Solitaire User Manual

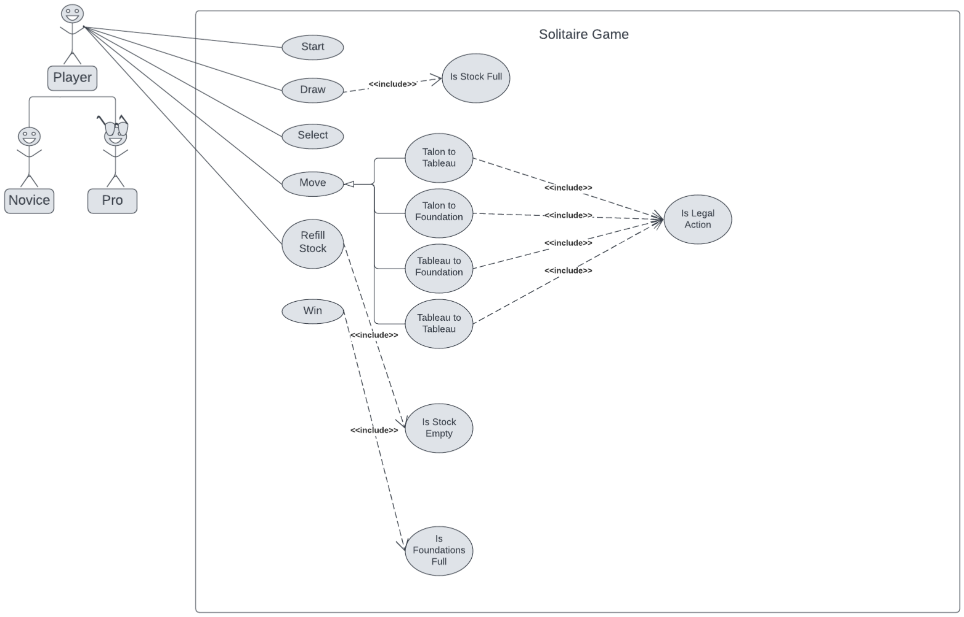
0. What is this? What am I looking at?

The program you are looking at is our rendition of the game *Solitaire.* More specifically, *Klondike Solitaire.*
"Solitaire? Klondike? Like that brand of ice cream sandwiches?"
… No. I'll explain.

1. Introduction to *Solitaire*

Solitaire is a series of card games designed to be played individually. No inputs from additional players are required, nor involved. Because of this, a large repertoire of *Solitaire* games have been developed on numerous operating systems, as the dealing process of these games can be automated by the computer. Many *Solitaire* game modes have been developed over time. Klondike, Spider, FreeCell, TriPeaks, and Addiction, are all names of different *Solitaire* game modes. In our case, we have developed its most popular game mode: Klondike.

2. Project Background

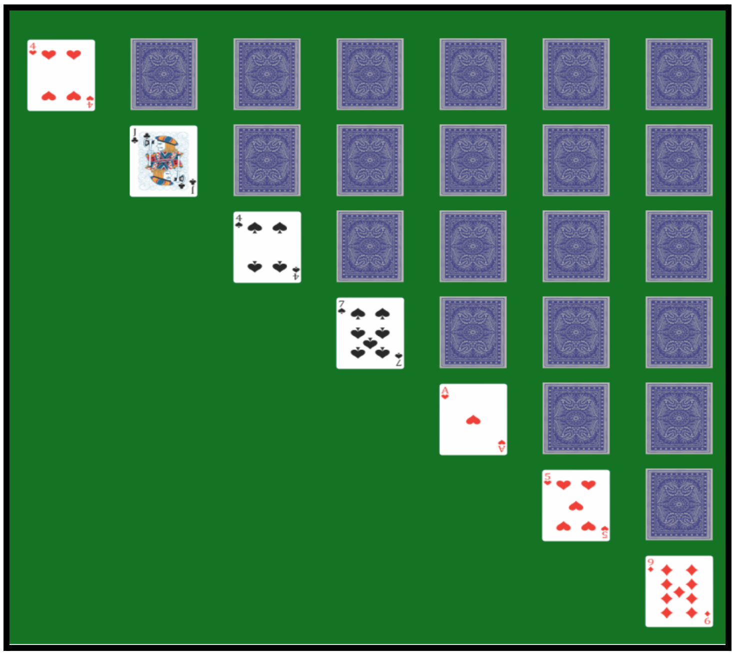


(The UML Use case diagram for Klondike. This is a brief rundown on each action the player must be able to perform. Actions such as 'Move' have different variations of move depending on where it is moving from or to. <<include>> refers to what we needed to 'include' in an action to make it work according to our specifications)

Our *Solitaire* project is developed from an older, existing card game with many rules- as well as quality of life features that tend to be a norm when playing on a computer. These features would be things like an undo button, a monitor for time elapsed since the start of the game, or a move counter. We gathered as much information on these features as possible, and used them as references to better understand how we as a team should establish our code to make these features possible for our own program. The Use Case Diagram (seen above) is a collection of all the actions or events that would occur during a run of our program. See how our 'player' has lines tied to multiple actions. These are the events that are player directed. Drawing, moving, and selecting cards are all actions that should not occur without a player interaction to signal such. The 'Win' event does not have a connection to anything since there should not be a 'win' button that a player can press (obviously). Of course these specific actions needed to obey the rules of Klondike. Each action has a conditional statement included, which acts as a decider on whether or not the event should trigger based on rules we provide. Of course, we also had to research the rules of Klondike to understand how each action should be handled conditionally. Klondike's rule set is as follows.

How Klondike Starts - Setup

Klondike first begins with a standard shuffled 52 deck of cards- jokers removed. From there, the top seven cards of the deck are distributed evenly along seven card 'slots', face down. When seven cards are placed, the left-most slot is turned up, then the remaining six 'slots' have one face-down card added on top. From there, the leftmost card is turned face up, and the cycle is repeated until the rightmost card being placed down is turned up as well. The result should look like this:

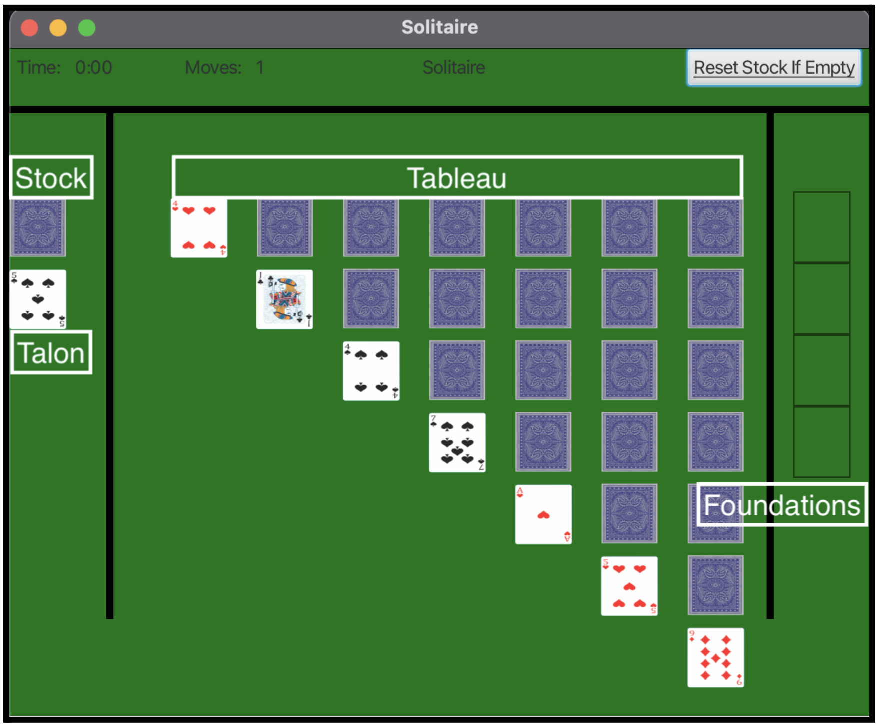


(Cards are stacked in ascending order as shown)

The remaining cards are placed to the side. Congratulations! This is how you set up your first game of Klondike *Solitaire.* While you can use this as a future reference for when you would like to play with real playing cards, our program automatically sets the game up for you. We performed this by effectively simulating a shuffled deck and having our code run in the same order as the cards should be distributed. Now, onto the layout.

3. The Layout

There are a total of 4 different sections in the layout of Klondike: the Stock, the Talon, the Tableau, and the Foundations. When you first start a game, it will look similar to this:



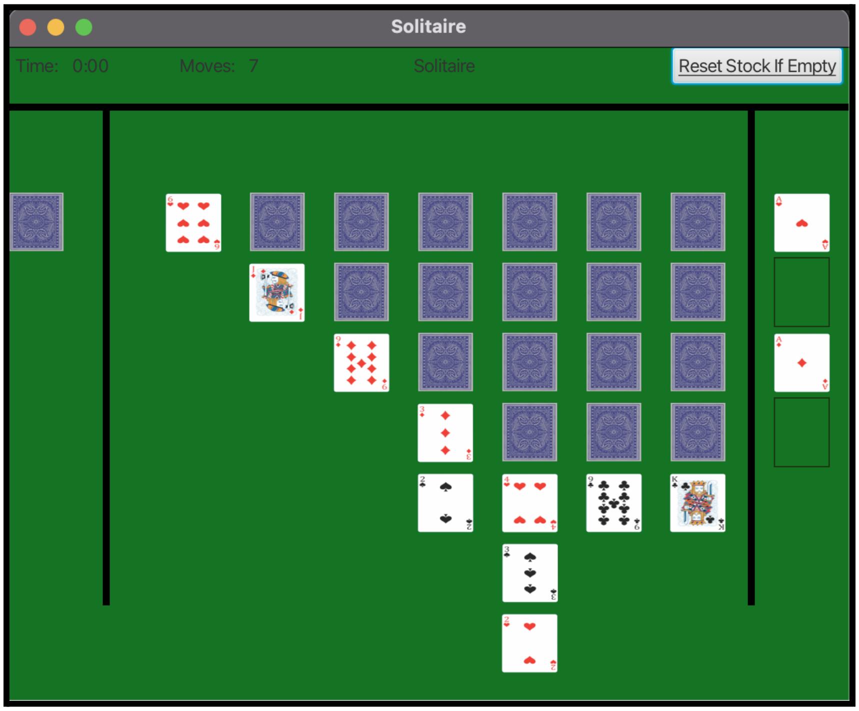
(The four sections of Klondike)

A little complicated visually, but I'm going to break it down into smaller parts. It makes a lot more sense that way. Let's start with the Stock.

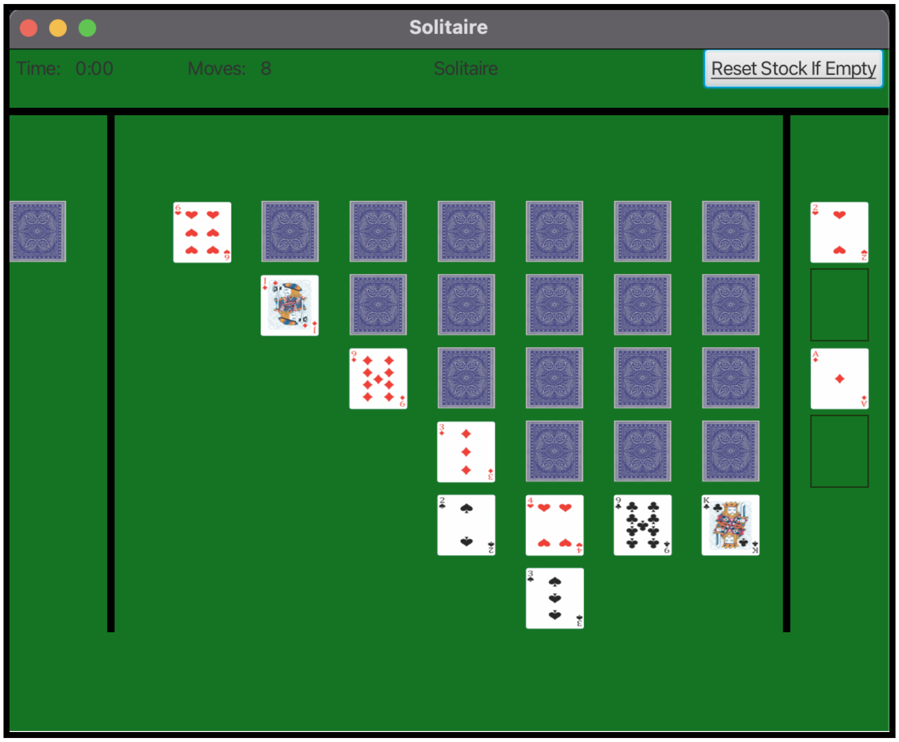
The Stock is effectively your deck. If you need a card, you'd click on the Stock, and it would draw the top card for you, placing it in the Talon. The Talon is where your drawn cards are located. At game start, the Talon starts empty. From the diagram, there appears to be a Five of Spades in the Talon. This means I had clicked on the Stock, and drawn the card as a result. If I were to click the Stock again, it would draw a new card, and place it on top of the existing Five of Spades, preventing me from selecting the Five of Spades. If I move this new card out of the way, however, the Five of Spades would be visible again, allowing selection.

The Tableau is the forefront of the game. This is where almost all card movement takes place. It is also the most complicated. Cards from the Talon can be moved to the Tableau, and cards in the Tableau can be moved around the Tableau itself as well. What makes movement complicated, however, is what is considered legal moves. We'll get back to that in a bit.

Finally, our Foundations. These are the four empty slots that act as card storage, holding each respective suit. The Foundations can only collect cards in an ascending order, starting with an ace, then a two, a three, continuing all the way to a king.



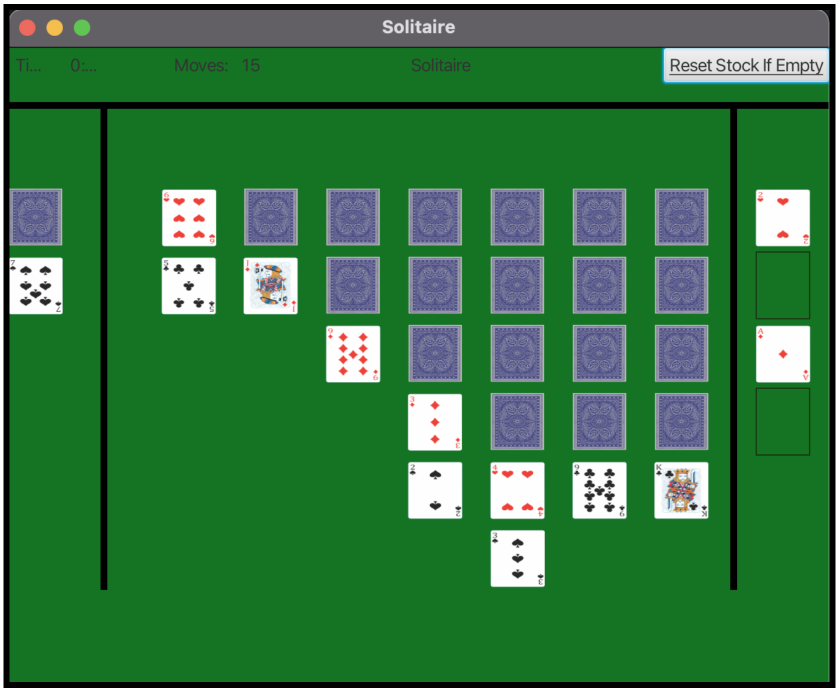
(An Ace of Hearts has been placed in the Foundations, which can allow the Two of Hearts to be moved on top of it)



(... like so!)

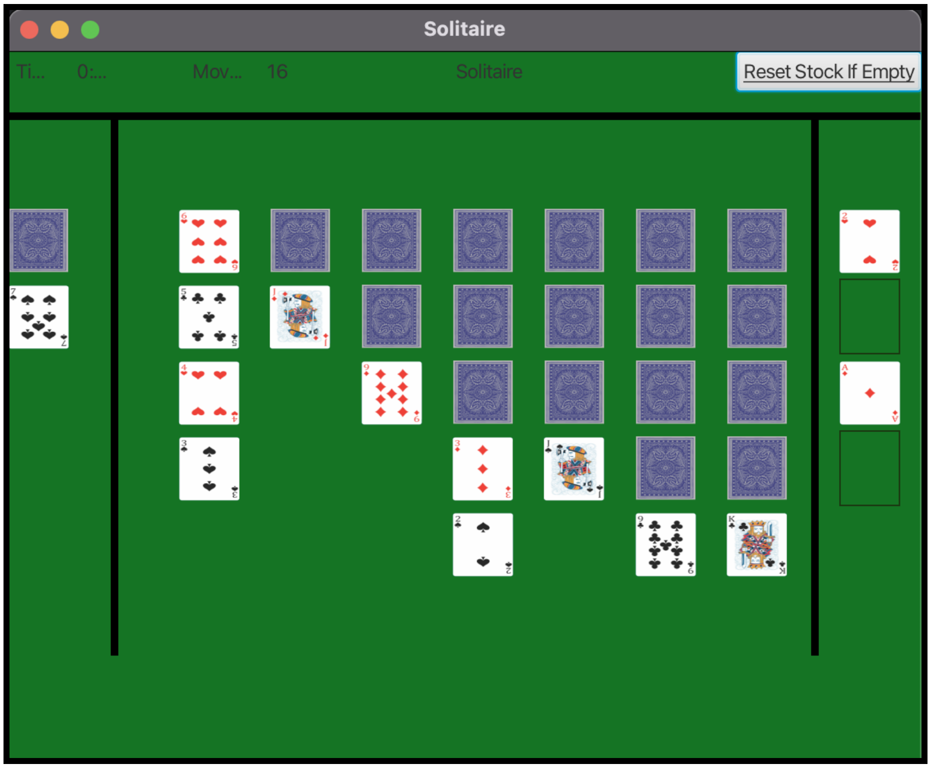
4. The Objective

The final objective of Klondike is to have <u>all four suits in the Foundations stacked to their kings.</u> Effectively, you're meant to clear the Tableau and Stock of their cards. This is done by slowly bringing cards from the Tableau into the Foundations. What will likely prevent you from completing the game objective are the face-down cards that have been set up in the Tableau. These can only be turned face-up once all face up cards beneath it have been moved away. This will lead to the bottom-most card being turned over. To move a card from place to place, certain conditions must be fulfilled. If you wanted to move a card from a pile, <u>its new location must be below a card that is incrementally above it by one, and must have a mismatched color.</u>



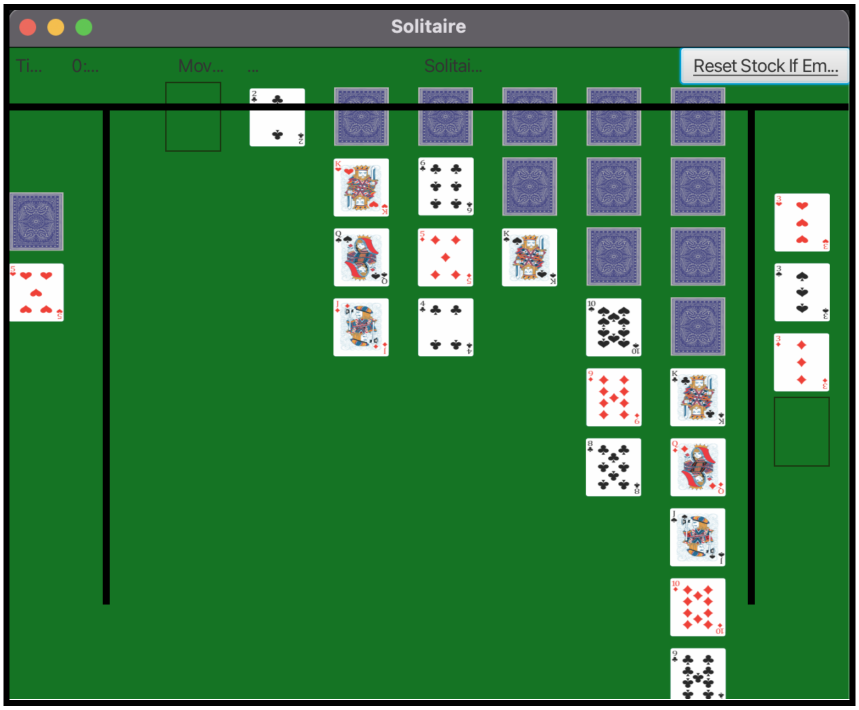
(Note that the movement of the Four of Hearts to the Five of Clubs will also cause the Three of Spades to move alongside it)

This is an excellent example of movement in the Tableau. On the left-most side is a pile of cards with a Five of Clubs at the bottom. The fifth column in the Tableau has a Four of Hearts and a Three of Spades, which can be moved legally to the first column. This is due to the fact that the Five of Clubs is one above four, and both these cards have differing suit colors.

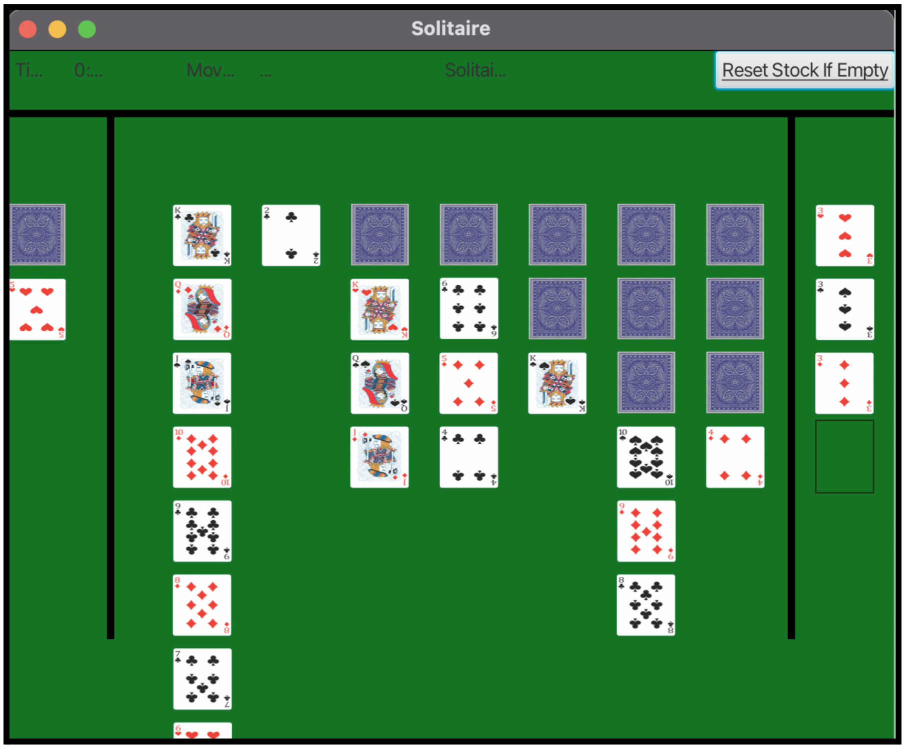


(Successful move! The fifth column has been emptied, and the bottom card has been flipped, revealing a Jack of Spades)

As the game progresses, you will slowly uncover each of these cards, eventually bringing them into these piles, then finally adding them to the Foundations. Another rule for Tableau movement is that a King is the only card that can be moved to a fully empty column.



(That King on the right-most column can move to that empty slot on the left)



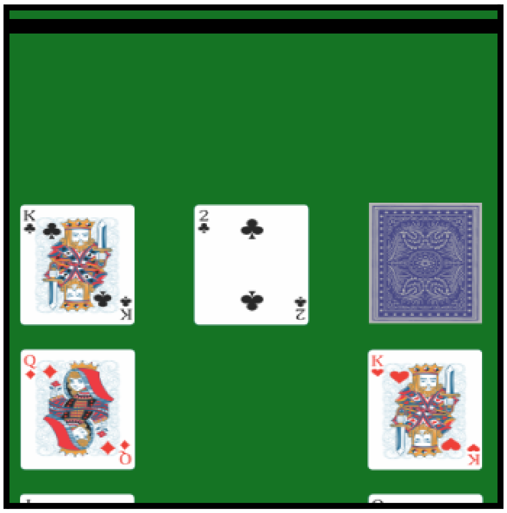
(Tada!)

Congratulations! You now know the rules of Klondike *Solitaire*. If there are further questions about how the game is played, there are a multitude of other help sources online that may relieve confusion.

5. Our Features

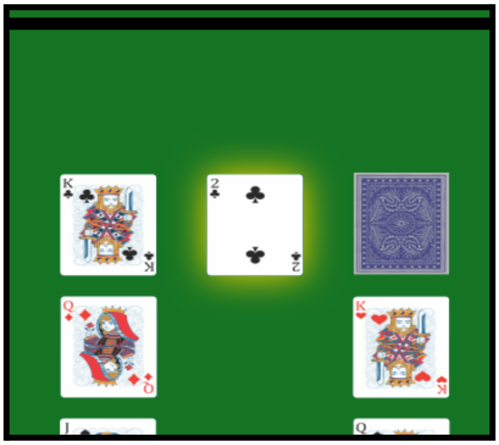
There are a variety of features included in our program beyond the logic behind how the game is played. Let me give a rundown:

Selecting cards should provide visual feedback to let you, the user, understand what it is you have selected. Sometimes when deciding an important move one may forget what it is they have selected, or whether they have even selected it to begin with. Enter the glow effect.



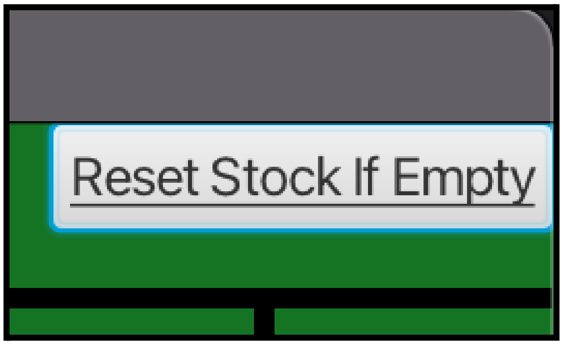
(An unselected card. No glow. No fun.)

The glow effect is simple, yet necessary. Click on a card, code it so a yellow drop shadow is added to make it appear glowing. While it is only a cosmetic feature to the game, it definitely reduces confusion for the average *Solitaire* user who is likely trying to relieve stress during or between work periods.



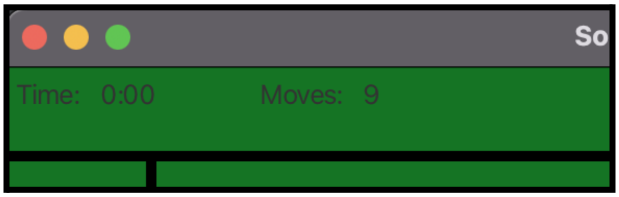
(A selected card. Yes glow. Yes fun.)

We also added a feature to reset the Stock once its been flipped through. This was mostly just a necessary feature, as the game would effectively be impossible (and quite short-lived) to finish without being able to flip the Talon back to the Stock.



(A big button at the top right of the playspace. It would only work once you've cleaned out the Stock)

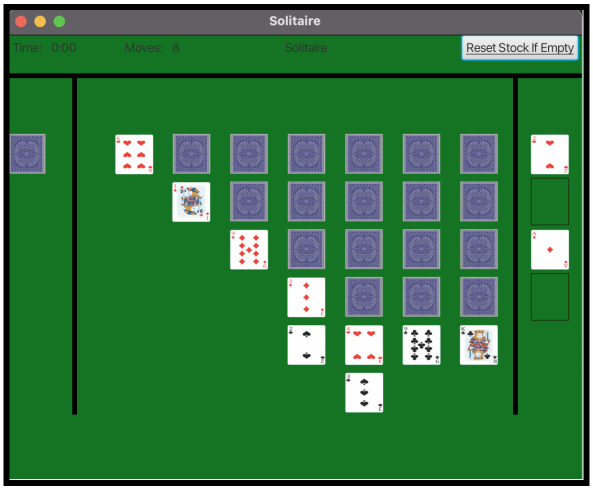
Lastly, we added a move counter. This can give the user an objective when replaying the game. If they can see how many moves it took to finish, they can try to beat their score the next time around.



(Nine moves recorded. Definitely far from finishing)

6. Our Motivations

Our motivations for our rendition of Solitaire stem from our interest in the original game itself. We wanted to better understand further how the game functions down to its code level, and tried to surmount it using our current skills in Java programming. We used our user stories as guidance on what exactly our goals should be when making our project; our users were mainly adults looking to have a simple and accessible game of Klondike. User input is minimal effort and the user has access to information on their game without needing to play guessing games on where to get it. From this, we knew we needed to have the program fit all on one window. No drop down menus or redundant information. Simply Solitaire.



Time: 0:00 Moves: 8 Solitaire

(Our Solitaire program at its release state)