Exploring Google Maps for iOS

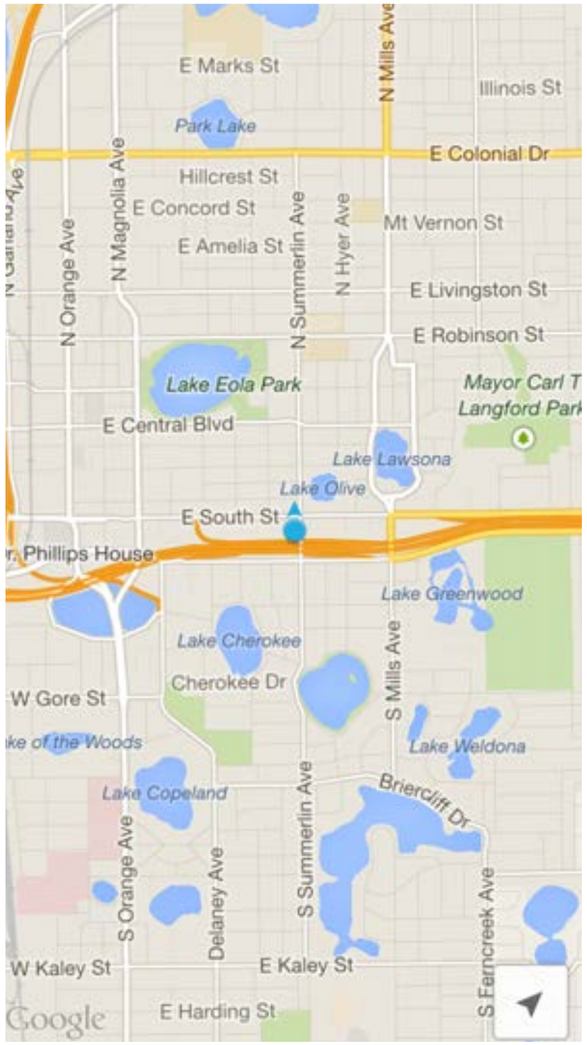
Exploring Google Maps for iOS

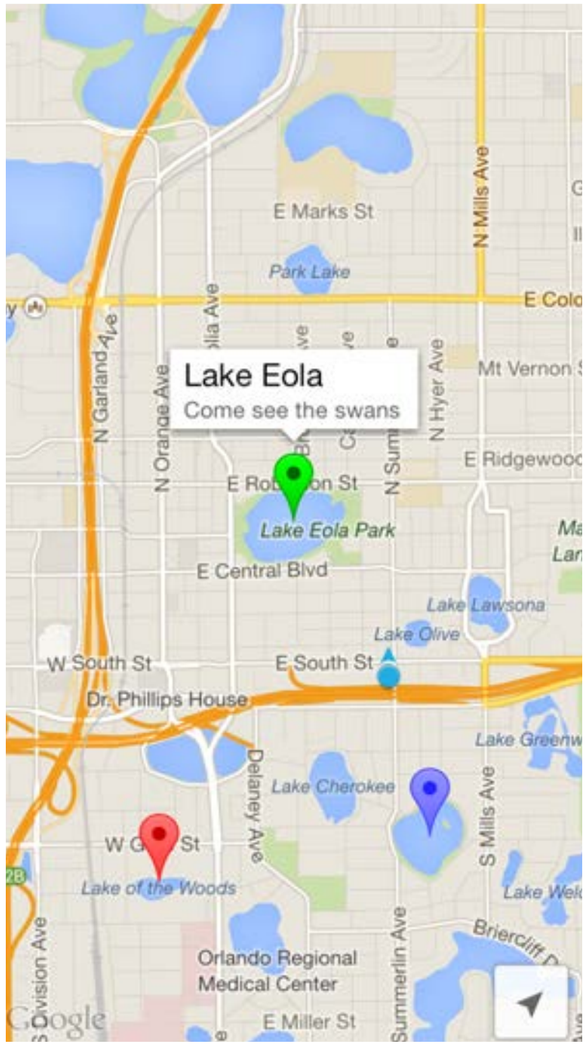
Level 1 - Displaying a map

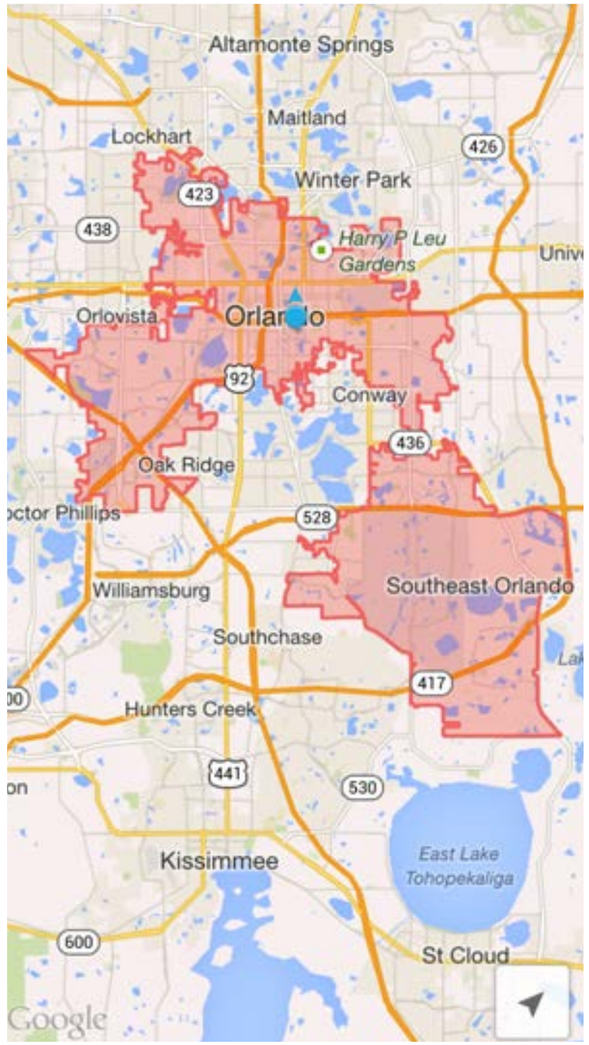
Exploring Google Maps for iOS

What is the Google Maps SDK for iOS?

A framework you can add into your app that lets you display a Google Map







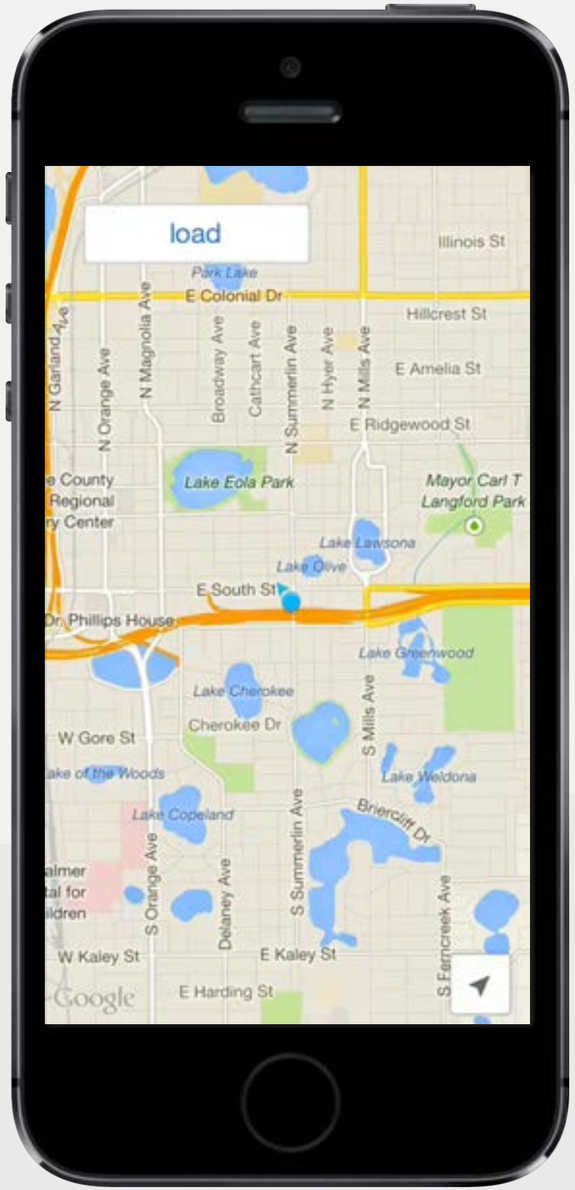
Why use Google Maps?

There are other options for displaying maps.

Using Google Maps gives you:

- Street View and Indoor Display views
- Easy integration with other Google services, like geocoding and directions
- Easy to respond to gestures like taps, swipes, and long presses

What are we going to learn to build?



- Map that displays markers from a network request
- Geocode an address and get directions between two places
- Draw lines and shapes on the map
- View marker locations in Street View

Exploring Google Maps for iOS

How to get the most out of this course

Having some basic iOS and Objective-C knowledge is necessary

tryobjective.codeschool.com

tryios.codeschool.com

Getting Started Step One: Get an API Key

Every app that includes Google Maps must have an API Key

`3kj2Jl3k1j3kl317hj13143hh6k1lkk4` ← The key will look like this

Each API Key is associated with an apps' Bundle Identifier

`com.jonfriskics.appname` ← You can check for your Bundle ID in your project's Info.plist file

Visit the Google Developers Console to set up an API Key for a Bundle Identifier

Exploring Google Maps for iOS

Getting Started Step Two: Set up the SDK

If you're completing the challenges in this Code School course, you won't need to set up the SDK because we've set it up for you.

If you're putting your own app together then watch the "Setting Up The SDK" screencast that walks you through all of the libraries and settings you need in Xcode.

Setting up your app to use the Google Maps SDK for iOS

AppDelegate.m

Import the GoogleMaps framework so you can access the SDK in your app

```
#import "AppDelegate.h"
#import <GoogleMaps/GoogleMaps.h>

@implementation AppDelegate

- (BOOL)application:(UIApplication *)application
    didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {

  self.window = [[UIWindow alloc] initWithFrame:[[UIScreen mainScreen] bounds]];

  [GMSServices provideAPIKey:@"3kj2Jl3k1j3kl317hj13143hh6k1lkk4"];

  self.window.backgroundColor = [UIColor whiteColor];
  [self.window makeKeyAndVisible];
  return YES;
}

@end
```

Use your own API key that you created when you start this course

Import and use LakeMapVC as the root view controller

AppDelegate.m

```
#import "AppDelegate.h"
#import <GoogleMaps/GoogleMaps.h>
#import "LakeMapVC.h"

@implementation AppDelegate

- (BOOL)application:(UIApplication *)application
    didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {

  self.window = [[UIWindow alloc] initWithFrame:[[UIScreen mainScreen] bounds]];

  [GMSServices provideAPIKey:@"3kj2Jl3k1j3kl317hj13143hh6k1lkk4"];

  LakeMapVC *lakeMapVC = [[LakeMapVC alloc] init];
  self.window.rootViewController = lakeMapVC;

  self.window.backgroundColor = [UIColor whiteColor];
  [self.window makeKeyAndVisible];
  return YES;
}

@end
```

Create a mapView property we'll use to display the map

LakeMapVC.m

You need to import the SDK into every class that needs to use it

```
#import "LakeMapVC.h"
#import <GoogleMaps/GoogleMaps.h>

@interface LakeMapVC ()

@property(strong, nonatomic) GMSMapView *mapView;

@end

@implementation LakeMapVC

@end
```

This property will hold a **strong** reference to the map so it stays on screen!

Exploring Google Maps for iOS

Create and display the map object when the view loads

LakeMapVC.m

```
@implementation LakeMapVC

- (void)viewDidLoad {
  [super viewDidLoad];
  self.mapView =
      [GMSMapView mapWithFrame:self.view.bounds camera:camera];
  [self.view addSubview:self.mapView];
}

@end
```

We need to create this

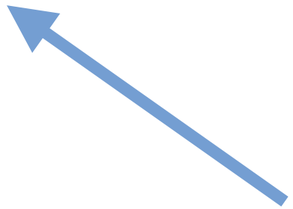
It is a `GMSCameraPosition` object

Exploring Google Maps for iOS

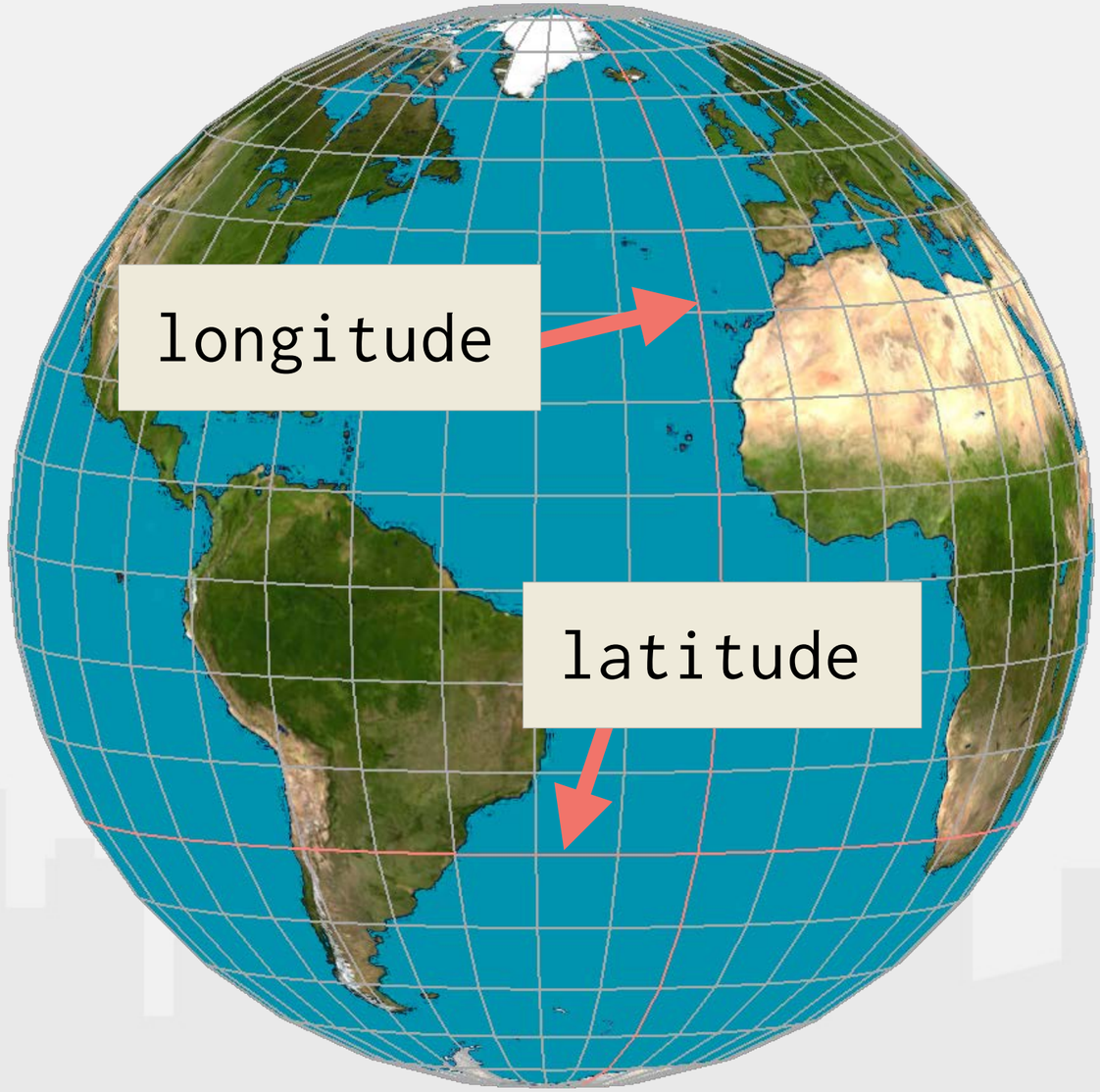
Creating a GMSCameraPosition

LakeMapVC.m

```
GMSCameraPosition *camera =
    [GMSCameraPosition cameraWithLatitude:
                                longitude:
                                     zoom:
                                  bearing:
                             viewingAngle:];
self.mapView =
    [GMSMapView mapWithFrame:self.view.bounds camera:camera];
[self.view addSubview:self.mapView];
```



GMSCameraPosition properties - latitude and longitude



A map of Earth is a grid of **points**

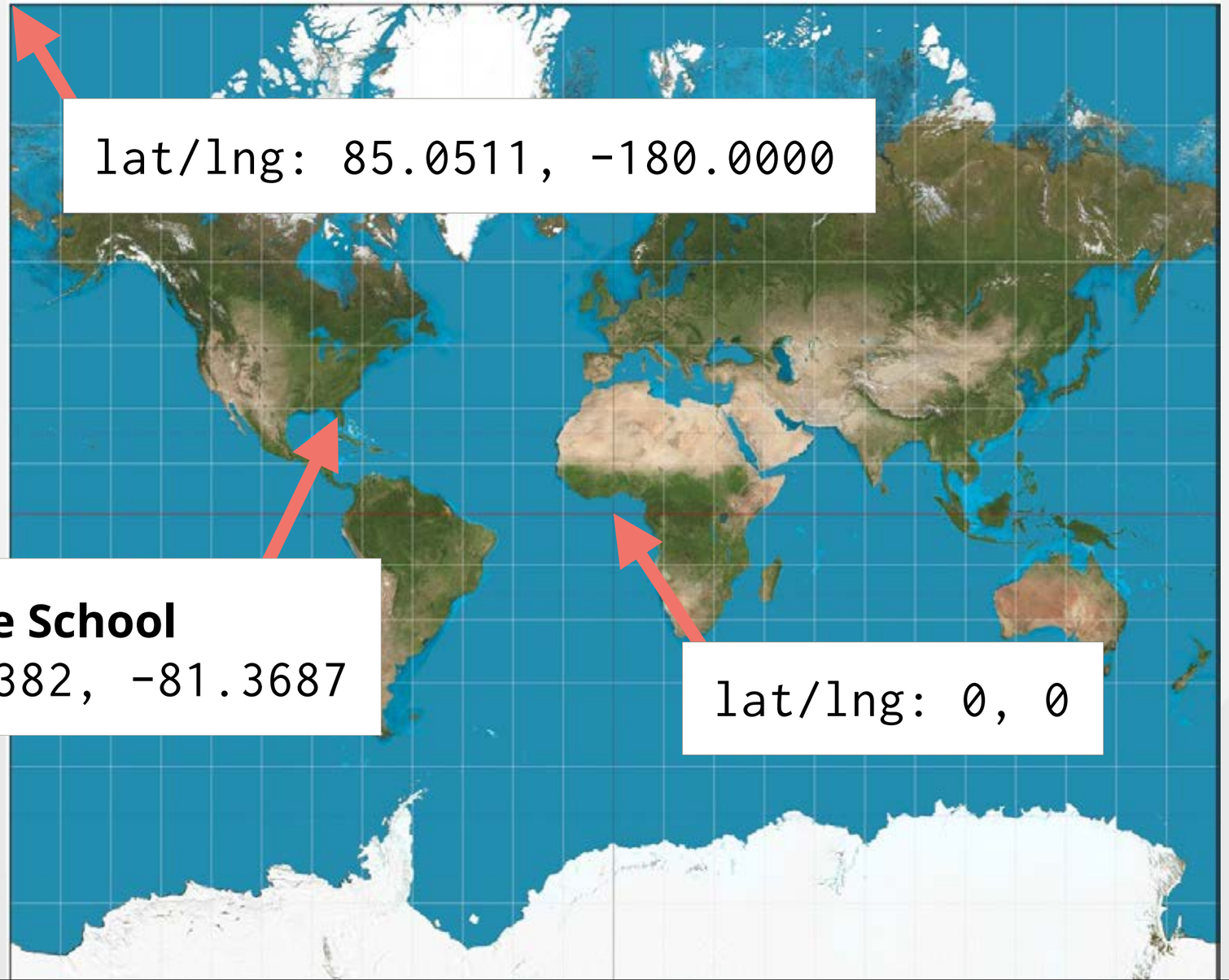
Every point has a **latitude/longitude** coordinate

latitude - runs east/west

longitude - runs north/south

Exploring Google Maps for iOS

Examples of latitude and longitude points on a map

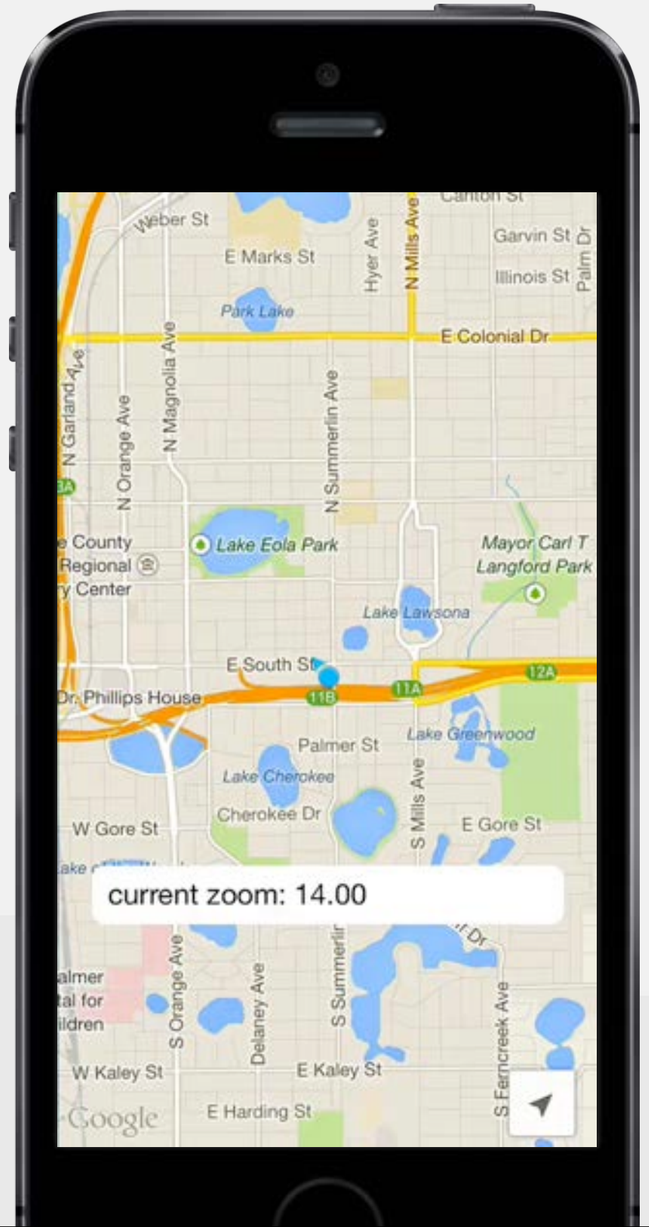


GMSCameraPosition properties - zoom

Show more or less of the map at once

A higher number means less of the map is showing (zoomed in)

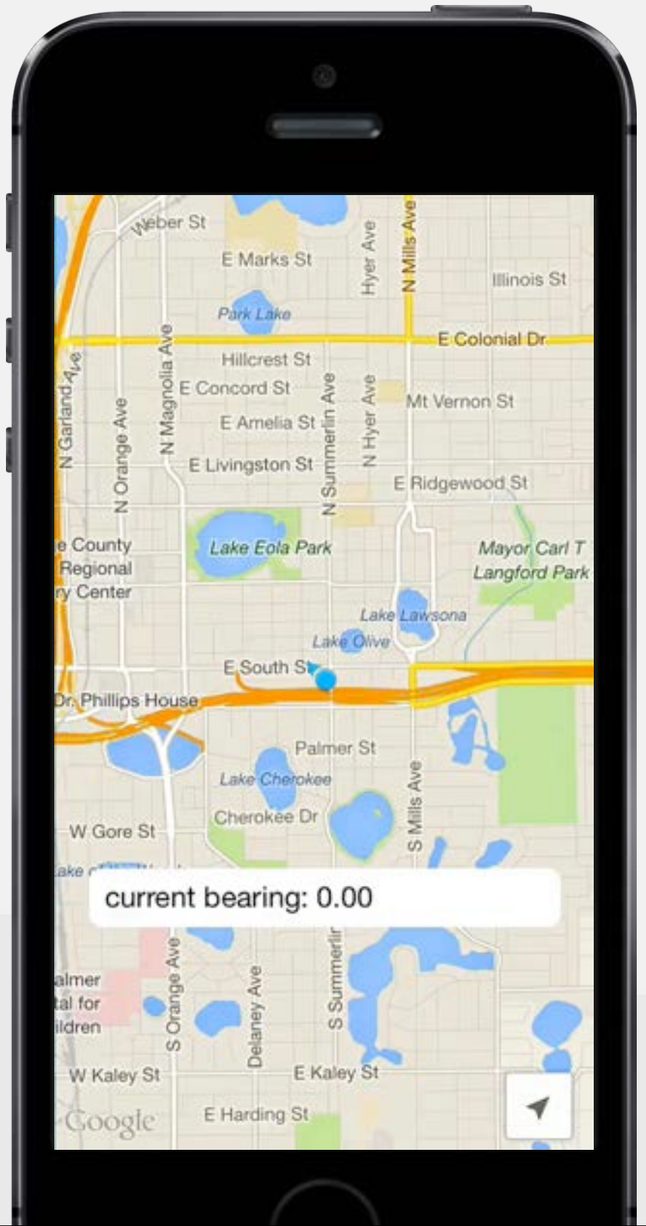
A lower number means more of the map is showing (zoomed out)



GMSCameraPosition properties - bearing

How much the map is rotated

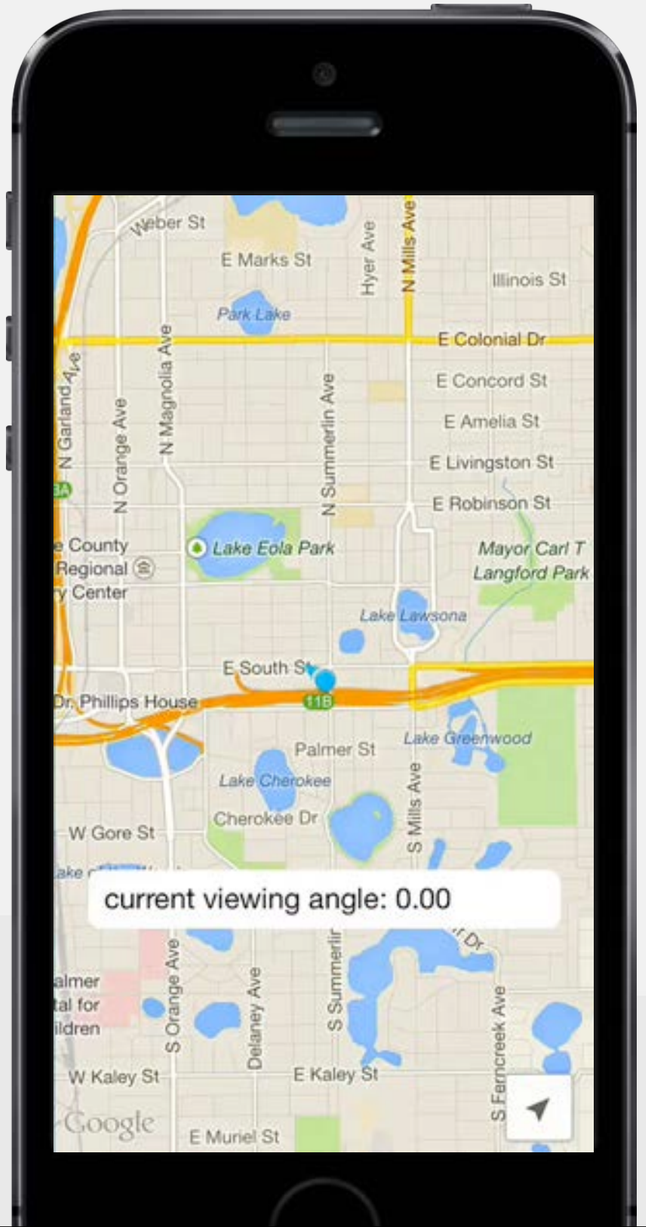
The default rotation is 0
(the top points north)



GMSCameraPosition properties - viewingAngle

Gives the impression of looking at the map at an angle

The default viewingAngle is 0
(looking straight down on the map)

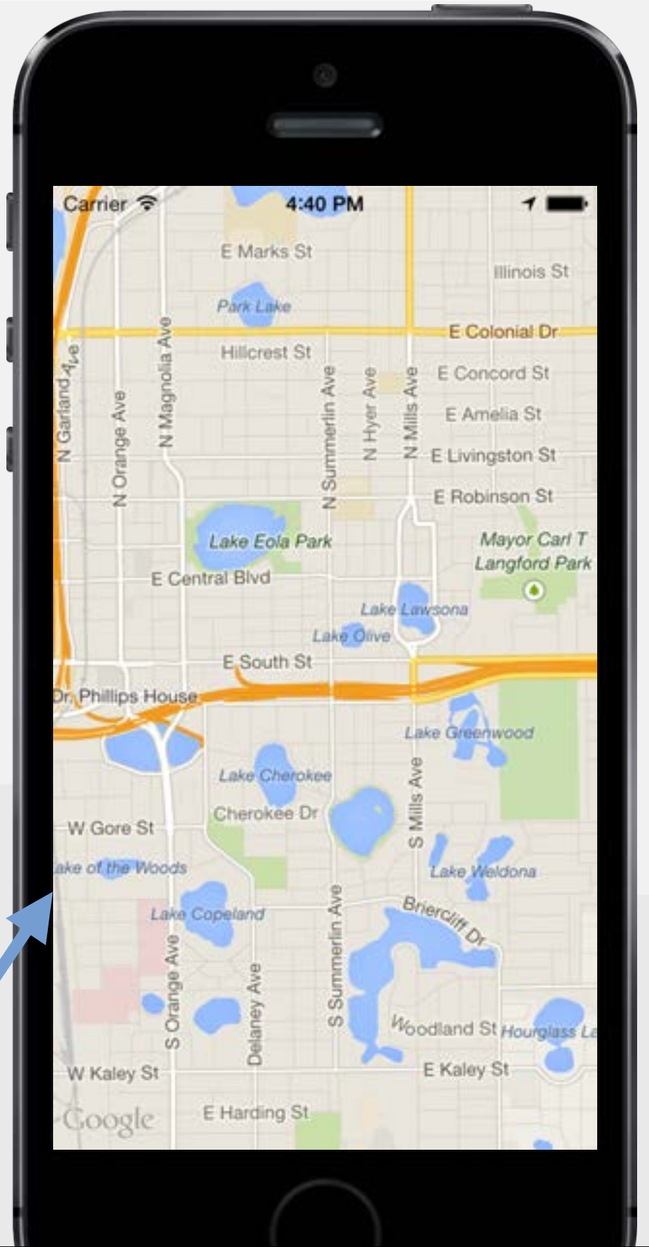


Creating a GMSCameraPosition object with a convenience initializer

LakeMapVC.m

```
@implementation LakeMapVC
- (void)viewDidLoad {
  [super viewDidLoad];
  GMSCameraPosition *camera =
      [GMSCameraPosition cameraWithLatitude:28.5382
                                  longitude:-81.3687
                                       zoom:14
                                    bearing:0
                               viewingAngle:0];
  self.mapView =
      [GMSMapView mapWithFrame:self.view.bounds camera:camera];
  [self.view addSubview:self.mapView];
}
@end
```

Here's what the map should look like now!



Changing the map type

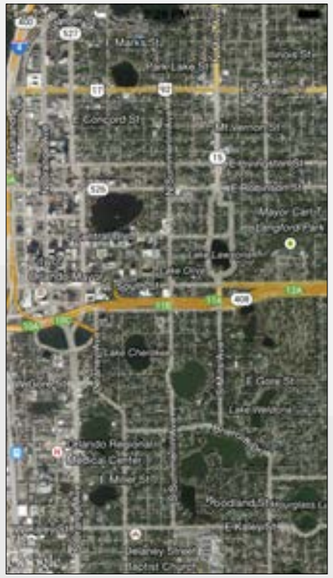
LakeMapVC.m

```
self.mapView =
    [GMSMapView mapWithFrame:self.view.bounds camera:camera];

self.mapView.mapType = kGMSTypeSatellite;

[self.view addSubview:self.mapView];
```

kGMSTypeNormal

kGMSTypeSatellite

kGMSTypeHybrid



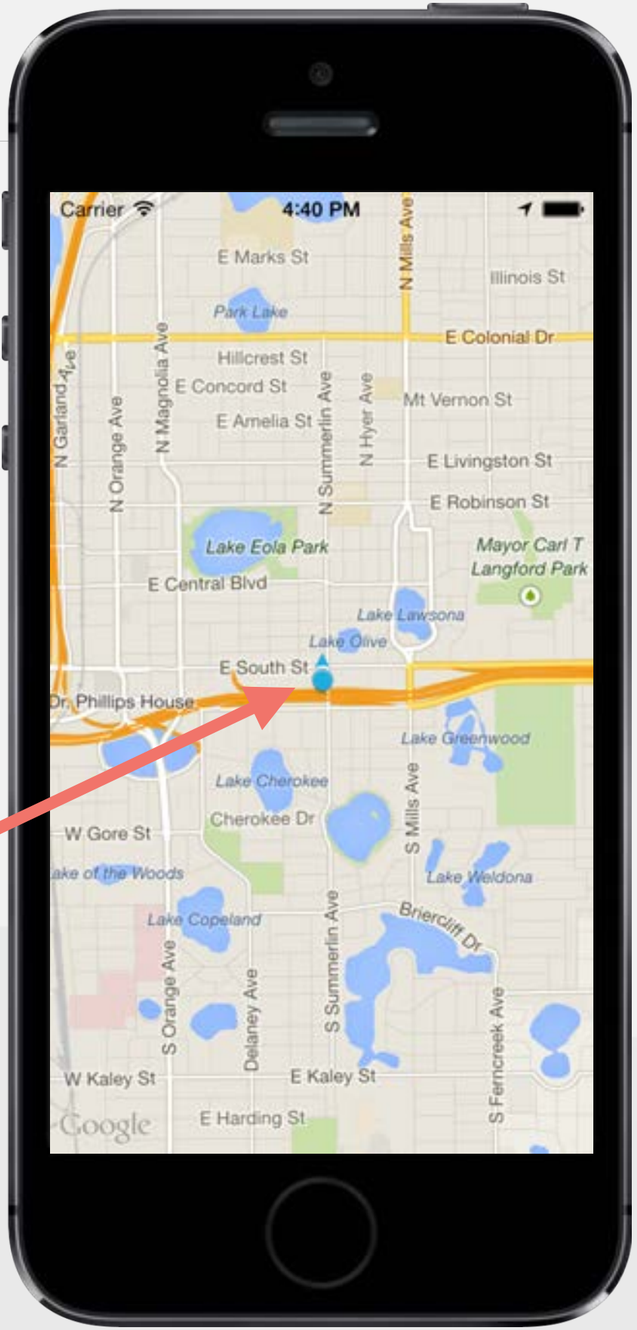
kGMSTypeTerrain

Showing your current location on the map

LakeMapVC.m

```
self.mapView =
    [GMSMapView mapWithFrame:self.view.bounds camera:camera];

self.mapView.mapType = kGMSTypeNormal;

self.mapView.myLocationEnabled = YES;

[self.view addSubview:self.mapView];
```

Your device's location



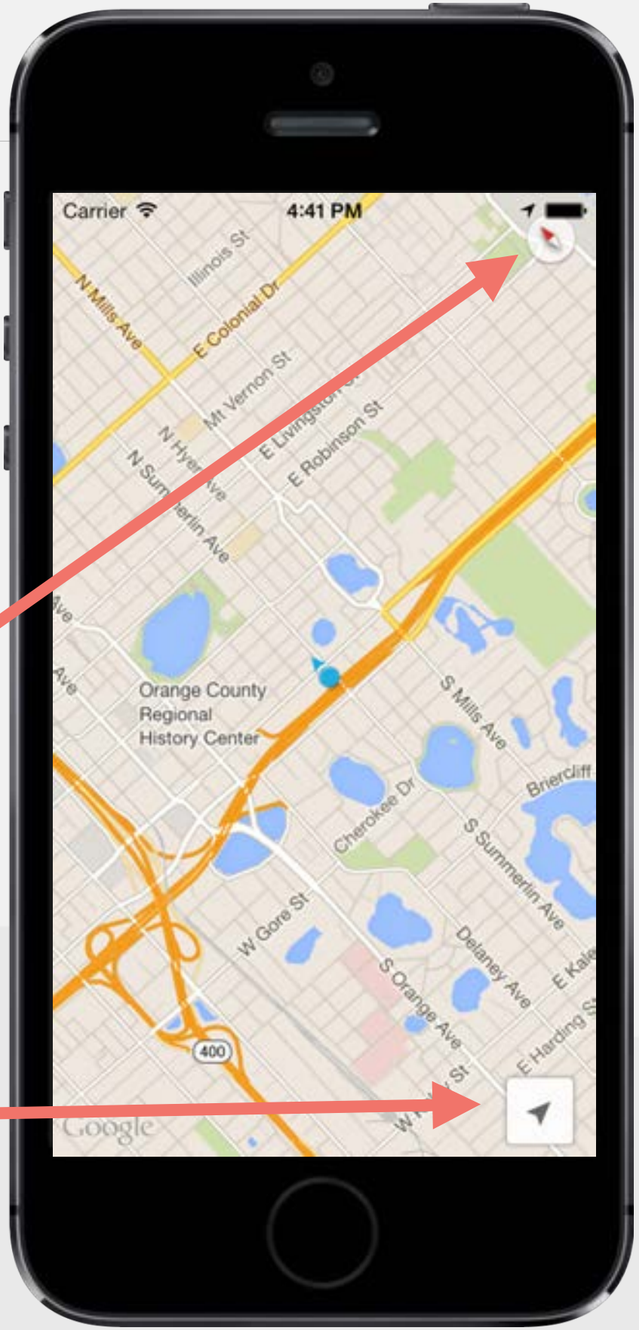
Displaying additional map controls

LakeMapVC.m

```
self.mapView =
    [GMSMapView mapWithFrame:self.view.bounds camera:camera];
self.mapView.mapType = kGMSTypeNormal;
self.mapView.myLocationEnabled = YES;
self.mapView.settings.compassButton = YES;
self.mapView.settings.myLocationButton = YES;
[self.view addSubview:self.mapView];
```

The compass shows up when the map is rotated

Find and center the map on your location



The mapView.settings property has a few other BOOL options

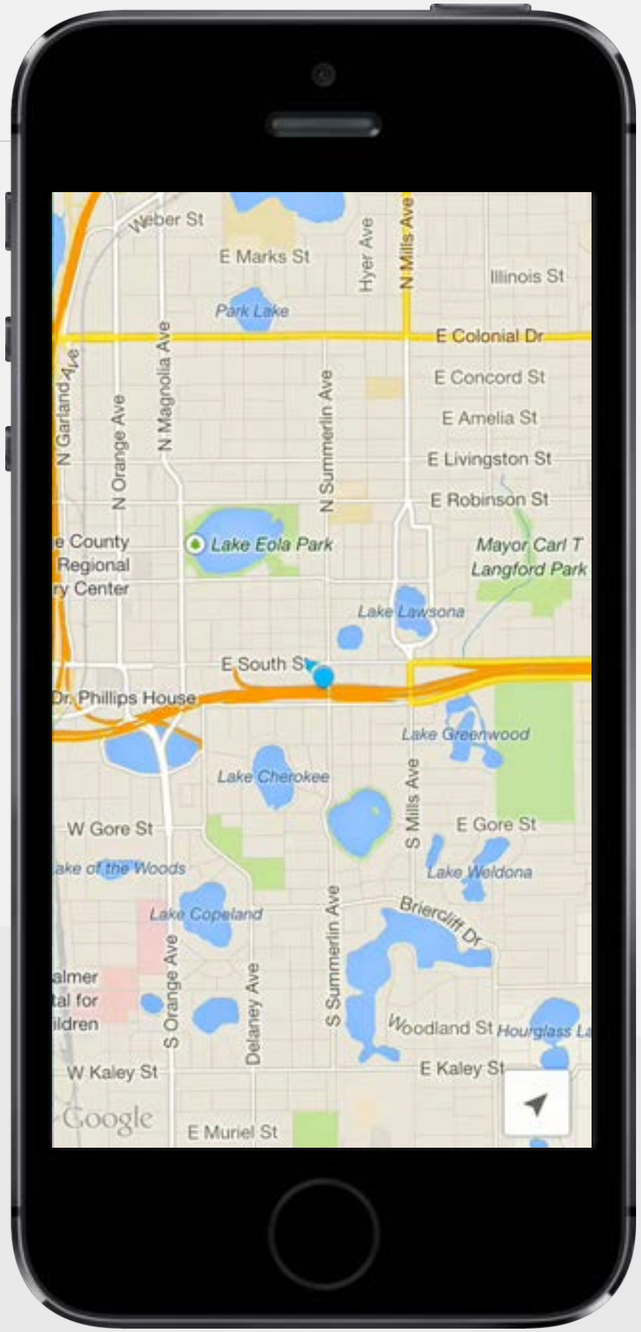
```
scrollGestures       rotateGestures
zoomGestures         indoorPicker
tiltGestures
consumesGesturesInView
```

Constraining the zoom options

LakeMapVC.m

```
self.mapView =
    [GMSMapView mapWithFrame:self.view.bounds camera:camera];
self.mapView.mapType = kGMSTypeNormal;
self.mapView.myLocationEnabled = YES;
self.mapView.settings.compassButton = YES;
self.mapView.settings.myLocationButton = YES;

[self.mapView setMinZoom:10 maxZoom:18];

[self.view addSubview:self.mapView];
```

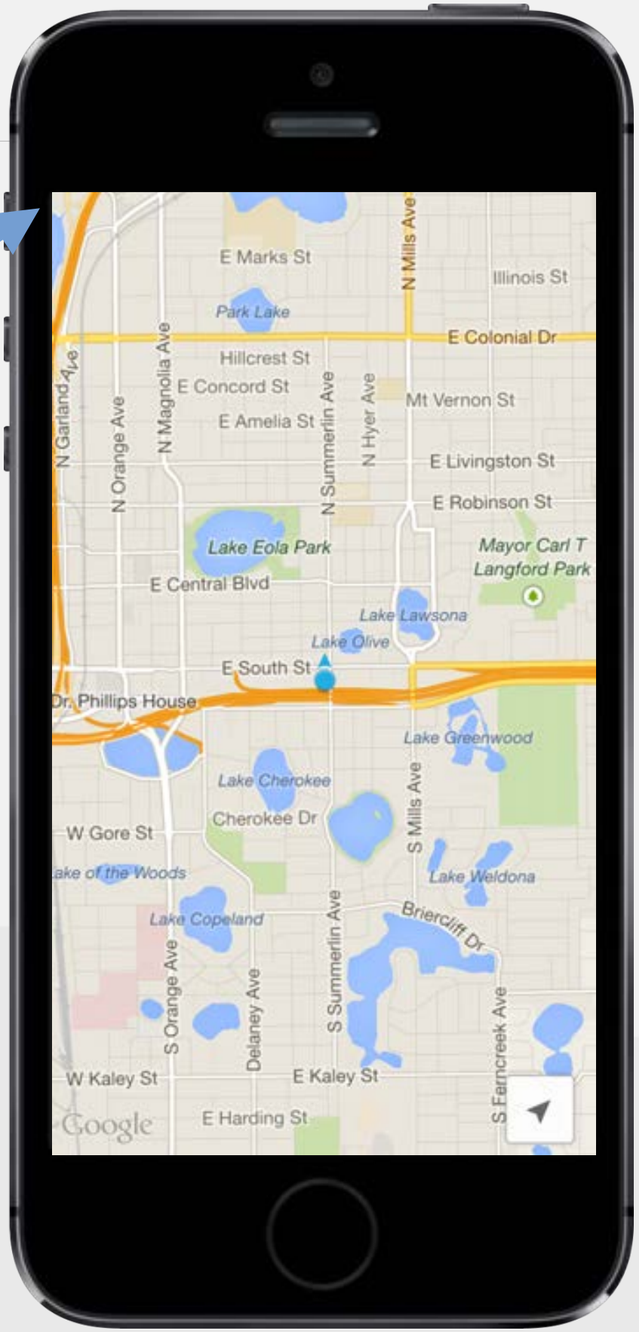
It might not always be desirable to allow zooming in or out all the way



Hiding the status bar in iOS 7

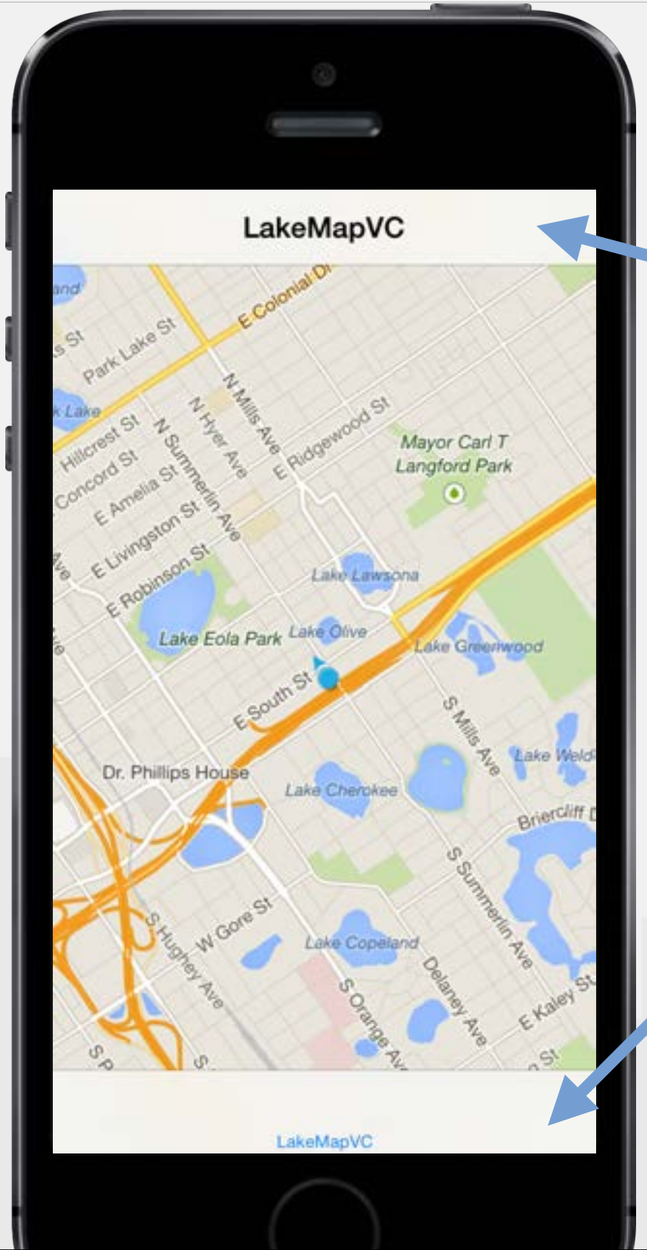
LakeMapVC.m

```
@implementation LakeMapVC

- (void)viewDidLoad {
  // map setup code
}

- (BOOL)prefersStatusBarHidden {
  return YES;
}

@end
```



In iOS 7, turn off the status bar if you're displaying the map full screen

Problem: Nav/tab bars hide the map controls



nav bar is hiding the compass

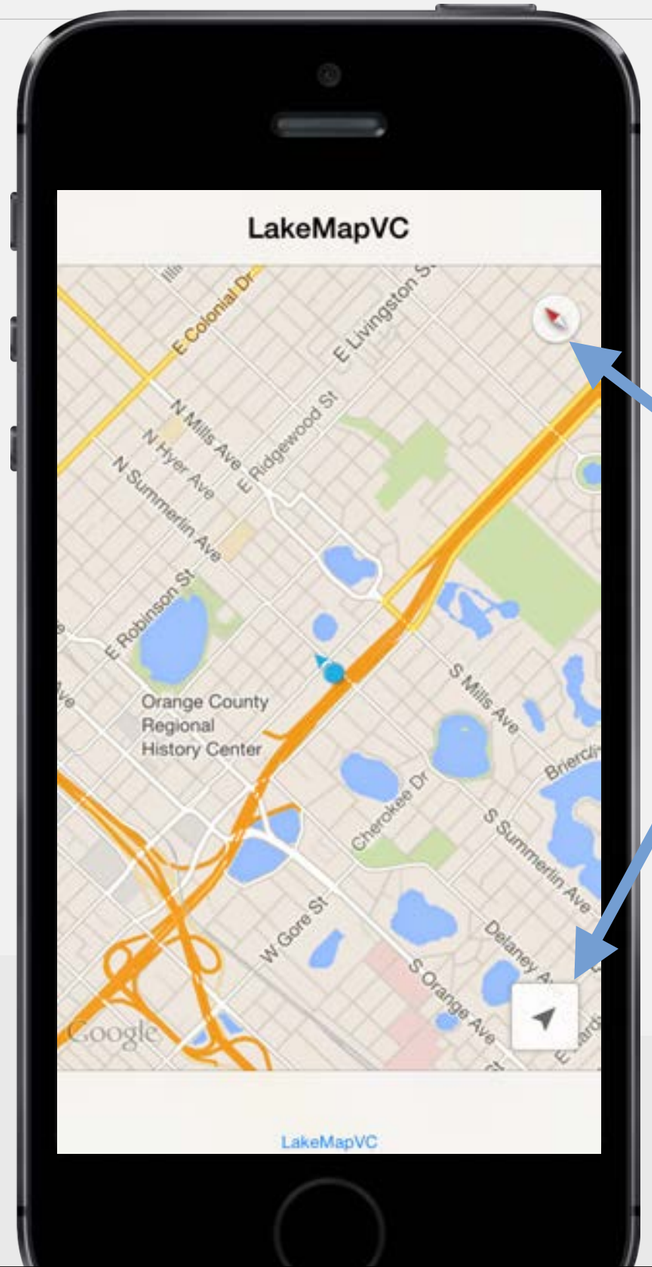
tab bar is hiding the `myLocation` button

Adjusting the map controls when nav/tab bars are showing

LakeMapVC.m

```
- (void)viewWillLayoutSubviews
{
   [super viewWillLayoutSubviews];

   self.mapView.padding =
       UIEdgeInsetsMake(self.topLayoutGuide.length + 5,
                        0,
                        self.bottomLayoutGuide.length + 5,
                        0);
}
```

NOTE: `topLayoutGuide` and `bottomLayoutGuide` are 0 in `viewDidLoad`, so access them in `viewWillLayoutSubviews` instead



controls are visible again because of map padding

Exploring Google Maps for iOS

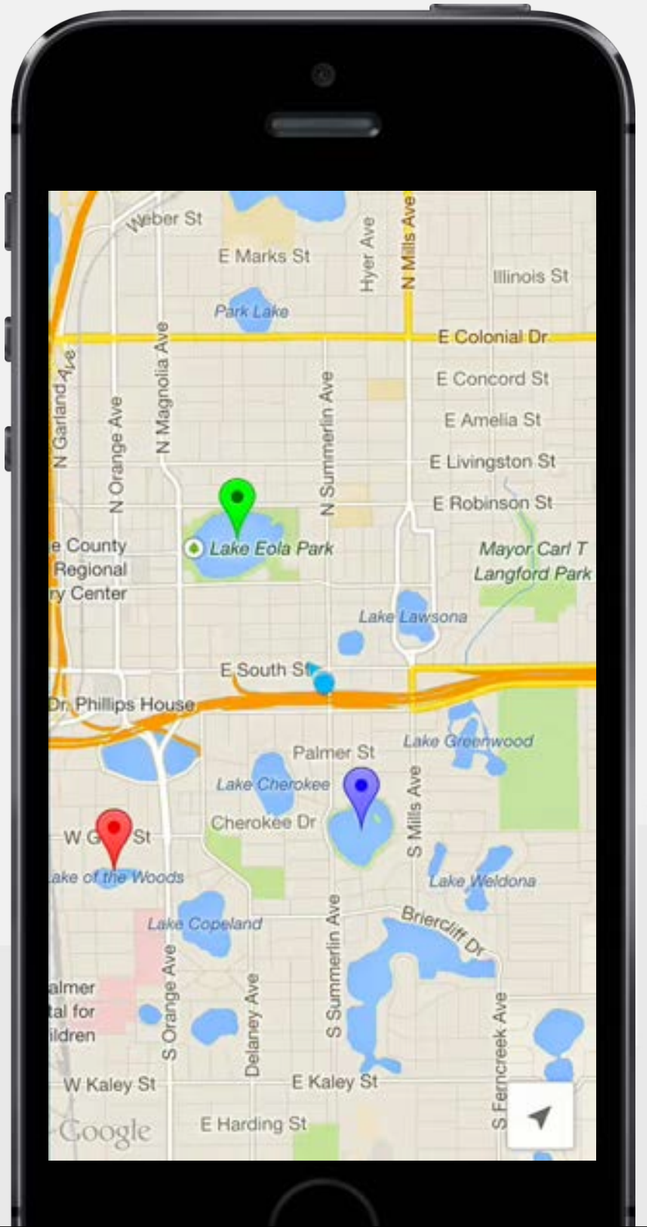
Adding markers to a map

Level 2

Exploring Google Maps for iOS

Demo: Markers and an info window

Markers are used to point out places on a map



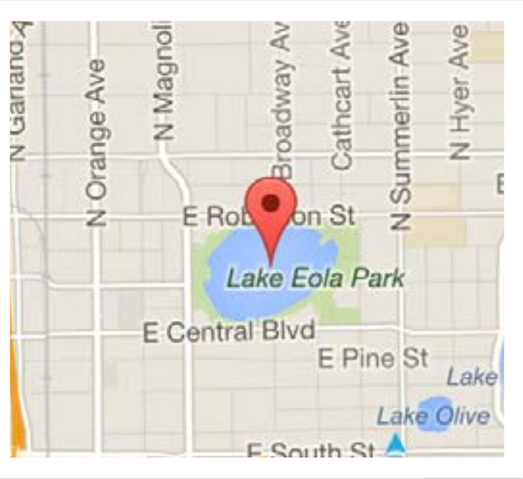
Exploring Google Maps for iOS

The simplest marker you can create

LakeMapVC.m

```
@implementation LakeMapVC

- (void)viewDidLoad {
  [super viewDidLoad];
  // map setup code

  GMSMarker *marker1 = [[GMSMarker alloc] init];
  marker1.position = CLLocationCoordinate2DMake(28.5441, -81.37301);
  marker1.map = self.mapView;
}
```

set the location of the marker



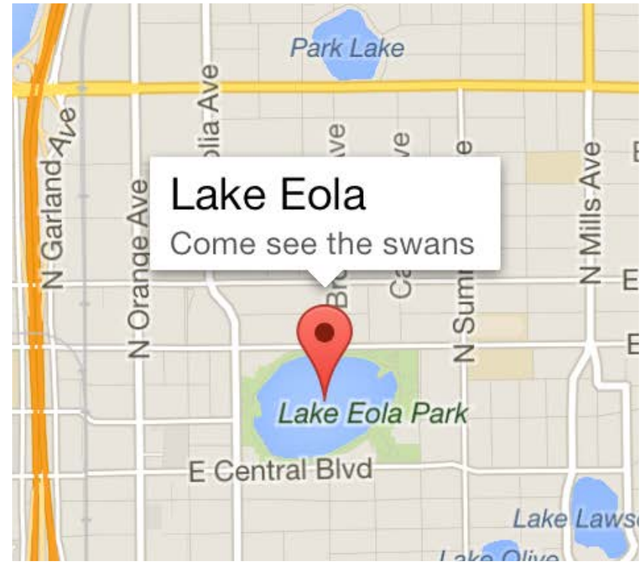
turn the marker **on** by setting this to your `mapView` property

if map is `nil`, the marker is **off**

Giving the marker data to show in the info window

LakeMapVC.m

```
@implementation LakeMapVC

- (void)viewDidLoad {
  [super viewDidLoad];
  // map setup code

  GMSMarker *marker1 = [[GMSMarker alloc] init];
  marker1.position = CLLocationCoordinate2DMake(...);
  marker1.title = @"Lake Eola";
  marker1.snippet = @"Come see the swans";
  marker1.map = self.mapView;
}
```



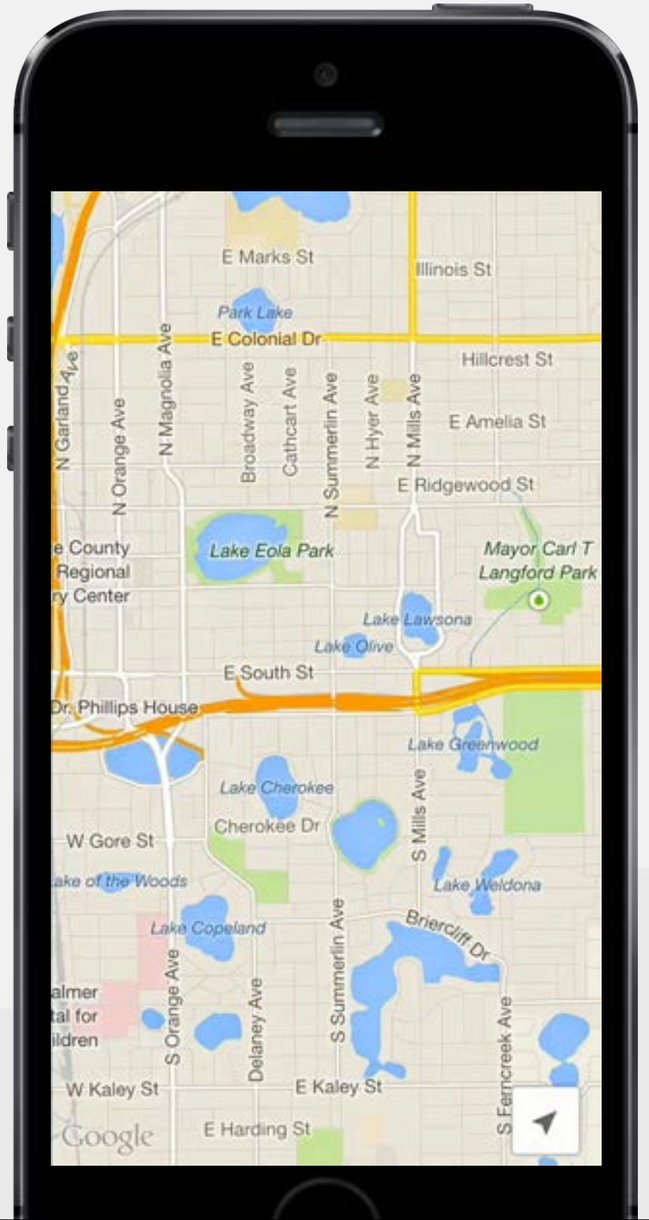
this will show up as a separate view when the marker is tapped

Animating in the marker when adding it

LakeMapVC.m

```
@implementation LakeMapVC

- (void)viewDidLoad {
  [super viewDidLoad];
  // map setup code

  GMSMarker *marker1 = [[GMSMarker alloc] init];
  marker1.position = CLLocationCoordinate2DMake(...);
  marker1.title = @"Lake Eola";
  marker1.snippet = @"Come see the swans";
  marker1.appearAnimation = kGMSMarkerAnimationPop;
  marker1.map = self.mapView;
}
```

this can also be kGMSMarkerAnimationNone

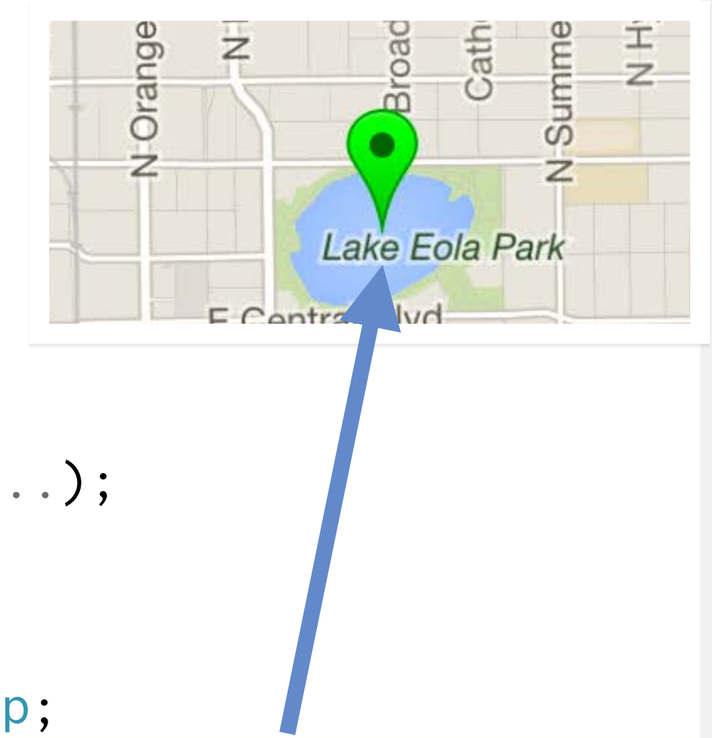


Changing the color of the marker

LakeMapVC.m

```
@implementation LakeMapVC

- (void)viewDidLoad {
  [super viewDidLoad];
  // map setup code

  GMSMarker *marker1 = [[GMSMarker alloc] init];
  marker1.position = CLLocationCoordinate2DMake(...);
  marker1.title = @"Lake Eola";
  marker1.snippet = @"Come see the swans";
  marker1.appearAnimation = kGMSMarkerAnimationPop;
  marker1.icon = [GMSMarker markerImageWithColor:[UIColor greenColor]];
  marker1.map = self.mapView;
}
```

GMSMarker color change

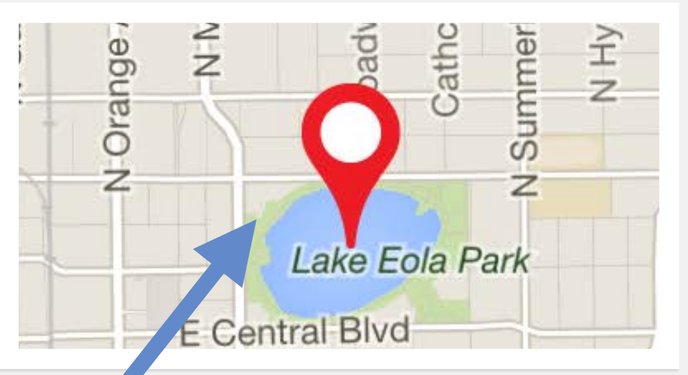


Changing the marker's icon

LakeMapVC.m

```
@implementation LakeMapVC

- (void)viewDidLoad {
  [super viewDidLoad];
  // map setup code

  GMSMarker *marker1 = [[GMSMarker alloc] init];
  marker1.position = CLLocationCoordinate2DMake(...);
  marker1.title = @"Lake Eola";
  marker1.snippet = @"Come see the swans";
  marker1.appearAnimation = kGMSMarkerAnimationPop;
  marker1.icon = [UIImage imageNamed:@"map-marker"]];
  marker1.map = self.mapView;
}
```

GMSMarker custom icon

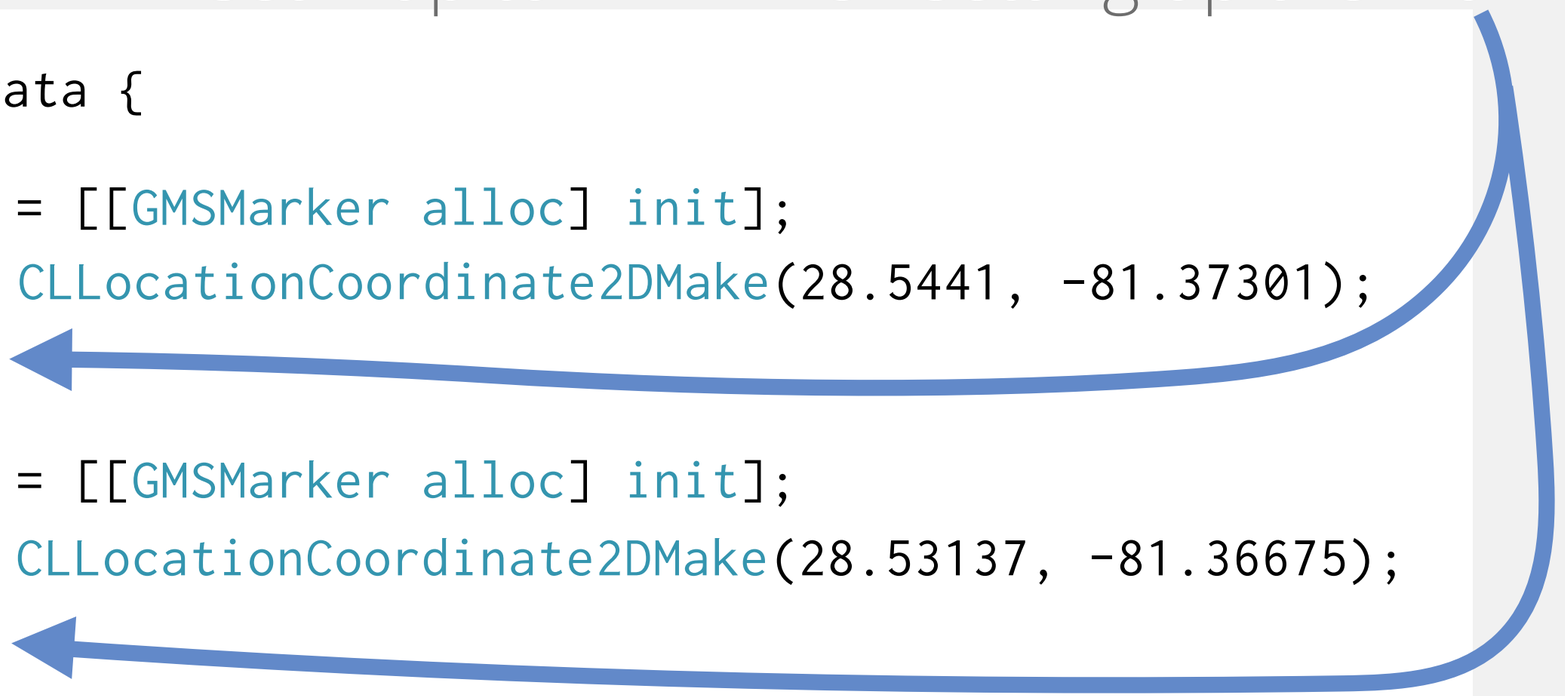


Creating more than one marker

LakeMapVC.m

set map to `nil` when setting up the marker

```
- (void)setupMarkerData {

  GMSMarker *marker1 = [[GMSMarker alloc] init];
  marker1.position = CLLocationCoordinate2DMake(28.5441, -81.37301);
  marker1.map = nil;

  GMSMarker *marker2 = [[GMSMarker alloc] init];
  marker2.position = CLLocationCoordinate2DMake(28.53137, -81.36675);
  marker2.map = nil;

}
```



this will leave the markers **off**
until we're ready to turn them **on**

Store markers in a set so you can turn them on and off at once

LakeMapVC.m

```
@interface LakeMapVC ()

@property(strong, nonatomic) GMSMapView *mapView;
@property(copy, nonatomic) NSSet *markers;

@end

@implementation LakeMapVC
- (void)setupMarkerData {
  GMSMarker *marker1 = [[GMSMarker alloc] init];
  ...
  self.markers =
      [NSSet setWithObjects:marker1, marker2, marker3, nil];
}
```

this initializer needs a `nil` terminator

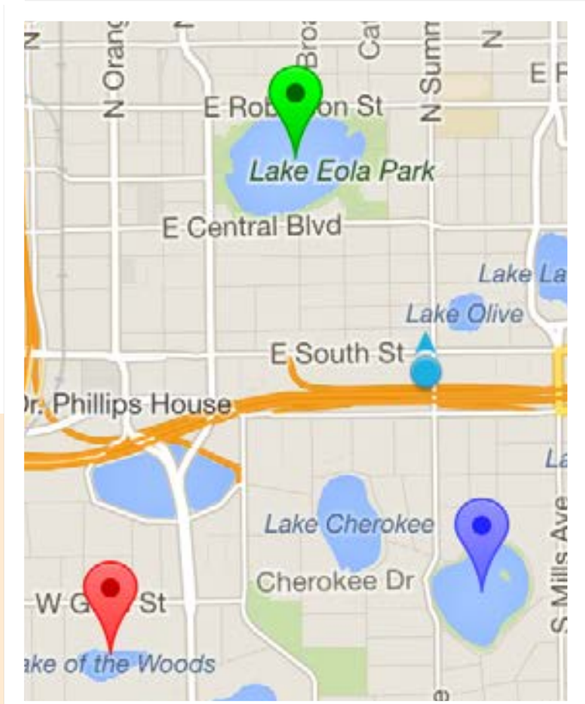
we'll explore why we're using a set instead of array in the next level

Exploring Google Maps for iOS

Loop through the set when you want to turn markers on

LakeMapVC.m

```
- (void)setupMarkerData {

  ...

  self.markers =
      [NSSet setWithObjects:marker1, marker2, marker3, nil];

  [self drawMarkers];
}

- (void)drawMarkers {
  for(GMSMarker *marker in self.markers) {
    marker.map = self.mapView;
  }
}
```



Only try to display markers if they aren't already on

LakeMapVC.m

```
- (void)setupMarkerData {
  ...
  self.markers =
      [NSSet setWithObjects:marker1, marker2, marker3, nil];
  [self drawMarkers];
}

- (void)drawMarkers {
  for(GMSMarker *marker in self.markers) {
    if(marker.map == nil) {
      marker.map = self.mapView;
    }
  }
}
```

Adopt and set the delegate so you can access delegate methods

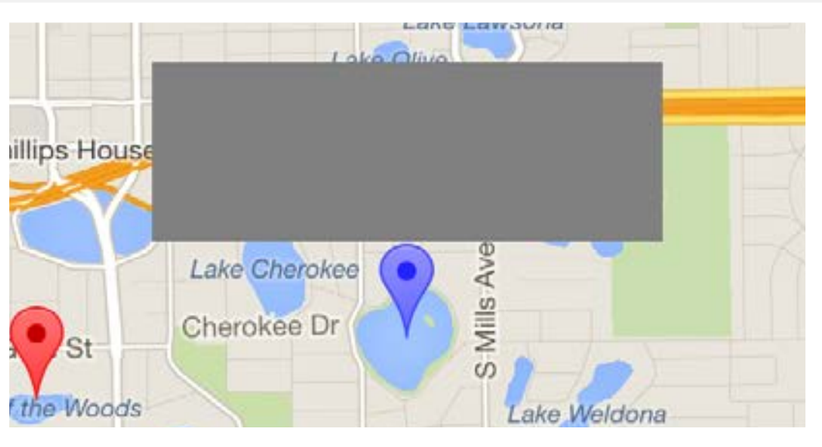
LakeMapVC.m

```
@interface LakeMapVC () <GMSMapViewDelegate>
...
@end

@implementation LakeMapVC

- (void)viewDidLoad {
  [super viewDidLoad];
  GMSCameraPosition *camera = ...
  self.mapView =
      [GMSMapView mapWithFrame:self.view.bounds camera:camera];
  self.mapView.delegate = self;
  ...
```

Display a custom info window

LakeMapVC.m

```
- (UIView *)mapView:(GMSMapView *)mapView
    markerInfoWindow:(GMSMarker *)marker {

  UIView *infoWindow = [[UIView alloc] init];
  infoWindow.frame = CGRectMake(0, 0, 200, 70);
  infoWindow.backgroundColor = [UIColor grayColor];
  return infoWindow;
}
```

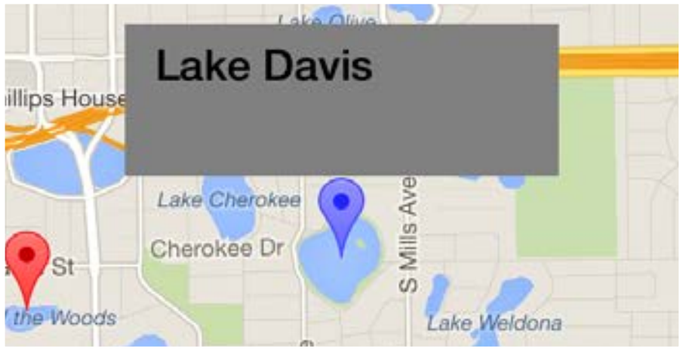


Use the selected marker's title property for the info window's main text

LakeMapVC.m

this method has access to the marker that was tapped

```
- (UIView *)mapView:(GMSMapView *)mapView
    markerInfoWindow:(GMSMarker *)marker {

  UIView *infoWindow = [[UIView alloc] init];
  // info window setup

  UILabel *titleLabel = [[UILabel alloc] init];
  titleLabel.frame = CGRectMake(14, 11, 175, 16);
  [infoWindow addSubview:titleLabel];
  titleLabel.text = marker.title;

  return infoWindow;
}
```

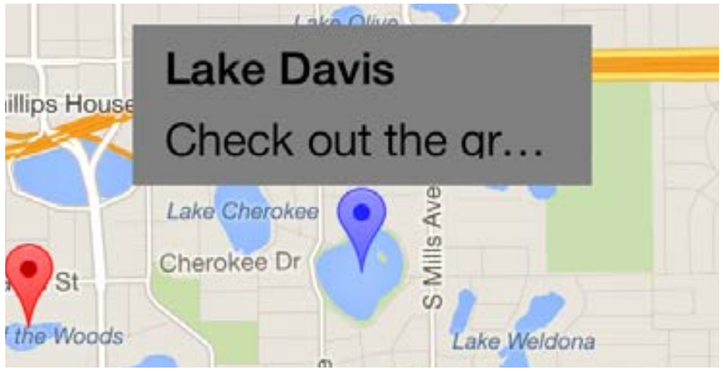


use that marker's `title` property for the label text

Use the selected marker's snippet property for the info window's additional text

LakeMapVC.m

```
- (UIView *)mapView:(GMSMapView *)mapView
    markerInfoWindow:(GMSMarker *)marker {

  UIView *infoWindow = [[UIView alloc] init];
  // info window setup
  // title label setup

  UILabel *snippetLabel = [[UILabel alloc] init];
  snippetLabel.frame = CGRectMake(14, 42, 175, 16);
  [infoWindow addSubview:snippetLabel];
  snippetLabel.text = marker.snippet;

  return infoWindow;
}
```



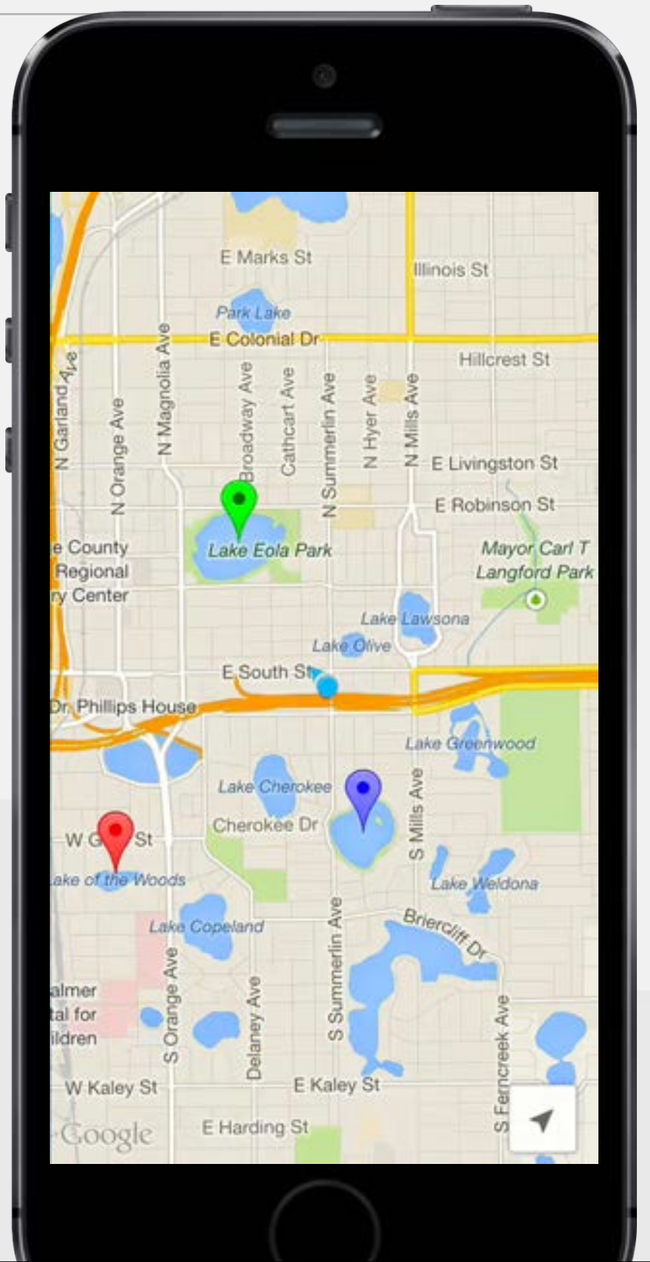
Trying to make the info window interactive

LakeMapVC.m

```
- (UIView *)mapView:(GMSMapView *)mapView
    markerInfoWindow:(GMSMarker *)marker {

  UIView *infoWindow = [[UIView alloc] init];
  // info window setup
  // title label setup
  // snippet label setup

  UIButton *infoWindowButton = [UIButton ...
  [infoWindow addSubview:infoWindowButton];

  return infoWindow;
}
```

this method returns a `UIView`, but the `GMSMapView` renders it as a static image

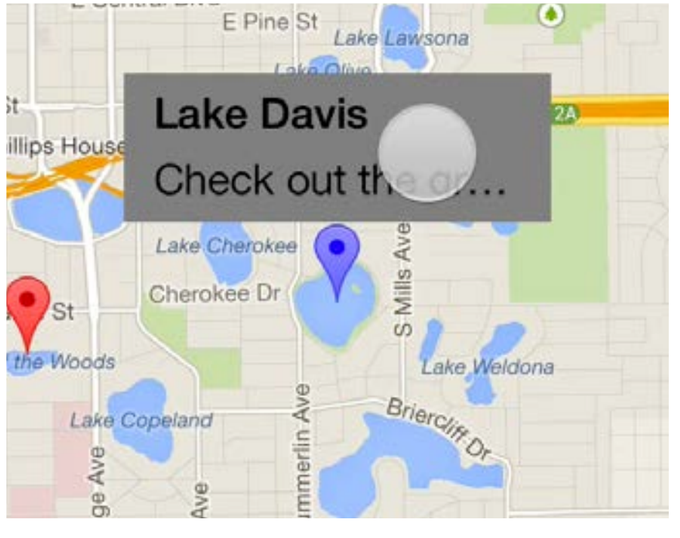
This means you can't add anything interactive to the info window



Use a delegate method to detect when the info window is tapped

LakeMapVC.m

```
- (void)mapView:(GMSMapView *)mapView
    didTapInfoWindowOfMarker:(GMSMarker *)marker {

  NSString *message =
      [NSString stringWithFormat:@"You tapped the info window
          for the %@ marker", marker.title];

  UIAlertView *windowTapped = [[UIAlertView alloc]
          initWithTitle:@"Info Window Tapped!"
                message:message
               delegate:nil
      cancelButtonTitle:@"Alright!"
      otherButtonTitles:nil];

  [windowTapped show];
}
```

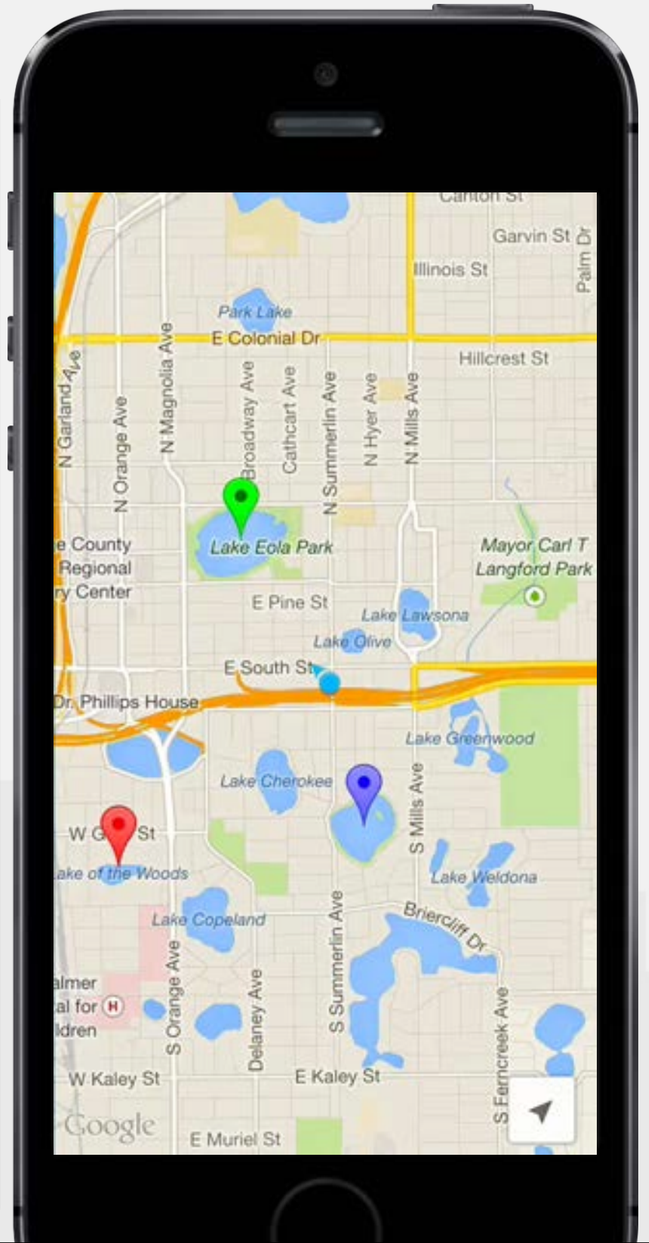


instead of an alert view, this might push to a different VC as part of a navigation controller

Use a custom background image to style the info window

LakeMapVC.m

```
- (UIView *)mapView:(GMSMapView *)mapView
    markerInfoWindow:(GMSMarker *)marker {

  UIView *infoWindow = [[UIView alloc] init];
  infoWindow.frame = CGRectMake(0, 0, 200, 70);

  UIImageView *backgroundImage =
      [[UIImageView alloc]
          initWithImage:[UIImage imageNamed:@"infoWindow"]];
  [infoWindow addSubview:backgroundImage];

  // also adjust styles for title and snippet labels

  return infoWindow;
}
```



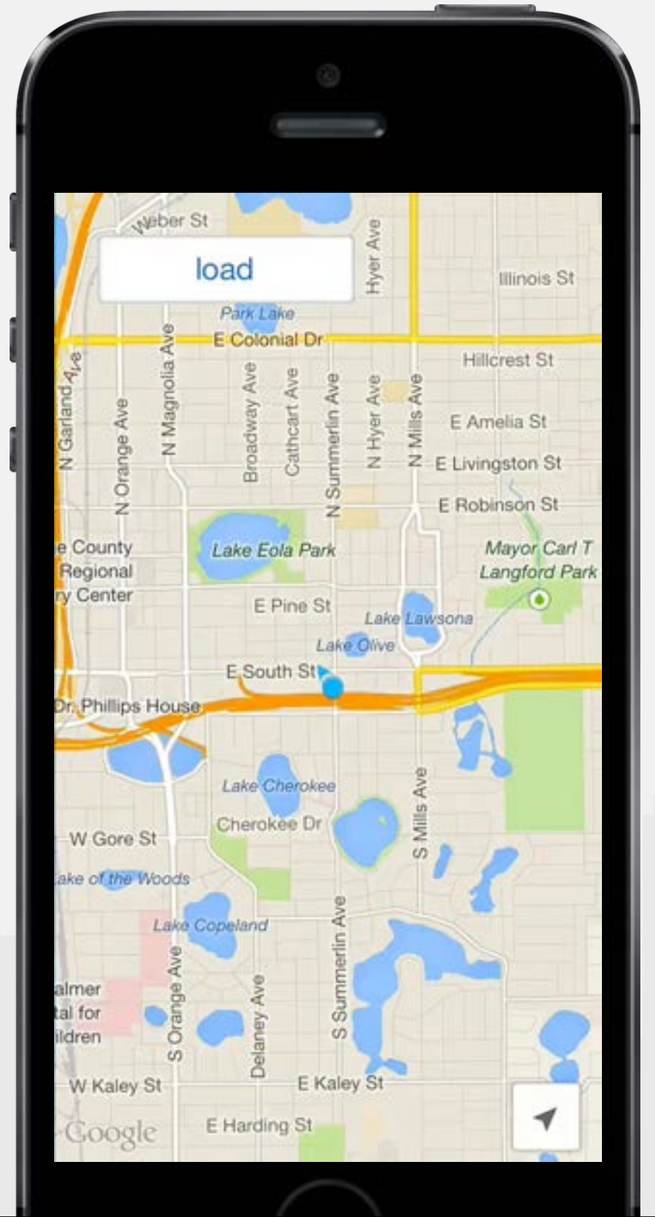
Exploring Google Maps for iOS

Creating markers from a network request

Level 3

Exploring Google Maps for iOS

Demo: Creating markers from a network request



Exploring Google Maps for iOS

Setting up the NSURLSession you'll use to make the network request

LakeMapVC.m

```
@interface LakeMapVC () <GMSMapViewDelegate>
...
@property(strong, nonatomic) NSURLSession *markerSession;

@end

@implementation LakeMapVC
- (void)viewDidLoad {
  [super viewDidLoad];
  NSURLSessionConfiguration *config =
      [NSURLSessionConfiguration defaultSessionConfiguration];
  config.URLCache = [[NSURLCache alloc] initWithMemoryCapacity:2 * 1024 * 1024
                                                  diskCapacity:10 * 1024 * 1024
                                                      diskPath:@"MarkerData"];
  self.markerSession = [NSURLSession sessionWithConfiguration:config];
}
```

review how **NSURLSession** works in Code School's Core iOS 7 course, Level 8

Set up the request

runs when you tap the load button

LakeMapVC.m

```
- (void)downloadMarkerData:(id)sender {

  NSURL *lakesURL =
      [NSURL URLWithString:@"http://googlemaps.codeschool.com/lakes.json"];

  NSURLSessionDataTask *task = [self.markerSession dataTaskWithURL:lakesURL
      completionHandler:^(NSData *data, NSURLResponse *response, NSError *e) {

  }];

  [task resume];
}
```

This URL gets JSON data

set up the request

start the request

Turn the JSON response into an NSArray

LakeMapVC.m

the response comes back as JSON

```
- (void)downloadMarkerData:(id)sender {

  NSURL *lakesURL =
      [NSURL URLWithString:@“http://googlemaps.codeschool.com/lakes.json"];

  NSURLSessionDataTask *task = [self.markerSession dataTaskWithURL:lakesURL
      completionHandler:^(NSData *data, NSURLResponse *response, NSError *e) {

    NSArray *json = [NSJSONSerialization JSONObjectWithData:data
                                                    options:0
                                                      error:nil];

    NSLog(@"json: %@",json);

  }];

  [task resume];
}
```

let’s look at the structure of this JSON response by logging it

serialize the JSON into an array

Examining the JSON structure

NSLog of json array

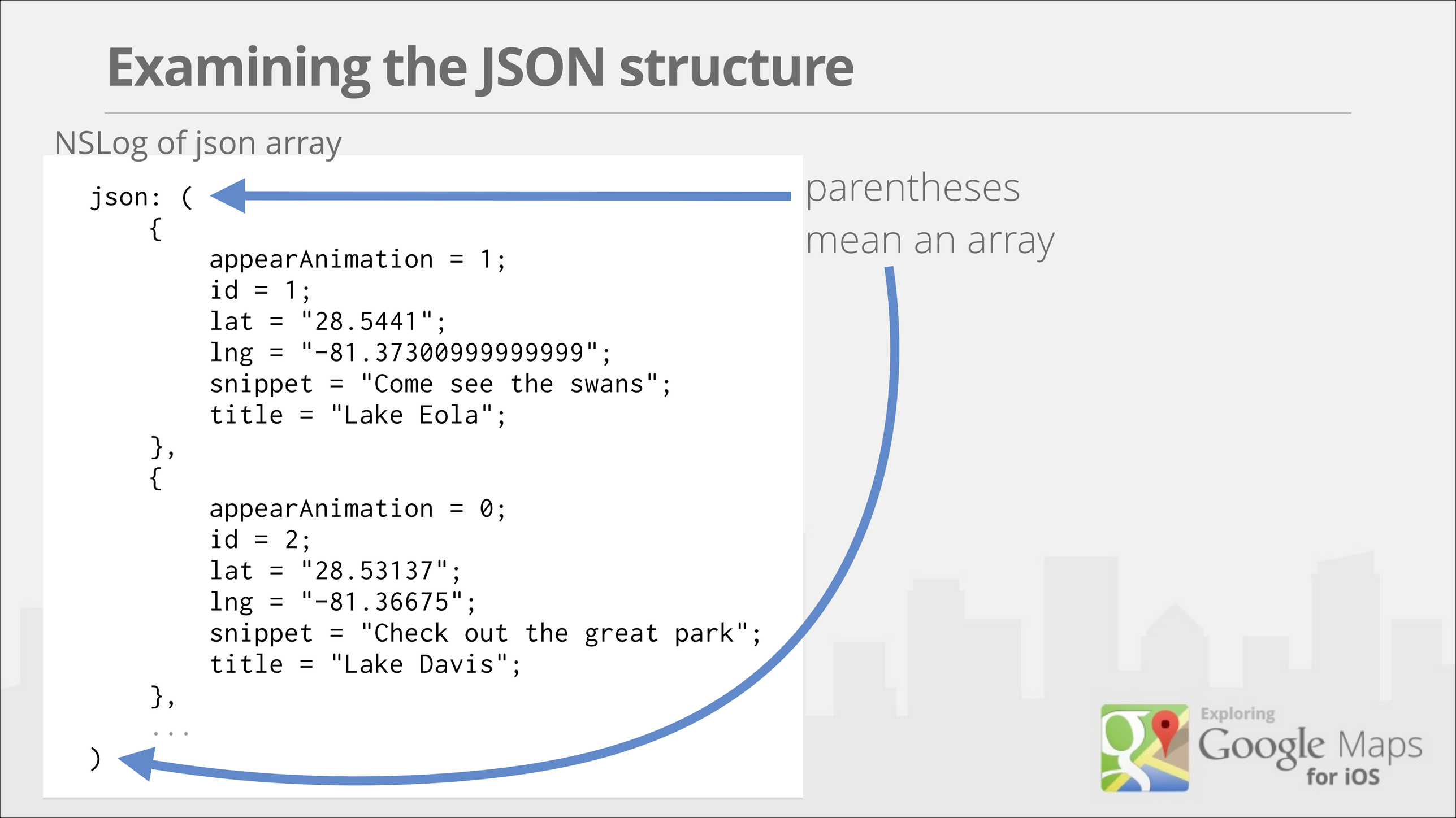
```
json: (
    {
        appearAnimation = 1;
        id = 1;
        lat = "28.5441";
        lng = "-81.37300999999999";
        snippet = "Come see the swans";
        title = "Lake Eola";
    },
    {
        appearAnimation = 0;
        id = 2;
        lat = "28.53137";
        lng = "-81.36675";
        snippet = "Check out the great park";
        title = "Lake Davis";
    },
    ...
)
```

Examining the JSON structure

NSLog of json array

```
json: (
    {
        appearAnimation = 1;
        id = 1;
        lat = "28.5441";
        lng = "-81.37300999999999";
        snippet = "Come see the swans";
        title = "Lake Eola";
    },
    {
        appearAnimation = 0;
        id = 2;
        lat = "28.53137";
        lng = "-81.36675";
        snippet = "Check out the great park";
        title = "Lake Davis";
    },
    ...
)
```

parentheses mean an array



Examining the JSON structure

NSLog of json array

```
json: (
    {
        appearAnimation = 1;
        id = 1;
        lat = "28.5441";
        lng = "-81.37300999999999";
        snippet = "Come see the swans";
        title = "Lake Eola";
    },
    {
        appearAnimation = 0;
        id = 2;
        lat = "28.53137";
        lng = "-81.36675";
        snippet = "Check out the great park";
        title = "Lake Davis";
    },
    ...
)
```

curly brackets
mean a dictionary

Exploring Google Maps for iOS

Examining the JSON structure

NSLog of json array

```
json: (
    {
        appearAnimation = 1;
        id = 1;
        lat = "28.5441";
        lng = "-81.37300999999999";
        snippet = "Come see the swans";
        title = "Lake Eola";
    },
    {
        appearAnimation = 0;
        id = 2;
        lat = "28.53137";
        lng = "-81.36675";
        snippet = "Check out the great park";
        title = "Lake Davis";
    },
    ...
)
```

this JSON response is an array of dictionaries

these are keys and values in the dictionary

Exploring Google Maps for iOS

Turn the JSON response into an NSArray

LakeMapVC.m

```
- (void)downloadMarkerData:(id)sender {

  NSURL *lakesURL =
      [NSURL URLWithString:@"http://googlemaps.codeschool.com/lakes"];

  NSURLSessionDataTask *task = [self.markerSession dataTaskWithURL:lakesURL
      completionHandler:^(NSData *data, NSURLResponse *response, NSError *e) {
    [self createMarkerObjectsWithJson:json];
  }];

  [task resume];
}
```

we'll write this method to do something with the JSON response data

Creating marker objects with the response data - the setup

LakeMapVC.m

```
- (void)createMarkerObjectsWithJson:(NSArray *)json {

}
```

1. get a mutable copy of the current markers in self.markers

2. loop through each item (dictionary) in the JSON array

2a. create a marker object for each dictionary in the JSON array

2b. add the new marker object to the mutable set

3. when the loop ends, save the current state of the mutable set as an immutable copy

4. call drawMarkers to update the displayed markers

Exploring Google Maps for iOS

Creating marker objects with the response data

1. get a mutable copy of the current markers in self.markers

LakeMapVC.m

```
- (void)createMarkerObjectsWithJson:(NSArray *)json {
    NSMutableSet *mutableSet = [[NSMutableSet alloc] initWithSet:self.markers];
}
```

Creating marker objects with the response data

2. loop through each item (a dictionary) in the JSON array

LakeMapVC.m

```
- (void)createMarkerObjectsWithJson:(NSArray *)json {
  NSMutableSet *mutableSet = [[NSMutableSet alloc] initWithSet:self.markers];
  for (NSDictionary *markerData in json) {

  }
}
```

Creating marker objects with the response data

3. when the loop ends, save the current state of the mutable set as an immutable copy

LakeMapVC.m

```
- (void)createMarkerObjectsWithJson:(NSArray *)json {
  NSMutableSet *mutableSet = [[NSMutableSet alloc] initWithSet:self.markers];
  for (NSDictionary *markerData in json) {

  }
  self.markers = [mutableSet copy];
}
```

Exploring Google Maps for iOS

Creating marker objects with the response data

4. call drawMarkers to update the markers to be displayed

LakeMapVC.m

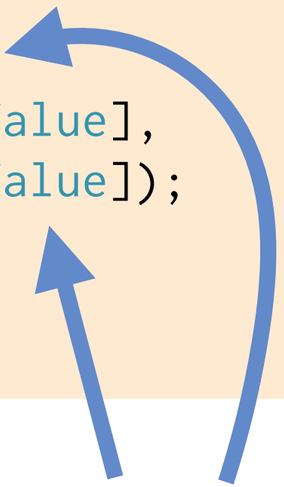
```
- (void)createMarkerObjectsWithJson:(NSArray *)json {
  NSMutableSet *mutableSet = [[NSMutableSet alloc] initWithSet:self.markers];
  for (NSDictionary *markerData in json) {

  }
  self.markers = [mutableSet copy];
  [self drawMarkers];
}
```

Creating marker objects with the response data

2a. create a marker object for each dictionary in the JSON array

LakeMapVC.m

```
- (void)createMarkerObjectsWithJson:(NSArray *)json {
  NSMutableSet *mutableSet = [[NSMutableSet alloc] initWithSet:self.markers];
  for (NSDictionary *markerData in json) {
    GMSMarker *newMarker = [[GMSMarker alloc] init];
    newMarker.appearAnimation = [markerData[@"appearAnimation"] integerValue];
    newMarker.position = CLLocationCoordinate2DMake([markerData[@"lat"] doubleValue],
                                                    [markerData[@"lng"] doubleValue]);
    newMarker.title = markerData[@"title"];
    newMarker.snippet = markerData[@"snippet"];
    newMarker.map = nil;
  }
  self.markers = [mutableSet copy];
  [self drawMarkers];
}
```



make sure the response data is in the format that **GMSMarker** is expecting

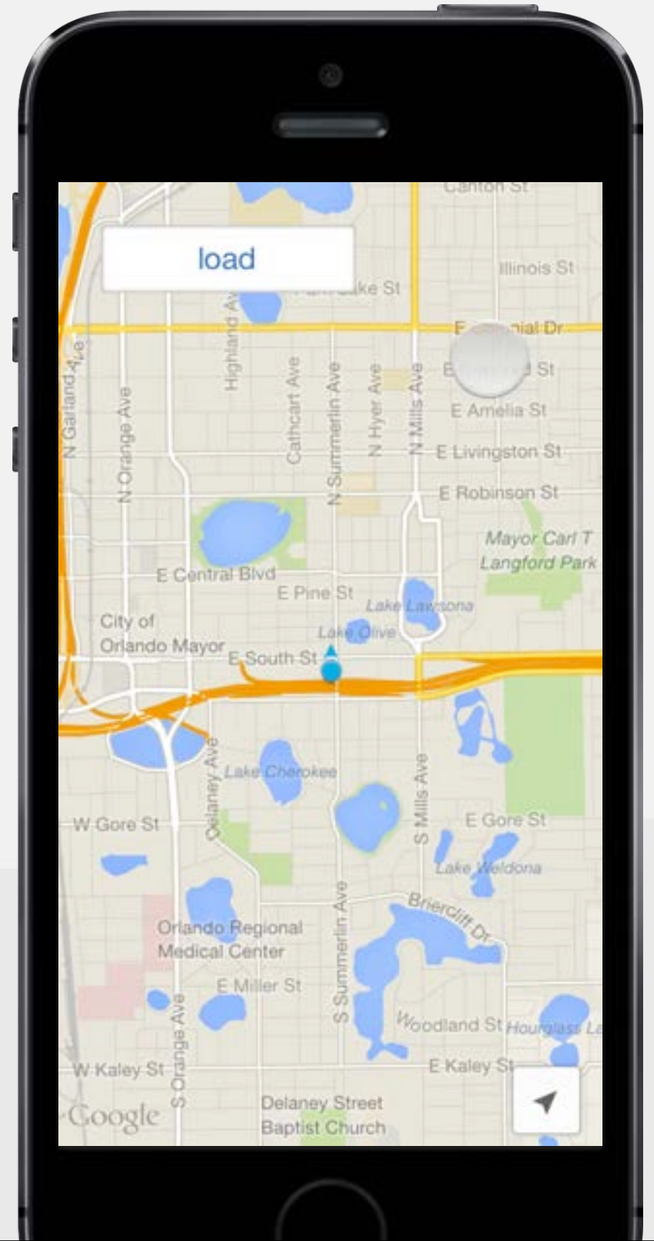
Creating marker objects with the response data

2b. add the new marker object to the mutable set

LakeMapVC.m

```
- (void)createMarkerObjectsWithJson:(NSArray *)json {
  NSMutableSet *mutableSet = [[NSMutableSet alloc] initWithSet:self.markers];
  for (NSDictionary *markerData in json) {
    GMSMarker *newMarker = [[GMSMarker alloc] init];
    newMarker.appearAnimation = [markerData[@"appearAnimation"] integerValue];
    newMarker.position = CLLocationCoordinate2DMake([markerData[@"lat"] doubleValue],
                                                    [markerData[@"lng"] doubleValue]);
    newMarker.title = markerData[@"title"];
    newMarker.snippet = markerData[@"snippet"];
    newMarker.map = nil;
    [mutableSet addObject:newMarker];
  }
  self.markers = [mutableSet copy];
  [self drawMarkers];
}
```

Problem: We're getting a thread exception error




```
2014-02-12 12:59:39.724 cs-maps-level3[15818:70b] Cannot find executable for
CFBundle 0xae566f0 </Applications/Xcode.app/Contents/Developer/Platforms/
iPhoneSimulator.platform/Developer/SDKs/iPhoneSimulator7.0.sdk/System/Library/
AccessibilityBundles/CertUIFramework.axbundle> (not loaded)
2014-02-12 12:59:39.735 cs-maps-level3[15818:70b] Cannot find executable for
CFBundle 0x9f93640 </Applications/Xcode.app/Contents/Developer/Platforms/
iPhoneSimulator.platform/Developer/SDKs/iPhoneSimulator7.0.sdk/System/Library/
AccessibilityBundles/GeoServices.axbundle> (not loaded)
2014-02-12 12:59:41.007 cs-maps-level3[15818:70b] Google Maps SDK for iOS
version: 1.7.0.7198
```

```
*** Terminating app due to uncaught exception
'GMSThreadException', reason: 'All calls to the
Google Maps SDK for iOS must be made from the UI
thread'
```

Solution: Force the method to run on the main thread

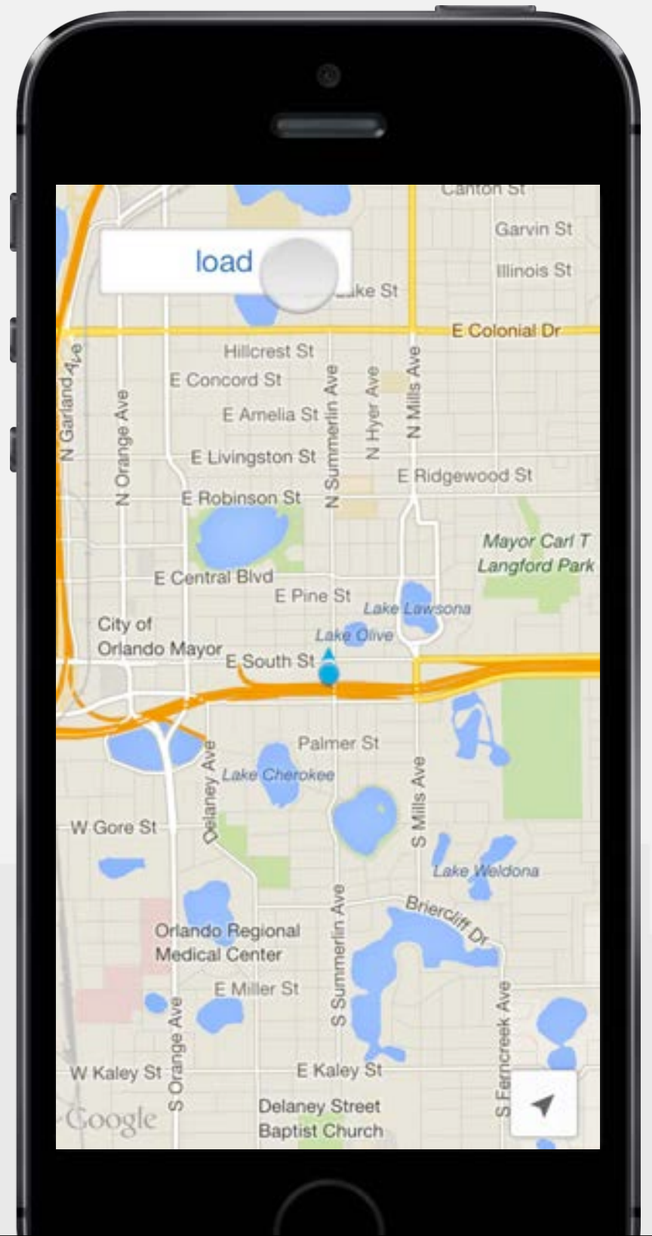
LakeMapVC.m

```
- (void)downloadMarkerData:(id)sender {

  NSURL *lakesURL =
      [NSURL URLWithString:@"http://googlemaps.codeschool.com/lakes"];

  NSURLSessionDataTask *task = [self.markerSession dataTaskWithURL:lakesURL
      completionHandler:^(NSData *data, NSURLResponse *response, NSError *e) {
    [[NSOperationQueue mainQueue] addOperationWithBlock:^{
      [self createMarkerObjectsWithJson:json];
    }];
  }];

  [task resume];
}
```

this method might not run on the main thread

any calls to the Google Maps SDK MUST happen on the main thread

Demo: Load button not crashing… but there's still a problem



markers are created, but they are **duplicated** each time load is tapped

```
2014-02-12 13:12:24.337 cs-maps-level3[16099:70b] Cannot find executable for
CFBundle 0xa171820 </Applications/Xcode.app/Contents/Developer/Platforms/
iPhoneSimulator.platform/Developer/SDKs/iPhoneSimulator7.0.sdk/System/Library/
AccessibilityBundles/CertUIFramework.axbundle> (not loaded)
2014-02-12 13:12:24.348 cs-maps-level3[16099:70b] Cannot find executable for
CFBundle 0x9db3cc0 </Applications/Xcode.app/Contents/Developer/Platforms/
iPhoneSimulator.platform/Developer/SDKs/iPhoneSimulator7.0.sdk/System/Library/
AccessibilityBundles/GeoServices.axbundle> (not loaded)
2014-02-12 13:12:25.686 cs-maps-level3[16099:70b] Google Maps SDK for iOS
version: 1.7.0.7198
```

Why is there duplicate data in our NSSet?

NSSet documentation

> `NSSet` declares the programmatic interface for static sets of distinct objects.

Sets resist duplicate data, but they need to be told what makes an object a duplicate

distinct == unique so why does our set contain non-unique data?

Exploring Google Maps for iOS

Why is there duplicate data in our NSSet?

is `GMSMarker A` equal to `GMSMarker B` ???

if you don't define what it means to be equal, a hash of the object will be compared

is `167516785421` equal to `153139487986` 

we need to tell NSSet what to compare when it is checking if two marker's are equal

is `GMSMarker A's ???` equal to `GMSMarker B's ???`

Designing a GMSMarker subclass

We need to:

1. Add an objectID property to each marker
2. Override the isEqual: and hash methods to use that objectID

We can't edit GMSMarker, but we can create a subclass

```
@interface CSMarker : GMSMarker
```

`CSMarker` inherits all of the properties and methods from `GMSMarker`

Setting up the GMSMarker subclass

CSMarker.h

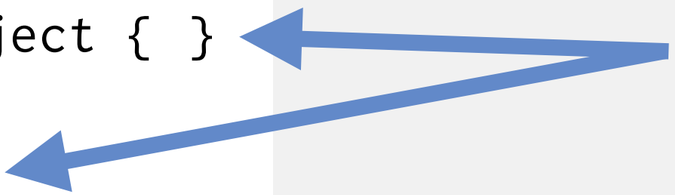
```
#import <Foundation/Foundation.h>
#import <GoogleMaps/GoogleMaps.h>

@interface CSMarker : GMSMarker

@property (nonatomic, copy) NSString *objectID;

@end
```


now our markers
have this property

Setting up the GMSMarker subclass

CSMarker.m

```
#import "CSMarker.h"

@implementation CSMarker

- (BOOL)isEqual:(id)object { }

- (NSUInteger)hash { }

@end
```



we need to override these two methods

Override the isEqual: and hash methods to use objectID

CSMarker.m

```
#import "CSMarker.h"

@implementation CSMarker

- (BOOL)isEqual:(id)object {
    CSMarker *otherMarker = (CSMarker *)object;
    if (self.objectID == otherMarker.objectID) {
        return YES;
    }
    return NO;
}

- (NSUInteger)hash {
    return [self.objectID hash];
}
@end
```

return **YES** if the marker IDs are equal

return **NO** for any other scenario

isEqual: documentation

If two objects are equal, they must have the same hash value. This last point is particularly important if you define `isEqual:` in a subclass and intend to put instances of that subclass into a collection. Make sure you also define `hash` in your subclass.

Create CSMarkers now instead of GMSMarkers

LakeMapVC.m

```
- (void)createMarkerObjectsWithJson:(NSArray *)json {
  NSMutableSet *mutableSet = [[NSMutableSet alloc] initWithSet:self.markers];
  for (NSDictionary *markerData in json) {
    CSMarker *newMarker = [[CSMarker alloc] init];
    newMarker.objectID = [markerData[@"id"] stringValue];
    newMarker.appearAnimation = [markerData[@"appearAnimation"] integerValue];
    newMarker.position = CLLocationCoordinate2DMake([markerData[@"lat"] doubleValue],
                                                    [markerData[@"lng"] doubleValue]);
    newMarker.title = markerData[@"title"];
    newMarker.snippet = markerData[@"snippet"];
    newMarker.flat = [markerData[@"flat"] boolValue];
    newMarker.map = nil;

    [mutableSet addObject:newMarker];
  }
  self.markers = [mutableSet copy];
  [self drawMarkers];
}
```

use `CSMarker` instead of `GMSMarker`

the JSON response data contains ids, so store them in the marker's `objectID` property

Update the drawing method to use CSMarker

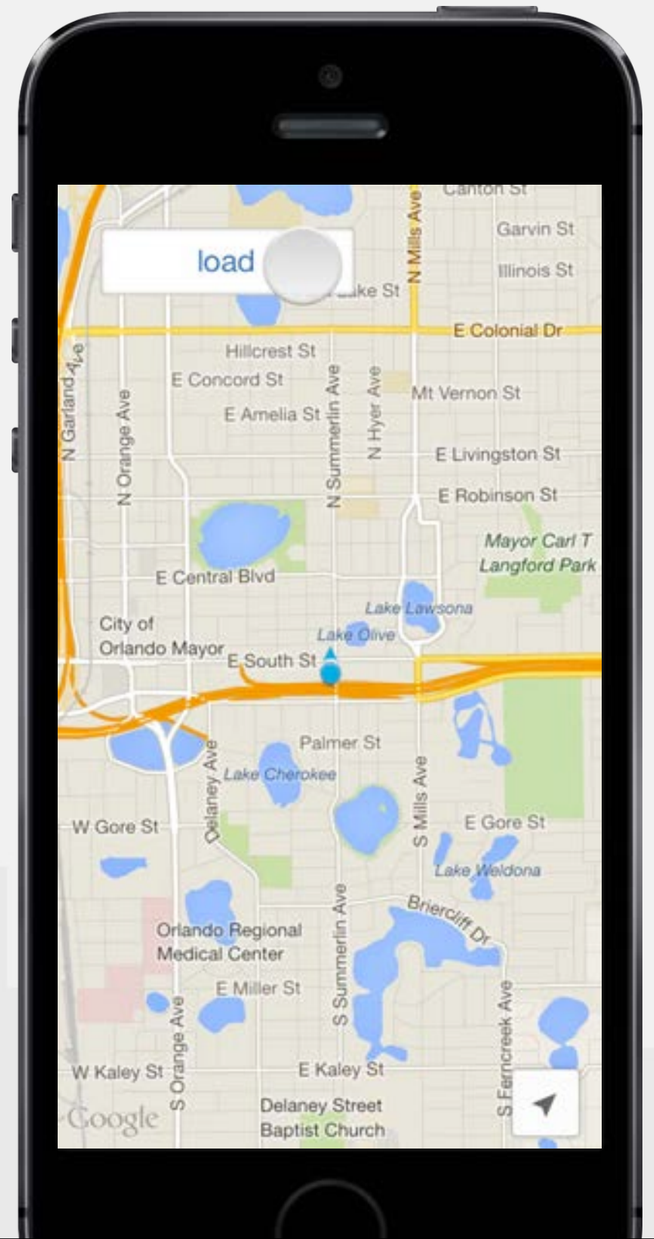
use `CSMarker` instead of `GMSMarker`

LakeMapVC.m

```
- (void)drawMarkers {
  for (CSMarker *marker in self.markers) {
    if (marker.map == nil) {
      marker.map = self.mapView;
    }
  }
}
```

since `CSMarker` is a subclass, it also contains all of the properties that `GMSMarker` (the super class) does

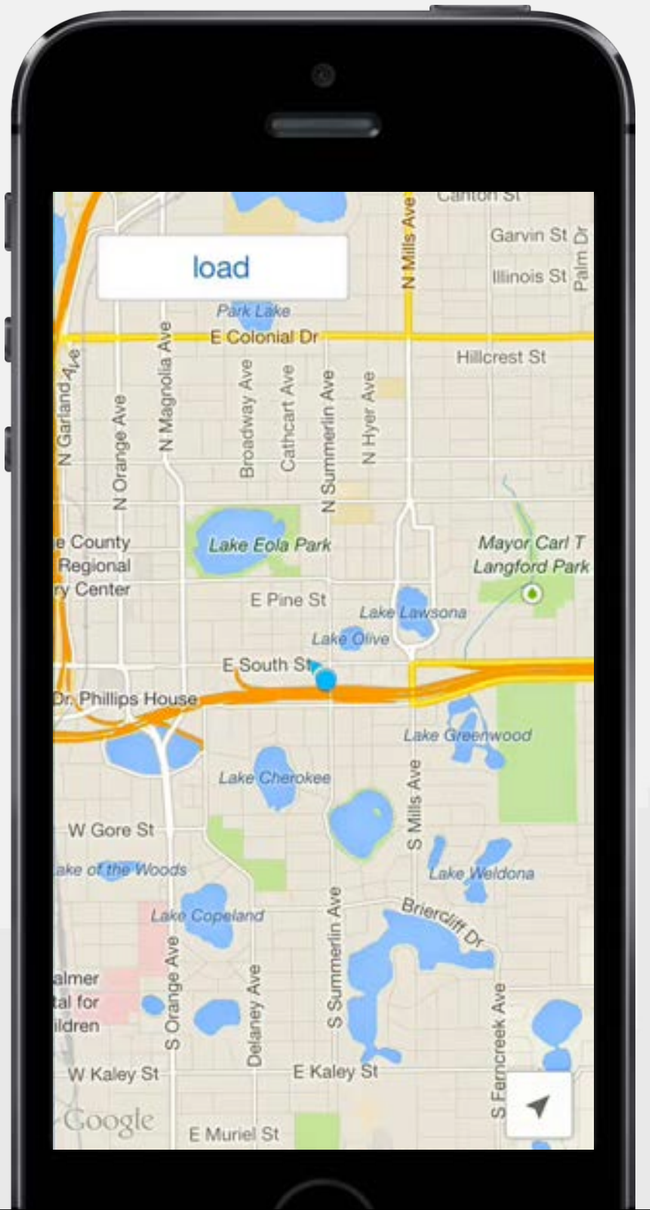
Demo: Load button working with no duplicate data




```
2014-02-12 13:18:21.078 cs-maps-level3[16305:70b] Cannot find executable for
CFBundle 0xf720c40 </Applications/Xcode.app/Contents/Developer/Platforms/
iPhoneSimulator.platform/Developer/SDKs/iPhoneSimulator7.0.sdk/System/Library/
AccessibilityBundles/GeoServices.axbundle> (not loaded)
2014-02-12 13:18:21.087 cs-maps-level3[16305:70b] Cannot find executable for
CFBundle 0x9d48850 </Applications/Xcode.app/Contents/Developer/Platforms/
iPhoneSimulator.platform/Developer/SDKs/iPhoneSimulator7.0.sdk/System/Library/
AccessibilityBundles/CertUIFramework.axbundle> (not loaded)
2014-02-12 13:18:22.436 cs-maps-level3[16305:70b] Google Maps SDK for iOS
version: 1.7.0.7198
```

Exploring Google Maps for iOS

Demo: Adding a user-created marker



Exploring Google Maps for iOS

Defining a property to hold the user-created marker

Comparing set markers vs. the user-created marker

markers in the set	user-created marker
Built from a network request	Built from a user tap
Displayed permanently once they are loaded	Displayed temporarily
Will be more than one shown at once	Only one will ever be shown at once

LakeMapVC.m

```
@interface LakeMapVC () <GMSMapViewDelegate>

...
@property(strong, nonatomic) CSMarker *userCreatedMarker;

@end
```

We'll want this to be separate from the main marker set because it has different rules

Creating and storing the user-created marker when a long-press happens

LakeMapVC.m

delegate method that runs when a long press event is received

```
- (void)mapView:(GMSMapView *)mapView
    didLongPressAtCoordinate:(CLLocationCoordinate2D)coordinate {

  CSMarker *marker = [[CSMarker alloc] init];
  marker.position = coordinate;
  marker.appearAnimation = kGMSMarkerAnimationPop;
  marker.title = @"created by user";
  marker.map = nil;

  self.userCreatedMarker = marker;

  [self drawMarkers];
}
```

use the **coordinate** passed into the delegate method (tap location)

assign this new **marker** to the **userCreatedMarker** property

Drawing the user-created marker on the map

LakeMapVC.m

```
- (void)drawMarkers {
  for (CSMarker *marker in self.markers) {
    if (marker.map == nil) {
      marker.map = self.mapView;
    }
  }
  if (self.userCreatedMarker != nil && self.userCreatedMarker.map == nil) {
    self.userCreatedMarker.map = self.mapView;
    self.mapView.selectedMarker = self.userCreatedMarker;
  }
}
```

if the `userCreatedMarker` property is not nil, but the `userCreatedMarker.map` property is nil, then turn the marker on

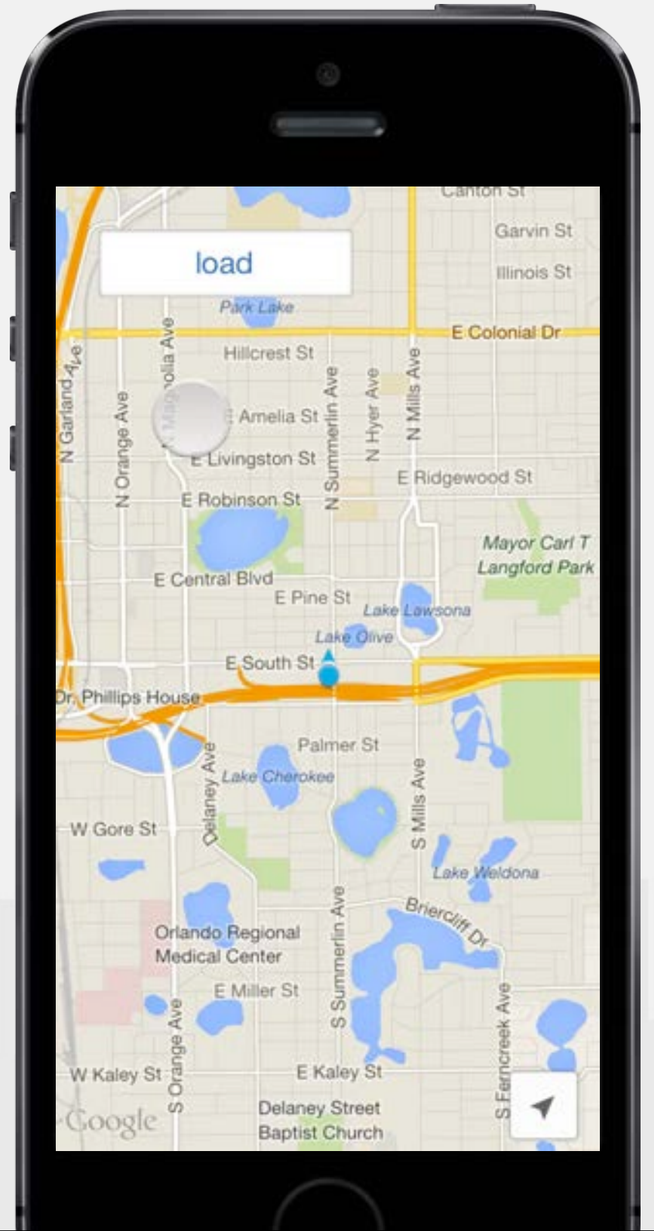
setting the `mapView.selectedMarker` property is a way to “tap” on the marker using code

Demo: Adding a user-created marker (so far)

Problems

Old user-created markers aren't removed when a new one is created

The map doesn't re-center on each newly created marker



Exploring Google Maps for iOS

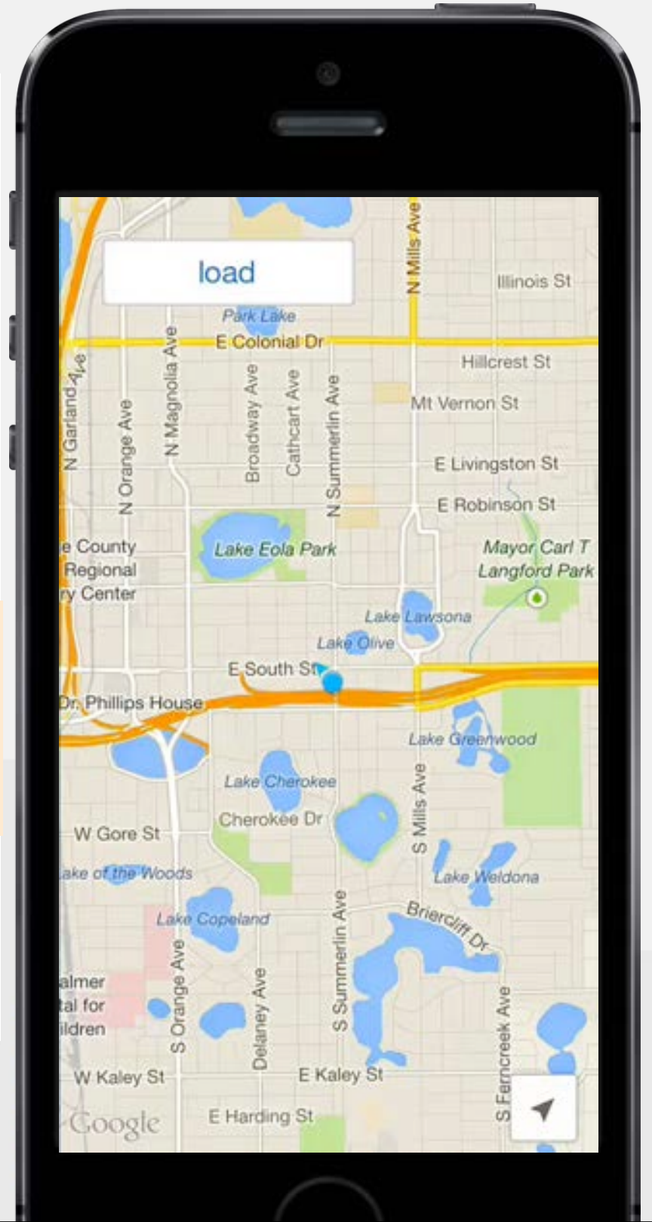
Update the camera position when the user-created marker is drawn

LakeMapVC.m

```
- (void)drawMarkers {
  ...
  if (self.userCreatedMarker != nil &&
      self.userCreatedMarker.map == nil) {
    self.userCreatedMarker.map = self.mapView;
    self.mapView.selectedMarker = self.userCreatedMarker;
    GMSCameraUpdate *cameraUpdate = [GMSCameraUpdate
        setTarget:self.userCreatedMarker.position];
    [self.mapView animateWithCameraUpdate:cameraUpdate];
  }
}
```

center the map on the new user-created marker

use `GMSCameraUpdate` to update the camera if you've already created one (with `GMSCameraPosition`)

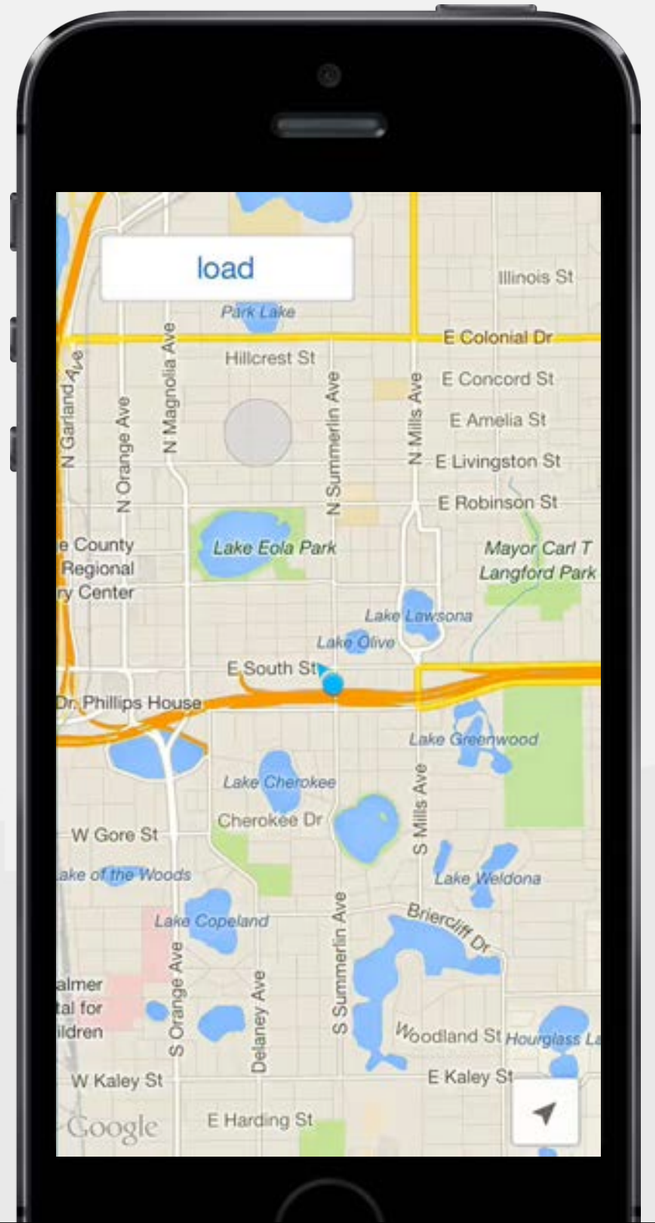


Show only one user-created marker at a time

LakeMapVC.m

```
- (void)mapView:(GMSMapView *)mapView
    didLongPressAtCoordinate:
        (CLLocationCoordinate2D)coordinate {

  if (self.userCreatedMarker != nil) {
    self.userCreatedMarker.map = nil;
    self.userCreatedMarker = nil;
  }

  // create the marker and call drawMarkers
}
```

if `userCreatedMarker` exists,
turn it off and reset the object



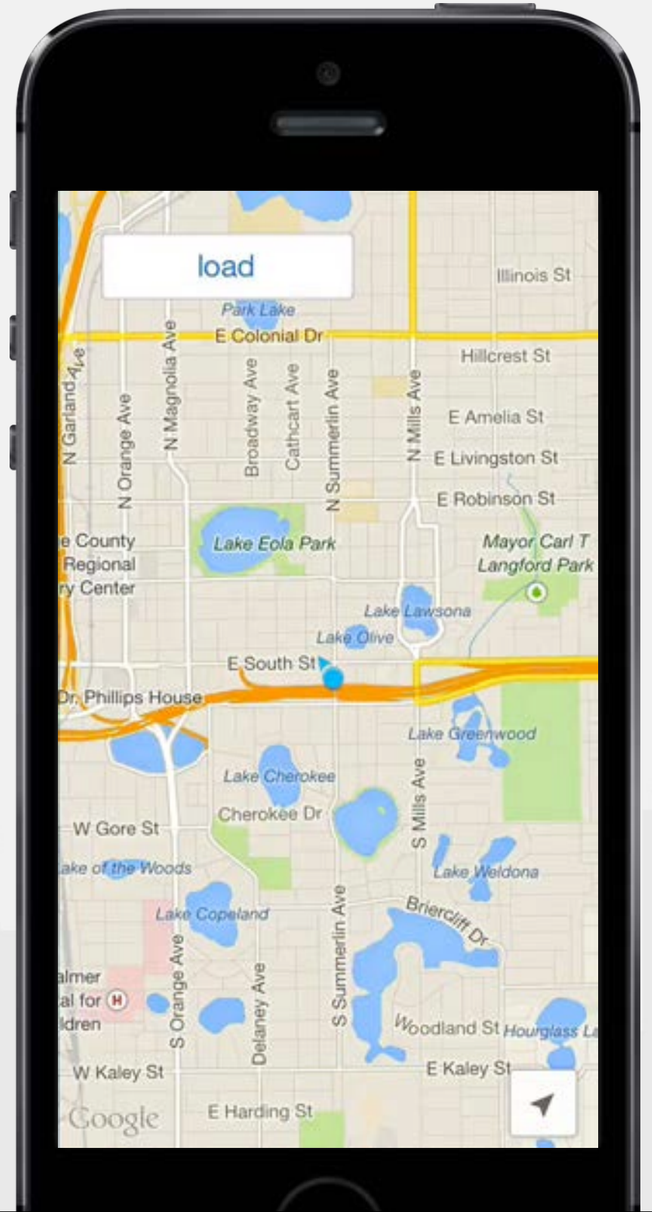
Exploring Google Maps for iOS

Using geocoding and finding directions

Level 4

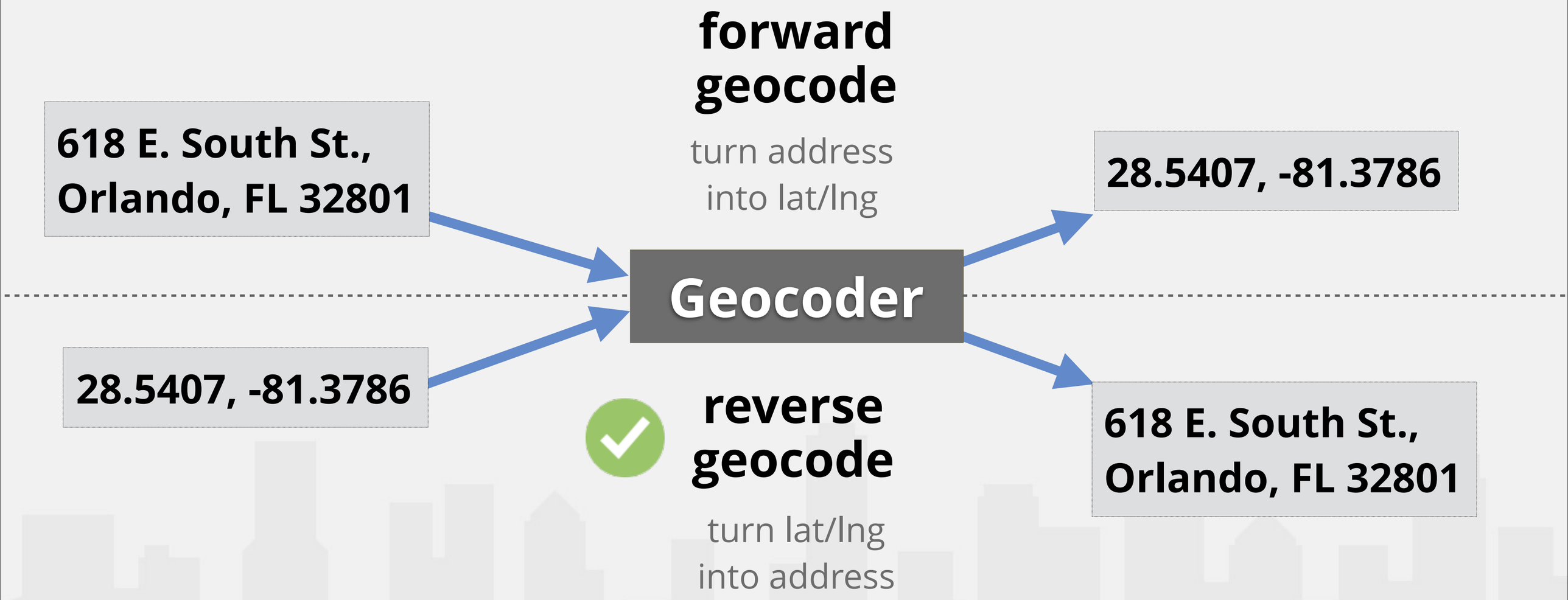
Exploring Google Maps for iOS

Demo: Geocoding the address for a user-created marker



Exploring Google Maps for iOS

What is a geocoder?



we have a coordinate that we want to turn into an address

Create a geocoder and pass it a coordinate

LakeMapVC.m

```
- (void)mapView:(GMSMapView *)mapView
    didLongPressAtCoordinate:(CLLocationCoordinate2D)coordinate {
  ...
  GMSGeocoder *geocoder = [GMSGeocoder geocoder];
  [geocoder reverseGeocodeCoordinate:coordinate
     completionHandler:^(GMSReverseGeocodeResponse *response, NSError *error) {
    // create the marker and use data in the geocode response
  }];
}
```

create the geocoder object

send the geocoder the coordinate where the user tapped

Exploring Google Maps for iOS

Use the geocoder response data for the marker's title and snippet

LakeMapVC.m

```
- (void)mapView:(GMSMapView *)mapView
    didLongPressAtCoordinate:(CLLocationCoordinate2D)coordinate {
  ...
  GMSGeocoder *geocoder = [GMSGeocoder geocoder];
  [geocoder reverseGeocodeCoordinate:coordinate
      completionHandler:^(GMSReverseGeocodeResponse *response, NSError *error) {
      CSMarker *marker = [[CSMarker alloc] init];
      marker.position = coordinate;
      marker.appearAnimation = kGMSMarkerAnimationPop;
      marker.map = nil;
      marker.title = response.firstResult.thoroughfare;
      marker.snippet = response.firstResult.locality;
      self.userCreatedMarker = marker;

      [self drawMarkers];
  }];
}
```

create the marker in the geocoder completion handler

this will get the street address and city name

Available data types in the geocoder response

`administrativeArea`	state / region / administrative area
`country`	country
`locality`	city or locality
`subLocality`	subdivision, district, or park
`thoroughfare`	street number and name

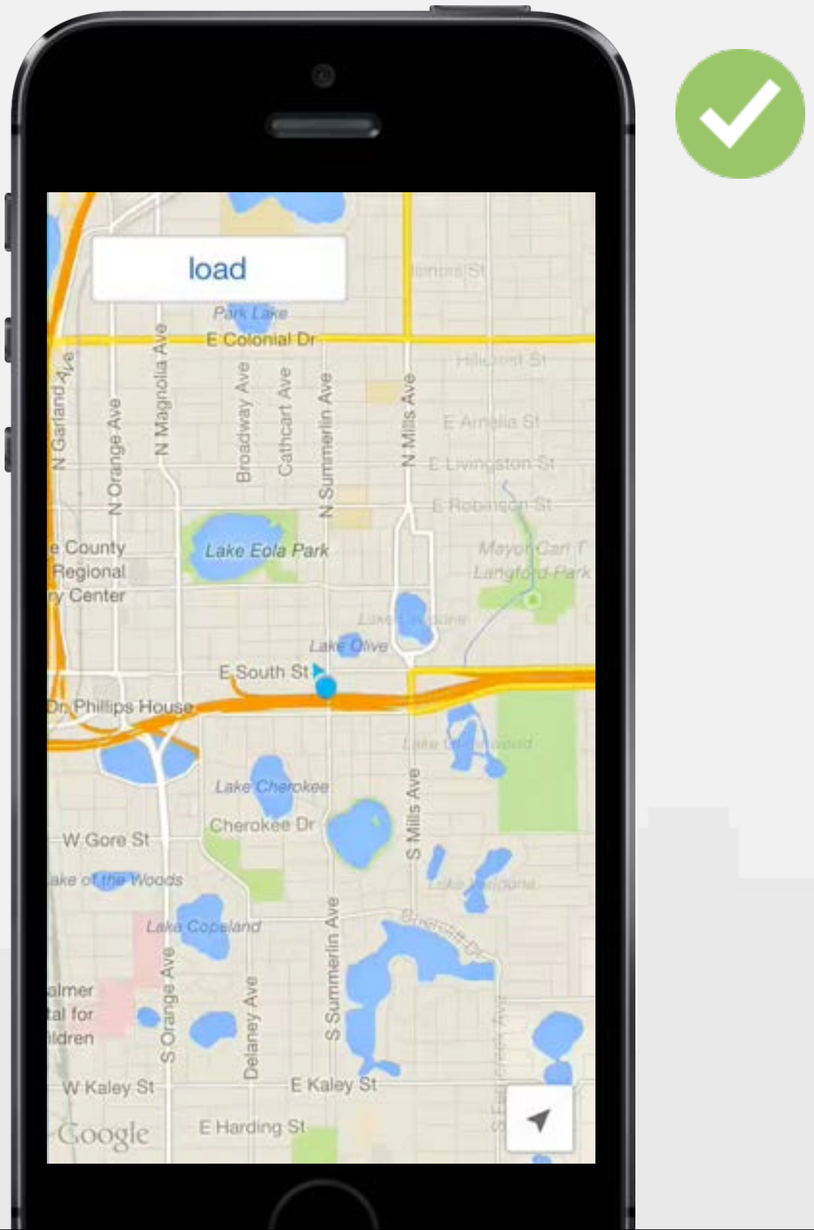
```
marker.title = response.firstResult.thoroughfare;
```

618 E. South Street

```
marker.snippet = response.firstResult.locality;
```

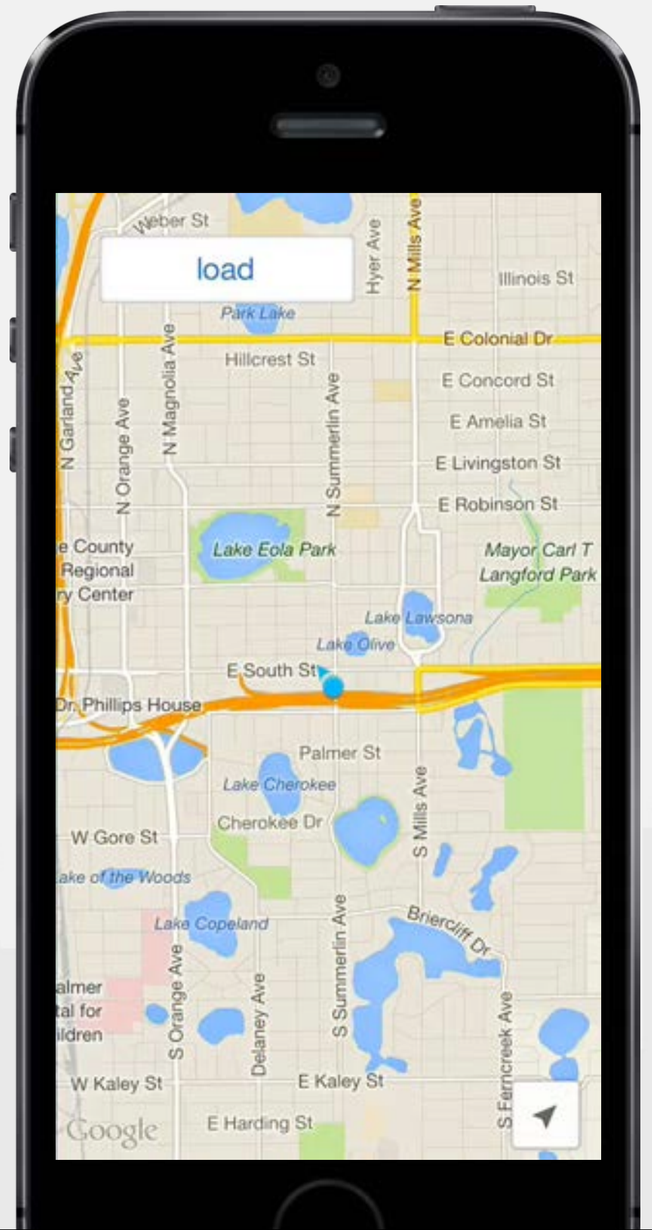
Orlando

Demo: Geocoder working



Exploring Google Maps for iOS

Demo: Getting directions to a location when a marker is tapped



Exploring Google Maps for iOS

A plan of attack for using the Directions API

LakeMapVC.m

another delegate method

```
- (BOOL)mapView:(GMSMapView *)mapView didTapMarker:(GMSMarker *)marker {
  if (mapView.myLocation != nil) {
```

don't even try getting directions if user location tracking is off!

1. Create an NSURL in the format that the Directions API wants using the user location and the tapped marker's location
2. Make a network request and get a JSON response back
3. Do something with the JSON response

```
  }
}
```

Required parameters for the Directions API

Base URL

```
https://maps.googleapis.com/maps/api/directions/json?
```

Required parameters

`origin=`	`latitude,longitude`	the start location
`destination=`	`latitude,longitude`	the end location
`sensor=`	`true`	MUST be true if using on a device with GPS (like the iPhone)
`key=`	`your “browser apps” API key`	

Making a Directions API request

1. Create an NSURL in the format that the Directions API wants using the user location and the tapped marker's location

LakeMapVC.m

use `NSString` format placeholders to build the string up

```
- (BOOL)mapView:(GMSMapView *)mapView didTapMarker:(GMSMarker *)marker {
  if (mapView.myLocation != nil) {

    NSString *urlString = [NSString stringWithFormat:
                           @"%@?origin=%f,%f&destination=%f,%f&sensor=true&key=%@",

the base URL               @"https://maps.googleapis.com/maps/api/directions/json",

the start location         mapView.myLocation.coordinate.latitude,
                           mapView.myLocation.coordinate.longitude,

the end location           marker.position.latitude,
                           marker.position.longitude,

the browser app key        @"AE26762kdznyk22lsncnuk42lslcn"];
  }
}
```

each of the 6 placeholders gets a value

Making a Directions API request

2. Make a network request and get a JSON response back

LakeMapVC.m

```
- (BOOL)mapView:(GMSMapView *)mapView didTapMarker:(GMSMarker *)marker {
  if (mapView.myLocation != nil) {
    NSString *urlString = [NSString stringWithFormat:
                           @"%@?origin=%f,%f&destination=%f,%f&sensor=true&key=%@",
                           @"https://maps.googleapis.com/maps/api/directions/json",
                           mapView.myLocation.coordinate.latitude,
                           mapView.myLocation.coordinate.longitude,
                           marker.position.latitude,
                           marker.position.longitude,
                           @"AE26762kdznyk22lsncnuk42lslcn"];
    NSURL *directionsURL = [NSURL URLWithString:urlString];
    NSURLSessionDataTask *directionsTask = [self.markerSession dataTaskWithURL:directionsURL
        completionHandler:^(NSData *data, NSURLResponse *response, NSError *e) {
    }];
  }
}
```

Making a Directions API request

2. Make a network request and get a JSON response back

LakeMapVC.m

```
- (BOOL)mapView:(GMSMapView *)mapView didTapMarker:(GMSMarker *)marker {
  if (mapView.myLocation != nil) {

    ...
    NSURL *directionsURL = [NSURL URLWithString:urlString];
    NSURLSessionDataTask *directionsTask = [self.markerSession dataTaskWithURL:directionsURL
        completionHandler:^(NSData *data, NSURLResponse *response, NSError *e) {
      NSError *error = nil;
      NSDictionary *json =
          [NSJSONSerialization JSONObjectWithData:data
                                          options:NSJSONReadingMutableContainers
                                            error:&error];

    }];
  }
}
```

this response is a dictionary, not an array

Examining the JSON response from the Directions API

structure of JSON response dictionary

```
json: {
    routes = (
        {
            bounds = { },
            copyrights = "",
            legs = (
                {
                    steps = (
                        { },
                        { },
                        { }
                    )
                }
            ),
            overview_polyline = { },
            ...
        }
    ),
    status = "OK"
}
```

```
json
[@"routes"]
[0]

[@"legs"]
[0]
[@"steps"]
```

this array of steps is what we really care about

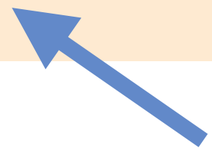
this code will get us that array of steps

```
json[@"routes"][0][@"legs"][0][@"steps"];
```

Storing the route steps in a new NSArray property

LakeMapVC.m

```
- (BOOL)mapView:(GMSMapView *)mapView didTapMarker:(GMSMarker *)marker {
  if (mapView.myLocation != nil) {
    ...
    NSURLSessionDataTask *directionsTask = [self.markerSession dataTaskWithURL:directionsURL
        completionHandler:^(NSData *data, NSURLResponse *response, NSError *e) {
      NSError *error = nil;
      NSDictionary *json =
          [NSJSONSerialization JSONObjectWithData:data
                                          options:NSJSONReadingMutableContainers
                                            error:&error];
      if (!error) {
        self.steps = json[@"routes"][0][@"legs"][0][@"steps"];
      }
    }];
  }
}
```



add a steps NSArray property to this classes' interface too

Examining the structure of the new steps array

the contents of `self.steps` after setting it to the array of steps in the JSON response

```
(
    distance = { },
    duration = { },
    end_location = { },
    html_instructions = "",
    polyline = { },
    start_location = { },
    travel_mode = ""
)
```

Head south on S Summerlin Ave toward E Anderson St

Since the directions for each step contain HTML, the easiest thing to do is use a UIWebView in each UITableViewCell to display the directions

We'll give you that code in the challenges, but it's not a Google Maps-specific thing

Sending the steps array data to a new table view controller

We've added a directions button to self.view

called when the directions button is tapped

LakeMapVC.m

```
- (void)directionsTapped:(id)sender {
  DirectionsListVC *directionsListVC = [[DirectionsListVC alloc] init];
  directionsListVC.steps = self.steps;
  [self presentViewController:directionsListVC
                     animated:YES
                   completion:^{
                     self.steps = nil;
                     self.mapView.selectedMarker = nil;
                   }];
}
```

send the current steps array to the new VC

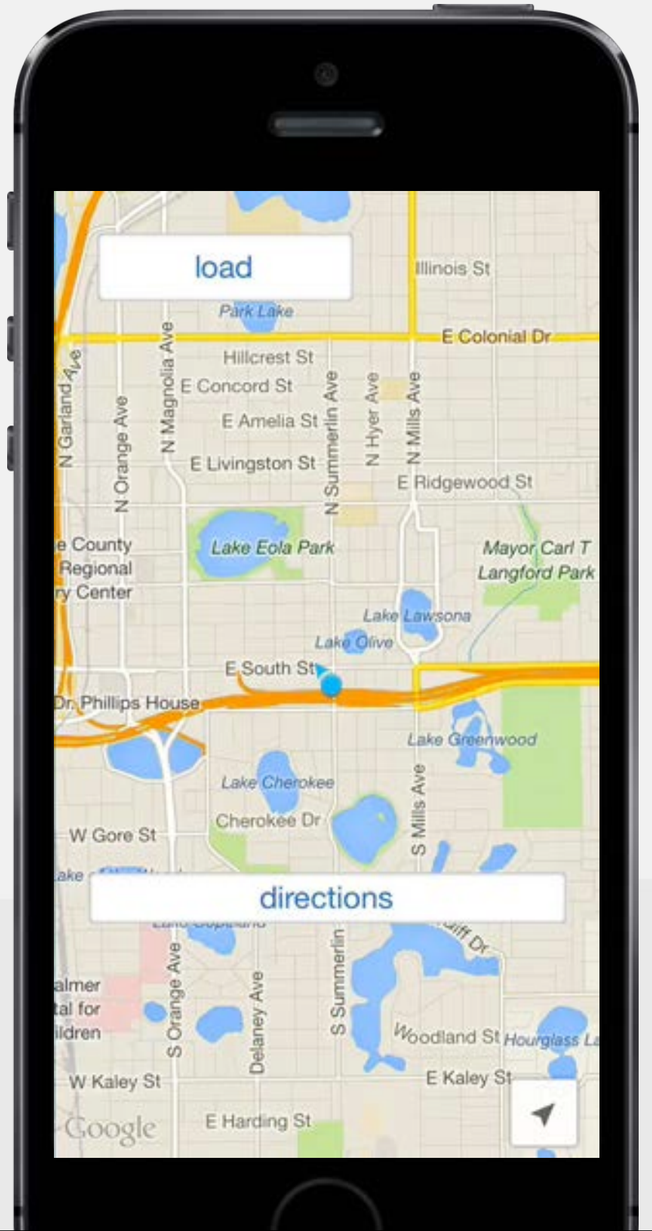
reset the steps array in this VC after a successful presentation

de-select the marker

check out Code School's Try iOS and Operation: Models courses if you're rusty on how table views work

Demo: directions are working, but the button is always displayed

The directions button should appear only when the marker is tapped, and disappear after the back button is tapped



Exploring Google Maps for iOS

Creating a directions button property to manage the display

LakeMapVC.m

```
@interface LakeMapVC () <GMSMapViewDelegate>
...
@property(strong, nonatomic) UIButton *directionsButton;
...
@end

@implementation LakeMapVC

- (void)viewDidLoad {
  [super viewDidLoad];

  ...
  self.directionsButton = [UIButton buttonWithType:UIButtonTypeSystem];
  self.directionsButton.alpha = 0.0;
  ...
}
```

make the button a property so you can modify it anywhere in the VC

make the initial button alpha 0.0 so it doesn't appear on screen

Show the button when the step array has data

LakeMapVC.m

```
- (BOOL)mapView:(GMSMapView *)mapView didTapMarker:(GMSMarker *)marker {
  if (mapView.myLocation != nil) {


      ...
      if (!error) {
        self.steps = json[@"routes"][0][@"legs"][0][@"steps"];
        [[NSOperationQueue mainQueue] addOperationWithBlock:^{
          self.directionsButton.alpha = 1.0;
        }];
      }
    }];
  }
}
```

show the button when `self.steps` has some data in it

Hide the button when the step array no longer has data

LakeMapVC.m

```
- (void)directionsTapped:(id)sender {
  DirectionsListVC *directionsListVC = [[DirectionsListVC alloc] init];
  directionsListVC.steps = self.steps;
  [self presentViewController:directionsListVC
                     animated:YES
                   completion:^{
                     self.steps = nil;
                     self.mapView.selectedMarker = nil;
                     self.directionsButton.alpha = 0.0;
                   }];
}
```

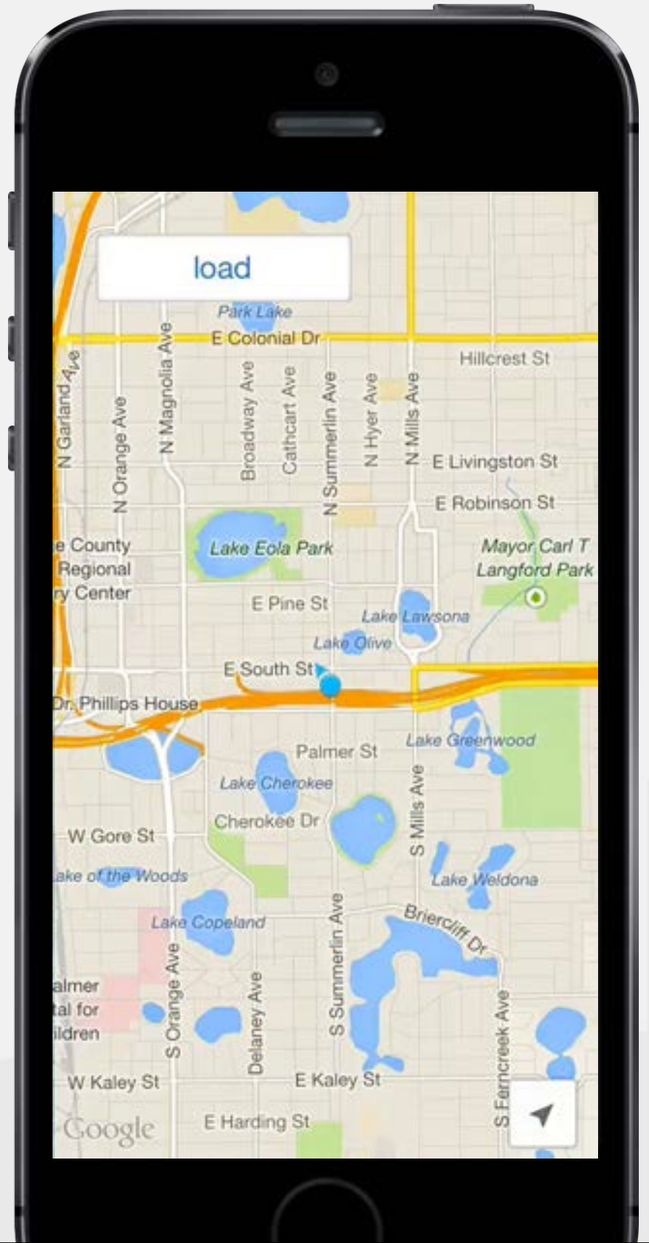
hide the button when `self.steps` no longer has data in it

Demo: Directions button show/hide on tap

The directions button should also be turned off when:

- the info window disappears

- the map moves



Exploring Google Maps for iOS

Hide the button when the map is tapped or moved

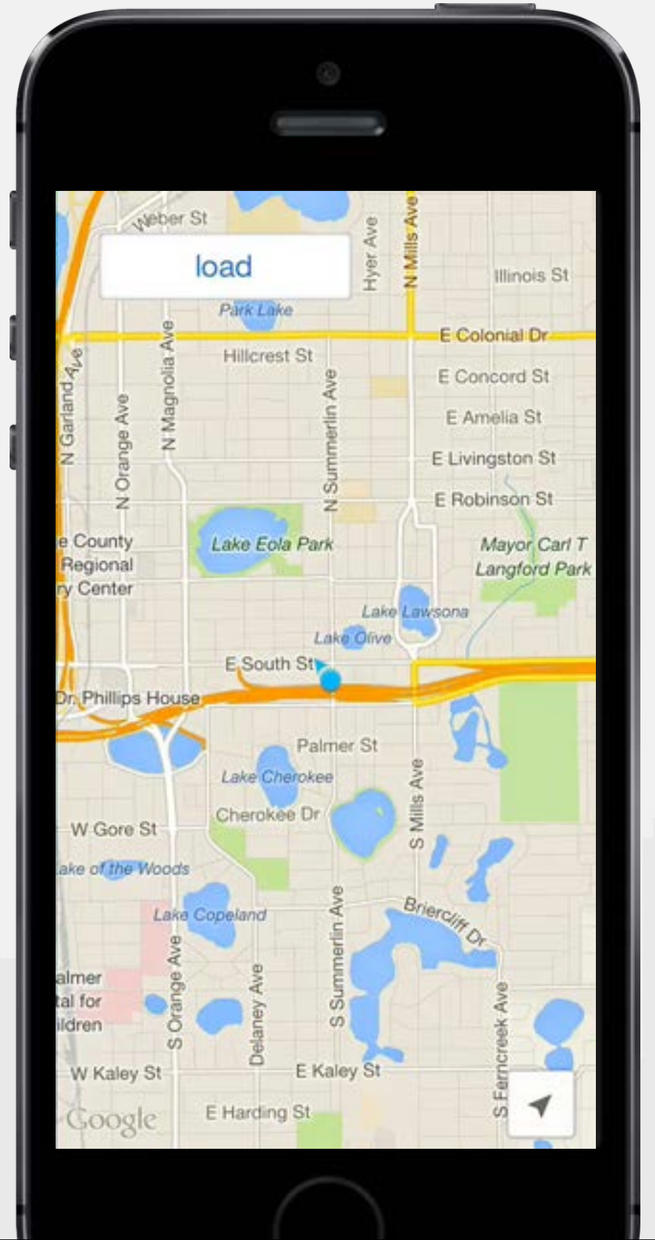
LakeMapVC.m

```
- (void)mapView:(GMSMapView *)mapView
    didTapAtCoordinate:(CLLocationCoordinate2D)coordinate {
  if (self.directionsButton.alpha > 0.0) {
    self.directionsButton.alpha = 0.0;
  }
}

- (void)mapView:(GMSMapView *)mapView willMove:(BOOL)gesture {
  if (self.directionsButton.alpha > 0.0) {
    self.directionsButton.alpha = 0.0;
  }
  self.mapView.selectedMarker = nil;
}
```

if the button is showing, hide it when the map is tapped

if the map is scrolled, hide the directions button if it is showing and de-select the marker

Demo: Everything working



Exploring Google Maps for iOS

Exploring Google Maps for iOS

Drawing lines and shapes on the map

Level 5

Exploring Google Maps for iOS

Drawing a line on the map

LakeMapVC.m

```
GMSMutablePath *singleLinePath = [[GMSMutablePath alloc] init];
[singleLinePath addLatitude:28.5382 longitude:-81.3687];
[singleLinePath addLatitude:28.5421 longitude:-81.3690];
```

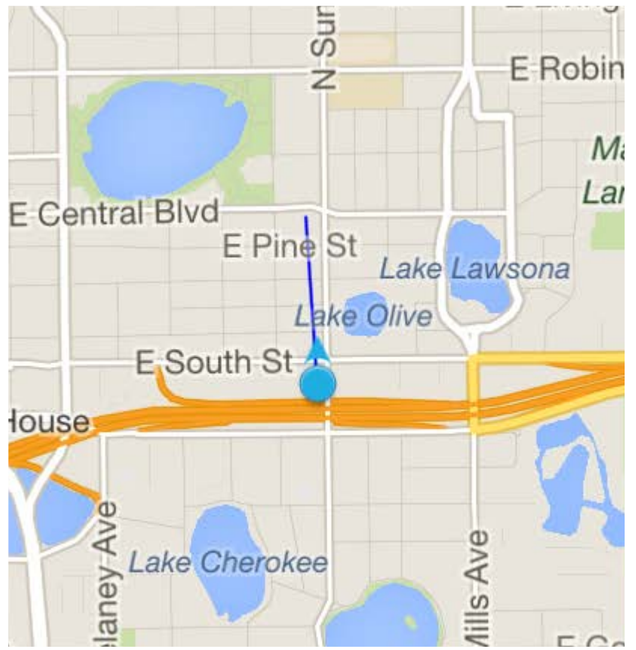
create a `GMSMutablePath`
and add two “points” as lat/lng

```
GMSPolyline *singleLine = [GMSPolyline
    polylineWithPath:singleLinePath];
```

use that path to create a `GMSPolyline`

```
singleLine.map = self.mapView;
```

turn the line **on** (same idea as turning a marker on)

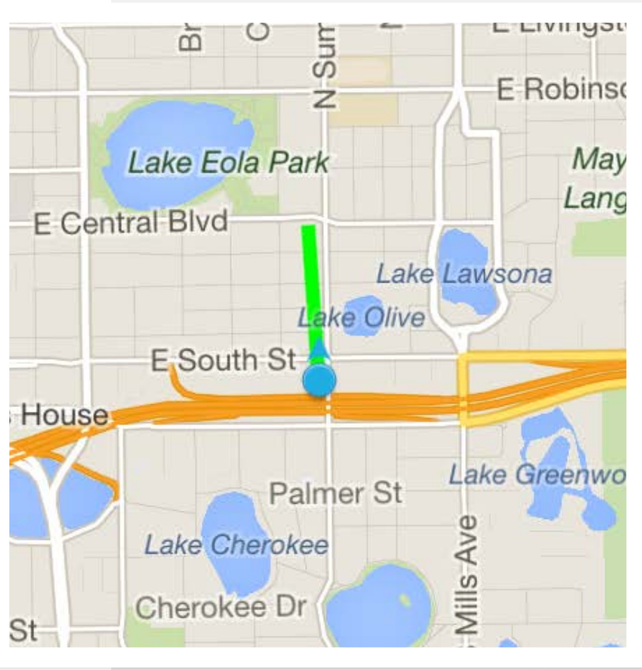


Changing the way the line looks

LakeMapVC.m

```
GMSMutablePath *singleLinePath = [[GMSMutablePath alloc] init];
[singleLinePath addLatitude:28.5382 longitude:-81.3687];
[singleLinePath addLatitude:28.5421 longitude:-81.3690];

GMSPolyline *singleLine = [GMSPolyline
    polylineWithPath:singleLinePath];
singleLine.strokeWidth = 5;
singleLine.strokeColor = [UIColor greenColor];
singleLine.map = self.mapView;
```

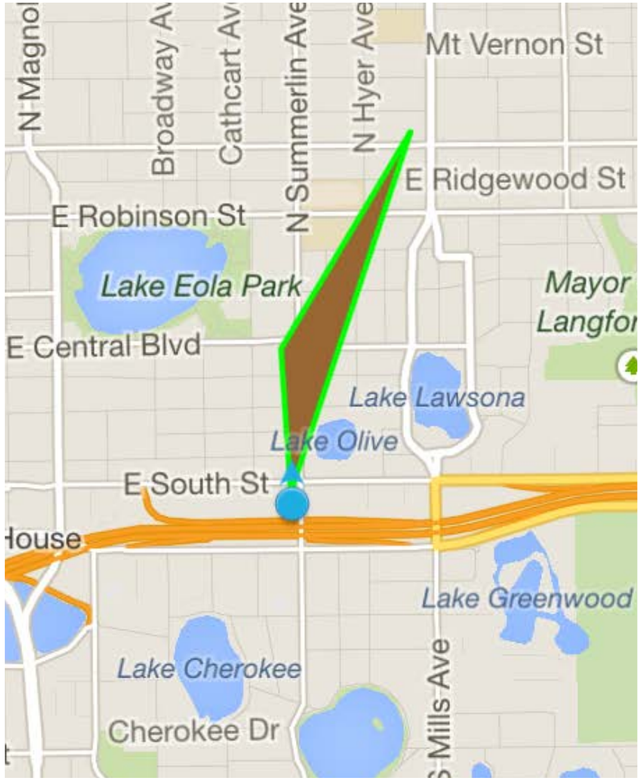
change the width and color of the line



Use a GMSPolygon to draw the shape

LakeMapVC.m

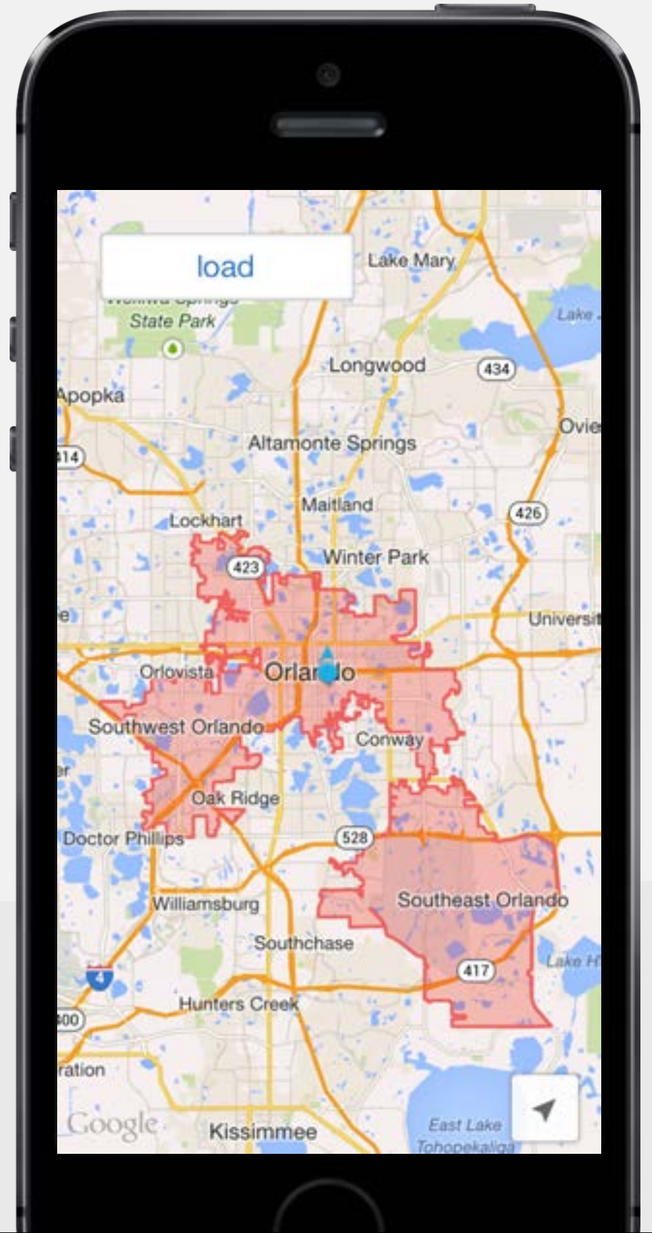
```
GMSMutablePath *shapePath = [[GMSMutablePath alloc] init];
[shapePath addLatitude:28.5382 longitude:-81.3687];
[shapePath addLatitude:28.5421 longitude:-81.3690];
[shapePath addLatitude:28.5480 longitude:-81.3650];

GMSPolygon *shape = [GMSPolygon polygonWithPath:shapePath];
shape.strokeWidth = 2;
shape.strokeColor = [UIColor greenColor];
shape.fillColor = [UIColor brownColor];
shape.map = self.mapView;
```



the `fillColor` will color in the area that the shape contains

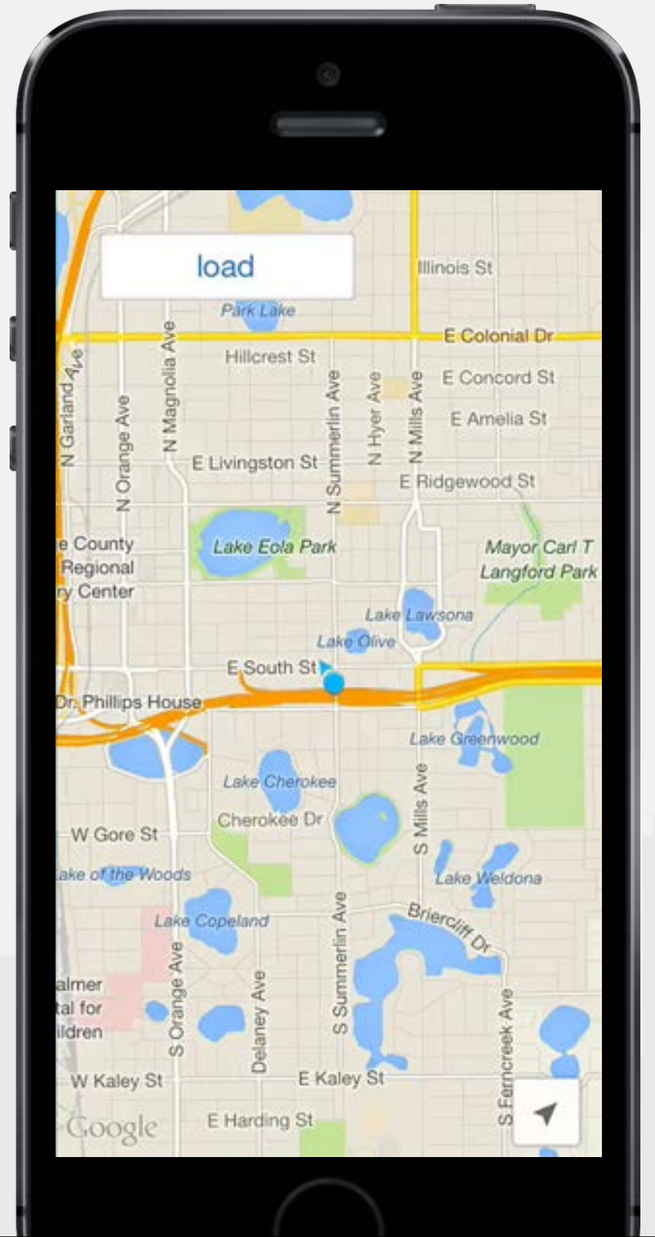
Example: Using polygons to display the city limit

The catch here is that you have to already have all of the point data



Exploring Google Maps for iOS

Using polylines to display directions



Exploring Google Maps for iOS

Using encoded polylines to display direction routes

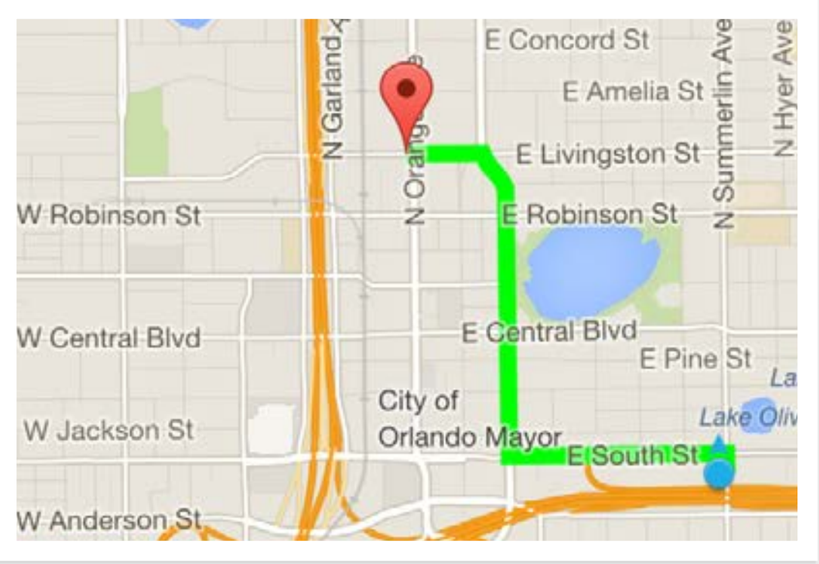
Encoded Polyline Algorithm Format

`https://developers.google.com/maps/documentation/utilities/polylinealgorithm`

this crazy string

```
_ydmDlgsoNsB@HjQB~T?`Ec@@yA@gMDaNFoN@o@Dw@^aA`Aa@\UNc@LBxC?jJ
```

easily becomes this awesome line



Create a GMSPolyline property to display the current route line

LakeMapVC.m

```
@interface LakeMapVC () <GMSMapViewDelegate>
...
@property(strong, nonatomic) GMSPolyline *polyline;

@end

@implementation LakeMapVC

- (void)viewDidLoad {
  [super viewDidLoad];

  GMSPath *encodedPath = @"_ydmDlgsoNsB@HjQB~T?`Ec@@yA@gMDaNFoN@o@Dw@^aA`Aa@\UNc@LBxC?jJ";
  self.polyline = [GMSPolyline polylineWithPath:encodedPath];
  self.polyline.strokeWidth = 7;
  self.polyline.strokeColor = [UIColor greenColor];
  self.polyline.map = self.mapView;
}
```

we need to find a way to get this encoded polyline string for any set of directions

Looking closer at the Directions API response

structure of JSON dictionary returned by Directions API request

```
json: {
    routes = (
        {
            bounds = { },
            copyrights = "",
            legs = (
                {
                    steps = (
                        { },
                        { },
                        { }
                    )
                }
            ),
            overview_polyline = { },
            ...
        }
    ),
    status = "OK"
}
```

this is an encoded polyline!

Exploring Google Maps for iOS

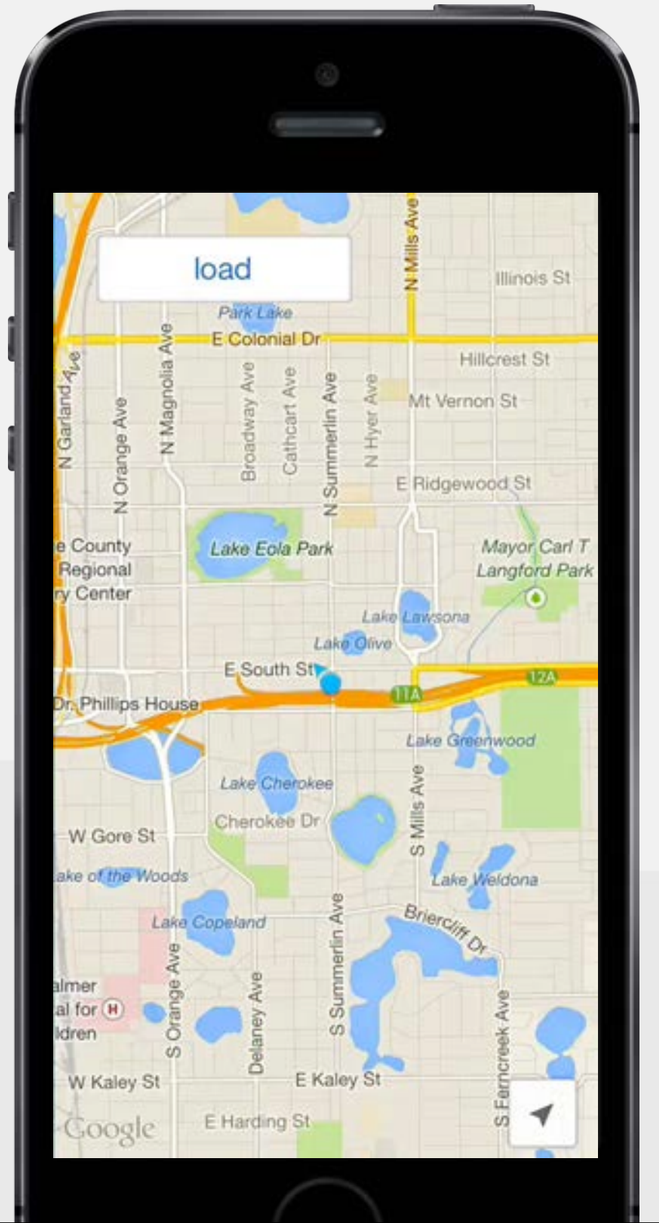
Drawing the overview polyline when the marker is tapped

LakeMapVC.m

```
- (BOOL)mapView:(GMSMapView *)mapView didTapMarker:(GMSMarker *)marker {
  if (mapView.myLocation != nil) {
    ...
    NSURLSessionDataTask *directionsTask = ... {
      NSError *error = nil;
      NSDictionary *json = ...
      if (!error) {
        self.steps = json[@"routes"][0][@"legs"][0][@"steps"];
        [[NSOperationQueue mainQueue] addOperationWithBlock:^{
          self.directionsButton.alpha = 1.0;
          GMSPath *path =
              [GMSPath pathFromEncodedPath:
                  json[@"routes"][0][@"overview_polyline"][@"points"]];
          self.polyline = [GMSPolyline polylineWithPath:path];
          self.polyline.strokeWidth = 7;
          self.polyline.strokeColor = [UIColor greenColor];
          self.polyline.map = self.mapView;
        }];
      }
    }];
  }
}
```

we're already using the array of steps for the DirectionsVC

we can also use the overview_polyline to draw a GMSPolyline on the map



Reset the polyline object before trying to draw a new one

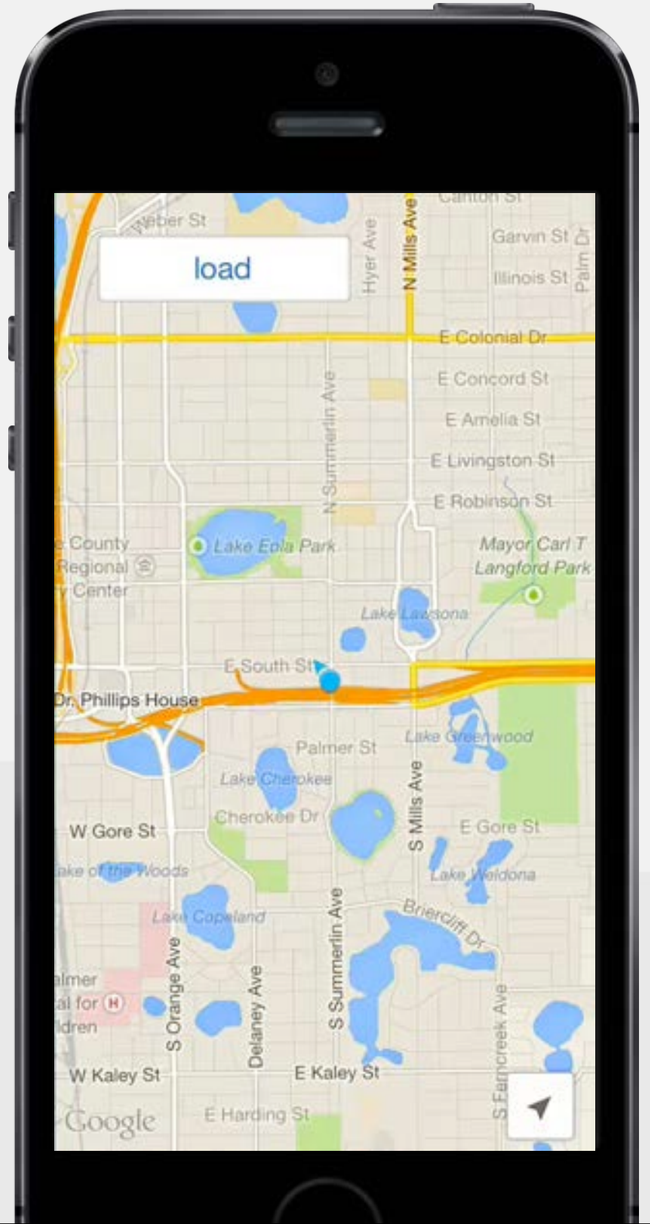
LakeMapVC.m

```
- (BOOL)mapView:(GMSMapView *)mapView didTapMarker:(GMSMarker *)marker {
  if (mapView.myLocation != nil) {
    ...

    self.polyline.map = nil;
    self.polyline = nil;

    NSURLSessionDataTask *directionsTask = ... {

    }];
  }
}
```

first turn off the polyline

then `nil` out the object

this is done to prevent two lines from being drawn at once



Turn off the polyline at the right time with delegate methods

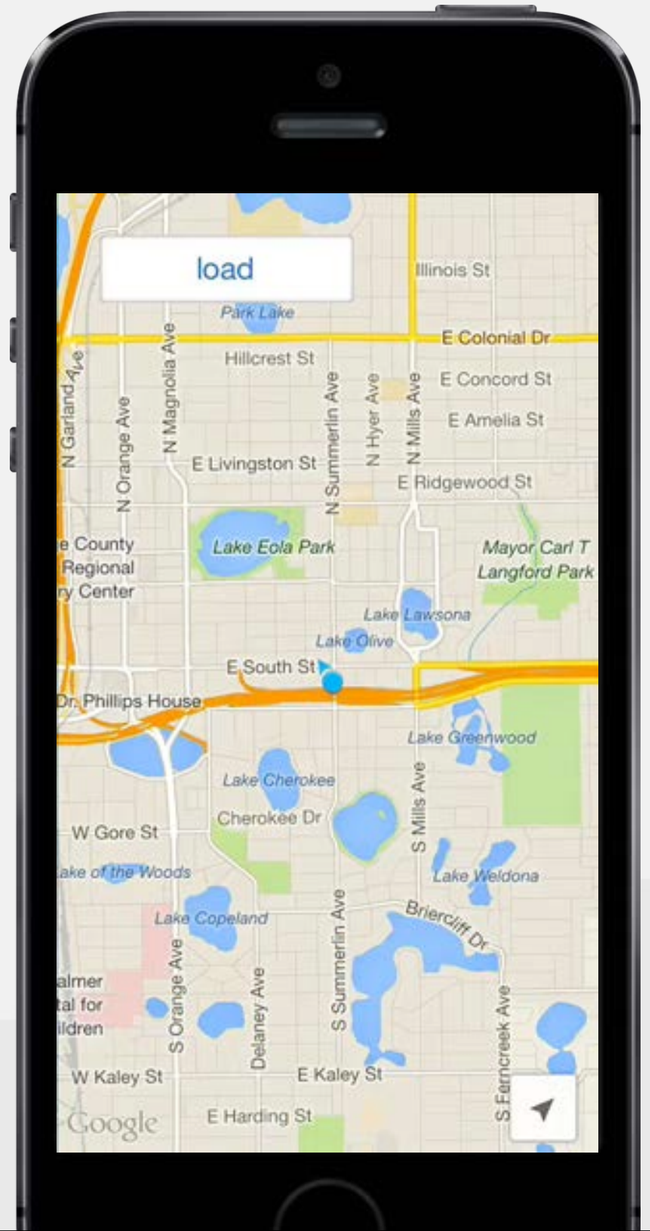
LakeMapVC.m

```
- (void)mapView:(GMSMapView *)mapView
    didLongPressAtCoordinate:(CLLocationCoordinate2D)coordinate {
  ...
  self.polyline.map = nil;
  self.polyline = nil;
}

- (void)mapView:(GMSMapView *)mapView
     didTapAtCoordinate:(CLLocationCoordinate2D)coordinate {
  ...
  self.polyline.map = nil;
  self.polyline = nil;
}

- (void)mapView:(GMSMapView *)mapView willMove:(BOOL)gesture {
  ...
  self.polyline.map = nil;
  self.polyline = nil;
}
```

turn off the polyline when there is a new user-created marker

and when the map is tapped at a non-marker location

and when the map is moved by the user dragging their finger



Exploring Google Maps for iOS

Google Street View

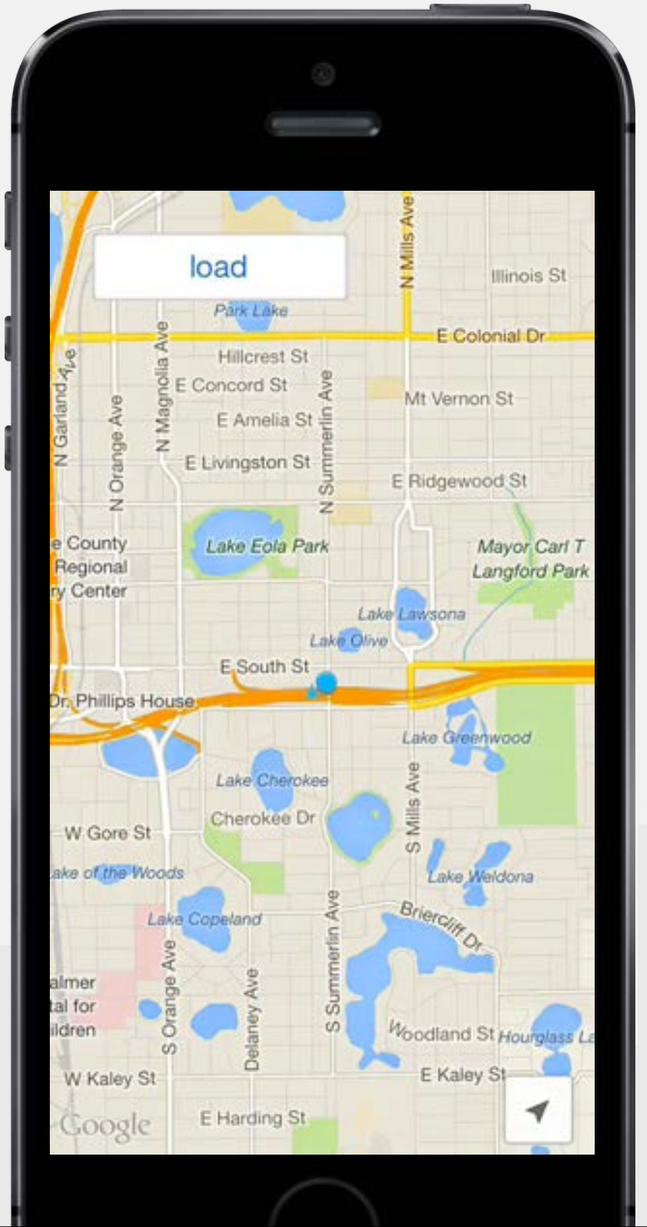
Level 6

Exploring Google Maps for iOS

Demo: Adding the ability to show a street view

Street View is unique to Google Maps and the Google Maps SDK

The app we've built so far



Exploring Google Maps for iOS

Setting up the StreetViewVC vs. DirectionsVC

Directions View

Create and add button to load DirectionsVC

Street View

Create and add button to load StreetViewVC

Show the button when a marker is active

Hide the button when that marker is deactivated

Pass the directions steps to DirectionsVC

same steps, different data!

Pass the marker coordinate to StreetViewVC

Create a property to store the coordinate of the currently selected marker

LakeMapVC.m

```
@interface LakeMapVC ()<GMSMapViewDelegate>
...
@property(strong, nonatomic) UIButton *streetViewButton;
@property(assign, nonatomic) CLLocationCoordinate2D activeMarkerCoordinate;

@end

@implementation LakeMapVC

- (BOOL)mapView:(GMSMapView *)mapView didTapMarker:(GMSMarker *)marker {
  self.activeMarkerCoordinate = marker.position;
  ...
```

create this button in viewDidLoad

store the currently selected marker's position (lat/lng coordinates)

This coordinate needs to be stored in a property so we can pass it to StreetViewVC

Pass the selected marker's coordinate into StreetViewVC

LakeMapVC.m

called when streetViewButton is tapped

```
- (void)showStreetView:(id)sender
{
  CSStreetViewVC *streetViewVC = [[CSStreetViewVC alloc] init];
  streetViewVC.coordinate = self.activeMarkerCoordinate;
  [self presentViewController:streetViewVC
                     animated:YES
                   completion:^{
                     self.streetViewButton.alpha = 0.0;
                     self.mapView.selectedMarker = nil;
                   }];
}
```

send the currently selected marker's coordinate to the streetViewVC

manage the street view button display just like you did with the directions button

Exploring Google Maps for iOS

StreetViewVC should take in a single coordinate

StreetViewVC.h

```
#import <UIKit/UIKit.h>
#import <GoogleMaps/GoogleMaps.h>

@interface CSStreetViewVC : UIViewController

@property (assign, nonatomic) CLLocationCoordinate2D coordinate;

@end
```

use assign instead of strong, since this is a struct, not an object

Request a panorama for the coordinate that was passed in

A single Street View is called a panorama in the SDK

StreetViewVC.m

```
- (void)viewDidAppear:(BOOL)animated
{
  [super viewDidAppear:animated];

  GMSPanoramaService *service = [[GMSPanoramaService alloc] init];

  [service requestPanoramaNearCoordinate:self.coordinate
      callback:^(GMSPanorama *panorama, NSError *error) {

  }];
}
```

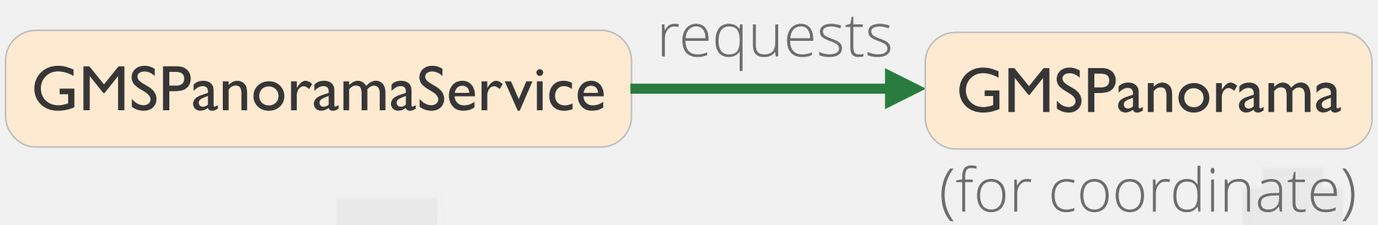
Use the panorama service object to request a panorama view

use the coordinate that was passed into the streetViewVC

A GMSPanoramaService requests a GMSPanorama object

StreetViewVC.m

```
[service requestPanoramaNearCoordinate:self.coordinate
    callback:^(GMSPanorama *panorama, NSError *error) {

}];
```



A GMSPanoramaCamera tells a panorama how to display

StreetViewVC.m

```
[service requestPanoramaNearCoordinate:self.coordinate
    callback:^(GMSPanorama *panorama, NSError *error) {
  GMSPanoramaCamera *camera =
      [GMSPanoramaCamera cameraWithHeading:180 pitch:0 zoom:1 FOV:90];
}];
```

GMSPanoramaCamera

tells the panorama view
which direction to display

What a standard GMSPanoramaCamera displays

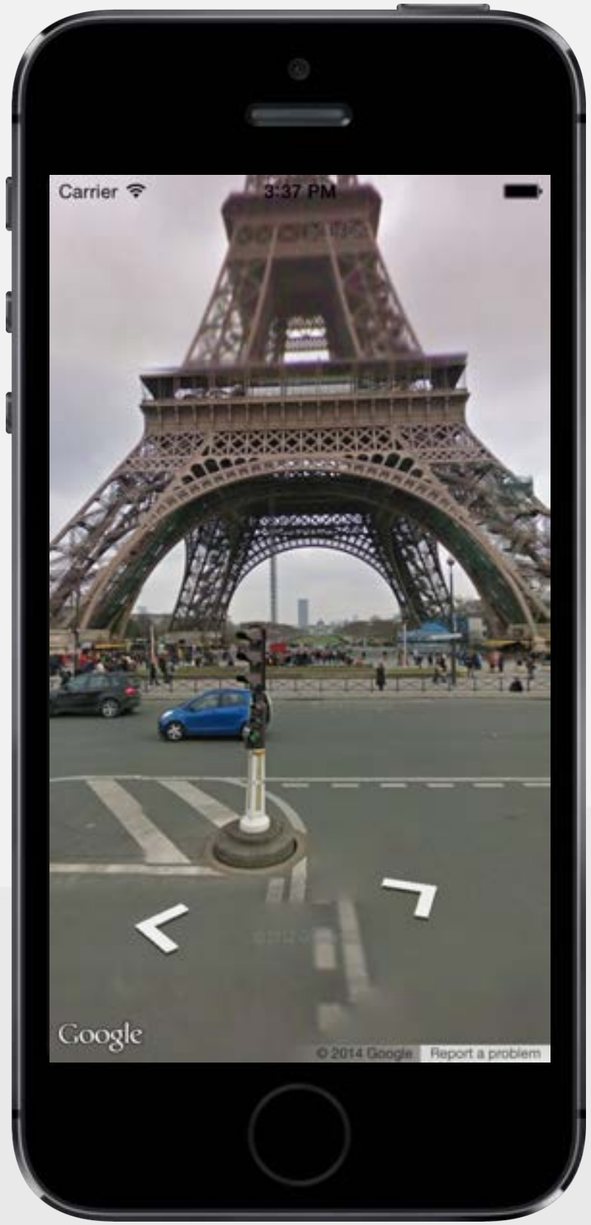
```
GMSPanoramaCamera *camera =
    [GMSPanoramaCamera cameraWithHeading:180
                                   pitch:0
                                    zoom:1
                                     FOV:90];
```

These are essentially “default” settings for a GMSPanoramaCamera

A GMSPanoramaCamera with the heading changed

```
GMSPanoramaCamera *camera =
    [GMSPanoramaCamera cameraWithHeading:130
                                   pitch:0
                                    zoom:1
                                     FOV:90];
```

heading determines which direction the camera is pointing (like bearing in GMSMapView)



A GMSPanoramaCamera with the pitch changed

```
GMSPanoramaCamera *camera =
    [GMSPanoramaCamera cameraWithHeading:130
                                   pitch:30
                                    zoom:1
                                     FOV:90];
```

`pitch` determines if the camera is tilted pointing up or down

- a pitch from 1 to 90 tilts the camera up
- a pitch from -1 to -90 tilts the camera down

A GMSPanoramaCamera with the FOV changed

```
GMSPanoramaCamera *camera =
    [GMSPanoramaCamera cameraWithHeading:130
                                   pitch:30
                                    zoom:1
                                     FOV:150];
```

FOV is like depth of field

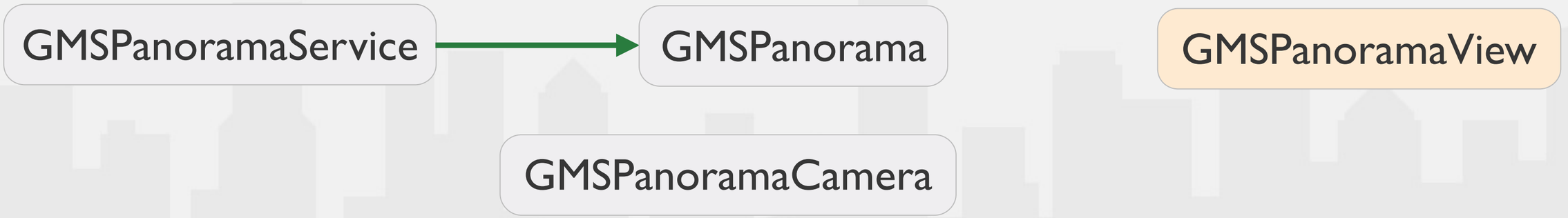
lower FOV looks more like a zoomed in tunnel (1 - 90)

higher FOV looks more like a fisheye lens (90 - 160)

Creating a GMSPanoramaView

StreetViewVC.m

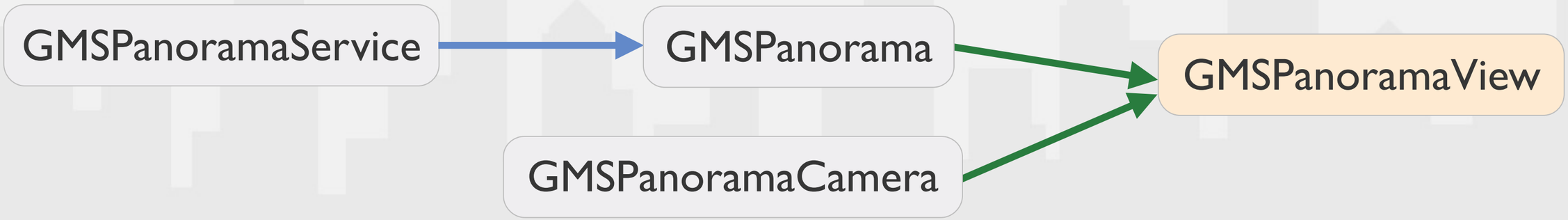
```
[service requestPanoramaNearCoordinate:self.coordinate
    callback:^(GMSPanorama *panorama, NSError *error) {
  GMSPanoramaCamera *camera =
      [GMSPanoramaCamera cameraWithHeading:180 pitch:0 zoom:1 FOV:90];
  GMSPanoramaView *panoView = [[GMSPanoramaView alloc] init];
}];
```



Setting properties on the GMSPanoramaView

StreetViewVC.m

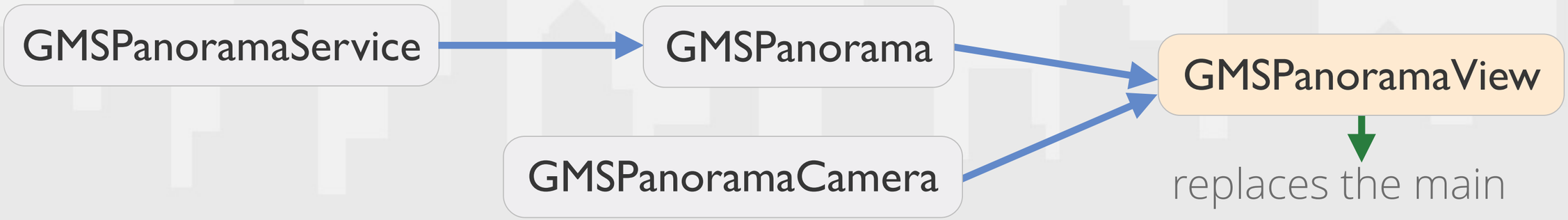
```
[service requestPanoramaNearCoordinate:self.coordinate
    callback:^(GMSPanorama *panorama, NSError *error) {
  GMSPanoramaCamera *camera =
     [GMSPanoramaCamera cameraWithHeading:180 pitch:0 zoom:1 FOV:90];
  GMSPanoramaView *panoView = [[GMSPanoramaView alloc] init];
  panoView.camera = camera;
  panoView.panorama = panorama;
}];
```



Replacing the StreetViewVC main view with a GMSPanoramaView

StreetViewVC.m

```
[service requestPanoramaNearCoordinate:self.coordinate
    callback:^(GMSPanorama *panorama, NSError *error) {
  GMSPanoramaCamera *camera =
     [GMSPanoramaCamera cameraWithHeading:180 pitch:0 zoom:1 FOV:90];
  GMSPanoramaView *panoView = [[GMSPanoramaView alloc] init];
  panoView.camera = camera;
  panoView.panorama = panorama;
  self.view = panoView;
}];
```



Adding a close button to go back to LakeMapVC

StreetViewVC.m

```
  [service requestPanoramaNearCoordinate:self.coordinate
      callback:^(GMSPanorama *panorama, NSError *error) {
    ...
    self.view = panoView;

    UIButton *closeStreetViewButton = [UIButton buttonWithType:UIButtonTypeSystem];
    ...
    [self.view addSubview:closeStreetViewButton];
  }];
}

- (void)closeStreetView:(id)sender {
  [self dismissViewControllerAnimated:YES completion:nil];
}
```

call this method when the button is tapped

Hide the street view button on long press

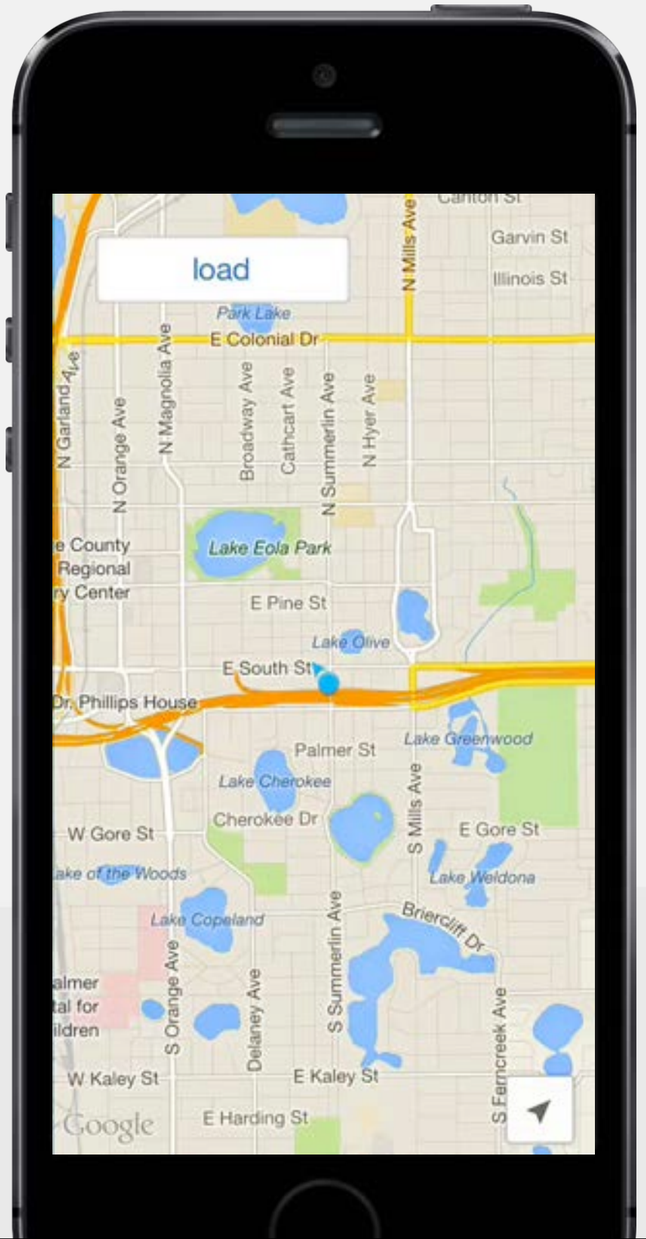
LakeMapVC.m

```
- (void)mapView:(GMSMapView *)mapView didLongPressAtCoordinate:(CLLocationCoordinate2D)coordinate {
  ...
  if (self.streetViewButton.alpha > 0.0) {
    self.streetViewButton.alpha = 0.0;
  }
}

- (void)mapView:(GMSMapView *)mapView didTapAtCoordinate:(CLLocationCoordinate2D)coordinate {
  ...
  if (self.streetViewButton.alpha > 0.0) {
    self.streetViewButton.alpha = 0.0;
  }
}

- (void)mapView:(GMSMapView *)mapView didMove:(BOOL)gesture {
  ...
  if (self.streetViewButton.alpha > 0.0) {
    self.streetViewButton.alpha = 0.0;
  }
}
```

Demo: Street View working (most of the time)

Problem

A blank view appears when you try to show a street view for a place that doesn't have one



Exploring Google Maps for iOS

Show an error message if the GMSPanorama object is nil

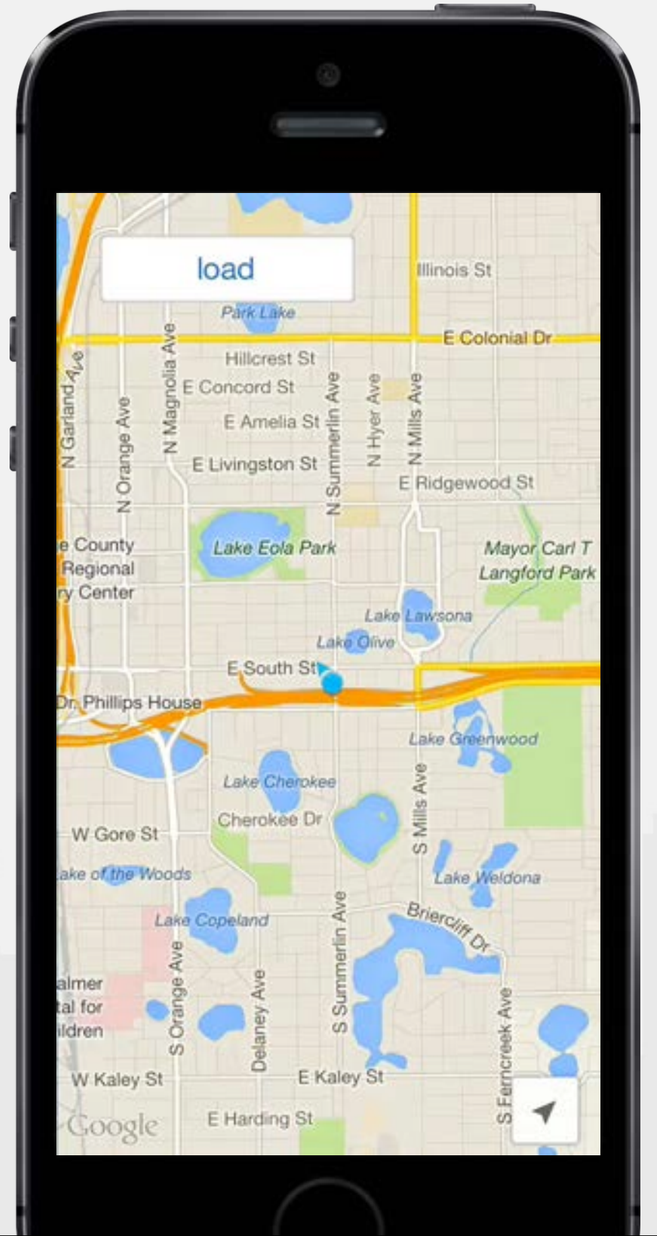
StreetViewVC.m

```
  [service requestPanoramaNearCoordinate:self.coordinate
      callback:^(GMSPanorama *panorama, NSError *error) {
    if (panorama != nil) {
      GMSPanoramaCamera *camera = ...
      GMSPanoramaView *panoView = [[GMSPanoramaView alloc] init];
      ...
      self.view = panoView;

      UIButton *closeStreetViewButton = ...
    } else {
      // show an alert view that says no data is available
      [self closeStreetView:nil];
    }
 }];
}
```

panorama will be `nil` if there is no street view data available at this coordinate

if panorama is `nil`, alert that no data is available and close `StreetViewVC`

Demo: Street View showing an error message when the panorama is nil



Exploring Google Maps for iOS