Indy Chess

Developed by

Kongtap Arunlakvilart 6030033321

Jiraphat Khupanit 6030087221

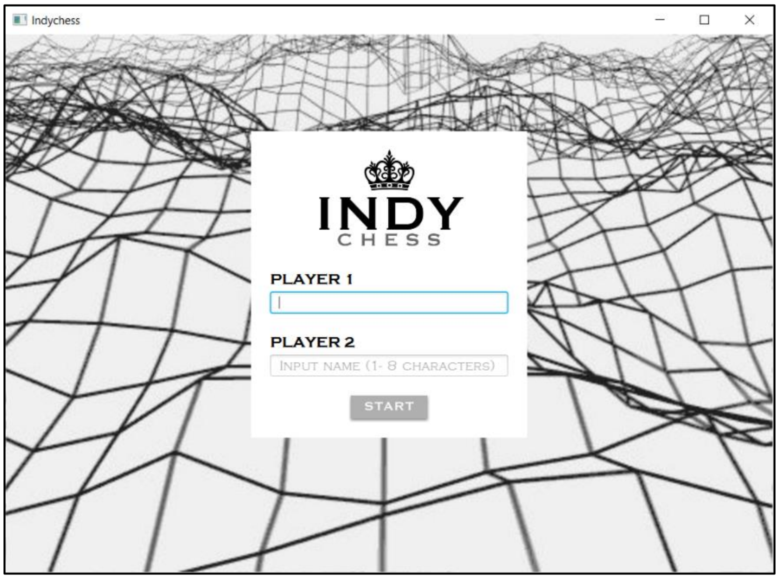
Subject

2110215 Programming Methodology (2018/1)

INDY CHESS

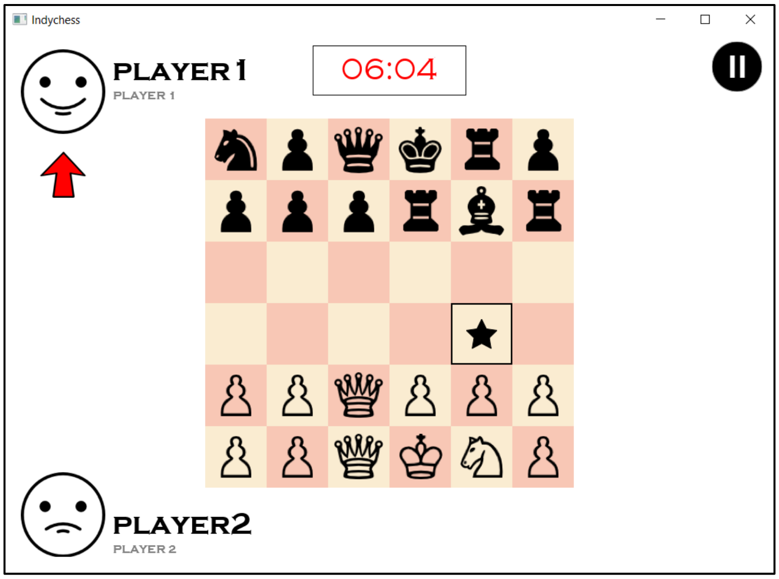
Chess is a two-player strategy board game played on a chessboard, a checkered gameboard with 64 squares arranged in an 8×8 grid. A play does not involve hidden information. Each player begins with 16 pieces: one king, one queen, two rooks, two knights, two bishops, and eight pawns. Each of the six-piece types moves differently. The objective is to checkmate the opponent's king by placing it under an inescapable threat of capture. But simple chess is repeatedly and take a long time. So, we proudly present Indy Chess.

Indy Chess is like simple chess that checkmate the opponent's king to win. But Indy Chess’s board is 6×6 grid which smaller than simple chess’s board that make it spend less time. Other difference is Indy Chess start with 11 randomly chess piece and only one King it make Indy Chess different in every game.



Login Screen

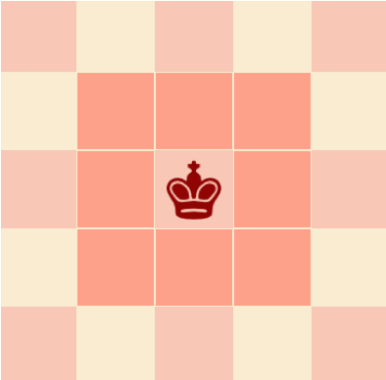
- Players need to input their name.
- Length of player’s name is 1-8 characters only.
- Player’s name should contain only a-z, A-Z, 0-9, _, -.
- click START button to start game.



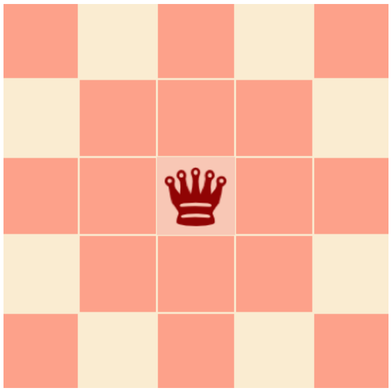
Game screen

- Two players start with 12 chess pieces each player with random piece type but only one king is fix position.
- The winner is who can capture opponent's king first.
- Left side display player's names and random player's emoji.
- Top center displayer timer. Each turn player has 30 second before change to opponent turn.
- Top right display pause button that direction to pause screen when click.

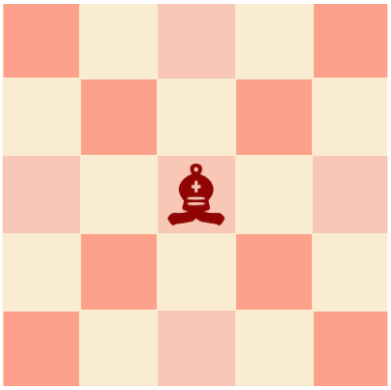
Movement of chess piece



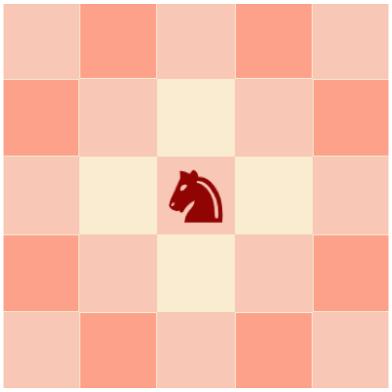
The king moves one square in any direction.



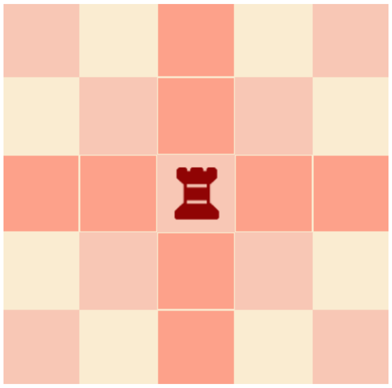
The queen combines the power of a rook and bishop and can move any number of squares along a rank, file, or diagonal, but cannot leap over other pieces.



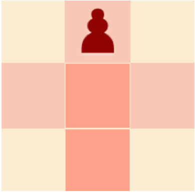
The bishop can move any number of squares diagonally, but cannot leap over other pieces.

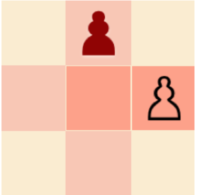


The knight moves to any of the closest squares that are not on the same rank, file, or diagonal, thus the move forms an "L"-shape: two squares vertically and one square horizontally, or two squares horizontally and one square vertically. The knight is the only piece that can leap over other pieces.



The rook can move any number of squares along a rank or file, but cannot leap over other pieces.





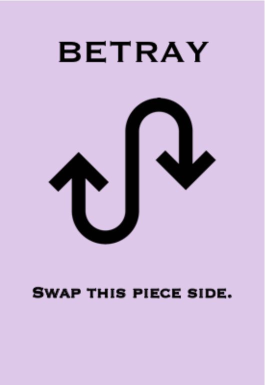
The pawn can move forward to the unoccupied square immediately in front of it on the same file, or on its first move it can advance two squares along the same file, provided both squares are unoccupied ; or the pawn can capture an opponent's piece on a square diagonally in front of it on an adjacent file, by moving to that square. A pawn has two special moves: the en passant capture and promotion.

Star Event

When some chess piece capture star (king can’t capture star), game will have random event. There are 5 events in this game.

CHANGE
CHANGE THIS PIECE TO “QUEEN”

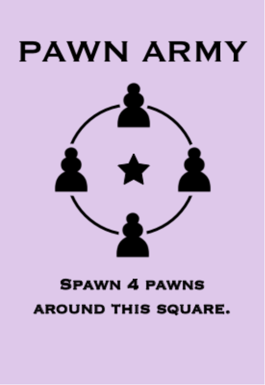
Change the chess piece that have captured star to queen, bishop, knight or rook randomly.



Change this chess piece’s side to opponent side.

SKIP
SKIP OPPONENT'S TURN.

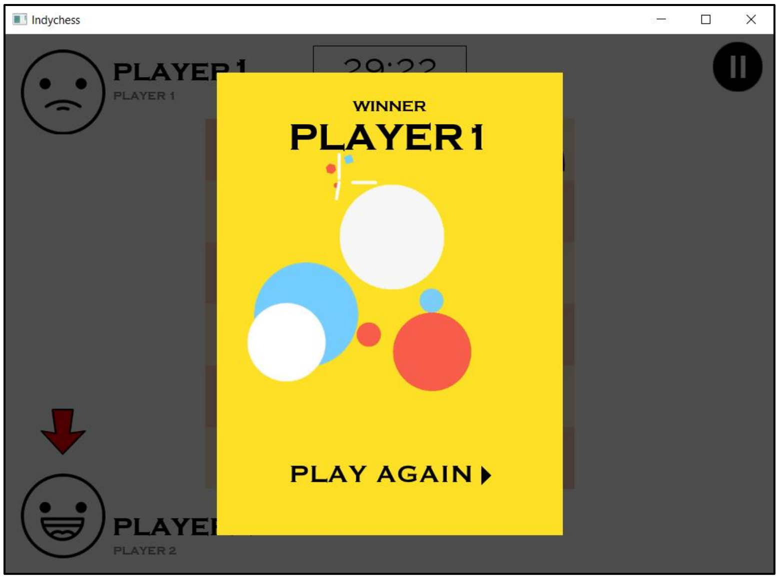
Skip opponent’s turn.



Spawn 4 pawns around star that don’t have any chess piece.

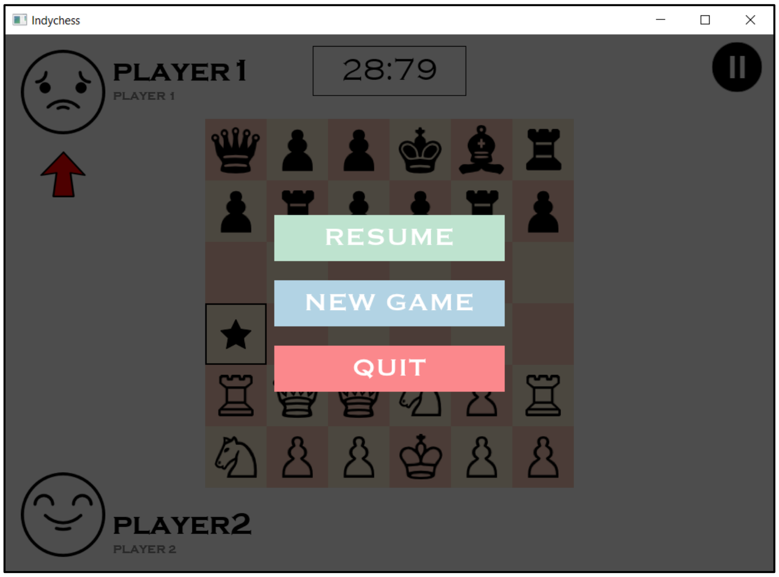
BOMB
REMOVE THIS PIECE.

Remove the chess piece that capture star.



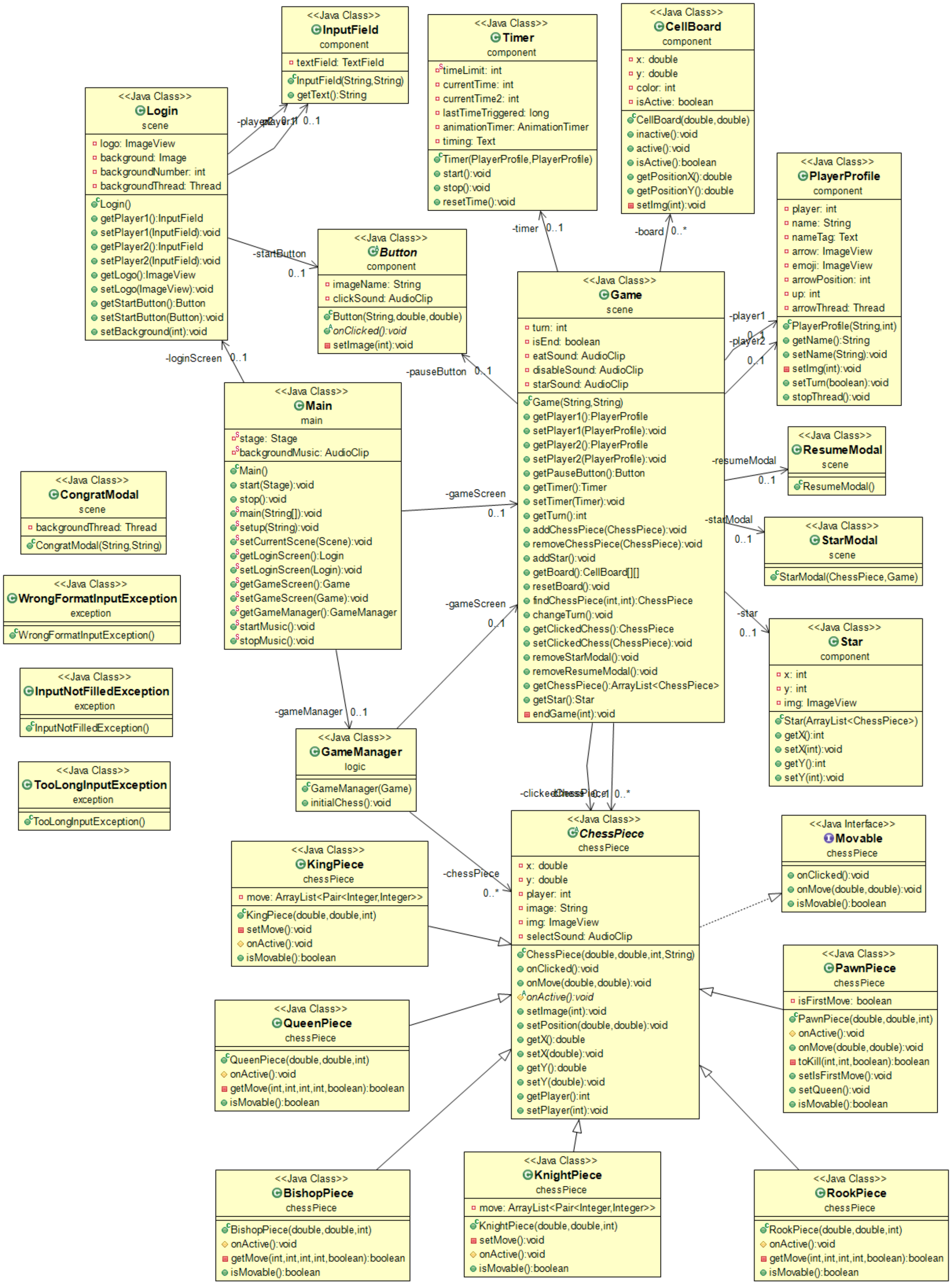
Win screen

When king was capture program will show win screen, click play again to return to login.



Pause screen

When click pause on top right of game screen, program will show pause screen. Click resume to continue playing. Click new game to return to login screen. Click quit to exit program.



An UML Diagram of Project

Implementation Details

1. Package main

1.1 Class Main extends Application

1.1.1 Fields

private static Stage stage	Stage.
private static Login loginScreen	Login screen.
private static Game gameScreen;	Game screen.
private static GameManager gameManager;	Game manager.
private static AudioClip backgroundMusic;	Background music.

1.1.2 Method

public void start(Stage primaryStage)	The main entry point for the JavaFX applications. Load background music from "sound/background.mp3", set this music loop continuously until stopped, set volume to 0.2. Setup login page. Set title “IndyChess”.
public void stop() throws Exception	This method is going to execute before JavaFX application terminates.
public static void main(String[] args)	An entry point of the application.
public static void setup(String managerType)	Setup the scene.
public static void setCurrentScene(Scene scene)	Set the current scene.
public static Login getLoginScreen()	Get the login screen.
public static void setLoginScreen(Login loginScreen)	Set the login screen.
public static Game getGameScreen()	Get the game screen.
public static void setGameScreen(Game gameScreen)	Set the game screen.
public static GameManager getGameManager()	Get the game manager.
public static void startMusic()	Play background music.
public static void stopMusic()	Stop background music.

2. Package component

2.1 Class Button extends ImageView

2.1.1 Fields

private String imageName	Image of button's address.
private AudioClip clickSound	Sound when click this button.

2.1.2 Constructor

public Button(String imageName, double x, double y)	Sets the value of the property layoutX with x. Sets the value of the property layoutY with y. Set sound when mouse click.

2.1.3 Method

public abstract void onClicked()	Do something when click this button.
private void setImage(int type)	Set image of button.

2.2 Class CellBoard extends ImageView

2.2.1 Fields

private double x	X position.
private double y	Y position.
private int color	Color of cell board.
private boolean isActive;	Is cell board active?

2.2.2 Constructor

public CellBoard(double x, double y)	Initializes color of cell board. Initializes position of cell board.

2.2.3 Method

public void inactive()	Inactive this cell board.
public void active()	Active this cell board.
public boolean isActive()	Is this cell board active?
public double getPositionX()	Get x position of this cell board.
public double getPositionY()	Get y position of this cell board.
private void setImg(int type)	Set image of this cell board.

2.3 Class InputField extends VBox

2.3.1 Fields

private TextField textField	Text field.

2.3.2 Constructor

public InputField(String title, String promptText)	Sets the value of the property padding to 5. Sets the value of the property spacing to 5. Initializes label. Initializes text field.

2.3.3 Method

public String getText()	Get this text field.

2.4 Class PlayerProfile extends Pane

2.4.1 Fields

private int player	Player’s number (1 or 2).
private String name	Player’s name.
private Text nameTag	Text of name
private ImageView arrow	Arrow that point to player who can play.

private ImageView emoji	Player’s emoji.
private int arrowPosition	Position of arrow.
private int up	Arrow is up or down.
private Thread arrowThread	Arrow’s thread.

2.4.2 Constructor

public PlayerProfile(String name, int player)	Initializes player’s number. Initializes name. Initializes random Player’s emoji. Initializes nametag. Initializes arrow.

2.4.3 Method

public String getName()	Get player’s name.
public void setName(String name)	Set player’s name.
private void setImg(int type)	Set arrow and emoji of player.
public void setTurn(boolean isPlaying)	Set visible of arrow.
public void stopThread()	Stop thread.

2.5 Class Star extends Pane

2.5.1 Fields

private int x	X position of this star.
private int y	Y position of this star.
private ImageView img	Image of this star.

2.5.2 Constructor

public Star(ArrayList<ChessPiece> chessPiece)	Initializes random position of star to cell board that don't have chess piece. Initializes image of star.

2.5.3 Method

public int getX()	Get X position of this star.
public void setX(int x)	Set X position of this star.
public int getY()	Get Y position of this star.
public void setY(int y)	Set Y position of this star.

2.6 Class Timer extends HBox

2.6.1 Fields

private static int timeLimit	Time limit of each turn is 30 seconds.
private int currentTime	Current time (second).
private int currentTime2	Current time (1/100 second).
private long lastTimeTriggered	Last time triggered.
private AnimationTimer animationTimer	Animation of timer.
private Text timing	Text of timer.

2.6.2 Constructor

public Timer(PlayerProfile player1, PlayerProfile player2)	Set height to 65, width 200. Start with 30 seconds. If less than 0 change player turn. Set text's color to black. When time less than 10 second, change text's color to red. And when time less than 0, change turn.

2.6.3 Method

public void start()	Start timing.
public void stop()	Stop timing.
public void resetTime()	Reset time to 30 seconds.

3. Package chessPiece

3.1 Class ChessPiece extends Pane implements Movable

3.1.1 Fields

private double x	X position of chess piece.
private double y	Y position of chess piece.
private int player	Owner of this chess piece.
private String image	Name of this chess piece’s image.
private ImageView img	Image of this chess piece.
private AudioClip selectSound	Sound when select this chess piece.

3.1.2 Constructor

public ChessPiece(double x, double y, int player, String image)	Initializes X position. Initializes Y position. Initializes image. Initializes owner.

3.1.3 Method

public void onClicked()	When click this chess piece will change color and play select sound.
public void onMove(double x, double y)	Move this chess piece to x and y position.
protected abstract void onActive()	Show the path that this chess piece can move.
public void setImage(int type)	Set image of this chess piece.
public void setPosition(double x, double y)	Set new position of this chess piece.

public double getX()	Get x position.
public void setX(double x)	Set x position.
public double getY()	Get y position.
public void setY(double y)	Set y position.
public int getPlayer()	Get owner of this chess piece.
public void setPlayer(int player)	Set owner of this chess piece.

3.2 Interface Movable

3.2.1 Method

void onClicked()	Do something when click this chess piece.
void onMove(double x, double y)	Move this chess piece.
boolean isMovable();	Can this chess piece move?

3.3 Class BishopPiece extends ChessPiece

3.3.1 Constructor

public BishopPiece(double x, double y, int player)	Initializes X position. Initializes Y position. Initializes owner. The bishop can move any number of squares diagonally, but cannot leap over other pieces.

3.3.2 Method

protected void onActive()	Show the path that bishop can move to.
private boolean getMove(int x, int y, int addX, int addY, boolean isClicked)	Check the path that bishop can move to.
public boolean isMovable()	Can bishop move?

3.4 Class KingPiece extends ChessPiece

3.4.1 Fields

private ArrayList<Pair<Integer, Integer>> move	ArrayList that contain path to move king.

3.4.2 Constructor

public KingPiece(double x, double y, int player)	Initializes X position. Initializes Y position. Initializes owner. The king moves one square in any direction.

3.4.3 Method

private void setMove()	Set path that king can move to
protected void onActive()	Show path that king can move to.
public boolean isMovable()	Can king move?

3.5 Class KnightPiece extends ChessPiece

3.5.1 Fields

private ArrayList<Pair<Integer, Integer>> move	ArrayList that contain path to move knight.

3.5.2 Constructor

public KnightPiece(double x, double y, int player)	Initializes X position. Initializes Y position. Initializes owner. The knight moves to any of the closest squares that are not on the same rank, file, or diagonal, thus the move forms an "L"-shape: two squares vertically and one square horizontally, or two squares horizontally

	and one square vertically. The knight is the only piece that can leap over other pieces.

3.5.3 Method

private void setMove()	Set path that knight can move to.
protected void onActive()	Show path that knight can move to.
public boolean isMovable()	Can knight move?

3.6 Class PawnPiece extends ChessPiece

3.6.1 Fields

private boolean isFirstMove	Have this pawn move before?
private boolean isQueen	Have this pawn promotion yet?

3.6.2 Constructor

public PawnPiece(double x, double y, int player)	Initializes X position. Initializes Y position. Initializes owner. The pawn can move forward to the unoccupied square immediately in front of it on the same file, or on its first move it can advance two squares along the same file, provided both squares are unoccupied ; or the pawn can capture an opponent's piece on a square diagonally in front of it on an adjacent file, by moving to that square. A pawn has two special moves: the en passant capture and promotion.

3.6.3 Method

protected void onActive()	Set path that pawn can move to.
public void onMove(double x, double y)	Move pawn to x and y position.
private boolean toKill(int x, int y, boolean isClicked)	Is this pawn can capture opponent's chess piece?
public void setIsFirstMove()	Set isFirstMove to false.
public void setQueen()	Promote this pawn.
public boolean isMovable()	Can pawn move?

3.7 Class QueenPiece extends ChessPiece

3.7.1 Constructor

public QueenPiece(double x, double y, int player)	Initializes X position. Initializes Y position. Initializes owner. The queen combines the power of a rook and bishop and can move any number of squares along a rank, file, or diagonal, but cannot leap over other pieces.

3.7.2 Method

protected void onActive()	Set path that queen can move to.
private boolean getMove(int x, int y, int addX, int addY, boolean isClicked)	Check the path that queen can move to.

3.8 Class RookPiece extends ChessPiece

3.8.1 Constructor

public RookPiece(double x, double y, int player)	Initializes X position. Initializes Y position. Initializes owner.

	The rook can move any number of squares along a rank or file, but cannot leap over other pieces. Along with the king, a rook is involved during the king's castling move.

3.8.2 Method

protected void onActive()	Set path that rook can move to.
private boolean getMove(int x, int y, int addX, int addY, boolean isClicked)	Check the path that rook can move to.
public boolean isMovable()	Can rook move?

4. Package Exception

4.1 Class InputNotFilledException extends Exception

4.1.1 Constructor

public InputNotFilledException()	This exception use when user haven't input yet. Print "Player Name is empty". Show Alert "Please Fill Player Name".

4.2 Class TooLongInputException extends Exception

4.2.1 Constructor

public TooLongInputException()	This exception use when user input too long. Print "Player Name is too long". Show Alert "Please Fill Player Name in correct format".

4.3 Class WrongFormatInputException extends Exception

4.3.1 Constructor

public WrongFormatInputException()	This exception use when user input wrong format. Print "Player name isn't correct". Show Alert "Please Fill Player Name in correct format".

5. Package logic

5.1 Class GameManager

5.1.1 Fields

private ArrayList<ChessPiece> chessPiece	Contain all chess piece on board.
private Game gameScreen	Game screen.

5.1.2 Constructor

public GameManager(Game gameScreen)	Initializes chess piece. Initializes star.

5.1.3 Method

public void initialChess()	Initializes 12 chess piece per player. King have only one per player and fix position. Pawn have change to spawn 1/2. Other is 1/10.

6. Package scene

6.1 Class CongratModal extends Pane

6.1.1 Fields

private Thread backgroundThread	Background thread.

6.1.2 Constructor

public CongratModal(String winnerName, String status)	Set height to 700 and width 1000. Stop background music and play congratulation music. Change background image continuously. Initializes play again button to start new game.

6.2 Class Game extends Pane

6.2.1 Fields

private PlayerProfile player1	Player 1.
private PlayerProfile player2	Player 2.
private Timer timer	Timer.
private Button pauseButton	Pause button.
private ResumeModal resumeModal	Resume modal.
private StarModal starModal	Event when capture star.
private ArrayList<ChessPiece> chessPiece	Contain all chess piece on board.
private CellBoard[][] board	Array that contain cell boards.
private ChessPiece clickedChess	Chess piece that user have select.
private int turn	Turn of player
private Star star	Star.
private boolean isEnd	Is this game end?
private AudioClip eatSound	Sound that play when capture.
private AudioClip disableSound	Sound that play when wrong select.
private AudioClip starSound	Sound that play when capture star.

6.2.2 Constructor

public Game(String player1Name, String player2Name)	Initializes cell board. Initializes pause button. Initializes profile of player 1 and player 2. Initializes timer. Initializes sound when click. - If some chess piece was capture, play eatSound. - If user select wrong, play disableSound. - If chess piece was capture star, play starSound.

6.2.3 Method

public PlayerProfile getPlayer1()	Get player 1 profile.
public void setPlayer1(PlayerProfile player1)	Set player 1 profile.
public PlayerProfile getPlayer2()	Get player 2 profile.
public void setPlayer2(PlayerProfile player2)	Set player 2 profile.
public Button getPauseButton()	Get pause button.
public Timer getTimer()	Get timer.
public void setTimer(Timer timer)	Set timer.
public int getTurn()	Get turn.
public void addChessPiece(ChessPiece e)	Add chess piece to chess board.
public void removeChessPiece(ChessPiece e)	Remove chess piece on chess board.
public void addStar()	Add star to chess board.
public CellBoard[][] getBoard()	Get array of chess piece on board.
public void resetBoard()	Reset chess board to empty.
public ChessPiece findChessPiece(int x, int y)	Get chess piece on x and y position.
public void changeTurn()	Swap to opponent turn.
public ChessPiece getClickedChess()	Get chess piece that have selected.
public void setClickedChess(ChessPiece clickedChess)	Set chess piece that have selected.
public void removeStarModal()	Return from star event to chess game.

public void removeResumeModal()	Return from pause screen to chess game.
public ArrayList<ChessPiece> getChessPiece()	Get ArrayList that contain chess piece.
public Star getStar()	Get star.
private void endGame(int type)	End of game. - If type is 1, player 1 win. - If type is 2, player 2 win. - If type is 3, draw.

6.3 Class Login extends VBox

6.3.1 Fields

private InputField player1	Input field of player 1 name.
private InputField player2	Input field of player 2 name.
private ImageView logo	Image of Indy Chess's logo.
private Image background	Background image.
private Button startButton	Start button.
private int backgroundNumber	Number of background image.
private Thread backgroundThread	Background thread.

6.3.2 Constructor

public Login()	Initializes Indy Chess's logo. Initializes input fields. Initializes start button that start game when click and throw exception when wrong input player name. Initializes timer.

6.3.3 Method

public InputField getPlayer1()	Get player 1 input field.
public void setPlayer1(InputField player1)	Set player 1 input field.

public InputField getPlayer2()	Get player 2 input field.
public void setPlayer2(InputField player2)	Set player 2 input field.
public ImageView getLogo()	Get logo.
public void setLogo(ImageView logo)	Set logo.
public Button getStartButton()	Get start button.
public void setStartButton(Button startButton)	Set start button.
public void setBackground(int number)	Set background image.

6.4 Class ResumeModal extends VBox

6.4.1 Constructor

public ResumeModal()	The screen that show when click pause button in top right of game screen that contain - Resume button that resume game when click. - New game button that end this game and return to login screen when click. - Quit button that end this program when click.

6.5 Class StarModal extends Pane

6.5.1 Constructor

public StarModal(ChessPiece clickedChess, Game gameScreen)	The screen that show star's event when some chess piece capture star. In this screen show - Event's name. - Event's icon. - Event's effect. - Ok button (continue game).