CS 47

Beginning iPhone Application Development

Week 7: Image/Camera, Accelerometer and Location

Office Hours

- Tuesday, March 2nd
- 7pm-8pm
- Don’t know where yet

Rubehouse

Agenda

- Device-oriented features
- Camera/Images
- Accelerometer
- Location

Camera/Images

- This topic covers anything related to images taken/residing on the device:
  - Still Picture Capture
  - Movie Capture
  - Editing
  - Photo Library

Photo Library

- Unfortunately you cannot access the photo library directly
- You must pop up an encapsulated view controller that allows the user to select an image and pass it to your application

Photo Library

- But you **can** programmatically save an image to the photo library

```
UIImage *image = ...;
UIImageWriteToSavedPhotosAlbum(image, nil, nil, NULL);
```

- Does this in the background, so you should pop up an authorization dialog to make the user wants this to happen

Photo Library

- So how do you get an image from the photo library?
- The same way you get a picture or movie from the camera
- UIImagePickerController

UIImagePickerController

- The UIImagePickerController is nothing more than a custom subclass of UINavigationController

- When you want to use one, just instantiate it like a normal view controller

```
UIImagePickerController *picker =
      [[[UIImagePickerController alloc] init] autorelease];
```

UIImagePickerController

- One thing you need to remember
- You cannot push a UINavigationController onto another UINavigationController
- So you cannot push a UIImagePickerController onto an existing UINavigationController flow

UIImagePickerController

- However, you can make it a tab inside of a UITabBarController
- Or you can add the UIImagePickerController's view to an existing view hierarchy and transition it in with an animation
- Or, most commonly, present it modally

UIImagePickerController

- Presenting a UIImagePickerController modally

```
/* Inside of an existing view controller */
...
UIImagePickerController *picker = ...;
[self presentModalViewController:picker animated:YES];
...
```

- The view controller pops up from the bottom

UIImagePickerController

- UIImagePickerControllers should have a delegate that implements the UIImagePickerControllerDelegate interface

```
UIImagePickerController *picker = ...;
picker.delegate = self;
```

- The delegate is normally the view controller that wants to show the image picker
- Handles cancel and selection

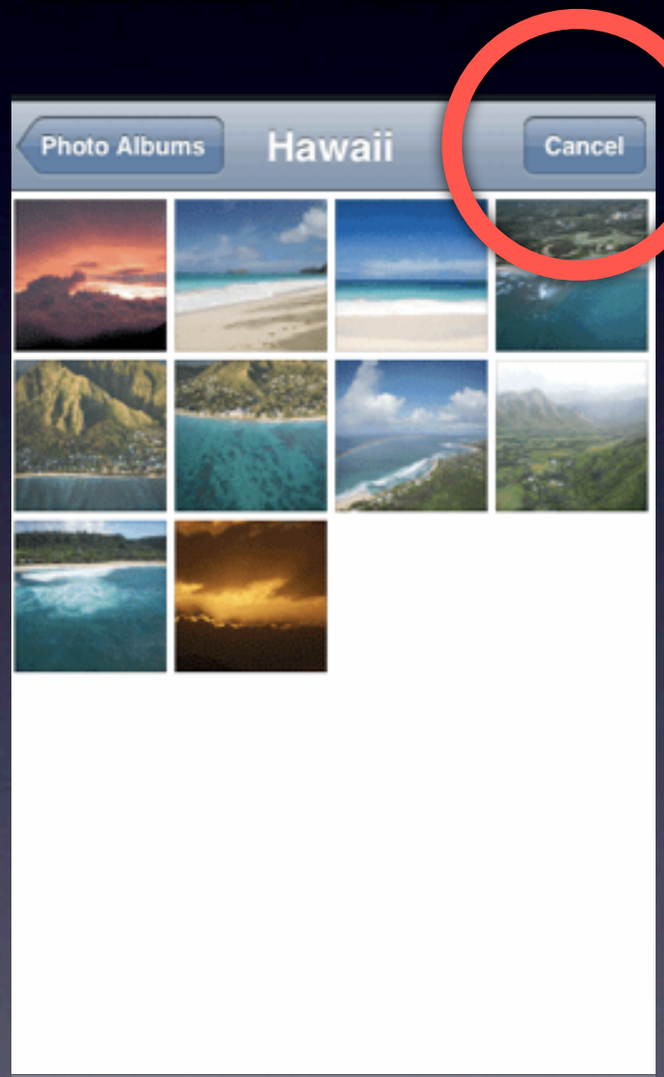
```
- (void)imagePickerControllerDidCancel:(UIImagePickerController *)picker

- (void)imagePickerController:(UIImagePickerController *)picker
          didFinishPickingMediaWithInfo:(NSDictionary *)info
```

UIImagePickerController

- The user hits the cancel button and you get the cancel callback
- Get the UIImagePickerController view out of the screen

```
[self dismissModalViewControllerAnimated:YES];
```

- If you presented an autoreleased controller, it will clean up automatically
- (or you can use your own transition)



UIImagePickerController

- If the user selects an image, the delegate receives didFinishPickingMediaWithInfo
- The return value is an NSDictionary filled in depending on what kind of media you were picking

```
UIImagePickerControllerMediaType      => NSString (kUTTypeImage/Movie)
UIImagePickerControllerOriginalImage => UIImage
UIImagePickerControllerEditedImage    => UIImage
UIImagePickerControllerCropRect       => NSValue (contains CGRect)
UIImagePickerControllerMediaURL       => NSURL (points to movie file)
```

UIImagePickerController

- Even if you get the selection callback, you still need to perform the view dismissal (like in the cancel callback)

UIImagePickerController

- How do you tell the picker what type of media to get?

```
UIImagePickerController *picker = ...;
picker.sourceType =
        UIImagePickerControllerSourceTypePhotoLibrary
        UIImagePickerControllerSourceTypeCamera
        UIImagePickerControllerSourceTypeSavedPhotosAlbum
```

- You can change this value dynamically, even while the picker is displayed

UIImagePickerController

- How do you know if a source type is available?

```
[UIPickerViewController isSourceTypeAvailable:xxx];
```

- Do not attempt to show a picker with a source type that is not supported
- Apple says you shouldn't even offer the source option if it's not available

UIImagePickerController

- What about a specific media type from within a source?

```
[UIPickerViewController availableMediaTypesForSourceType:xxx];
```

- Gives back an array of NSString objects
- Right now the only options are kUTTypeImage and kUTTypeMovie
- Use this to fill in the `mediaTypes` property for what the picker can display

UIImagePickerController

- What if we want to edit the media?

```
UIImagePickerController *picker = ...;
picker.allowsEditing            = YES; /* Default: NO */
```

- (Set this before you show it)
- For images: you can scale and crop
- For movies: you can trim the ends
- Adds an additional step to the picker flow once the media is selected

UIImagePickerController

- What about making our own custom camera interface? (e.g. RedLaser)
- Turn off default camera controls

```
picker.showCameraControls = NO; /* Not necessary */
```

- Add your own overlay

```
picker.cameraOverlayView = ...;
```

- Take pictures!

```
[picker takePicture];
```

UIImagePickerController

- Important note on images taken with the camera
- The iPhone does some insane internal photo configuration
- If you take a UIImage from the camera and use UIImagePNGRepresentation to save the image to disk, you cannot simply upload that PNG to a server

UIImagePickerController

- Need to perform some special scale/rotate magic on the image so that it appears right-side-up (and not huge)
- Use the scaleAndRotate function in the sample code for this week (it's magic from the internet)

Accelerometer

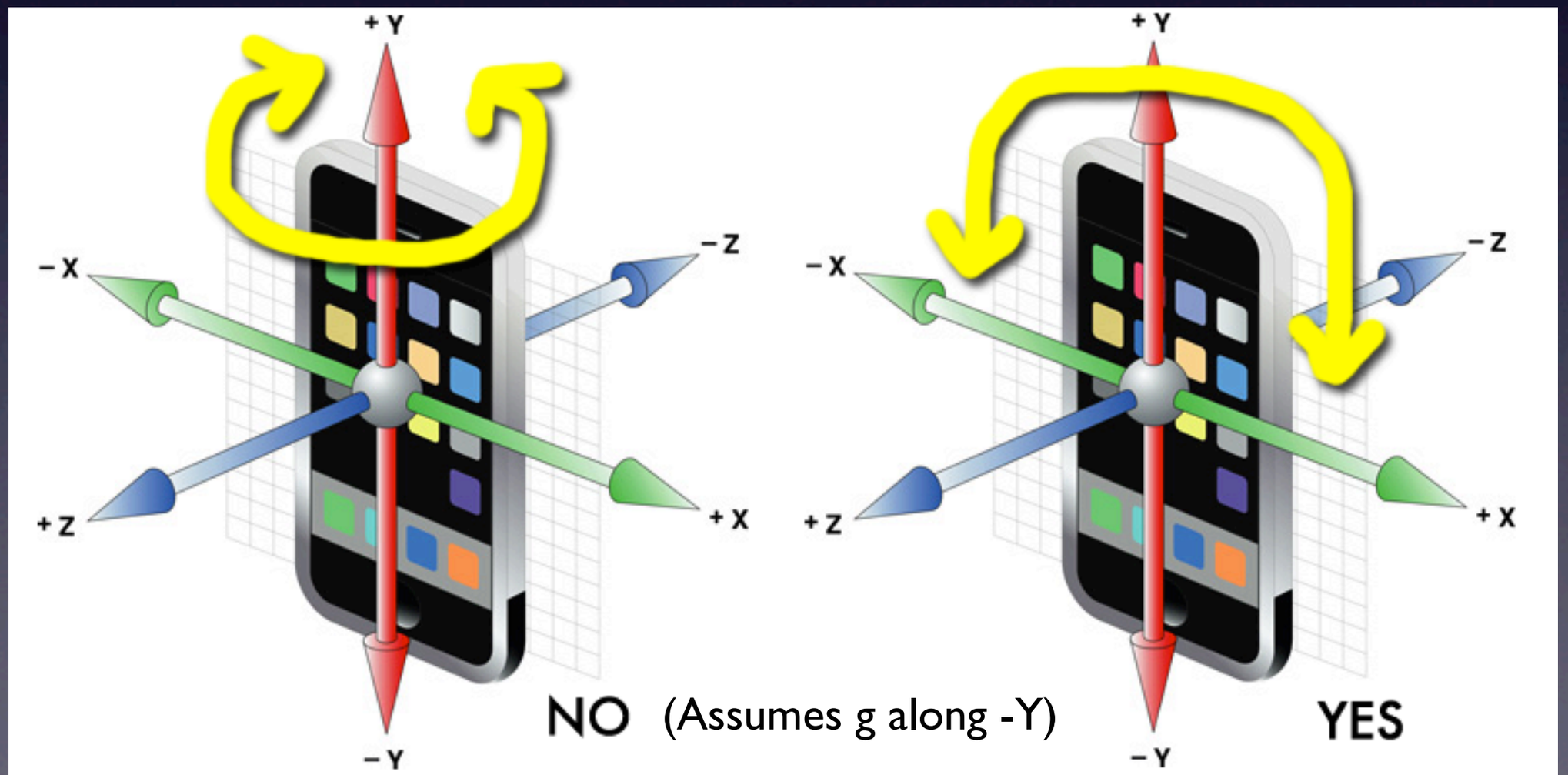
- Detects **force** along the x, y and z axis
- Create a force measurement by accelerating the device along an axis
- Remember, gravity provides a continuous force of 1g towards the Earth

Accelerometer

- What is it good for?
- Tilt games (Labyrinth)
- Fake drinking (iBeer)
- Gravity reckoning (bubble measure)
- Detect motion/shaking/bumping

Accelerometer

- What can't we measure with this?
- Rotation about the axis created by gravity



Accelerometer

- Other things we can't measure with the Accelerometer
  - Which direction we're facing
  - Constant velocity
  - Not precise enough to do velocity reckoning by tracking acceleration

Accelerometer

- The accelerometer uses the singleton design pattern. There is only one accelerometer object in the system

```
UIAccelerometer *a = [UIAccelerometer sharedAccelerometer];
```

- Where have we seen this before?
- A lot of device-specific classes use this (UIDevice, MPMusicPlayerController)

Accelerometer

- Turn on the accelerometer by assigning an delegate and updateInterval

```
[UIAccelerometer sharedAccelerometer].delegate        = self;
[UIAccelerometer sharedAccelerometer].updateInterval = 0.1;
```

- Turn off the accelerometer by setting the delegate to nil
- You can only set updateInterval when the delegate is non-nil

Accelerometer

- Since we are changing the delegate of a shared object, you can only have one active delegate at a time
- If you need to send accelerometer data to multiple receivers, you should create a single marshaling object that has an array of delegates

Accelerometer

- The delegate must conform to UIAccelerometerDelegate, and implement

```
- (void)accelerometer:(UIAccelerometer *)accelerometer
        didAccelerate:(UIAcceleration *)acceleration
```

- Gets called in the main thread
- Use .x, .y and .z properties to see force along those axis (value is multiples of g)
- Use .timestamp for time-based calculations

Core Location

- Core Location is comprised of the AGPS unit and the magnetic compass
  - AGPS: find your coordinates, altitude, speed and course (with reported accuracy)
  - Compass: find your magnetic and true heading (with reported accuracy)

AGPS

- What is Assisted GPS?
- It means you do not have a full GPS unit on the phone. It does not have the satellite library built in, and it cannot output location by itself every second
- It must communicate with a central server to assist in GPS calculations

AGPS

- So if you want GPS-based location calculations, you must have some sort of network connection (wifi or cell)
- Does not work in Airplane Mode
- Kind of slow

AGPS

- Assisted GPS progressively approximates your location
- So you will get an initial inaccurate location (e.g. 1km accuracy, perhaps based on cell triangulation), and then over time get locations with better and better accuracy

CLLocationManager

- Location is managed by... CLLocationManager
- This is not a singleton class, you need to allocate each location manager that you are going to use

```
CLLocationManager *manager = [[CLLocationManager alloc] init];
```

- If you need multiple outlets for location updates, you should still use a marshaling object on a single location manager

CLLocationManager

- The CLLocationManager gives you access to both the AGPS unit and the magnetometer
- AGPS = “Location”
- Magnetometer = “Heading”
    - Not to be confused with course from AGPS

CLLocationManager

- The location manager can be turned off globally in the iPhone settings
- Check `locationManager.locationServicesEnabled` before attempting to use them
- Even if enabled globally, the user can deny your app access to location
- Users must say yes three times before it's automatically enabled)

CLLocationManager

- You need to set a few things

```
CLLocationManager *manager = ...;
manager.delegate           = self;
manager.distanceFilter     = kCLDistanceFilterNone;   (m)
manager.headingFilter      = kCLHeadingFilterNone;    (deg)
manager.desiredAccuracy    = kCLLocationAccuracyBest; (m)
```

- And then start updating

```
if (manager.locationServicesEnabled) {
  [manager startUpdatingLocation];

  if (manager.headingAvailable) {
    [manager startUpdatingHeading];
  }
}
```

CLLocationManager

- Now your delegate will get callbacks when new locations/headings are detected

```
- locationManager:didUpdateToLocation:fromLocation:
- locationManager:didUpdateHeading:
- locationManagerShouldDisplayHeadingCalibration:
- locationManager:didFailWithError:
```

CLLocationManager

- The locations you get back are wrapped in CLLocation objects

```
coordinate            => CLLocationCoordinate2D (lat, lon)
altitude              => CLLocationDistance (double, meters)
horizontalAccuracy    => CLLocationDistance (double, meters)
verticalAccuracy      => CLLocationDistance (double, meters)
timestamp             => NSDate
speed                 => CLLocationSpeed (double, m/s)
course                => CLLocationDirection (double, degrees)
```

- Negative values are invalid

CLLocationManager

- The headings you get back are wrapped in CLHeading objects

```
magneticHeading  => CLLocationDirection (double, degrees)
trueHeading      => CLLocationDirection (double, degrees)
headingAccuracy  => CLLocationDirection (double, degrees)
timestamp        => NSDate

x, y, z          => CLHeadingComponentValue (double, raw)
```

CLLocationManager

- Cached location values are a problem
- You may get "old" readings sent to your delegate - presumably just to get some data to you ASAP
- You should filter out new locations by their timestamp property to make sure they are recent

```
if (fabs([newLocation.timestamp timeIntervalSinceNow]) > 5) return;
```

CLLocationManager

- Stop the location manager when you don't need to be updated anymore

```
[manager startUpdatingLocation];
[manager startUpdatingHeading];
```

- Leaving this on will thrash your battery and make your phone hot
- Best practice is to turn it on intermittently