⭐️ Assignment 5 - Playing Cards

开始任务

* **截止** 2月15日 由 14:00 编辑

* **得分** 20

* **提交** 一个文本输入框或一份上传文件

* **文件类型** txt

* **可用** 2月1日 8:00 至 2月18日 14:00

**Due Wednesday @ 11:59 PM**

Goals

* In this assignment, we will be designing a program using mouse events

 Instructions

Hand in only one program, please.

Mouse events for Card Images

Understand the Classes and Problem

Now we can add mouse events to our card game.  We'll do this in stages.

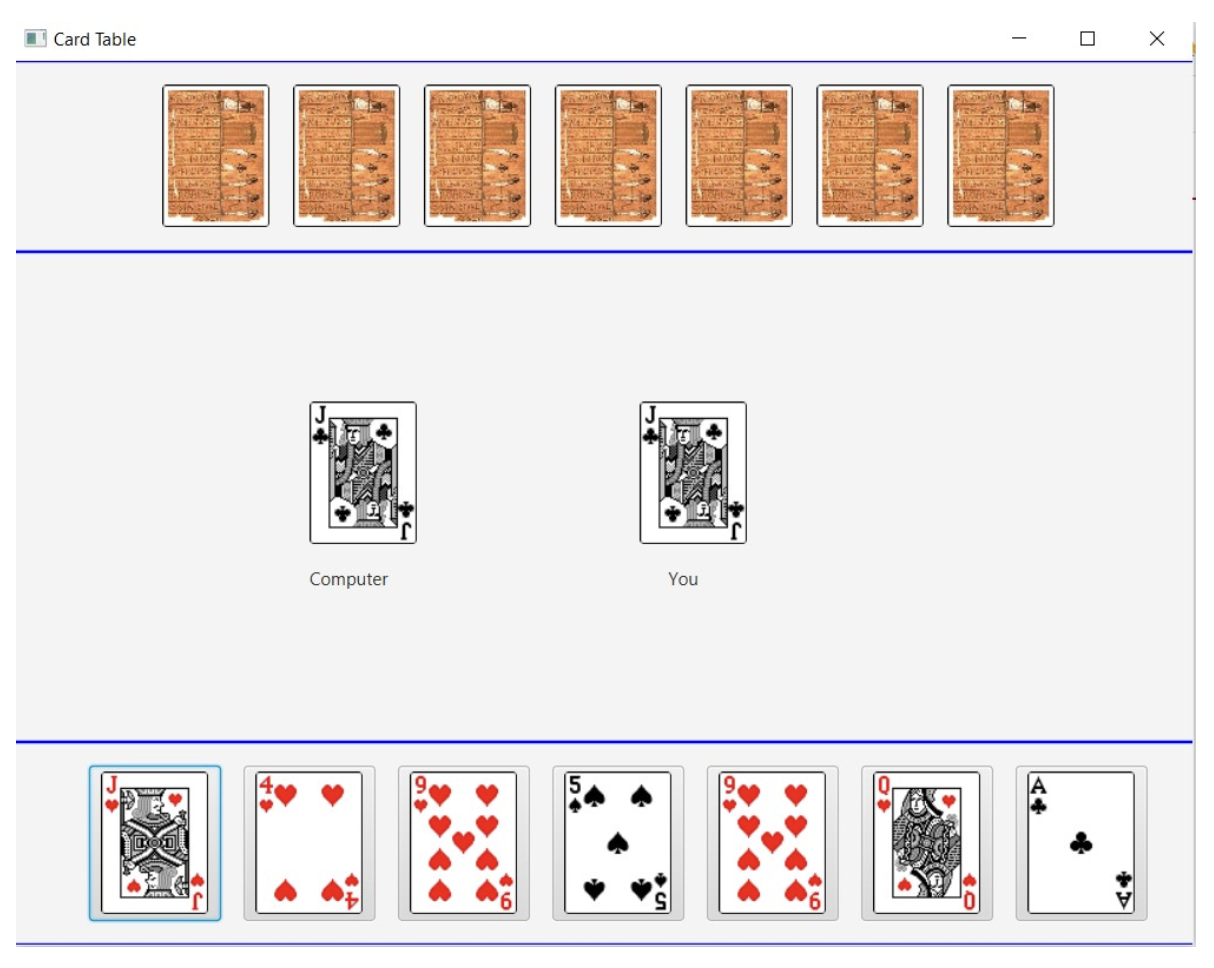
1. **Making buttons** - create buttons and then put the image of the card on it to display.
2. **Putting Cards out into the playing area**-  use event-driven programming to move a card from your hand.

This will be done in***two phases***.  The main, public class of each program must be named **Foothill** so I can easily run it, and other classes must *not* be public.  *You do not submit any runs - I will run your programs*.

This is a continuation from Week 4.

Phase 1: Making Human's cards into buttons

* All of this will only affect the human's hand of cards.  We need to create a button for each of the 7 cards that are in the human's pane.
* Inside of the For loop where we are generating the Images and ImageViews, create a button for each and use the setGraphic(ImageView) method to put the card graphic on the button.
* Put the buttons into the human pane, instead of the ImageViews.
* Since the cards will now be surrounded by a border to signify that they are now buttons, you will have to adjust the Insets for the human pane so that it all fits on the screen.



Phase 2 - Putting Cards out into the playing area

* Here is a new private method that will help the start():

**private void moveCard(Image cTemp, Image hTemp, Pane pPane)**- an image for the computer side and an image for the human side of the players pane (middle) in the playing area will be the arguments.  You will need to clear the pane using .getChildren().clear() and then put the images into the playedImages array and convert them to ImageViews.  These along with the labels should be added to the pPane.  Several of those lines of code can simply be moved from the start() method.

* Next it is time to use the setOnAction() event and call the method, moveCard(Image, Image, Pane).  In the for loop where the human images and buttons are being created, we will send it the image for the human on that round of k as well as a random card for the computer player.  In order for these images to be sent, we will have to create some temporary Image variables and mark them final so that setOnAction() will accept them.  We will call moveCard() anytime we click the mouse on one of the cards (as well as during the initial run through of start()).  If we set the computer value inside of the .setOnAction(), then we will get a new card each time we click on a given card image button. Here is some code to help you:

         final Image hTemp = ...  
         humanBtn[k].setOnAction(e -> {  
            final Image cTemp = ...  
            moveCard(. . . ));  
            }

* Of course in order to use the pPane early, you will have to move its declaration up sooner, prior to the start() method.
* For the initial run of start(), make a call to moveCard() just before you call setTop(), setCenter(), and setBottom().  Put in a couple of jokers for the start.
* Here is a how it could look after you click on one of the cards.  The clicked card will appear on the right side and a random card will appear on the left for the computer.

