Class Specification Documentation

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Constructors | Properties | Methods |
| SolitaireFrame | SolitaireFrame(): Initializes the frame and the title, dimensions, canvas, gameMenu, statusbar, and board | * Frame\_height: Sets the frame height * Frame\_width: Sets the frame width * Canvas: Sets/updates the canvas to hold the graphics * GameMenu: Generates drop down menu that includes a new game, undo move, and quit options * Statusbar: Sets the statusbar to start tracking time and moves made * Cards: Gets the card graphics * Slots: Gets the slot’s graphics for Foundation and Tableau piles * Game: Updates the board for a new game | * start(): Creates a new game and grabs cardgraphics to draw to screen and starts up our statsbar to start tracking time and moves made * undoMove(): Undo’s the user’s previous move/ updates move count. * exit(): Exits the application * updateScreen(): Gets an updated state of the game |
| DrawPanel | DrawPanel(): Adds a mouse listener and creates the mousePressed method | - | * mousePressed(MouseEvent e): Detects if the mouse has been clicked and if the move was allowed(Updates accordingly) * paint(Graphics g): Calls methods to generate graphics for the slots,cards, and win screen. * loadImage(String filename): Helper method to load an image file * drawBackGround(Graphics2D g): Loads the background image in the frame * drawWinScreen(Graphics2D g): Outputs message indicating the user has won * drawCards(Graphics2D g): Loads images for the cards and sets their size/location. * drawSlots(Graphics2D g): Loads the outline for slots and sets their size/location. |
| GameMenu | GameMenu(): Initializes the New Game, Undo Move, and Quit events and adds action listeners to them. | * Menu: Generates drop down menu that includes a new game, undo move, and quit options * StartMenuItem: Creates a new game * UndoMenuItem: Undo’s the user’s previous move * ExitMenuItem: Exits the application | * actionPerformed(ActionEvent e): Detects if either the new game, undo move, or quit options were selected and acts accordingly. |
| StatsPanel | StatsPanel(): Initializes and sets both the number of moves and time to 0. Then starts the timer. | * MoveCountText: Outputs text indicating the number of moves made by the user * TimerText:Outputs text indicating the amount of time the user has been playing * Timer: Sets the timer either on or off or resets to zero * CurrentTime: Stores the amount of time the user has been playing * MoveCount: Records the number of moves made by the user. | * setTimer(boolean state): Turns the timer on or off * reset(): Resets time and moves back to zero * getCurrentTime(): Passes how long you’ve been playing to the result screen * getMoveCount(): Passes how many moves you made to the result screen. * incrementMoveCount(): Increases numer of moves by 1 * decrementMoveCount(): Decreases number of moves by2 * actionPerformed(ActionEvent e): Updates how long you’ve been playing |
| Board | Board(ArrayList<Slot> slots):  Initializes the board entities, slot entities, and the card entities. | * CardEntities: Gets and generates the card placement (within a pile, cards follow in descending order) * SlotEntities: Gets and generates slot placement * SelectedCardGraphics: Selects the card’s graphics * SlotStartPosition: Determines the starting location for slots | * getSelectedCards(): Gets the card selected by the user * resetSelectedCards(): Clears the selected card so that nothing is selected * checkForCardCollisions(int x, int y): Checks for collsions that may occur between cards * checkForSlotCollision(int x, int y): Checks for collsions that may occur between slots * generateSlotEntities(ArrayList<Slot> slots): Determines the location of slots and gives each slot a correlating id. * generateCardEntities(ArrayList<Slot> slots): * getSlotEntities(): Gets the slot entities for view to initialize it's its array of slot cardgraphics * getCardEntities(): Gets the card entities for view to initialize it's its array of card cardgraphics |
| Card | Card(int rank, int suit) : Initializes a card that has a rank (2-10, J,Q,K,A) and suit (Club, Spade, Heart, Diamond) | * Rank : Gets or sets the rank of the card * Suit : Gets or sets the suit of the card | * Equals(Card) : Checks to see if the card equals another card. * toString() : Prints the rank and the suit name for that card. |
| CardGraphic | CardGraphic(int, int, int)  -CardGraphic(int, int, Card, int) | * PosX : Gets the X position. * PosY : Gets the Y position. * Slot\_id : Gets the ID of the slot that the card is in. | * getCard() : Gets the card that the graphic is associated to. * contains(int, int) : Checks to see if the location(x,y) is inside of this card graphic. * setHighLight() : Sets the card to be highlighted. * isHighLighted() : Boolean for if the card is highlighted or not. * getSlotId() : Gets the ID of the slot. * toString() : Returns the location on the board of the card and what slot it is in. |
| Deck | Deck() : Creates a new deck of cards that are all between the parameters. | * ArrayList<Card> cards : An array of cards that compose the deck. | * shuffleDeck() : Shuffles all the cards in the deck. * -toString() : Prints the suits and ranks of each card in the deck. * -drawCard() : Removes the bottom card of the deck. |
| Slot | Slot(String): Creates a new slot for the cards and adds the cards to the slot. | * Type : Gets the type of the card. * Cards : Creates a new set of cards for the slot. | * createSlot(String) : Creates a slot for the table. * getCards() : Returns cards in the slot. * lookAtTopCard : Returns the top card of the stack (If empty, return null). * removeTopCard() : Removes a card from the top of the stack. * addCard(Card) : Adds a card to the top of the stack. * size() : Gets the size of the cards in the slot. * getType() : Gets the type of the slot. * toString() : Prints the rank and suit names of each card in the slot. |
| Solitaire | Solitaire():  Initializes the logic, slots, and game state history. | * Logic: Defines the logic for the Baker’s Dozen Solitaire Game * GameBoard: Updates the game board/ returns card and slot entities. * AllSlots: Sets the slots by copying the game state history * GameStateHistory: Sets an array list for the game state history | * undoMove():Undo’s the user’s previous move/ most recent state. * updateBoard(): Gets a new instance of the board and updates the slots * getCardGraphics(): Gets the graphics for the cards * resetSelectedCards(): Resets the cards that have been selected * tryToMakeMove(): Determines if the displacement of one card onto another card was successful or not. * selectCard(int x, int y): Determines if the card selected is applicable and no card collisions occur * selectSlot(int x, int y): Determines if the slot selected is applicable and no slot collisions occur * winCondition(): Detects whether or not the player has won the game ( if foundation piles each have 13 cards) |
| SolitaireLogic | SolitaireLogic() | **-** | * setUpFoundationSlots(): Sets up the foundation piles(4 slots) * setUpSlots(): Initializes all slots for both Foundation and Tableau piles. * setUpTableauSlots(): Shuffles and sets up the Tableau * canMoveToFoundation(Slot from, Slot to): * canMoveToTableau(Slot from, Slot to): * canMove(Slot from, Slot to): Checks logic to see if the top card from one slot is allowed to move to top of another slot * moveCard(Slot from, Slot to): Moves a card from the top of one slot to the top of another slot |