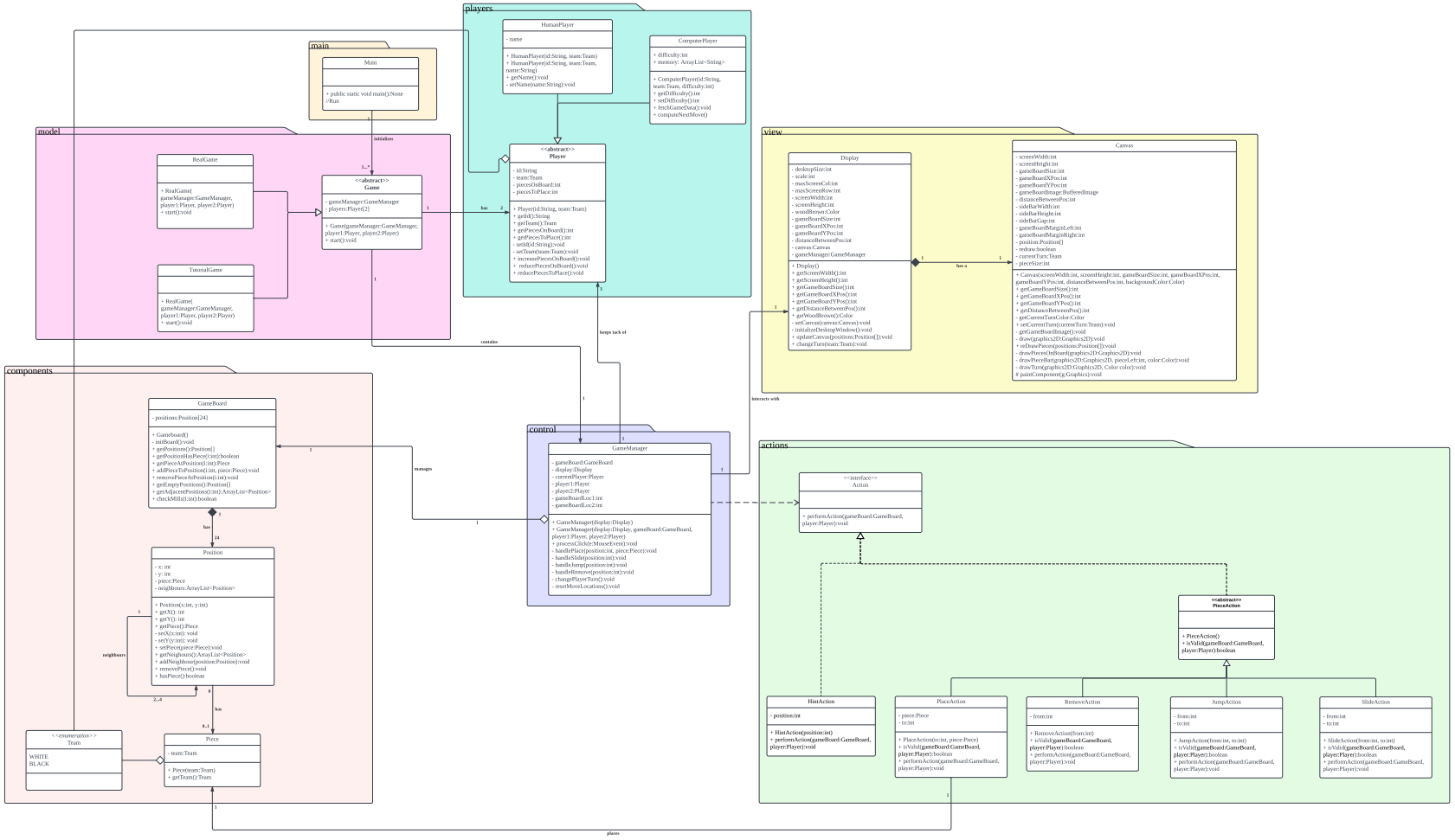
main
Main
+ public static void main():None
//Run
initializes
1..*
model
RealGame
+ RealGame(
gameManager:GameManager,
player1:Player, player2:Player)
+ start():void
<<abstract>>
Game
- gameManager:GameManager
- players:Player[2]
+ Game(gameManager:GameManager,
player1:Player, player2:Player)
+ start():void
TutorialGame
+ RealGame(
gameManager:GameManager,
player1:Player, player2:Player)
+ start():void
contains
1
1
players
HumanPlayer
- name
+ HumanPlayer(id:String, team:Team)
+ HumanPlayer(id:String, team:Team,
name:String)
+ getName():void
- setName(name:String):void
ComputerPlayer
+ difficulty:int
+ memory: ArrayList<String>
+ ComputerPlayer(id:String,
team:Team, difficulty:int)
+ getDifficulty():int
+ setDifficulty():int
+ fetchGameData():void
+ computeNextMove()
<<abstract>>
Player
- id:String
- team:Team
- piecesOnBoard:int
- piecesToPlace:int
+ Player(id:String, team:Team)
+ getId():String
+ getTeam():Team
+ getPiecesOnBoard():int
+ getPiecesToPlace():int
- setId(id:String):void
- setTeam(team:Team):void
+ increasePiecesOnBoard():void
+ reducePiecesOnBoard():void
+ reducePiecesToPlace():void
has
2
1
keeps tack of
1
view
Display
- desktopSize:int
- scale:int
- maxScreenCol:int
- maxScreenRow:int
- screenWidth:int
- screenHeight:int
- woodBrown:Color
- gameBoardSize:int
- gameBoardXPos:int
- gameBoardYPos:int
- distanceBetweenPos:int
- canvas:Canvas
- gameManager:GameManager
+ Display()
+ getScreenWidth():int
+ getScreenHeight():int
+ getGameBoardSize():int
+ getGameBoardXPos():int
+ getGameBoardYPos():int
+ getDistanceBetweenPos():int
+ getWoodBrown():Color
- setCanvas(canvas:Canvas):void
- initializeDesktopWindow():void
+ updateCanvas(positions:Position[]):void
+ changeTurn(team:Team):void
1
has a
1
Canvas
- screenWidth:int
- screenHeight:int
- gameBoardSize:int
- gameBoardXPos:int
- gameBoardYPos:int
- gameBoardImage:BufferedImage
- distanceBetweenPos:int
- sideBarWidth:int
- sideBarHeight:int
- sideBarGap:int
- gameBoardMargin Left:int
- gameBoardMargin Right:int
- position:Position[]
- redraw:boolean
- currentTurn:Team
- pieceSize:int
+ Canvas(screenWidth:int, screenHeight:int, gameBoardSize:int, gameBoardXPos:int,
gameBoardYPos:int, distanceBetweenPos:int, backgroundColor:Color)
+ getGameBoardSize():int
+ getGameBoardXPos():int
+ getGameBoardYPos():int
+ getDistanceBetweenPos():int
+ getCurrentTurnColor:Color
+ setCurrentTurn(currentTurn:Team):void
+ getGameBoardImage():void
+ draw(graphics2D:Graphics2D):void
+ reDrawPieces(positions:Position[]):void
- drawPiecesOnBoard(graphics2D:Graphics2D):void
- drawPieceBar(graphics2D:Graphics2D, pieceLeft:int, color:Color):void
- drawTurn(graphics2D:Graphics2D, Color color):void
paintComponent(g:Graphics):void
1
interacts with
components
GameBoard
- positions:Position[24]
+ Gameboard()
- initBoard():void
+ getPositions():Position[]
+ getPositionHasPiece():int):boolean
+ getPieceAtPosition():int):Piece
+ addPieceToPosition(:int, piece:Piece):void
+ removePieceAtPosition(:int):void
+ getEmptyPositions():Position[]
+ getAdjacentPositions(:int):ArrayList<Position>
+ checkMills(:int):boolean
1
has
24
Position
- x: int
- y: int
- piece:Piece
- neighbours:ArrayList<Position>
+ Position(x:int, y:int)
+ getX():int
+ getY():int
+ getPiece():Piece
- setX(x:int): void
- setY(y:int): void
+ setPiece(piece:Piece):void
+ getNeighbours():ArrayList<Position>
+ addNeighbour(position:Position):void
+ removePiece():void
+ hasPiece():boolean
1
neighbours
2..4
0
has
0..1
<<enumeration>>
Team
WHITE
BLACK
Piece
- team:Team
+ Piece(team:Team)
+ getTeam():Team
1
places
manages
1
1
control
GameManager
- gameBoard:GameBoard
- display:Display
- currentPlayer:Player
- player1:Player
- player2:Player
- gameBoardLoc1:int
- gameBoardLoc2:int
+ GameManager(display:Display)
+ GameManager(display:Display, gameBoard:GameBoard,
player1:Player, player2:Player)
+ processClick(e:MouseEvent):void
- handlePlace(position:int, piece:Piece):void
- handleSlide(position:int):void
- handleJump(position:int):void
- handleRemove(position:int):void
- changePlayerTurn():void
- resetMoveLocations():void
1
1
actions
<<interface>>
Action
+ performAction(gameBoard:GameBoard,
player:Player):void
<<abstract>>
PieceAction
+ PieceAction()
+ isValid(gameBoard:GameBoard,
player:Player):boolean
HintAction
- position:int
+ HintAction(position:int)
+ performAction(gameBoard:GameBoard,
player:Player):void
PlaceAction
- piece:Piece
- to:int
+ PlaceAction(to:int, piece:Piece)
+ isValid(gameBoard:GameBoard,
player:Player):boolean
+ performAction(gameBoard:GameBoard,
player:Player):void
RemoveAction
- from:int
+ RemoveAction(from:int)
+ isValid(gameBoard:GameBoard,
player:Player):boolean
+ performAction(gameBoard:GameBoard,
player:Player):void
JumpAction
- from:int
- to:int
+ JumpAction(from:int, to:int)
+ isValid(gameBoard:GameBoard,
player:Player):boolean
+ performAction(gameBoard:GameBoard,
player:Player):void
SlideAction
- from:int
- to:int
+ SlideAction(from:int, to:int)
+ isValid(gameBoard:GameBoard,
player:Player):boolean
+ performAction(gameBoard:GameBoard,
player:Player):void
1