Core Location, MapKit & MediaPlayer

Gloria Hedlund
Teaching Fellow
CS76 – Building Mobile Applications
Harvard Extension School
Core Location Framework

• There is no User Interface

• Basic Object: CLLocation
  • CLLocationCoordinate2D coordinate (struct)
  • CLLocationDistance altitude (meters)
  • CLLocationSpeed speed (meters/second)
  • CLLocationDirection course (degrees, 0 = north)
  • NSDate *timestamp
@property (readonly) CLLocationCoordinate2D coordinate;

typedef {
    CLLocationCoordinateDegrees latitude;    //double
    CLLocationCoordinateDegrees longitude;
} CLLocationCoordinate2D;

// This is an approximation... Dependent on how much accuracy you specify that you need
Core Location Accuracy

@property (readonly) CLLocationAccuracy horizontalAccuracy;
@property (readonly) CLLocationAccuracy verticalAccuracy;

- measured in meters
kCLLocationAccuracyBestForNavigation;  // will drain your battery
kCLLocationAccuracyBest;  // GPS
kCLLocationAccuracyNearestTenMeters;  // Wifi
kCLLocationAccuracyHundredMeters;
kCLLocationAccuracyKilometer;  // uses cell towers – low power
kCLLocationAccuracyThreeKilometers;

(set this as low as possible for your application)
To use Core Location

- Import framework and `<CoreLocation/CoreLocation.h>`
- Create a CLLocationManager & set yourself as a `<CLLocationManagerDelegate>` delegate
- Check to see what hardware is available on device
- Configure manager to the type of location updating that you want (if it's available)
- Start the manager monitoring for location changes
Hardware Check

+ (BOOL)locationServicesEnabled; //also check for user permission
+ (BOOL)headingAvailable;  //compass?
+ (BOOL)significantLocationChangeMonitoringAvailable; //cellular?
+ (BOOL)regionMonitoringAvailable; //only certain devices
+ (BOOL)regionMonitoringEnabled;

DON’T FORGET TO CHECK THIS STUFF!

@property (copy) NSString *purpose; //use to ask user permission
Core Location Monitoring

• Choose type of monitoring
  • @property CLLocationAccuracy desiredAccuracy
  • @property CLLocationDistance distanceFilter;

• Start/Stop monitoring
  • (void)startUpdatingLocation;
  • (void)stopUpdatingLocation;

• Receive Notifications
  - (void)locationManager:(CLLocationManager *)manager
didUpdateToLocation:(CLLocation *)newLocation
    fromLocation:(CLLocation *)oldLocation;
MapKit Framework

• Import Framework and `<Mapkit/Mapkit.h>`
• MKMapView
  • UIView that displays a map
• Annotations (pins or markers on the map)
  • NSArray of Annotation objects
  • Annotation must implement `<MKAnnotation>` protocol
  • Coordinate, title, subtitle
• Annotations can callout (when you click on a pin)
  • Shows title and subtitle by default
  • Can have left and right accessory views
MapKit Annotation w/ callout

Seen here is an Annotation Callout with a leftCalloutAccessoryView that is an UIImage (a store logo) and a rightCalloutAccessoryView that is a UIButton of type UIButtonTypeDetailDisclosure.

There is also a Title and Subtitle showing in the callout (you get that for free).
<MKAnnotation>

- Use of protocol that tells objects what methods and properties they must implement
  - @property CLLocationCoordinate2D coord;
  - @property NSString *title; //optional
  - @property NSString *subtitle; //optional

- To add/remove annotations to your MKMapView
  - (void)addAnnotation:(id <MKAnnotation>)annotation;
  - (void)addAnnotations:(NSArray *) annotations;
  - (void)removeAnnotation:(id <MKAnnotation>)annotation;
  - (void)removeAnnotations:(NSArray *) annotations;
MKMapViewDelegate

Create an annotation:
- (MKAnnotationView *)mapView:(MKMapView *)sender
  viewForAnnotation:(id <MKAnnotation>)annotation

When an annotation is selected:
- (void)mapView:(MKMapView *)sender
didSelectAnnotationView:(MKAnnotationView *) view;

When a callout accessory view is touched:
- (void) mapView:(MKMapView *)sender
  annotationView:(MKAnnotationView *)view
calloutAccessoryControlTapped:( UIControl *)control
MKAnnotationView

@property id <MKAnnotation> annotation;
@property UIImage *image;  // to set image other than a pin
@property (retain) UIView *leftCalloutAccessoryView;
@property (retain) UIView *rightCalloutAccessoryView;
@property BOOL enabled;   // NO it will ignore being selected
@property CGPoint centerOffset;
@property BOOL draggable;  // annotation MUST implement setCoordinate
Creating Annotations

- (MKAnnotationView *)mapView: (MKMapView *)mapView
  viewForAnnotation: (id<MKAnnotation>)annotation
{
  // check and see if there are any markers available in our re-use pool
  MKPinAnnotationView *pin = (MKPinAnnotationView *)[mapView
      dequeReusableAnnotationViewWithIdentifier:@"Pin"];
  // if there are none available, make a new one
  if (!pin) {
      pin = [[MKPinAnnotationView alloc] initWithAnnotation:annotation
                      reuseIdentifier:@"Pin"];
  }
  // yes, if this is not a dequeue, this is set twice to annotation
  pin.annotation = annotation;
  return pin;
}
Overlays

- MKOverlayView
  - (void)addOverlay:(id <MKOverlay>)overlay;
  - (void)removeOverlay:(id <MKOverlay>)overlay;

- <MKOverlay> protocol
  - Specific type of annotation with a point and map area

- To create an Overlay
  (MKOverlayView *)mapView:(MKMapView *)mapView
  viewForOverlay:(id)overlay
MediaPlayer Framework

- Import Framework & <MediaPlayer/MediaPlayer.h>
- MPMoviePlayerController
- Uses NSNotificationCenter (instead of delegation)
MPMoviePlayerController Notifications

- MPMoviePlayerContentPreloadDidFinishNotification
- MPMoviePlayerPlaybackDidFinishNotification
- MPMoviePlayerPlaybackStateDidChangeEvent
- MPMoviePlayerScalingModeDidChangeNotification