

Curso: Programação Orientada a Objetos com Java

<http://educandoweb.com.br>

Prof. Dr. Nelio Alves

Capítulo: Projeto Sistema de Jogo de Xadrez

Objetivo geral:

- Aplicar os conhecimentos aprendidos até o momento no curso para a construção de um projeto

System design

<https://github.com/acenelio/chess-system-design>

Creating project and git repository

Checklist:

- Github: create a new project
 - **NOTE:** choose `.gitignore` type as Java
- Open a terminal in project folder, and perform the following commands:

```
git init
git remote add origin https://github.com/acenelio/chess-system-java.git
git pull origin master
git add .
git commit -m "Project created"
git push -u origin master
```

First class: Position

Checklist:

- Class Position [public]
- **OOP Topics:**
 - Encapsulation
 - Constructors
 - ToString (Object / overriding)

Starting to implement Board and Piece

Checklist:

- Classes Piece, Board [public]
- **OOP Topics:**
 - Associations
 - Encapsulation / Access Modifiers
- **Data Structures Topics:**
 - Matrix

Chess layer and printing the board

```
8 - - - - - - - -
7 - - - - - - - -
6 - - - - - - - -
5 - - - - - - - -
4 - - - - - - - -
3 - - - - - - - -
2 - - - - - - - -
1 - - - - - - - -
  a b c d e f g h
```

Checklist:

- Methods: Board.Piece(row, column) and Board.Piece(position)
- Enum Chess.Color
- Class Chess.ChessPiece [public]
- Class Chess.ChessMatch [public]
- Class ChessConsole.UI
- **OOP Topics:**
 - Enumerations
 - Encapsulation / Access Modifiers
 - Inheritance
 - Downcasting
 - Static members
 - Layers pattern
- **Data Structures Topics:**
 - Matrix

Placing pieces on the board

Checklist:

- Method: Board.PlacePiece(piece, position)
- Classes: Rook, King [public]
- Method: ChessMatch.InitialSetup
- **OOP Topics:**
 - Inheritance
 - Overriding
 - Polymorphism (ToString)

BoardException and defensive programming

Checklist:

- Class BoardException [public]
- Methods: Board.PositionExists, Board.TherelsAPiece
- Implement defensive programming in Board methods
- **OOP Topics:**
 - Exceptions
 - Constructors (a string must be informed to the exception)

ChessException and ChessPosition

Checklist:

- Class ChessException [public]
- Class ChessPosition [public]
- Refactor ChessMatch.InitialSetup
- **OOP Topics:**
 - Exceptions
 - Encapsulation
 - Constructors (a string must be informed to the exception)
 - Overriding
 - Static members
 - Layers pattern

Little improvement in board printing

Color in terminal:

- Windows: Git Bash
- Mac: Google "osx terminal color"

Checklist:

- Place more pieces on the board
- Distinguish piece colors in UI.PrintPiece method

Moving pieces

Checklist:

- Method Board.RemovePiece
- Method UI.ReadChessPosition
- Method ChessMatch.PerformChessMove
 - Method ChessMatch.MakeMove
 - Method ChessMatch.ValidateSourcePosition
- Write basic logic on Program.cs
- **OOP Topics:**
 - Exceptions
 - Encapsulation

Handling exceptions and clearing screen

Clear screen using Java:

```
// https://stackoverflow.com/questions/2979383/java-clear-the-console  
public static void clearScreen() {  
    System.out.print("\033[H\033[2J");  
    System.out.flush();  
}
```

Checklist:

- ChessException
- InputMismatchException

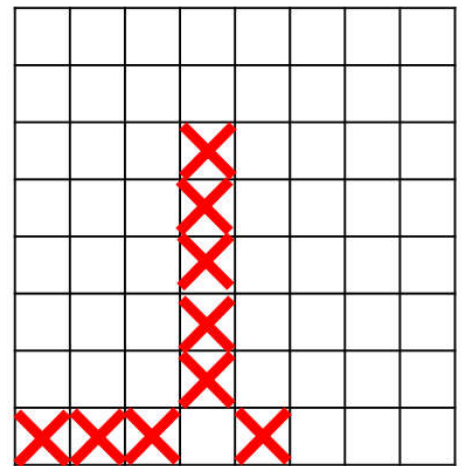
Possible moves of a piece



Input: a piece



Output: a boolean matrix of possible movements



Checklist:

- Methods in Piece:
 - PossibleMoves [abstract]
 - PossibleMove
 - IsThereAnyPossibleMove
- Basic PossibleMove implementation for Rook and King
- Update ChessMatch.ValidadeSourcePosition

- **OOP Topics:**
 - Abstract method / class
 - Exceptions

Implementing possible moves of Rook

Checklist:

- Method ChessPiece.IsThereOpponentPiece(position) [protected]
- Implement Rook.PossibleMoves
- Method ChessMatch.ValidateTargetPosition
- **OOP Topics:**
 - Polymorphism
 - Encapsulation / access modifiers [protected]
 - Exceptions

Printing possible moves

Checklist:

- Method ChessMatch.PossibleMoves
- Method UI.PrintBoard [overload]
- Refactor main program logic
- **OOP Topics:**
 - Overloading

Implementing possible moves of King

Checklist:

- Method King.CanMove(position) [private]
- Implement King.PossibleMoves
- **OOP Topics:**
 - Encapsulation
 - Polymorphism

Switching player each turn

Checklist:

- Class ChessMatch:
 - Properties Turn, CurrentPlayer [private set]
 - Method NextTurn [private]
 - Update PerformChessMove
 - Update ValidadeSourcePosition
- Method UI.PrintMatch
- **OOP Topics:**
 - Encapsulation
 - Exceptions

Handling captured pieces

Checklist:

- Method UI.PrintCapturedPieces
- Update UI.PrintMatch
- Update Program logic
- Lists in ChessMatch: `_piecesOnTheBoard`, `_capturedPieces`
 - Update constructor
 - Update PlaceNewPiece
 - Update MakeMove
- **OOP Topics:**
 - Encapsulation
 - Constructors
- **Data Structures Topics:**
 - List

Check logic

Rules:

- Check means your king is under threat by at least one opponent piece
- You can't put yourself in check

Checklist:

- Property ChessPiece.ChessPosition [get]
- Class ChessMatch:
 - Method UndoMove
 - Property Check [private set]
 - Method Opponent [private]
 - Method King(color) [private]
 - Method TestCheck
 - Update PerformChessMove
- Update UI.PrintMatch

Checkmate logic

Checklist:

- Class ChessMatch:
 - Property Checkmate [private set]
 - Method TestCheckmate [private]
 - Update PerformChessMove
- Update UI.PrintMatch
- Update Program logic

Piece move count

Checklist:

- Class ChessPiece:
 - Property MoveCount [private set]
 - Method IncreaseMoveCount [internal]
 - Method DecreaseMoveCount [internal]
- Class ChessMatch:
 - Update MakeMove
 - Update UndoMove
- **OOP Topics:**
 - Encapsulation

Pawn

Checklist:

- Class Pawn
- Update ChessMatch.InitialSetup
- **OOP Topics:**
 - Encapsulation
 - Inheritance
 - Polymorphism

Bishop

Checklist:

- Class Bishop
- Update ChessMatch.InitialSetup
- **OOP Topics:**
 - Encapsulation
 - Inheritance
 - Polymorphism

Knight

Checklist:

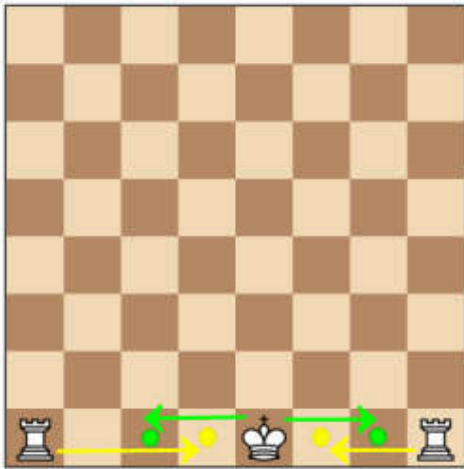
- Class Knight
- Update ChessMatch.InitialSetup
- **OOP Topics:**
 - Encapsulation
 - Inheritance
 - Polymorphism

Queen

Checklist:

- Class Queen
- Update ChessMatch.InitialSetup
- **OOP Topics:**
 - Encapsulation
 - Inheritance
 - Polymorphism

Special move - Castling



Checklist:

- Update King
- Update ChessMatch.MakeMove
- Update ChessMatch.UndoMove

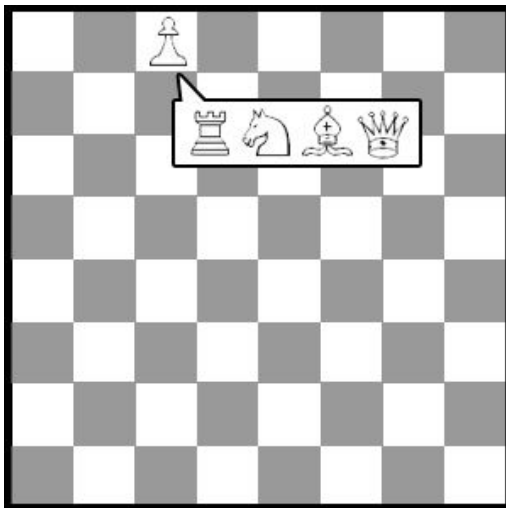
Special move - En Passant



Checklist:

- Register a pawn which can be captured by en passant on next turn
 - Property ChessMatch.EnPassantVulnerable
 - Update ChessMatch.PerformChessMove
- Update Pawn.PossibleMoves
- Update ChessMatch.MakeMove
- Update ChessMatch.UndoMove
- Update ChessMatch.InitialSetup

Special move - Promotion



Checklist:

- Property ChessMatch.Promoted
- Update ChessMatch.PerformChessMove
- Method ChessMatch.ReplacePromotedPiece
- Update Program logic