

Code Smells

Code Smell 1 - Long Method

Existem métodos demasiado grandes que tornam o código difícil de entender. Esses métodos dificultam a manutenção e a leitura do código. Um exemplo deste code smell no código do jogofreeCol é na classe net/sf/freecol/client/control/ConnectController.java o método startSavedGame (este método tem quase 100 linhas de código, demonstra a complexidade gigante deste código).

```
361 public boolean startSavedGame(File file) {
362     final FreeColClient fcc = getFreeColClient();
363     final GUI gui = getGUI();
364     fcc.setMapEditor(false);
365
366     //...
367     final ClientOptions options = getClientOptions();
368     final boolean defaultSinglePlayer;
369     final boolean defaultPublicServer;
370     FreeColSavegameFile fis = null;
371     try {...} catch (FileNotFoundException fnfe) {...} catch (IOException io) {...}
372     options.merge(fis);
373     options.fixClientOptions();
374
375     List<String> values = null;
376     try {...} catch (Exception ex) {...}
377     if (values != null && values.size() == savedKeys.size()) {...} else {...}
378
379     // Reload the client options saved with this game.
380     final boolean publicServer = defaultPublicServer;
381     final boolean singlePlayer;
382     final String serverName;
383     final InetAddress address;
384     final int port;
385     final int sgo = options.getInteger(ClientOptions.SHOW_SAVEGAME_SETTINGS);
386     boolean show = sgo == ClientOptions.SHOW_SAVEGAME_SETTINGS_ALWAYS
387         || (!defaultSinglePlayer && sgo == ClientOptions.SHOW_SAVEGAME_SETTINGS_MULTIPLAYER);
388     if (show) {...} else {...}
389     Messages.loadActiveModMessageBundle(options.getActiveMods(),
390         FreeCol.getLocale());
391 }
```

Resolução Code Smell 1:

Para resolver este problema de métodos demasiado longos, eu recomendaria separá-los em métodos mais pequenos e mais específicos, ou seja, ter métodos mais pequenos em que cada um realize uma única tarefa bem definida. No exemplo que têm em cima, separava este método em dois mais pequenos, já que a parte inicial é para obter uma sugestão do nome do jogador, single/multiplayer ... e a parte final refere-se a carregar as opções do cliente guardadas neste jogo.

Code Smell 2 -Message Chain

Há várias chamadas de métodos encadeados numa única linha de código, como por exemplo na classe net/sf/freecol/server/generator/FreeColMapLoader.java no método loadMap na linha: tile.setType(game.getSpecification().getTileType(template.getType().getId()));(linha 95)

```
Tile tile = new Tile(game, type, null, x, y);

// import tile types
tile.setType(game.getSpecification().getTileType(template.getType().getId()));
tile.setMoveToEurope(template.getMoveToEurope());
if (highestLayer.compareTo(Layer.REGIONS) >= 0) {
```

Resolução Code Smell 2:

Para resolver este problema, eu optaria por separar as chamadas aos métodos que são feitas, guardando os resultados intermédios em variáveis.

Code Smell 3 -Large Class

Como podemos verificar, a classe `net/sf/freecol/client/control/InGameController.java` tem cerca de 5387 linhas de código, ou seja, é um sinal de que esta classe está a fazer demasiadas coisas tornando o código difícil de entender e ler.

Resolução Code Smell 3:

Começaria por dividir esta classe em classes mais pequenas em que cada uma teria uma responsabilidade específica dentro desta classe `InGameController`. Uma maneira de pensar poderia ser, criando diferentes classes para lidar com os diferentes tipos (jogadores, colónias, unidades, etc), isso iria reduzir bastante a complexidade da classe `InGameController` e ficaria mais fácil de gerenciar e entender.