



Exploring
Google Maps
for iOS

Exploring Google Maps for iOS

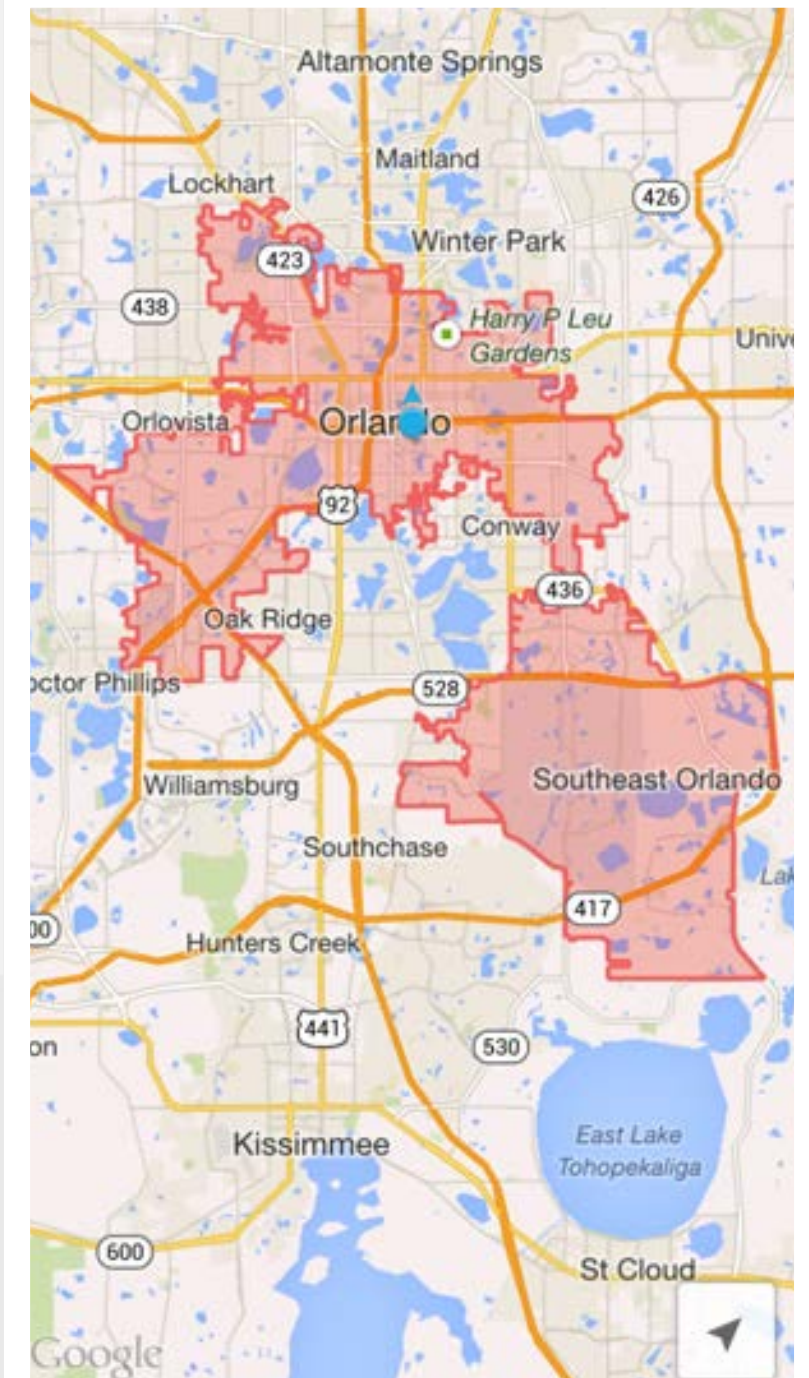
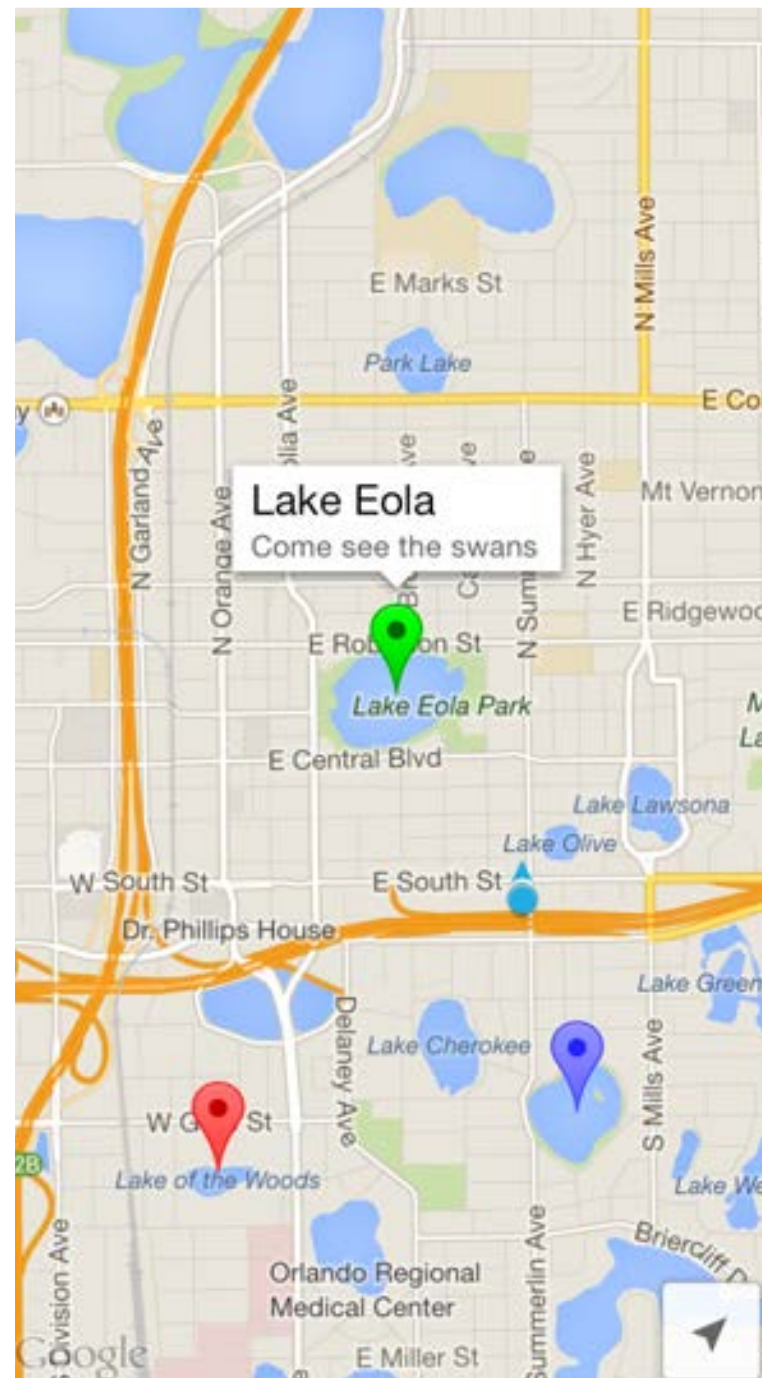
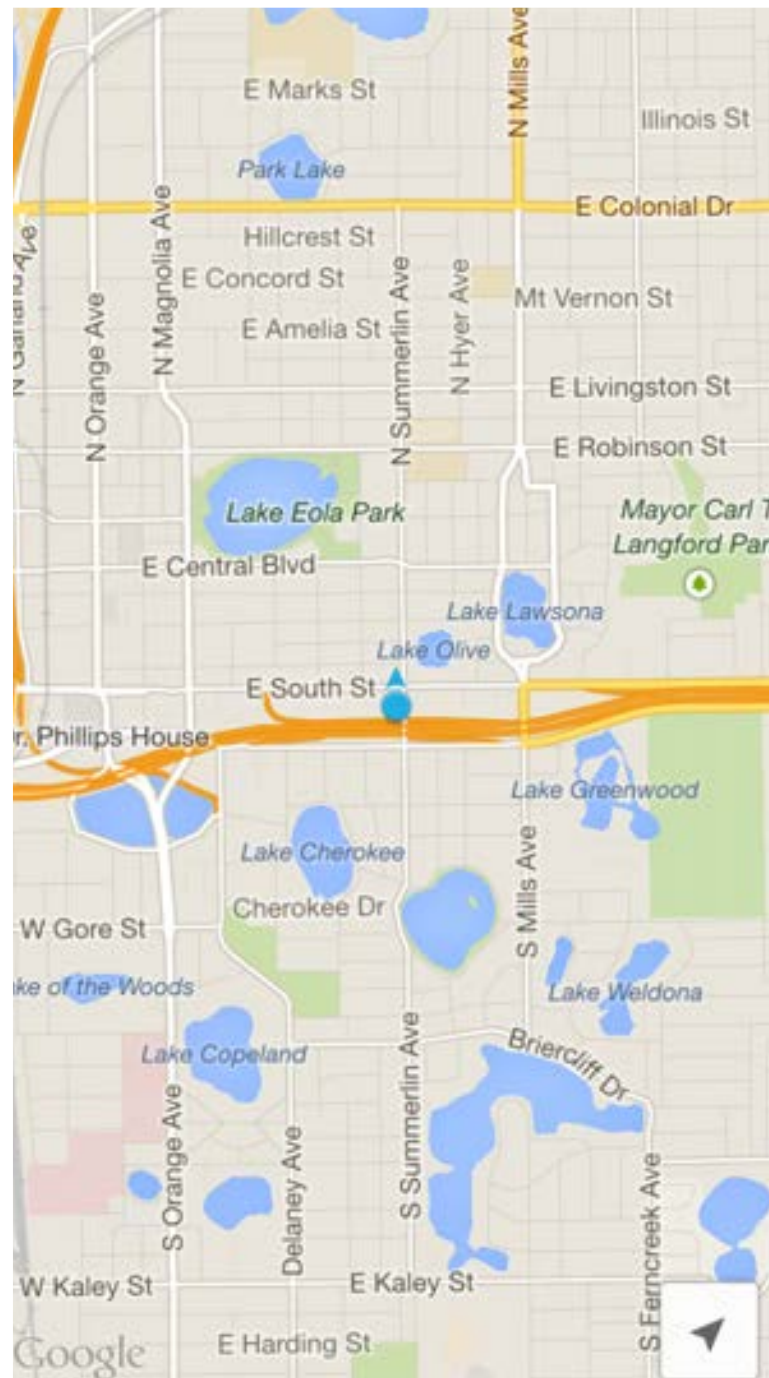
Level 1 - Displaying a map



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.

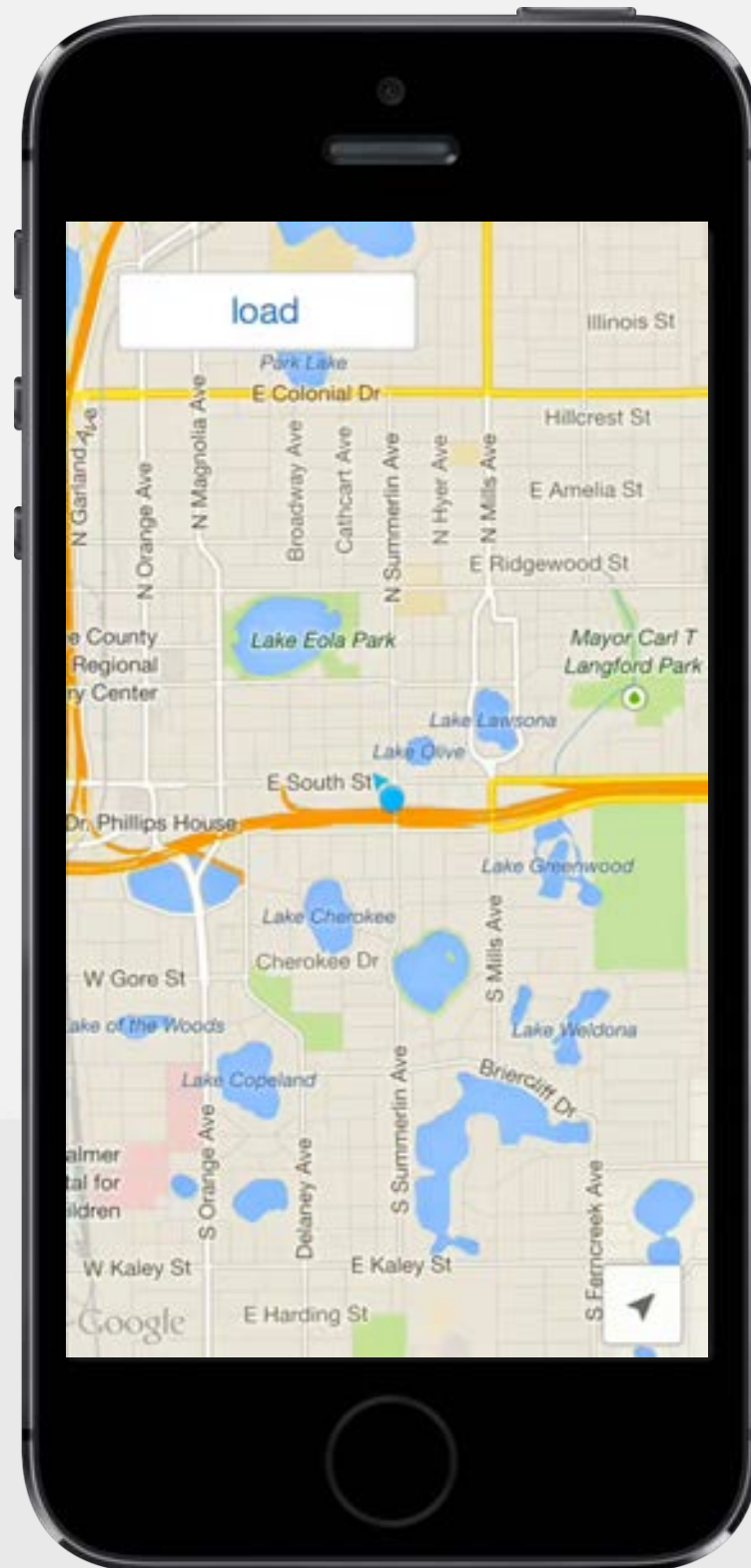
Using Google Maps gives you:

- 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



Exploring
Google Maps
for iOS

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



Exploring
Google Maps
for iOS

Getting Started Step One: Get an API Key

Every app that includes Google Maps must have an API Key

`3kj2Jl3k1j3k1317hj13143hh6k1lkk4` ← 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.



Exploring
Google Maps
for iOS

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:@"3kj2Jl3k1j3k1317hj13143hh6k1lkk4"];

    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:@"3kj2Jl3k1j3k1317hj13143hh6k1lkk4"];

    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

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

LakeMapVC.m

```
#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

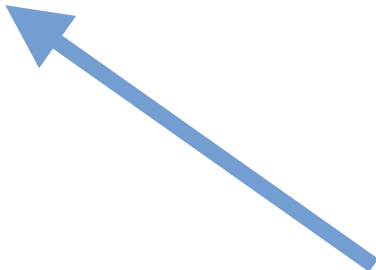


Exploring
Google Maps
for iOS

Creating a GMSCameraPosition

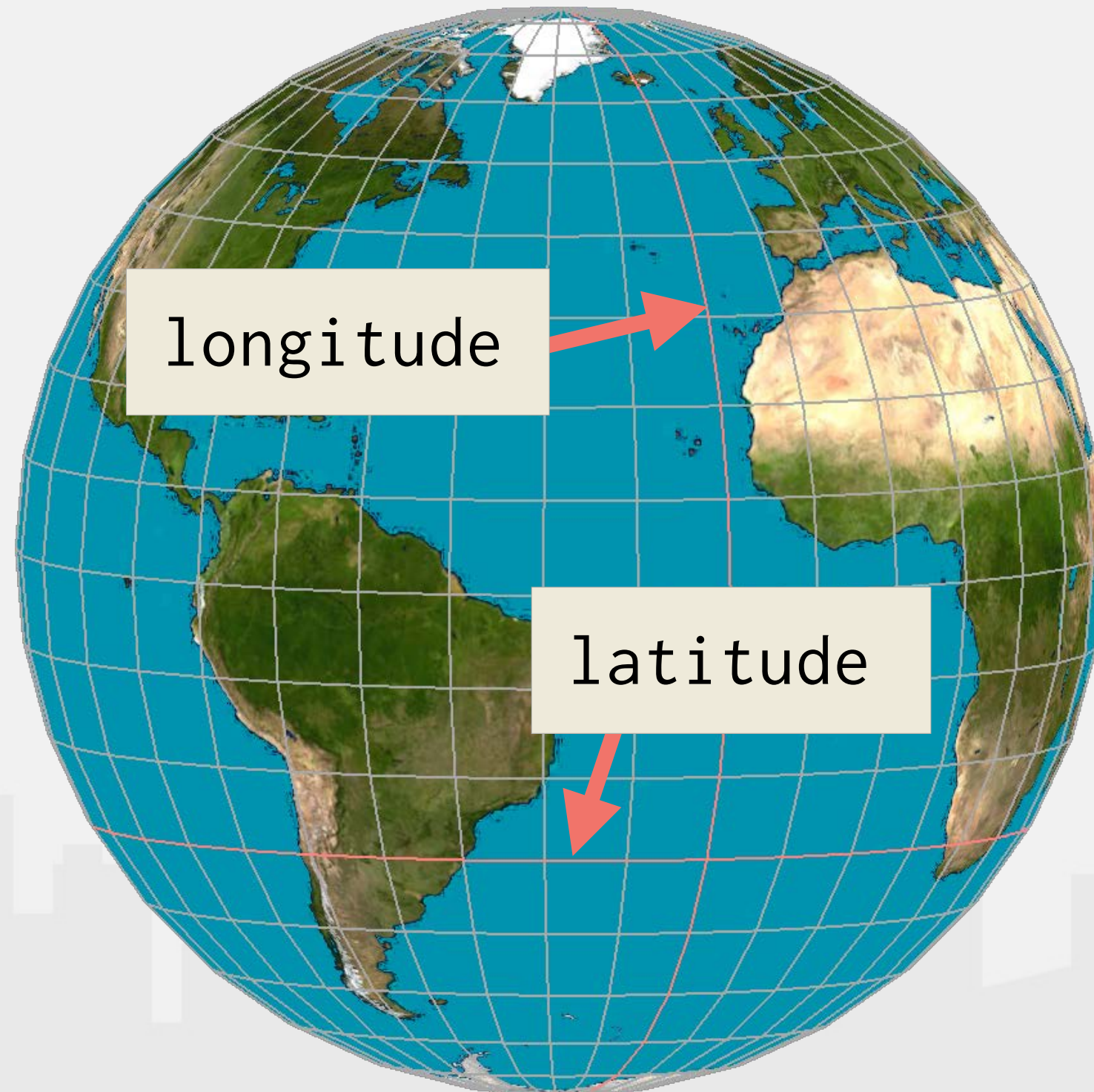
LakeMapVC.m

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



Exploring
Google Maps
for iOS

GMSCameraPosition properties - latitude and longitude



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

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

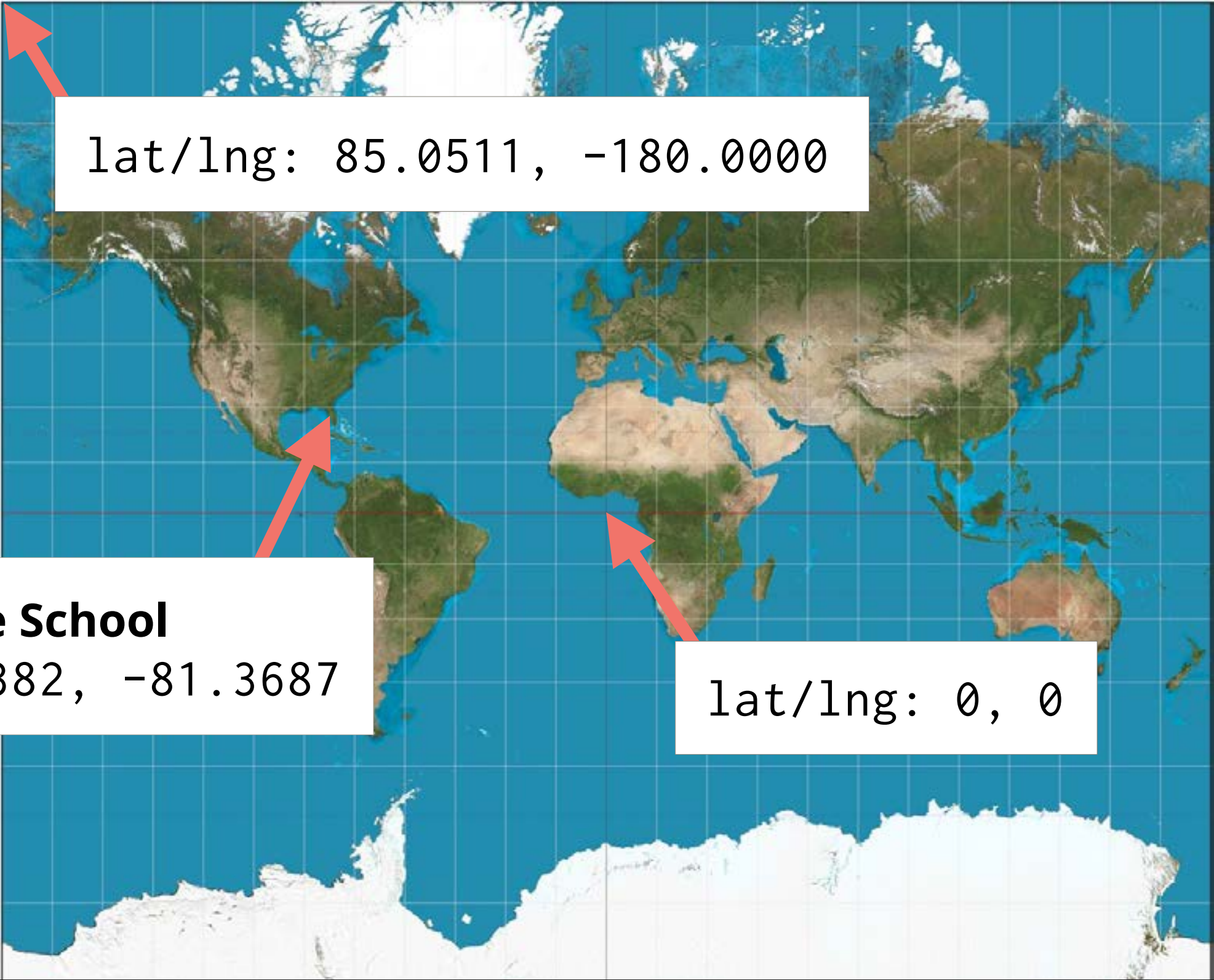
latitude - runs east/west

longitude - runs north/south



Exploring
Google Maps
for iOS

Examples of latitude and longitude points on a map

A world map showing the continents and oceans with a grid of latitude and longitude lines. Three red arrows point to specific locations: one to the North Pole, one to the Gulf of Mexico, and one to the Equator in Africa.

lat/lng: 85.0511, -180.0000

Envy Labs / Code School

lat/lng: 28.5382, -81.3687

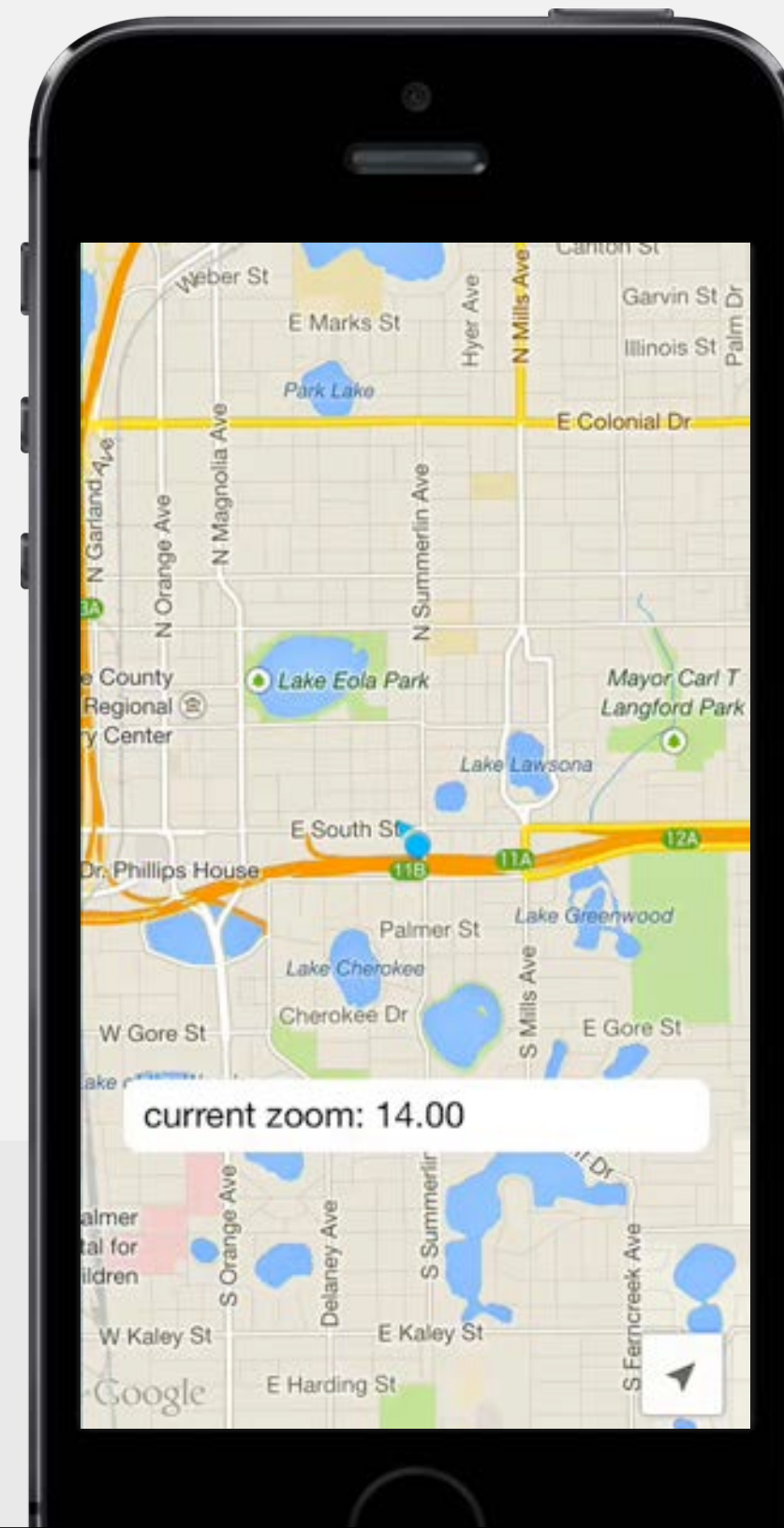
lat/lng: 0, 0

GMSCameraPosition properties - zoom

Show more or less of the map at once

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

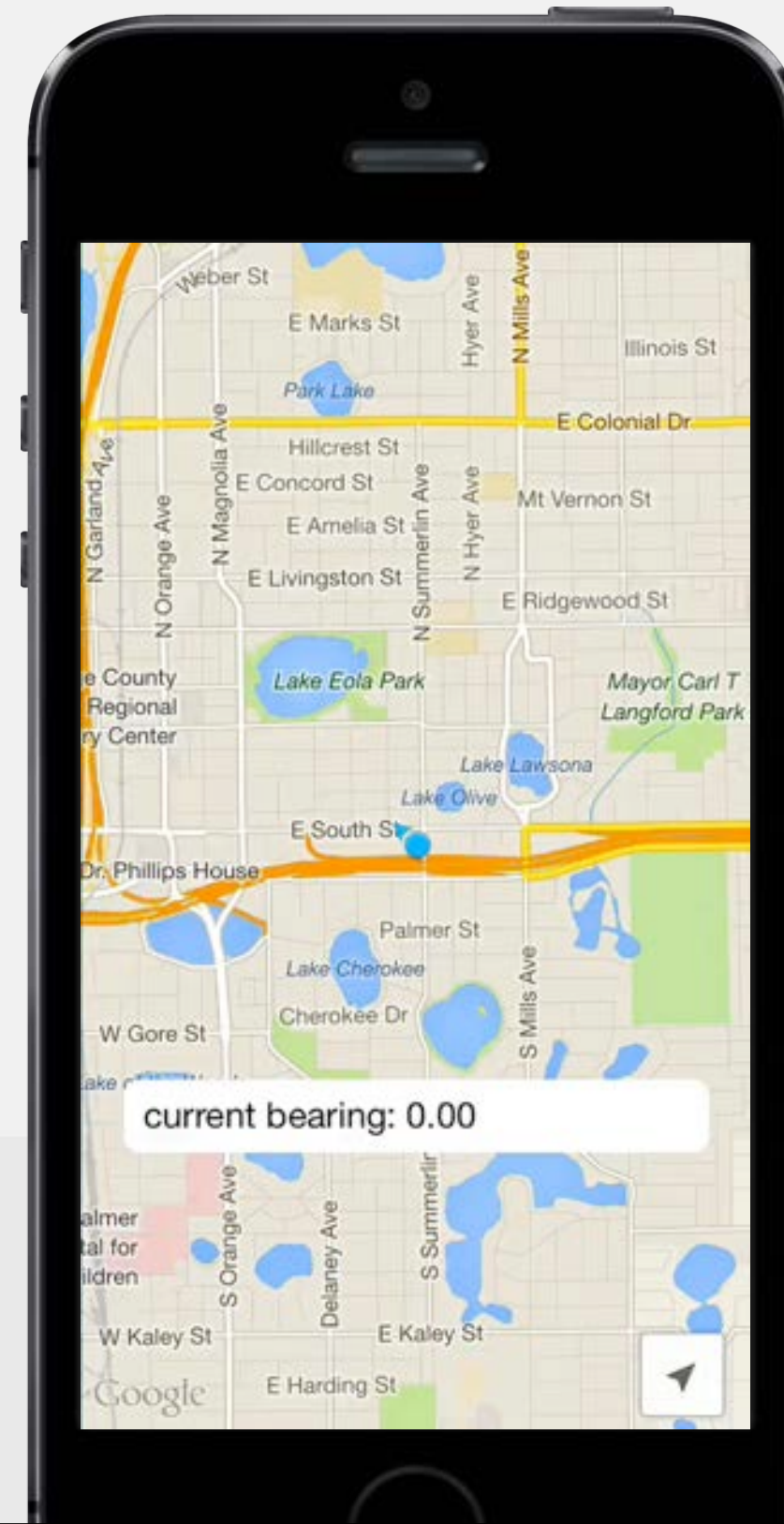
A lower number means more of the map is showing (zoomed out)



GMSCameraPosition properties - bearing

How much the map is rotated

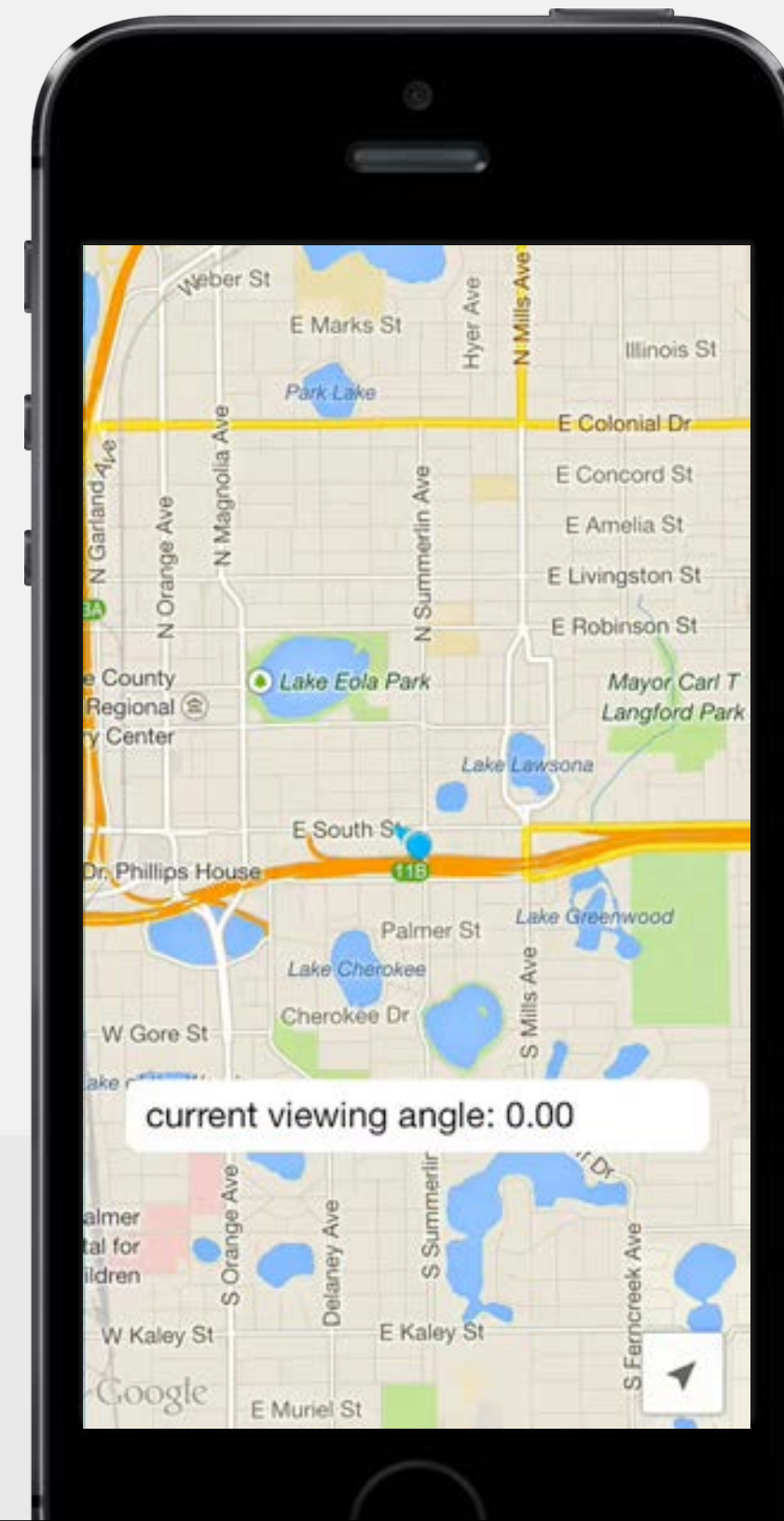
The default rotation is 0
(the top points north)



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)



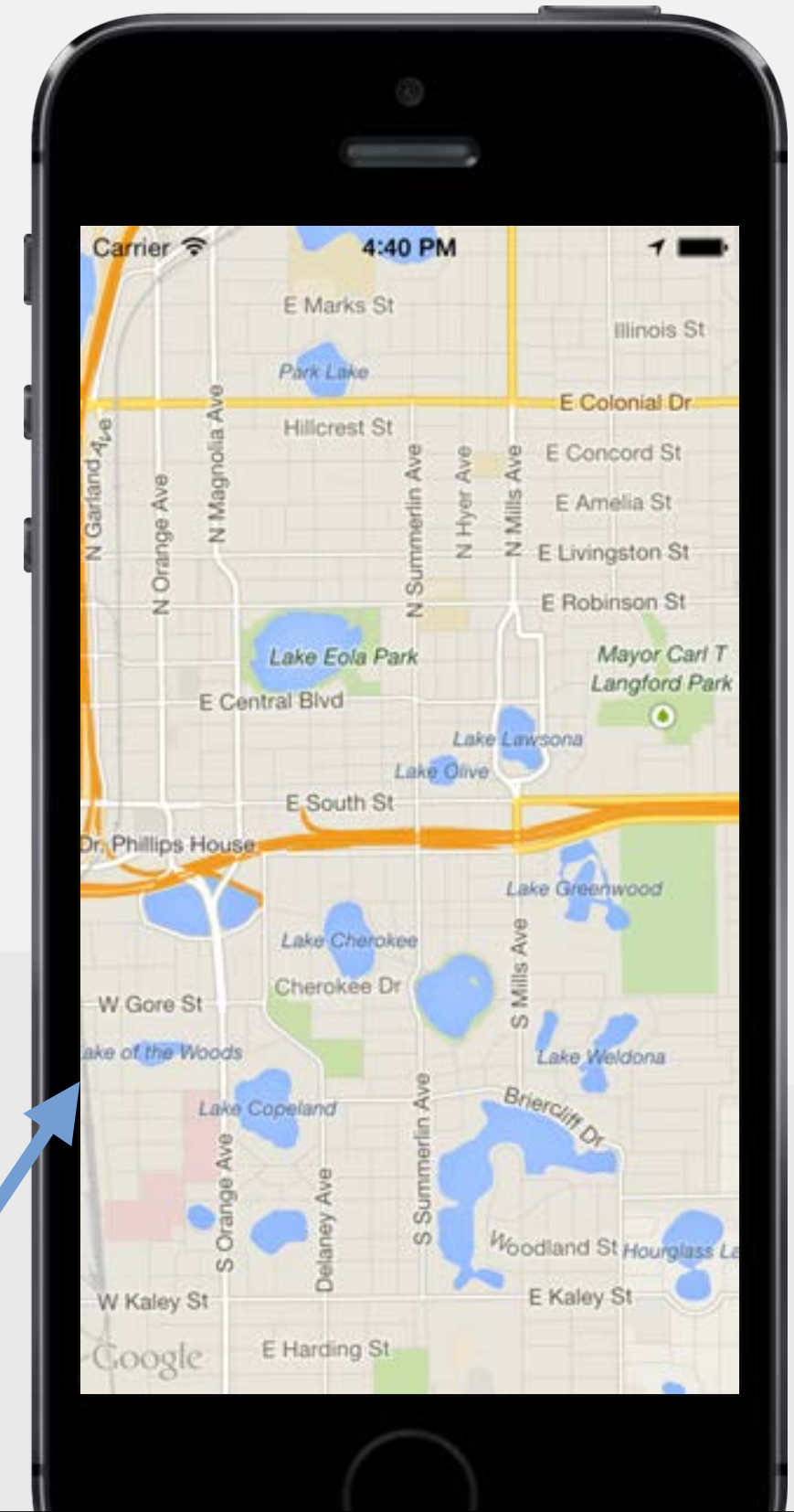
Creating a GMSCameraPosition object with a convenience initializer

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!

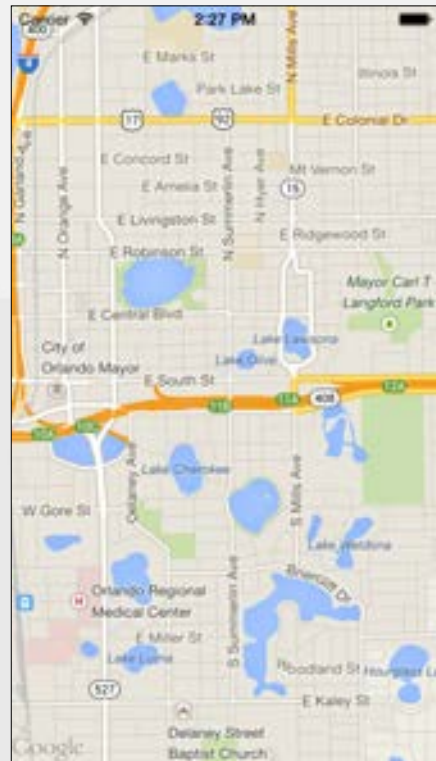


Changing the map type

LakeMapVC.m

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

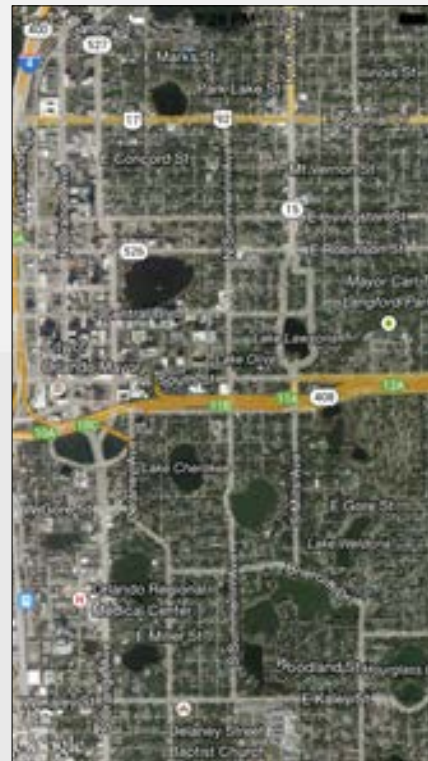
kGMSTypeNormal



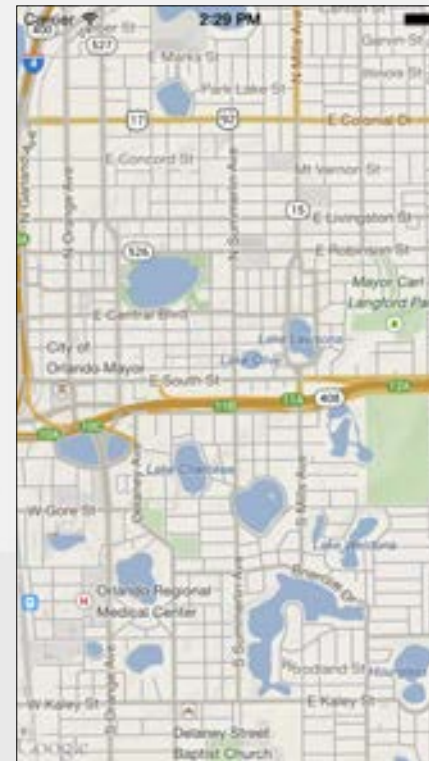
kGMSTypeSatellite



kGMSTypeHybrid



kGMSTypeTerrain

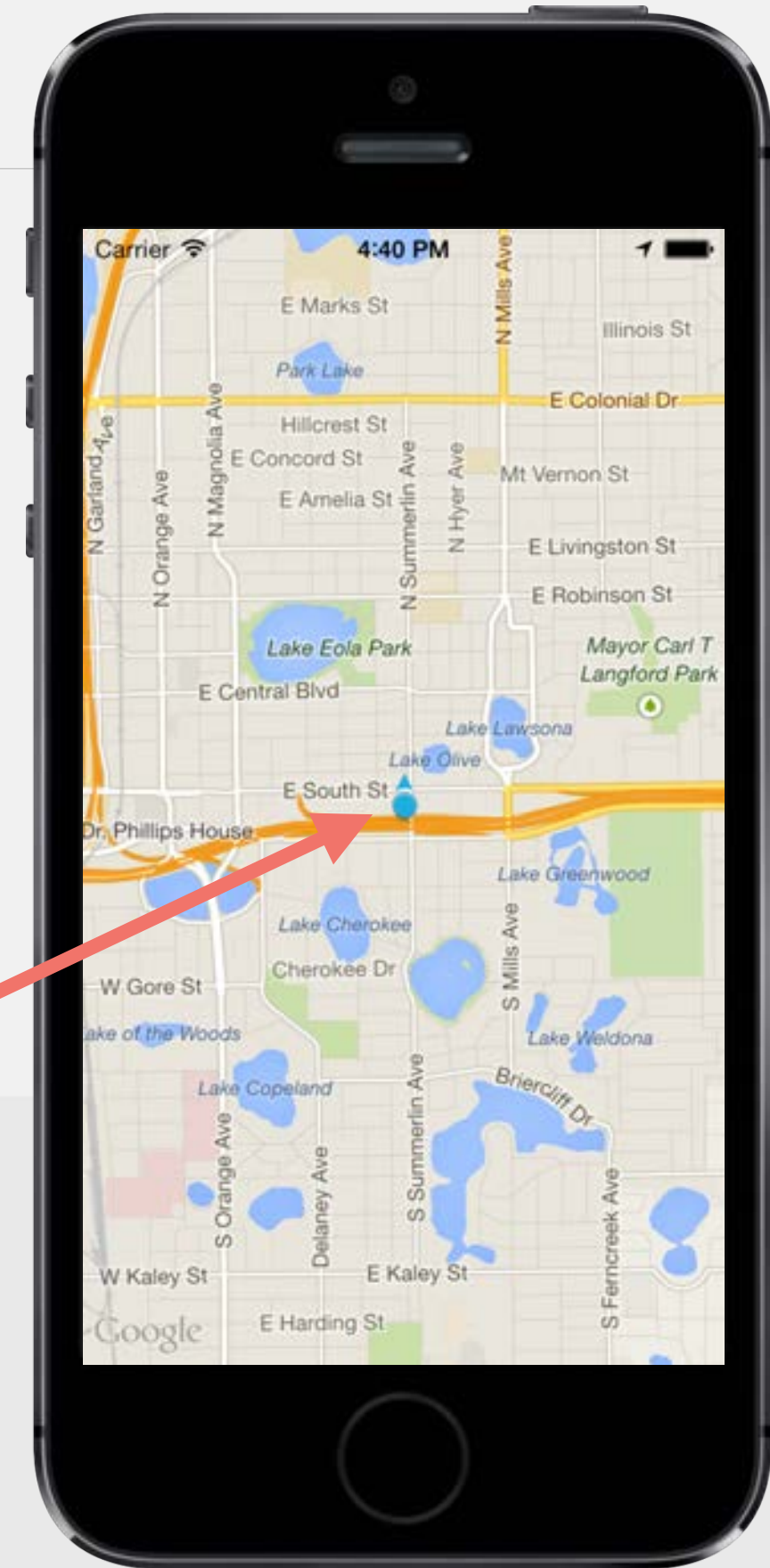


Showing your current location on the map

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 mapView.settings property has a few other BOOL options

scrollGestures

zoomGestures

tiltGestures

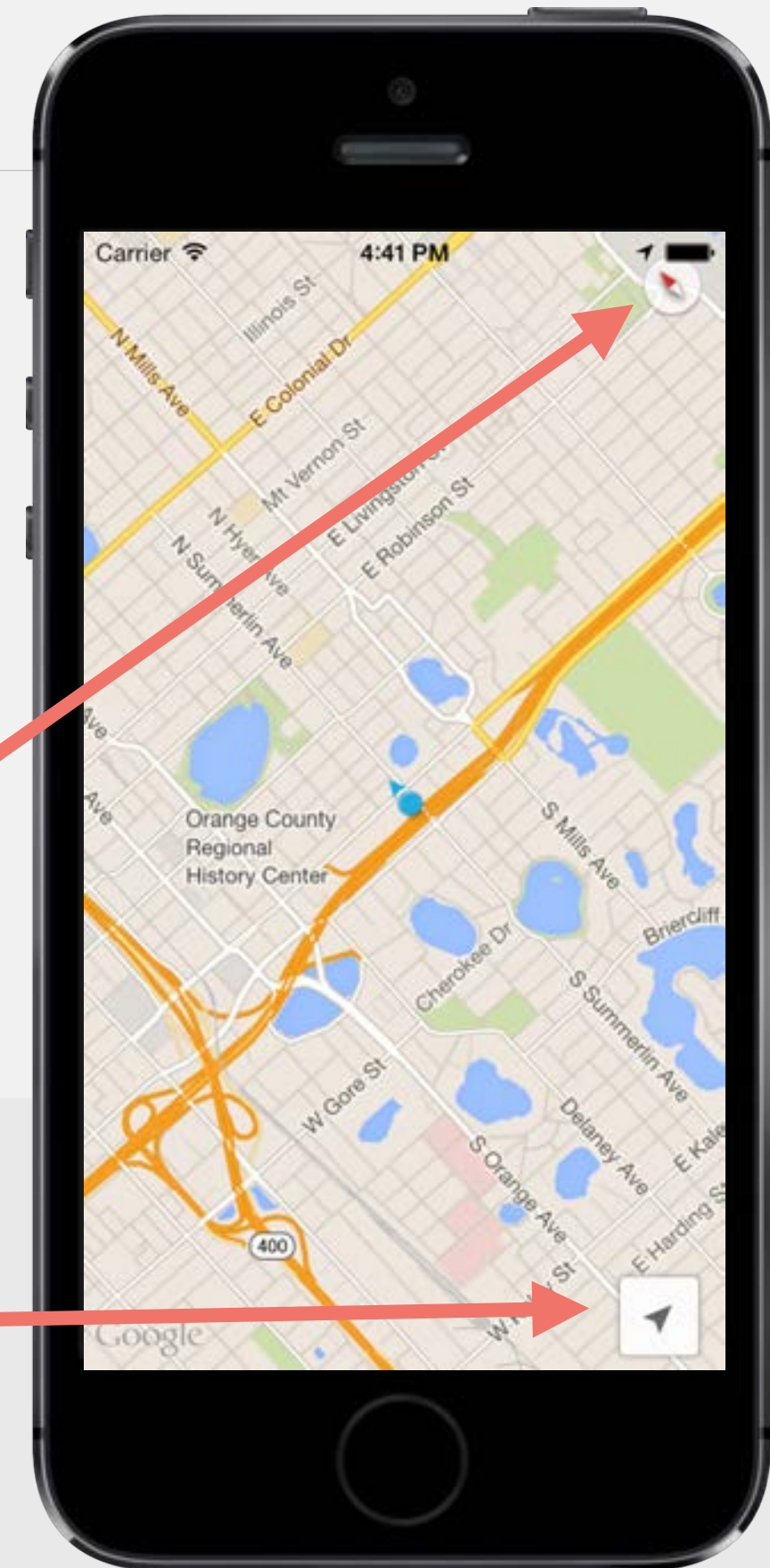
consumesGesturesInView

rotateGestures

indoorPicker

The compass
shows up when the
map is rotated

Find and center
the map on your
location

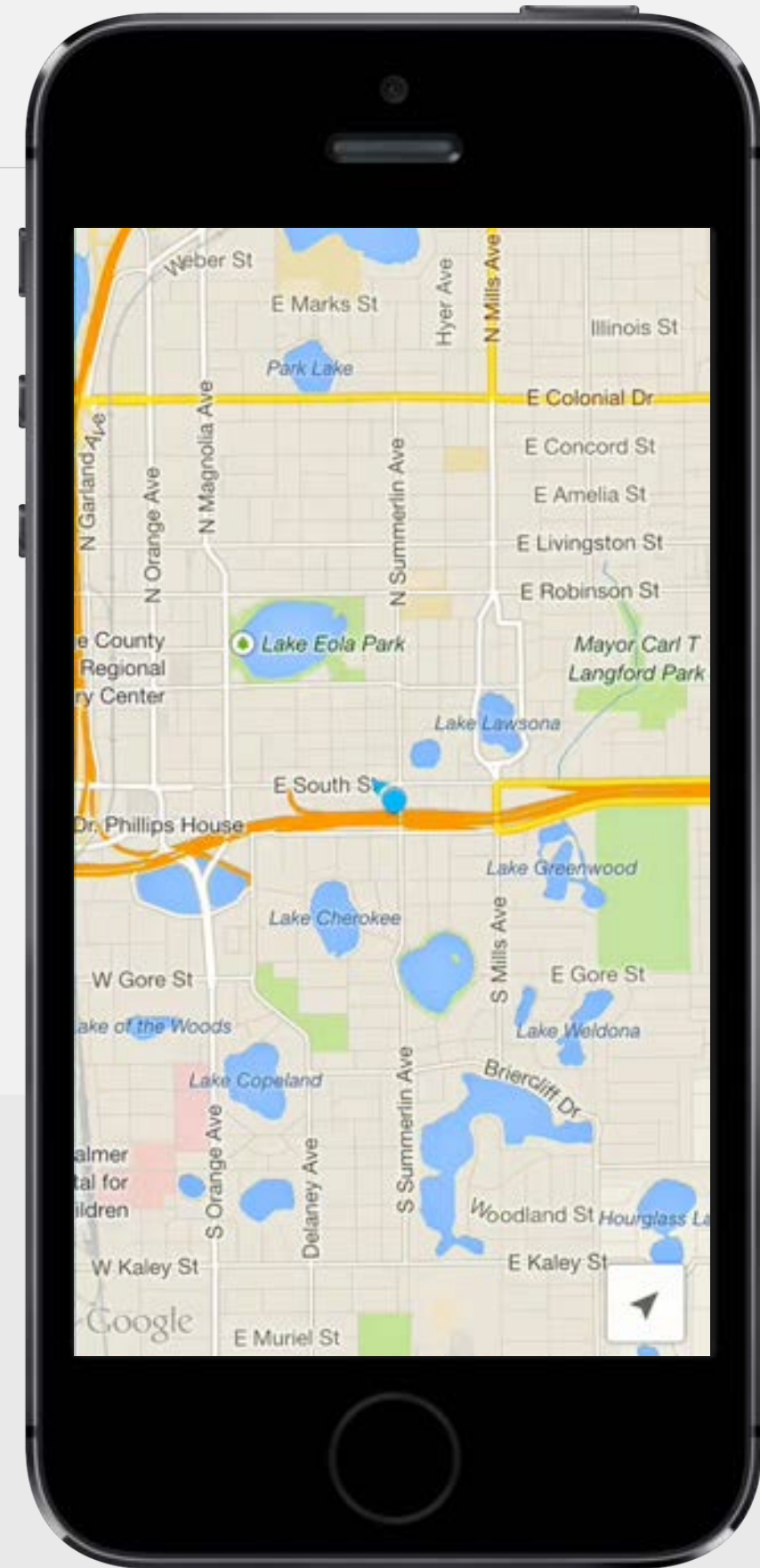


Constraining the zoom options

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

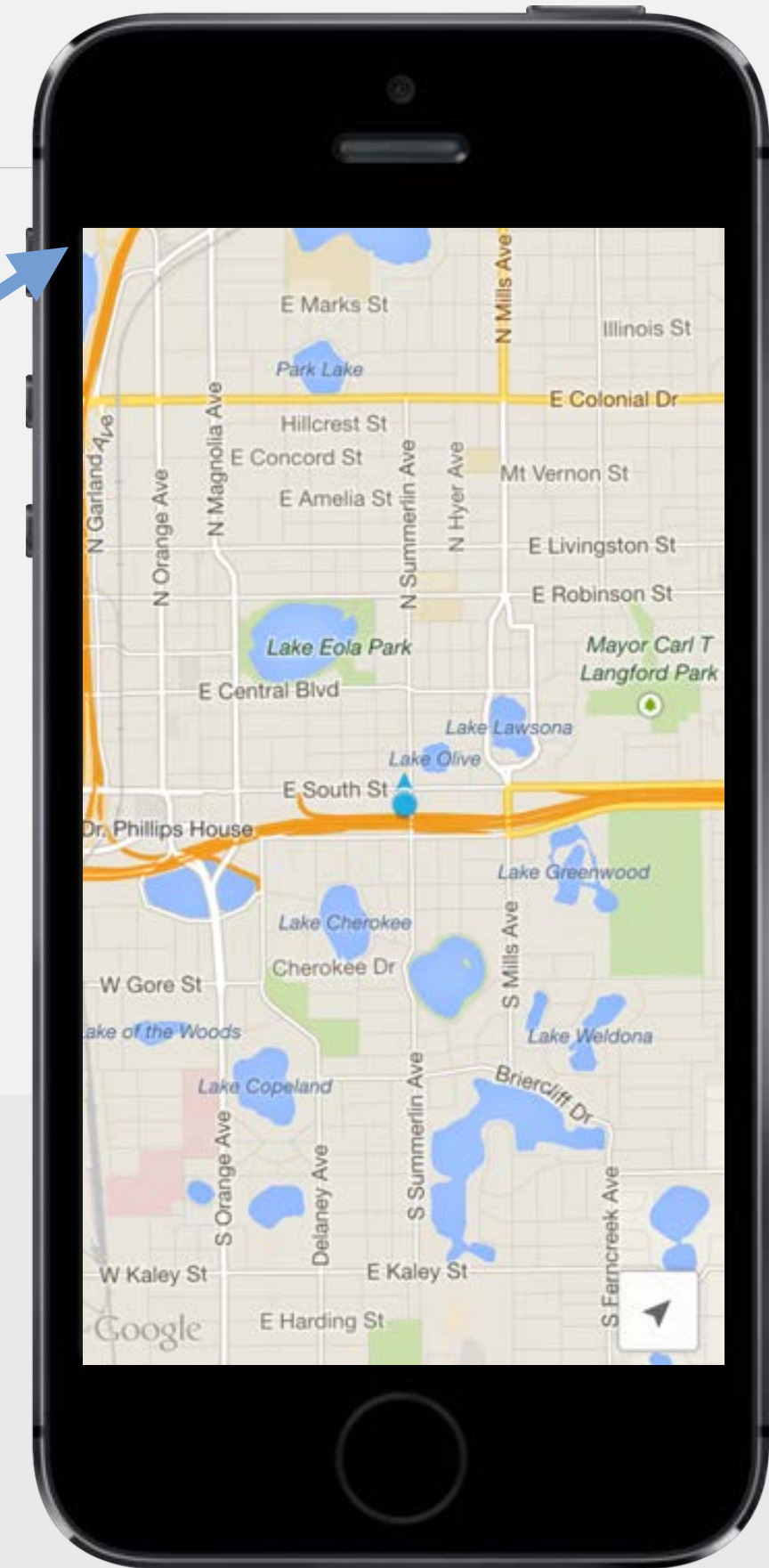
```
@implementation LakeMapVC
```

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

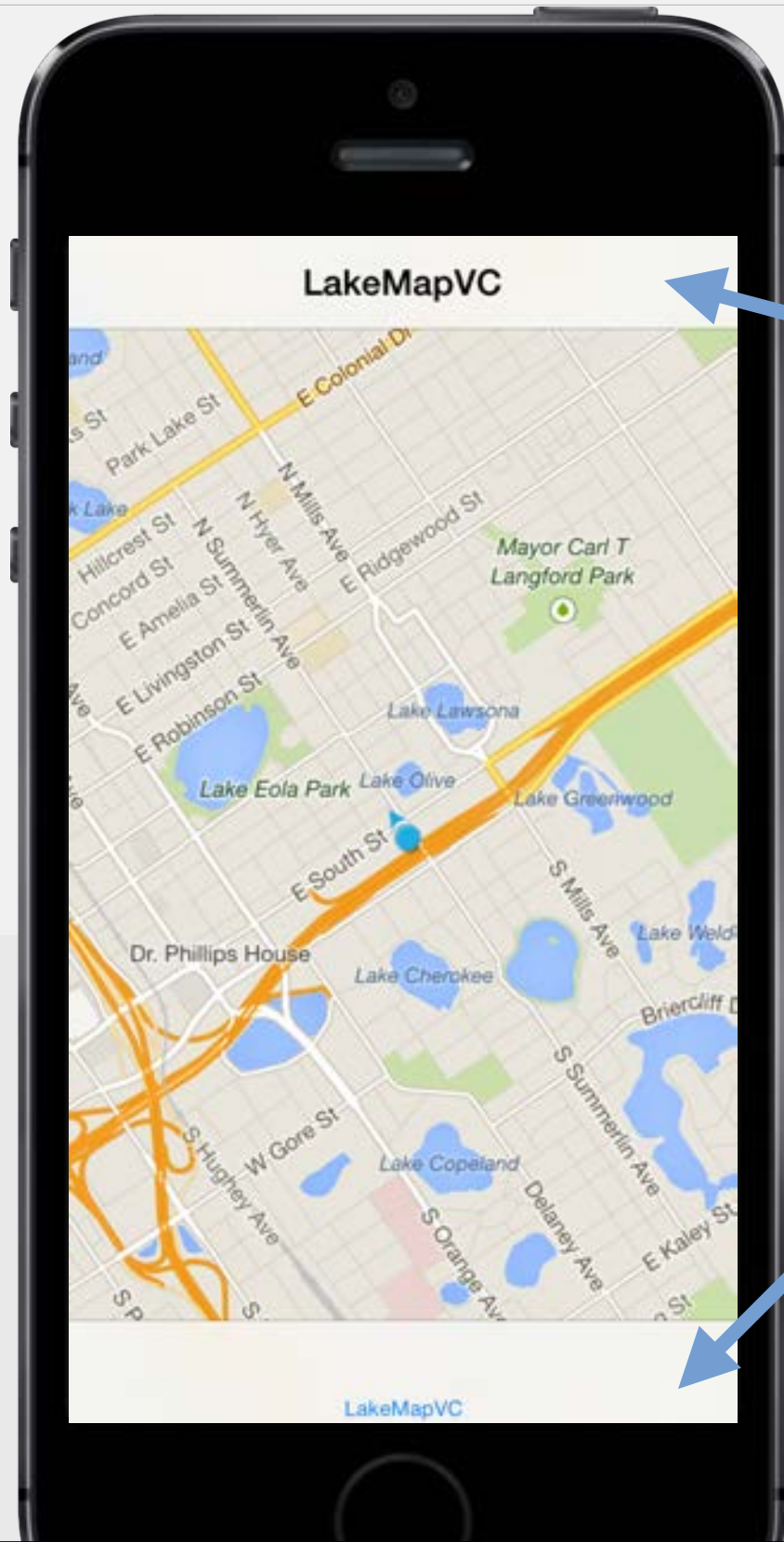
```
- (BOOL)prefersStatusBarHidden {  
    return YES;  
}
```

```
@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

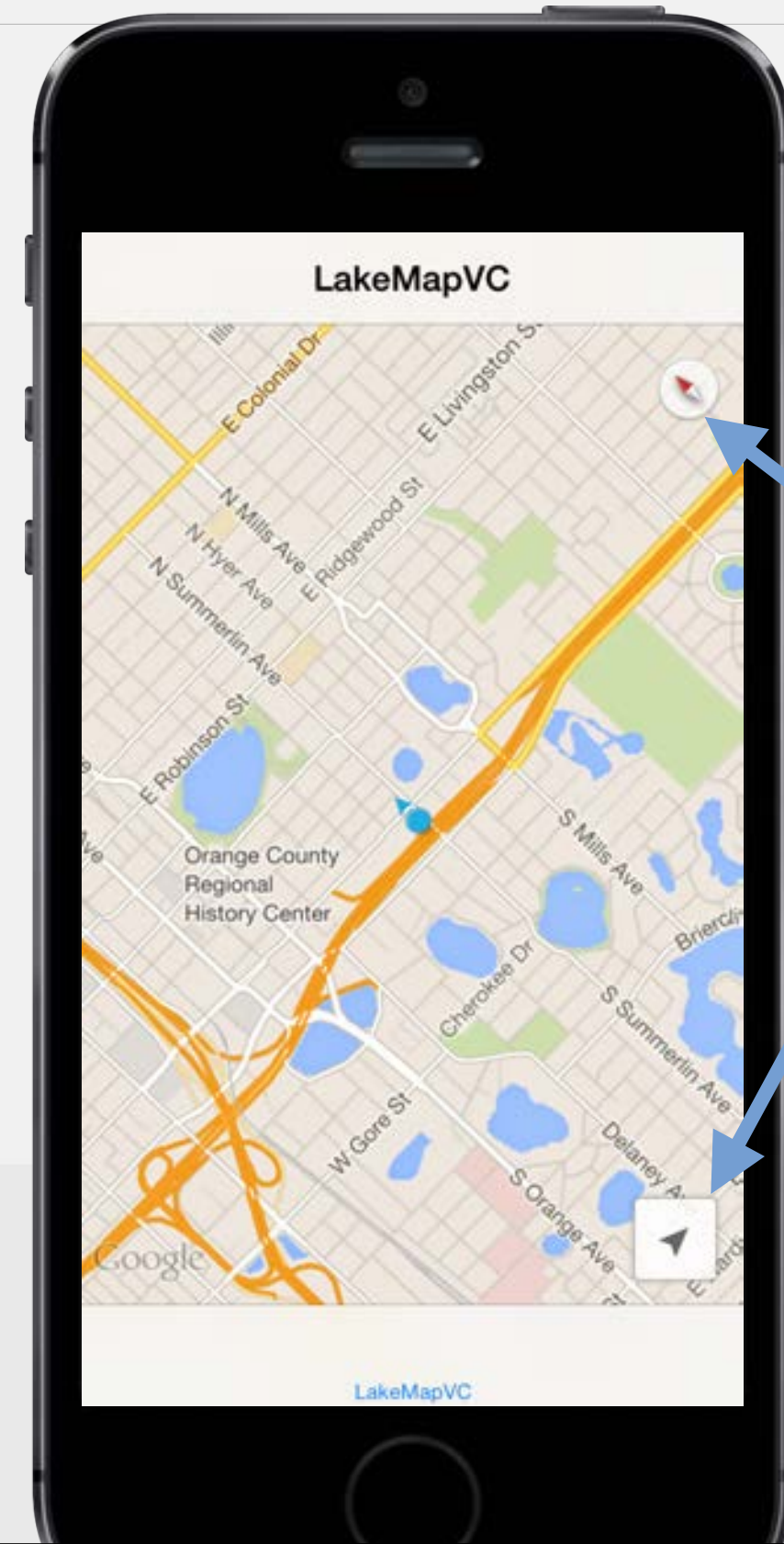
Adjusting the map controls when nav/tab bars are showing

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