A Tic-Tac-Toe Example Application

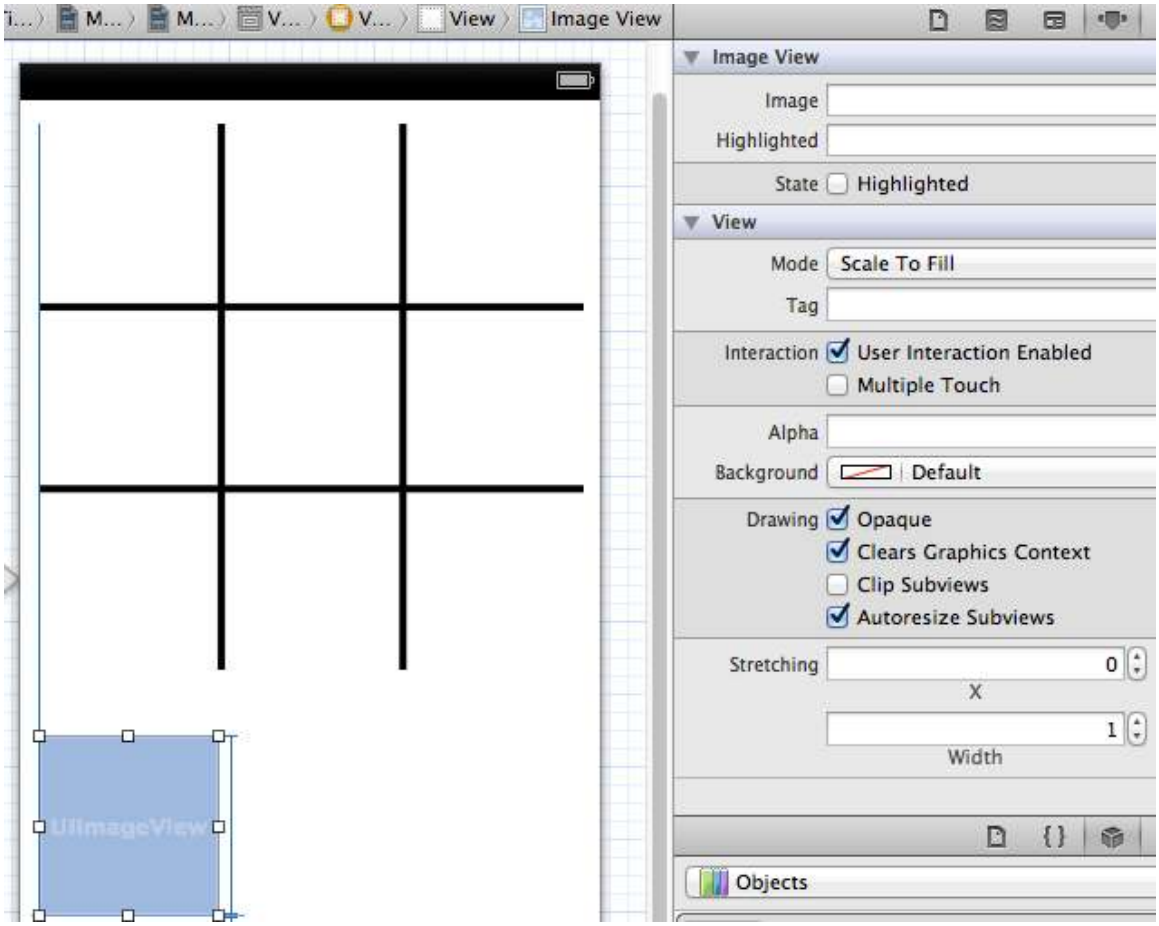
1. Create a new Xcode **Single View Application** project. Call it something like "**TicTacToe**" or another title of your choice. Use the Storyboard support and enable Automatic Reference Counting (ARC). You can create an iPhone, iPad or Universal project type.

The game will work by the user touching a square on the game board that will place an X or an O depending on whose turn it is. We will check for a winner after the user places their mark on the board. If there is no winner, then we will update the text label to say that it is the next persons turn. We will also create a button for the user to reset the game. **Copy the supplied images into the project**.

2. Create the User Interface: Open ViewController.xib or MainStoryBoard.storyboard by double clicking it.

A. Drag a UIImageView to the window and resize it to be approximately 300 x 300 and set its image to board.gif by using the Attributes Inspector.

B. Drag another UIImageView to the window and resize it to 100 x 100 then check the box for User Interaction Enabled. See the following screen shot:



C. Then copy and paste that 8 more times. Move those boxes to be hovering over the spaces of the game board lining them up. The finished game board looks like this:

D. Drag a UILabel to the window and position it under the board.

E. Drag a UIButton to the window and rename the text to Reset Game. The finished interface looks like this:

3. Wire the GUI components to the ViewController.h file:

First, add two IBOutlets to be used for the UIImages and an NSInteger to keep track of the players as follows:

```
IBOutlet UIImage * oImg;
IBOutlet UIImage * xImg;
NSInteger playerToken;
```

The finished code looks like this:

```
#import <UIKit/UIKit.h>

@interface ViewController : UIViewController {
        // the X or O images
        IBOutlet UIImage * oImg;
        IBOutlet UIImage * xImg;

    NSInteger playerToken;
}

@end
```

These items will be set programmatically in the ViewContoller.m implementation so they are not wired to any of the GUI components.

Next, wire the board to "board" and the label to "whoseTurn." Also wire the button to a "resetButton" property. The code looks as follows:

```
@property (weak, nonatomic) IBOutlet UIImageView
*board;
@property (weak, nonatomic) IBOutlet UILabel
*whoseTurn;
@property (weak, nonatomic) IBOutlet UIButton
*resetButton;
```

Next, wire all 8 buttons to UIImageView IBOutlets labeled s1-s8. The finished code looks like this:

```
@property (weak, nonatomic) IBOutlet UIImageView
*s1;
@property (weak, nonatomic) IBOutlet UIImageView
*s2;
```

```
@property (weak, nonatomic) IBOutlet UIImageView
*s3;
@property (weak, nonatomic) IBOutlet UIImageView
*s4;
@property (weak, nonatomic) IBOutlet UIImageView
*s5;
@property (weak, nonatomic) IBOutlet UIImageView
*s6;
@property (weak, nonatomic) IBOutlet UIImageView
*s7;
@property (weak, nonatomic) IBOutlet UIImageView
*s8;
@property (weak, nonatomic) IBOutlet UIImageView
*s9;
```

Next, add the IBAction to trigger when the touch down event on the UIButton occurs. Call the event "buttonReset." The finished source code looks like this:

```
- (IBAction)buttonReset:(UIButton *)sender;
```

Next create three method prototypes to be used for the game logic:

```
-(void) updatePlayerInfo;
-(void) resetBoard;
-(BOOL) checkForWin;
```

4. Open up the tictactoeViewController.m and type the following code.

Initialize the data for the game in the ViewDidLoad method as follows:

```
-(void)viewDidLoad{
    [super viewDidLoad];
    // add the images
```

```
    oImg = [UIImage imageNamed:@"O.png"];
    xImg = [UIImage imageNamed:@"X.png"];

    // set the player to 1
    playerToken = 1;
    // update the label
    whoseTurn.text =@"X goes first";
}
```

Add the method to update the player info as follows:

```
- (void) updatePlayerInfo{

        if(playerToken == 1) {
            playerToken = 2;
            whoseTurn.text = @"It is O turn";
        NSLog(@"playerToken = %d", playerToken);
        }
        else if(playerToken == 2) {
            playerToken = 1;
            whoseTurn.text =@"It is X turn";
        NSLog(@"playerToken = %d", playerToken);
    }
}
```

Add the action for the reset button:

```
- (IBAction)buttonReset:(UIButton *)sender {
    [self resetBoard];
}
```

Add the implementation for the “resetBoard” method:

```
-(void) resetBoard{
    /// clear the images stored in the UIIMageView
    s1.image = NULL;
    s2.image = NULL;
    s3.image = NULL;
    s4.image = NULL;
    s5.image = NULL;
    s6.image = NULL;
    s7.image = NULL;
    s8.image = NULL;
    s9.image = NULL;

    // reset the player and update the label text
    playerToken= 1;
    whoseTurn.text = @"X goes first";
}
```

Next, implement the game logic for the touching of the spaces on the board.

```
// the touch event for the tic tac toe game
- (void)touchesBegan:(NSSet *)touches
withEvent:(UIEvent *)event{

    UITouch *touch = [[event allTouches] anyObject];

    // check to see which UIImage view was touched
    if(CGRectContainsPoint([s1 frame], [touch
locationInView:self.view])){
```

```
        if(playerToken==1){ s1.image = xImg; }
        if(playerToken==2){ s1.image = oImg; }
    }

    if(CGRectContainsPoint([s2 frame], [touch
locationInView:self.view])){
        if(playerToken==1){ s2.image = xImg; }
        if(playerToken==2){ s2.image = oImg; }
    }

    if(CGRectContainsPoint([s3 frame], [touch
locationInView:self.view])){
        if(playerToken==1){ s3.image = xImg; }
        if(playerToken==2){ s3.image = oImg; }
    }

    if(CGRectContainsPoint([s4 frame], [touch
locationInView:self.view])){
        if(playerToken==1){ s4.image = xImg; }
        if(playerToken==2){ s4.image = oImg; }
    }

    if(CGRectContainsPoint([s5 frame], [touch
locationInView:self.view])){
        if(playerToken==1){ s5.image = xImg; }
        if(playerToken==2){ s5.image = oImg; }
    }


    if(CGRectContainsPoint([s6 frame], [touch
locationInView:self.view])){
```

```
        if(playerToken==1){ s6.image = xImg; }
        if(playerToken==2){ s6.image = oImg; }
    }

    if(CGRectContainsPoint([s7 frame], [touch
locationInView:self.view])){
        if(playerToken==1){ s7.image = xImg; }
        if(playerToken==2){ s7.image = oImg; }
    }

    if(CGRectContainsPoint([s8 frame], [touch
locationInView:self.view])){
        if(playerToken==1){ s8.image = xImg; }
        if(playerToken==2){ s8.image = oImg; }
    }

    if(CGRectContainsPoint([s9 frame], [touch
locationInView:self.view])){
        if(playerToken==1){ s9.image = xImg; }
        if(playerToken==2){ s9.image = oImg; }
    }
    [self updatePlayerInfo];

}
```

Now run the game and see what happens. It should work but there is no check for a winner yet. Add the following method implementation to check for a win:

```
// method that will check to see if someone has won
returns TRUE if someone wins
-(BOOL) checkForWin {
```

```
        // HORIZONTAL WINS
        if((s1.image == s2.image) & (s2.image ==
s3.image) & (s1.image != NULL))
        {
            return YES;
        }
        if((s4.image == s5.image) & (s5.image ==
s6.image) & (s4.image != NULL))
        {
            return YES;
        }
        if((s7.image == s8.image) & (s8.image ==
s9.image) & (s7.image != NULL))
        {
            return YES;
        }
        // VERTICAL WINS
        if((s1.image == s4.image) & (s4.image ==
s7.image) & (s1.image != NULL))
        {
            return YES;
        }
        if((s2.image == s5.image) & (s5.image ==
s8.image) & (s2.image != NULL))
        {
            return YES;
        }
        if((s3.image == s6.image) & (s6.image ==
s9.image) & (s3.image != NULL))
        {
            return YES;
```

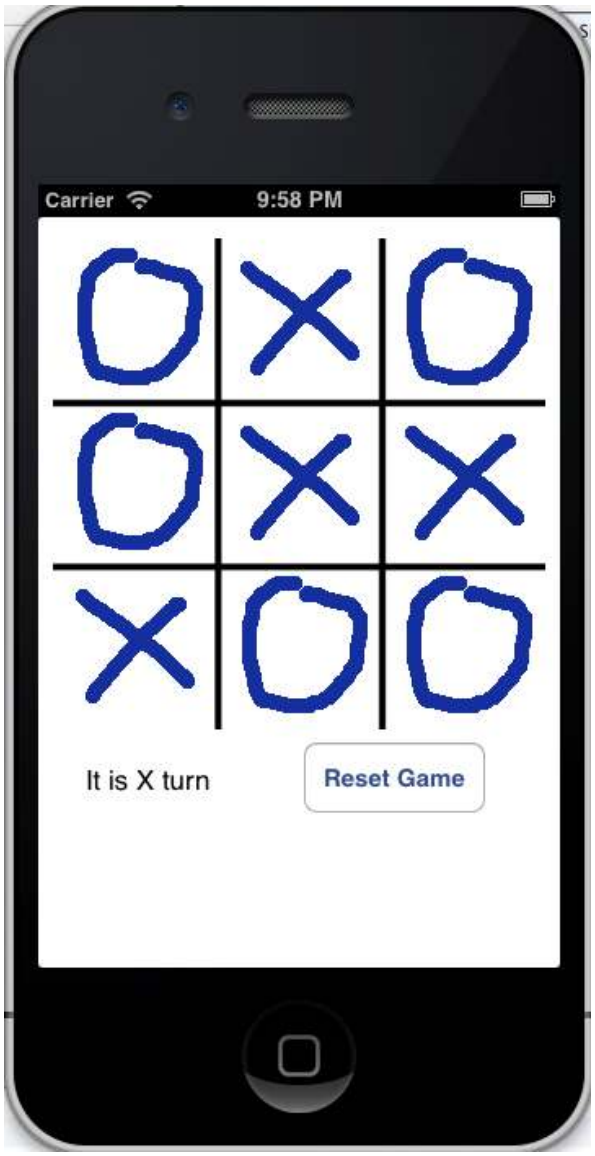
```
        }
        // DIAGONAL WINS
        if((s1.image == s5.image) & (s5.image ==
s9.image) & (s1.image != NULL))
        {
            return YES;
        }
        if((s3.image == s5.image) & (s5.image ==
s7.image) & (s3.image != NULL))
        {
            return YES;
        }
        return NO;
}
```

You now have to check for the win someplace. You could do it in updatePlayerInfo method. If so you could put something like this in that method:

```
if ([self checkForWin]) {
        UIAlertView *someonewon = [[UIAlertView
alloc]initWithTitle:@"There's a winner!"
message:@"Someone Won. You have to figure out
who and how you want to report it. I can't do
everything for you." delegate:self
cancelButtonTitle:@"ok" otherButtonTitles: nil];
            [someonewon show];
            [self resetBoard];
    }
```

Decide how you want to report the win and you have a working Tic-Tac-Toe game.

The final game looks like this:



Congratulations you have a working Tic-Tac-Toe game.