Program description

Diagram 1: The first launch screen which let the user to choose game mode. User can select either 2 players, 3 players or 4 players mode.

Diagram 2: After user choose 4 players mode, it will enter to the screen which let the user to enter players' name based on number of players.
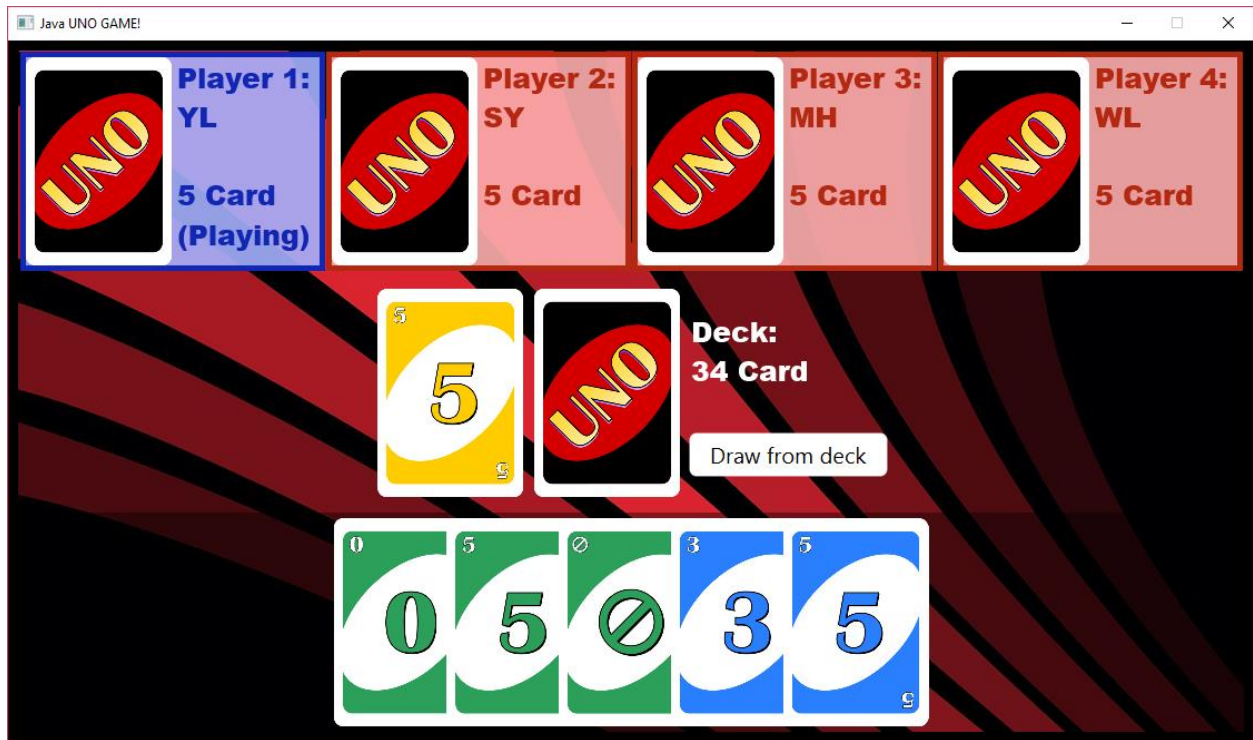
Diagram 3: The most upper row shows the players' total hand cards and players' name. The blue color area at player 1 shows that the current player is playing whereas the others are in red color shows they are waiting for their turn. The middle row shows the pile card and deck cards. The bottom row shows the hand cards of the current player which is player 1.
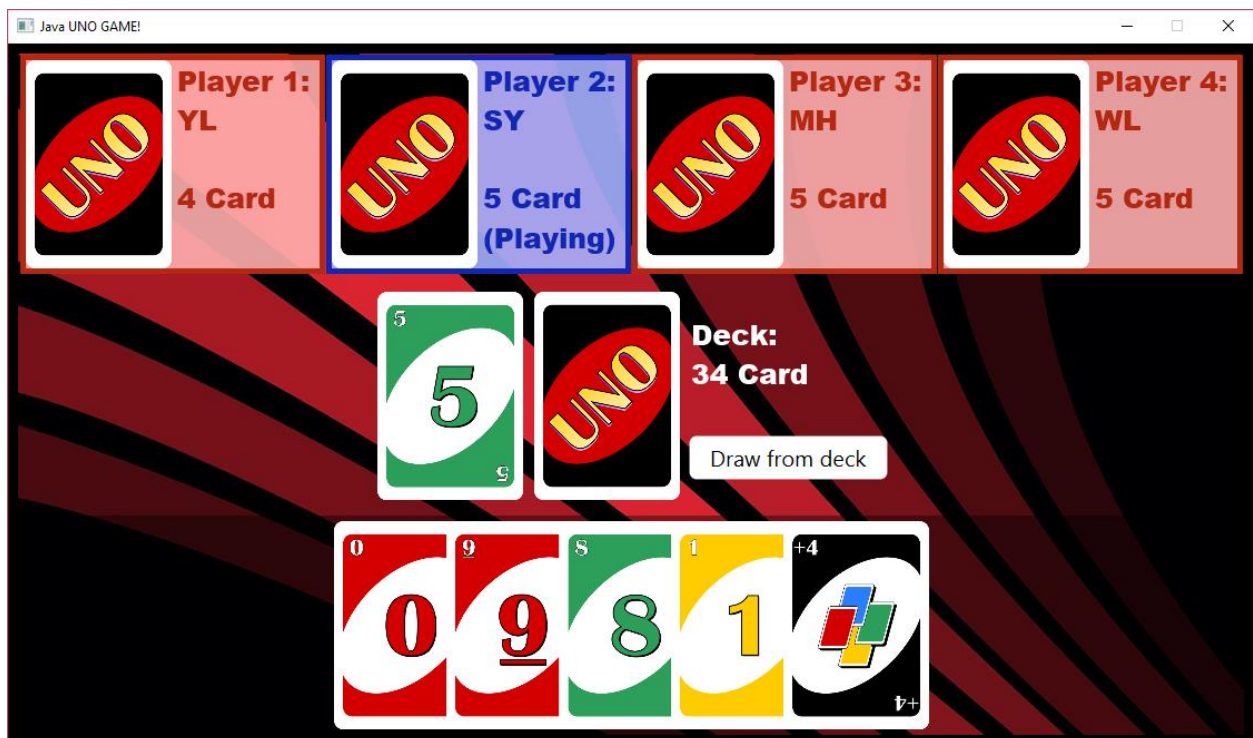

Diagram 4: When player 1 select green 5 card, it will be placed at the pile. The total number of hand card for player 1 will be minus 1. Then, is the turn for the player 2 to discard the card. The bottom row shows the hand card of the player 2. If select a card that not match color or symbol of the pile card will alert error and user need to select again.
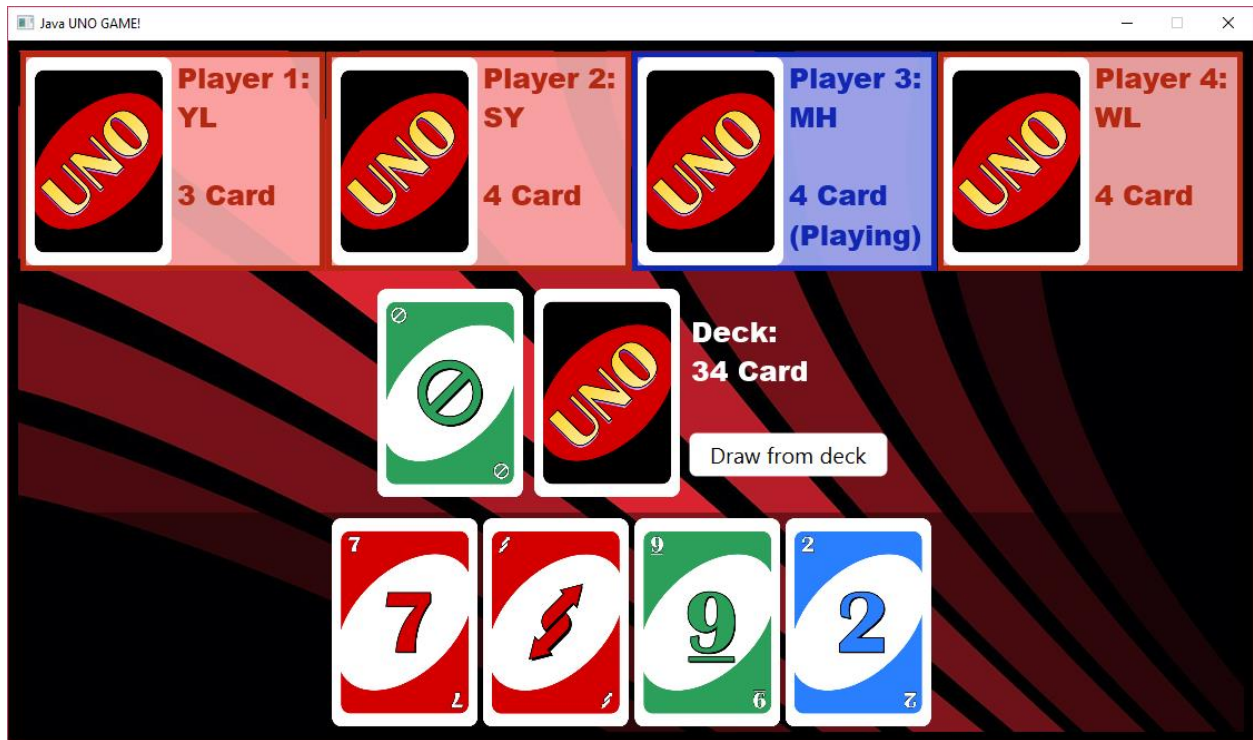
Diagram 5: After the player 1 select skip card. It will skip one player and pass the turn to player 3.

Diagram 6: After player 4 select draw two cards, the result is show as above diagram. Player 1 need to draw 2 cards from deck. The bottom row shows hand cards of player 1 which had added the two drawed card.

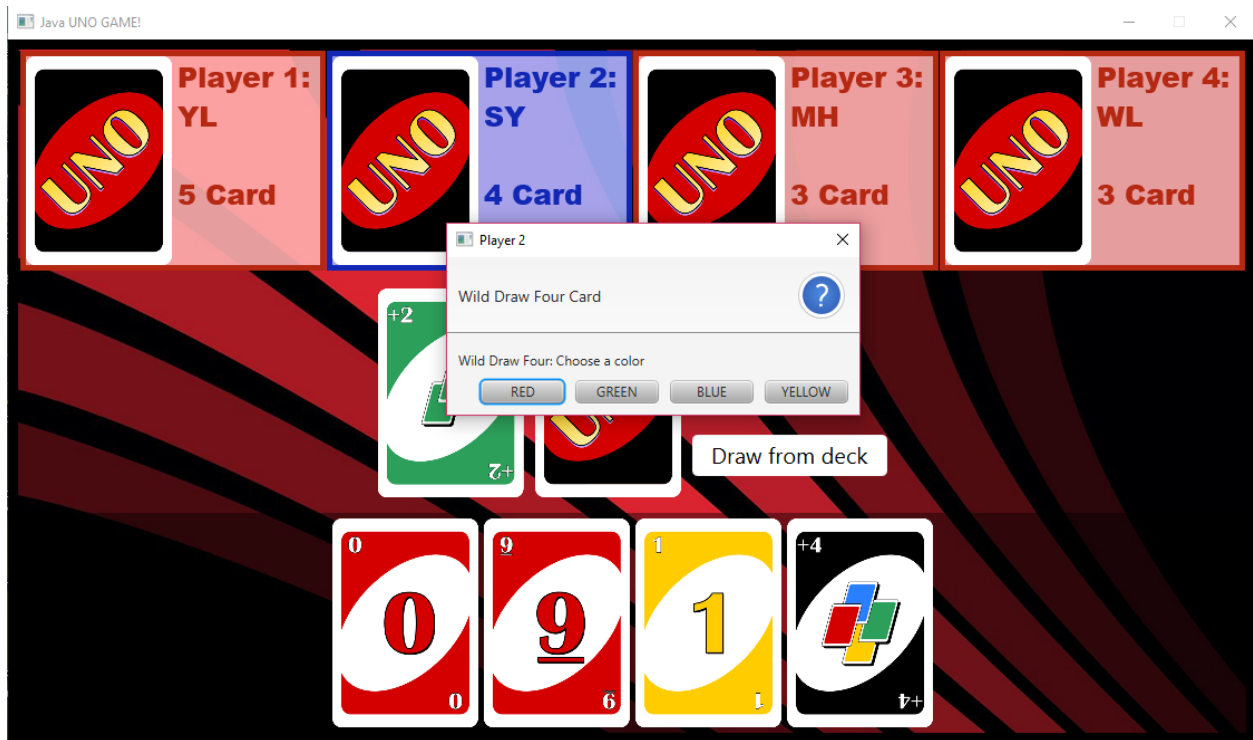Diagram 7: Player 2 does not have playable card, so she only can discard wild draw four card. When she selects the card, she also need to choose one of the colors for the next player to discard based on the color she chosen.
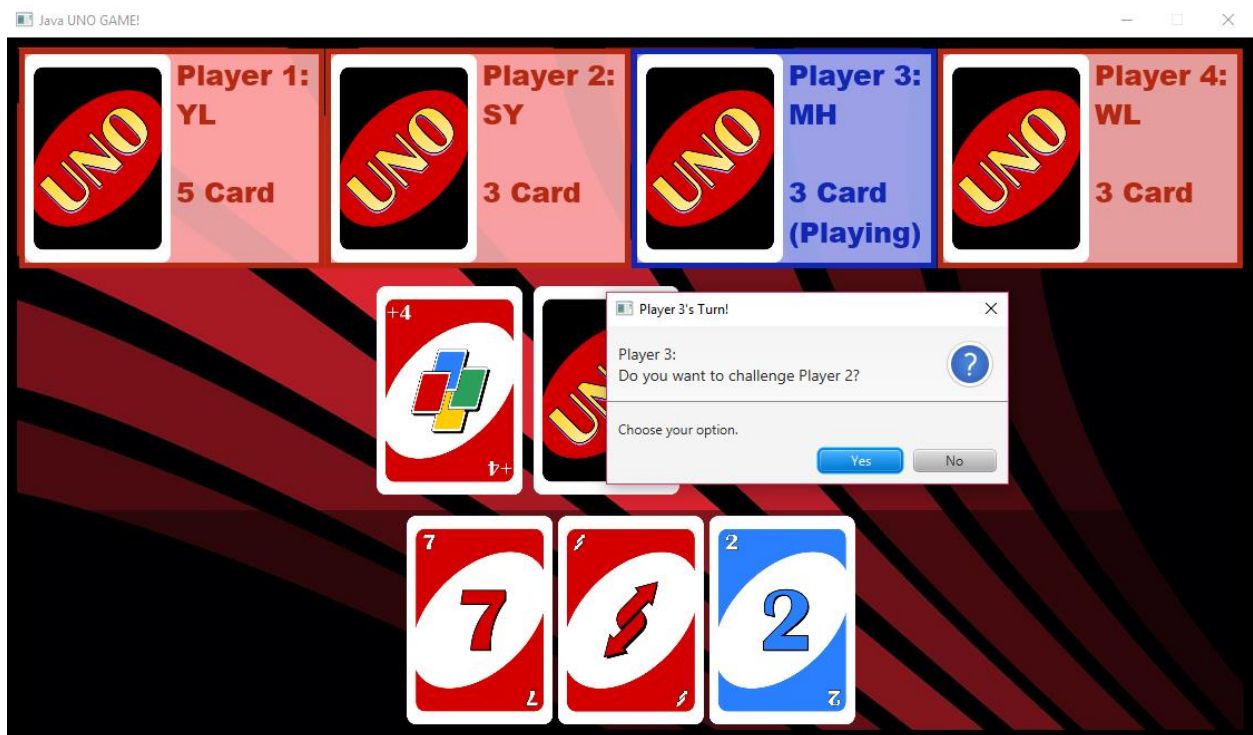
Diagram 8: After player 2 choose red color, the wild draw four card will change to red color and place at the pile. Then, prompt out an alert box to ask player 3 whether want to challenge player 2 or not.

Diagram 9: if player 3 want to challenge player 2, it will prompt out challenge fail because the player 2 does not have playable card. Thus player 3 need to draw 6 cards from the deck. If the player 3 do not want to challenge player 2, player 3 only draws 4 cards from deck.
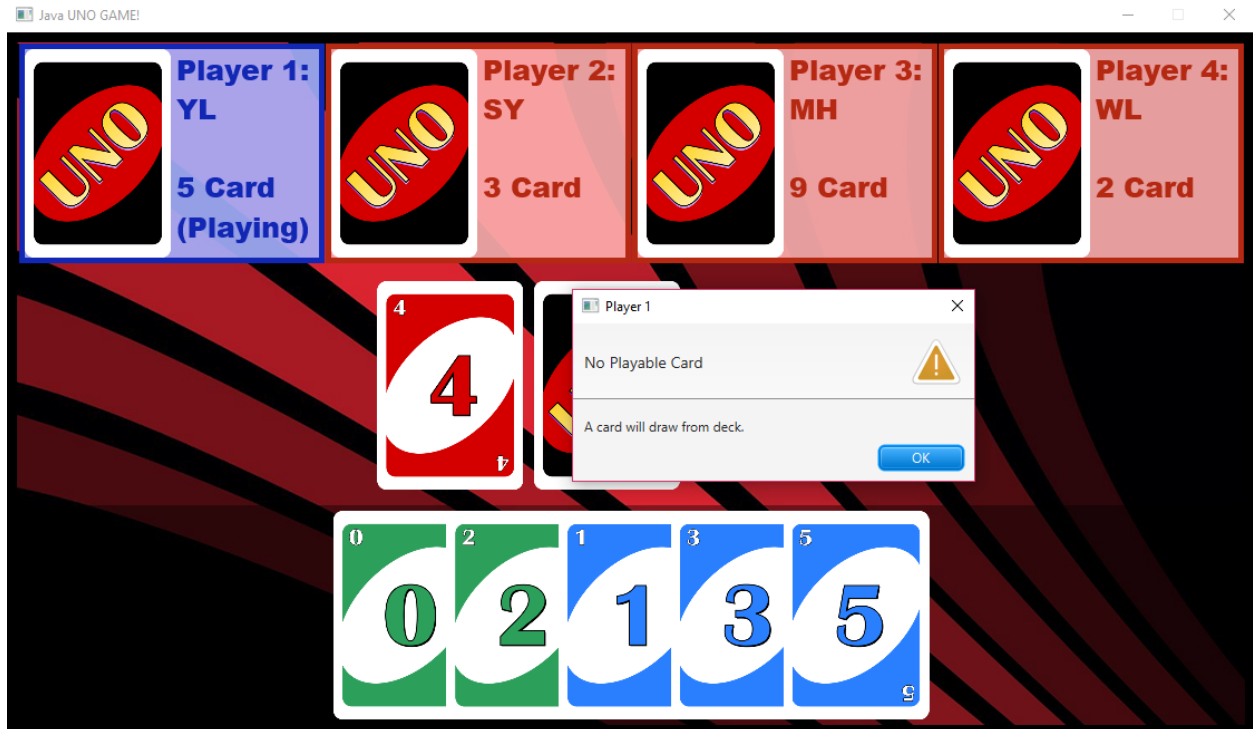
Diagram 10: When player 1 's turn, he does not have playable card. Alert box prompt out to inform player and automatic add a card for him. Then, pass the turn to next player.
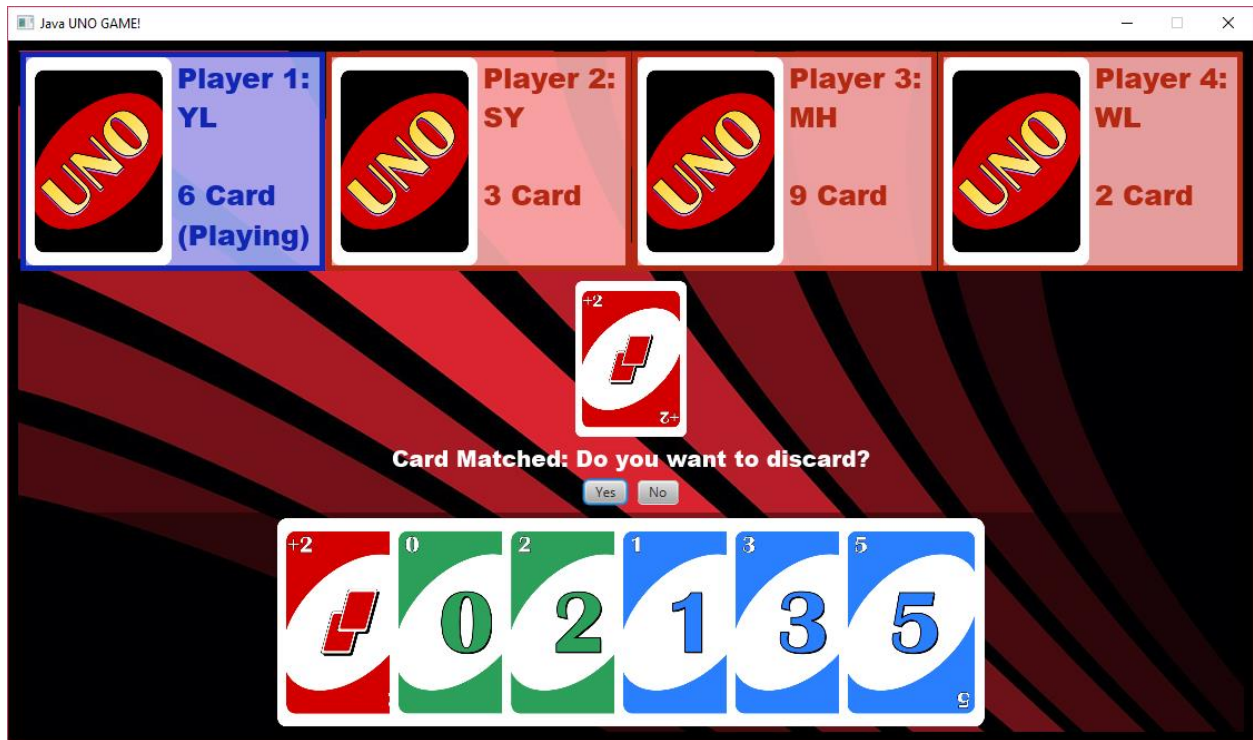
Diagram 11: If the drawed card match the card at the pile. System will ask the player want to discard the card or not.
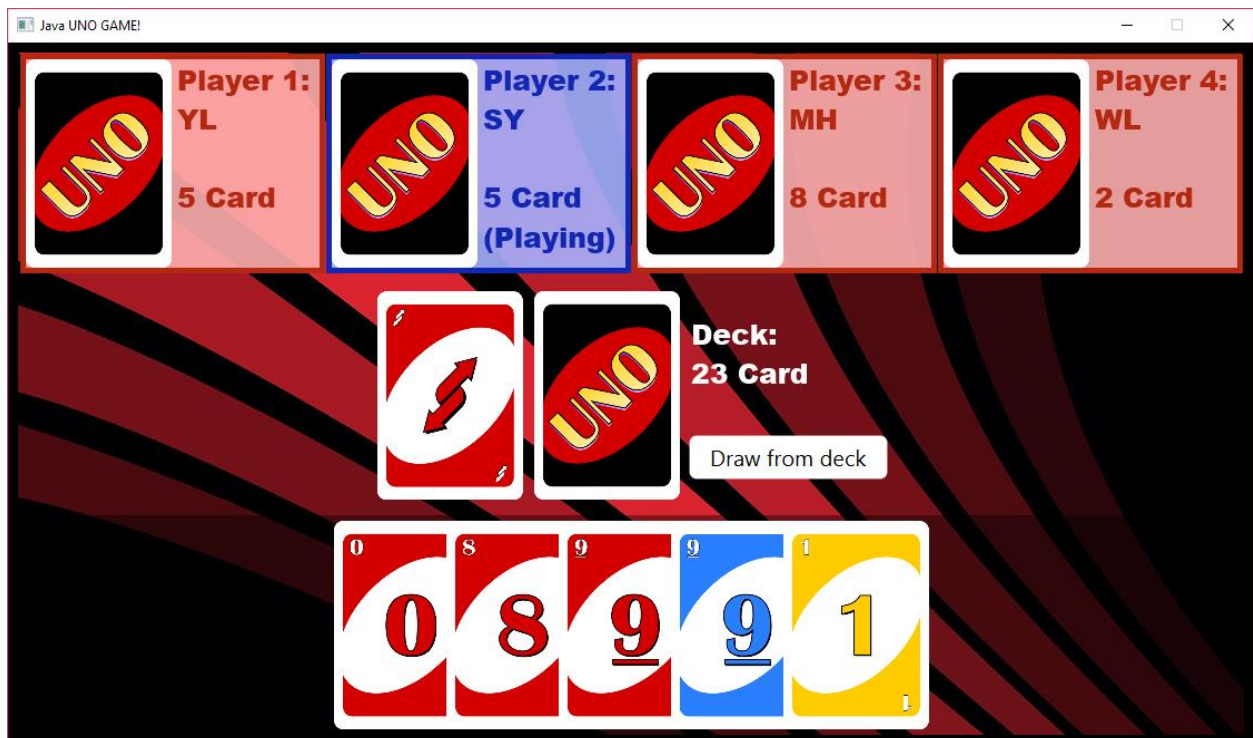
Diagram 12: If player 3 select reverse card, the direction from ascending order will change to descending order. Thus, from player 1 -> player 4 change to player 4 -> player 1.
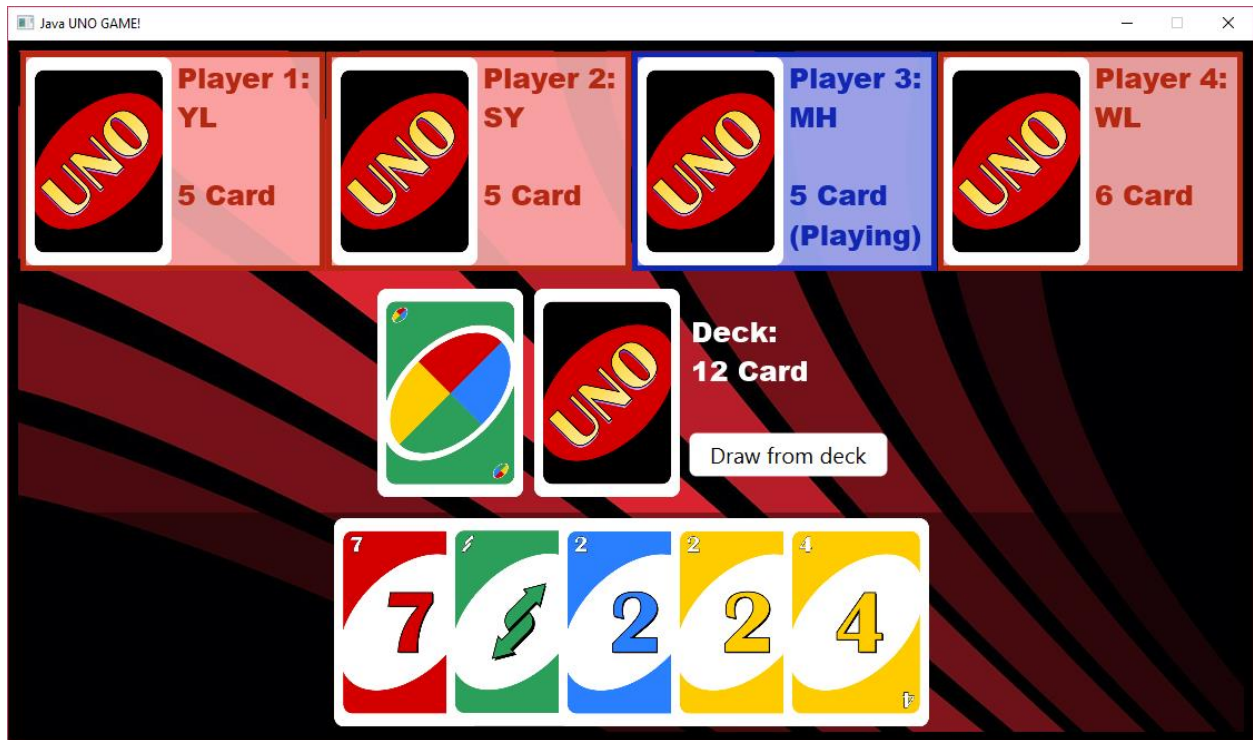
Diagram 13: After the player 4 choose wild card in green color, the wild card will change green color and place at the pile, thus the player 3 need to discard green color card.
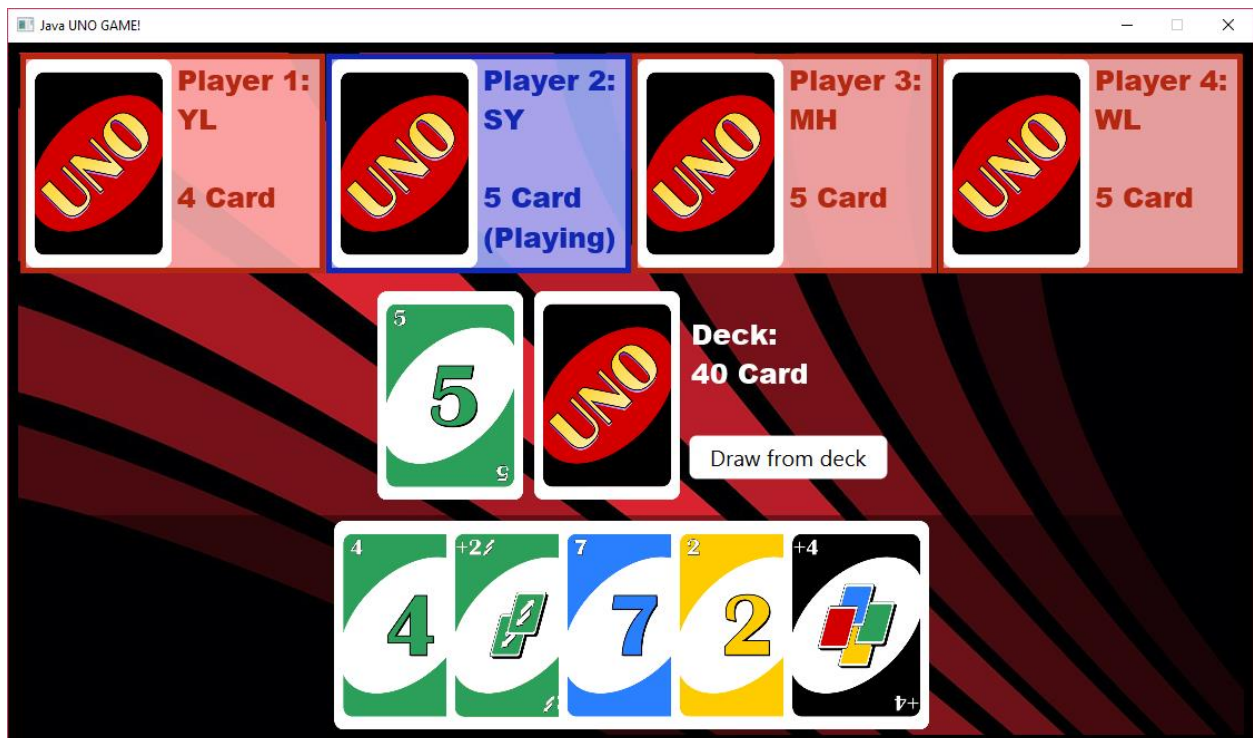
Diagram 14: We have added extra action card which is "draw two reverse" card. We have added 4 draw two reverse card with each color one card. This card's function is change the direction of the turn and add 2 cards to the next player. Thus, the direction in this game is from player 1 -> player 4 then change to player 4 -> player 1 and add 2 cards to the player 1.
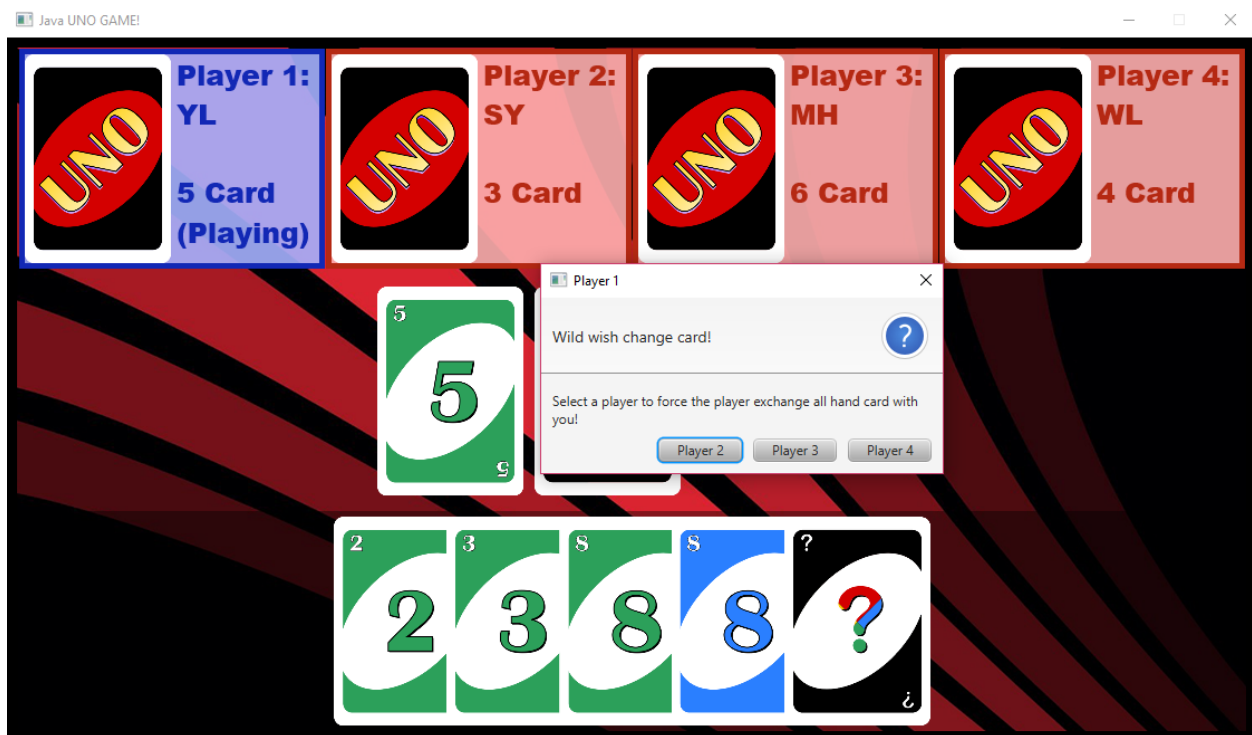
Diagram 15 & 16: We have added extra action card called "wild wish change" card. Only one wild wish change card in the game. This card's function is the player who discard this card can choose one of the player and force the player to exchange all cards with him. Then, the player can choose the color to change the card color at the pile.
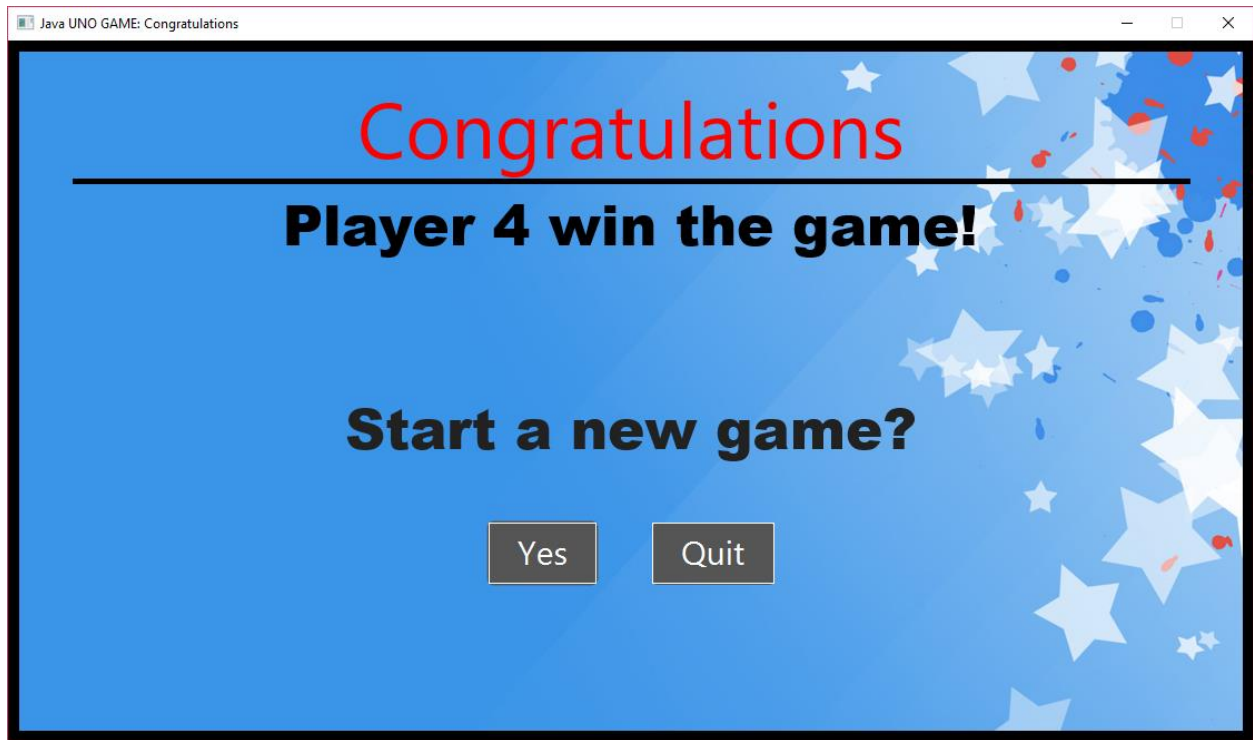
Diagram 17: If one of the player have empty hand card, this screen will be shown and state which player win the game. System will ask player want to start new game or quit.

UML Class Diagram

GameUI
+GameUI()
+start(Stage) : void
+main(String[]) : void
+setGameMode(Stage) : void
+setPlayerName(Stage, int) : void
+startGame(Stage, String[]) : void
-gameEnd(Stage, Game) : void

Game
-fPlayIndex : int
-fAscending : boolean
-fPlayer : ArrayList<Player>
-fDeck : ArrayList<Card>
-fPile : ArrayList<Card>

+Game(String[])
+getPlayer(int) : Player
+getPlayerCount() : int
+getPlayIndex() : int
+nextPlayIndex() : int
+prePlayIndex() : int
+getPlaying() : Player
+getPrePlaying() : Player
+getNextPlaying() : Player
+getPlayingCard() : ArrayList<Card>
+getPlayingCard(int) : Card
+setPlayIndex(int) : void
+nextPlayer() : void
+isAscending() : boolean
+changeDirection() : void
+getFromDeck(int) : Card
+deckCount() : int
+getFromPile(int) : Card
+addToPile(Card) : void
+drawFromDeck() : Card
-createCard() : void
-shuffleCard() : void
-distributeCard(String[]) : void
-actionWild(String String) : int
+actionDraw(int) : int
+actionWishChange() : int
+endGame() : boolean

Player
-fName : String
-numOfPlayer : int
-fHandCard : ArrayList<Card>

+Player(String)
+Player(Player)
+getName() : String
+getCard(int) : Card
+getCard() : ArrayList<Card>
+getNum() : int
+setName(String) : void
+setHandCards(ArrayList<Card>) : void
+toString() : String
+addCard(Card) : void
+removeCard(int) : Card
+removeCard(Card) : int
+isPlayable(Card) : ArrayList<Card>
+win() : boolean

-fPlayer   0..*

-fPile   0..*   -fDeck   0..*

-fHandCard   0..*

<<enumeration>> eSymbol
#ZERO : eSymbol
#ONE : eSymbol
#TWO : eSymbol
#THREE : eSymbol
#FOUR : eSymbol
#FIVE : eSymbol
#SIX : eSymbol
#SEVEN : eSymbol
#EIGHT : eSymbol
#NINE : eSymbol
#SKIP : eSymbol
#REVERSE : eSymbol
#DRAW_2 : eSymbol
#WILD : eSymbol
#WILD_DRAW_4 : eSymbol
#DRAW_2R : eSymbol
#WISH_CHANGE : eSymbol

+eSymbol()

-fSymbol   0..1

Card
-fSymbol : eColor
-fColor : eColor

+Card(int, int)
+Card(Card)
+getColor() : int
+getSymbol() : int
+toString() : String
+compareTo(Card) : int
+getPath() : String
+isMatch(Card) : boolean
+action(Game) : int

-fColor   0..1

<<enumeration>> eColor
#RED : eColor
#GREEN : eColor
#BLUE : eColor
#YELLOW : eColor
#WILD : eColor

+eColor()

0..1

-fSelectedColor

SkipCard
+SkipCard(int)
+action(Game) : int

ReverseCard
+ReverseCard(int)
+action(Game) : (int)

DrawTwoReverseCard
+DrawTwoReverseCard(int)
+action(Game) : int

DrawTwoCard
+DrawTwoCard(int)
+action(Game) : int

WildCard
-fSelectedColor : eColor

+WlidCard(int)
+setSelectedColor(int) : void
+getSelectedColor() : int
+action(Game) : int
+isMatch(Card) : boolean
+toString() : String
+compareTo(Card) : int
+getPath() : String

WildWishChangeCard
+WildWishChangeCard()
+action(Game) : int

WildD4Card
+WildD4Card()
+action(Game) : int