Instituto Superior de Engenharia de Coimbra

2024/2025

Licenciatura em Engenharia Informática

Programação Avançada

Trabalho Prático

Chess Game

Bruno Pinto, nº 2021129642

Diogo Ferreira, nº 2021129669

Índice

Introdução

Este trabalho prático tem como objetivo o desenvolvimento do jogo Xadrez em **Java** com o uso de **JavaFX** para o desenvolvimento da *UI*. De forma a facilitar o trabalho em grupo e a gestão de cada fase do trabalho foi usado o ***GitHub*** permitindo criar um repositório remoto onde foi possível partilhar e integrar facilmente as alterações realizadas por cada elemento do grupo, garantindo uma colaboração eficiente e organizada.

Além disso, ao longo do desenvolvimento foram aplicados vários ***Software Design Patterns*** com o objetivo de tornar a aplicação mais modular, reutilizável e de fácil manutenção. Entre os padrões utilizados destacam-se o *Singleton*, o *Observer/Observable* e o *Facade*.

Arquitetura do Sistema

O projeto foi estruturado segundo o princípio de separação de responsabilidades, dividindo-se em duas partes principais: ***UI*** (*User Interface*) e ***Model***. Esta divisão permitiu uma organização mais clara do código, facilitando a manutenção, os testes e futuras expansões.

Para efeitos de comunicação entre a UI e o modelo de dados foram criadas duas *Facades*: o ***ChessGameManager***, usado como *Facade* para a UI, e o ***ChessGame*** como *Facade* para o modelo de dados.

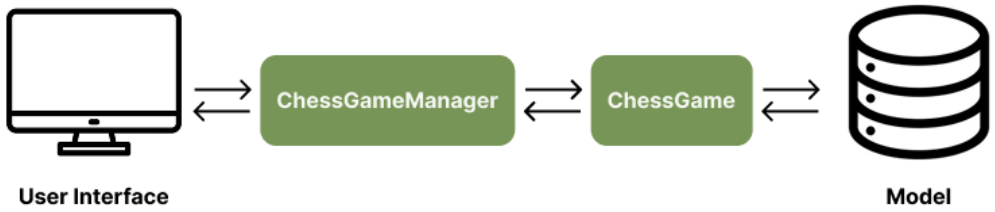

Figura 1 - Arquitetura geral

Model

Toda a lógica do Xadrez encontra-se centralizada, principalmente em três classes principais:

- ***ChessGame*:** classe responsável por gerir o estado do jogo e interagir com os dados de maneira a realizar as ações pretendidas, servindo como *Facade* para o modelo de dados.
- ***Board*:** classe responsável por gerir e interagir com as peças do jogo.
- ***Piece***: classe abstrata que representa um peça de xadrez definido a estrutura métodos base que cada peça deverá seguir.

Para que a aplicação seja modular e escalável, foram criadas algumas ferramentas, enumerações e classes que ajudam a separar responsabilidades e a facilitar a manutenção do código.

- ***PieceFactory*:** classe responsável por criar peças com base numa descrição textual (ex: KD1*, pH5), simplificando o processo de construção e reinicialização do tabuleiro.
- ***ChessSerialization*:** classe responsável por serializar e desserializar objetos da classe *ChessGame*, permitindo guardar e restaurar o estado do jogo em ficheiros.
- ***ColorType*:** enumeração que representa a cor de cada peça (*WHITE* ou *BLACK*)
- ***MoveType*:** enumeração que define os diferentes tipos de movimentos (*MOVE*, *TAKE*, *EVOLVE*, *ENPASSAT*), ajudando na validação e execução das jogadas.
- ***PieceType*:** enumeração que representa os tipos de peças existentes (*KING*, *QUEEN*, *ROOK*, etc.), essencial para a criação dinâmica de peças através da *PieceFactory*.

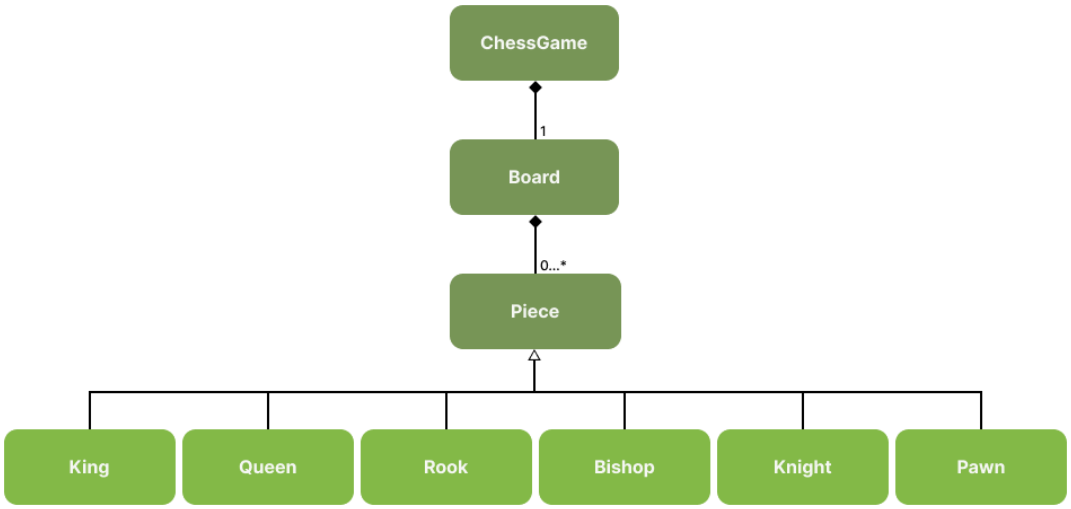

Figura 2 - Modelo de dados

Para o gerir os registos gerados pelo modelo de dados foi criada a classe ***ModelLog***. Esta classe segue o padrão Singleton, garantindo que apenas uma instância existe em toda a aplicação. O *ModelLog* permite o registo de novos *listeners*, a adição de novos *logs* (movimentos realizados, jogos criados, etc) como também a limpeza de todos os *logs* existentes.

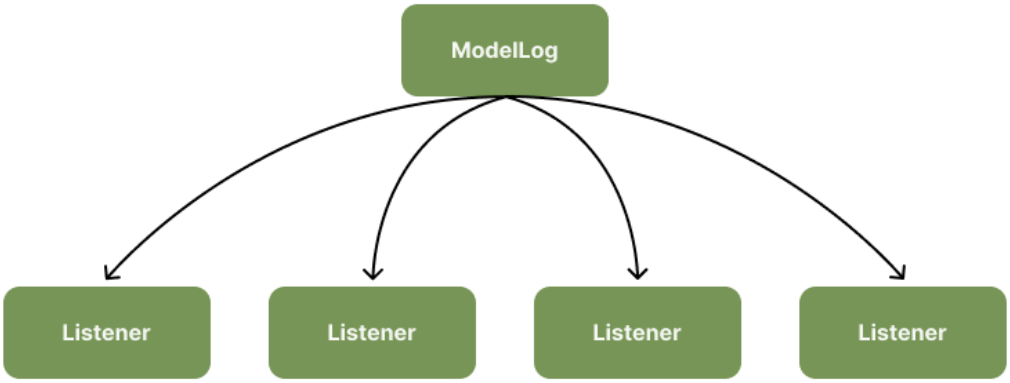

Figura 3 - ModelLog

User Interface

A interface gráfica foi desenvolvida com o uso de **JavaFX**, seguindo o modelo de **MVC@PA**. Para efeitos de abstração da lógica de jogo, foi criada uma *Facade* (***ChessGameManager***) que atua como intermediária entre a interface e a *Facade* do modelo, expondo apenas os métodos necessários para a interação com a UI. Para garantir uma comunicação reativa entre o modelo e a interface, foi utilizado o mecanismo de ***PropertyChangeSupport***, permitindo que a interface gráfica seja notificada automaticamente sempre que ocorrem alterações no estado do jogo, como movimentos, capturas ou trocas de estado.

Interface gráfica

Board

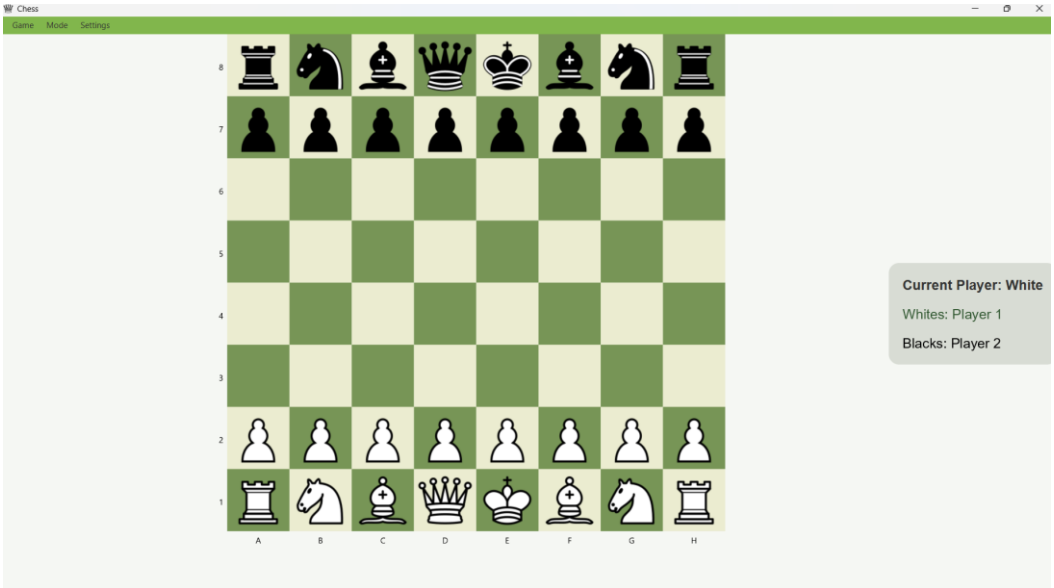

Figura 4 - Disposição inicial de um novo jogo de Xadrez.

A *Figura 4* representa a disposição inicial de um novo jogo de xadrez, no centro do ecrã está representado o tabuleiro e as peças de cada jogador. A ***SideBar*** representada à direita da Imagem indica o ***"Current Player"*** (o jogador atual) assim como os nomes de ambos os jogadores, a cor das peças que representa e ,quando existirem, mostra ainda por baixo de cada jogador as peças por eles "comidas".

No topo da *Figura 4* encontram -se as definições do jogo (***Settings***), os modos de jogo (***Mode***) e ainda as opções de jogo (***Game***).

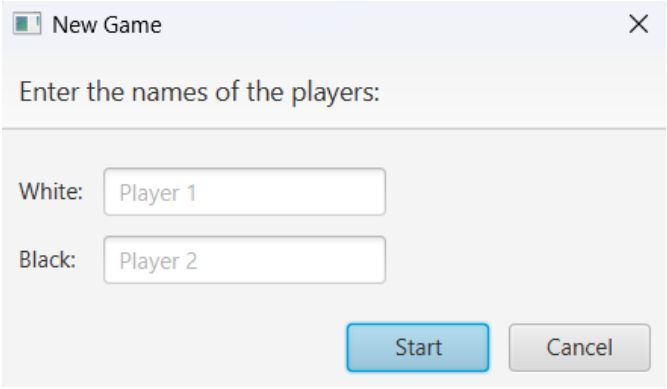

Figura 5 - Introduzir nomes dos jogadores

A *Figura 5* aparece quando abrimos um jogo ou quando é criado um novo jogo, nesta janela os jogadores são obrigados a inserir os nomes com os quais pretendem jogar.

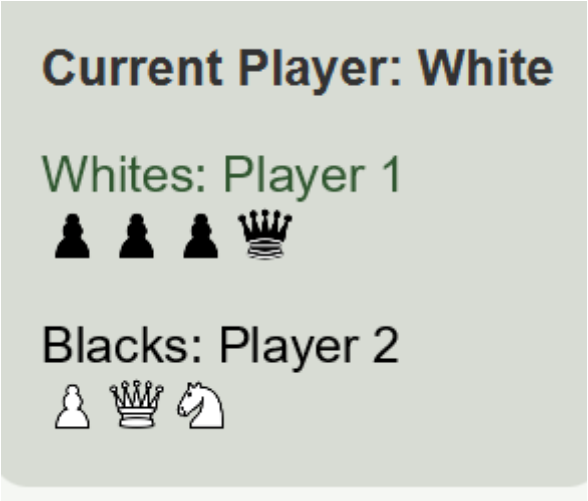

Figura 6 - Peças capturadas por cada um dos jogadores

Na *Figura 6* estão representadas as peças capturadas por cada um dos jogadores.

Header

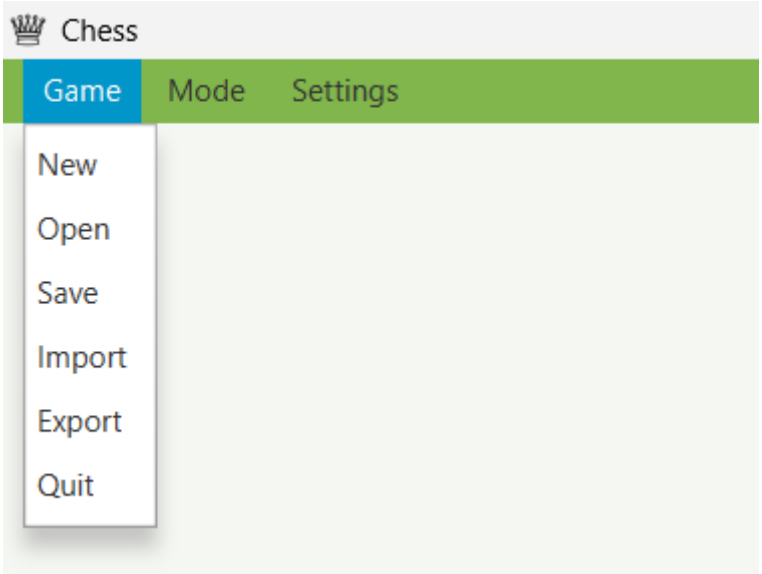

Figura 7 - Menu "Game"

Na *Figura 7* está "expandido" o menu ***Game*** que contém as opções ***New*** (começar novo jogo), ***Open*** e ***Save*** (permite abrir um jogo já começado e guardar o jogo atual através da Serialização), ***Import*** e ***Export*** (permite importar um jogo existente e guardar o jogo atual através de uma formatação textual) e o ***Quit*** que nos permite sair da aplicação.

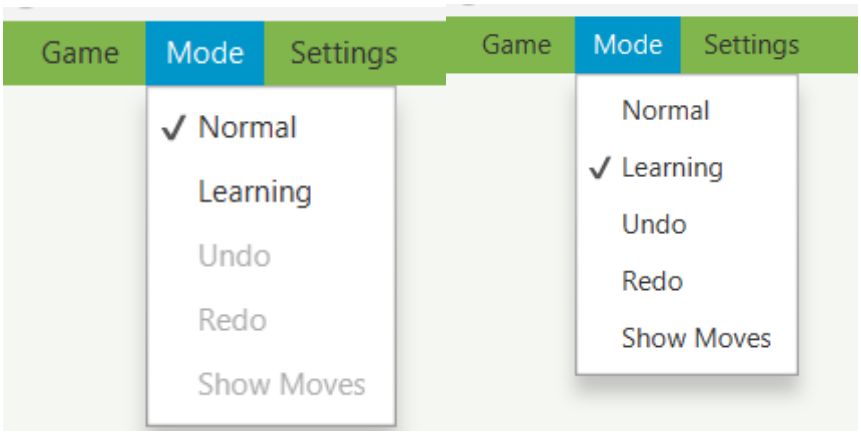

Figura 8 - Menu "Mode"

A *Figura 8* permite alterar o modo de Jogo (***Normal*** e ***Learning***) o modo ***Normal*** permite ser jogada uma partida tradicional de Xadrez (como se o jogo estivesse a ser jogado fisicamente). Já o modo ***Learning*** possibilita as opções ***Undo, Redo*** e ***Show Moves*** (apenas disponíveis neste modo). O ***Undo*** permite desfazer a última jogada, já o ***Redo*** permite desfazer a jogada desfeita. A opção de ***Show Moves*** permite ver os movimentos possíveis da peça selecionada (representado na *Figura 9*).

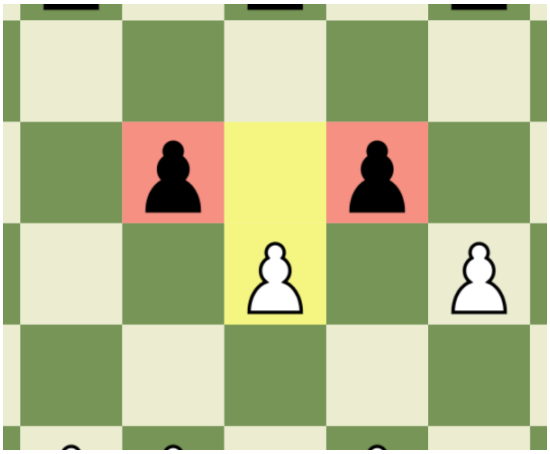

Figura 9 - Show Moves ativo

Model Log

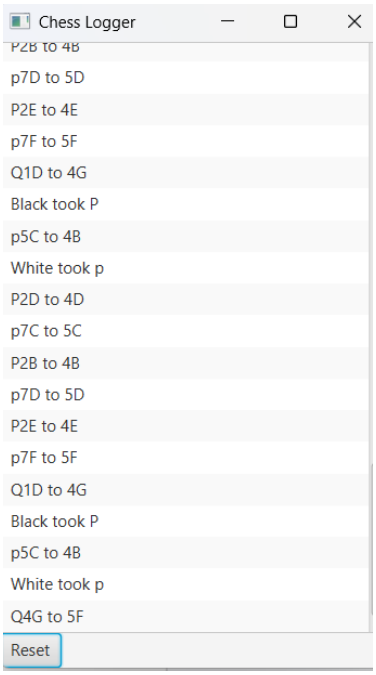

Figura 10 - Chess Logger

Na janela ***Chess Logger*** está representada cada jogada, a posição inicial e final da peça, assim como cada "***Take***" de cada jogador. É possível "limpar" o ***Chess Logger*** através do botão ***Reset***.

Checkmate

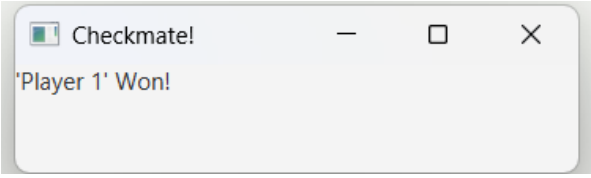

Figura 11 - Checkmate

A *Figura 11* mostra a janela de ***Checkmate*** e indica que o jogo acabou e quem venceu.

Evolve

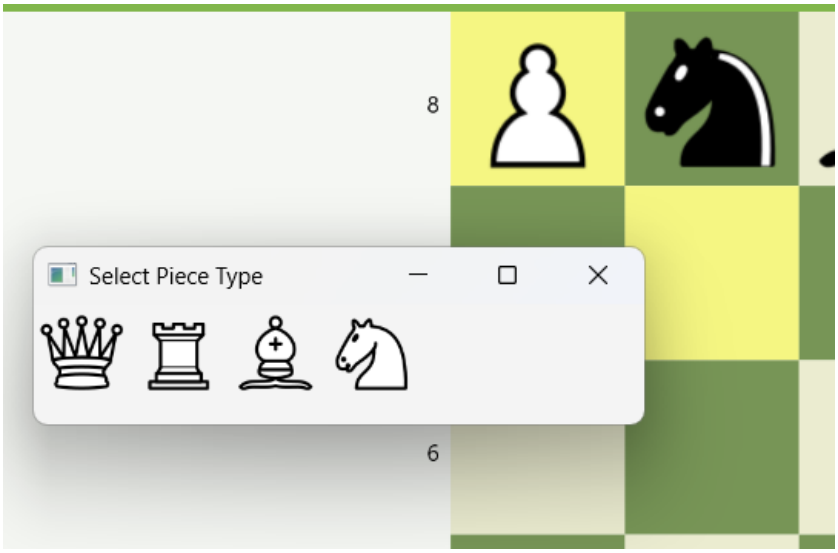

Figura 12 - Evolve do Peão

A *Figura 12* é apresentada quando um peão desbloqueia a sua funcionalidade de evoluir para outra peça. Nesta janela é possível escolher a peça pretendida para "ocupar" o lugar do peão.

Implementações

Implementação	Nível de Implementação
Roque	Completamente implementado
En Passant	Completamente implementado
Evolve do Peão	Completamente implementado
Checkmate	Parcialmente implementado (Verificação incompleta)
Property Change Support	Completamente implementado
undo e *redo*	Completamente implementado
Testes Unitários	Completamente implementado
ModelLog	Completamente implementado
Som	Completamente implementado

Conclusão

Concluindo, no desenvolvimento do trabalho prático foram explorados os conceitos lecionados na Unidade Curricular.

Este trabalho permitiu-nos pôr em prática a matéria lecionada e consequentemente aprofundar o nosso conhecimento em ***JavaFx*** e ***Java*** no geral, assim como aprofundou o nosso conhecimento no jogo de Xadrez e nas respectivas regras.