CS193p

Spring 2010

Announcements

- Check out the slides at the end of this presentation and let us know which topics are of the most interest to you
- Sonali's office hours changed tomorrow only
  4 to 6pm (same place)
- Update to Final Project proposal request
  In your proposal, include a description of your application as it might appear on the AppStore. Sell us!

Today's Topics

- Core Location & MapKit
- UIWebView & WebKit

Core Location

- Framework for managing locations and headings
  No user-interface, just location information.
- Basic object is a `CLLocation`
- Represented by a latitude and longitude
- But more than that, also an "accuracy"
  Because not every location-generating method is as accurate as others.
  More on this in a few slides ...
- Also can encapsulate some other information
  Course and speed
  Timestamp (when the measurement of the location was taken)
  Altitude

CLLocation

- Properties

```
@property CLLocationCoordinate2D coordinate;

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

```
@property CLLocationAccuracy horizontalAccuracy;
@property CLLocationAccuracy verticalAccuracy;

    kCLLocationAccuracyBest;
    kCLLocationAccuracyNearestTenMeters;
    kCLLocationAccuracyHundredMeters;
    kCLLocationAccuracyKilometer;
    kCLLocationAccuracyThreeKilometers;
```

CLLocation

- Properties

```
@property CLLocationSpeed speed;         // meters per second
@property CLLocationDirection course;    // degrees, 0 is north, clockwise
@property CLLocationDistance altitude; // meters
```

- Comparing CLLocation objects

```
- (CLLocationDistance)distanceFromLocation:(CLLocation *)location;
```

Core Location

- How do you find out where you are?
  More complicated of a question than you might think
- `CLLocationManager` is the object to use
- Create using `alloc/init`, then configure, then start

```
CLLocationManager *clm = [[CLLocationManager alloc] init];
clm.distanceFilter = 10.0;   // minimum distance change to report, in meters
clm.desiredAccuracy = kCLLocationAccuracyTenMeters;
[clm startUpdatingLocation];
```

- Do not specify more accuracy than you need!
  More accuracy requires more time and more power
  Cell tower triangulation (not very accurate, but low power)
  WiFi node database lookup (more accurate, but more power)
  GPS (very accurate, lots of power)

Core Location

- You can then ask the manager for the location

```
CLLocation *location = clm.location;
```

- Or, more likely, get notified as map view's `delegate`

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

- User will be asked if it's okay to check location

  You can provide a string to tell user the "purpose" of this use

```
@property (copy) NSString *purpose;
```

- Delegate gets told of failure

```
- (void)locationManager:(CLLocationManager *)manager
       didFailWithError:(NSError *)error;
```

Core Location

- Heading is similar

```
CLLocationManager *clm = [[CLLocationManager alloc] init];
clm.headingFilter = 10.0;  // minimum degree change to report
[clm startUpdatingHeading];
```

- Verifying that the device supports your need

```
@property BOOL headingAvailable;
@property BOOL locationServicesEnabled; // changeable in Settings
```

- Delegate method for heading updates

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

`CLHeading` class gives true, magnetic, accuracy, timestamp, x, y, z

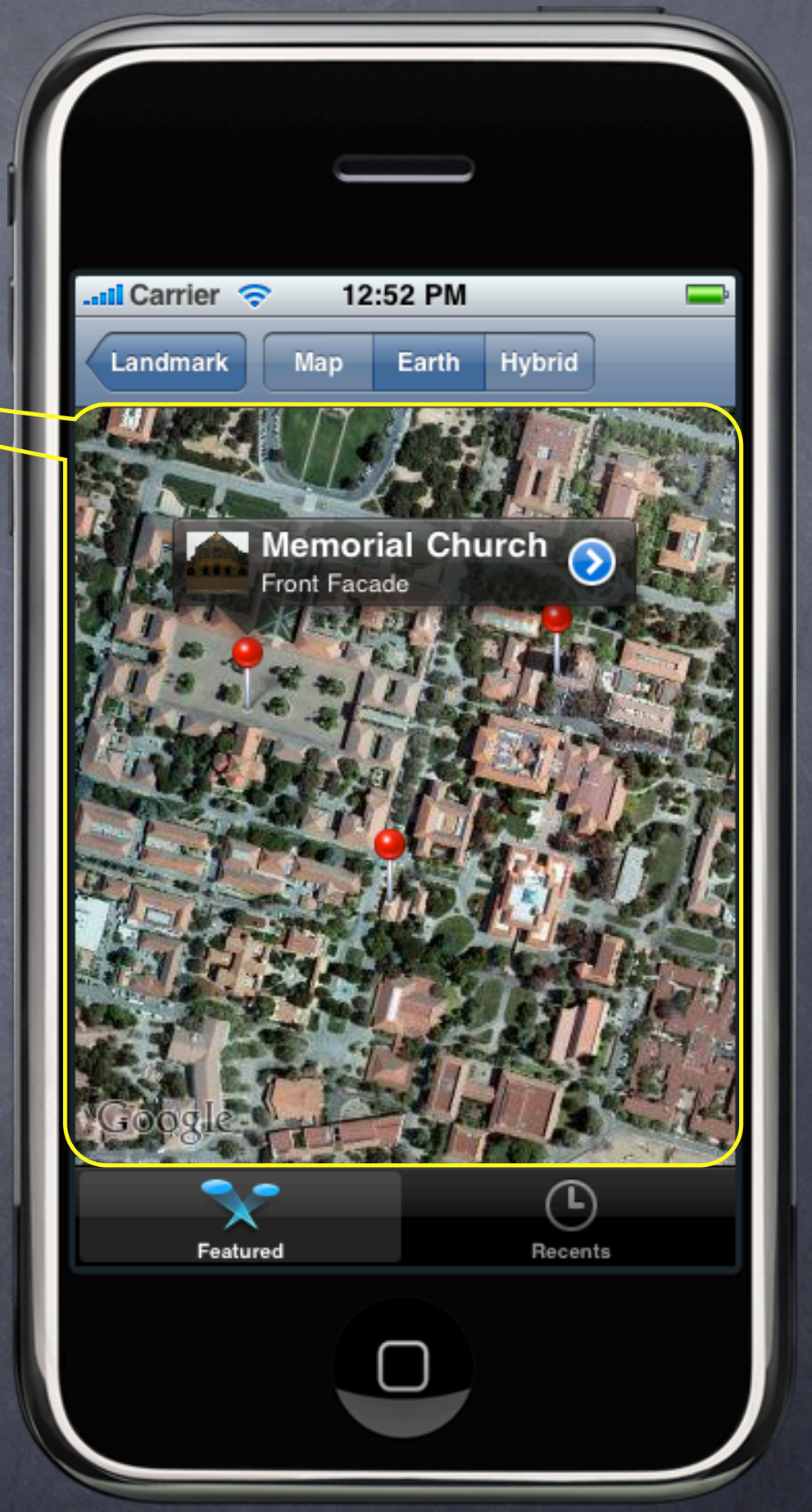
- Delegate can also prevent heading calibration

```
- (BOOL)locationManagerShouldDisplayHeadingCalibration:(CLLocationManager *);
```

If heading not calibrated, delegate will get "magnetic interference" errors

MKMapView

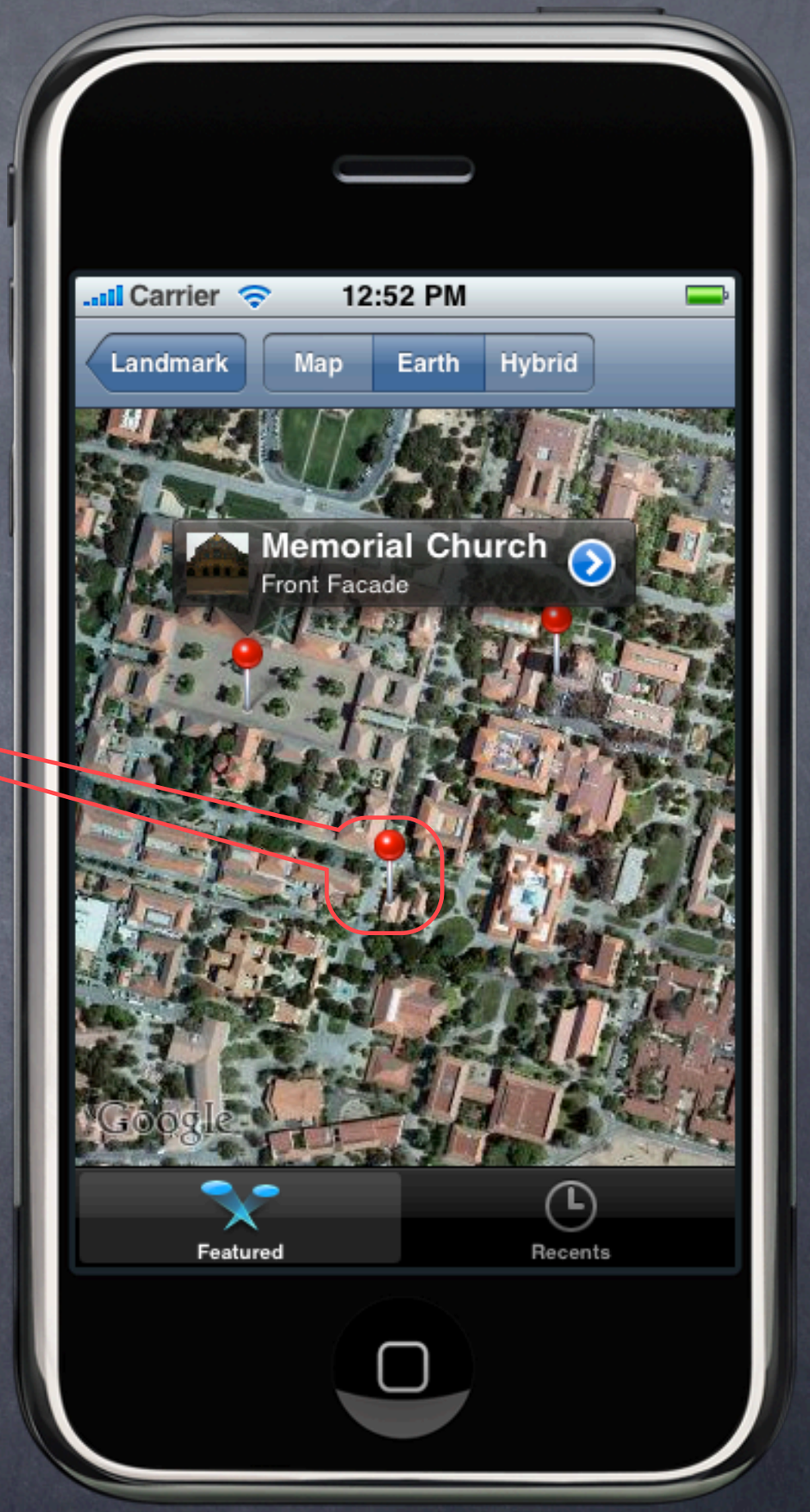
- Displays a map



MKMapView

- Displays a map
- The map can have annotations on it

  Each annotation is just a coordinate with a title and subtitle. They are displayed using an `MKAnnotationView` (`MKPinAnnotationView` shown here).



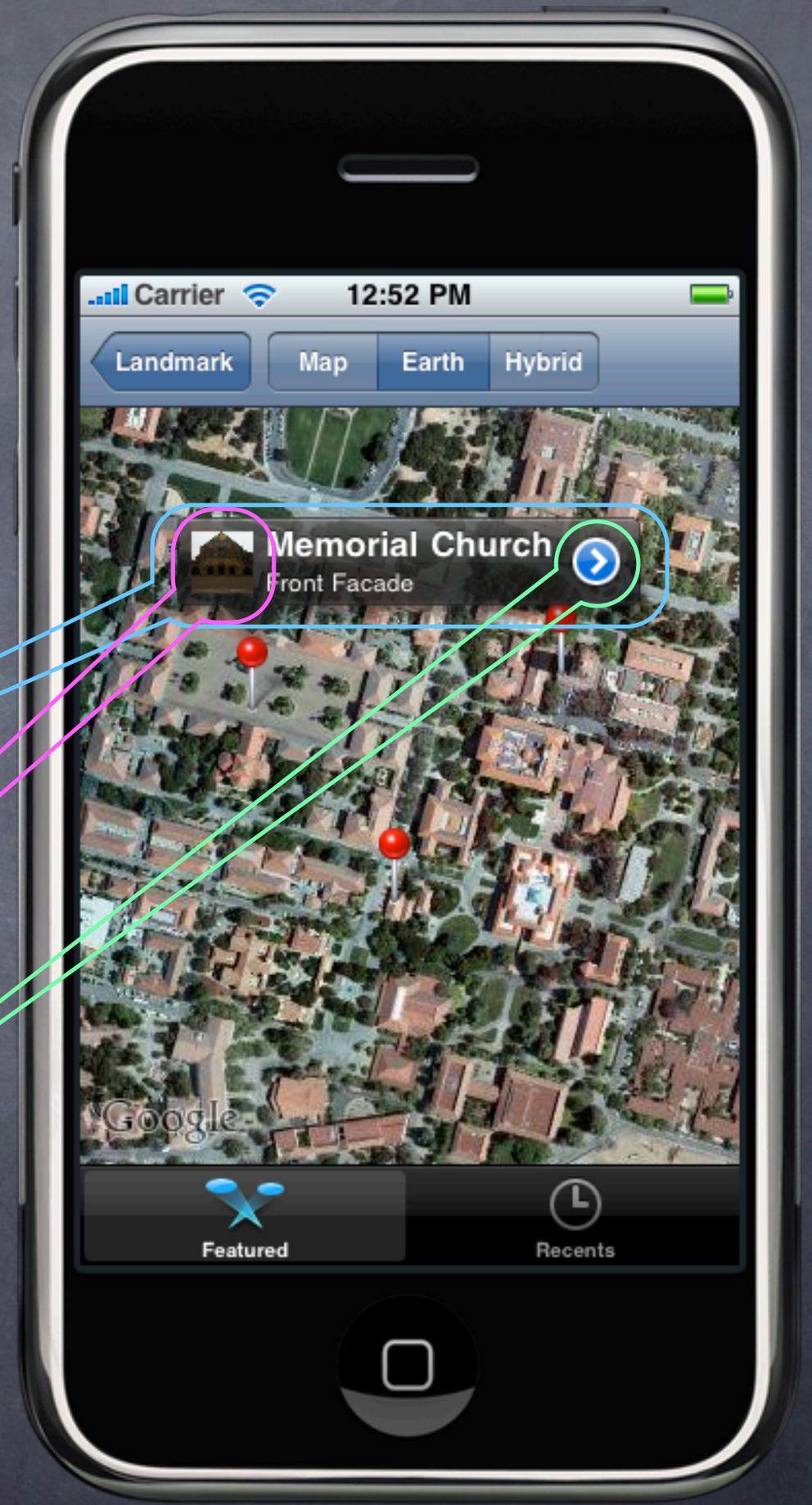
MKMapView

- Displays a map
- The map can have annotations on it

  Each annotation is just a coordinate with a title and subtitle. They are displayed using an MKAnnotationView (MKPinAnnotationView shown here).

- Annotations can have a callout associated with them (shown when clicked)

  By default, it just shows the title and subtitle, but you can add accessory views to the left and right (in this case, UIImageView on left, UIButton (of type UIButtonTypeDetailDisclosure) on right).



MKMapView

- Create with `alloc/init` or drag out in IB
  Almost always needs a `delegate` (which is usually a view controller).

- Important MKMapView `delegate` method

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

  This returns the "pin" view (or other view) to show an annotation.
  For performance, these are "dequeued" just like table view cells are.

- Note that MKAnnotation is a protocol, not a class
  Any object can be an annotation as long as it implements this protocol

```
@protocol MKAnnotation <NSObject>
@property (readonly) CLLocationCoordinate2D coordinate;
@optional
@property (readonly) NSString *title;
@property (readonly) NSString *subtitle;
@end
```

MKMapView

- What is an `MKAnnotationView`?
- View that draws a pin or other marker on the map
- Interesting methods/properties

```
@property BOOL enabled; // whether it ignores touch events (different display?)
@property (retain) UIImage *image;   // a pin or some such image
@property (retain) id <MKAnnotation> annotation; // the annotation

@property BOOL canShowCallout;   // whether it shows a callout when selected
@property (retain) UIView *leftCalloutAccessoryView;
@property (retain) UIView *rightCalloutAccessoryView;

@property BOOL selected;  // whether it is selected (showing callout)
- (void)setSelected:(BOOL)selected animated:(BOOL)animated;
```

- Usually use subclass called `MKPinAnnotationView`

  Just puts a "pin" on the map. Can be one of three colors.

  Also has an animation of the pin dropping onto the map (if you want).

MKMapView

- Typical implementation of `viewForAnnotation:`

```
- (MKAnnotationView *)mapView:(MKMapView *)mapView
            viewForAnnotation:(id <MKAnnotation>)annotation;
{
   MKAnnotationView *annotationView =
      [mapView dequeueReusableAnnotationViewWithIdentifier:@"xx"];
   if (!annotationView) {
      annotationView = [[MKPinAnnotationView alloc] // or custom subclass
         initWithAnnotation:annotation reuseIdentifier:@"xx"];
      [annotationView autorelease];
      annotationView.canShowCallout = YES;   // clicking does nothing otherwise
      annotationView.leftCalloutAccessoryView = ...;   // maybe not here
      annotationView.rightCalloutAccessoryView = ...; // maybe not here
   }
   annotationView.annotation = annotation;
   // maybe set up left and/or right callout accessory view based on annotation
   return annotationView;
}
```

This is called for every visible annotation, so if you're showing the entire world, for example, it could be ALL of your annotations, so this wants to be efficient.

MKMapView

- Subclassing MKAnnotationView

  For efficiency, sometimes you do not want to create a callout accessory view until the callout is actually asked for by the user.

  Simplest way to do this is to subclass MK(Pin)AnnotationView.

  Then override the setSelected:animated: method.

```
- (void)setSelected:(BOOL)selected animated:(BOOL)animated
{
    if (selected && !self.leftCalloutAccessoryView) {
        if ([self.annotation conformsToProtocol:...]) { // or …
        if ([self.annotation respondsToSelector:...]) { // or …
        if ([self.annotation isKindOfClass:...]) {
            // get needed information from self.annotation
            UIView *accessoryView = ...; // create desired callout view
            self.leftCalloutAccessoryView = accessoryView;
        }
    }
    [super setSelected:selected animated:animated];
}
```

  Then use this class in your webView:viewForAnnotation: to create new (reusable) MKAnnotationView objects.

MKMapView

- Once `delegate` is set up, add annotations to map

```
- (void)addAnnotation:(id <MKAnnotation>)annotation;
- (void)addAnnotations:(NSArray *)array; // of id <MKAnnotation>
```

- You can remove annotations as well

```
- (void)removeAnnotation:(id <MKAnnotation>)annotation;
- (void)removeAnnotations:(NSArray *)array; // of id <MKAnnotation>
```

- And get (but not set) the current array of them

```
@property (readonly) NSArray *annotations; // of id <MKAnnotation>
```

MKMapView

- Configuring the map view's display type

```
@property MKMapType mapType;
```

`MKMapTypeStandard`, `MKMapTypeSatellite`, or `MKMapTypeHybrid`

- Showing the user's current location

```
@property BOOL showsUserLocation;
@property (readonly) BOOL isUserLocationVisible;
@property (readonly) MKUserLocation *userLocation;
```

MKUserLocation is an object which conforms to the MKAnnotation protocol which represents the user's location.

MKMapView

- Controlling the region the map is displaying

```
@property MKCoordinateRegion region;

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

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

- Can just set the center point if you want

```
@property CLLocationCoordinate2D centerCoordinate;
```

- Both can be set animated

```
- (void)setRegion:(MKCoordinateRegion) animated:(BOOL);
- (void)setCenterCoordinate:(CLLocationCoordinate2D) animated:(BOOL);
```

MKMapView

- Converting to/from lat-long to view coordinates

  The view must be in the map view's window (or `nil` for window coordinates)

```
- (CGPoint)convertCoordinate:(CLLocationCoordinate2D)coord
                 toPointToView:(UIView *)view;
- (CLLocationCoordinate2D)convertPoint:(CGPoint)point
                 toCoordinateFromView:(UIView *)view;
- (MKCordinateRegion)convertRect:(CGRect)rect
                 toRegionFromView:(UIView *)view;
- (CGRect)convertRegion:(MKCoordinateRegion)region
                 toRectToView:(UIView *)view;
```

MKMapView

- MKMapViewDelegate region change notifications

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

- Map loading notifications

```
- (void)mapViewWillStartLoadingMap:(MKMapView *);
```

Sent when MKMapView goes out to the internet for map image data

```
- (void)mapViewDidFinishLoadingMap:(MKMapView *);
```

Sent when the entire visible MKMapView's imagery has been downloaded

```
- (void)mapViewDidFailLoadingMap:(MKMapView *) withError:(NSError *);
```

Usually due to network failure. Notify user of this (usually).

MKMapView

- Special message sent if callout accessory tapped

```
- (void)mapView:(MKMapView *) annotationView:(MKAnnotationView *)
    calloutAccessoryControlTapped:(UIControl *);
```

Only sent if a callout accessory is a `UIControl` and only on tap.
Simply a convenience method to not have to deal with target/action.
The callout accessory in question is usually a disclosure button.

For example, you might put this code in your `mapView:viewForAnnotation:`.

```
annotationView.rightCalloutAccessoryView =
    [UIButton buttonWithType:UIButtonTypeDetailDisclosure];
```

Then push a view controller onto the navigation stack (or some such)
when the above `delegate` method gets called.

MKMapView

- Demo

UIWebView

- Basically a "browser in a view"
  - Not just for displaying HTML pages
  - Also can be used to display PDF's or other complex documents
- Easy to put "back," "reload," etc., buttons around it
- Based on WebKit
  - Open source framework (started by Apple)
  - HTML renderer
- Delegate to control/observe URL requests
  - You can prevent user from clicking through certain links
  - Observe and respond to user's navigation
- Supports JavaScript
  - But limited to 5 seconds or 10 megabytes of memory allocation

UIWebView

- Three ways to load up the view

```
- (void)loadRequest:(NSURLRequest *)request;
- (void)loadHTMLString:(NSString *)string baseURL:(NSURL *)url;
- (void)loadData:(NSData *)data
        MIMEType:(NSString *)MIMEtype
textEncodingName:(NSString *)encodingName
         baseURL:(NSURL *)url;
```

- Base URL is "environment" to load images, etc.
  i.e., it's the base url for relative url's in the data or HTML string

- MIME type says how to interpret the data
  Multimedia Internet Mail Extension
  Standard way to denote certain file types (like PDF)
  Think "e-mail attachments" (that's where the type came from)

UIWebView

- NSURLRequest

```
+ (NSURLRequest *)requestWithURL:(NSURL *)url
                     cachePolicy:(NSURLRequestCachePolicy)policy
                 timeoutInterval:(NSTimeInterval)timeoutInterval;

+ (NSURLRequest *)requestWithURL:(NSURL *)url;
```

- NSURL

  Basically just an NSString, but enforced to be "well-formed"

  Examples: "`file://...`" or "`http://...`"

  Is actually the recommended way to read/write files (rather than paths)

```
+ (NSURL *)urlWithString:(NSString *)urlString;
+ (NSURL *)fileURLWithPath:(NSString *)path isDirectory:(BOOL)isDir;
```

- NSURLCachePolicy

  Ignore local cache; ignore caches on internet; use expired cache;
  use cache only (don't go out onto the network); use cache if validated

UIWebView

- Putting browser UI around the view

```
@property BOOL loading;
@property BOOL canGoBack;
@property BOOL canGoForward;

- (void)reload;
- (void)stopLoading;
- (void)goBack;
- (void)goForward;
```

- Other control of the display

  Fit the web page to fit the size of the view

```
@property BOOL scalesPageToFit;
```

  Detect phone numbers and/or web links and make them clickable

```
@property UIDataDetectorTypes dataDetectorTypes;
```

- Execute JavaScript

```
- (NSString *)stringByEvaluatingJavaScriptFromString:(NSString *)javaScript;
```

UIWebView

- Delegate

```
- (BOOL)webView:(UIWebView *)webView
    shouldStartLoadWithRequest:(NSURLRequest *)request
    navigationType:(UIWebViewNavigationType)navigationType;
```

- UIWebViewNavigationType

  Various sorts of things a user can click on to start a load in a page, e.g.,

  UIWebViewNavigationTypeLinkClicked

  UIWebViewNavigationTypeFormSubmitted

- Delegate will also notify you of activity

```
- (void)webViewDidStartLoad:(UIWebView *)webView;

- (void)webViewDidFinishLoad:(UIWebView *)webView;

- (void)webView:(UIWebView *)webView
    didFailLoadWithError:(NSError *)error;
```

Future Lecture Options

- Text

  Inputting text (via the soft keyboard) is generally a minor part of a phone interface because the tiny keyboard is (relatively) difficult to use (although that's not so true on the iPad). This lecture topic would cover UITextField and UITextView and maybe a little bit on keyboard customization.

- Address Book

  The API for getting and storing address book information is not object-oriented (it's a C API), but it's quite extensive.

- Modal Views and Alerts

  Putting up an alert to the user or asking for some information in a modal (i.e. can't do anything else until the information is provided) fashion is not uncommon and very useful in many circumstances.

- Animating Views

  It is possible to set certain attributes of a UIView (like it's frame or opacity) and then have the UIKit animate the changes to those attributes. This is how groups of controls "slide" or "curl" or "flip" in and out. It could also be used to do some simple 2D animation (though it might not be suitable for a full-on animated game).

Future Lecture Options

- Game Kit

  The Game Kit provides an API for communicating in a peer-to-peer fashion between devices using the internet or bluetooth connectivity. This is NOT simulatable, so you would need to physical devices to develop/test this.

- Multitouch

  We learned how to do multitouch with UIGestureRecognizers, but this only works in iPhone OS 3.2 (i.e. only on iPads for now). There is an API for doing this on 3.1 and earlier.

- Popover Controller

  We covered this in lecture, but did not demo it.

- Accelerometer

  Self-explanatory.

- Camera

  Capturing images or video from the camera. Playing recorded video.

Future Lecture Options

- Audio
  Recording and playing back audio.
- Localization
  Your application can be constructed so that it can be translated into other languages. Many developers ignore this when building their application, then realize they have given up a substantial worldwide market for their application and have to go back and make this work.
- Leak Detection Tool
  You've been searching for leaks simply by inspecting your code so far, but there is a tool which can help you find leaks that could be demo'ed in class.
- Settings Bundle
  It is possible to add user-interface elements to the Settings application. For example, you could set the maximum cache size of your SPoT application.
- Other
  If there is some other topic of general interest, propose away!