Stanford CS193p

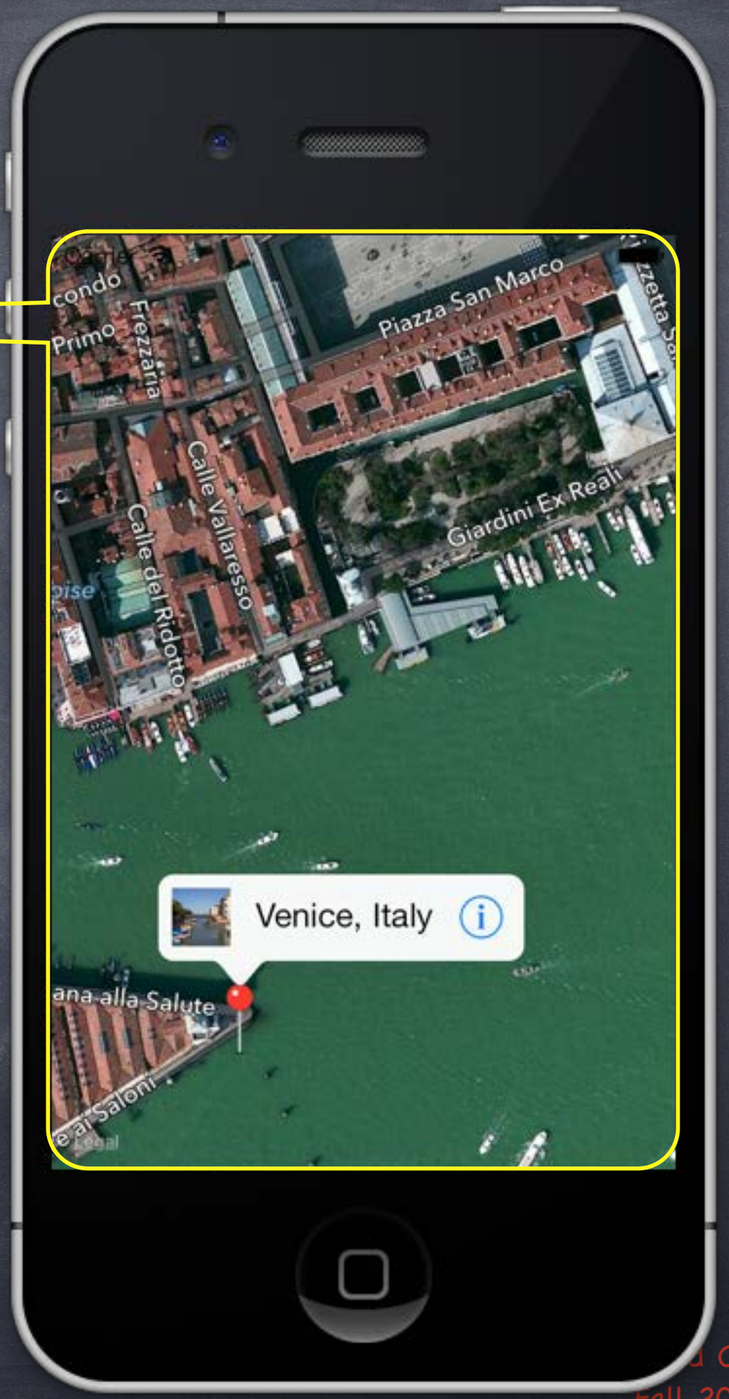
Developing Applications for iOS
Fall 2013-14

Today

- MapKit
  User interface for dealing with locations.
- Embed Segue
  Putting one VC's `self.view` inside another VC's View
- Photomania Map Demo
  Embedding a Map View Controller into our View Controller that displays a Photo

Map Kit

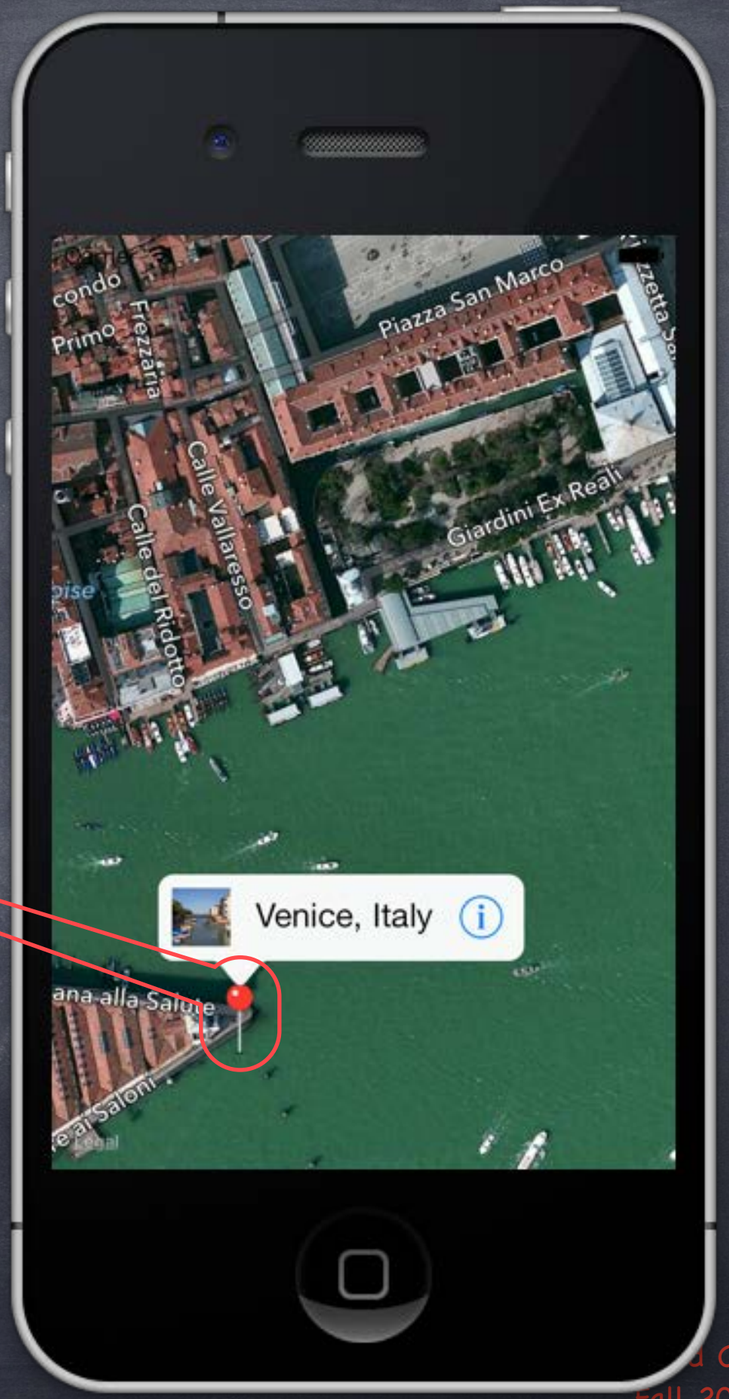
- `MKMapView` displays a map



Map Kit

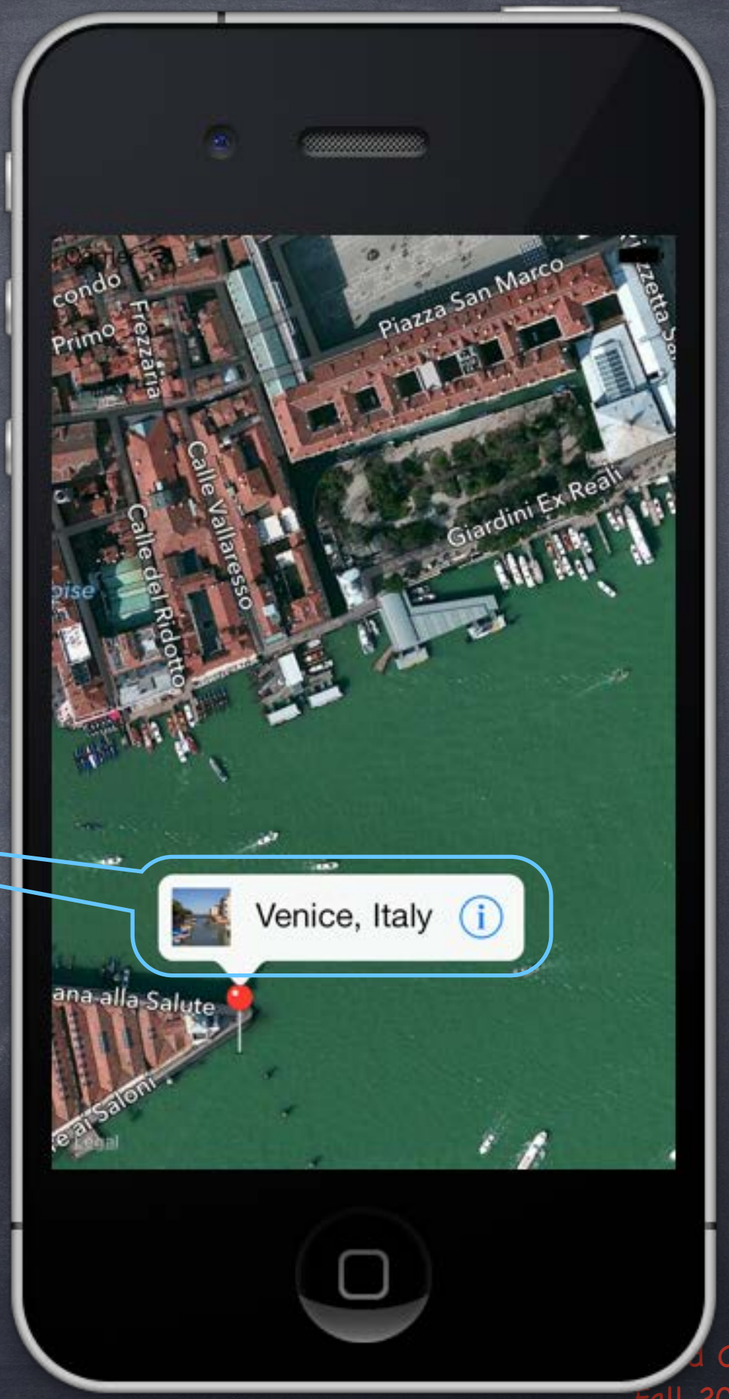
- `MKMapView` displays a map
- The map can have annotations on it

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



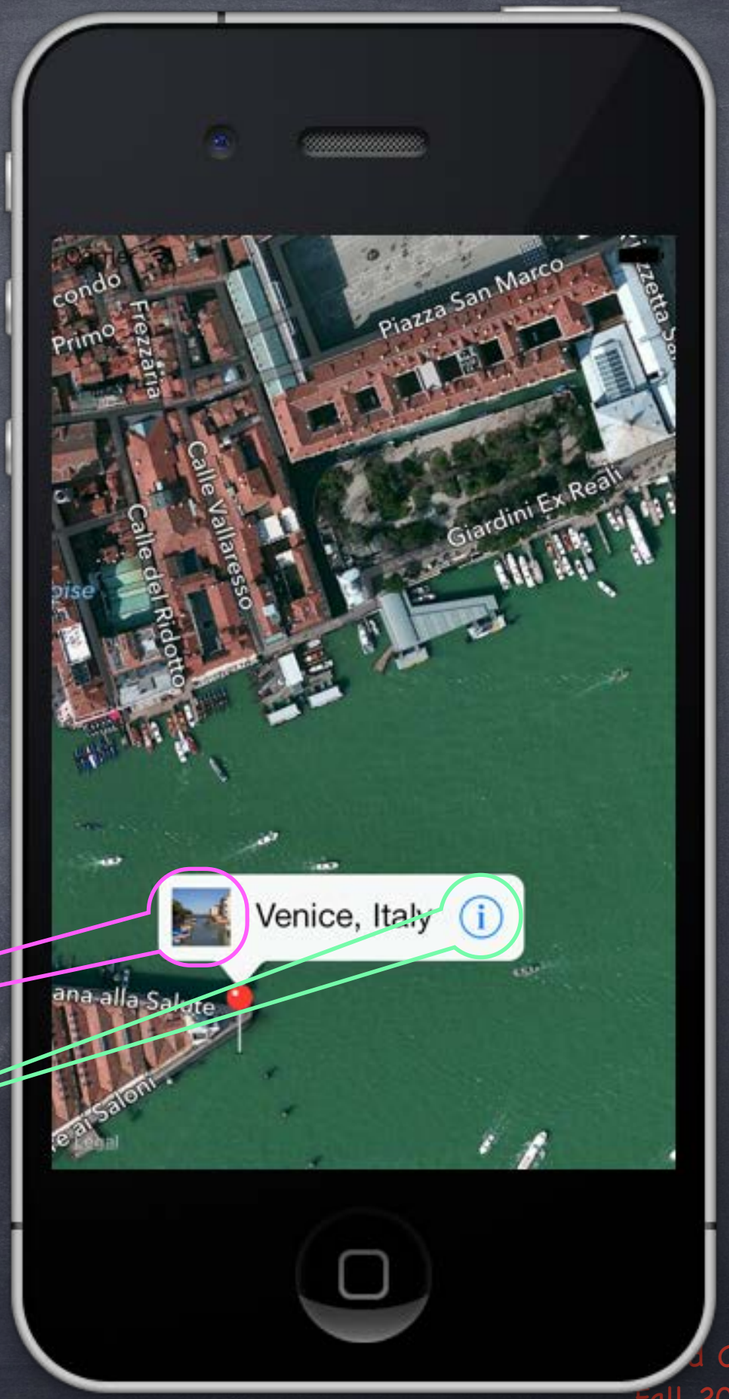
Map Kit

- MKMapView displays a map
- The map can have annotations on it
  Each annotation is simply a coordinate, a title and a subtitle.
  They are displayed using an MKAnnotationView
  (MKPinAnnotationView shown here).
- Annotations can have a callout
  It appears when the annotation view is clicked.
  By default just shows the title and subtitle.



Map Kit

- MKMapView displays a map
- The map can have annotations on it
  - Each annotation is simply a coordinate, a title and a subtitle.
  - They are displayed using an MKAnnotationView (MKPinAnnotationView shown here).
- Annotations can have a callout
  - It appears when the annotation view is clicked.
  - By default just shows the title and subtitle.
- But callout can also have accessory views
  - In this example, the left is a UIImageView,
  - the right is a UIButton (UIButtonTypeDetailDisclosure)



MKMapView

- Create with alloc/init or drag from object palette in Xcode
- Displays an array of objects which implement MKAnnotation

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

- MKAnnotation protocol

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

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

MKAnnotation

- Note that the annotations property is readonly, so …

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

  Must add/remove annotations explicitly

```
- (void)addAnnotation:(id <MKAnnotation>)annotation;
- (void)addAnnotations:(NSArray *)annotations;
- (void)removeAnnotation:(id <MKAnnotation>)annotation;
- (void)removeAnnotations:(NSArray *)annotations;
```

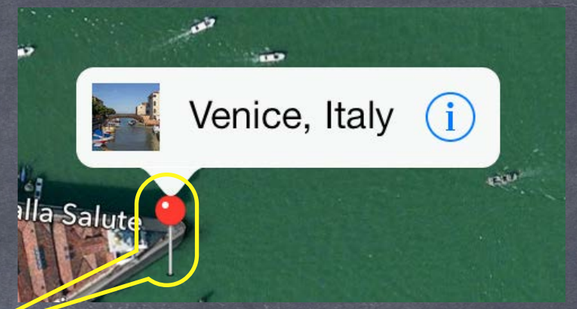
- Generally a good idea to add all your annotations up-front

  Allows the MKMapView to be efficient about how it displays them

  Annotations are light-weight, but annotation views are not.

  Luckily MKMapView reuses annotation views similar to how UITableView reuses cells.

MKAnnotation



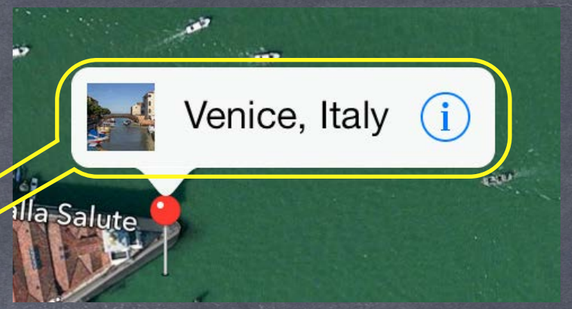
- What do annotations look like on the map?

  Annotations are drawn using an `MKAnnotationView` subclass.

  The default one is `MKPinAnnotationView` (which is why they look like pins by default).

  You can subclass or set properties on existing `MKAnnotationViews` to modify the look.

MKAnnotation



- What do annotations look like on the map?
  - Annotations are drawn using an `MKAnnotationView` subclass.
  - The default one is `MKPinAnnotationView` (which is why they look like pins by default).
  - You can subclass or set properties on existing `MKAnnotationViews` to modify the look.
- What happens when you touch on an annotation (e.g. the pin)?
  - Depends on the `MKAnnotationView` that is associated with the annotation (more on this later).
  - By default, nothing happens, but if `canShowCallout` is `YES` in the `MKAnnotationView`, then a little box will appear showing the annotation's title and subtitle.
  - And this little box (the callout) can be enhanced with `left/rightCalloutAccessoryViews`.

  The following `delegate` method is also called…

```
- (void)mapView:(MKMapView *)sender didSelectAnnotationView:(MKAnnotationView *)aView;
```

  This is a great place to set up the `MKAnnotationView`'s callout accessory views lazily.
  For example, you might want to wait until this method is called to download an image to show.

MKAnnotationView

- How are MKAnnotationViews created & associated w/annotations?

  Very similar to UITableViewCells in a UITableView.

  Implement the following MKMapViewDelegate method (if not implemented, returns a pin view).

```
- (MKAnnotationView *)mapView:(MKMapView *)sender
            viewForAnnotation:(id <MKAnnotation>)annotation
{
    MKAnnotationView *aView = [sender dequeueReusableAnnotationViewWithIdentifier:IDENT];
    if (!aView) {
        aView = [[MKPinAnnotationView alloc] initWithAnnotation:annotation
                                                 reuseIdentifier:IDENT];
        // set canShowCallout to YES and build aView's callout accessory views here
    }
    aView.annotation = annotation; // yes, this happens twice if no dequeue
    // maybe load up accessory views here (if not too expensive)?
    // or reset them and wait until mapView:didSelectAnnotationView: to load actual data
    return aView;
}
```

  You can see why you might want to only show visible annotations (to keep view count low)

MKAnnotationView

- MKAnnotationView

  Interesting properties (all nonatomic, strong if a pointer) …

```
@property id <MKAnnotation> annotation;  // the annotation; treat as if readonly
@property UIImage *image;  // instead of the pin, for example
@property UIView *leftCalloutAccessoryView;  // maybe a UIImageView
@property UIView *rightCalloutAccessoryView;  // maybe a "disclosure" UIButton
@property BOOL enabled;  // NO means it ignores touch events, no delegate method, no callout
@property CGPoint centerOffset;  // where the "head of the pin" is relative to the image
@property BOOL draggable;  // only works if the annotation implements setCoordinate:
```

- If you set one of the callout accessory views to a UIControl

  e.g. `aView.rightCalloutAccessoryView = [UIButton buttonWithType:UIButtonTypeDetailDisclosure];`

  The following MKMapViewDelegate method will get called when the accessory view is touched …

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

MKAnnotationView

- Using `didSelectAnnotationView:` to load up callout accessories

  Example … downloaded thumbnail image in `leftCalloutAccessoryView`.

  Create the `UIImageView` and assign it to `leftCalloutAccessoryView` in `mapView:viewForAnnotation:`.

  Reset the `UIImageView`'s image to `nil` there as well.

  Then load the image on demand in `mapView:didSelectAnnotationView:` …

```
- (void)mapView:(MKMapView *)sender didSelectAnnotationView:(MKAnnotationView *)aView
{
    if ([aView.leftCalloutAccessoryView isKindOfClass:[UIImageView class]]) {
        UIImageView *imageView = (UIImageView *)aView.leftCalloutAccessoryView;
        imageView.image = ...;  // if you do this in a GCD queue, be careful, views are reused!
    }
}
```

MKMapView

- Configuring the map view's display type

```
@property MKMapType mapType;
MKMapTypeStandard, MKMapTypeSatellite, 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 MKAnnotation which holds the user's location.

- Restricting the user's interaction with the map

```
@property BOOL zoomEnabled;
@property BOOL scrollEnabled;
@property BOOL pitchEnabled;   // 3D
@property BOOL rotateEnabled;
```

MKMapCamera

- Setting where the user is seeing the map from (in 3D)

```
MKMapView @property (copy) MKMapCamera *camera;
```

- MKMapCamera

  Specify `centerCoordinate`, `heading`, `pitch` and `altitude` of the camera.

  Or use convenient initializer …

```
+ (MKMapCamera *)cameraLookingAtCenterCoordinate:(CLLocationCoordinate2D)coord
                                   fromEyeCoordinate:(CLLocationCoordinate2D)cameraPosition
                                         eyeAltitude:(CLLocationDistance)eyeAltitude;
```

MKMapView

- Controlling the region (part of the world) the map is displaying

```
@property MKCoordinateRegion region;
typedef struct {
    CLLocationCoordinate2D center;
    MKCoordinateSpan span;
} MKCoordinateRegion;
typedef struct {
    CLLocationDegrees latitudeDelta;
    CLLocationDegrees longitudeDelta;
}
- (void)setRegion:(MKCoordinateRegion)region animated:(BOOL)animated; // animate
```

- Can also set the center point only or set to show annotations

```
@property CLLocationCoordinate2D centerCoordinate;
- (void)setCenterCoordinate:(CLLocationCoordinate2D)center animated:(BOOL)animated;
- (void)showAnnotations:(NSArray *)someAnnotations animated:(BOOL)animated;
```

MKMapView

- Lots of C functions to convert points, regions, rects, etc.

  See documentation, e.g. `MKMapRectContainsPoint`, `MKMapPointForCoordinate`, etc.

- Converting to/from map points/rects from/to view coordinates

  ```
  - (MKMapPoint)mapPointForPoint:(CGPoint)point;
  - (MKMapRect)mapRectForRect:(CGRect)rect;
  - (CGPoint)pointForMapPoint:(MKMapPoint)mapPoint;
  - (CGRect)rectForMapRect:(MKMapRect)mapRect;
  ```

  Etc.

MKMapView

- Another MKMapViewDelegate method …

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

This is a good place to "chain" animations to the map.

When you display somewhere new in the map that is far away, zoom out, then back in.

This method will let you know when it's finished zooming out, so you can then zoom in.

MKLocalSearch

- Searching for places in the world

  Can search by "natural language" strings asynchronously (uses the network) …

```
MKLocalSearchRequest *request = [[MKLocalSearchRequest alloc] init];
request.naturalLanguageQuery = @"Ike's";
request.region = …; // e.g., Stanford campus
MKLocalSearch *search = [[MKLocalSearch alloc] initWithRequest:request];
[search startWithCompletionHandler:^(MKLocalSearchResponse *response, NSError *error) {
    // response contains an array of MKMapItem which contains MKPlacemark
}];
```

- MKMapItem

  You can open one of these in the Maps app!

```
- (BOOL)openInMapsWithLaunchOptions:(NSDictionary *)options; // options like region, show traffic
```

- MKPlacemark

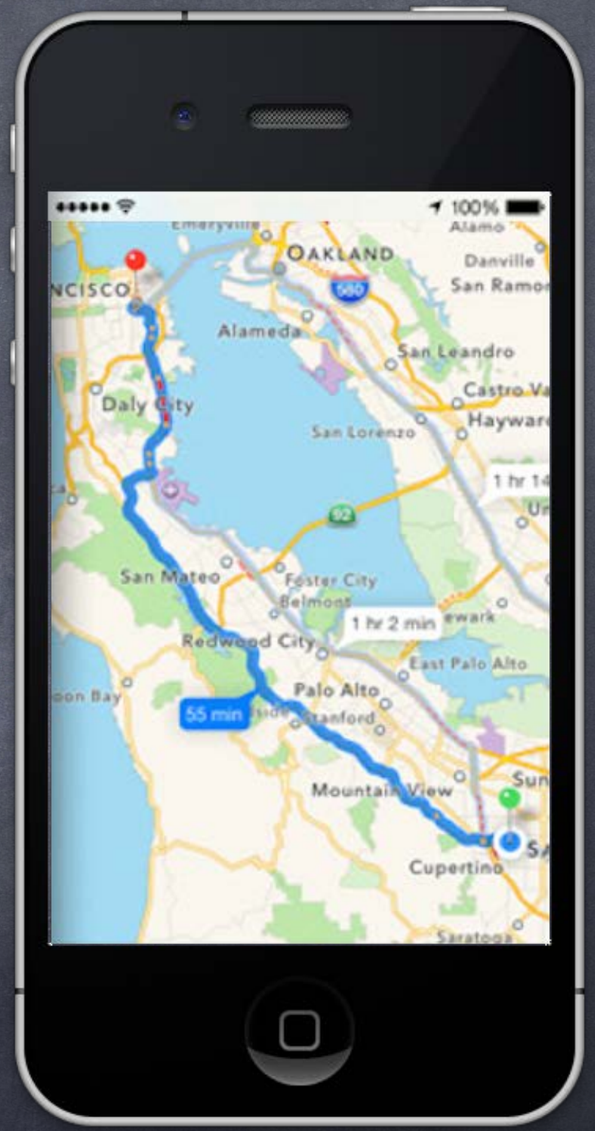
  Contains location, name of `location`, `postalCode`, `region`, etc.

MKDirections

- Getting directions from one place to another
  - Very similar API to searching.
  - Specify source and destination `MKMapItem`.
  - Asynchronous API to get a bunch of `MKRoute`s.

  `MKRoute` includes a name for the route, turn-by-turn directions, expected travel time, etc.

  Also come with `MKPolyline` descriptions of the routes which can be overlaid on the map …



Overlays

- Overlays

  Add overlays to the MKMapView and it will later ask you for a renderer to draw the overlay.

  ```
  - (void)addOverlay:(id <MKOverlay>)overlay level:(MKOverlayLevel)level;
  ```

  Level is (currently) either `AboveRoads` or `AboveLabels` (over everything but annotation views).

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

- MKOverlay protocol

  Protocol which includes MKAnnotation plus …

  ```
  @property (readonly) MKMapRect boundingMapRect;
  - (BOOL)intersectsMapRect:(MKMapRect)mapRect; // optional, uses boundingMapRect otherwise
  ```

- Overlays are associated with MKOverlayRenderers via delegate

  Just like annotations are associated with MKAnnotationViews, so are renderers with overlays …

  ```
  - (MKOverlayRenderer *)mapView:(MKMapView *)sender
                 rendererForOverlay:(id <MKOverlay>)overlay;
  ```

MKOverlayView

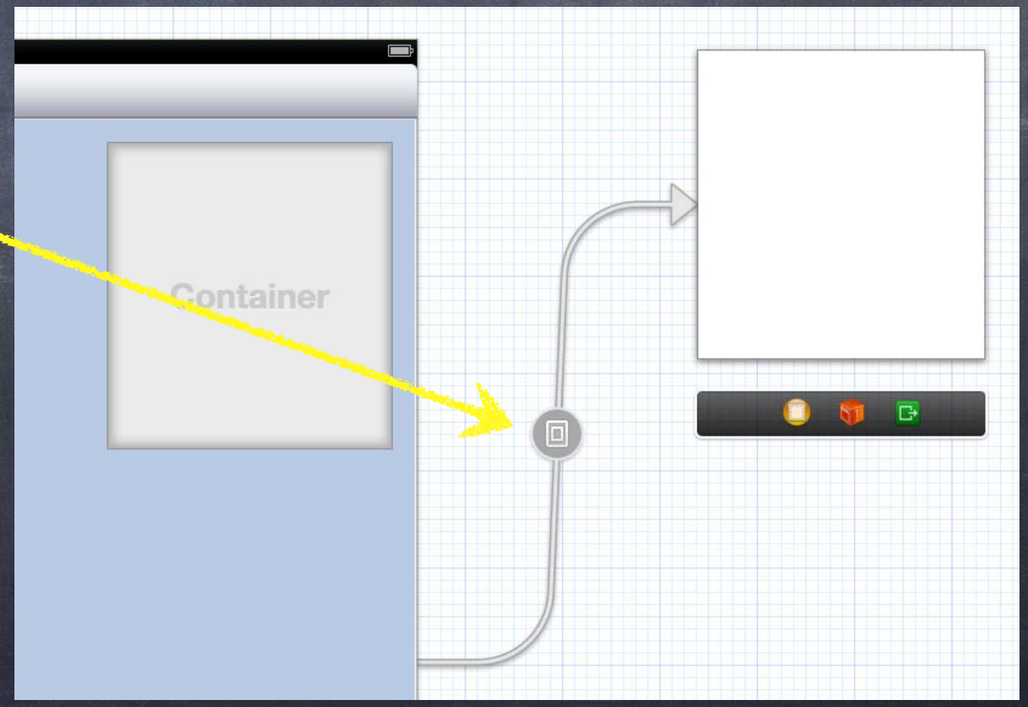
- Built-in Overlays and Renderers for numerous shapes …

  `MKCircleRenderer`
  `MKPolylineRenderer`
  `MKPolygonRenderer`
  `MKTileOverlayRenderer` // can also be used to replace the map data from Apple

  There's a whole set of `MKShape` and subclasses thereof for you to explore.

Embed Segues

- Putting a VC's `self.view` in another VC's view hierarchy!
  - This can be a very powerful encapsulation technique.
- Xcode makes this easy
  - Drag out a Container View from the object palette into the scene you want to embed it in.
  - Automatically sets up an "Embed Segue" from container VC to the contained VC.
- Embed Segue
  - Works just like other segues.
  - `prepareForSegue:sender:`, et. al.



Embed Segues

- Putting a VC's `self.view` in another VC's view hierarchy!
  - This can be a very powerful encapsulation technique.
- Xcode makes this easy
  - Drag out a Container View from the object palette into the scene you want to embed it in.
  - Automatically sets up an "Embed Segue" from container VC to the contained VC.
- Embed Segue
  - Works just like other segues.
  - `prepareForSegue:sender:`, et. al.
- View Loading Timing
  - Don't forget, though, that just like other segued-to VCs the embedded VC's outlets are not set at the time `prepareForSegue:sender:` is called.

Demo

- Photomania Maps
  - Instead of showing a table of photos, show a map of them.
  - Maps show `id <MKAnnotation>`s, so we'll turn a `Photo` object into an `MKAnnotation`!
  - Show thumbnails when users click on photo pins in the map.
  - Allow user to segue to a full view of the photo from the callout.
  - On iPad embed the map inside a `ImageViewController`.

Coming Up

- Homework
  Due Friday
- Friday
  Core Image
- Next Week
  Miscellaneous Topics