Class Dictionary

Jean-Philippe Miguel-Gagnon, Jérémy Gaouette, Raphaël Rail

Thursday, 31th of march 2022

Presented to : Charles Jacob

Contents

1 Introduction

The goal of this document is to inform the programmer about classes used to create a C# OOP Chess game.

We'll go through this with the MVC model approch to make it clearer for the programmer where to implement his code.

2 Models

2.1 Classe Piece

This is an abstract class for a base piece for the game.

(Abstract) Piece
-_colour : Colour
+CanCollide() : bool
+ValidMove(int x1, int y1, int x2, int y2) : bool
+ToString() : string
+Canpromote() : bool
+IsEssential() : bool

2.1.1 Fields

Field Name	Type	Visibility
_colour	Colour	Private

Description : This field represent the colour a piece, wich can only be black or white as it is for a regular chess game.

2.1.2 Methods

Method Name	Parameters	Returned Type	Visibility
ValidMove	x1, y1, x2, y2 : int	bool	Public

Parameters : x1 ans y1 represent the coordinates of the position before a possible move and x2 and y2 are the coordinates of the position after the move. These are all of Integer(16) type.
Description : The method returns true if the piece can move to the 2nd position.

Method Name	Parameters	Returned Type	Visibility
CanCollide	None	bool	Public

Description : Returns true if the piece can't go over other pieces, otherwise it returns false.

Method Name	Parameters	Returned Type	Visibility
ToString	None	String	Public

Description : Empty method te be overrided by child classes.

Method Name	Parameters	Returned Type	Visibility
CanPromote	None	bool	Public

Description : Returns true if the piece is promotable.

Method Name	Parameters	Returned Type	Visibility
IsEssential	None	bool	Public

Description : Returns true if the piece is essential.

2.1.3 Properties

Property Name	Parameters	Returned Type	Visibility
Colour	None	Colour	Public

Description : Gets the colour of a piece.

2.2 Class StartingPiece : Piece

Another abstract Class that inherits to the base Class Piece. This Class is for pieces that have different behaviour after they have mouved for the first time.

(Abstract) StartingPiece
-_hasMoved : bool
+ValidMove(int x1, int y1, int x2, int y2) : bool
+ToString()

2.2.1 Fields

Field Name	**Type**	**Visibility**
_hasMoved	bool	Private

Description : This field is used to tell if the piece has already made its first move.

2.2.2 Methods

Same as Class Piece.

2.2.3 Properties

Property Name	**Parameters**	**Returned Type**	**Visibility**
HasMoved	None	bool	Public

Description : Gets or Sets the property _hasMoved of a piece to true or false.

2.3 Class Pawn : StartingPiece

This Class inherits the Class StartingPiece. It represents the pawn piece of a regular chess game.

Pawn
+ValidMove(x1, y1, x2, y2) : bool
+CanPromote() : bool
+ToString() : string

2.3.1 Fields

Description : This Class doesn't provide a new field. It just inherits the fields of its parent.

2.3.2 Methods

Method Name	**Parameters**	**Returned Type**	**Visibility**
ValidMove	x1, y1, x2, y2 : int	bool	Public

Parameters : x1 ans y1 represent the coordinates of the position before a possible move and x2 and y2 are the coordinates of the position after the move. These are all of Integer(16) type.
Description : This method checks if the move provided is valid considering the basic moves of a pawn which are 1 or 2 cells ahead, or 1 diagonal cell if an opponent piece is present. In thoses cases it returns true, otherwise it returns false.

Method Name	**Parameters**	**Returned Type**	**Visibility**
CanPromote	None	bool	Public

Description : Returns true if the piece can promote.

Method Name	**Parameters**	**Returned Type**	**Visibility**
ToString	None	String	Public

Description : This method overrides ToString virtual method and returns minus "p" if the piece is white and upper "P" if it is black.

2.3.3 Properties

Same as parent.

2.4 Class Rook : StartingPiece

This Class inherits the Class StartingPiece. It represents the rook piece of a regular chess game.

Rook
+ValidMove(x1, y1, x2, y2) : bool
+ToString() : string

2.4.1 Fields

Description : This Class doesn't provide a new field. It just inherits the fields of its parent.

2.4.2 Methods

Method Name	**Parameters**	**Returned Type**	**Visibility**
ValidMove	x1, y1, x2, y2 : int	bool	Public

Parameters : x1 ans y1 represent the coordinates of the position before a possible move and x2 and y2 are the coordinates of the position after the move. These are all of Integer(16) type.
Description : This method checks if the move provided is valid considering the basic moves of a rook which are forward, backward or sideways to any empty cell. In thoses cases it returns true, otherwise it returns false.

Method Name	**Parameters**	**Returned Type**	**Visibility**
ToString	None	String	Public

Description : This method overrides ToString virtual method and returns minus "r" if the piece is white and upper "R" if it is black.

2.4.3 Properties

Same as parent.

2.5 Class King : StartingPiece

This Class inherits the Class StartingPiece. It represents the king piece of a regular chess game.

King
+ValidMove(x1, y1, x2, y2) : bool
+IsEssential() : bool
+ToString() : string

2.5.1 Fields

Description : This Class doesn't provide a new field. It just inherits the fields of its parent.

2.5.2 Methods

Method Name	**Parameters**	**Returned Type**	**Visibility**
ValidMove	x1, y1, x2, y2 : int	bool	Public

Parameters : x1 ans y1 represent the coordinates of the position before a possible move and x2 and y2 are the coordinates of the position after the move. These are all of Integer(16) type.
Description : This method checks if the move provided is valid considering the basic moves of a king which are one square horizontally, vertically, or diagonally unless the square is already occupied by a friendly piece or the move would place the king in check. In thoses cases it returns true, otherwise it returns false.

Method Name	**Parameters**	**Returned Type**	**Visibility**
IsEssential	None	bool	Public

Description : Returns true if the piece is essential.

Method Name	**Parameters**	**Returned Type**	**Visibility**
ToString	None	String	Public

Description : This method overrides ToString virtual method and returns minus "k" if the piece is white and upper "K" if it is black.

2.5.3 Properties

Same as parent.

2.6 Class Knight : Piece

This Class inherits the Class Piece. It represents the knight piece of a regular chess game.

Knight
+ValidMove(x1, y1, x2, y2) : bool
+CanCollide() : bool
+ToString() : string

2.6.1 Fields

Description : This Class doesn't provide a new field. It just inherits the fields of its parent.

2.6.2 Methods

Method Name	**Parameters**	**Returned Type**	**Visibility**
ValidMove	x1, y1, x2, y2 : int	bool	Public

Parameters : x1 ans y1 represent the coordinates of the position before a possible move and x2 and y2 are the coordinates of the position after the move. These are all of Integer(16) type.
Description : This method checks if the move provided is valid considering the basic moves of a knight which are "L-shape"—that is, they can move two squares in any direction vertically followed by one square horizontally, or two squares in any direction horizontally followed by one square vertically. In thoses cases it returns true, otherwise it returns false.

Method Name	**Parameters**	**Returned Type**	**Visibility**
CanCollide	None	bool	Public

Description : Returns true if the piece can collide.

Method Name	**Parameters**	**Returned Type**	**Visibility**
ToString	None	String	Public

Description : This method overrides ToString virtual method and returns minus "n" if the piece is white and upper "N" if it is black.

2.6.3 Properties

Same as parent.

2.7 Class Bishop : Piece

This Class inherits the Class Piece. It represents the bishop piece of a regular chess game.

Bishop
+ValidMove(x1, y1, x2, y2) : bool
+ToString() : string

2.7.1 Fields

Description : This Class doesn't provide a new field. It just inherits the fields of its parent.

2.7.2 Methods

Method Name	**Parameters**	**Returned Type**	**Visibility**
ValidMove	x1, y1, x2, y2 : int	bool	Public

Parameters : x1 ans y1 represent the coordinates of the position before a possible move and x2 and y2 are the coordinates of the position after the move. These are all of Integer(16) type.
Description : This method checks if the move provided is valid considering the basic moves of a bishop which are any direction diagonally with no limit of cells unless there is another piece obstructing its path. In thoses cases it returns true, otherwise it returns false.

Method Name	**Parameters**	**Returned Type**	**Visibility**
ToString	None	String	Public

Description : This method overrides ToString virtual method and returns minus "b" if the piece is white and upper "B" if it is black.

2.7.3 Properties

Same as parent.

2.8 Class Queen : Piece

This Class inherits the Class Piece. It represents the queen piece of a regular chess game.

Queen
+ValidMove(x1, y1, x2, y2) : bool
+ToString() : string

2.8.1 Fields

Description : This Class doesn't provide a new field. It just inherits the fields of its parent.

2.8.2 Methods

Method Name	Parameters	Returned Type	Visibility
ValidMove	x1, y1, x2, y2 : int	bool	Public

Parameters : x1 ans y1 represent the coordinates of the position before a possible move and x2 and y2 are the coordinates of the position after the move. These are all of Integer(16) type.
Description : This method checks if the move provided is valid considering the basic moves of a bishop which are any direction diagonally with no limit of cells unless there is another piece obstructing its path. In thoses cases it returns true, otherwise it returns false.

Method Name	Parameters	Returned Type	Visibility
ToString	None	String	Public

Description : This method overrides ToString virtual method and returns minus "q" if the piece is white and upper "Q" if it is black.

2.8.3 Properties

Same as parent.

2.9 Class Match

This Class represents the model of a chess match wich compose the Game-Controller. It will keep all changes that the game controller will return.

Match
-_board : Board
-_current : Colour
-_history : string[]
-_turnNumber : int
+ExportBoard() : string
+ExportHistory() : string[]
+ValidTurn(int origin, int target) : bool
+MakeTurn(int origin, int target) : void
+ValidSelection(int cell, bool firstClick) : bool

2.9.1 Fields

Field Name	**Type**	**Visibility**
_board	Board	Private

Description : This field represent the board of a match wich compose the match. Its type is Board wich is the next class to discuss.

Field Name	**Type**	**Visibility**
_current	Colour	Private

Description : This field tells us wich piece colour is currently playing, white or black.

Field Name	**Type**	**Visibility**
_history	string[]	Private

Description : This field is a table that contains strings that represent previous board states to keep track of what has been played. For exemple, the first string would look like that :
"RNBKQBNRPPPPPPPP.....pppppppprnkqbnr".

Field Name	Type	Visibility
_turnNumber	int	Private

Description : This field tracks the number of turns that have been played.

2.9.2 Methods

Method Name	Parameters	Returned Type	Visibility
ExportBoard	none	string	Public

Description : The method returns the the board as a string of 64 char.

Method Name	Parameters	Returned Type	Visibility
ExportHistory	none	string[]	Public

Description : The method returns the table _history.

Method Name	Parameters	Returned Type	Visibility
ValidTurn	int origin, int target	bool	Public

Parameters : Parameter origin represents the cell where the piece is before the move and target is the targeted cell.

Description : The method returns true if the turn is valid.

Method Name	Parameters	Returned Type	Visibility
MakeTurn	int origin, int target	void	Public

Parameters : Parameter origin represents the cell where the piece is before the move and target is the targeted cell.

Description : The method makes all changes to the board in order to make the turn.

Method Name	Parameters	Returned Type	Visibility
ValidSelection	int cell, bool firstClick	bool	Public

Parameters : Parameter cell represents the cell where the player clicks and the parameter firstClick is a bool that returns true if its the first click.

Description : The method checks if the selection made in the board is valid.

2.9.3 Properties

Property Name	Parameters	Returned Type	Visibility
Current	None	Colour	Public

Description : Gets or Sets the property _current of a match (Colour white or black).

2.10 Class Board

This Class represents the model of a chess board wich compose the match. It will return all changes to the match.

Board
-_cells : Cell[]
+ToString() : string
+Collision(int origin, int target) : bool
+SameColour (int cell, Colour colour) : bool
+ValidMove(int origin, int target) : bool
+MoveCellTo(int origin, int target) : void
+BoardFromString(string board) : void
+IsEssetialExposed(Colour colour) : bool

2.10.1 Fields

Field Name	**Type**	**Visibility**
_cells	Cell[]	Private

Description : This field represent all 64 cells on a chess board. Each cell will contain a piece or be empty.

2.10.2 Methods

Method Name	**Parameters**	**Returned Type**	**Visibility**
Collision	int origin, int target	bool	Public

Parameters : Parameter origin represents the cell where the piece is before the move and target is the targeted cell.

Description : This method returns true if it detects a possible collision in the path to the targeted cell.

Method Name	**Parameters**	**Returned Type**	**Visibility**
SameColour	int cell, Colour colour	bool	Public

Parameters : Parameter cell represents the index of one of the 64 cells that is selected and colour is the colour of the selected cell.

Description : This method returns true the cell selected contains a piece at the same colour of the current turn.

Method Name	Parameters	Returned Type	Visibility
ValidMove	int origin, int target	bool	Public

Parameters : Parameter origin represents the index of the origin cell and the target parameter is the cell targeted.

Description : This method returns true if the move is valid.

Method Name	Parameters	Returned Type	Visibility
MoveCellTo	int origin, int target	bool	Public

Parameters : Parameter origin represents the index of the origin cell and the target parameter is the cell targeted.

Description : This method swaps cells origin and target.

Method Name	Parameters	Returned Type	Visibility
BoardFromString	string board	void	Public

Parameters : Board parameter is a 64 char string that represent a board state(tells where the pieces are supposed to be).

Description : This method takes a string as board and transforms each of the 64 char as the content of each of the 64 cells.

Method Name	Parameters	Returned Type	Visibility
IsEssentialExposed	Colour colour	bool	Public

Parameters : The parameter colour represent the colour to test.

Description : Checks if the essential piece of the colour white or black is exposed.

2.11 Class Cell

This Class represents the model of a chess cell and its content.

Cell
-_piece : Nullable<Piece>
+IsEmpty() : bool
+HasCollision() : bool
+HasPromotable() : bool
+HasEssetial() : bool
+ValidMove(int x1, int y1, int x2, int y2) : bool
+Colour() : Colour

2.11.1 Fields

Field Name	Type	Visibility
_piece	Nullable<Piece>	Private

Description : This field represents de content of a cell if it is a piece or null if it's empty.

2.11.2 Methods

Method Name	Parameters	Returned Type	Visibility
IsEmpty	none	bool	Public

Description : This method returns true if the _piece is null, so it's empty.

Method Name	Parameters	Returned Type	Visibility
HasCollision	none	bool	Public

Description : This method returns true if the cell contains a piece that can collide.

Method Name	Parameters	Returned Type	Visibility
HasPromotable	none	bool	Public

Description : This method returns true if the cell contains a promotable piece.

Method Name	Parameters	Returned Type	Visibility
HasEssential	none	bool	Public

Description : This method returns true if the cell contains a piece that is essential.

Method Name	Parameters	Returned Type	Visibility
ValidMove	int x1, int y1, int x2, int y2	bool	Public

Parameters : x1 ans y1 represent the coordinates of the position before a possible move and x2 and y2 are the coordinates of the position after the move. These are all of Integer(16) type.

Description : This method returns true if the move is valid.

Method Name	Parameters	Returned Type	Visibility
Colour	none	Colour	Public

Description : This method returns the colour if the field _piece is not null.

2.12 Class Player

This Class represents a player in a chess game.

Player
-_name : string
-_points : int

2.12.1 Fields

Field Name	Type	Visibility
_name	string	Private

Description : This field is the name of the player.

Field Name	Type	Visibility
_points	int	Private

Description : This field is the points that the player collects during a game.

3 Controllers

3.1 Class Chess

Chess
-_listGames : List<GameController>
+main() : void
+NewGame() : void
+StartGame(Player[2] players) : void
+ManagePlayers() : void
+Exit() : void

3.2 Class GameController

GameController
-_main : Chess
-_selected : int
-_match : Match
-_playerA : Player
-_playerB : Player
-_tieCounter : int
-_view : FormMatch
-Check() : bool
-SelfCheck() : bool
-Checkmate() : bool
-Castle() : bool
-Promotion() : bool
-FiftyTurns() : bool
-SameBoard() : bool
+Turn(int origin, int target) : void
+Selection(int cell) : void
+Resign() : void

3.3 Class PlayerController

PlayerController
-_main : Chess
-_list : List<Player>
+Add() : void
+Remove() : void

4 Views

4.1 Class FormSelection

4.2 Class FormMenu

4.3 Class FormPromotion

4.4 Class FormMatch

4.5 Class FormLeaderboard