**Patterns**

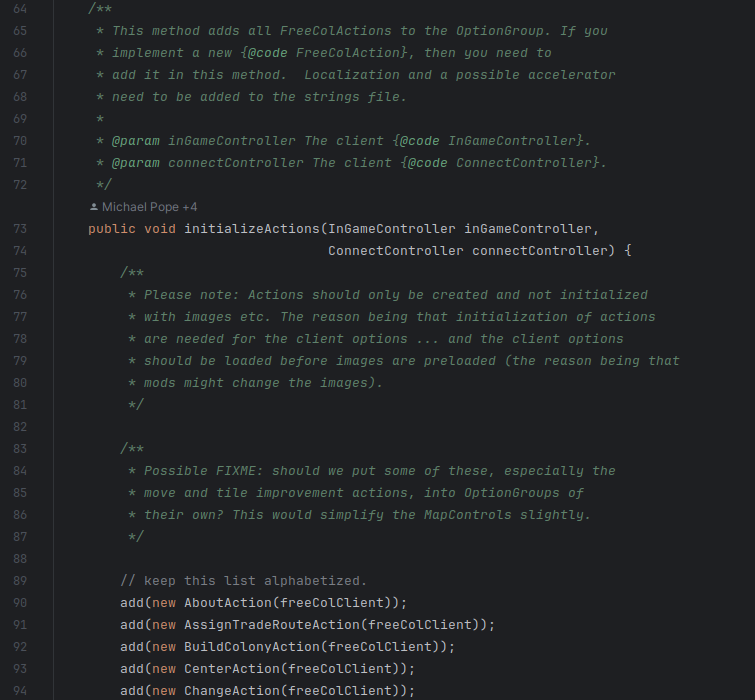
José Morgado 59457

1. **Command Pattern:**

The class ActionManager.java contained in the package src/net/sf/freecol/client/gui/action seems to exhibit the Command pattern, as it is designed to encapsulate requests as objects, allowing clients to parameterize them, queue them, and record their history (which, however, is unused in this method, although implemented). In the context of the ActionManager:

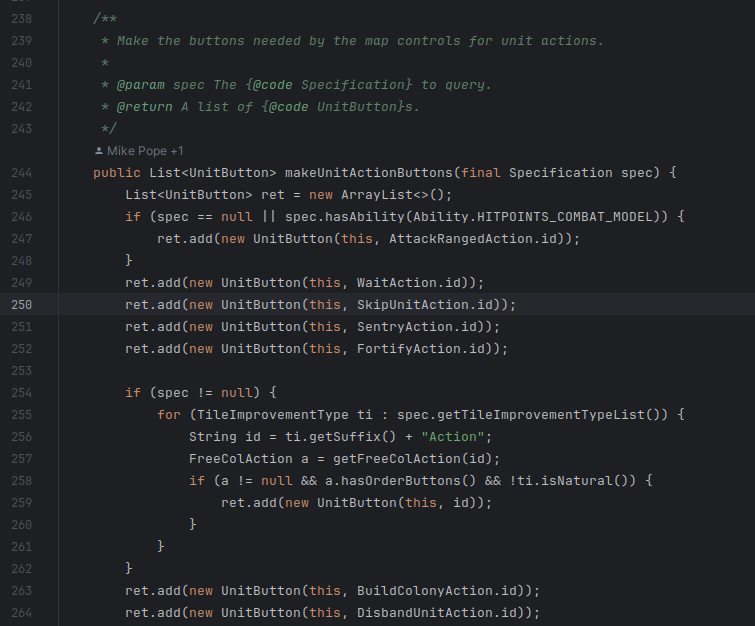
Actions are represented as FreeColAction objects, encapsulating specific requests or commands that can be executed within the game.

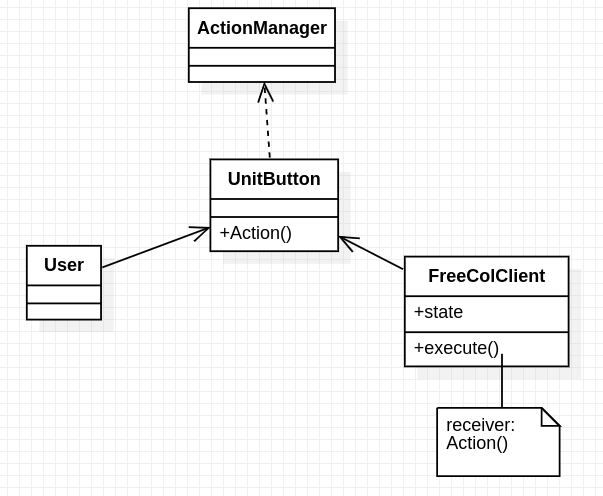
Fig. 1: Some implemented actions



These actions are then mapped to buttons, enabling the user to request actions without knowing the specific details of how they are executed or how the commands are processed.

Fig. 2 - Mapping of actions to buttons



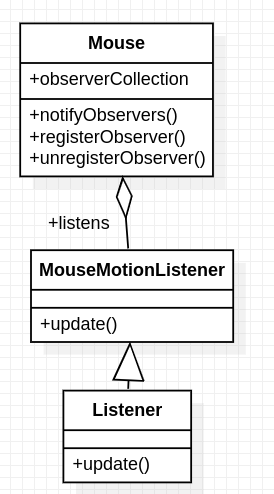


1. **Observer Pattern:**

The class FreeColMenuBar.java, located in the package src/net/sf/freecol/client/gui/menu, contains listener elements that exhibit the Observer design pattern. One of these elements is the MouseMotionListener, which observes user-controlled mouse movement and provides a way of decoupling between the subject (in this case, the menu bar) and the observers, allowing observers to be notified and respond to events independently.

Fig. 3 - Initialization of the MouseMotionListener



**

1. **Factory Method Pattern:**

The class AbstractUnit.java in src/net/sf/freecol/common/model seems to exhibit the Factory Method pattern, the existence of different constructors like AbstractUnit(), AbstractUnit(String id, String roleId, int number) and AbstractUnit(UnitType unitType, String roleId, int number) provides various ways to instantiate AbstractUnit objects with different parameters and initialization approaches.

This versatility in object creation points to a Factory Method Pattern, where multiple factory methods or constructors exist in a class to create instances of objects, allowing flexibility in how these objects are created and initialized based on different criteria or parameters.

Fig. 4 - Different constructors

