Core Location and MapKit

Kevin Y. Kim

http://apporchard.com

Twitter: **@kykim**
Blog: **http://kykim.com**
Email: **kykim@me.com** or **me@kykim.com**

Who Am I?

- Co-Founder & Developer at AppOrchard
- Worked at Apple (WebObjects)
- Finance, Government, Embedded, Supercomputing
- More iOS6 Development (Apress, 2012)

Agenda

- Basic Geography
- Core Location
- Map Kit
- Demo
- Lab/Exercise

Geography

- Earth is spherical (essentially)
- Latitude and Longitude
- Mercator projection

Core Location

- Device Location
- Device Heading
- Geofencing

Core Location

- Basic Object is CLLocation
- Properties

```
@property(readonly, nonatomic) CLLocationCoordinate2D coordinate;
@property(readonly, nonatomic) CLLocationDistance altitude;
@property(readonly, nonatomic) CLLocationAccuracy horizontalAccuracy;
@property(readonly, nonatomic) CLLocationAccuracy verticalAccuracy;
@property(readonly, nonatomic) CLLocationDirection course;
@property(readonly, nonatomic) CLLocationSpeed speed;
@property(readonly, nonatomic) NSDate *timestamp;
```

Core Location Data Types

```
typedef struct {
   CLLocationDegrees latitude;
   CLLocationDegrees longitude;
} CLLocationCoordinate2D;
```

```
typedef double CLLocationDegrees; (Latitude or Longitude in degrees)
```

```
typedef double CLLocationDistance; (Distance in meters)
```

```
typedef double CLLocationAccuracy; (Accuracy in meters, lower is better, negative is invalid)
```

```
typedef double CLLocationDirection; (0-359.9, think compass heading, negative is invalid)
```

```
typedef double CLLocationSpeed; (Speed in meters per second)
```

CLLocationManager

- Handles location and heading events
- Notifies your application via delegate
- Provides class methods for checking service availability

Using CLLocationManager

1. Check Availability of Services
2. Create instance of CLLocationManager
3. Assign delegate
4. Configure additional properties
5. Invoke appropriate start method

CLLocationManager Check Availability of Services

- Via class (static) methods

```
+ (BOOL)locationServicesEnabled;
+ (BOOL)headingAvailable;
+ (BOOL)significantLocationChangeMonitoringAvailable;
+ (BOOL)regionMonitoringAvailable;
+ (CLAuthorizationStatus)authorizationStatus;
```

```
typedef enum {
    kCLAuthorizationStatusNotDetermined = 0,
    kCLAuthorizationStatusRestricted,
    kCLAuthorizationStatusDenied,
    kCLAuthorizationStatusAuthorized
} CLAuthorizationStatus;
```

- Handle cases when service not available

Create CLLocationManager instance

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

That's it

CLLocationManagerDelegate

```
- (BOOL)locationManagerShouldDisplayHeadingCalibration:(CLLocationManager *)manager;
- (void)locationManager:(CLLocationManager *)manager didFailWithError:(NSError *)error;
- (void)locationManager:(CLLocationManager *)manager
      didChangeAuthorizationStatus:(CLAuthorizationStatus)status;
- (void)locationManagerDidPauseLocationUpdates:(CLLocationManager *)manager;
- (void)locationManagerDidResumeLocationUpdates:(CLLocationManager *)manager;
- (void)locationManager:(CLLocationManager *)manager
      didFinishDeferredUpdatesWithError:(NSError *)error;
```

CLLocationManager properties

```
@property(assign, nonatomic) CLActivityType activityType;
@property(assign, nonatomic) CLLocationDistance distanceFilter;
@property(assign, nonatomic) CLLocationAccuracy desiredAccuracy;
@property(assign, nonatomic) BOOL pausesLocationUpdatesAutomatically;
@property(assign, nonatomic) CLLocationDegrees headingFilter;
@property(assign, nonatomic) CLDeviceOrientation headingOrientation;
```

```
enum {
    CLActivityTypeOther = 1,
    CLActivityTypeAutomotiveNavigation,
    CLActivityTypeFitness,
    CLActivityTypeOtherNavigation
};
typedef NSInteger CLActivityType;
```

```
typedef enum {
    CLDeviceOrientationUnknown = 0,
    CLDeviceOrientationPortrait,
    CLDeviceOrientationPortraitUpsideDown,
    CLDeviceOrientationLandscapeLeft,
    CLDeviceOrientationLandscapeRight,
    CLDeviceOrientationFaceUp,
    CLDeviceOrientationFaceDown
} CLDeviceOrientation;
```

Maps to UIDeviceOrientation

CLLocationAccuracy

```
extern const CLLocationAccuracy kCLLocationAccuracyBestForNavigation;
extern const CLLocationAccuracy kCLLocationAccuracyBest;
extern const CLLocationAccuracy kCLLocationAccuracyNearestTenMeters;
extern const CLLocationAccuracy kCLLocationAccuracyHundredMeters;
extern const CLLocationAccuracy kCLLocationAccuracyKilometer;
extern const CLLocationAccuracy kCLLocationAccuracyThreeKilometers;
```

Other CLLocationManager properties

```
@property (readonly, nonatomic) CLLocation *location;
@property (readonly, nonatomic) CLHeading *heading;
@property (readonly, nonatomic) CLLocationDistance maximumRegionMonitoringDistance;
@property (readonly, nonatomic) NSSet *monitoredRegions;
```

Invoke Appropriate CLLocationManager start method

```
- (void)startUpdatingLocation;
- (void)stopUpdatingLocation;

- (void)startUpdatingHeading;
- (void)stopUpdatingHeading;

- (void)startMonitoringSignificantLocationChanges;
- (void)stopMonitoringSignificantLocationChanges;

- (void)startMonitoringForRegion:(CLRegion *)region;
- (void)stopMonitoringForRegion:(CLRegion *)region;
```

CLLocationManagerDelegate

- For Location

```
- (void)startUpdatingLocation;
- (void)startMonitoringSignificantLocationChanges;

- (void)locationManager:(CLLocationManager *)manager
    didUpdateToLocation:(CLLocation *)newLocation
           fromLocation:(CLLocation *)oldLocation;

- (void)locationManager:(CLLocationManager *)manager
    didUpdateLocations:(NSArray *)locations;
```

CLLocationManagerDelegate

- For Heading `- (void)startUpdatingHeading;`

```
- (void)locationManager:(CLLocationManager *)manager
       didUpdateHeading:(CLHeading *)newHeading;
```

CLLocationManagerDelegate

- For Region `- (void)startMonitoringForRegion:(CLRegion *)region;`

```
- (void)locationManager:(CLLocationManager *)manager
        didEnterRegion:(CLRegion *)region;
- (void)locationManager:(CLLocationManager *)manager
         didExitRegion:(CLRegion *)region;
```

```
- (void)locationManager:(CLLocationManager *)manager
        monitoringDidFailForRegion:(CLRegion *)region
        withError:(NSError *)error;
```

Geocoding

- Forward Geocoding

  User description (i.e. address) to Latitude/Longitude

- Reverse Geocoding

  Latitude/Longitude to User description (i.e. address)

- CLGeocoder

Forward Geocoding

- CLGeocoder

```
- (void)geocodeAddressString:(NSString *)addressString
           completionHandler:(CLGeocodeCompletionHandler)completionHandler;

- (void)geocodeAddressString:(NSString *)addressString
                    inRegion:(CLRegion *)region
           completionHandler:(CLGeocodeCompletionHandler)completionHandler;
```

Forward Geocoding continued

```
- (void)geocodeAddressDictionary:(NSDictionary *)addressDictionary
               completionHandler:(CLGeocodeCompletionHandler)completionHandler;
```

- AddressBook framework

  <AddressBook/ABPerson.h>

```
AB_EXTERN const CFStringRef kABPersonAddressStreetKey;
AB_EXTERN const CFStringRef kABPersonAddressCityKey;
AB_EXTERN const CFStringRef kABPersonAddressStateKey;
AB_EXTERN const CFStringRef kABPersonAddressZIPKey;
AB_EXTERN const CFStringRef kABPersonAddressCountryKey;
AB_EXTERN const CFStringRef kABPersonAddressCountryCodeKey;
```

ReverseGeocoding

```
- (void)reverseGeocodeLocation:(CLLocation *)location
             completionHandler:(CLGeocodeCompletionHandler)completionHandler;
```

CLGeocodeCompletionHandler

- Block

```
typedef void (^CLGeocodeCompletionHandler)(NSArray *placemarks, NSError *error);
```

- Array of CLPlacemark objects

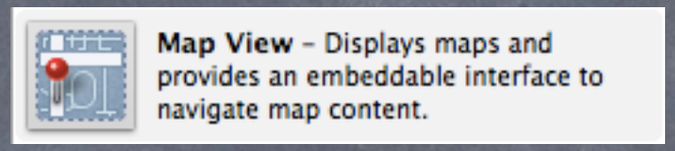
```
@interface CLPlacemark : NSObject <NSCopying, NSCoding>
@property (nonatomic, readonly) CLLocation *location;
@property (nonatomic, readonly) CLRegion *region;
@property (nonatomic, readonly) NSDictionary *addressDictionary;
@property (nonatomic, readonly) NSString *name;
@property (nonatomic, readonly) NSString *thoroughfare;
@property (nonatomic, readonly) NSString *subThoroughfare;
@property (nonatomic, readonly) NSString *locality;
@property (nonatomic, readonly) NSString *subLocality;
@property (nonatomic, readonly) NSString *administrativeArea;
@property (nonatomic, readonly) NSString *subAdministrativeArea;
@property (nonatomic, readonly) NSString *postalCode;
@property (nonatomic, readonly) NSString *ISOcountryCode;
@property (nonatomic, readonly) NSString *country;
@property (nonatomic, readonly) NSString *inlandWater;
@property (nonatomic, readonly) NSString *ocean;
@property (nonatomic, readonly) NSArray *areasOfInterest;
@end
```

MapKit

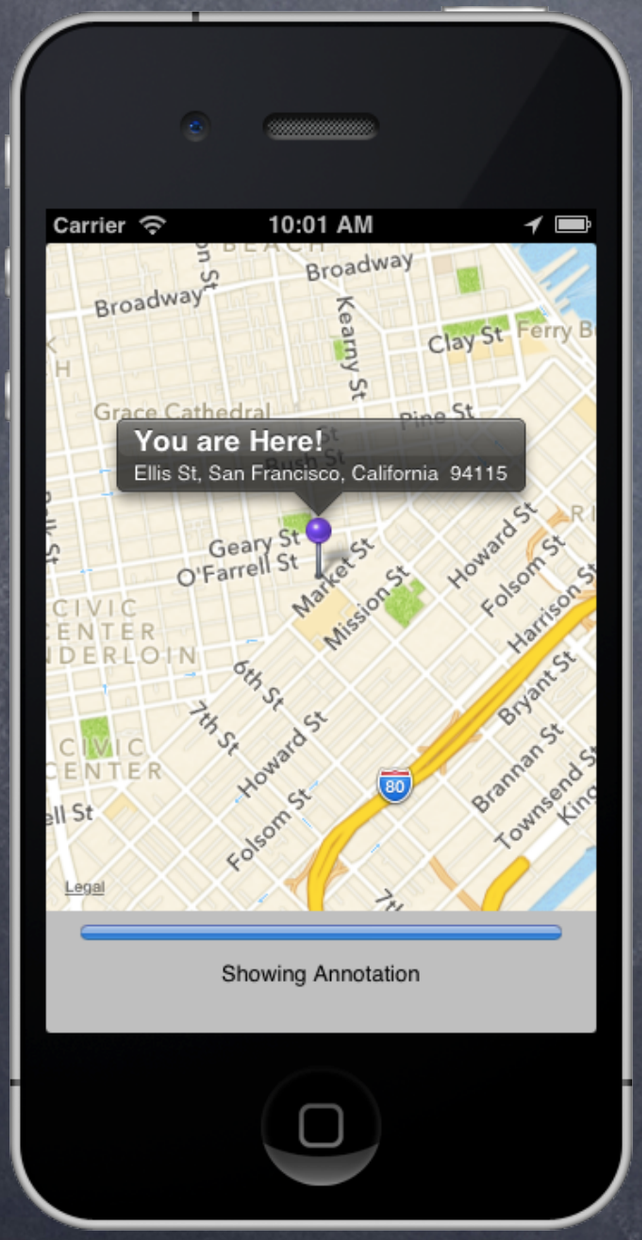
- Displays Map on device
- Fundamentals
  - Pan & Zoom
  - Annotations
  - Locations
  - Overlays
- MKMapView

MKMapView

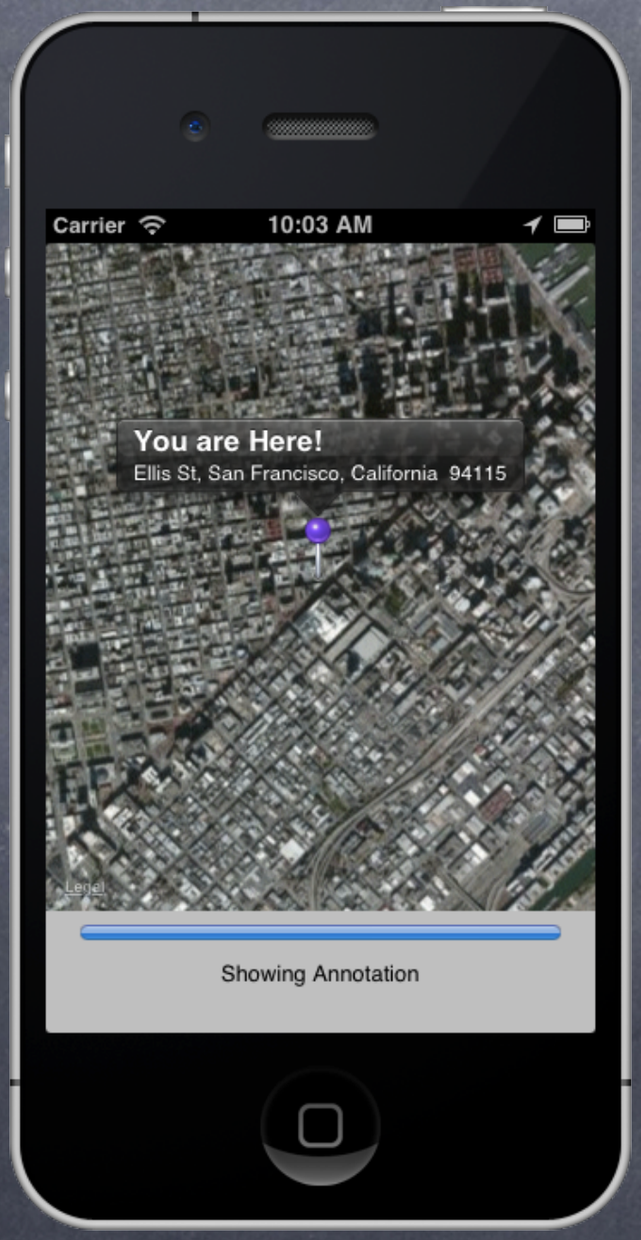
- UIView subclass
- Usually use via IB
- Easy to use
- Highly performant
- Don't subclass



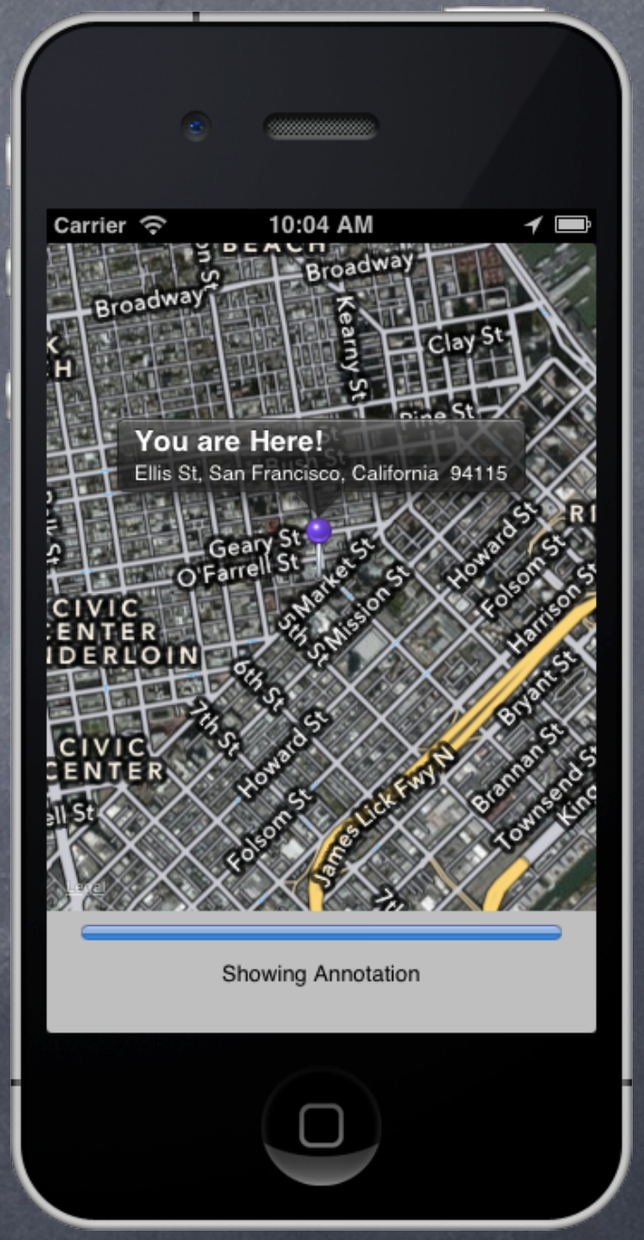
Map Types



Standard



Satellite



Hybrid

Map Types

- MKMapType

```
enum {
    MKMapTypeStandard = 0,
    MKMapTypeSatellite,
    MKMapTypeHybrid
};
typedef NSUInteger MKMapType;
```

- Property in MKMapView

```
@property (nonatomic) MKMapType mapType;
```

Showing User Location

- showsUserLocation property
- Boolean property

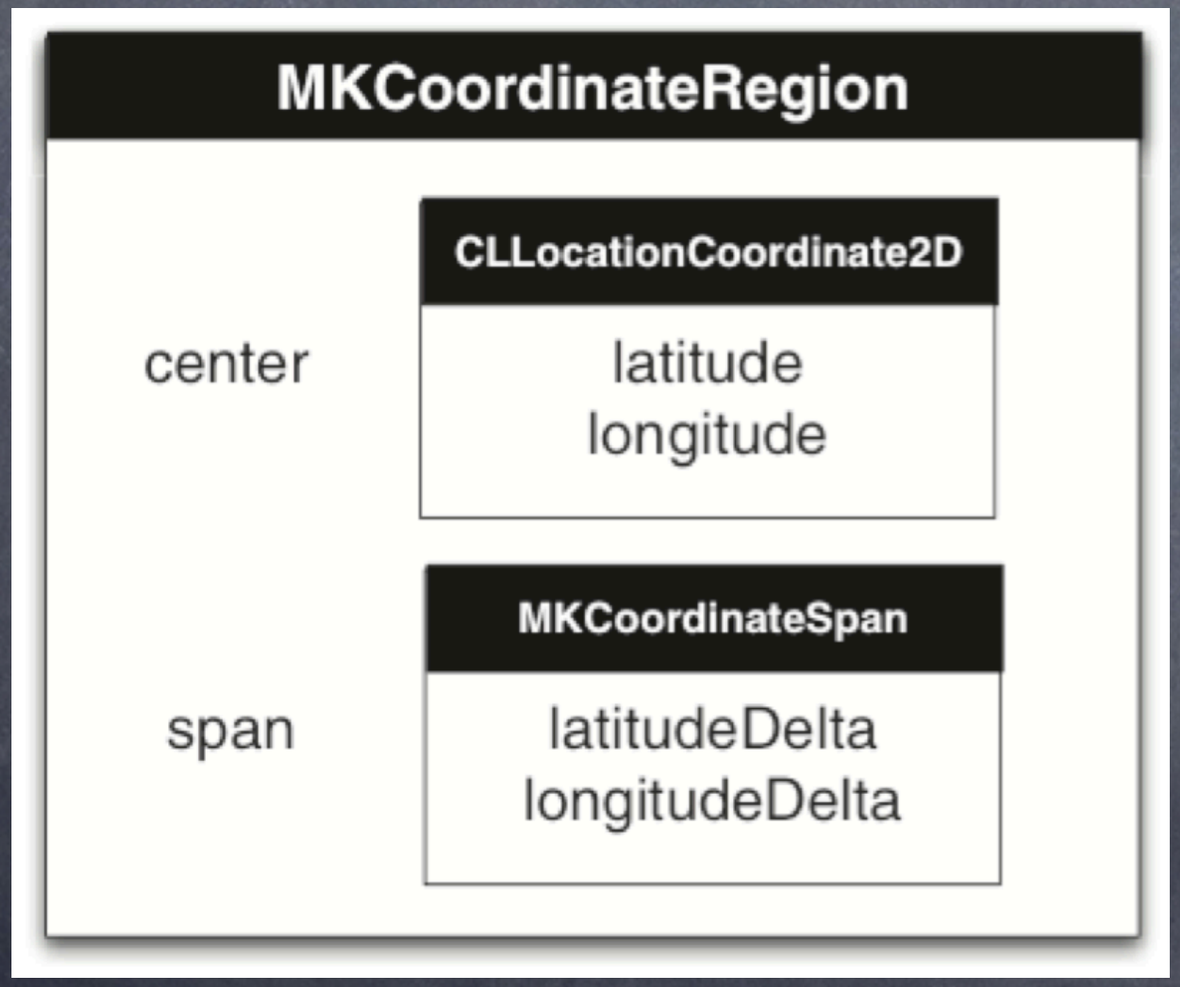
```
mapView.showsUserLocation = YES;
```

Defining Map View Region

- region property in MKMapView
- MKCoordinateRegion
- mapView.region = aNewRegion;
- [mapView setRegion:aNewRegion
  animated:YES];
- alternately, use centerCoordinate property

  CLCoordinateLocation2D

MKCoordinateRegion




```
typedef struct {
    CLLocationCoordinate2D center;
    MKCoordinateSpan span;
} MKCoordinateRegion;
```

```
typedef struct {
    CLLocationDegrees latitudeDelta;
    CLLocationDegrees longitudeDelta;
} MKCoordinateSpan;
```

Region Display Issues

- MKMapView uses Mercator projection
- Areas further from Equator get stretched out
- MKMapPoint

MKMapPoint

- Linearly proportional to screen point
- Utility functions

```
MKMapPoint MKMapPointForCoordinate(CLLocationCoordinate2D coordinate);
CLLocationCoordinate2D MKCoordinateForMapPoint(MKMapPoint mapPoint);
```

- Always store data as lat/long

MKMapRect

- MKMapPoint analogue for MKCoordinateRegion

```
typedef struct {
    MKMapPoint origin;
    MKMapSize size;
} MKMapRect;
```

```
typedef struct {
    double width;
    double height;
} MKMapSize;
```

- Utility functions (<MapKit/MKGeometry.h>)

```
MKCoordinateRegion MKCoordinateRegionForMapRect(MKMapRect rect);
```

- MKMapView property: visibleMapRect

Annotations

- Describes specific location
- Two components:

  MKAnnotation protocol

  MKAnnotationView

MKAnnotation protocol

```
@protocol MKAnnotation <NSObject>

@property (nonatomic, readonly) CLLocationCoordinate2D coordinate;

@optional
@property (nonatomic, readonly, copy) NSString *title;
@property (nonatomic, readonly, copy) NSString *subtitle;

- (void)setCoordinate:(CLLocationCoordinate2D)newCoordinate;

@end
```

MKAnnotationView

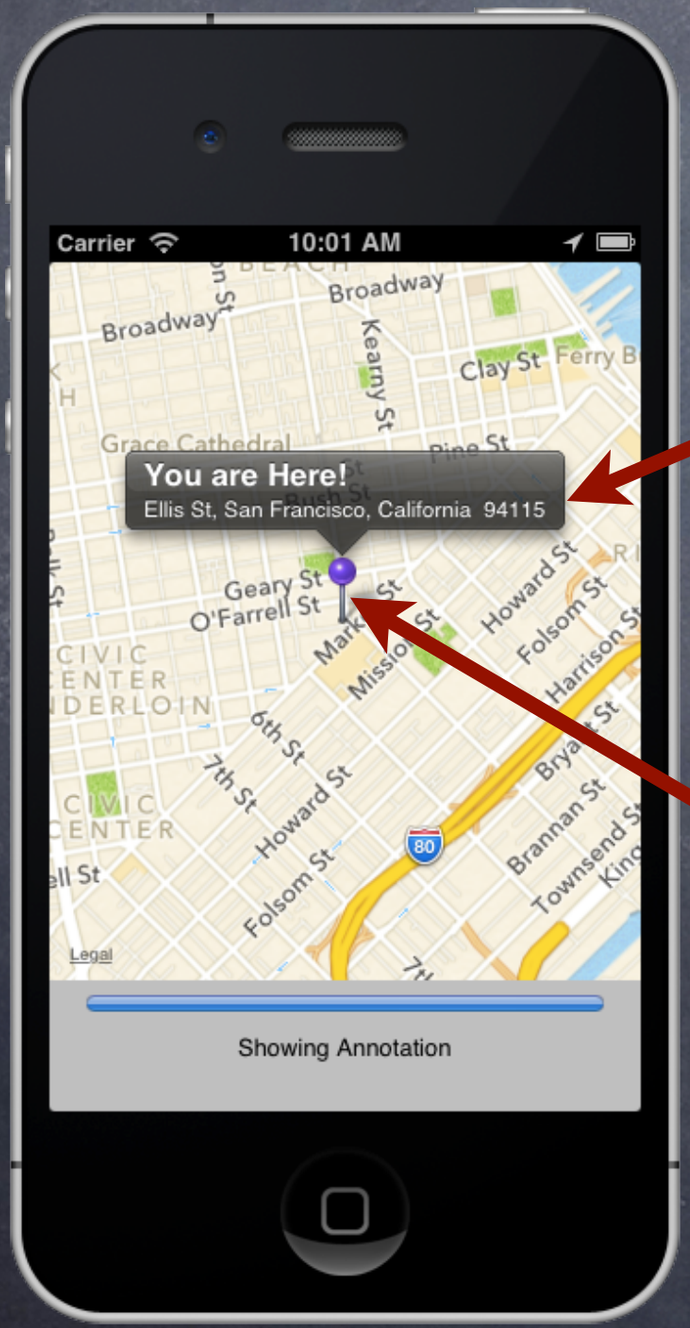
```
@interface MKAnnotationView : UIView

- (id)initWithAnnotation:(id <MKAnnotation>)annotation
         reuseIdentifier:(NSString *)reuseIdentifier;

@property (nonatomic, retain) id <MKAnnotation> annotation;

@property (nonatomic, getter=isEnabled) BOOL enabled;

@property (nonatomic, retain) UIImage *image;
@property (retain, nonatomic) UIView *leftCalloutAccessoryView;
@property (retain, nonatomic) UIView *rightCalloutAccessoryView;

@property (nonatomic) BOOL canShowCallout;

@property (nonatomic, getter=isDraggable) BOOL draggable;

@end
```

(selected methods & properties)

MKAnnotationView

- `- (id)initWithAnnotation:(id <MKAnnotation>)annotation reuseIdentifier:(NSString *)reuseIdentifier;`
  - UITableView
- MKAnnotationViews get reused
- Handled in MKMapViewDelegate
  - UITableViewDelegate

MKAnnotationView callouts



Callout View

MKPinAnnotationView

Annotation Flow

- Add id<MKAnnotation> to Map
- MKMapViewDelegate method

```
- (MKAnnotationView *)mapView:(MKMapView *)mapView
    viewForAnnotation:(id <MKAnnotation>)annotation;
```

- MKAnnotationView added MKMapView
- MKMapViewDelegate method

```
- (void)mapView:(MKMapView *)mapView didAddAnnotationViews:(NSArray *)views;
```

- (optional) animations called

Custom Annotation Views

- Subclass MKAnnotationView

  override drawRect: UIView method

- Easier to just use image property in MKAnnotationView

MKMapViewDelegate

- It's complicated
- But it makes sense
- Tries to do everything

Map Position Changes
Loading Map Data
Tracking User Location
Managing Annotation Views
Dragging Annotation Views
Selecting Annotation Views
Managing Overlay Views

Map Position Changes

```
- (void)mapView:(MKMapView *)mapView regionWillChangeAnimated:(BOOL)animated;
- (void)mapView:(MKMapView *)mapView regionDidChangeAnimated:(BOOL)animated;
```

- Changes due to scroll or zoom
- Potentially to aggregate annotation views
- Keep as lightweight as possible

Loading Map Data

```
- (void)mapViewWillStartLoadingMap:(MKMapView *)mapView;
- (void)mapViewDidFinishLoadingMap:(MKMapView *)mapView;
- (void)mapViewDidFailLoadingMap:(MKMapView *)mapView withError:(NSError *)error;
```

- Changes due to scroll or zoom
- Error if data can't be loaded
- Not called if data is cached

Tracking User Location

```
- (void)mapViewWillStartLocatingUser:(MKMapView *)mapView;
- (void)mapViewDidStopLocatingUser:(MKMapView *)mapView;
- (void)mapView:(MKMapView *)mapView didUpdateUserLocation:(MKUserLocation *)userLocation;
- (void)mapView:(MKMapView *)mapView didFailToLocateUserWithError:(NSError *)error;
```

- Will ask for permission
- Need to handle case if permission denied
- Internal CLLocationManager instance

Demo

Exercise

- Download http://kykim.com/capitals.plist
- Add plist to project
- Parse plist information and add annotation for each capital
- Add callout view for each pin
- Add disclosure button for each callout
- Have disclosure button open new view/alert/ something

Extras

- Add all annotation delegate calls, and log when each happens