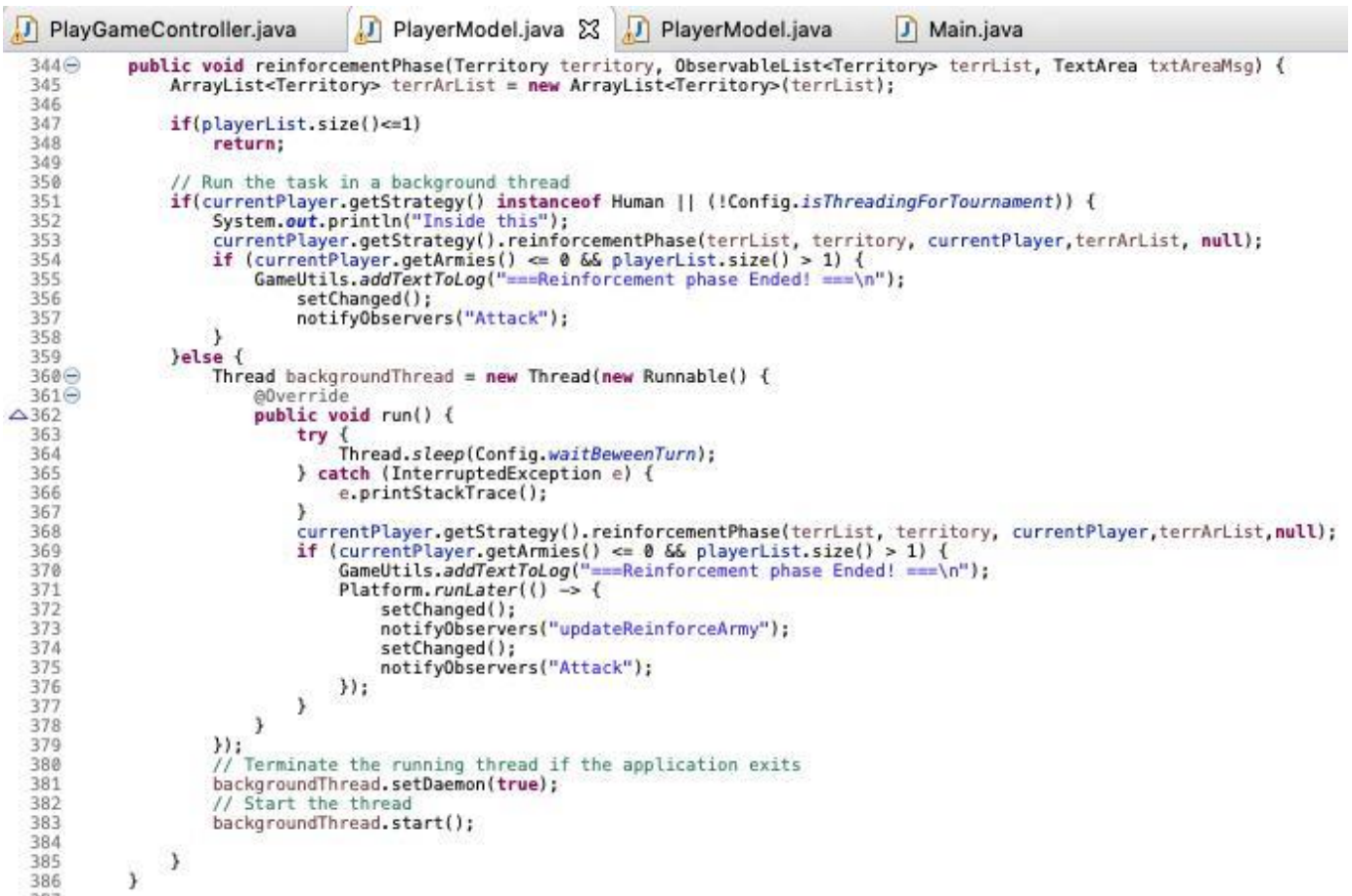
Team: Group_U_J

Mehul Prajapati	(40076930)
Komal Panchal	(40130791)
Maryam Giahi	(40016260)
Mahmoudreza Entezami	(40058782)
Parth Patel	(40081116)
Shubham Ranadive	(40083991)

Coding Conventions

1. Code Layout

- To minimize the length and maximize the readability of the code, the curly braces are appended to the statement preceding it.
- Blank lines are added to separate sections or components to increase the readability.



```
PlayGameController.java | PlayerModel.java | PlayerModel.java | Main.java
344 public void reinforcementPhase(Territory territory, ObservableList<Territory> terrList, TextArea txtAreaMsg) {
345     ArrayList<Territory> terrArList = new ArrayList<Territory>(terrList);
346
347     if(playerList.size()<=1)
348         return;
349
350     // Run the task in a background thread
351     if(currentPlayer.getStrategy() instanceof Human || (!Config.isThreadingForTournament)) {
352         System.out.println("Inside this");
353         currentPlayer.getStrategy().reinforcementPhase(terrList, territory, currentPlayer,terrArList, null);
354         if (currentPlayer.getArmies() <= 0 && playerList.size() > 1) {
355             GameUtils.addTextToLog("===Reinforcement phase Ended! ===\n");
356             setChanged();
357             notifyObservers("Attack");
358         }
359     }else {
360         Thread backgroundThread = new Thread(new Runnable() {
361             @Override
362             public void run() {
363                 try {
364                     Thread.sleep(Config.waitBetweenTurn);
365                 } catch (InterruptedException e) {
366                     e.printStackTrace();
367                 }
368                 currentPlayer.getStrategy().reinforcementPhase(terrList, territory, currentPlayer,terrArList,null);
369                 if (currentPlayer.getArmies() <= 0 && playerList.size() > 1) {
370                     GameUtils.addTextToLog("===Reinforcement phase Ended! ===\n");
371                     Platform.runLater(() -> {
372                         setChanged();
373                         notifyObservers("updateReinforceArmy");
374                         setChanged();
375                         notifyObservers("Attack");
376                     });
377                 }
378             }
379         });
380         // Terminate the running thread if the application exits
381         backgroundThread.setDaemon(true);
382         // Start the thread
383         backgroundThread.start();
384     }
385 }
386 }
```

2. Naming Conventions

- Class name:
 - The class names are written as the first letter as the uppercase of each word.
 - Example:

```
22 */
23 public class MapReader {
24
25     // Map class to return, once map is processed successfully.
26     private Hmap map;
```

- Function name:
 - Each function name is started with the lower case, followed by the uppercase of first letter of other words to bifurcate between the words.
 - Example:

```
21 /**
22  * Removes continent from the map.
23  * @param map Current map object.
24  * @param name Name of the continent.
25  * @return true if continent got removed successfully, otherwise false
26  */
27 public static boolean removeContinent(Hmap map, String continentName) {
28
```

- Variable name:
 - Constant variables is started with **all uppercase** with words separated by underscores ("_").
 - Example:

```
9 public static final String MAP_COMMAND_EDIT_CONTINENT = "editcontinent";
10 public static final String MAP_COMMAND_EDIT_COUNTRY = "editcountry";
11 public static final String MAP_COMMAND_EDIT_NEIGHBOR = "editneighbor";
12
13 public static final String MAP_COMMAND_SHOWMAP = "showmap";
14 public static final String MAP_COMMAND_SAVEMAP = "savemap";
```

- Folder and Package names:
 - All folder and package names are written in lowercase.
 - Example:

```
soen-6441-risk-game [soen-6441-risk-game master]
├── src/main
│   ├── com.commandparser
│   └── com.config
│       ├── CardType.java
│       ├── Commands.java
│       ├── Config.java
│       └── GameState.java
```

3. Comments

- Commenting is done as per conventions for Java Doc.
- Each class declaration precedes by a comment explaining what the class is for.
- Each method or function have comments explaining what it does, as well as what is the purpose of parameters and return type description if the method's return is non-void.

```
6+ import java.io.File;
15
16- /**
17  * @author Komal
18  * @author Mehul
19  * This class is responsible to write the map file when user creates the map.
20  *
21  */
22 public class MapWriter {
23
24-     /**
25      * This method processes the map by calling three different methods and makes a
26      * string to be written in the map file.
27      *
28      * @param map object of the map which is being processed
29      * @return String to be written in the map file
30      */
31-     private String parseHmapAndGetString(Hmap map) {
```

- “commented out” code:

```
35         String content = parseMapAndReturnString(map);
36         //String country = parseMapAndReturnString(country);
37         FileWriter fileWriter = new FileWriter(file, false);
38         fileWriter.write(content);
39         fileWriter.close();
40     }
```

4. Indentation

- Code is indented according to its nesting level to improve code readability. Indentation of the body of the function is done with respect to its header. Similarly, for the for, while, switch, if and other statements, it is done with respect to its first line.

```
107         for (Continent continent : map.getContinents()) {
108             List<Country> countriesList = continent.getCountries();
109             if (countriesList != null) {
110                 for (Country country : countriesList) {
111                     countryData.append(country.getName() + "," + country.getxCoordinate() + ","
112                                     + country.getyCoordinate() + "," + country.getBelongToContinent().getName());
113                     for (Country adjacentCountries : country.getAdjacentCountries()) {
114                         countryData.append(",");
115                         countryData.append(adjacentCountries.getName());
116                     }
117                     countryData.append("\n");
118                 }
119                 countryData.append("\n");
120             }
121             return countryData;
122         }
123     }
```

References:

1. <https://www.geeksforgeeks.org/java-naming-conventions/>
2. <https://google.github.io/styleguide/javaguide.html#s6.2-caught-exception>