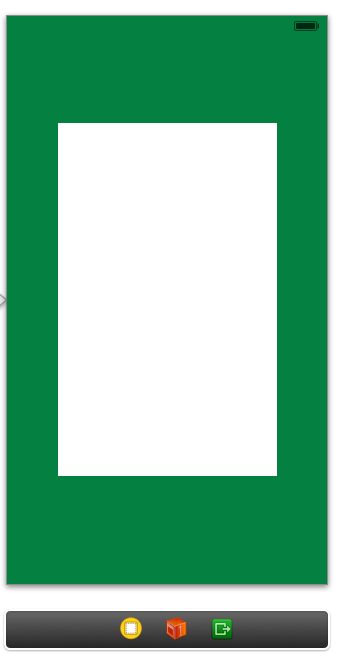
Practice 2 and Review for Test on Feb. 24, 2014

Create a single view application called SuperCard. Set the prefix text to be SuperCard as well.

Set the background of the view to the Moss color from the Crayons Color Picker. You get the Crayons color picker by selecting ‘Other’ in the background color menu item and then select the Crayons icon at the top of the panel that appears.

Add a UIView to the storyboard which is centered but somewhat smaller than the current window. It should look something like this:



Create a subclass of UIView called PlayingCardView.

Add the following properties to PlayingCardView

1. NSUInteger called *rank*
2. NSString called suit
3. BOOL called faceUp

All three are nonatomic however only one is strong. Which one and why?

Create three setters for these three properties which call *setNeedsDisplay* to insure that whenever any of these properties are changed, the view is redrawn.

Define a constant called CORNER\_FONT\_STANDARD\_HEIGHT to be 180.0

Define a constant called CORNER\_RADIUS to be 12.0

Create a function called cornerScaleFactor which returns a CGFloat value equal to the view height divided by CORNER\_FONT\_STANDARD\_HEIGHT.

Create a function called cornerRadius which returns a CGFloat value equal to CORNER\_RADIUS multiplied by the return value of the cornerScaleFunction you just created.

Create a function called *cornerOffset* which returns 3.0 times the number returned by the function *cornerRadius*.

## In drawRect of PlayingCardView:

Create a UIBezierPath called roundedRect by using the class method *bezierPathWithRoundedRect.* The first argument is the bounds of the view and the second argument is the return value of *cornerRadius*.

The goal is to create a rounded rectangular view that will be able to scale. Large cards need a larger corner radius.

Clip the view to the rounded rectangle you just created. Use the *addClip* method.

Send the *setFill* message to [UIColor whiteColor]. This sets the graphics context fill color to white.

Fill the rectangle defined by the bounds of the view. Use *UIRectFill.*

## In PlayingCardView.m

Create a method called *setup* which sets the background color to *nil*, *opaque* to NO, and *contentMode* to *UIViewContentModeRedraw*. If you don’t understand these properties, look them up.

Call *setup* inside *awakeFromNib.* You need to create *awakeFromNib.*

## In drawRect of PlayingCardView:

Draw a black outline around the card by first sending the *setStroke* method to [UIColor blackColor] and then send the *stroke* method to *roundedRect.*

## In the storyboard

Change the class of the smaller view you added to the view controller to be PlayingCardView

## In PlayingCardView.m

Create a void function called drawCorners. Call this function in *drawRect*.

## In drawCorners

Create an instance of *NSMutableParagraphStyle* called *paragraphStyle.*  Set the paragraph style alignment property to *NSTextAlignmentCenter*.

We will use NSMutableParagraphStyle to set the paragraph style for our attributed strings.

Create a UIFont object called *cornerFont* and set it to *UIFontStyleTextBody* using the UIFont class method called *preferredFontForTextStyle*.

Change the font size of *cornerFont* by setting *cornerFont* equal to the result of *fontWithSize*. The new size will be the *pointSize* property multiplied by the result of *cornerScaleFactor.*

(cornerFont =

[cornerFont fontWithSize:cornerFont.pointSize \* [self cornerScaleFactor]])

## In PlayingCardView.m

Create a function called rankAsString which returns an NSString pointer and takes an NSUInteger as an argument. The return value is @”?” for 0, @”A” for 1, @”2” for 2, @”3” for 3, … , @”J” for 11, @”Q” for 12, and @”K” for 13.

## In drawCorners

Create an NSString object called *cornerString* which is the result of using the class method, *stringWithFormat,* on the two properties; rank and suit. Use *rankAsString* to get the rank as a string object and use @”%@\n%@” as the formatting string.

Create an *NSAttributedString* object called *cornerText.* Use the *initWithString* class method and use *cornerString* as the argument. For the properties dictionary argument to *initWithString*, set *NSFontAttributeName* to be *cornerFont* and *NSParagraphStyleAttributeName* to be *paragraphStyle.*

Declare a *CGRect* variable called *textBounds*.

Set the origin of *textBounds* to be (x,y) where x and y are both equal to the result of calling *cornerOffset* .

Set the size of *textBounds* to be equal to the size of *cornerText*.

Send the message *drawInRect* to *cornerText* with the argument of *textBounds.*

## In SuperCardViewController

Import “PlayingCardView.h”.

Attach a referencing outlet to the *PlayingCardView* subview and call it *cardView.*

In *viewDidLoad* set the suit property of *cardView* to @"♥" and rank to King. (13)

If you run the app now you should see a playing card with the king of hearts label on the upper left hand corner.

## In drawCorners

Create a CGContextRef variable called *ctx* by calling *UIGraphicsGetCurrentContext*(). You will use this variable in the next two function calls.

Translate the context by the width and height of the window by using *CGContextTranslateCGM*().

Rotate the context by 180 degrees by calling CGContextRotateCGM().

Send *drawInRect* to *cornerText* again with *textBounds* as the argument so the opposite corner will also display the rank and suit.

Now when you run the program you should see both the upper left and lower right corners with labels of rank and suit.

