**Patterns**

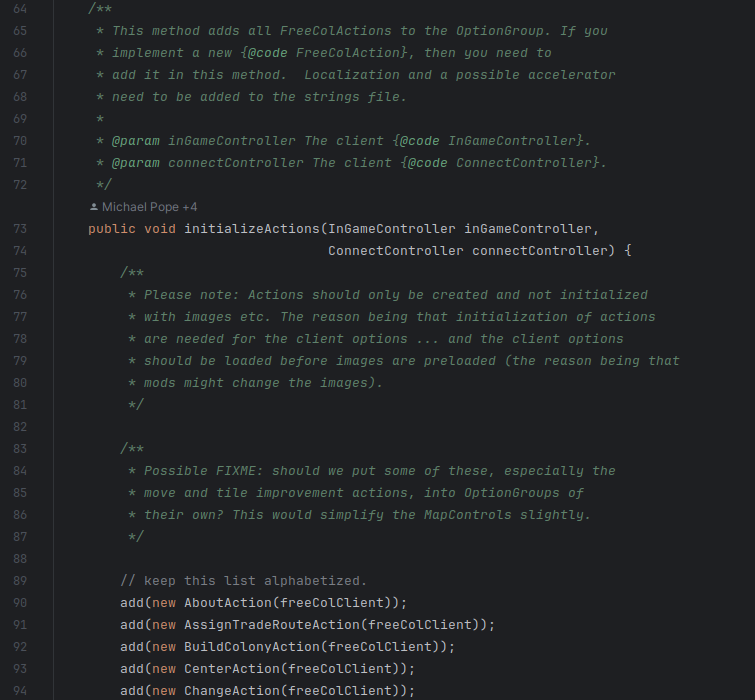
José Morgado 59457

1. **Command Pattern:**

A classe ActionManager.java contida na package src/net/sf/freecol/client/gui/action parece apresentar o padrão Command pattern, uma vez que este é projetado para encapsular solicitações como objetos, permitindo que os clientes as parametrizem, enfileirem-nas, registem o seu histórico (que está, no entanto, inutilizado neste método, embora esteja implementado). No contexto do ActionManager:

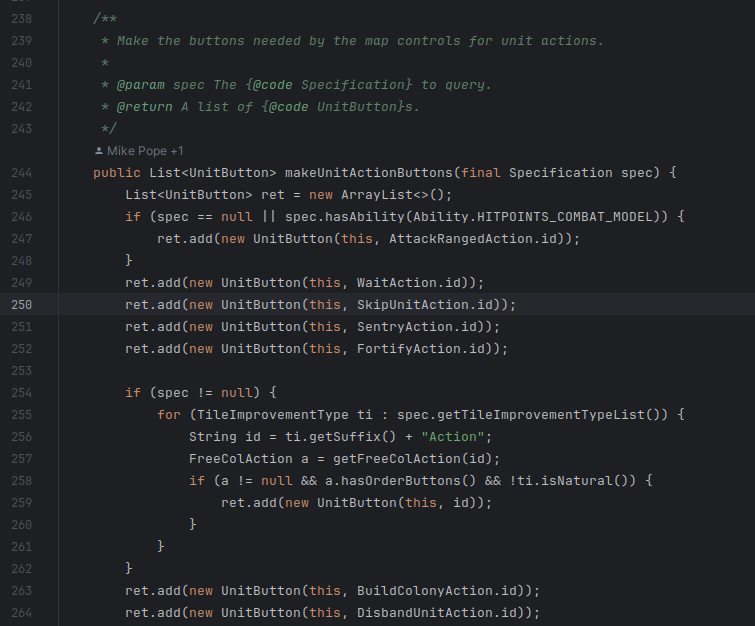
As ações são representadas como objetos FreeColAction, que encapsulam solicitações específicas ou comandos que podem ser executados no jogo.

Fig. 1: Algumas ações implementadas



Estas são depois mapeadas para botões, permitindo que o utilizador solicite ações sem conhecer os detalhes específicos de como estas são executadas ou como os comandos são processados.

Fig. 2 - Mapeamento de ações para botões



1. **Observer Pattern:**

A classe FreeColMenuBar.java contida no package src/net/sf/freecol/client/gui/menu contém elementos listeners que apresentam o Observer design pattern.

Um desses elementos é o MouseMotionListener listener que observa o movimento do rato controlado pelo utilizador e fornece uma forma de desacoplamento entre o sujeito (neste caso, a barra de menu) e os observadores, permitindo que os observadores sejam notificados e respondam a eventos de forma independente.

Fig. 3 - Inicialização do MouseMotionListener



1. **Singleton Pattern:**

Na classe *FreeColClient,* contida no package src/net/sf/freecol/client é inicializado o controlador principal usado em muitas outras classes no código, o FreeColClient.

O construtor é privado, o que impede outras classes de instanciar um objeto destes, é criada uma única instância e tem um ponto de acesso global, logo segue as normas de um pattern Singleton.

Fig.4 - Criação do controlador principal FreeColClient

