Coding Conventions

Code Layout Convention:

- Code is indented according to its nesting level.
- Amount of indentation = 4 space.
- The body of a function/method indented with respect to its function header
- The body of a for, while, or switch statement is indented with respect to its first line; and similarly for if statements and other nested structures.
- Blank lines are added to separate code components/sections.

Naming Convention:

- Folder and Package names are written in lowercase. I.e. controller
- File names are written in UpperCamelCase. I.e. Risk
- Class names are written in UpperCamelCase. I.e. CreateMapFile
- Method and Attribute names are written in lowerCamelCase.
  I.e. setPlayers(), isMapValid
- Constant names use all uppercase letters, with each word separated from the next by a single underscore.
  I.e. private static final Logger LOGGER = Logger.getLogger(InitializeData.class.getName());
- Parameter names are written in lowerCamelCase

```java
/**
 * This constructor is used to set the data members of class.
 * @param filePath path of map File.
 * @param playerCount Number of Players in game.
 * @param armies Number of armies per Player.
 * @param players model object of Players.
 */
public InitializeData(String filePath, int playerCount, int armies, Players players) {
    this.filePath = filePath;
    this.playerCount = playerCount;
    this.armies = armies;
    this.players = players;
}
```

- Local variable names are written in lowerCamelCase.

```
public boolean generateBoardData() {
    continentData = new StringBuilder();
    territoryData = new StringBuilder();
    continentObject = new Continent();
    territoryObject = new Territory();
    territoryObject.addCardValue("Infantry", 1);
    territoryObject.addCardValue("Cavalry", 5);
    territoryObject.addCardValue("Artillery", 10);
    territoryObject.addCardValue("Wild Card", 0);
    ArrayList<String> tempCardName = new ArrayList<>();
    tempCardName.add("Infantry");
    tempCardName.add("Cavalry");
    tempCardName.add("Artillery");
    int cardNo = 0;
```

Commenting Convention:

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

```
/**
 * This class is used to invoke Initial Reinforcement process if data of map file is valid.
 *
 */
public class InitializeData extends GamePanels {
    String filePath;
    int playerCount;
    int armies;
    Continent continent;
    Territory territory;
    Players players;
    private static final Logger LOGGER = Logger.getLogger(InitializeData.class.getName());

    /**
     * This constructor is used to set the data members of class.
     * @param filePath path of map File.
     * @param playerCount Number of Players in game.
     * @param armies Number of armies per Player.
     * @param players model object of Players.
     */
    public InitializeData(String filePath, int playerCount, int armies, Players players) {
        this.filePath = filePath;
        this.playerCount = playerCount;
        this.armies = armies;
        this.players = players;
    }
```