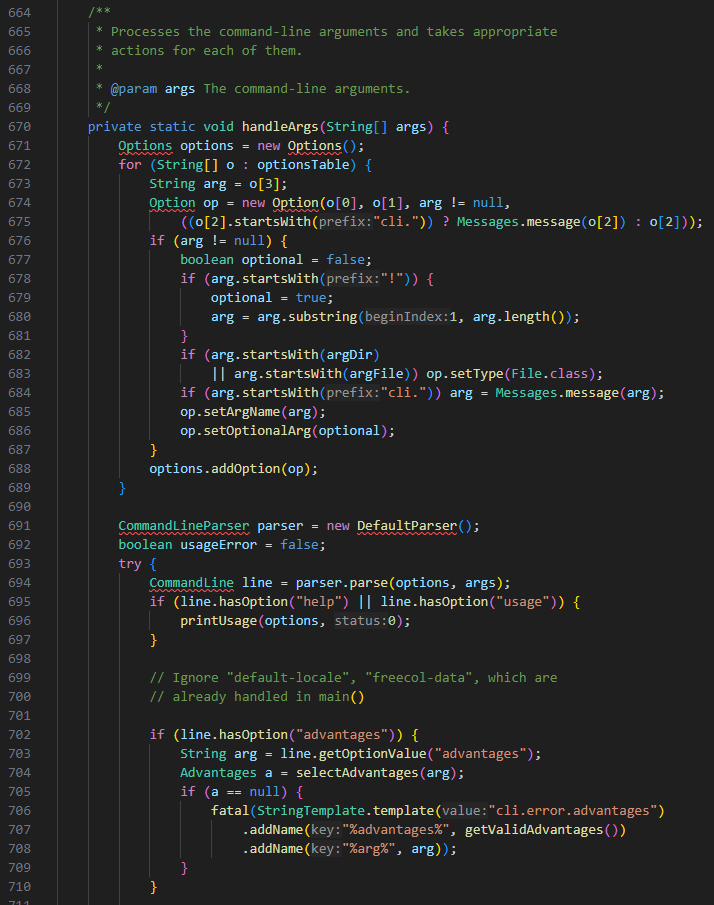
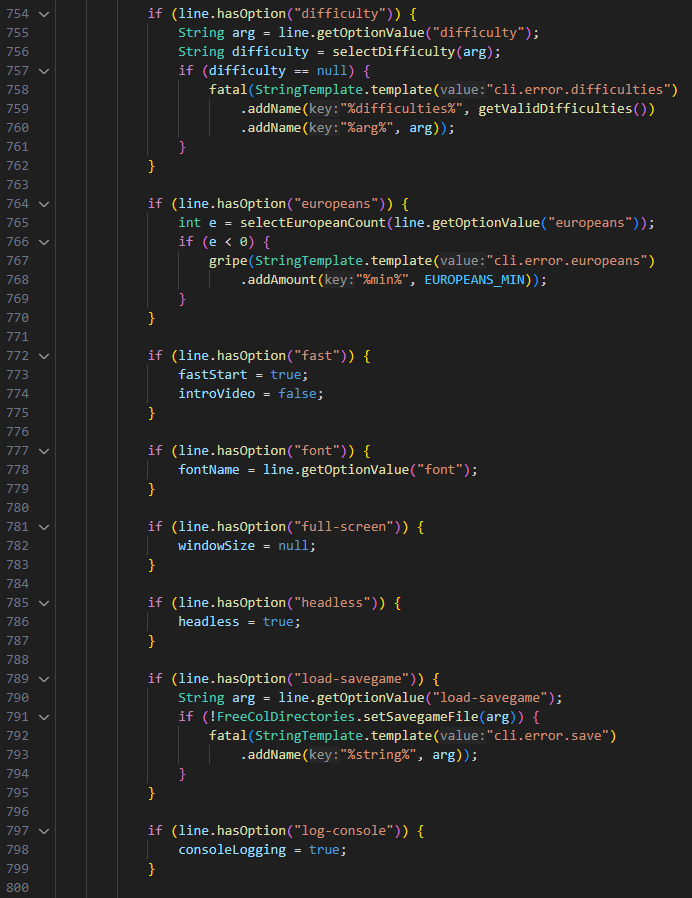
José Trigueiro 58119

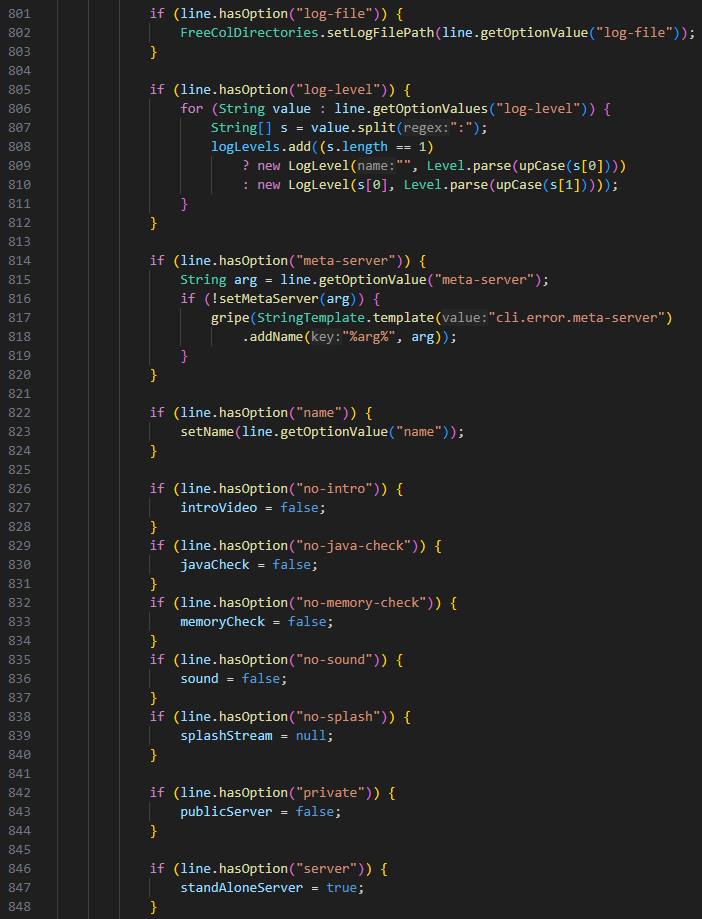
**Code Smells:**

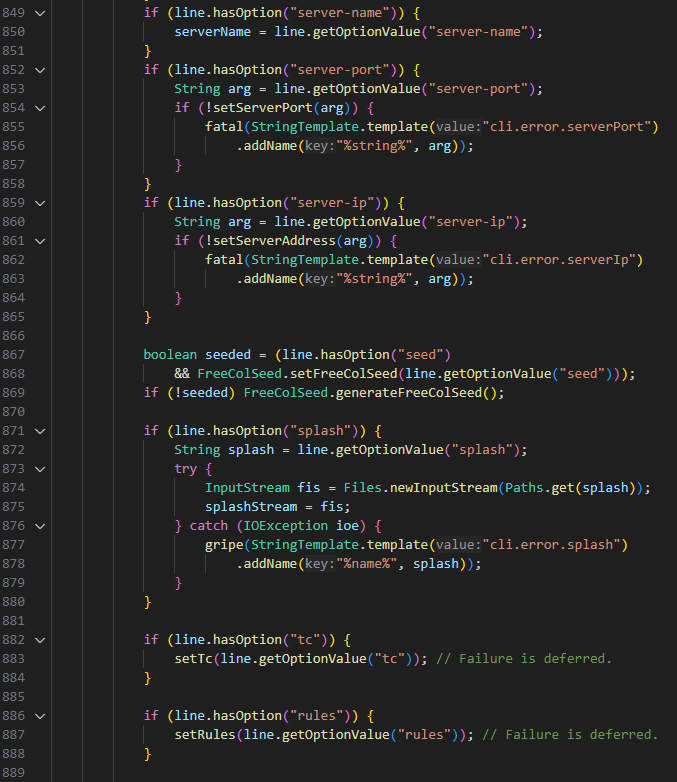
1. **Long Method - “handleArgs()”** in **(src/net/sf/freecol/tools/FreeCol.java)**The method is 273 lines long, to mitigate this we could extract chunks of code into separate methods. Each of these methods should do one thing and do it well. For example, have separate methods for handling each command line option like “handleAdvantagesOption”, “handleCheckSavegameOption”, “handleClientOptions”, etc and have generic method to handle them all, the use of polymorphism could be used. Also the string literals(like “advantages”, “check-savegame”, “clientOptions”, etc) that are used to check the command line options, could be replaced with constants or enumerations.



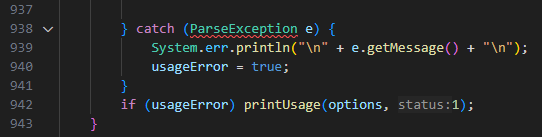


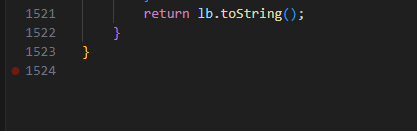
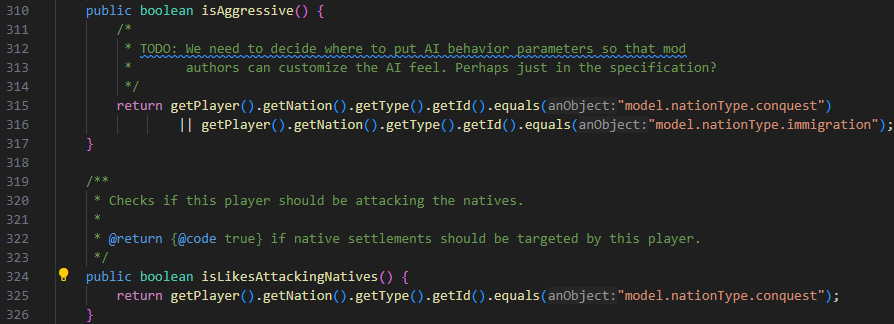


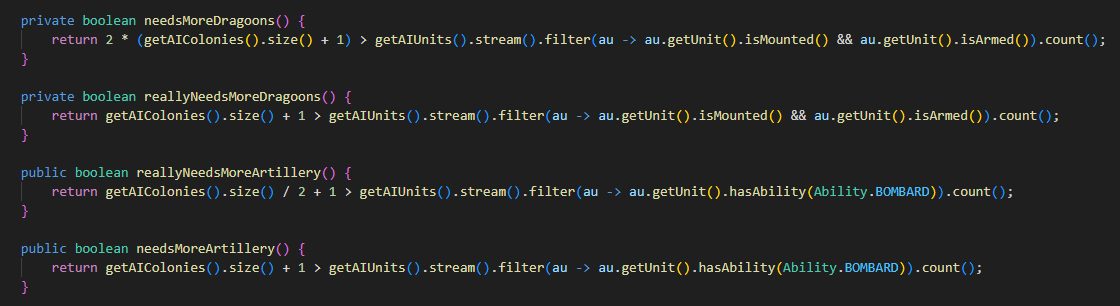








1. **Long Class- “ColonyPlan.java”** in **(src/net/sf/freecol/server/ai/ColonyPlan.java)**The class is 1523 lines long! To solve this problem each responsibility should have its own class and only after be called in this class that manages the colony plan.  
   ****
2. **Message Chains - “isAggressive()” & “isLikesAttackingNatives()” & “needsMoreDragoons()” & “reallyNeedsMoreDragoons()” & “reallyNeedsMoreArtillery()” & “needsMoreArtillery()”** in **(src/net/sf/freecol/server/ai/EuropeanAiPlayer.java)**These methods have really long message chains to get an object back due to rigidity and complexity in the design. There are several ways to solve this code smell. We could hide all the calls under a single method or reduce coupling between objects by moving behavior to the object that has the data or instead of querying objects for data and making decisions based on that data, instruct objects to perform actions.  
    ****

****