

Announcement

- **Lab 4 is due on Monday by 11:59 PM**

Today's Topics

- **Camera**
- **Gesture Recognition**
- **Audio and Video in iOS**

Using the Camera in iOS

iOS Camera Topics

- Take photos and videos with camera
- Save photos and videos to Camera Roll album
- Access photos and videos from Camera Roll

Taking pictures with the camera

- Instantiate and modally present the camera interface using `UIImagePickerController` class
- System supplied user interface for taking pictures and movies
- Image picker controller manages user interactions and delivers results to delegate object

Accessing Camera

- Use `UIImagePickerController` to access your camera
- Set `sourceType` to `.camera`

```
let picker = UIImagePickerController()  
picker.delegate = self  
picker.sourceType = .camera  
present (picker, animated: true, completion: nil)
```

Accessing Photo Library

- Use UIImagePickerController to access your photo library
- Set sourceType to .photoLibrary

```
let picker = UIImagePickerController()  
picker.delegate = self  
picker.sourceType = .photoLibrary  
present (picker, animated: true, completion: nil)
```

Saving content to Photo Library

- Use the UIImageWriteToSavedPhotosAlbum method to save an image to the photo library

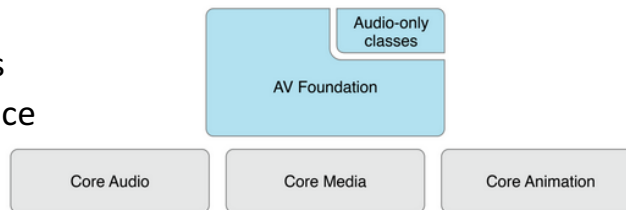
```
UIImageWriteToSavedPhotosAlbum(imageDisplay.image!, self,  
#selector(ViewController.image(image:didFinishSavingWithError:contextInfo:)), nil)
```

- Optionally compress the image with UIImageJPEGRepresentation

```
let imageData = UIImageJPEGRepresentation(imageDisplay.image!, 0.6)  
let compressedImage = UIImage(data: imageData)  
UIImageWriteToSavedPhotosAlbum(compressedImage!, self,  
#selector(ViewController.image(image:didFinishSavingWithError:contextInfo:)), nil)
```

AVFoundation

- Provides a large number of additional camera and video configurations
- Create an `AVCaptureSession`
 - Allows you coordinate data flow between inputs and outputs (microphone and camera to image or video)
- Supports many additional features
 - Exposure
 - Flash modes
 - White Balance

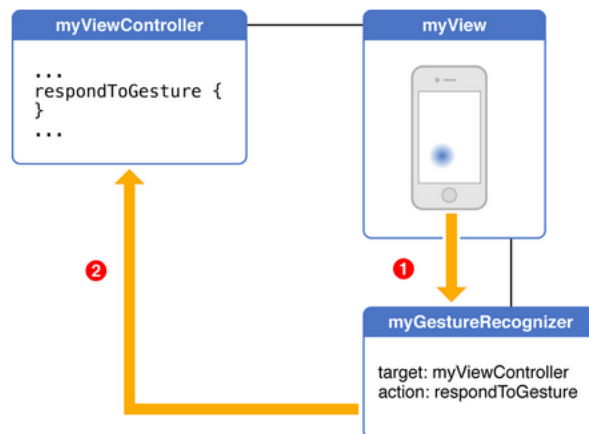


Camera Demo

Gesture Recognition in iOS

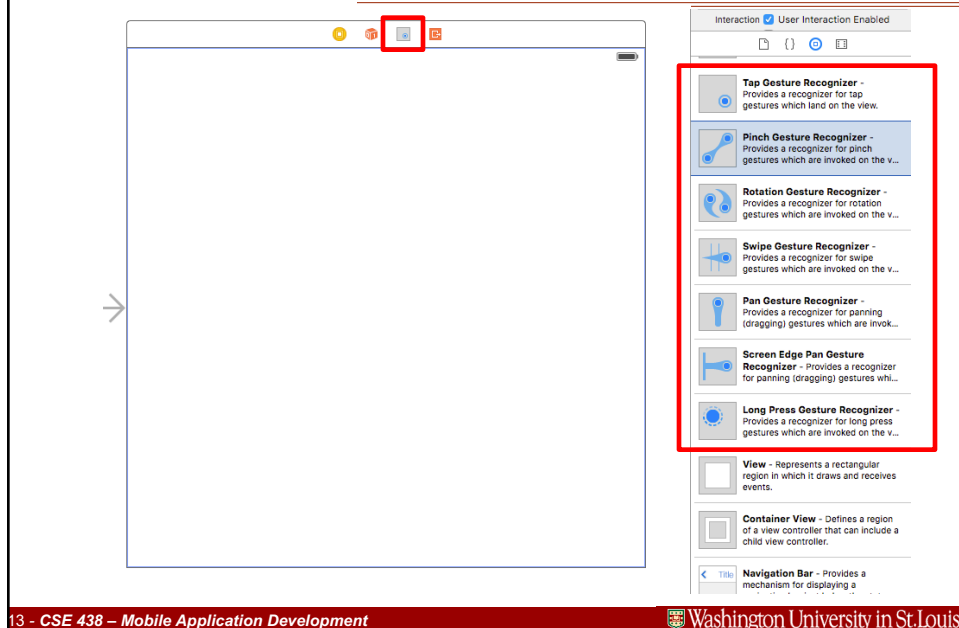
- iOS provides several built in gesture recognizers
 - Tapping
 - Pinching
 - Panning
 - Swiping
 - Long press
- No need to define your own “gestures” using touches began, touches moved, and touches ended

Gesture Recognition



https://developer.apple.com/library/content/documentation/EventHandling/Conceptual/EventHandlingiPhoneOS/GestureRecognizer_basics/GestureRecognizer_basics.html


Add gestures via Storyboard



The screenshot shows the Xcode Storyboard interface. On the left, a white rectangular view is shown with a red box highlighting the top-right corner. On the right, the 'Interaction' palette is visible, showing a list of gesture recognizers. A red box highlights the following items:

- Tap Gesture Recognizer - Provides a recognizer for tap gestures which land on the view.
- Pinch Gesture Recognizer - Provides a recognizer for pinch gestures which are invoked on the v...
- Rotation Gesture Recognizer - Provides a recognizer for rotation gestures which are invoked on the v...
- Swipe Gesture Recognizer - Provides a recognizer for swipe gestures which are invoked on the v...
- Pan Gesture Recognizer - Provides a recognizer for panning (dragging) gestures which are invoc...
- Screen Edge Pan Gesture Recognizer - Provides a recognizer for panning (dragging) gestures whi...
- Long Press Gesture Recognizer - Provides a recognizer for long press gestures which are invoked on the v...

Below these are other UI elements like View, Container View, and Navigation Bar.

13 - CSE 438 – Mobile Application Development  Washington University in St. Louis

Add Gestures Programmatically

- **//Implement right swipe detection**

```
let swipeRight = UISwipeGestureRecognizer(target: self, action:
#selector(ViewController.respondToSwipeGesture(_:)))
swipeRight.direction = .right
self.view.addGestureRecognizer(swipeRight)
```
- **//Implement down swipe detection**

```
let swipeDown = UISwipeGestureRecognizer(target: self, action:
#selector(ViewController.respondToSwipeGesture(_:)))
swipeDown.direction = .down
self.view.addGestureRecognizer(swipeDown)
```
- **//Handle swipe detection**

```
func respondToSwipeGesture(_ gesture: UIGestureRecognizer) {
}
```

Gestures Demo

Audio and Video in iOS

Uses for Audio

- **Sound effects**
 - button clicks
 - alert sounds
 - short sounds accompanying user actions
- **Arbitrary length sounds (music, podcasts, spoken content)**
- **Streamed content from web services**
- **Recording audio**

How to do it?

- **Could be complex:**
 - Potentially multiple simultaneous sources
 - Numerous possible outputs
 - Dynamic events, often out of user's control
 - Different priorities for seemingly similar actions
- **The OS manages the sound system**
 - You can ask for behavior, but the OS has control

CoreAudio

- **High level, easy to use**
 - System Sound API - short sounds
 - AVAudioPlayer class - ObjC, simple API
- **Lower level, takes more effort but much more control**
 - Audio Toolbox - recording and playback, streaming, full control
 - Audio Units - processing audio
 - OpenAL - 3D positional sound
- **Which one you use depends on what you're trying to do**
 - Many of you are fine with System Sounds and AVAudioPlayer

Playing Short Sounds

- **“short” means less than 5 seconds**
- **Very simple API, but has restrictions**
 - No looping
 - No volume control
 - Immediate playback
 - Limited set of formats
 - Linear PCM or IMA4
 - .caf, .aif or .wav file

Playing Short Sounds

- **Two step process**
 - Register the sound, get a “sound ID” in return
 - Play the sound
 - Optionally can get callback when sound finishes playing

```
var soundID: SystemSoundID
let fileURL = CFBundleCopyResourceURL(CFBundleGetMainBundle(), "mySound" as
    CFString!, "caff" as CFString!, nil)
// First register the sound
AudioServicesCreateSystemSoundID (fileURL!, &soundID);

// Then you can play the sound
AudioServicesPlaySystemSound (soundID);
```

Converting Sounds

- **Command line utility to convert sounds**
/usr/bin/afconvert
- **Supports wide variety of input and output formats**
- **See man page for details**
- **Easily convert sounds to System Sounds formats**

```
/usr/bin/afconvert -f aiff -d BE16 input.mp3 output.aif
```

AVAudioPlayer

- **Play longer sounds (> 5 seconds)**
- **Locally stored files or in-memory (no network streaming)**
- **Can loop, seek, play, pause**
- **Provides metering**
- **Play multiple sounds simultaneously**
- **Cocoa-style API**
 - Initialize with file URL or data
 - Allows for delegate
- **Supports many more formats**
 - Everything the AudioFile API supports

AVAudioPlayer

- **Create from file URL or data**
- **Simple methods for starting/stopping**

```
var player = AVAudioPlayer()
let url = Bundle.main.url(forResource: "myAudio", withExtension: "mp3")!
do {
    player = try AVAudioPlayer(contentsOf: url, fileTypeHint: nil)
}
catch let error as NSError {print(error.description) }

player.numberOfLoops = 1
player.prepareToPlay()
player.play()
```

AVAudioPlayerDelegate

- **Told when playback finishes**
- **Informed of audio decode errors**
- **Given hooks for handling interruptions**
 - Incoming phone calls

OpenAL

- **High level, cross-platform API for 3D audio mixing**
 - Great for games
 - Mimics OpenGL conventions
- **Models audio in 3D space**
 - Buffers: Container for Audio
 - Sources: 3D point emitting Audio
 - Listener: Position where Sources are heard
- **More Information: <http://www.openal.org/>**

Frameworks and APIs

- **MediaPlayer**
 - Allows you to find and player user-installed media items
 - Songs, podcasts, audio books
 - Play custom movies
 - Cut scenes in games
- **AVFoundation**
 - Record, edit, and play audio and video
 - Configure audio sessions
 - Respond to changing device environment
- **AudioToolbox**
 - Record or play audio
 - Convert formats
 - Parse audio streams
 - Configure audio session

Playing Video

- **Uses for Video:**
 - Provide cut-scene animation in a game
 - Stream content from web sites
 - Play local movies
- **Play videos from application bundle or remote URL**
 - Always full screen
 - Configurable scaling modes
 - Optional controls
- **Supports:**
 - .mov, .mp4, .m4v, .3gp

Audio Demo

Video Demo