JAVATAIRE

Implementation Manual

Class Card:

The heart of Javataire. Assigns each object a Suit & Value

- Card(Suit, Value) : Constructor.
- getSuit() : returns Suit of the card.
- getValue() : returns Value of the card.
- getColor(): returns Color of the card.
- getName() : returns a String of the Card's name based on its suit and value. For use in loading images.
- isFlipped() : returns a Boolean stating whether the Card is flipped (visible).
- isSelected() : returns a Boolean stating whether the Card is selected by the user.
- flip() : flips the Card accordingly.
- select() : selects the Card accordingly.

Class Deck:

Manages Card and stores Card objects. Creates piles for stock & waste, as well as lists of stacks of Cards for the tableau and foundations.

- initialize() : creates a standard deck of Cards.
- swap(int, int) : used for shuffling Cards.
- shuffle() : randomizes the order of each Card in the deck.
- createTableau() : places tableaus of Cards in default starting positions.
- popStock() : takes card at top of the stock and places it into waste.
- reStock() : restocks waste cards.
- checkWin() : checks if win condition has been met. Boolean value true returned if game is won.
- selectCard(Card) : sets Card's status to selected.
- deselectAll() : sets all currently selected Cards back to deselected (isSelected() == false).

Class Game:

Class that runs the game using event-driven code written for JavaFX GUI systems.

- start(Stage) : Starts the game and sets a primary panel.
- displayHelp() : Shows tutorial video.
- showMenu() : Brings up options menu for the user.

- drawCards() : Updates game board’s cards every time the user moves something or a deck is made.
- startMove(Deck) : Opens a window that lets the user pick where they want to move.
- executeMove(Deck, Integer, Integer) : Uses input from startMove() to move a card, so long as the move is valid.
- main(String[]) : Launches program using launch arguments.

enum Suit:

For Card objects. Assigns one of the four suits to a Card:

CLUBS, SPADES, HEARTS, DIAMONDS.

enum Value:

For Card objects. Assigns a weight to each card depending on its value (aces low, 2-10, then jack – king):

ACE, TWO, THREE, FOUR, FIVE, SIX, SEVEN, EIGHT, NINE, TEN, JACK, QUEEN, KING.

enum Color:

For Card objects. Has use in comparing Cards to see if there is a match:

BLACK, RED.

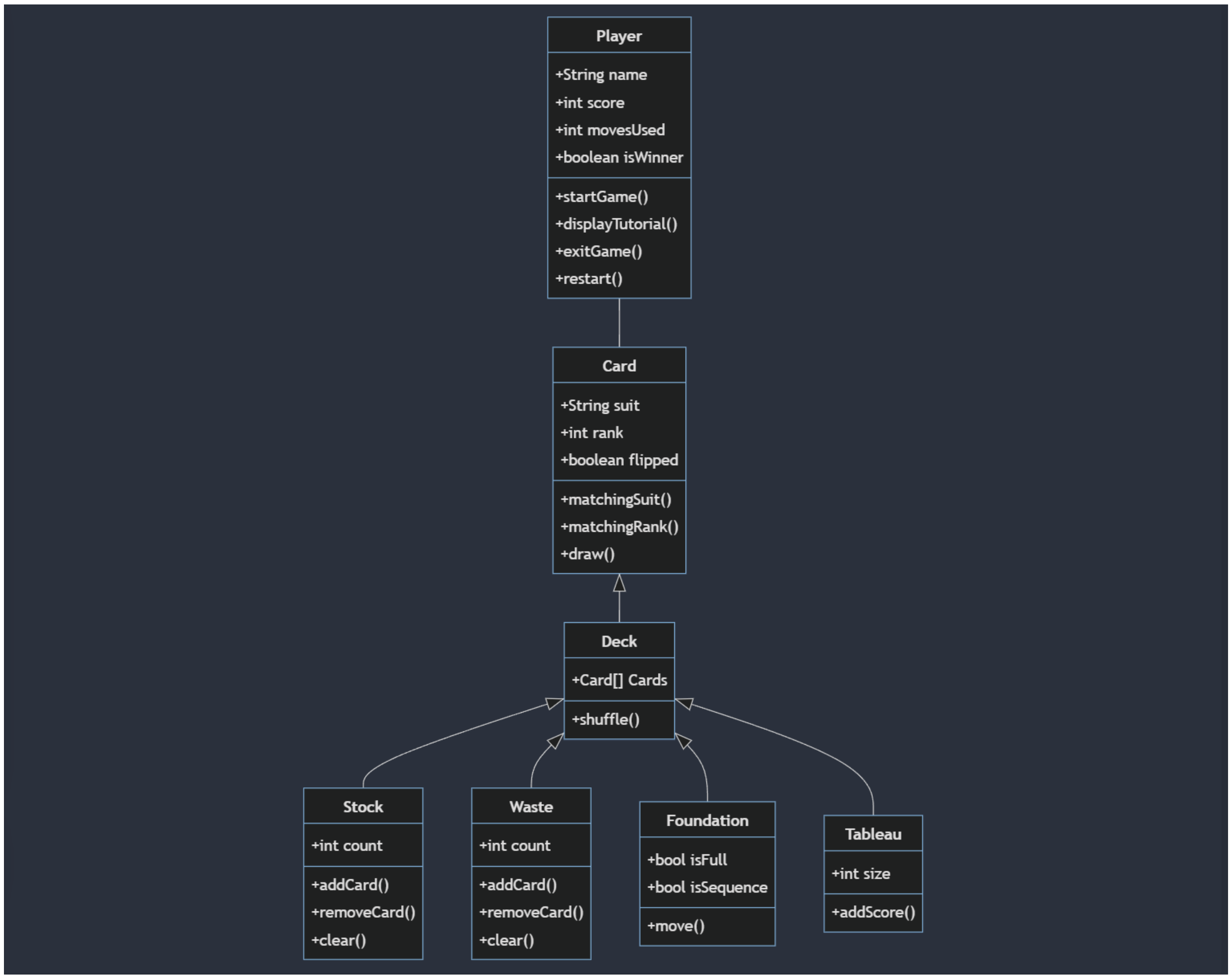

Figure 1: Initial UML for this application.

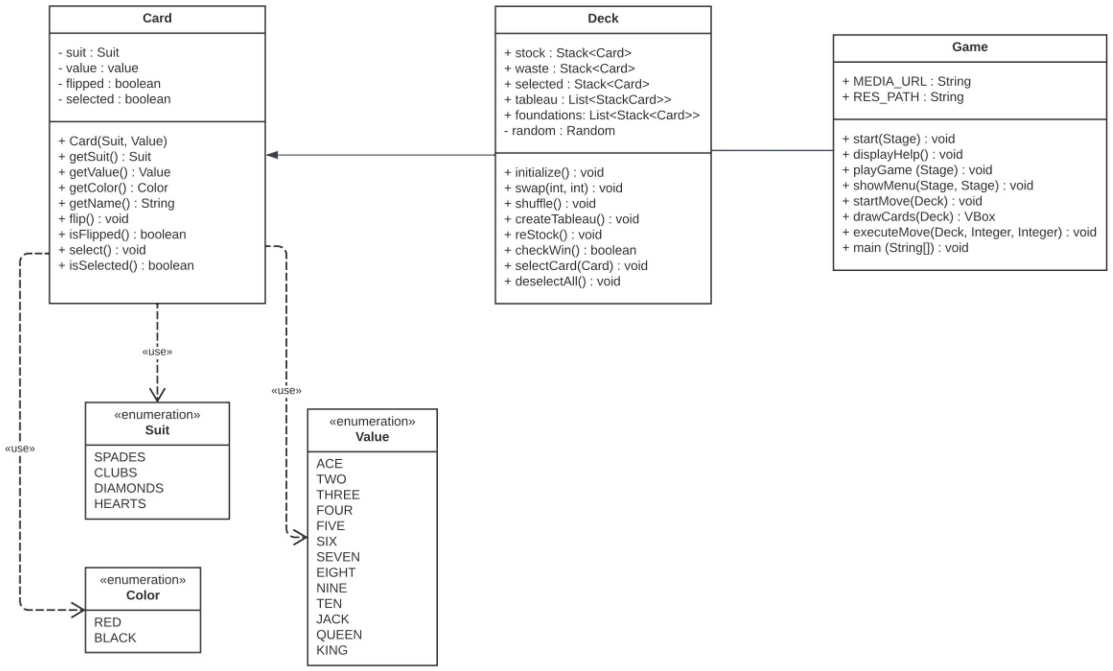

Figure 2: Final UML for this application.

Much has changed since the original proposal for this project. After referencing many other similar projects and programs that implemented decks of cards, it made more sense to implement the elements of solitaire into a Deck class full of Card objects, as opposed to having a master Player class that holds Card objects. In my original approach, the scope of solitaire was in the wrong place. The code in the final build of the application gets rid of the Player class entirely and instead opts for the JavaFX controls to instead integrate Deck and Card as if they were any other data type.