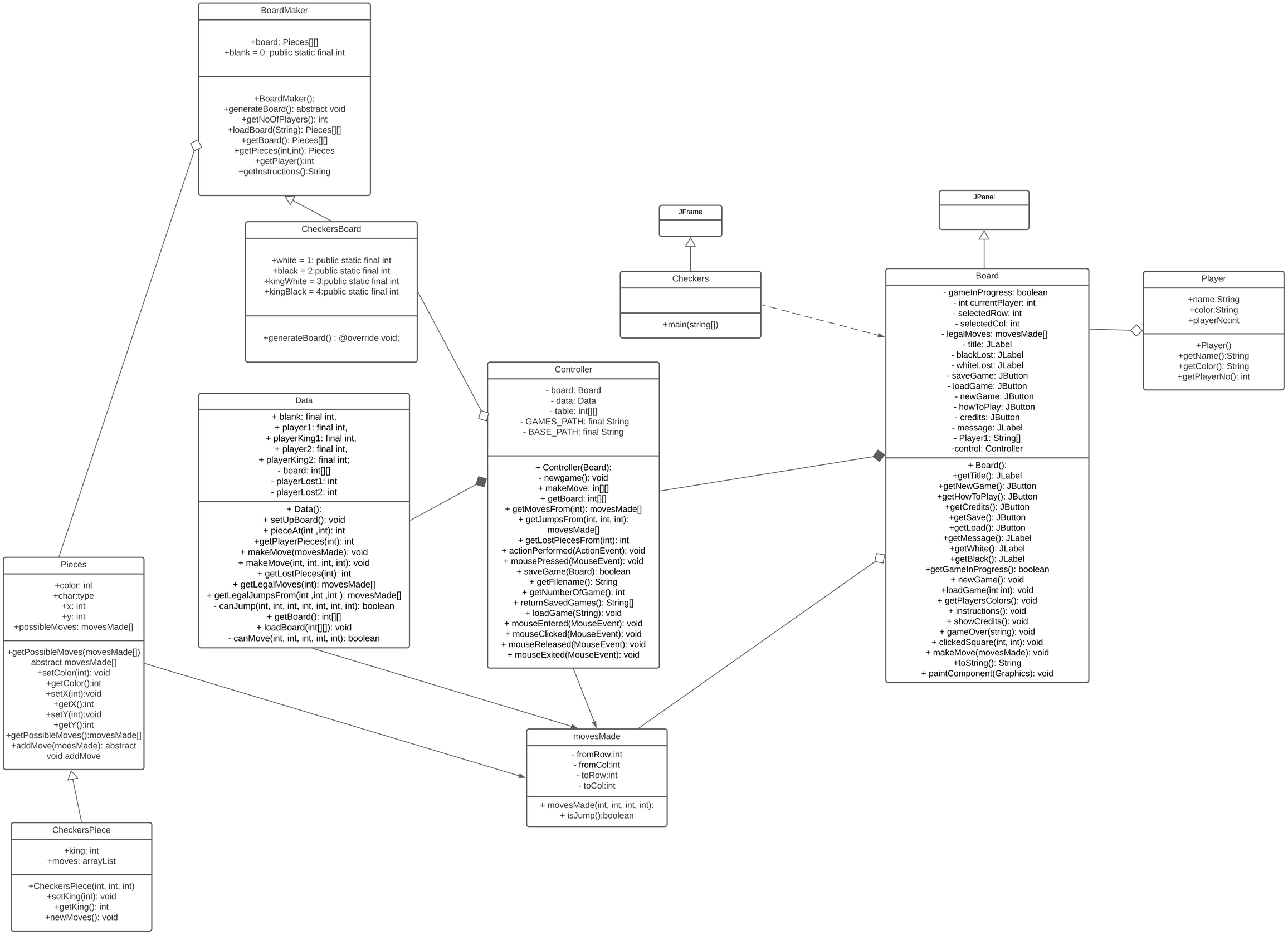
BoardMaker
+board: Pieces[][]
+blank = 0: public static final int
+BoardMaker();
+generateBoard(): abstract void
+getNoOfPlayers(): int
+loadBoard(String): Pieces[][]
+getBoard(): Pieces[][]
+getPieces(int,int): Pieces
+getPlayer():int
+getInstructions():String
CheckersBoard
+white = 1: public static final int
+black = 2:public static final int
+kingWhite = 3:public static final int
+kingBlack = 4:public static final int
+generateBoard() : @override void;
JFrame
Checkers
+main(string[])
JPanel
Board
- gameInProgress: boolean
- int currentPlayer: int
- selectedRow: int
- selectedCol: int
- legalMoves: movesMade[]
- title: JLabel
- blackLost: JLabel
- whiteLost: JLabel
- saveGame: JButton
- loadGame: JButton
- newGame: JButton
- howToPlay: JButton
- credits: JButton
- message: JLabel
- Player1: String[]
-control: Controller
+ Board():
+getTitle(): JLabel
+getNewGame(): JButton
+getHowToPlay(): JButton
+getCredits(): JButton
+getSave(): JButton
+getLoad(): JButton
+getMessage(): JLabel
+getWhite(): JLabel
+getBlack(): JLabel
+getGameInProgress(): boolean
+ newGame(): void
+loadGame(int int): void
+ getPlayersColors(): void
+ instructions(): void
+ showCredits(): void
+ gameOver(string): void
+ clickedSquare(int, int): void
+ makeMove(movesMade): void
+toString(): String
+ paintComponent(Graphics): void
Player
+name:String
+color:String
+playerNo:int
+Player()
+getName():String
+getColor(): String
+getPlayerNo(): int
Controller
- board: Board
- data: Data
- table: int[][]
- GAMES_PATH: final String
- BASE_PATH: final String
+ Controller(Board):
- newgame(): void
+ makeMove: in[][]
+ getBoard: int[][]
+ getMovesFrom(int): movesMade[]
+ getJumpsFrom(int, int, int):
movesMade[]
+ getLostPiecesFrom(int): int
+ actionPerformed(ActionEvent): void
+ mousePressed(MouseEvent): void
+ saveGame(Board): boolean
+ getFilename(): String
+ getNumberOfGame(): int
+ returnSavedGames(): String[]
+ loadGame(String): void
+ mouseEntered(MouseEvent): void
+ mouseClicked(MouseEvent): void
+ mouseReleased(MouseEvent): void
+ mouseExited(MouseEvent): void
Data
+ blank: final int,
+ player1: final int,
+ playerKing1: final int,
+ player2: final int,
+ playerKing2: final int;
- board: int[][]
- playerLost1: int
- playerLost2: int
+ Data():
+ setUpBoard(): void
+ pieceAt(int ,int): int
+getPlayerPieces(int): int
+ makeMove(movesMade): void
+ makeMove(int, int, int, int): void
+ getLostPieces(int): int
+ getLegalMoves(int): movesMade[]
+ getLegalJumpsFrom(int ,int ,int): movesMade[]
- canJump(int, int, int, int, int, int): boolean
+ getBoard(): int[][]
+ loadBoard(int[][]): void
- canMove(int, int, int, int, int): boolean
Pieces
+color: int
+char:type
+x: int
+y: int
+possibleMoves: movesMade[]
+getPossibleMoves(movesMade[])
abstract movesMade[]
+setColor(int): void
+getColor():int
+setX(int):void
+getX():int
+setY(int):void
+getY():int
+getPossibleMoves():movesMade[]
+addMove(moesMade): abstract
void addMove
movesMade
- fromRow:int
- fromCol:int
- toRow:int
- toCol:int
+ movesMade(int, int, int, int):
+ isJump():boolean
CheckersPiece
+king: int
+moves: arrayList
+CheckersPiece(int, int, int)
+setKing(int): void
+getKing(): int
+newMoves(): void