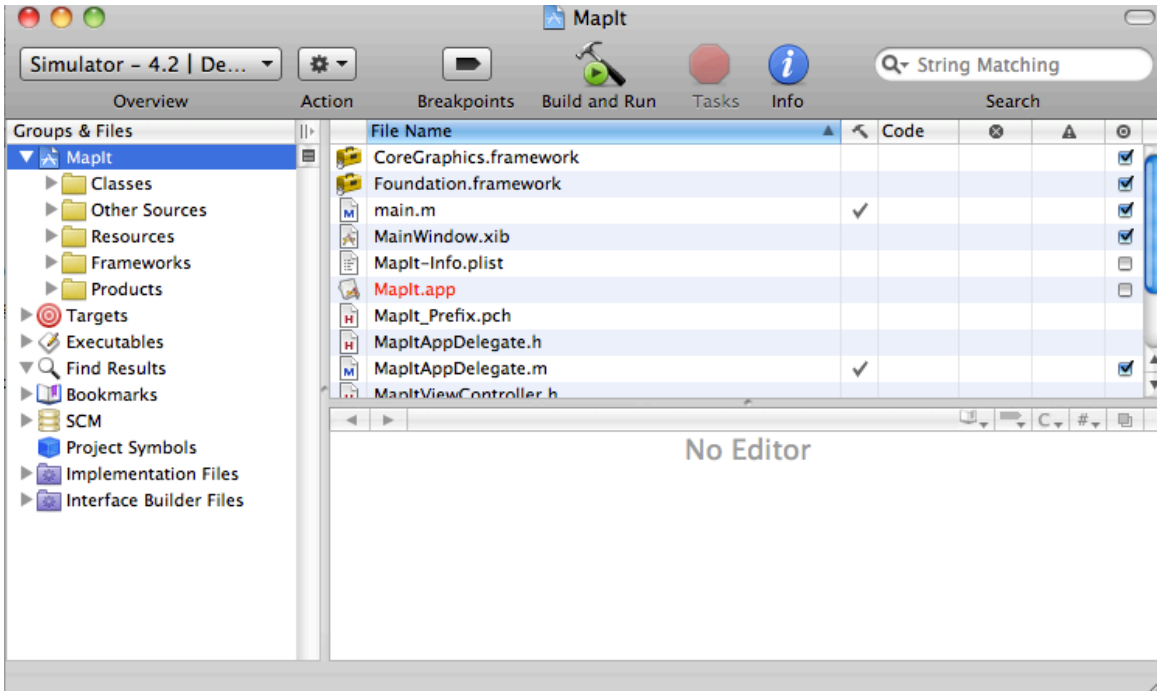
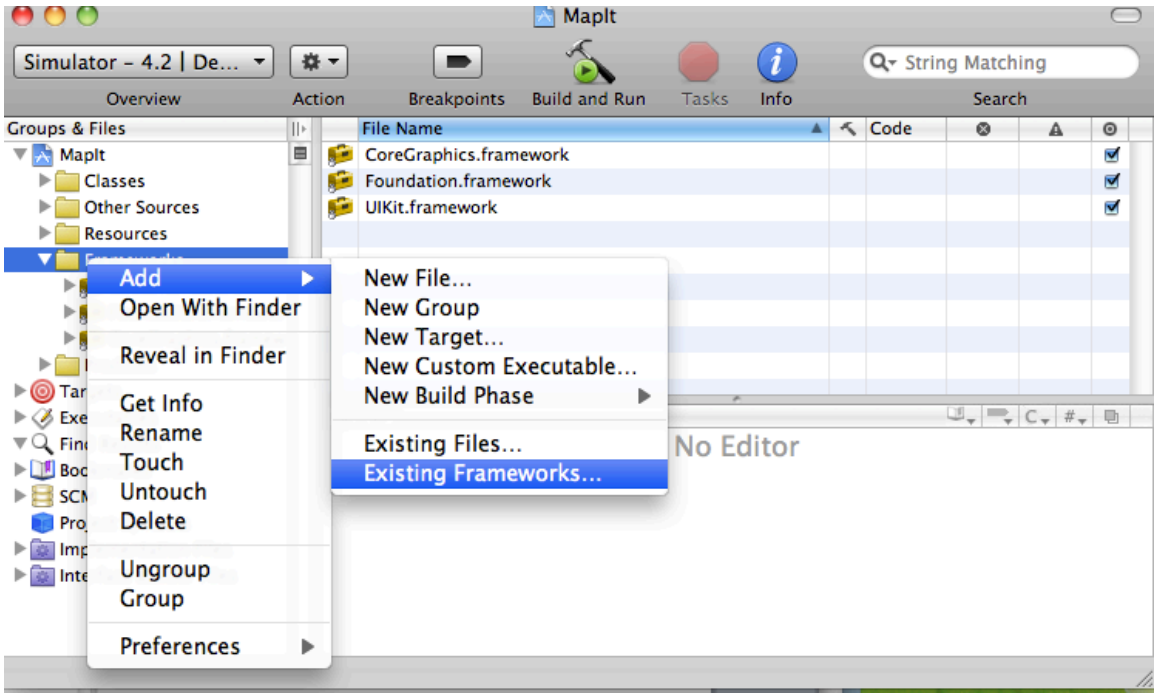
MapIt—Lab 4

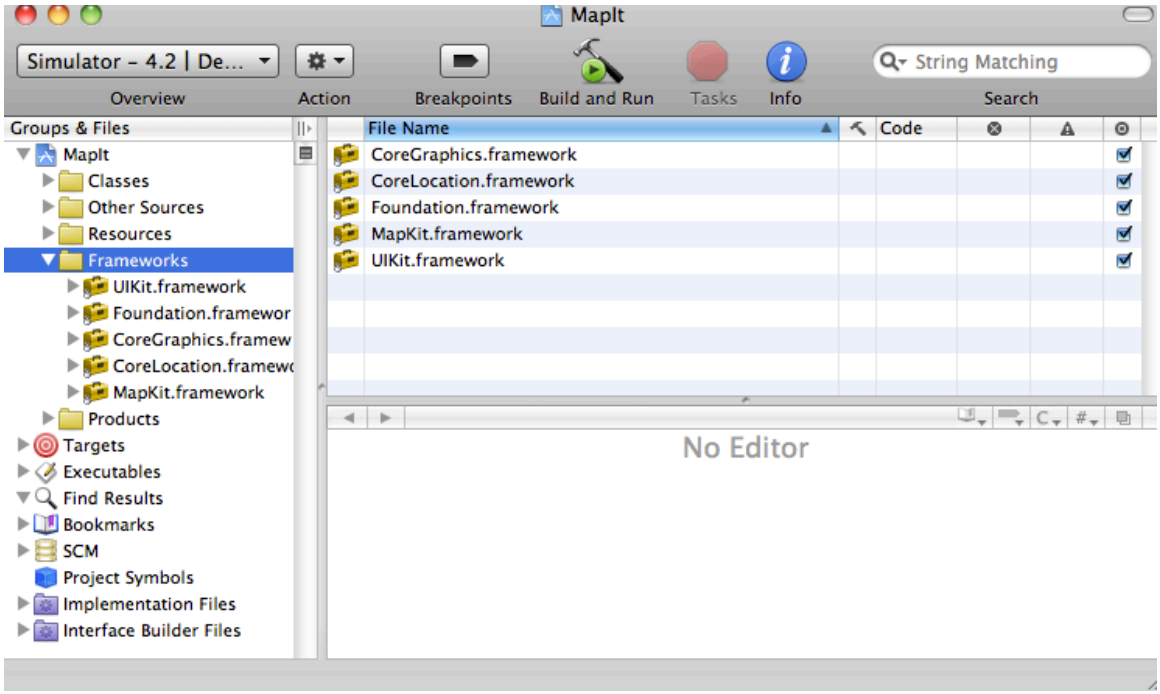
Program logic from iPhone and iPad Apps for Absolute Beginners by Rory Lewis

1). Create a View-based Application. Name it MapIt

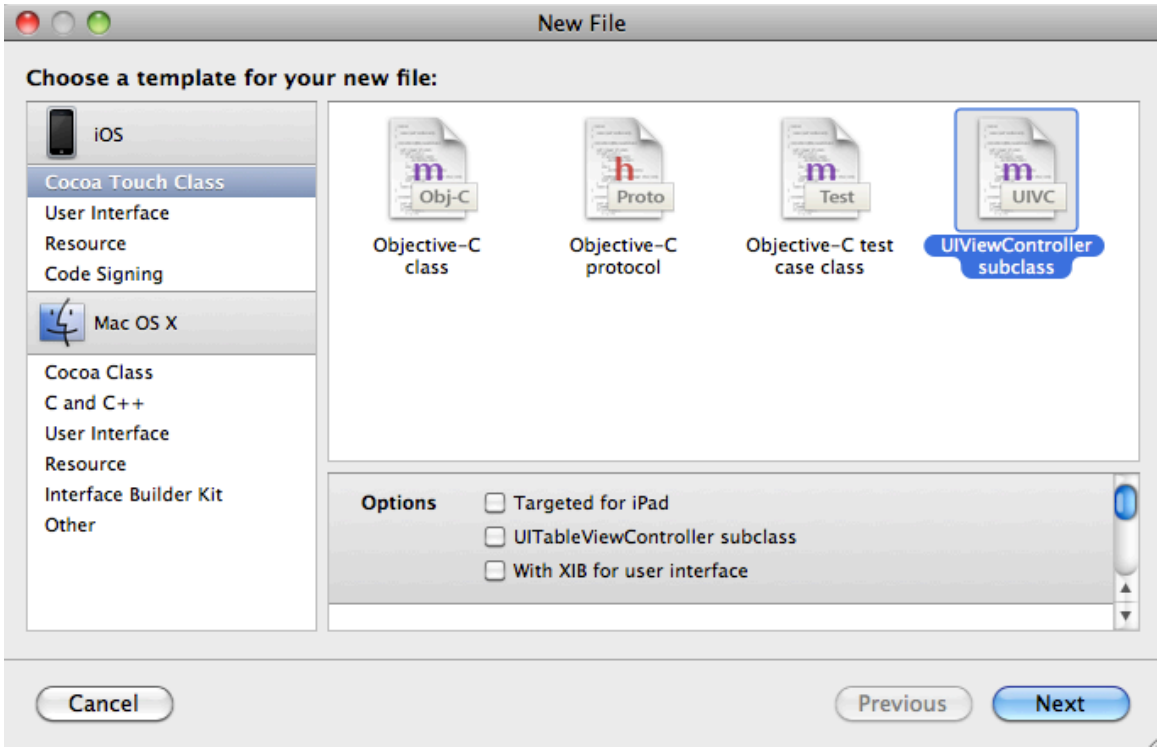


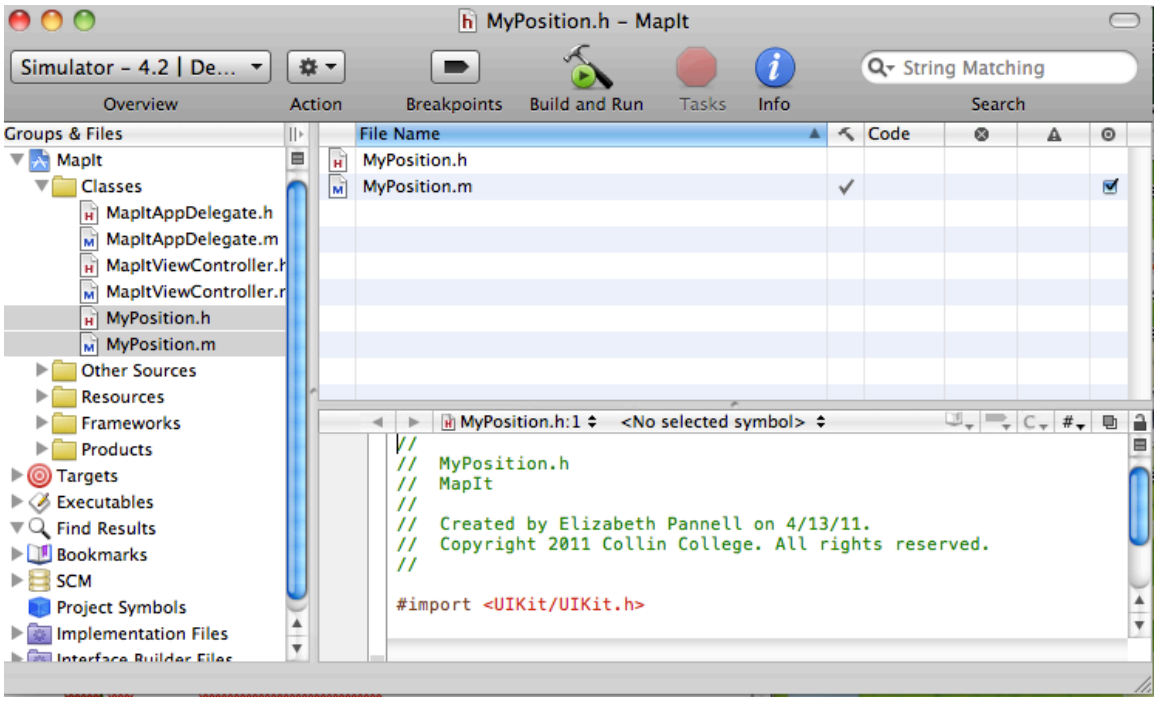
2). Right click on Frameworks and ADD both CoreLocation and MapKit Framework.. You can add them at the same time if you hold down the Command key when you select.



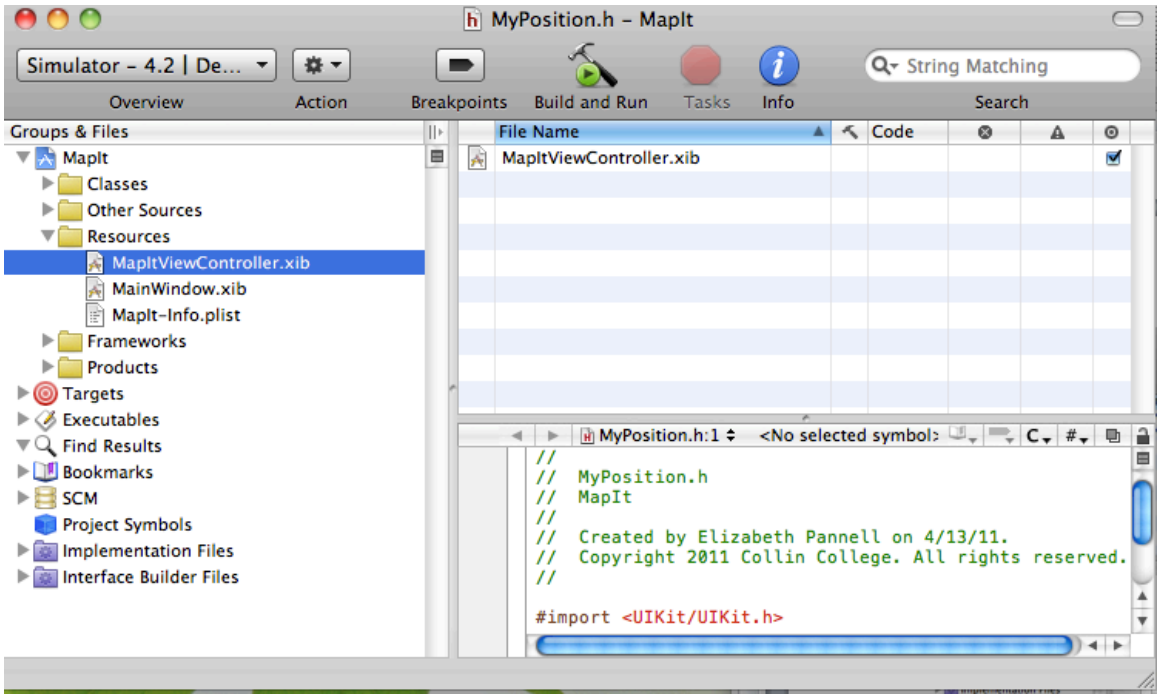


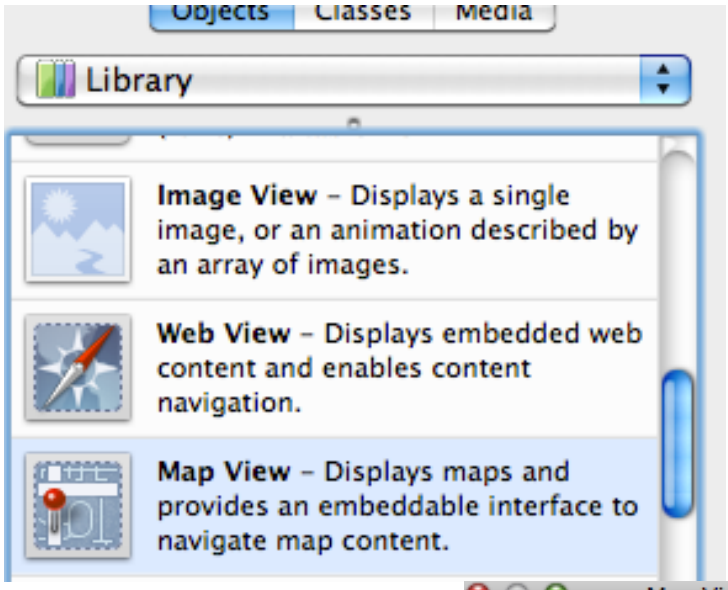
3. We are going to use annotations to show locations on a map, so…Select classes, and add a NEW UIViewController.h and m file. Name it MyPosition. Make sure that the iPad, xib file and UITableViewController are NOT selected.

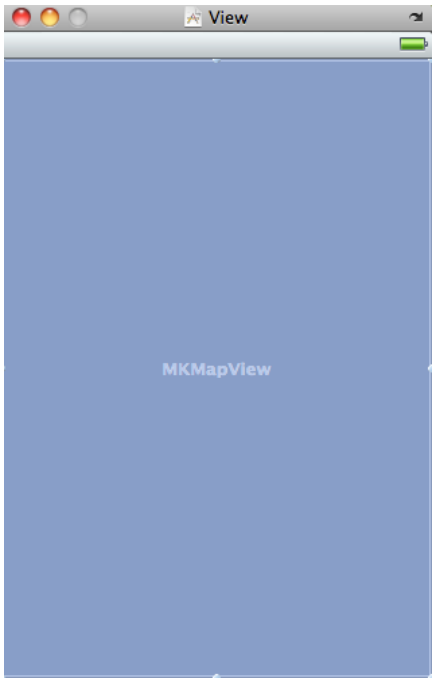


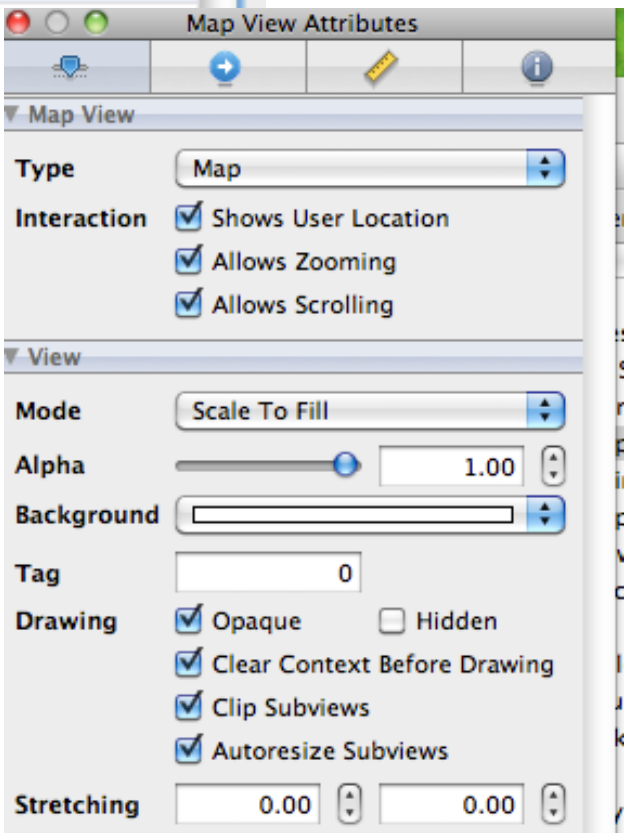


4. We will add content to the MyPosition ViewController later, for now let's go ahead and open MapItViewController.xib and go into Interface Builder and add a MapView to the View. Go to Inspector (command 1) and make sure you click the "shows user location" option. Save and close Interface Builder. See screen shots below:

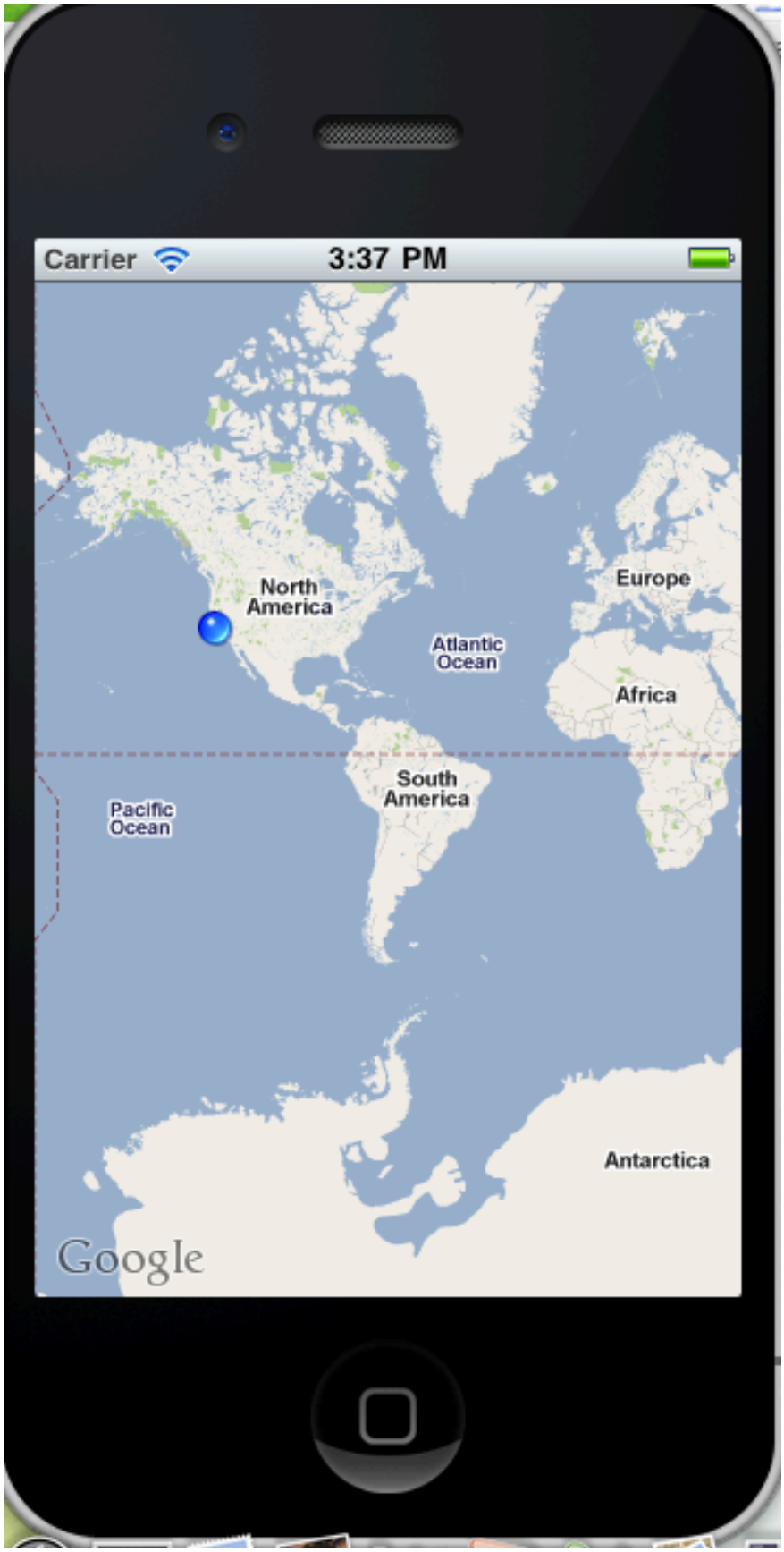








5. Save XCode and Build. You should see the location of the Apple home office on the West Coast. This is the default location. If all you see is Europe and Asia, use your mouse to drag the map image until you see North America. A Blue pin should be on top of Apple's location.


Carrier
3:37 PM
North America
Europe
Atlantic Ocean
Africa
South America
Pacific Ocean
Antarctica
Google

6. Now we are going to customize and set your own "custom" information. On an actual device, you can set it to detect your current location, however, the simulator (and your computer) does not have this ability so we are going to "hard" code the geo-location.

Open MapItViewController.h file and import the MapKit/MapKit.h framework. We will also add the MKMapViewDelegagate to the UIViewController class –see the code below! Also we will create an Outlet for the MKMapView.

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

@interface MapItViewController : UIViewController
<MKMapViewDelegate>{
    IBOutlet MKMapView *mapView;
}
@property (nonatomic, retain) IBOutlet MKMapView *mapView;

@end
```

7. Now open the MapItViewController.m file. This is where you are going to set your geo location and set the pin to show the location. In this example, the location is set to Preston Ridge Campus in Frisco Texas…and the pin color was set as Red.

@synthesize your mapView.

Then cut the dealloc method (at the bottom of the file and move it up to just below the @synthesize…we are going to release our mapView at the top…and then use ViewDidLoad to set the values.

I went ahead and deleted the methods we are not going to use, you can leave them or not, your choice.

If you wish to copy this code, please use the MapKit_Code.txt file. If you copy/paste from this document, it may jumble the opening and closing brackets and comments. If you want to type in the sections, please do so ☺

```
#import "MapItViewController.h"
#import "MyPosition.h"

@implementation MapItViewController
@synthesize mapView;


- (void)dealloc {
    [mapView release];
    [super dealloc];
}
```

```
// Implement viewDidLoad to do additional setup after loading the
view, typically from a nib.
- (void)viewDidLoad {
    [super viewDidLoad];
    [mapView setMapType:MKMapTypeStandard];
    [mapView setZoomEnabled:YES];
    [mapView setScrollEnabled:YES];

// Set map coordinates to 0, 0

    MKCoordinateRegion region = {{0.0, 0.0}, {0.0, 0.0}};

// Set your own coordinates for latitude and longitude

    region.center.latitude = 33.13033;
    region.center.longitude = -96.79290;

// set span and region—recommend you leave these settings alone

    region.span.longitudeDelta = 0.01f;
    region.span.latitudeDelta = 0.01f;
    [mapView setRegion:region animated:YES];
    [mapView setDelegate:self];

// Change title and subtitle to customize for your location

    MyPosition *campus=[[MyPosition alloc] init];
    campus.title=@"Preston Ridge Campus";
    campus.subtitle=@"Collin College";
    campus.coordinate=region.center;
    [mapView addAnnotation:campus];

}
// To set the pin to show—you may modify if you like, for example
you can change the pin color..

-(MKAnnotationView *)mapView:(MKMapView *)mV viewForAnnotation:
(id <MKAnnotation>)annotation{
    MKPinAnnotationView *pinView = nil;
    if(annotation != mapView.userLocation){
     static NSString *defaultPinID= @"com.invasivecode.pin";
     pinView = (MKPinAnnotationView *)[mapView
dequeueReusableAnnotationViewWithIdentifier:defaultPinID];
     if (pinView == nil) {
          pinView=[[[MKPinAnnotationView
alloc]initWithAnnotation:annotation
reuseIdentifier:defaultPinID]autorelease];
          pinView.pinColor = MKPinAnnotationColorRed;
          pinView.canShowCallout = YES;
```

```
            pinView.animatesDrop = YES;
        }
        else {
            [mapView.userLocation setTitle:@"I am here"];
        }}
        return pinView;


}

// Override to allow orientations other than the default portrait
orientation.
-
(BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientat
ion)interfaceOrientation {
    // Return YES for supported orientations
    return (interfaceOrientation ==
UIInterfaceOrientationPortrait);
}


- (void)didReceiveMemoryWarning {
    // Releases the view if it doesn't have a superview.
    [super didReceiveMemoryWarning];

    // Release any cached data, images, etc that aren't in use.
}

- (void)viewDidUnload {
    // Release any retained subviews of the main view.
    // e.g. self.myOutlet = nil;
}


@end
```

8. Now we have to go and modify both the MyPositionh and MyPosition.m files…We will start with the .h file.

Open MyPosition.h file and import the Foundation/Foundation.h framework, and the MapKit/MKAnnotation.h frameworks.

In the definition of the class add the <MKAnnotation> protocol.

Create a variable to hold a CLLocationCoordinate2D value and two NSString instance variables. You will need to @property and assign the CLLocationCoordinate2d variable, and @property and copy both of the NSString instance variables.

```
#import <UIKit/UIKit.h>
#import <Foundation/Foundation.h>
#import <MapKit/MKAnnotation.h>


@interface MyPosition : UIViewController<MKAnnotation> {
    CLLocationCoordinate2D coordinate;
    NSString *title;
    NSString *subtitle;
}
@property (nonatomic, assign) CLLocationCoordinate2D coordinate;
@property (nonatomic, copy) NSString *title;
@property (nonatomic, copy) NSString *subtitle;
@end
```

9. In the MyPosition.m file, all you need to do is synthesize the three variables, and release the two NSStrings.

```
#import "MyPosition.h"


@implementation MyPosition
@synthesize coordinate, title, subtitle;


- (void)didReceiveMemoryWarning {
    // Releases the view if it doesn't have a superview.
    [super didReceiveMemoryWarning];

    // Release any cached data, images, etc. that aren't in use.
}

- (void)viewDidUnload {
    [super viewDidUnload];
    // Release any retained subviews of the main view.
    // e.g. self.myOutlet = nil;
}


- (void)dealloc {
    [title release];
    [subtitle release];
    [super dealloc];
}


@end
```

10. The last bit of coding is to modify the AppDelegate.m file. In this file you need to adjust by removing self. from the applicationDidFinishLaunching….

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

    // Override point for customization after application launch.

    // Add the view controller's view to the window and display.
    [window addSubview:viewController.view];
    [window makeKeyAndVisible];

    return YES;
}
```

11. Save. Now one last thing before we Build. Open the MapItViewController.xib file and Control-Drag from the File's Owner to the MapView (in the View window) and select the mapView outlet. Save and close Interface Builder.

12. Save and Build. You should see a map with a location showing PRC, with a red pin on the location. If you click on the pin you should see "Preston Ridge Campus" as the title and there will also be a subtitle.

Lab 4 Requirements----------

Find a GPS location using Google Earth or other GPS tool (online maps, or even your own GPS location App) and gather the longitude and latitude (I used Google Earth app on my iPad and after you locate your chosen location on the Earth, there is a Wrench icon (Options) at the top, and when you open it, you can see Position> Select the arrow and you will see "Show Position As" and select the one in decimals. (you may be able to see this even without going into Options if your Position feature is turned on). Preston Ridge shows as 33.13150 N (latitude) and 96.79053 W (longitude). To get the "correct" longitude, you have to add a – in front of the longitude (for being west of the Prime Meridian line). So we end up with 33.13150 latitude and -96.79053 for longitude.

After finding your own location, add the latitude and longitude into this app, and change the description for title and subtitle to reflect your location. You can change the pin color too if you like, but that is optional. ☺

Zip and upload to your lab 4 dropbox.