ATLS 4120/5120: Mobile Application Development
Week 8: Gestures

Create a single view app called Gestures.
Add an image to your project. Png is preferred so you don't have a background. (yoda2.png)
In the storyboard add an image view and set it to your image.
Resize the image view to match the size of your image. Editor | Size to fit content.
In the attributes inspector check User Interaction Enabled so the view will accept the touches.

Pan

Add a pan Gesture Recognizer and drag it on top of your Image View. This both creates the pan gesture recognizer, and associates it with the Image View.
Verify you got it connected OK by clicking on the Image View and go into the Connections Inspector and making sure the Pan Gesture Recognizer is in the gestureRecognizers Outlet Collection.
Create a connection for the pan gesture recognizer as an action called handlePan.

ViewController.swift
```
    @IBAction func handlePan(sender: UIPanGestureRecognizer) {
        let translation = sender.translationInView(view)
//returns the new location
        sender.view!.center = CGPoint(x: sender.view!.center.x +
translation.x, y: sender.view!.center.y + translation.y)
        sender.setTranslation(CGPointZero, inView: view)
//set the translation back to 0
    }
```

You should now be able to move your image around.

Snapshot: pan basic

Now let's add to this so we can add some deceleration when the move ends. We will detect when the gesture ends, figure out how fast the touch was moving, and animate the image moving to a final destination based on the touch speed.

Add the following to the end of handlePan

```
if sender.state == UIGestureRecognizerState.Ended { //when the move ends
            //figure out the velocity
            let velocity = sender.velocityInView(self.view)
            let magnitude = sqrt((velocity.x * velocity.x) + (velocity.y *
velocity.y))
            let slideMultiplier = magnitude / 200
    //if the length is < 200, then decrease the base speed, otherwise
increase it
            let slideFactor = 0.1 * slideMultiplier //increase for a greater
slide
            //calculate a final point based on the velocity and the
slideFactor
            var finalPoint = CGPoint(x:sender.view!.center.x + (velocity.x *
slideFactor), y:sender.view!.center.y + (velocity.y * slideFactor))
            //make sure the final point is within the view's bounds
            finalPoint.x = min(max(finalPoint.x, 0),
self.view.bounds.size.width)
            finalPoint.y = min(max(finalPoint.y, 0),
self.view.bounds.size.height)
            //animate the view
            UIView.animateWithDuration(Double(slideFactor * 2), delay: 0,
options: UIViewAnimationOptions.CurveEaseOut, animations:
{sender.view!.center = finalPoint }, completion: nil)
        }
```

Snapshot: pan deceleration

Pinch and Rotation
Now let's add pinch and rotation gestures.
Go into the storyboard and drag a pinch gesture recognizer and a rotation gesture recognizer on top of your image.
You can check these connections in the image view's connection inspector.
Create an action connection for pinch recognizer called handlePinch
Create an action connection for rotation recognizer called handleRotate

ViewController.swift
```
    @IBAction func handlePinch(sender: UIPinchGestureRecognizer) {
        sender.view!.transform =
CGAffineTransformScale(sender.view!.transform, sender.scale, sender.scale)
        sender.scale=1 //resets scale
    }

    @IBAction func handleRotate(sender: UIRotationGestureRecognizer) {
        sender.view!.transform =
CGAffineTransformRotate(sender.view!.transform, sender.rotation)
        sender.rotation=0 //reset rotation
    }
```

Simulator:
Alt click drag to zoom, go in a circle to rotate.

Snapshot: pinch rotate

Multiple gestures
By default, once one gesture recognizer on a view "claims" the gesture, no others can recognize a gesture from that point on.
Let's update it so we can have multiple gestures by overriding a function in the UIGestureRecognizer delegate.
Adopt the UIGestureRecognizer protocol.
```
class ViewController: UIViewController, UIGestureRecognizerDelegate
```

Implement this method, default is false.

```
    func gestureRecognizer(gestureRecognizer: UIGestureRecognizer,
shouldRecognizeSimultaneouslyWithGestureRecognizer otherGestureRecognizer:
UIGestureRecognizer) -> Bool {
        return true //allow multiple gestures to be recognized
    }
```

Main.storyboard
For each gesture recognizer connect its delegate outlet in the connections inspector to the view controller.

Now when you run it you can perform multiple gestures at the same time.

Snapshot: multiple gestures 2

Long Press
For the long press gesture we're going to play a short audio clip using the AVFoundation framework.
Add AVFoundation.framework to your apps target (build phases tab).
Copy Last of the Jedi.mp3 or your own into resources. Make sure you check Copy File and in the Utilities pane under Target Membership check media so it's included when the project is compiled.
(look at class reference documentation for supported formats)

Go into the storyboard and drag a long press recognizer on top of your image.
Connect the longpress delegate outlet in the connections inspector to the view controller.
Create an action connection for the long press recognizer called handleLongPress.

ViewController.swift
```
import AVFoundation
```

Create an instance variable for our audioplayer
```
    var audioPlayer : AVAudioPlayer?
```

```
@IBAction func handleLongPress(sender: UILongPressGestureRecognizer) {
        let audioFilePath = NSBundle.mainBundle().pathForResource("Last of 
the Jedi", ofType: "mp3")
        let fileURL = NSURL(fileURLWithPath: audioFilePath!)
        audioPlayer = AVAudioPlayer(contentsOfURL: fileURL, error: nil)
        if audioPlayer != nil{
            audioPlayer!.play()
        }

    }
```

Snapshot: audio

Reference: http://www.appcoda.com/ios-gesture-recognizers/