App
- acceptedArgs : List<String>
- start(stage : Stage) : void
- main(args : String[]) : void

loads FXML

GameController
- og : OthelloGame
- guiBoard : GUISpace[][]
- skippedTurns : int
- initGame(arg1 : String, arg2 : String) : void
- takeTurn(player : Player) : void
- showMoves(human : HumanPlayer) : void
- computerDecision(computer : ComputerPlayer) : void
- selectSpace(...)
- updateGUIBoard(...)
- gameOver() : void

uses game logic   *renders GUI squares*

OthelloGame
- board : BoardSpace[][]
- playerOne : Player
- playerTwo : Player
- OthelloGame(p1 : Player, p2 : Player)
- initBoard() : void
- getAvailableMoves(player : Player) : Map<BoardSpace, List<BoardSpace>>
- takeSpace(...)
- takeSpaces(...)
- computerDecision(computer : ComputerPlayer) : BoardSpace

GUISpace
- x : int
- y : int
- type : BoardSpace.SpaceType
- squarePane : Pane
- bg : Rectangle
- disc : Circle
- GUISpace(x : int, y : int, type : BoardSpace.SpaceType)
- setBgColor(color : Color) : void
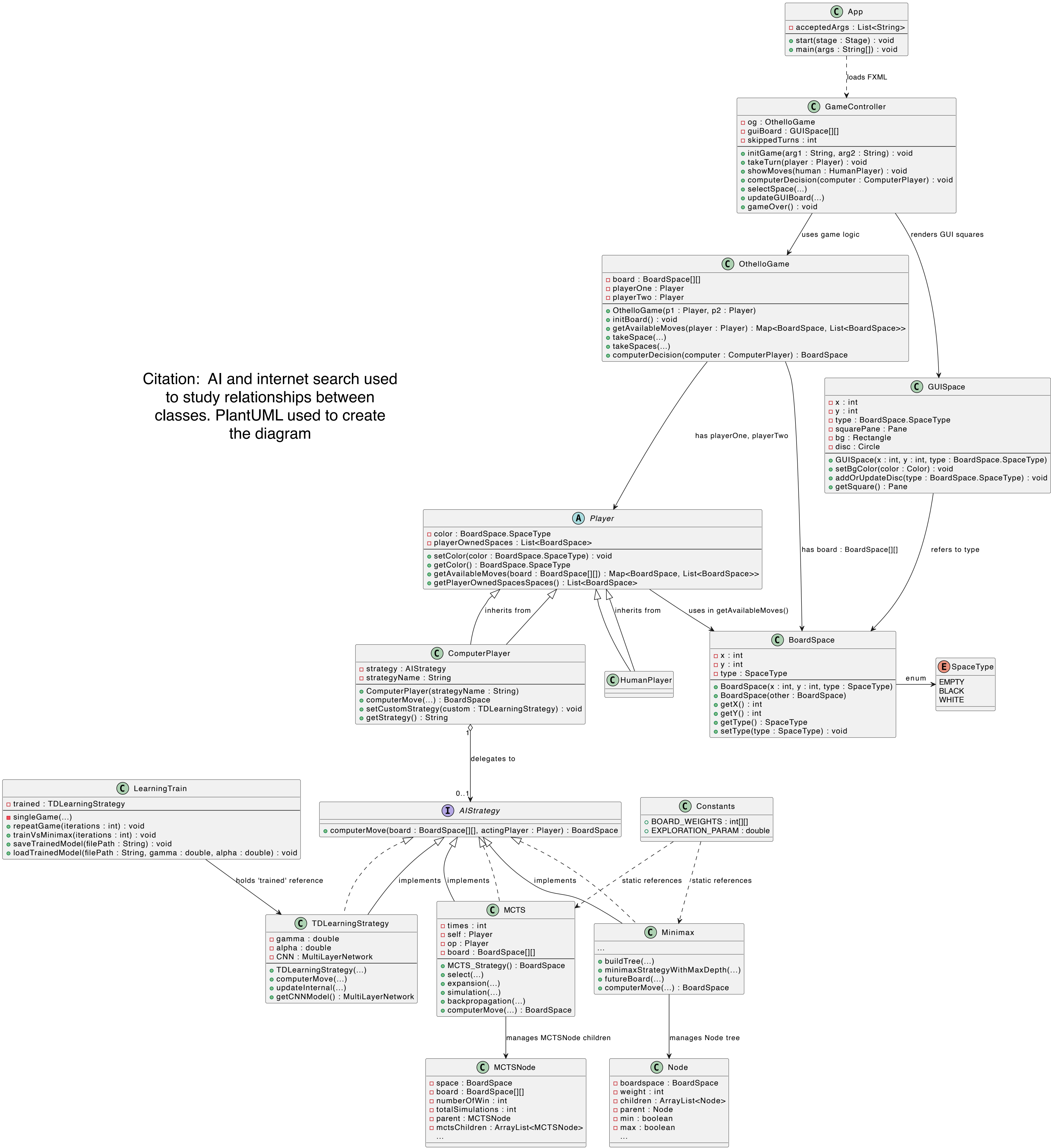- addOrUpdateDisc(type : BoardSpace.SpaceType) : void
- getSquare() : Pane

Citation: AI and internet search used to study relationships between classes. PlantUML used to create the diagram

has playerOne, playerTwo   *has board : BoardSpace[][]*   *refers to type*

Player (A)
- color : BoardSpace.SpaceType
- playerOwnedSpaces : List<BoardSpace>
- setColor(color : BoardSpace.SpaceType) : void
- getColor() : BoardSpace.SpaceType
- getAvailableMoves(board : BoardSpace[][]) : Map<BoardSpace, List<BoardSpace>>
- getPlayerOwnedSpacesSpaces() : List<BoardSpace>

inherits from   *inherits from*   *uses in getAvailableMoves()*

ComputerPlayer
- strategy : AIStrategy
- strategyName : String
- ComputerPlayer(strategyName : String)
- computerMove(...) : BoardSpace
- setCustomStrategy(custom : TDLearningStrategy) : void
- getStrategy() : String

HumanPlayer

BoardSpace
- x : int
- y : int
- type : SpaceType
- BoardSpace(x : int, y : int, type : SpaceType)
- BoardSpace(other : BoardSpace)
- getX() : int
- getY() : int
- getType() : SpaceType
- setType(type : SpaceType) : void

enum

SpaceType (E)
EMPTY
BLACK
WHITE

delegates to

LearningTrain
- trained : TDLearningStrategy
- singleGame(...)
- repeatGame(iterations : int) : void
- trainVsMinimax(iterations : int) : void
- saveTrainedModel(filePath : String) : void
- loadTrainedModel(filePath : String, gamma : double, alpha : double) : void

0..1

AIStrategy (I)
- computerMove(board : BoardSpace[][], actingPlayer : Player) : BoardSpace

Constants
- BOARD_WEIGHTS : int[][]
- EXPLORATION_PARAM : double

holds 'trained' reference   *implements*   *implements*   *implements*   *static references*   *static references*

TDLearningStrategy
- gamma : double
- alpha : double
- CNN : MultiLayerNetwork
- TDLearningStrategy(...)
- computerMove(...)
- updateInternal(...)
- getCNNModel() : MultiLayerNetwork

MCTS
- times : int
- self : Player
- op : Player
- board : BoardSpace[][]
- MCTS_Strategy() : BoardSpace
- select(...)
- expansion(...)
- simulation(...)
- backpropagation(...)
- computerMove(...) : BoardSpace

Minimax
- ...
- buildTree(...)
- minimaxStrategyWithMaxDepth(...)
- futureBoard(...)
- computerMove(...) : BoardSpace

manages MCTSNode children   *manages Node tree*

MCTSNode
- space : BoardSpace
- board : BoardSpace[][]
- numberOfWin : int
- totalSimulations : int
- parent : MCTSNode
- mctsChildren : ArrayList<MCTSNode>
- ...

Node
- boardspace : BoardSpace
- weight : int
- children : ArrayList<Node>
- parent : Node
- min : boolean
- max : boolean
- ...