Developer's Guide - Crazy Eights

By Victor, Aurthy, and Sophia

Sections

Overview

The programming of this game, Crazy Eights, uses a variety of special objects to hold the information of the game, such as the specific cards in someone's hand. While running the game logic in the main class, these objects are all collectively modified and updated as the game progresses, leading to visual changes in the GUI the player can interact with when the GUI updates.

Classes and Methods

Main Class

This is the class that holds the bulk of the game logic, including the AI. In the main method of this class, methods are used to create instances of the other classes. It first brings out an instance of the menuScreen class, then the GuiMain class, and then the endScreen class before looping back to the start. Key methods in this class include:

run() - holds the game logic, such as turn actions
setup() - sets up all deck and hand objects to starting game positions
specialCard() - checks if a card is special and triggers special events accordingly
getPlayerMove() - uses GUI to receive input from player
getAIMove() - chooses an action for the AI to perform

GuiMain Class

This is the class that holds all of the GUI components. It is very closely tied with the Main class, and updates its drawArea component with graphics of the cards as the game is run in the Main class. Since most of the game input is from mouse clicks, a MouseAdapter is linked to the drawArea, which spans the whole screen of the GUI. Most of the methods in this class are getter or setter methods used by the Main class, as game logic and functionality is not present in this class. For example, addText() is a method used by the Main Class to add text into the GUI's textArea.

Card Class

This is the class that creates the most basic object, a playing card. This has 3 important data fields: rank, suit, and a boolean indicator showing whether the card should be displayed face up or not. Many of the methods are getter methods that retrieve data from the card such as its rank, or setter methods that set parameters such as its orientation. A notable method is show(), which receives a graphics object and draws the card image at a specified location/

Deck Class

This class creates a slightly more developed object, that is made up of numerous Card objects compiled in an ArrayList. It offers many standard methods for manipulation of the cards in the ArrayList, as well as methods used to display cards on the GUI. Notable methods include:

shuffle() - randomizes the order of items in the ArrayList
deal() - removes card from deck and returns it, allowing transfer of cards between decks
quickSort() - sorts cards by rank
selectionSort() - sorts cards by suit and rank
show() - displays cards layered over one another using show() from Card class

Discard Class

This is a specialized Deck object, inheriting from the Deck class. This is used for the pile of cards that the players play on top of. It uses a special show() method that overrides the one from the Deck Class, drawing cards on top of each other like a pile. It also has a special method called dealAll() that is used to transfer all but one card back into the draw pile once the draw pile is empty.

Hand Class

This is a specialized Deck object, inheriting from the Deck class. This is used for the collection of cards in each players hand. This class contains a few special methods that retrieve information about the hand for the AI to process, and methods that manipulate the cards such as slightly modified quickSort() and selectionSort() methods.

Pickup Class

This is a specialized Deck object, inheriting from the Deck class. This is used for the deck of cards players draw from. It uses a special show() method that overrides the one from the Deck Class, drawing cards on top of each other like a pile.

menuScreen Class

A simple class with a constructor that creates a GUI window. This window only consists of graphical components (drawArea), except for the MouseAdapter that listens for user click that will prompt this window to close and move on to the game (GuiMain Class).

endScreen Class

Also a simple class with a constructor that creates a GUI window. Like above, it only consists of graphical components and a MouseAdapter. After the user clicks anywhere to continue, the game will loop back to the menuScreen class instance so the user can play again.

Suggestions for Improvement

Since the GUI of the program uses a full screen drawArea with a full screen MouseAdapter that tracks mouse clicks, it must use a JLayeredPane to add more components such as buttons and labels. Unfortunately, JLayeredPanes are not compatible with layout managers, so these components must be added manually. In addition, the GUI is not resizable. These are some parts that could be improved to make the card game more developer and user friendly.