

Developing Applications for iOS



Lab 9: Nearby Deals (5 of 6)

Radu Ionescu
raducu.ionescu@gmail.com
Faculty of Mathematics and Computer Science
University of Bucharest

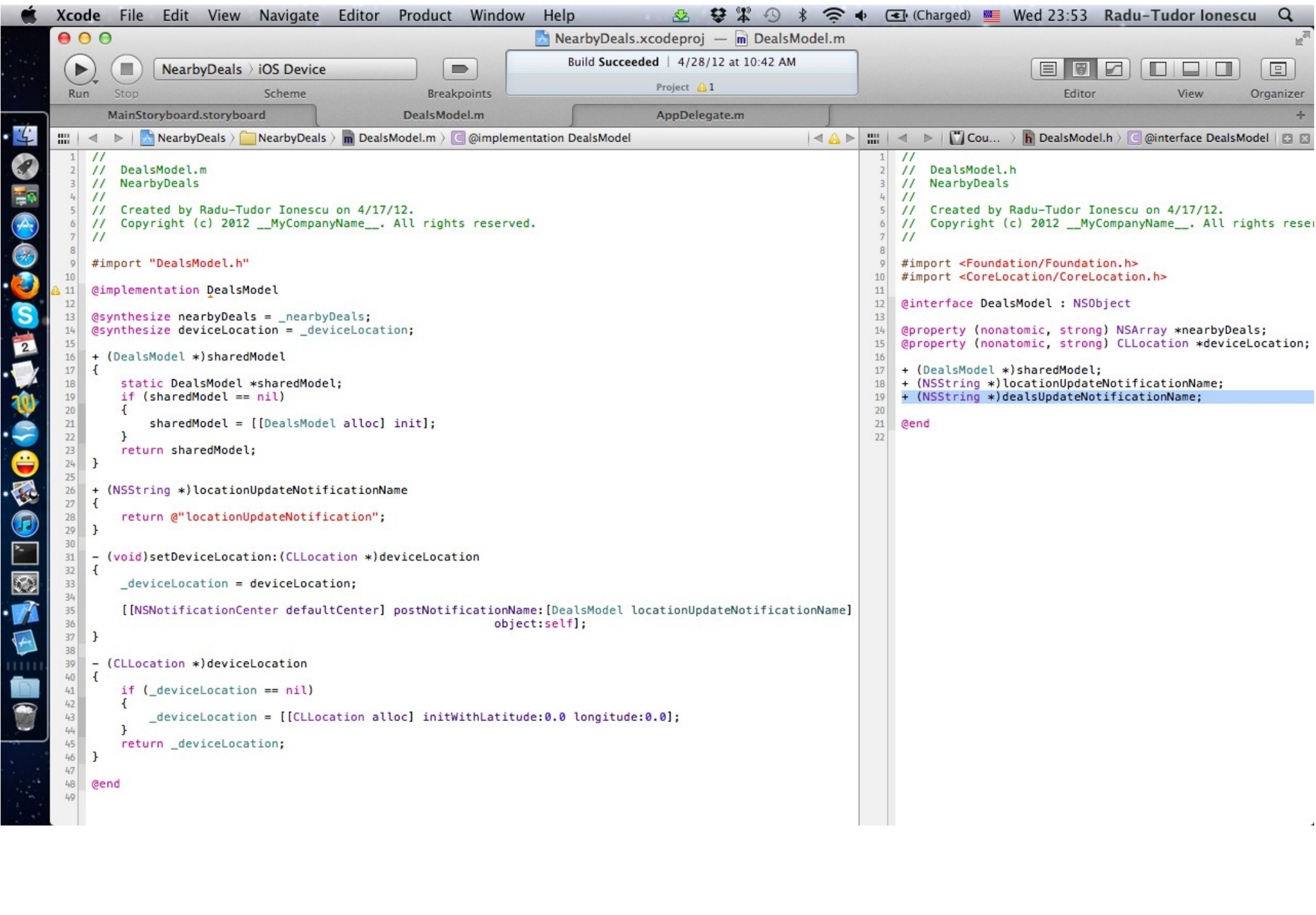
Task 1

Task: Add pins on the map for the nearby deals.

1. Launch Xcode and go to “File > Open” and select the Xcode project (.xcodeproj) inside the “NearbyDeals(4of6)” folder.
2. Run the application in iOS Simulator and take a look over the application to remember what was done last time.
3. Stop running the application.
4. In order to create annotations on the map, we need to know when the nearby deals have loaded. Thus, the `sharedModel` must post a notification inside the `nearbyDeals` setter. The Map View Controller is going to be an observer of the `sharedModel` to receive the notification. Upon receiving this notification, it will add annotations on its Map View.

Switch to the `DealsModel.m` tab in Xcode.

5. Declare a new class method for the notification name in the header file. Name it `dealsUpdateNotificationName`.



```
1 //
2 // DealsModel.m
3 // NearbyDeals
4 //
5 // Created by Radu-Tudor Ionescu on 4/17/12.
6 // Copyright (c) 2012 __MyCompanyName__. All rights reserved.
7 //
8
9 #import "DealsModel.h"
10
11 @implementation DealsModel
12
13 @synthesize nearbyDeals = _nearbyDeals;
14 @synthesize deviceLocation = _deviceLocation;
15
16 + (DealsModel *)sharedModel
17 {
18     static DealsModel *sharedModel;
19     if (sharedModel == nil)
20     {
21         sharedModel = [[DealsModel alloc] init];
22     }
23     return sharedModel;
24 }
25
26 + (NSString *)locationUpdateNotificationName
27 {
28     return @"locationUpdateNotification";
29 }
30
31 - (void)setDeviceLocation:(CLLocation *)deviceLocation
32 {
33     _deviceLocation = deviceLocation;
34
35     [[NSNotificationCenter defaultCenter] postNotificationName:[DealsModel locationUpdateNotificationName]
36                                         object:self];
37 }
38
39 - (CLLocation *)deviceLocation
40 {
41     if (_deviceLocation == nil)
42     {
43         _deviceLocation = [[CLLocation alloc] initWithLatitude:0.0 longitude:0.0];
44     }
45     return _deviceLocation;
46 }
47
48 @end
49
```

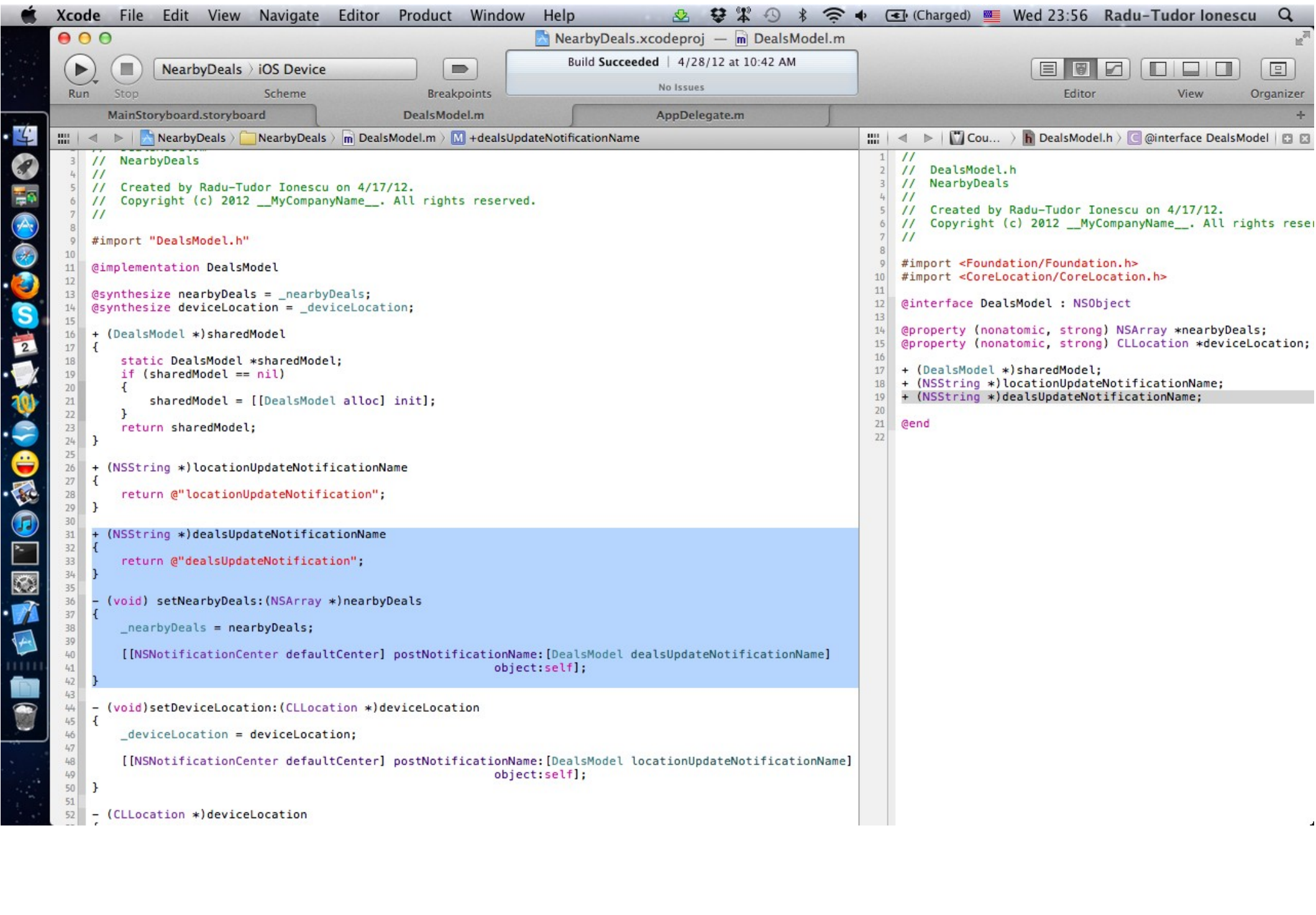
```
1 //
2 // DealsModel.h
3 // NearbyDeals
4 //
5 // Created by Radu-Tudor Ionescu on 4/17/12.
6 // Copyright (c) 2012 __MyCompanyName__. All rights reserved.
7 //
8
9 #import <Foundation/Foundation.h>
10 #import <CoreLocation/CoreLocation.h>
11
12 @interface DealsModel : NSObject
13
14 @property (nonatomic, strong) NSArray *nearbyDeals;
15 @property (nonatomic, strong) CLLocation *deviceLocation;
16
17 + (DealsModel *)sharedModel;
18 + (NSString *)locationUpdateNotificationName;
19 + (NSString *)dealUpdateNotificationName;
20
21 @end
22
```

Task 1

Task: Add pins on the map for the nearby deals.

6. Implement the `dealsUpdateNotificationName` method so that it returns the `@"dealsUpdateNotification"` string.
7. Implement the `nearbyDeals` setter to post the notification using the previously implemented class method for its name.

Look over the next slide for help.



Task 1

Task: Add pins on the map for the nearby deals.

8. Let's configure the Map View Controller to add annotations on the map when it receives the deals update notification.

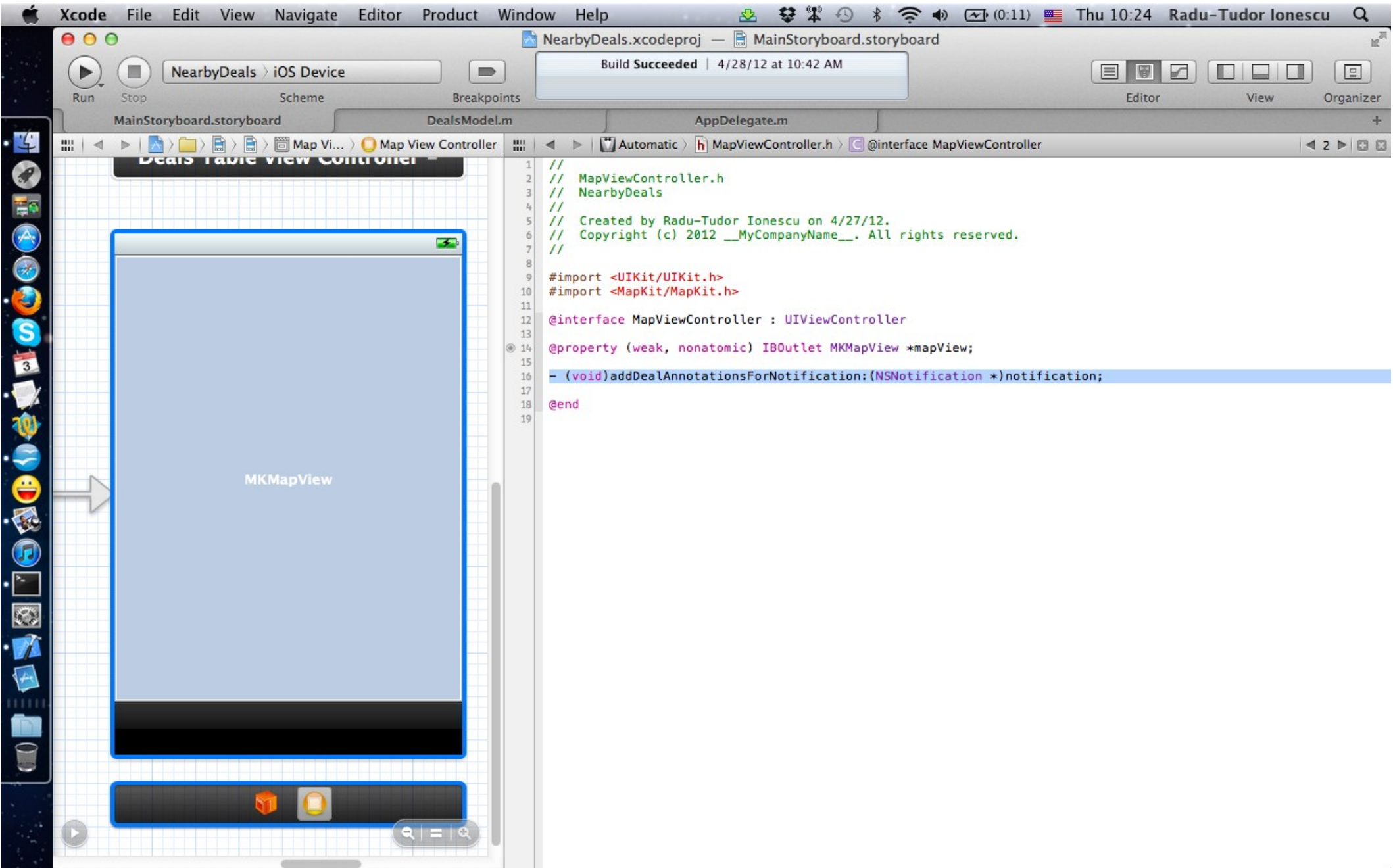
Switch to the MainStoryboard.storyboard tab in Xcode.

9. Click on the Map View Controller in Interface Builder to select its associated files in Assistant Editor.

Make sure MapViewController.h is selected.

10. Declare a method that will be executed when the Map View Controller receives the notification about deals update. This method will add annotations on the `mapView`. The method name should be `addDealAnnotationsForNotification:` and it should have a `NSNotification` argument.

The next screenshot shows how to declare this method in the `@interface`.



Build Succeeded | 4/28/12 at 10:42 AM

NearbyDeals > iOS Device

Run Stop

Scheme

Breakpoints

Editor

View

Organizer

MainStoryboard.storyboard

DealsModel.m

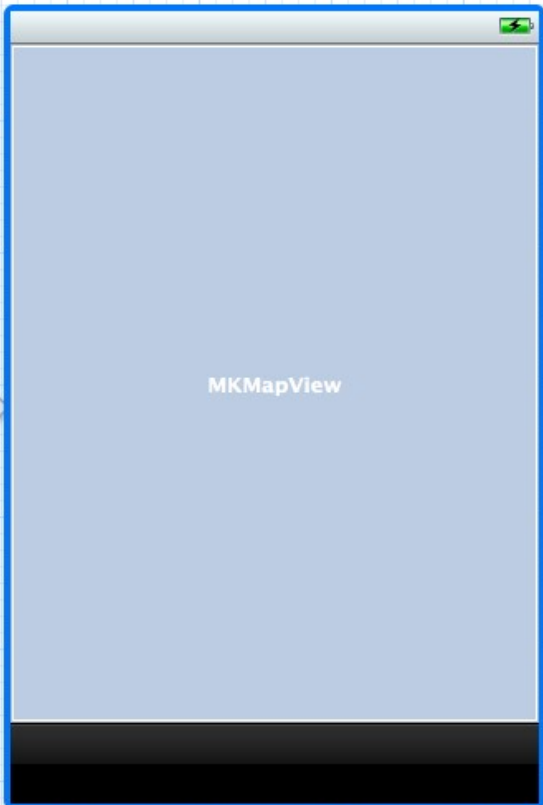
AppDelegate.m

Map Vi... > Map View Controller

Automatic > MapViewController.h > @interface MapViewController

2 > + > x

Deals Table View Controller



MKMapView

```
1 //
2 // MapViewController.h
3 // NearbyDeals
4 //
5 // Created by Radu-Tudor Ionescu on 4/27/12.
6 // Copyright (c) 2012 __MyCompanyName__. All rights reserved.
7 //
8
9 #import <UIKit/UIKit.h>
10 #import <MapKit/MapKit.h>
11
12 @interface MapViewController : UIViewController
13
14 @property (weak, nonatomic) IBOutlet MKMapView *mapView;
15
16 - (void)addDealAnnotationsForNotification:(NSNotification *)notification;
17
18 @end
19
```

Task 1

Task: Add pins on the map for the nearby deals.

11. Open the `MapViewController.m` in Assistant Editor and let's add the implementation of `addDealAnnotationsForNotification:`.
12. The first thing to do is to `#import` the `DealsModel` and `DealAnnotation` header files.
13. This method will go through the `nearbyDeals` array of the `sharedModel` using a for-in block. For each deal it will add an annotation. Remember that annotations have a `title`, a `subtitle` and a `coordinate` (represented by `latitude` and `longitude`) on the map. We obtain all this information from the deal's `NSDictionary`.

To add annotations on the map we send the `addAnnotation:` message to the `mapView`.

The next slides show how to perform these steps.

Xcode File Edit View Navigate Editor Product Window Help

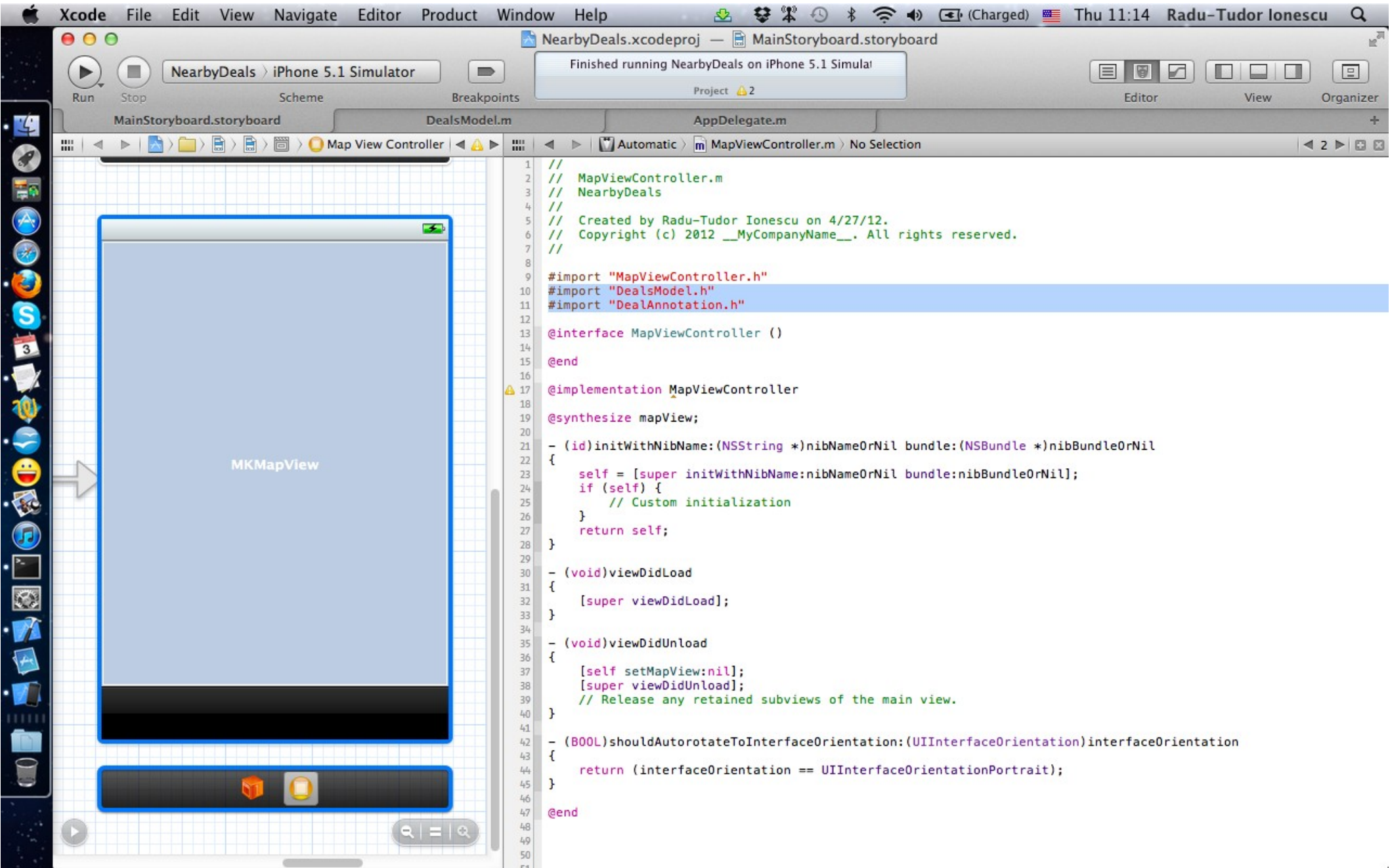
NearbyDeals.xcodeproj — MainStoryboard.storyboard

Finished running NearbyDeals on iPhone 5.1 Simulator

Run Stop Scheme Breakpoints Project 2

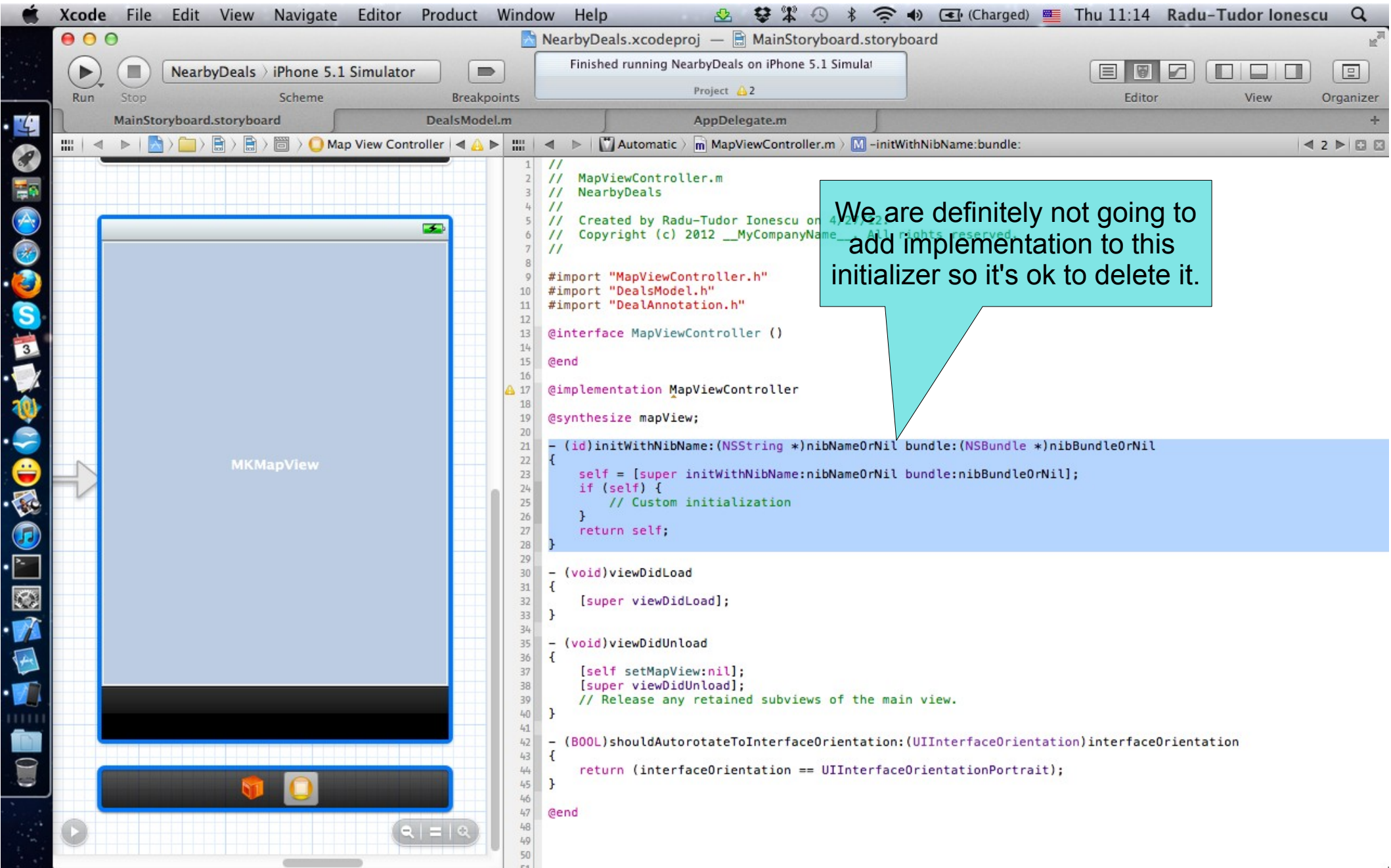
MainStoryboard.storyboard DealsModel.m AppDelegate.m

Map View Controller Automatic MapViewController.m No Selection



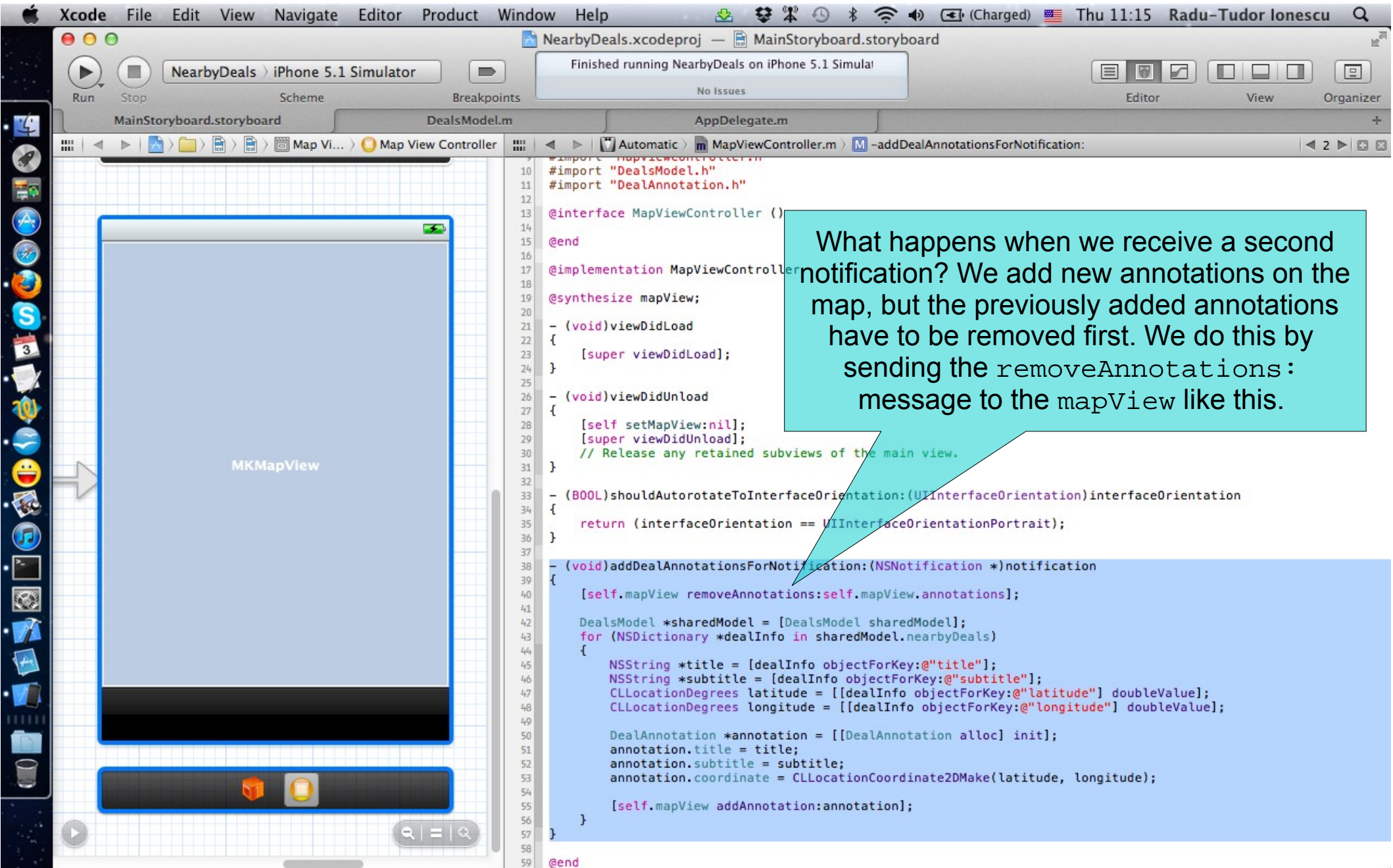
The image shows the Xcode IDE with a storyboard on the left and a code editor on the right. The storyboard displays a single view controller containing an MKMapView. The code editor shows the implementation of MapViewController.m, including imports for MapViewController.h, DealsModel.h, and DealAnnotation.h. The code defines an interface and an implementation for MapViewController, including methods for initWithNibName:bundle:, viewDidLoad, viewDidUnload, and shouldAutorotateToInterfaceOrientation:.

```
1 //
2 // MapViewController.m
3 // NearbyDeals
4 //
5 // Created by Radu-Tudor Ionescu on 4/27/12.
6 // Copyright (c) 2012 __MyCompanyName__. All rights reserved.
7 //
8
9 #import "MapViewController.h"
10 #import "DealsModel.h"
11 #import "DealAnnotation.h"
12
13 @interface MapViewController ()
14
15 @end
16
17 @implementation MapViewController
18
19 @synthesize mapView;
20
21 - (id)initWithNibName:(NSString *)nibNameOrNil bundle:(NSBundle *)nibBundleOrNil
22 {
23     self = [super initWithNibName:nibNameOrNil bundle:nibBundleOrNil];
24     if (self) {
25         // Custom initialization
26     }
27     return self;
28 }
29
30 - (void)viewDidLoad
31 {
32     [super viewDidLoad];
33 }
34
35 - (void)viewDidUnload
36 {
37     [self setMapView:nil];
38     [super viewDidUnload];
39     // Release any retained subviews of the main view.
40 }
41
42 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
43 {
44     return (interfaceOrientation == UIInterfaceOrientationPortrait);
45 }
46
47 @end
```



We are definitely not going to add implementation to this initializer so it's ok to delete it.

```
1 //
2 // MapViewController.m
3 // NearbyDeals
4 //
5 // Created by Radu-Tudor Ionescu on 4/24/12.
6 // Copyright (c) 2012 __MyCompanyName__ All rights reserved.
7 //
8
9 #import "MapViewController.h"
10 #import "DealsModel.h"
11 #import "DealAnnotation.h"
12
13 @interface MapViewController ()
14
15 @end
16
17 @implementation MapViewController
18
19 @synthesize mapView;
20
21 - (id)initWithNibName:(NSString *)nibNameOrNil bundle:(NSBundle *)nibBundleOrNil
22 {
23     self = [super initWithNibName:nibNameOrNil bundle:nibBundleOrNil];
24     if (self) {
25         // Custom initialization
26     }
27     return self;
28 }
29
30 - (void)viewDidLoad
31 {
32     [super viewDidLoad];
33 }
34
35 - (void)viewDidUnload
36 {
37     [self setMapView:nil];
38     [super viewDidUnload];
39     // Release any retained subviews of the main view.
40 }
41
42 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
43 {
44     return (interfaceOrientation == UIInterfaceOrientationPortrait);
45 }
46
47 @end
```



Task 1

Task: Add pins on the map for the nearby deals.

14. The last thing to do is add the Map View Controller as an observer of the `@"dealsUpdateNotification"`.

As for the Table View Controller, we have to send the `addObserver:selector:name:object:message` to the default `NSNotificationCenter` in the `viewDidLoad` method.

15. Remove the Map View Controller observer in `viewDidUnload`.

The next slide shows how to perform these two steps.

16. Run the application in iOS Simulator.

17. Simulate locations using the BucharestLocations GPX file.

18. Look on the map for the added annotations (note that you have to pan and zoom to find them). Tap on an annotation to see its callout.

19. Stop running the application.

Xcode interface showing the development of a map application. The left pane displays the storyboard with an MKMapView component. The right pane shows the implementation of MapViewController.m.

Storyboard: A storyboard titled "NearbyDeals" is shown in the "iPhone 5.1 Simulator" scheme. The storyboard contains a single scene with a large MKMapView component. The storyboard is titled "MainStoryboard.storyboard" and the scene is titled "Map View Controller".

Code Implementation (MapViewController.m):

```
10 #import "DealsModel.h"
11 #import "DealAnnotation.h"
12
13 @interface MapViewController ()
14
15 @end
16
17 @implementation MapViewController
18
19 @synthesize mapView = _mapView;
20
21 - (void)viewDidLoad
22 {
23     [super viewDidLoad];
24     [[NSNotificationCenter defaultCenter] addObserver:self
25                                             selector:@selector(addDealAnnotationsForNotification:)
26                                             name:[DealsModel dealsUpdateNotificationName]
27                                             object:[DealsModel sharedModel]];
28 }
29
30 - (void)viewDidUnload
31 {
32     [self setMapView:nil];
33     [super viewDidUnload];
34
35     [[NSNotificationCenter defaultCenter] removeObserver:self];
36 }
37
38 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
39 {
40     return (interfaceOrientation == UIInterfaceOrientationPortrait);
41 }
42
43 - (void)addDealAnnotationsForNotification:(NSNotification *)notification
44 {
45     [self.mapView removeAnnotations:self.mapView.annotations];
46
47     DealsModel *sharedModel = [DealsModel sharedModel];
48     for (NSDictionary *dealInfo in sharedModel.nearbyDeals)
49     {
50         NSString *title = [dealInfo objectForKey:@"title"];
51         NSString *subtitle = [dealInfo objectForKey:@"subtitle"];
52         CLLocationDegrees latitude = [[dealInfo objectForKey:@"latitude"] doubleValue];
53         CLLocationDegrees longitude = [[dealInfo objectForKey:@"longitude"] doubleValue];
54
55         DealAnnotation *annotation = [[DealAnnotation alloc] init];
56         annotation.title = title;
57         annotation.subtitle = subtitle;
58         annotation.coordinate = CLLocationCoordinate2DMake(latitude, longitude);
59
60         [self.mapView addAnnotation:annotation];
61     }
62 }
```

Task 1

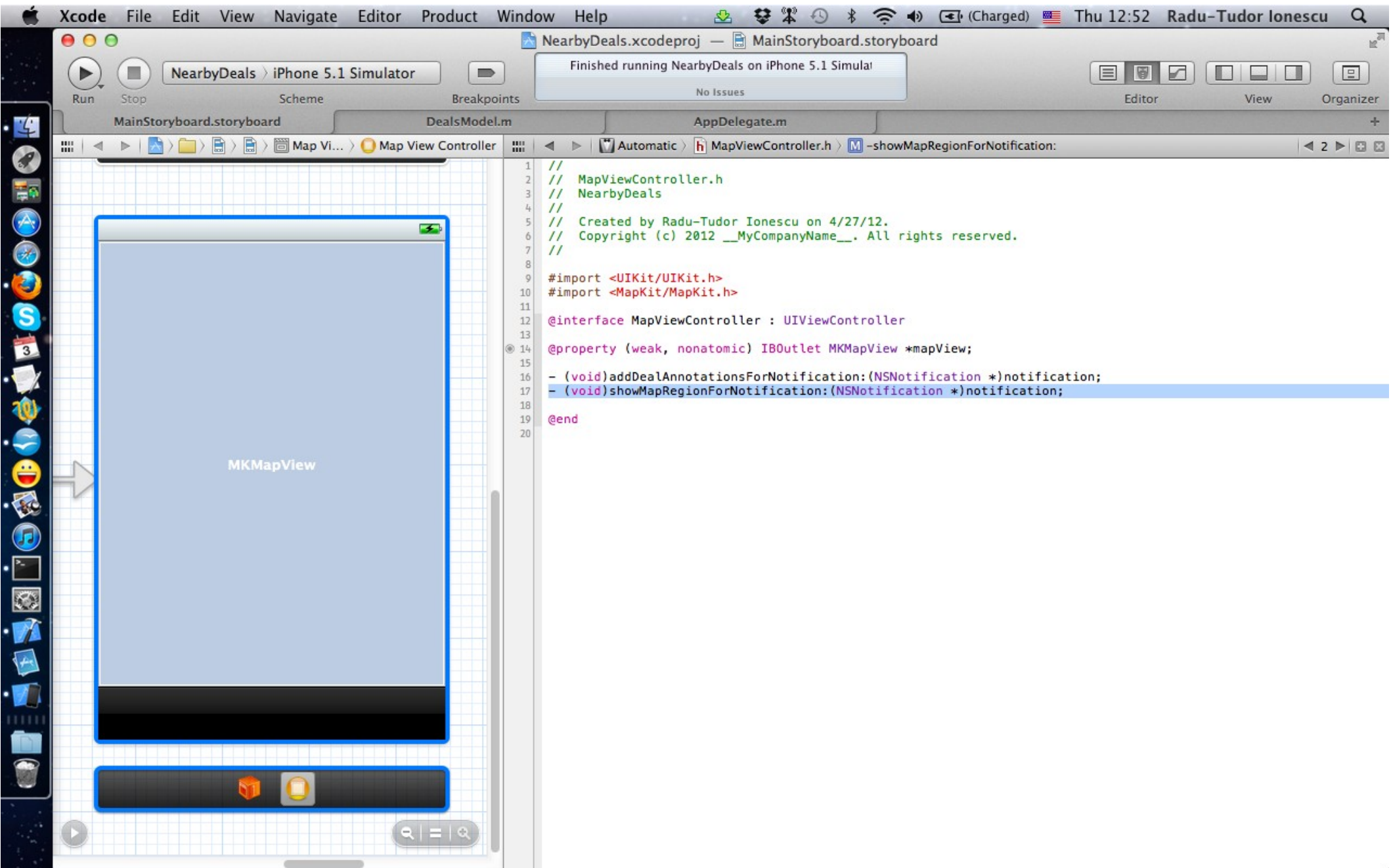
Task: Add pins on the map for the nearby deals.

20. The user shouldn't do any extra effort to find the pins on the map. It would be nice to help the user by centering and zooming the map on current user location. We can do this automatically when the `deviceLocation` gets updated. The Map View Controller must observe the `sharedModel` for location update notifications.

Switch to the `MapViewController` header file in Assistant Editor.

21. Declare a method that will zoom the map on the current user location upon receiving a location update notification. Name this method `showMapRegionForNotification:`.

Look over the next slide to see how to declare it.



Task 1

Task: Add pins on the map for the nearby deals.

22. Set the Map View Controller as an observer for the `sharedModel` object to register for the `@"locationUpdateNotification"`.

Switch to the `MapViewController` implementation file in Assistant Editor.

23. As soon as the `viewDidLoad`s, we can register the observer by sending the `addObserver:selector:name:object: message` to the default `NSNotificationCenter`.

24. Implement the `showMapRegionForNotification:` method to set the visible region of the map on current user location. The region's span should be around 10 km.

To set `mapView`'s region send the `setRegion:animated:`. Use animation only if the Map View Controller is currently on screen (check the `self.view.hidden` property).

Look over the next slides for help.

Xcode interface showing the development of a Map View Controller for an iPhone 5.1 Simulator. The interface includes a toolbar with Run, Stop, Scheme, and Breakpoints buttons. The top status bar displays the project name "NearbyDeals.xcodeproj" and the current storyboard "MainStoryboard.storyboard".

The left pane shows the storyboard with a single view controller named "Map View Controller" containing an "MKMapView" widget. The right pane displays the corresponding Objective-C code for "MapViewController.m".

```
13 @interface MapViewController ()
14
15 @end
16
17 @implementation MapViewController
18
19 @synthesize mapView = _mapView;
20
21 - (void)viewDidLoad
22 {
23     [super viewDidLoad];
24     [[NSNotificationCenter defaultCenter] addObserver:self
25                                             selector:@selector(addDealAnnotationsForNotification:)
26                                             name:[DealsModel dealsUpdateNotificationName]
27                                             object:[DealsModel sharedModel]];
28
29     [[NSNotificationCenter defaultCenter] addObserver:self
30                                             selector:@selector(showMapRegionForNotification:)
31                                             name:[DealsModel locationUpdateNotificationName]
32                                             object:[DealsModel sharedModel]];
33 }
34
35 - (void)viewDidUnload
36 {
37     [self setMapView:nil];
38     [super viewDidUnload];
39
40     [[NSNotificationCenter defaultCenter] removeObserver:self];
41 }
42
43 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
44 {
45     return (interfaceOrientation == UIInterfaceOrientationPortrait);
46 }
47
48 - (void)addDealAnnotationsForNotification:(NSNotification *)notification
49 {
50     [self.mapView removeAnnotations:self.mapView.annotations];
51
52     DealsModel *sharedModel = [DealsModel sharedModel];
53     for (NSDictionary *dealInfo in sharedModel.nearbyDeals)
54     {
55         NSString *title = [dealInfo objectForKey:@"title"];
56         NSString *subtitle = [dealInfo objectForKey:@"subtitle"];
57         CLLocationDegrees latitude = [[dealInfo objectForKey:@"latitude"] doubleValue];
58         CLLocationDegrees longitude = [[dealInfo objectForKey:@"longitude"] doubleValue];
59
60         DealAnnotation *annotation = [[DealAnnotation alloc] init];
61         annotation.title = title;
62         annotation.subtitle = subtitle;
63         annotation.coordinate = CLLocationCoordinate2DMake(latitude, longitude);
```

The image shows the Xcode IDE with an iPhone 5.1 Simulator on the left and the source code for `MapViewController.m` on the right. The simulator displays an `MKMapView`. The code includes the following methods:

```
30     selector:@selector(showMapRegionForNotification:)
31     name:[DealsModel locationUpdateNotificationName]
32     object:[DealsModel sharedModel]];
33 }
34
35 - (void)viewDidUnload
36 {
37     [self setMapView:nil];
38     [super viewDidUnload];
39
40     [[NSNotificationCenter defaultCenter] removeObserver:self];
41 }
42
43 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
44 {
45     return (interfaceOrientation == UIInterfaceOrientationPortrait);
46 }
47
48 - (void)addDealAnnotationsForNotification:(NSNotification *)notification
49 {
50     [self.mapView removeAnnotations:self.mapView.annotations];
51
52     DealsModel *sharedModel = [DealsModel sharedModel];
53     for (NSDictionary *dealInfo in sharedModel.nearbyDeals)
54     {
55         NSString *title = [dealInfo objectForKey:@"title"];
56         NSString *subtitle = [dealInfo objectForKey:@"subtitle"];
57         CLLocationCoordinate2D latitude = [[dealInfo objectForKey:@"latitude"] doubleValue];
58         CLLocationCoordinate2D longitude = [[dealInfo objectForKey:@"longitude"] doubleValue];
59
60         DealAnnotation *annotation = [[DealAnnotation alloc] initWithTitle:title subtitle:subtitle latitude:latitude longitude:longitude];
61         annotation.title = title;
62         annotation.subtitle = subtitle;
63         annotation.coordinate = CLLocationCoordinate2DMake(latitude, longitude);
64
65         [self.mapView addAnnotation:annotation];
66     }
67 }
68
69 - (void)showMapRegionForNotification:(NSNotification *)notification
70 {
71     DealsModel *sharedModel = [DealsModel sharedModel];
72
73     CLLocationCoordinate2D coordinate = sharedModel.deviceLocation.coordinate;
74     MKCoordinateRegion visibleRegion = MKCoordinateRegionMakeWithDistance(coordinate, 10000, 10000);
75
76     [self.mapView setRegion:visibleRegion animated:!self.view.hidden];
77 }
78
79 @end
80
```

We set the `visibleRegion` to be 10 km (10000 meters) around the current user location coordinates.

The map will animate to this region only when the `self.view` is not hidden.

Task 1

Task: Add pins on the map for the nearby deals.

25. Run the application in iOS Simulator.
26. Go on the Map View tab and simulate locations using the BucharestLocations GPX file.
27. Notice how the visible region of the map follows the user current location. Stop simulating location updates.
28. Notice the user's location is no longer displayed on the map. This happens because we remove all annotations from the map when the Map View Controller receives the nearby deals update notification.

The user location pin is also an annotation and it shouldn't be removed. There is an easy fix for this: stop showing the user location before removing the pins, then start showing it again after the other pins have been removed. The `showsUserLocation` BOOL property of the `mapView` controls this.

Look over the next slide for hints.

Xcode File Edit View Navigate Editor Product Window Help

NearbyDeals.xcodeproj — MainStoryboard.storyboard

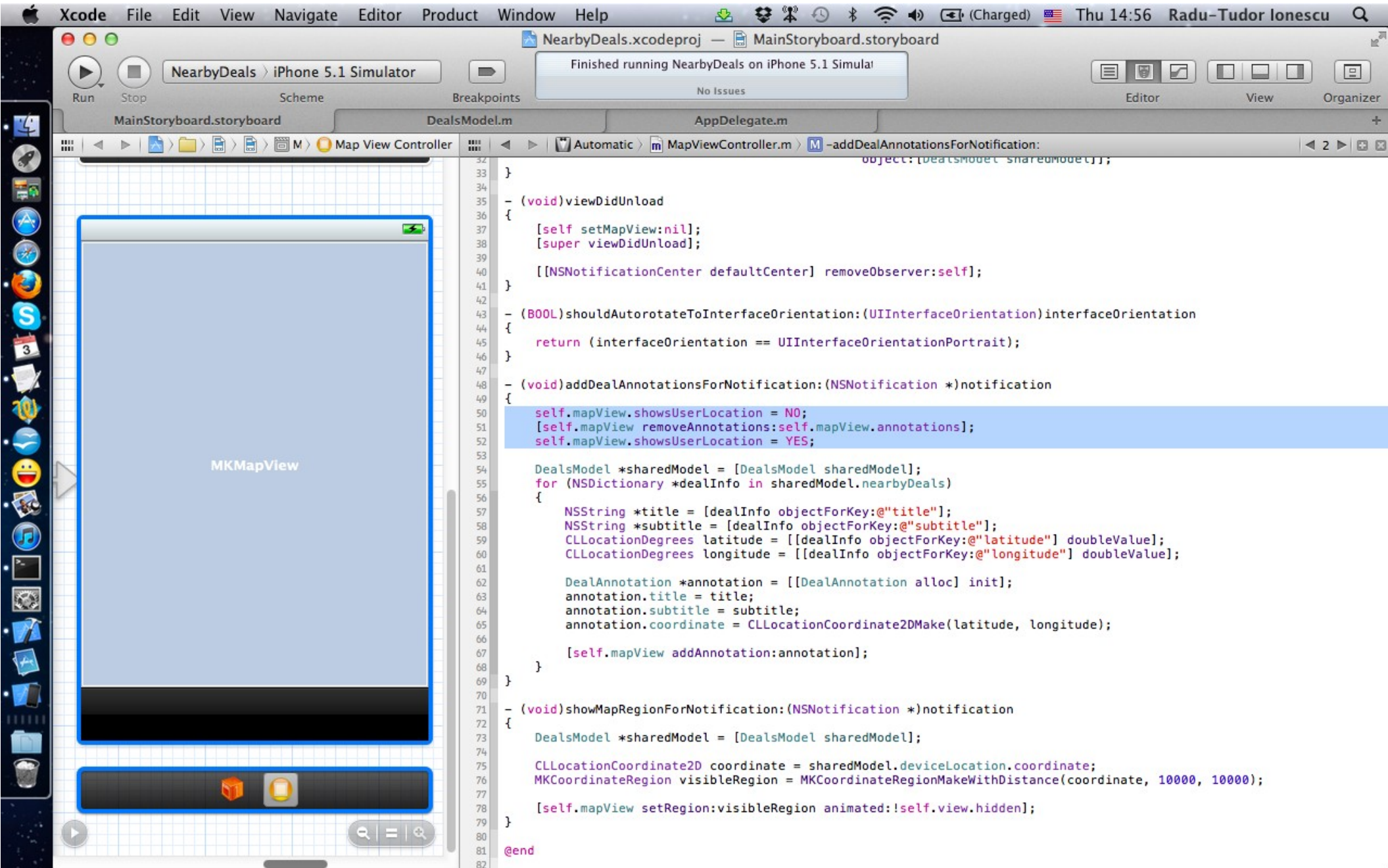
Finished running NearbyDeals on iPhone 5.1 Simulator

No Issues

Run Stop Scheme Breakpoints Editor View Organizer

MainStoryboard.storyboard DealsModel.m AppDelegate.m

Map View Controller Automatic MapViewController.m -addDealAnnotationsForNotification:



```
32 }
33 }
34
35 - (void)viewDidLoad
36 {
37     [self setMapView:nil];
38     [super viewDidLoad];
39
40     [[NSNotificationCenter defaultCenter] removeObserver:self];
41 }
42
43 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
44 {
45     return (interfaceOrientation == UIInterfaceOrientationPortrait);
46 }
47
48 - (void)addDealAnnotationsForNotification:(NSNotification *)notification
49 {
50     self.mapView.showsUserLocation = NO;
51     [self.mapView removeAnnotations:self.mapView.annotations];
52     self.mapView.showsUserLocation = YES;
53
54     DealsModel *sharedModel = [DealsModel sharedModel];
55     for (NSDictionary *dealInfo in sharedModel.nearbyDeals)
56     {
57         NSString *title = [dealInfo objectForKey:@"title"];
58         NSString *subtitle = [dealInfo objectForKey:@"subtitle"];
59         CLLocationDegrees latitude = [[dealInfo objectForKey:@"latitude"] doubleValue];
60         CLLocationDegrees longitude = [[dealInfo objectForKey:@"longitude"] doubleValue];
61
62         DealAnnotation *annotation = [[DealAnnotation alloc] init];
63         annotation.title = title;
64         annotation.subtitle = subtitle;
65         annotation.coordinate = CLLocationCoordinate2DMake(latitude, longitude);
66
67         [self.mapView addAnnotation:annotation];
68     }
69 }
70
71 - (void)showMapRegionForNotification:(NSNotification *)notification
72 {
73     DealsModel *sharedModel = [DealsModel sharedModel];
74
75     CLLocationCoordinate2D coordinate = sharedModel.deviceLocation.coordinate;
76     MKCoordinateRegion visibleRegion = MKCoordinateRegionMakeWithDistance(coordinate, 10000, 10000);
77
78     [self.mapView setRegion:visibleRegion animated:!self.view.hidden];
79 }
80
81 @end
82
```

Task 1

Task: Add pins on the map for the nearby deals.

29. When the application starts it will load only the Table View Controller. The Map View Controller will load later when the user goes on the second tab of the application. If the `sharedModel` sends notifications before the Map View Controller is loaded (and its `viewDidLoad` gets executed) it will never receive those notifications. Thus the Map View Controller will not display any pins, even if the nearby deals have loaded.

We have to add implementation to the `viewDidLoad` method to create annotations for the current nearby deals (if any). We will also set the visible region of the map on the current user location if it's different from the initial (0,0) coordinates.

Look over the next slide for hints.

Xcode File Edit View Navigate Editor Product Window Help

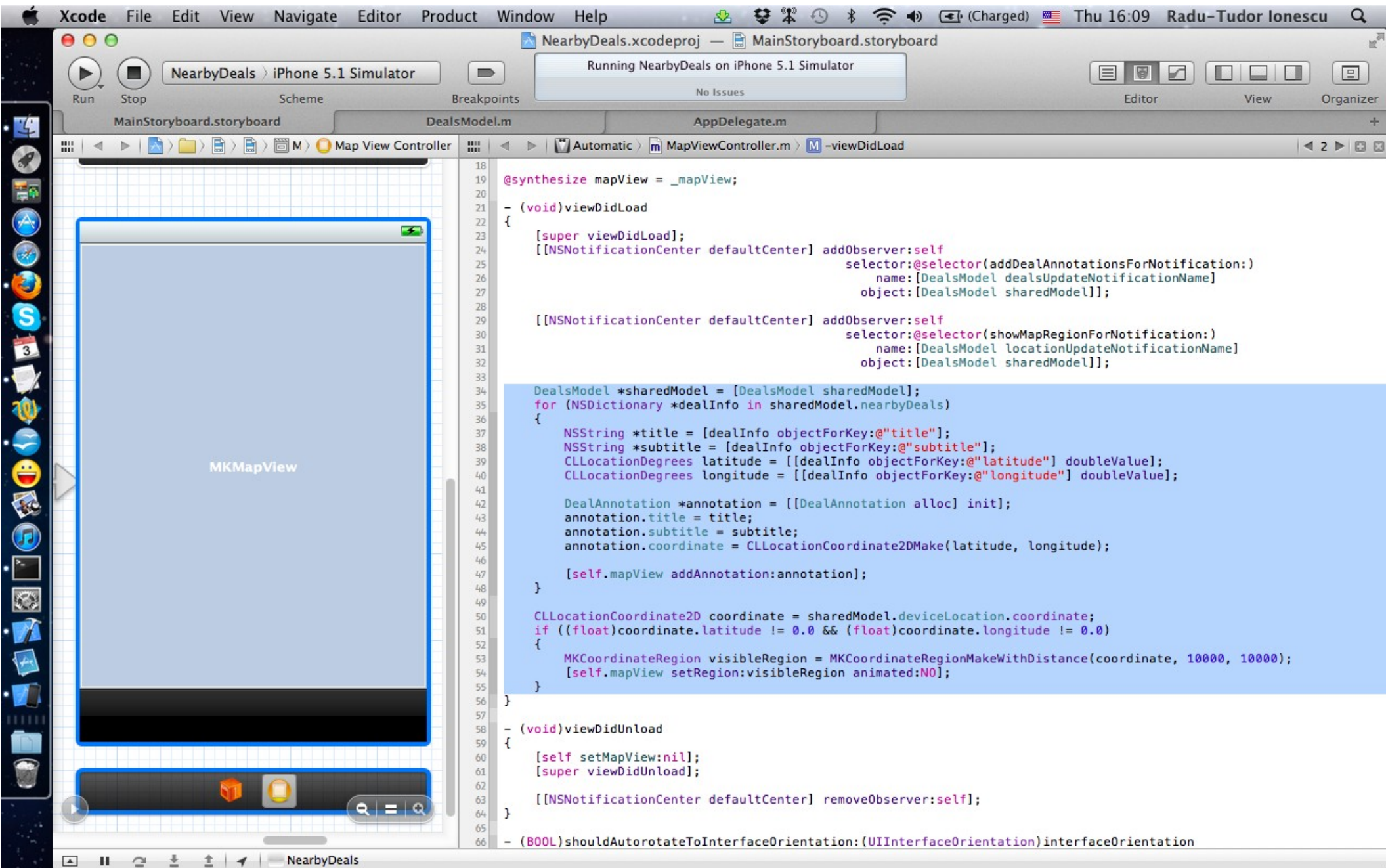
NearbyDeals.xcodeproj — MainStoryboard.storyboard

Running NearbyDeals on iPhone 5.1 Simulator

No Issues

MainStoryboard.storyboard DealsModel.m AppDelegate.m

Map View Controller Automatic MapViewController.m -viewDidLoad



The screenshot shows the Xcode IDE with an iPhone 5.1 simulator on the left and a code editor on the right. The simulator displays a large blue rectangle labeled 'MKMapView'. The code editor shows the implementation of the `-viewDidLoad` method in `MapViewController.m`. The code initializes a map view, registers observers for notifications from the `DealsModel`, and iterates through the `sharedModel.nearbyDeals` array to create and add annotations to the map. It also sets a visible region for the map based on the device location.

```
18
19 @synthesize mapView = _mapView;
20
21 - (void)viewDidLoad
22 {
23     [super viewDidLoad];
24     [[NSNotificationCenter defaultCenter] addObserver:self
25                                             selector:@selector(addDealAnnotationsForNotification:)
26                                             name:[DealsModel dealsUpdateNotificationName]
27                                             object:[DealsModel sharedModel]];
28
29     [[NSNotificationCenter defaultCenter] addObserver:self
30                                             selector:@selector(showMapRegionForNotification:)
31                                             name:[DealsModel locationUpdateNotificationName]
32                                             object:[DealsModel sharedModel]];
33
34     DealsModel *sharedModel = [DealsModel sharedModel];
35     for (NSDictionary *dealInfo in sharedModel.nearbyDeals)
36     {
37         NSString *title = [dealInfo objectForKey:@"title"];
38         NSString *subtitle = [dealInfo objectForKey:@"subtitle"];
39         CLLocationDegrees latitude = [[dealInfo objectForKey:@"latitude"] doubleValue];
40         CLLocationDegrees longitude = [[dealInfo objectForKey:@"longitude"] doubleValue];
41
42         DealAnnotation *annotation = [[DealAnnotation alloc] init];
43         annotation.title = title;
44         annotation.subtitle = subtitle;
45         annotation.coordinate = CLLocationCoordinate2DMake(latitude, longitude);
46
47         [self.mapView addAnnotation:annotation];
48     }
49
50     CLLocationCoordinate2D coordinate = sharedModel.deviceLocation.coordinate;
51     if ((float)coordinate.latitude != 0.0 && (float)coordinate.longitude != 0.0)
52     {
53         MKCoordinateRegion visibleRegion = MKCoordinateRegionMakeWithDistance(coordinate, 10000, 10000);
54         [self.mapView setRegion:visibleRegion animated:NO];
55     }
56 }
57
58 - (void)viewDidUnload
59 {
60     [self setMapView:nil];
61     [super viewDidUnload];
62
63     [[NSNotificationCenter defaultCenter] removeObserver:self];
64 }
65
66 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
```

Task 1

Task: Add pins on the map for the nearby deals.

30. Run the application in iOS Simulator.
31. Simulate locations using the BucharestLocations GPX file.
32. Wait until you get nearby deals, then stop simulating location updates.
33. Navigate to the second tab of the application. It should center and zoom the map right from the beginning.

Everything should run smooth now.

Task 2

Task: Add functionality to access deal details by selecting a pin from the map.

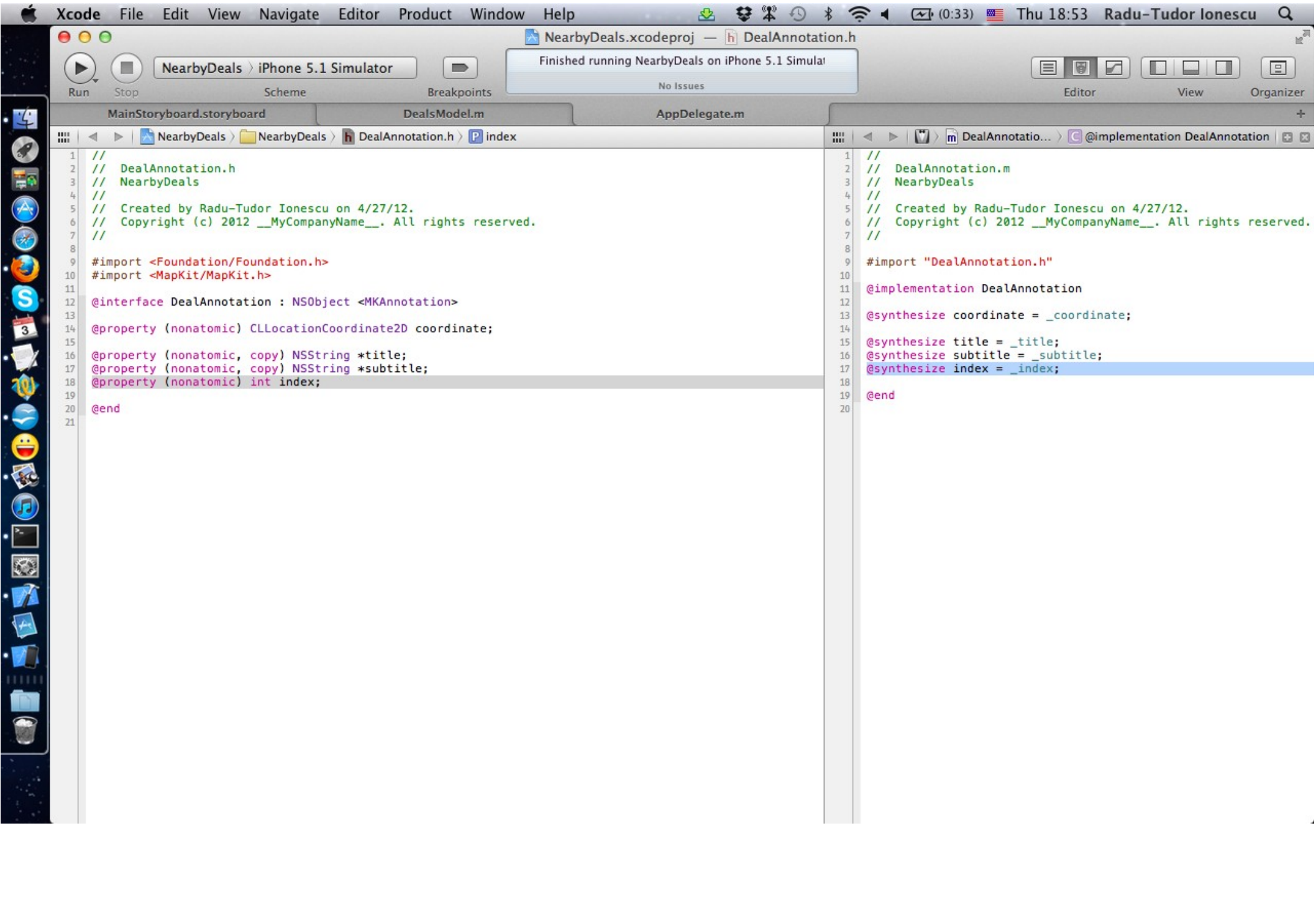
1. We want to be able to click on pins and see detailed information about the associated deal in a new View Controller. Thus it is necessary that each annotation knows about its deal.

We will add a new `@property` to the `DealAnnotation` class. This property will be an `int` that will store the index of the deal in the `nearbyDeals` `NSArray` object.

Switch to the `AppDelegate.m` tab in Xcode.

2. Select the `DealAnnotation.m` file on the left side of the Editor. The `DealAnnotation.h` header should appear in Assistant Editor (on the right side).
3. Declare and `@synthesize` a `nonatomic int` property named `index`. Prefix its instance variable with underscore.

The next slide gives you a hint about this step.



Task 2

Task: Add functionality to access deal details by selecting a pin from the map.

4. Switch back to the MainStoryboard.storyboard tab in Xcode. Make sure the MapViewController.m is selected in Assistant Editor.
5. Let's use the `index @property` of each annotation to store the index of the associated deals. To obtain the index of deal's `NSDictionary` object we send the `indexOfObject:` message to the `nearbyDeals` array.

Note that we have to modify the implementation of the `viewDidLoad` and `addDealAnnotationsForNotification:` methods.

The next slides show how the two methods should be modified.

Also note that the code duplication will be resolved later in your first assignment.

Xcode File Edit View Navigate Editor Product Window Help

NearbyDeals.xcodeproj — MainStoryboard.storyboard

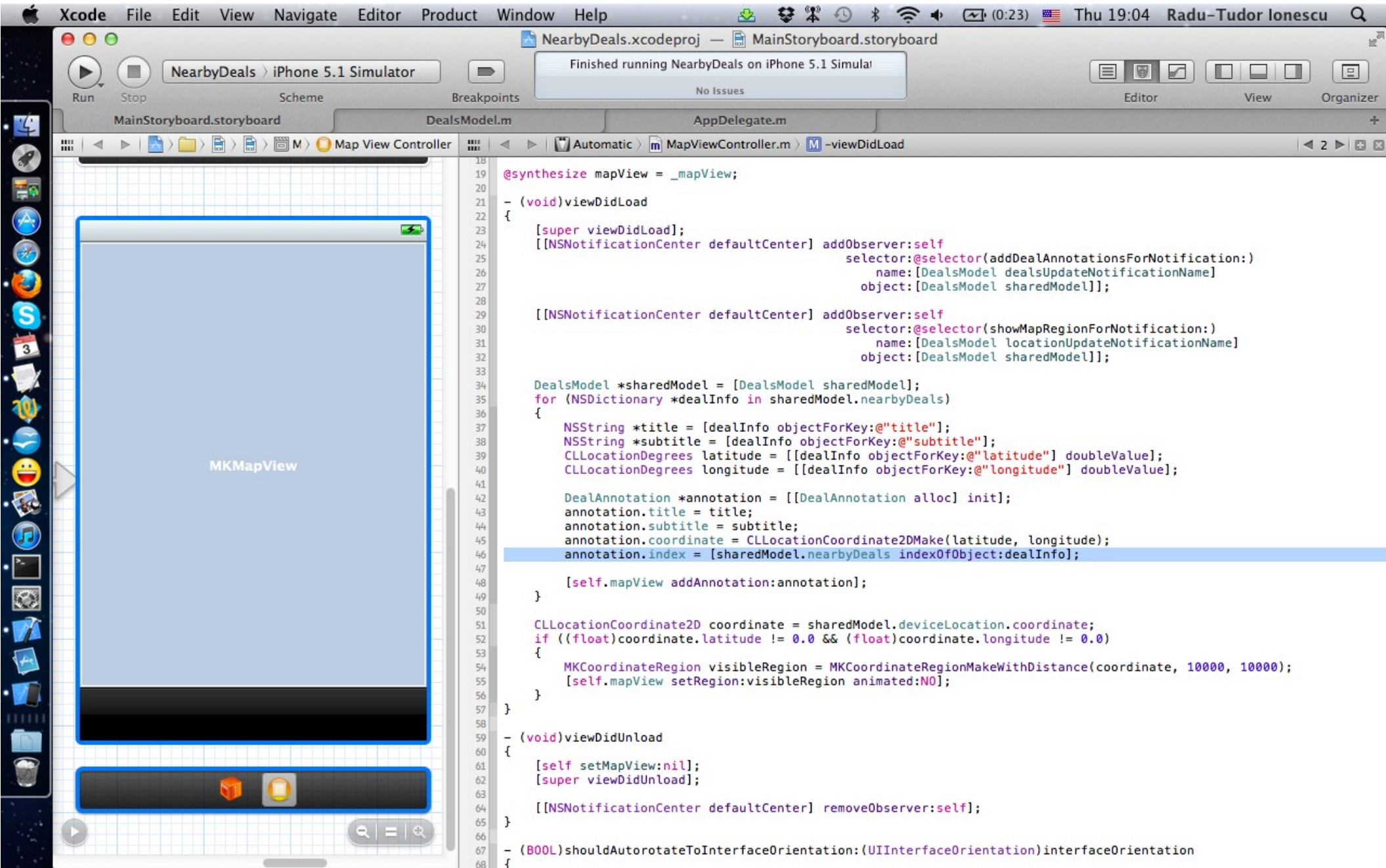
Finished running NearbyDeals on iPhone 5.1 Simulator

No Issues

Run Stop Scheme Breakpoints Editor View Organizer

MainStoryboard.storyboard DealsModel.m AppDelegate.m

Map View Controller Automatic MapViewController.m -viewDidLoad



```
18
19 @synthesize mapView = _mapView;
20
21 - (void)viewDidLoad
22 {
23     [super viewDidLoad];
24     [[NSNotificationCenter defaultCenter] addObserver:self
25                                             selector:@selector(addDealAnnotationsForNotification:)
26                                             name:[DealsModel dealsUpdateNotificationName]
27                                             object:[DealsModel sharedModel]];
28
29     [[NSNotificationCenter defaultCenter] addObserver:self
30                                             selector:@selector(showMapRegionForNotification:)
31                                             name:[DealsModel locationUpdateNotificationName]
32                                             object:[DealsModel sharedModel]];
33
34     DealsModel *sharedModel = [DealsModel sharedModel];
35     for (NSDictionary *dealInfo in sharedModel.nearbyDeals)
36     {
37         NSString *title = [dealInfo objectForKey:@"title"];
38         NSString *subtitle = [dealInfo objectForKey:@"subtitle"];
39         CLLocationDegrees latitude = [[dealInfo objectForKey:@"latitude"] doubleValue];
40         CLLocationDegrees longitude = [[dealInfo objectForKey:@"longitude"] doubleValue];
41
42         DealAnnotation *annotation = [[DealAnnotation alloc] init];
43         annotation.title = title;
44         annotation.subtitle = subtitle;
45         annotation.coordinate = CLLocationCoordinate2DMake(latitude, longitude);
46         annotation.index = [sharedModel.nearbyDeals indexOfObject:dealInfo];
47
48         [self.mapView addAnnotation:annotation];
49     }
50
51     CLLocationCoordinate2D coordinate = sharedModel.deviceLocation.coordinate;
52     if ((float)coordinate.latitude != 0.0 && (float)coordinate.longitude != 0.0)
53     {
54         MKCoordinateRegion visibleRegion = MKCoordinateRegionMakeWithDistance(coordinate, 10000, 10000);
55         [self.mapView setRegion:visibleRegion animated:NO];
56     }
57 }
58
59 - (void)viewDidUnload
60 {
61     [self setMapView:nil];
62     [super viewDidUnload];
63
64     [[NSNotificationCenter defaultCenter] removeObserver:self];
65 }
66
67 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
68 {
```

Xcode File Edit View Navigate Editor Product Window Help

NearbyDeals.xcodeproj — MainStoryboard.storyboard

Finished running NearbyDeals on iPhone 5.1 Simulator

No Issues

Run Stop Scheme Breakpoints Editor View Organizer

MainStoryboard.storyboard DealsModel.m AppDelegate.m

Map View Controller

Automatic MapViewController.m -addDealAnnotationsForNotification:

MKMapView

```
50 }
51 }
52 }
53 }
54 }
55 }
56 }
57 }
58 }
59 - (void)viewDidLoad
60 {
61     [self setMapView:nil];
62     [super viewDidLoad];
63
64     [[NSNotificationCenter defaultCenter] removeObserver:self];
65 }
66
67 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
68 {
69     return (interfaceOrientation == UIInterfaceOrientationPortrait);
70 }
71
72 - (void)addDealAnnotationsForNotification:(NSNotification *)notification
73 {
74     self.mapView.showsUserLocation = NO;
75     [self.mapView removeAnnotations:self.mapView.annotations];
76     self.mapView.showsUserLocation = YES;
77
78     DealsModel *sharedModel = [DealsModel sharedModel];
79     for (NSDictionary *dealInfo in sharedModel.nearbyDeals)
80     {
81         NSString *title = [dealInfo objectForKey:@"title"];
82         NSString *subtitle = [dealInfo objectForKey:@"subtitle"];
83         CLLocationDegrees latitude = [[dealInfo objectForKey:@"latitude"] doubleValue];
84         CLLocationDegrees longitude = [[dealInfo objectForKey:@"longitude"] doubleValue];
85
86         DealAnnotation *annotation = [[DealAnnotation alloc] init];
87         annotation.title = title;
88         annotation.subtitle = subtitle;
89         annotation.coordinate = CLLocationCoordinate2DMake(latitude, longitude);
90         annotation.index = [sharedModel.nearbyDeals indexOfObject:dealInfo];
91
92         [self.mapView addAnnotation:annotation];
93     }
94 }
95
96 - (void)showMapRegionForNotification:(NSNotification *)notification
97 {
98     DealsModel *sharedModel = [DealsModel sharedModel];
99
100     CLLocationCoordinate2D coordinate = sharedModel.deviceLocation.coordinate;
101     MKCoordinateRegion visibleRegion = MKCoordinateRegionMakeWithDistance(coordinate, 10000, 10000);
102
103     [self.mapView setRegion:visibleRegion animated:!self.view.hidden];
104 }
105
106 @end
```

Task 2

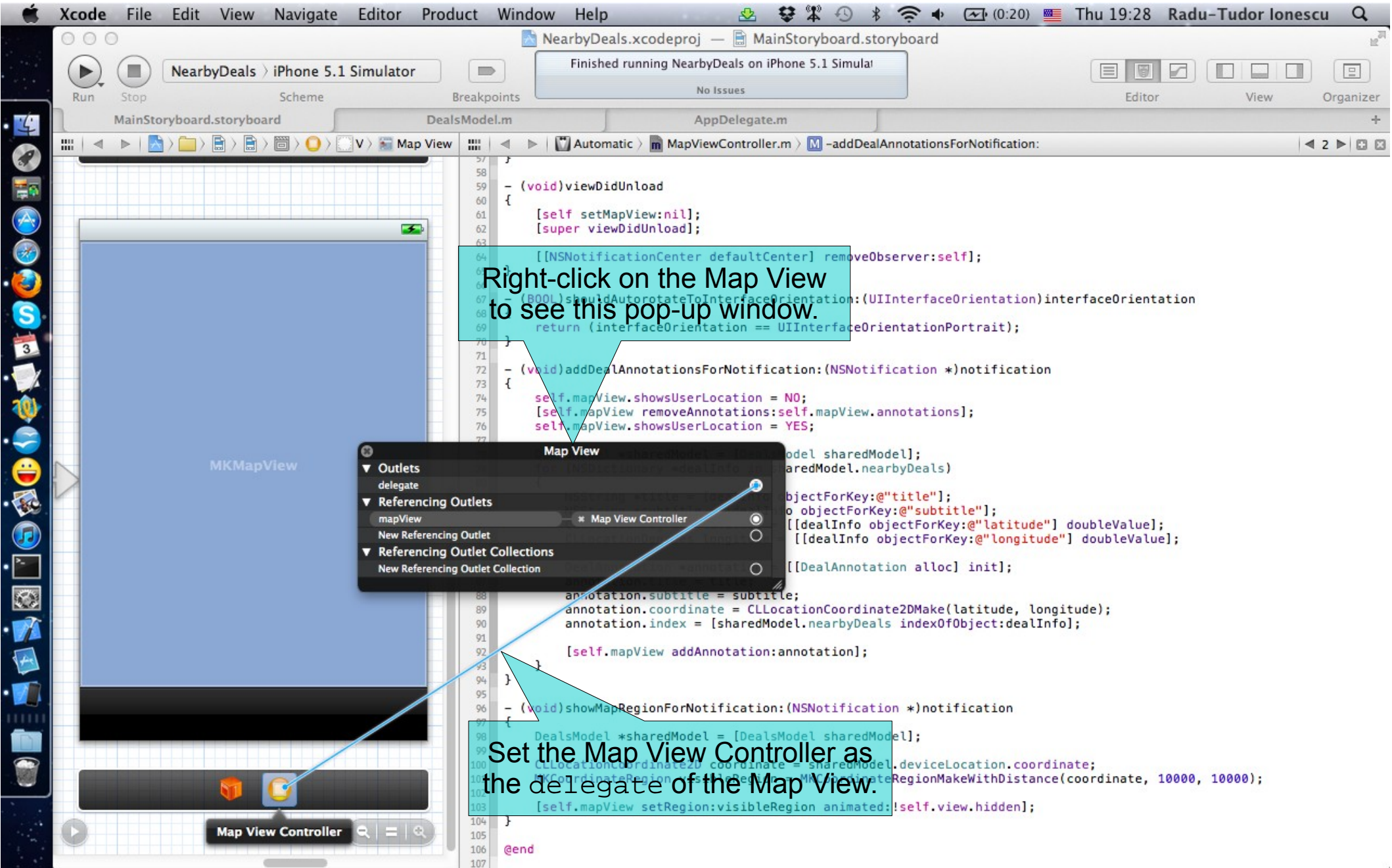
Task: Add functionality to access deal details by selecting a pin from the map.

4. We will customize the callout view of the annotation with left and right accessory views. Next, we are going to focus on adding a detail disclosure `UIButton` as the `rightCalloutAccessoryView`. We do this in the `mapView:viewForAnnotation:delegate` method.

The Map View Controller needs to adopt the `MKMapViewDelegate` protocol.

Select the Map View Controller header file in Assistant Editor.

5. Add the `MKMapViewDelegate` protocol after the superclass declaration.
6. Return to the Map View Controller implementation file in Assistant Editor and continue with the steps from the following slides.

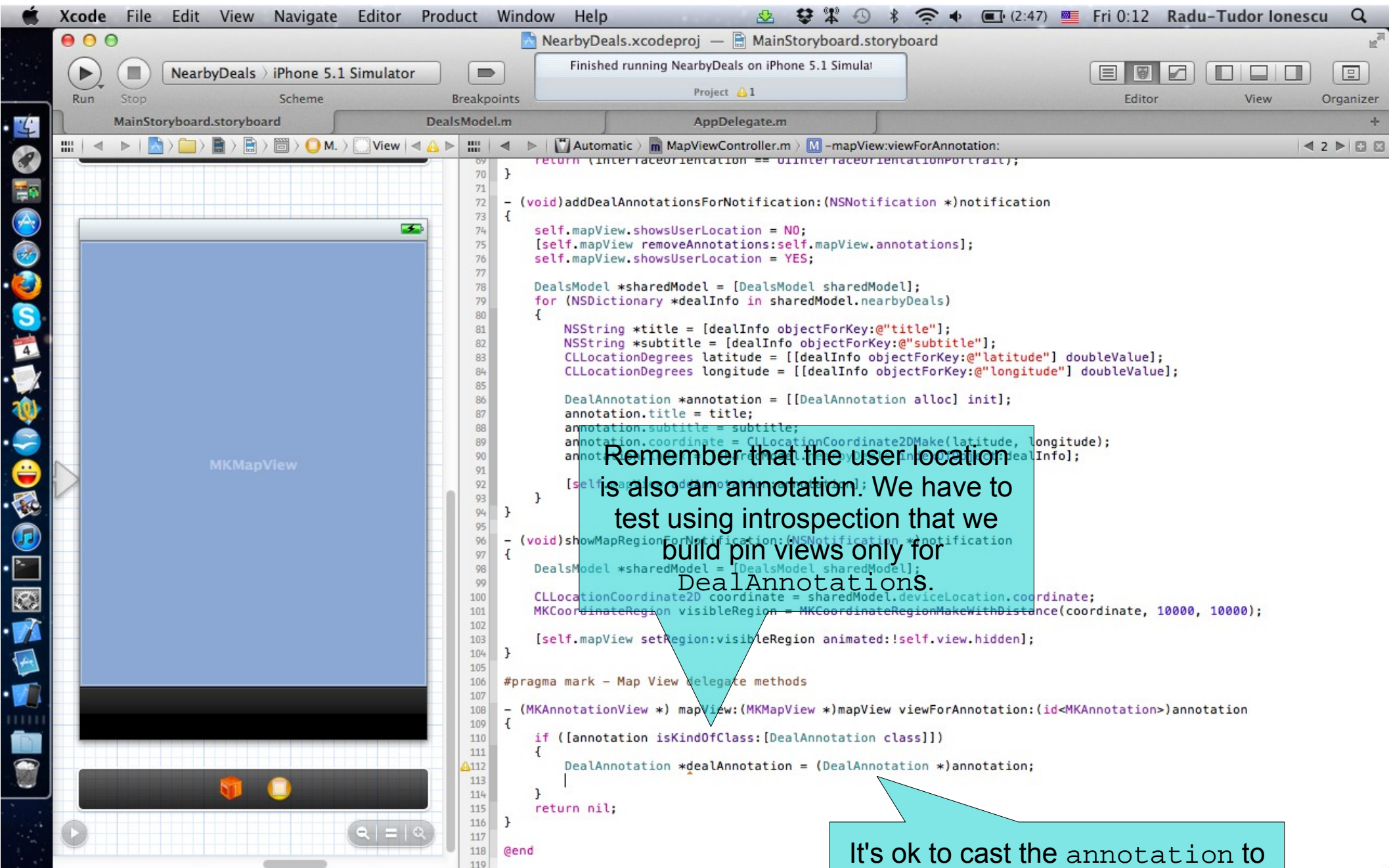


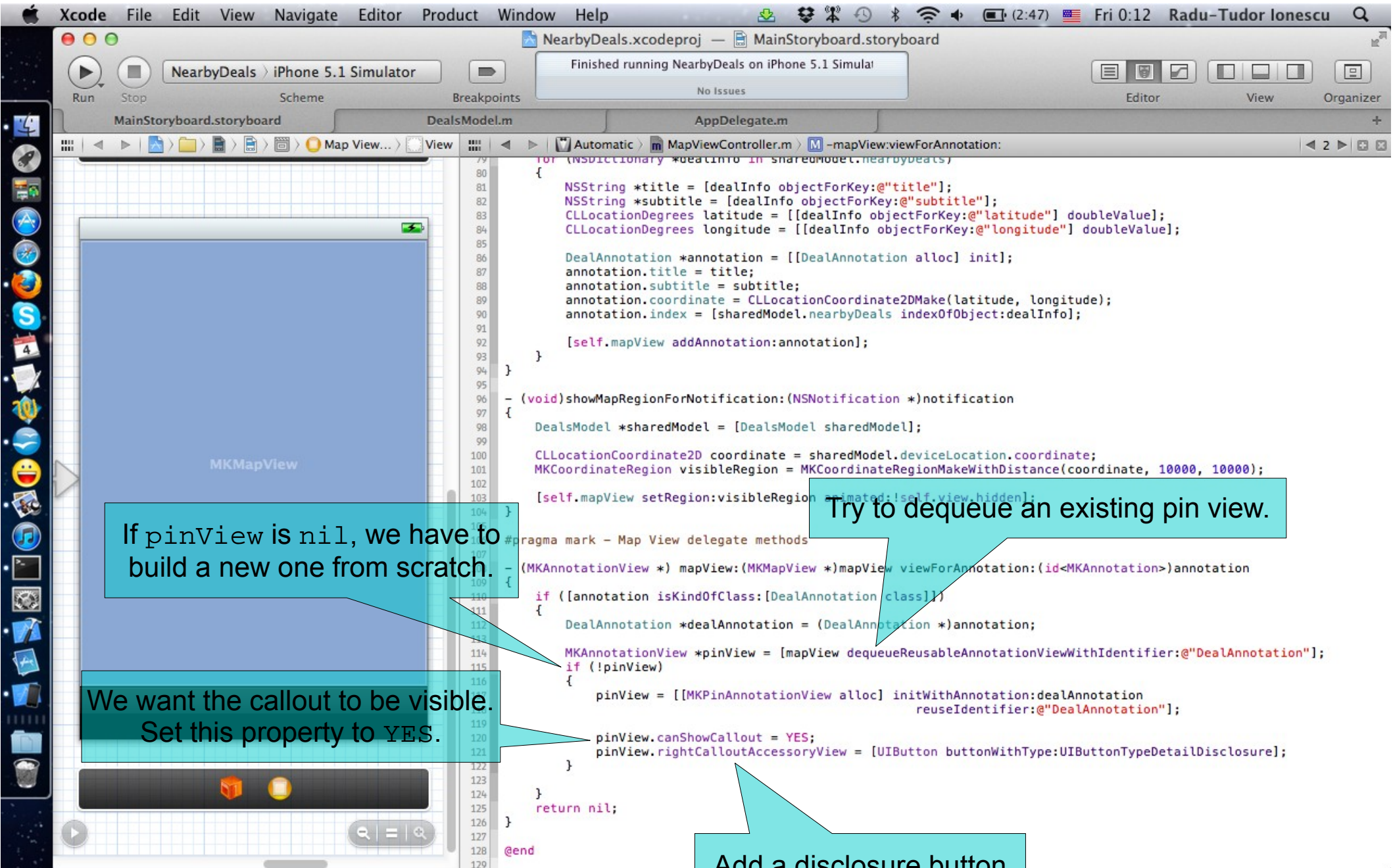
The image shows the Xcode IDE interface. On the left is the iPhone 5.1 Simulator displaying a blue screen with the text "MKMapView". On the right is the code editor for "MapViewController.m". The code includes the following methods:

```
64 }
65 }
66
67 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
68 {
69     return (interfaceOrientation == UIInterfaceOrientationPortrait);
70 }
71
72 - (void)addDealAnnotationsForNotification:(NSNotification *)notification
73 {
74     self.mapView.showsUserLocation = NO;
75     [self.mapView removeAnnotations:self.mapView.annotations];
76     self.mapView.showsUserLocation = YES;
77
78     DealsModel *sharedModel = [DealsModel sharedModel];
79     for (NSDictionary *dealInfo in sharedModel.nearbyDeals)
80     {
81         NSString *title = [dealInfo objectForKey:@"title"];
82         NSString *subtitle = [dealInfo objectForKey:@"subtitle"];
83         CLLocationCoordinate2D latitude = [[dealInfo objectForKey:@"latitude"] doubleValue];
84         CLLocationCoordinate2D longitude = [[dealInfo objectForKey:@"longitude"] doubleValue];
85
86         DealAnnotation *annotation = [[DealAnnotation alloc] init];
87         annotation.title = title;
88         annotation.subtitle = subtitle;
89         annotation.coordinate = CLLocationCoordinate2DMake(latitude, longitude);
90         annotation.index = [sharedModel.nearbyDeals indexOfObject:dealInfo];
91     }
92     [self.mapView addAnnotations:annotation];
93 }
94
95 - (MKAnnotationView *)mapView:(MKMapView *)mapView viewForAnnotation:(id<MKAnnotation> *)notification
96 {
97     DealsModel *sharedModel = [DealsModel sharedModel];
98
99     CLLocationCoordinate2D coordinate = sharedModel.deviceLocation.coordinate;
100     MKCoordinateRegion visibleRegion = MKCoordinateRegionMakeWithDistance(coordinate, 10000, 10000);
101
102     [self.mapView setRegion:visibleRegion animated:!self.view.hidden];
103 }
104
105 #pragma mark - Map View delegate methods
106
107 - (MKAnnotationView *) mapView:(MKMapView *)mapView viewForAnnotation:(id<MKAnnotation>)annotation
108 {
109 }
110
111 }
112
113 @end
114
```

A teal callout box with a pointer to line 96 contains the text: "Use #pragma mark to delimit the section of code with MKMapViewDelegate methods."

Another teal callout box with a pointer to line 107 contains the text: "And let's add implementation to this method."



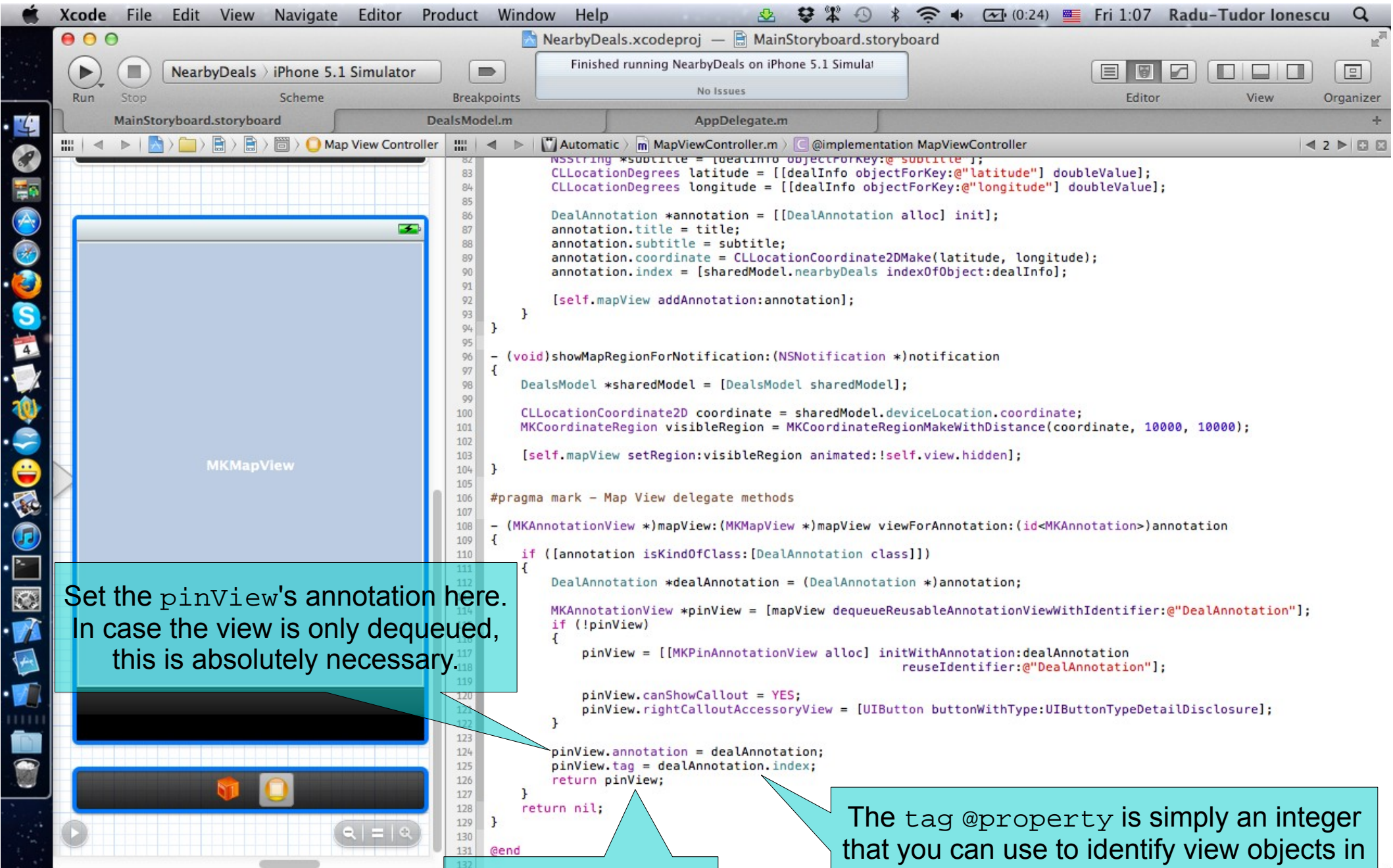


If pinView is nil, we have to build a new one from scratch.

We want the callout to be visible. Set this property to YES.

Try to dequeue an existing pin view.

Add a disclosure button as the right accessory.



Task 2

Task: Add functionality to access deal details by selecting a pin from the map.

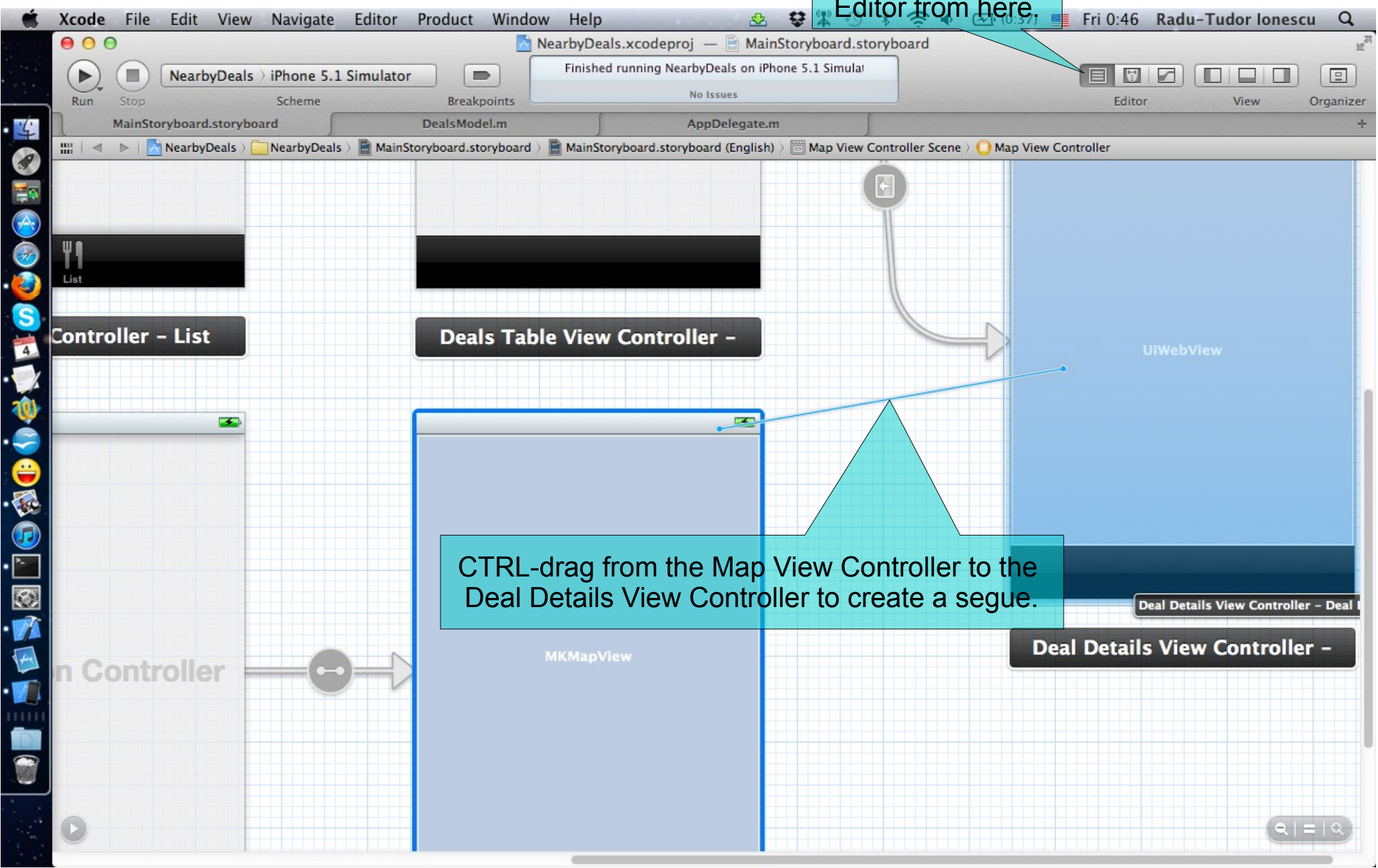
7. The next thing to do is to implement the `MKMapViewDelegate` method that will get called when the accessory view is touched.

This method will simply perform a segue that we need to create in Interface Builder first.

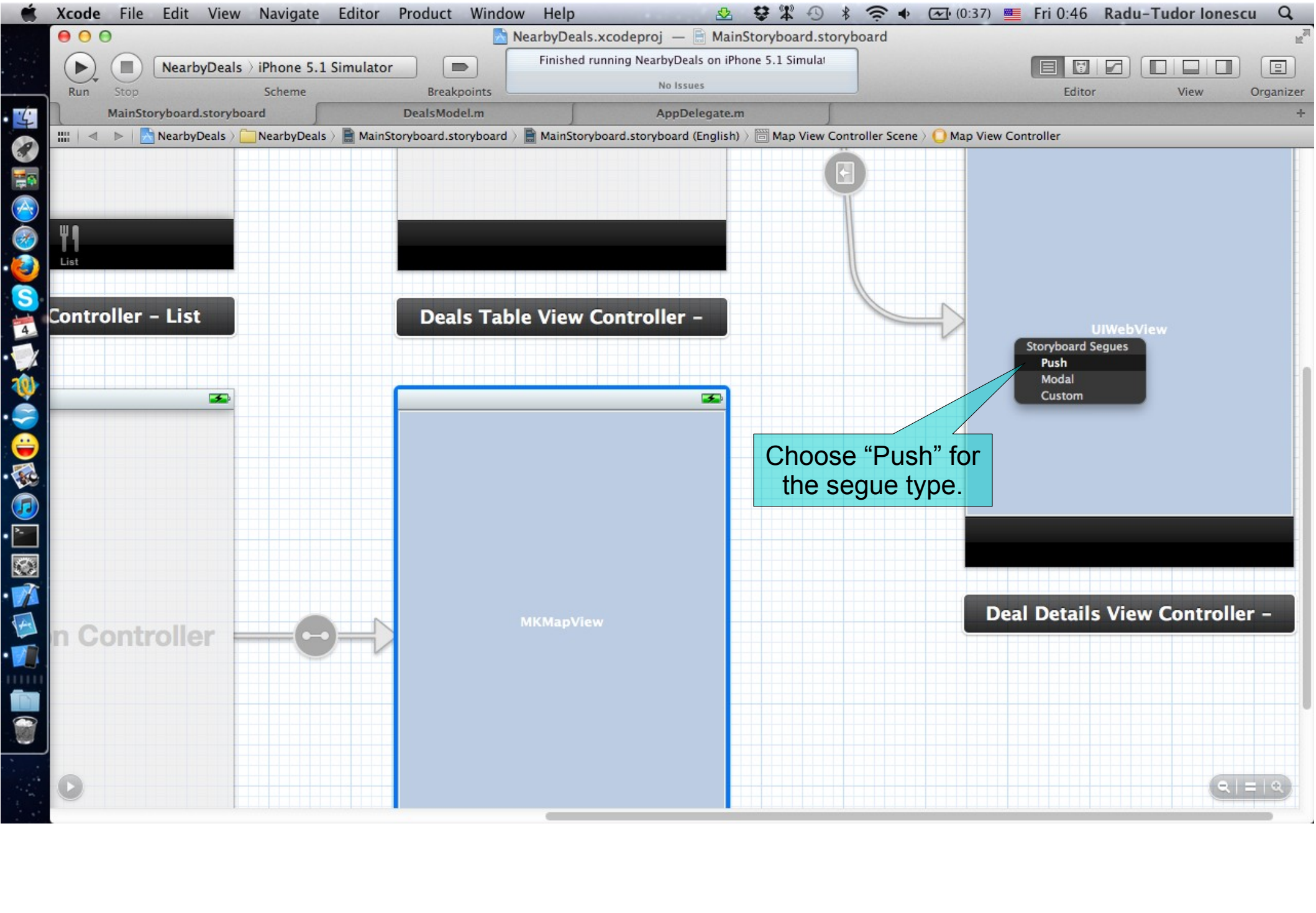
Make more room for the storyboard by selecting the standard Editor.

Follow the steps from the next slides to add and configure the segue.

Open the standard Editor from here,



CTRL-drag from the Map View Controller to the Deal Details View Controller to create a segue.



Finished running NearbyDeals on iPhone 5.1 Simulator

No Issues

MainStoryboard.storyboard

DealsModel.m

AppDelegate.m

NearbyDeals > NearbyDeals > MainStoryboard.storyboard > MainStoryboard.storyboard (English) > Map View Controller Scene > Map View Controller

Controller - List

Deals Table View Controller -

MKMapView

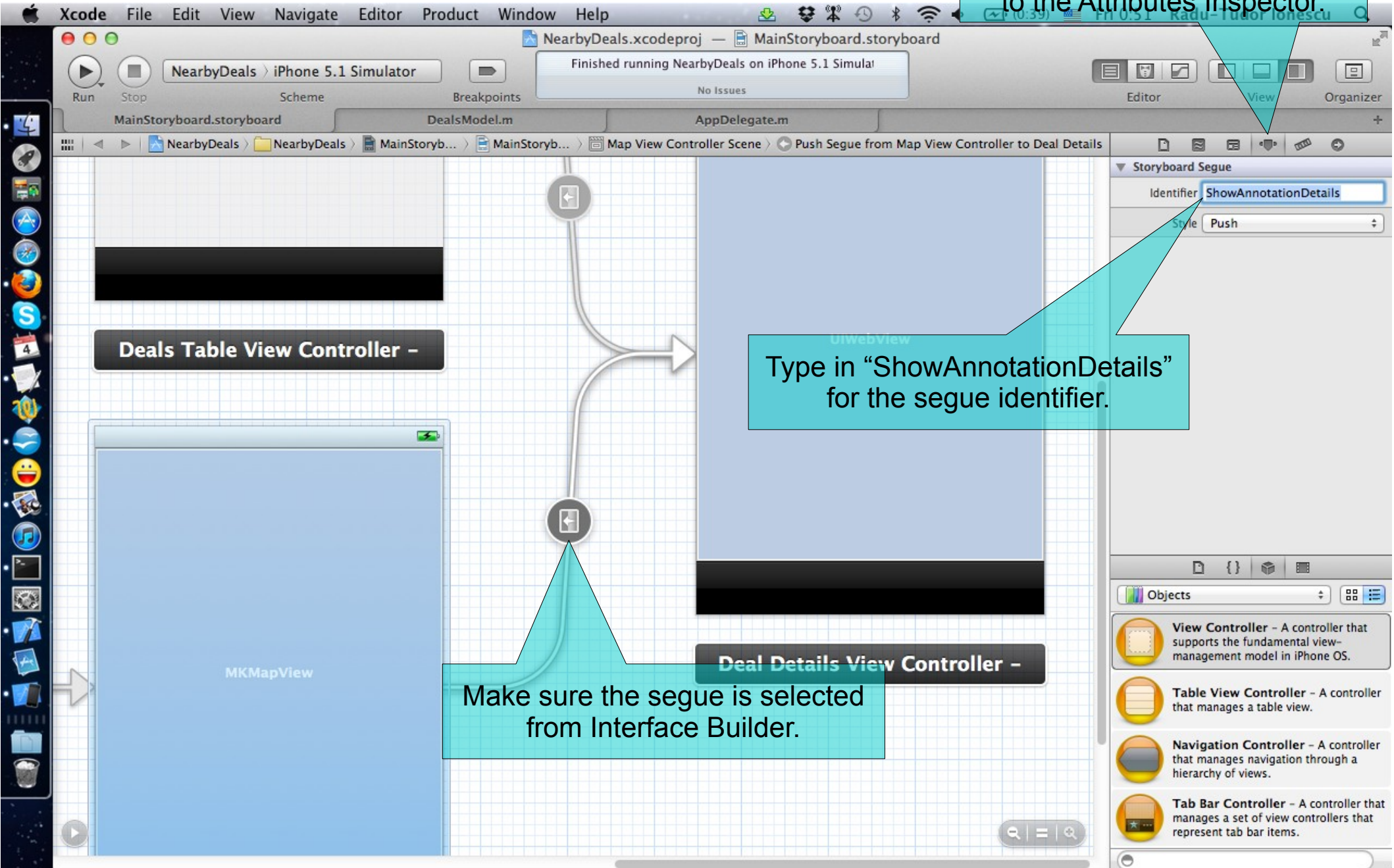
n Controller

Deal Details View Controller -

- Storyboard Segues
- Push
- Modal
- Custom

Choose "Push" for the segue type.

Open Utilities area and switch to the Attributes Inspector



Type in "ShowAnnotationDetails" for the segue identifier.

Make sure the segue is selected from Interface Builder.

Task 2

Task: Add functionality to access deal details by selecting a pin from the map.

8. Hide Utilities area and open Assistant Editor again.
9. Click on the Map View Controller in Interface Builder to see its associated files in Assistant Editor. Select the MapViewController.m file.
10. Add the following method to the Map View delegate methods section of code:

```
mapView:annotationView:calloutAccessoryControlTapped:.
```

11. Implement this method to perform the “ShowAnnotationDetails” segue. Use the `performSegueWithIdentifier:sender:` and pass the `MKAnnotationView` object as the sender argument.

Look over the next slide for hints.

Xcode interface showing the development of a map application. The top status bar indicates the build succeeded at 01:07 AM. The left sidebar shows the project structure with files like MainStoryboard.storyboard, DealsModel.m, AppDelegate.m, and MapViewController.m. The right pane displays the source code for MapViewController.m, which implements map delegate methods for showing map regions and handling map annotations.

```
87     annotation.title = title;
88     annotation.subtitle = subtitle;
89     annotation.coordinate = CLLocationCoordinate2DMake(latitude, longitude);
90     annotation.index = [sharedModel.nearbyDeals indexOfObject:dealInfo];
91
92     [self.mapView addAnnotation:annotation];
93 }
94 }
95
96 - (void)showMapRegionForNotification:(NSNotification *)notification
97 {
98     DealsModel *sharedModel = [DealsModel sharedModel];
99
100    CLLocationCoordinate2D coordinate = sharedModel.deviceLocation.coordinate;
101    MKCoordinateRegion visibleRegion = MKCoordinateRegionMakeWithDistance(coordinate, 10000, 10000);
102
103    [self.mapView setRegion:visibleRegion animated:!self.view.hidden];
104 }
105
106 #pragma mark - Map View delegate methods
107
108 - (MKAnnotationView *)mapView:(MKMapView *)mapView viewForAnnotation:(id<MKAnnotation>)annotation
109 {
110     if ([annotation isKindOfClass:[DealAnnotation class]])
111     {
112         DealAnnotation *dealAnnotation = (DealAnnotation *)annotation;
113
114         MKAnnotationView *pinView = [mapView dequeueReusableAnnotationViewWithIdentifier:@"DealAnnotation"];
115         if (!pinView)
116         {
117             pinView = [[MKPinAnnotationView alloc] initWithAnnotation:dealAnnotation
118                       reuseIdentifier:@"DealAnnotation"];
119
120             pinView.canShowCallout = YES;
121             pinView.rightCalloutAccessoryView = [UIButton buttonWithType:UIButtonTypeDetailDisclosure];
122         }
123
124         pinView.annotation = dealAnnotation;
125         pinView.tag = dealAnnotation.index;
126         return pinView;
127     }
128     return nil;
129 }
130
131 - (void)mapView:(MKMapView *)sender annotationView:(MKAnnotationView *)aView calloutAccessoryControlTapped:(UIControl *)control
132 {
133     [self performSegueWithIdentifier:@"ShowAnnotationDetails" sender:aView];
134 }
135
136 @end
137
```

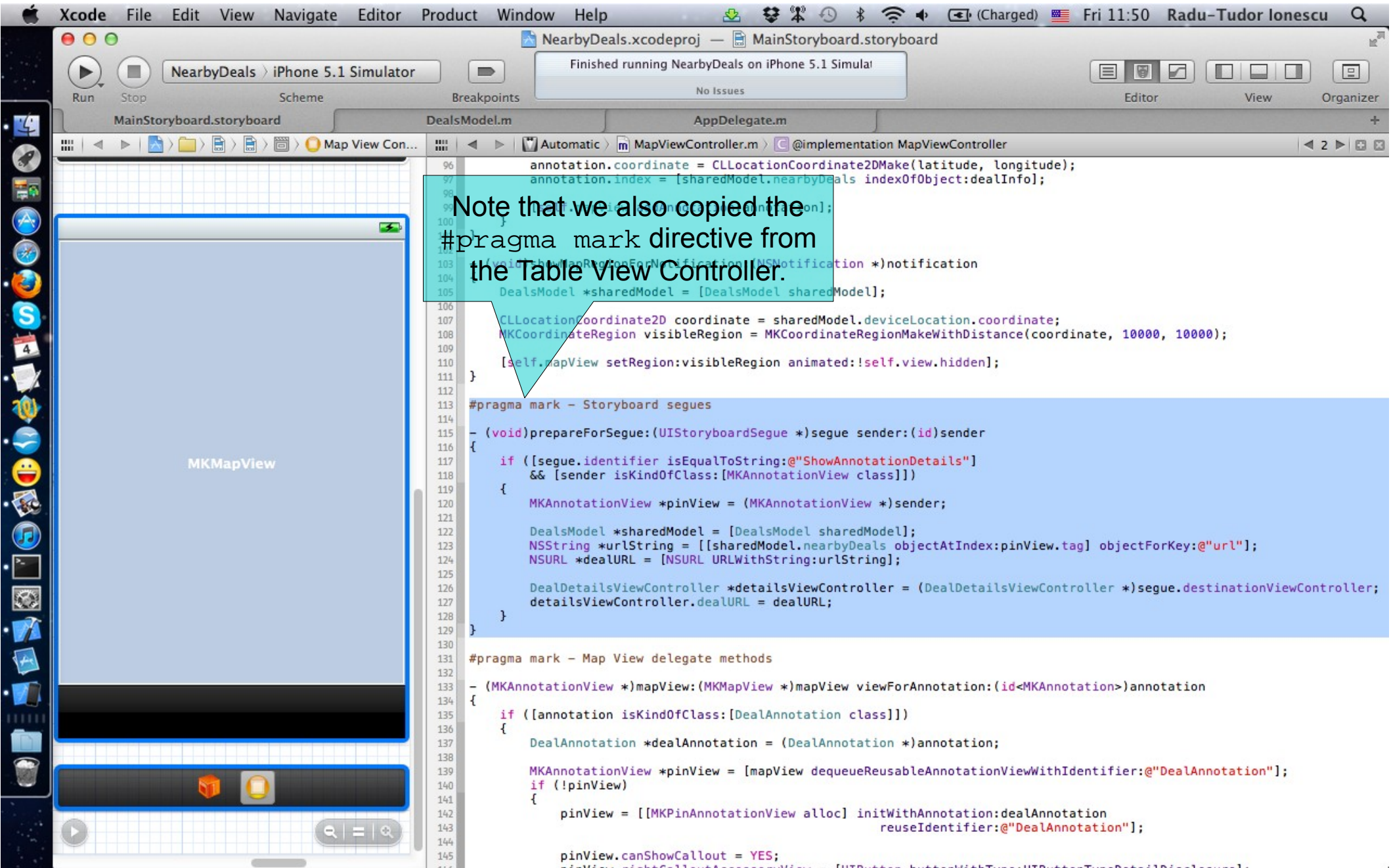

Task 2

Task: Add functionality to access deal details by selecting a pin from the map.

12. The Map View Controller must prepare for this segue in a very similar way to the Table View Controller. Let's copy and paste the `prepareForSegue:sender:` from the Table View Controller implementation file.
13. Note that we have to `#import` the “DealDetailsViewController.h” header file.
14. Use introspection to verify that the `sender` argument is `isKindOfClass: MKAnnotationView`. Cast the `sender` to the `MKAnnotationView` strong type and save it in a local variable named `pinView`.

We have to pass the deal URL to the Deal Details View Controller. Use the `pinView.tag` to get the URL of the appropriate deal from the `nearbyDeals` array.

Look over the next slide for help.



Note that we also copied the #pragma mark directive from the Table View Controller.

```
96     annotation.coordinate = CLLocationCoordinate2DMake(latitude, longitude);
97     annotation.index = [sharedModel.nearbyDeals indexOfObject:dealInfo];
98
99     [self.mapView addAnnotation:annotation];
100
101     // MapViewDelegate implementation
102     (void)mapView:(MKMapView *)mapView viewForAnnotation:(id<MKAnnotation>)annotation
103     {
104         DealsModel *sharedModel = [DealsModel sharedModel];
105
106         CLLocationCoordinate2D coordinate = sharedModel.deviceLocation.coordinate;
107         MKCoordinateRegion visibleRegion = MKCoordinateRegionMakeWithDistance(coordinate, 10000, 10000);
108
109         [self.mapView setRegion:visibleRegion animated:!self.view.hidden];
110     }
111
112
113 #pragma mark - Storyboard segues
114
115 - (void)prepareForSegue:(UIStoryboardSegue *)segue sender:(id)sender
116 {
117     if ([segue.identifier isEqualToString:@"ShowAnnotationDetails"]
118         && [sender isKindOfClass:[MKAnnotationView class]])
119     {
120         MKAnnotationView *pinView = (MKAnnotationView *)sender;
121
122         DealsModel *sharedModel = [DealsModel sharedModel];
123         NSString *urlString = [[sharedModel.nearbyDeals objectAtIndex:pinView.tag] objectForKey:@"url"];
124         NSURL *dealURL = [NSURL URLWithString:urlString];
125
126         DealDetailsViewController *detailsViewController = (DealDetailsViewController *)segue.destinationViewController;
127         detailsViewController.dealURL = dealURL;
128     }
129 }
130
131 #pragma mark - Map View delegate methods
132
133 - (MKAnnotationView *)mapView:(MKMapView *)mapView viewForAnnotation:(id<MKAnnotation>)annotation
134 {
135     if ([annotation isKindOfClass:[DealAnnotation class]])
136     {
137         DealAnnotation *dealAnnotation = (DealAnnotation *)annotation;
138
139         MKAnnotationView *pinView = [mapView dequeueReusableAnnotationViewWithIdentifier:@"DealAnnotation"];
140         if (!pinView)
141         {
142             pinView = [[MKPinAnnotationView alloc] initWithAnnotation:dealAnnotation
143                       reuseIdentifier:@"DealAnnotation"];
144
145             pinView.canShowCallout = YES;
146             pinView.rightCalloutAccessoryView = [UIButton buttonWithType:UIButtonTypeDetailDisclosure];
147         }
148     }
149 }
```

Task 2

Task: Add functionality to access deal details by selecting a pin from the map.

15. Run the application in iOS Simulator.
16. Simulate locations using the BucharestLocations GPX file.
17. Wait until you get nearby deals, then stop simulating location updates.
18. Navigate to the second tab of the application. Tap on a pin from the map to see its callout view. Tap on the disclosure button to see details about that deal.
19. Note that there is no way to return to the Map View because the Navigation Bar is hidden.
20. Stop running the application and let's fix this.

Task 2

Task: Add functionality to access deal details by selecting a pin from the map.

21. Click on the Deal Details View Controller in Interface Builder and select the DealDetailsViewController.m file in Assistant Editor.

22. To make sure that the Navigation Bar is always visible on screen we must set the `navigationBarHidden` `BOOL` @property of the `navigationController` to `NO`, right when `viewWillAppear:`.

23. But because it's the same navigation controller of the Map View Controller, setting this property to `NO` will also make the Navigation Bar visible on the Map View. The solution is to set this property to `YES` when the Map View will appear on screen.

Click on the Map View Controller in Interface Builder and select the `MapViewController.m` file in Assistant Editor.

24. Implement the `viewWillAppear:` method and let `super` prepare for the `viewWillAppear:` event. Set `navigationBarHidden` to `YES`. Look over the next slide to see how to implement this.

Xcode interface showing the development of a map application. The top bar displays the menu (Xcode, File, Edit, View, Navigate, Editor, Product, Window, Help) and system status (Charged, Fri 12:13, Radu-Tudor Ionescu). The main window is titled "NearbyDeals.xcodeproj - MainStoryboard.storyboard".

The interface is split into three main areas:

- Left Panel (Storyboard):** Shows a preview of an iPhone 5.1 Simulator. The screen displays a large blue rectangle labeled "MKMapView". The bottom dock contains two icons: a red cube and a yellow speech bubble.
- Top Panel (File Manager):** Shows the project structure with files: MainStoryboard.storyboard, DealsModel.m, AppDelegate.m, and MapViewCon... (partially visible).
- Right Panel (Code Editor):** Displays the source code for MapViewController.m, specifically the `-viewWillAppear:` method. The code is as follows:

```
42 DealAnnotation *annotation = [[DealAnnotation alloc] init];
43 annotation.title = title;
44 annotation.subtitle = subtitle;
45 annotation.coordinate = CLLocationCoordinate2DMake(latitude, longitude);
46 annotation.index = [sharedModel.nearbyDeals objectAtIndex:dealInfo];
47
48 [self.mapView addAnnotation:annotation];
49
50 }
51
52 CLLocationCoordinate2D coordinate = sharedModel.deviceLocation.coordinate;
53 if ((float)coordinate.latitude != 0.0 && (float)coordinate.longitude != 0.0)
54 {
55     MKCoordinateRegion visibleRegion = MKCoordinateRegionMakeWithDistance(coordinate, 10000, 10000);
56     [self.mapView setRegion:visibleRegion animated:NO];
57 }
58 }
59
60 - (void)viewDidUnload
61 {
62     [self setMapView:nil];
63     [super viewDidUnload];
64
65     [[NSNotificationCenter defaultCenter] removeObserver:self];
66 }
67
68 - (void)viewWillAppear:(BOOL)animated
69 {
70     [super viewWillAppear:YES];
71     self.navigationController.navigationBarHidden = YES;
72 }
73
74 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
75 {
76     return (interfaceOrientation == UIInterfaceOrientationPortrait);
77 }
78
79 - (void)addDealAnnotationsForNotification:(NSNotification *)notification
80 {
81     self.mapView.showsUserLocation = NO;
82     [self.mapView removeAnnotations:self.mapView.annotations];
83     self.mapView.showsUserLocation = YES;
84
85     DealsModel *sharedModel = [DealsModel sharedModel];
86     for (NSDictionary *dealInfo in sharedModel.nearbyDeals)
87     {
88         NSString *title = [dealInfo objectForKey:@"title"];
89         NSString *subtitle = [dealInfo objectForKey:@"subtitle"];
90         CLLocationCoordinateDegrees latitude = [[dealInfo objectForKey:@"latitude"] doubleValue];
91         CLLocationCoordinateDegrees longitude = [[dealInfo objectForKey:@"longitude"] doubleValue];
```

Task 3

Task: Add thumbnail images to the pin callout views on the map.

1. In a similar fashion to the Table View Controller we can download the thumbnail images synchronously before the callout view appears on screen. But if we look closer at the Table View Controller we will notice that scrolling through the nearby deals list is not very fluent. This happens because each Table View Cell has to wait for its thumbnail to download before it can appear on screen.

A solution to this problem is to load images asynchronously. Let's do this right now using the Grand Central Dispatch API.

Switch to the DealsModel.m tab in Xcode.

2. Add (and `@synthesize`) an `NSMutableDictionary` `@property` that will hold our downloaded thumbnail images. Each object in this dictionary will be an `NSData` stream and its key will be the image URL.
3. Prefix the instance variable with underscore in the `@synthesize` declaration. It should look like in the following screenshot.

Xcode File Edit View Navigate Editor Product Window Help

NearbyDeals.xcodeproj — DealsModel.m

Finished running NearbyDeals on iPhone 5.1 Simulator
No Issues

Run Stop Scheme Breakpoints Editor View Organizer

MainStoryboard.storyboard DealsModel.m AppDelegate.m

NearbyDeals > NearbyDeals > DealsModel.m > images

```
8
9 #import "DealsModel.h"
10
11 @implementation DealsModel
12
13 @synthesize nearbyDeals = _nearbyDeals;
14 @synthesize deviceLocation = _deviceLocation;
15 @synthesize images = _images;
16
17 + (DealsModel *)sharedModel
18 {
19     static DealsModel *sharedModel;
20     if (sharedModel == nil)
21     {
22         sharedModel = [[DealsModel alloc] init];
23     }
24     return sharedModel;
25 }
26
27 + (NSString *)locationUpdateNotificationName
28 {
29     return @"locationUpdateNotification";
30 }
31
32 + (NSString *)dealsUpdateNotificationName
33 {
34     return @"dealsUpdateNotification";
35 }
36
37 - (void)setNearbyDeals:(NSArray *)nearbyDeals
38 {
39     _nearbyDeals = nearbyDeals;
40
41     [[NSNotificationCenter defaultCenter] postNotificationName:[DealsModel dealsUpdateNotificationName]
42                                             object:self];
43 }
44
45 - (void)setDeviceLocation:(CLLocation *)deviceLocation
46 {
47     _deviceLocation = deviceLocation;
48
49     [[NSNotificationCenter defaultCenter] postNotificationName:[DealsModel locationUpdateNotificationName]
50                                             object:self];
51 }
52
53 - (CLLocation *)deviceLocation
54 {
55     if (_deviceLocation == nil)
56     {
57         _deviceLocation = [[CLLocation alloc] initWithLatitude:0.0 longitude:0.0];
58     }
59 }
```

Counterparts > DealsModel.h > images

```
1 //
2 // DealsModel.h
3 // NearbyDeals
4 //
5 // Created by Radu-Tudor Ionescu on 4/17/12.
6 // Copyright (c) 2012 __MyCompanyName__. All rights reserved.
7 //
8
9 #import <Foundation/Foundation.h>
10 #import <CoreLocation/CoreLocation.h>
11
12 @interface DealsModel : NSObject
13
14 @property (nonatomic, strong) NSArray *nearbyDeals;
15 @property (nonatomic, strong) CLLocation *deviceLocation;
16 @property (nonatomic, strong) NSMutableDictionary *images;
17
18 + (DealsModel *)sharedModel;
19 + (NSString *)locationUpdateNotificationName;
20 + (NSString *)dealsUpdateNotificationName;
21
22 @end
23
```

Task 3

Task: Add thumbnail images to the pin callout views on the map.

4. We should start downloading the thumbnail images right after the `nearbyDeals` is set. Thus we will add implementation to its setter.

We use a serial dispatch queue to download images and save them in the `images` `NSMutableDictionary`. We will add images one by one and store them as `NSData` objects (streams of bytes).

Follow the instructions from the next slide to finish this step.

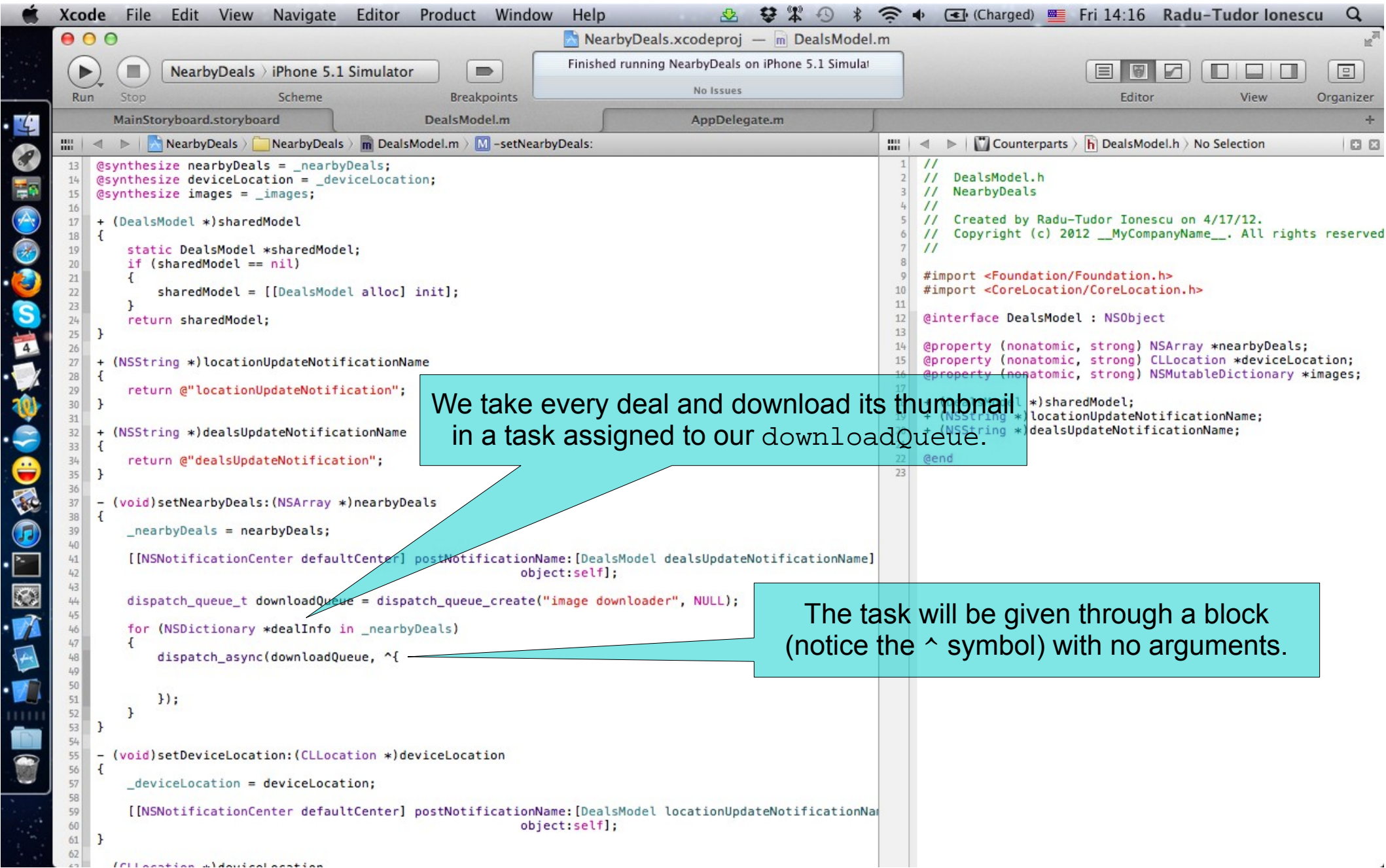

```
Xcode File Edit View Navigate Editor Product Window Help
NearbyDeals.xcodeproj - DealsModel.m
Finished running NearbyDeals on iPhone 5.1 Simulator
Project 1
MainStoryboard.storyboard DealsModel.m AppDelegate.m
NearbyDeals > NearbyDeals > DealsModel.m > M -setNearbyDeals:
Counterparts > DealsModel.h > No Selection

13 @synthesize nearbyDeals = _nearbyDeals;
14 @synthesize deviceLocation = _deviceLocation;
15 @synthesize images = _images;
16
17 + (DealsModel *)sharedModel
18 {
19     static DealsModel *sharedModel;
20     if (sharedModel == nil)
21     {
22         sharedModel = [[DealsModel alloc] init];
23     }
24     return sharedModel;
25 }
26
27 + (NSString *)locationUpdateNotificationName
28 {
29     return @"locationUpdateNotification";
30 }
31
32 + (NSString *)dealsUpdateNotificationName
33 {
34     return @"dealsUpdateNotification";
35 }
36
37 - (void)setNearbyDeals:(NSArray *)nearbyDeals
38 {
39     _nearbyDeals = nearbyDeals;
40
41     [[NSNotificationCenter defaultCenter] postNotificationName:[DealsModel dealsUpdateNotificationName]
42     object:self];
43
44     dispatch_queue_t downloadQueue = dispatch_queue_create("image downloader", NULL);
45 }
46
47 - (void)setDeviceLocation:(CLLocation *)deviceLocation
48 {
49     _deviceLocation = deviceLocation;
50
51     [[NSNotificationCenter defaultCenter] postNotificationName:[DealsModel locationUpdateNotificationName]
52     object:self];
53 }
54
55 - (CLLocation *)deviceLocation
56 {
57     if (_deviceLocation == nil)
58     {
59         _deviceLocation = [[CLLocation alloc] initWithLatitude:0.0 longitude:0.0];
60     }
61     return _deviceLocation;
62 }

1 //
2 // DealsModel.h
3 // NearbyDeals
4 //
5 // Created by Radu-Tudor Ionescu on 4/17/12.
6 // Copyright (c) 2012 __MyCompanyName__. All rights reserved.
7 //
8
9 #import <Foundation/Foundation.h>
10 #import <CoreLocation/CoreLocation.h>
11
12 @interface DealsModel : NSObject
13
14 @property (nonatomic, strong) NSArray *nearbyDeals;
15 @property (nonatomic, strong) CLLocation *deviceLocation;
16 @property (nonatomic, strong) NSMutableDictionary *images;
17
18 + (DealsModel *)sharedModel;
19 + (NSString *)locationUpdateNotificationName;
20 + (NSString *)dealsUpdateNotificationName;
21
22 @end
```

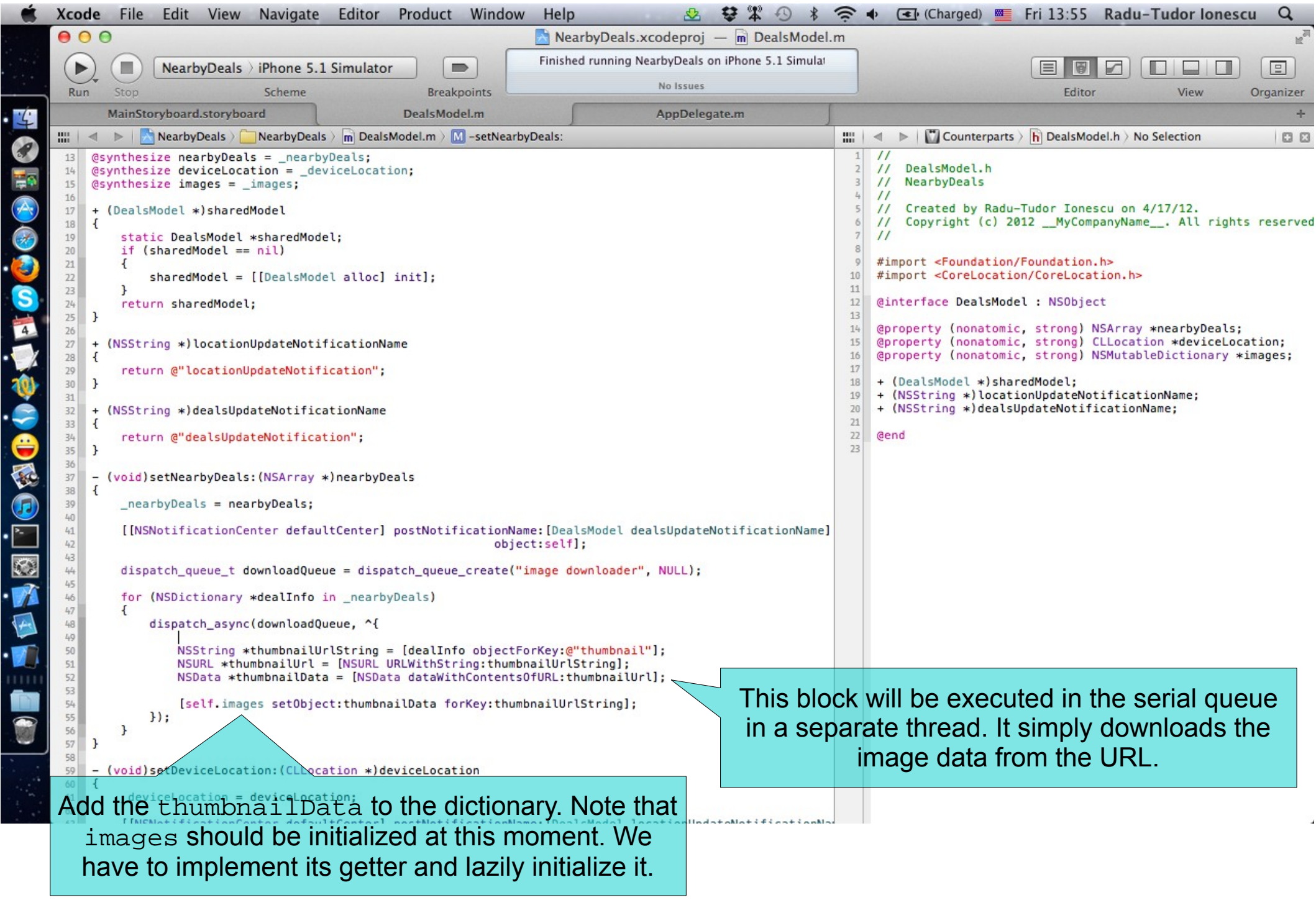
Create a dispatch serial queue first.

This is C API so the strings are char * (no @ here).



We take every deal and download its thumbnail in a task assigned to our downloadQueue.

The task will be given through a block (notice the ^ symbol) with no arguments.



```
13 @synthesize nearbyDeals = _nearbyDeals;
14 @synthesize deviceLocation = _deviceLocation;
15 @synthesize images = _images;
16
17 + (DealsModel *)sharedModel
18 {
19     static DealsModel *sharedModel;
20     if (sharedModel == nil)
21     {
22         sharedModel = [[DealsModel alloc] init];
23     }
24     return sharedModel;
25 }
26
27 + (NSString *)locationUpdateNotificationName
28 {
29     return @"locationUpdateNotification";
30 }
31
32 + (NSString *)dealsUpdateNotificationName
33 {
34     return @"dealsUpdateNotification";
35 }
36
37 - (void)setNearbyDeals:(NSArray *)nearbyDeals
38 {
39     _nearbyDeals = nearbyDeals;
40
41     [[NSNotificationCenter defaultCenter] postNotificationName:[DealsModel dealsUpdateNotificationName]
42                                         object:self];
43
44     dispatch_queue_t downloadQueue = dispatch_queue_create("image downloader", NULL);
45
46     for (NSDictionary *dealInfo in _nearbyDeals)
47     {
48         dispatch_async(downloadQueue, ^{
49             NSString *thumbnailUrlString = [dealInfo objectForKey:@"thumbnail"];
50             NSURL *thumbnailUrl = [NSURL URLWithString:thumbnailUrlString];
51             NSData *thumbnailData = [NSData dataWithContentsOfURL:thumbnailUrl];
52
53             [self.images setObject:thumbnailData forKey:thumbnailUrlString];
54         });
55     }
56 }
57
58 - (void)setDeviceLocation:(CLLocation *)deviceLocation
59 {
60     _deviceLocation = deviceLocation;
61
62     [[NSNotificationCenter defaultCenter] postNotificationName:[DealsModel locationUpdateNotificationName]
```

```
1 //
2 // DealsModel.h
3 // NearbyDeals
4 //
5 // Created by Radu-Tudor Ionescu on 4/17/12.
6 // Copyright (c) 2012 __MyCompanyName__. All rights reserved.
7 //
8
9 #import <Foundation/Foundation.h>
10 #import <CoreLocation/CoreLocation.h>
11
12 @interface DealsModel : NSObject
13
14 @property (nonatomic, strong) NSArray *nearbyDeals;
15 @property (nonatomic, strong) CLLocation *deviceLocation;
16 @property (nonatomic, strong) NSMutableDictionary *images;
17
18 + (DealsModel *)sharedModel;
19 + (NSString *)locationUpdateNotificationName;
20 + (NSString *)dealsUpdateNotificationName;
21
22 @end
```

Add the thumbnailData to the dictionary. Note that images should be initialized at this moment. We have to implement its getter and lazily initialize it.

This block will be executed in the serial queue in a separate thread. It simply downloads the image data from the URL.

```
Xcode File Edit View Navigate Editor Product Window Help
NearbyDeals.xcodeproj - DealsModel.m
NearbyDeals > iPhone 5.1 Simulator
Finished running NearbyDeals on iPhone 5.1 Simula
No Issues
MainStoryboard.storyboard DealsModel.m AppDelegate.m
NearbyDeals > NearbyDeals > DealsModel.m > M -setNearbyDeals:
Counterparts > DealsModel.h > No Selection

13 @synthesize nearbyDeals = _nearbyDeals;
14 @synthesize deviceLocation = _deviceLocation;
15 @synthesize images = _images;
16
17 + (DealsModel *)sharedModel
18 {
19     static DealsModel *sharedModel;
20     if (sharedModel == nil)
21     {
22         sharedModel = [[DealsModel alloc] init];
23     }
24     return sharedModel;
25 }
26
27 + (NSString *)locationUpdateNotificationName
28 {
29     return @"locationUpdateNotification";
30 }
31
32 + (NSString *)dealsUpdateNotificationName
33 {
34     return @"dealsUpdateNotification";
35 }
36
37 - (void)setNearbyDeals:(NSArray *)nearbyDeals
38 {
39     _nearbyDeals = nearbyDeals;
40
41     [[NSNotificationCenter defaultCenter] postNotificationName:[DealsModel dealsUpdateNotificationName]
42                                         object:self];
43
44     dispatch_queue_t downloadQueue = dispatch_queue_create("image downloader", NULL);
45
46     for (NSDictionary *dealInfo in _nearbyDeals)
47     {
48         dispatch_async(downloadQueue, ^{
49
50             NSString *thumbnailUrlString = [dealInfo objectForKey:@"thumbnail"];
51             NSURL *thumbnailUrl = [NSURL URLWithString:thumbnailUrlString];
52             NSData *thumbnailData = [NSData dataWithContentsOfURL:thumbnailUrl];
53
54             [self.images setObject:thumbnailData forKey:thumbnailUrlString];
55         });
56     }
57
58     dispatch_release(downloadQueue);
59 }
60
61 - (void)setDeviceLocation:(CLLocation *)deviceLocation
62 {
63     _deviceLocation = deviceLocation;
64 }

1 //
2 // DealsModel.h
3 // NearbyDeals
4 //
5 // Created by Radu-Tudor Ionescu on 4/17/12.
6 // Copyright (c) 2012 __MyCompanyName__. All rights reserved.
7 //
8
9 #import <Foundation/Foundation.h>
10 #import <CoreLocation/CoreLocation.h>
11
12 @interface DealsModel : NSObject
13
14 @property (nonatomic, strong) NSArray *nearbyDeals;
15 @property (nonatomic, strong) CLLocation *deviceLocation;
16 @property (nonatomic, strong) NSMutableDictionary *images;
17
18 + (DealsModel *)sharedModel;
19 + (NSString *)locationUpdateNotificationName;
20 + (NSString *)dealsUpdateNotificationName;
21
22 @end
```

Release the downloadQueue. Don't worry, it will stay in the heap until it finishes all its tasks.

```
Xcode File Edit View Navigate Editor Product Window Help
NearbyDeals.xcodeproj - DealsModel.m
Finished running NearbyDeals on iPhone 5.1 Simulator
No Issues
MainStoryboard.storyboard DealsModel.m AppDelegate.m
NearbyDeals > NearbyDeals > DealsModel.m > Images
Counterparts > DealsModel.h > Images

38 {
39     _nearbyDeals = nearbyDeals;
40
41     [[NSNotificationCenter defaultCenter] postNotificationName:[DealsModel dealsUpdateNotificationName]
42                                             object:self];
43
44     dispatch_queue_t downloadQueue = dispatch_queue_create("image downloader", NULL);
45
46     for (NSDictionary *dealInfo in _nearbyDeals)
47     {
48         dispatch_async(downloadQueue, ^{
49
50             NSString *thumbnailUrlString = [dealInfo objectForKey:@"thumbnail"];
51             NSURL *thumbnailUrl = [NSURL URLWithString:thumbnailUrlString];
52             NSData *thumbnailData = [NSData dataWithContentsOfURL:thumbnailUrl];
53
54             [self.images setObject:thumbnailData forKey:thumbnailUrlString];
55         });
56     }
57     dispatch_release(downloadQueue);
58
59 }
60 - (void)setDeviceLocation:(CLLocation *)deviceLocation
61 {
62     _deviceLocation = deviceLocation;
63
64     [[NSNotificationCenter defaultCenter] postNotificationName:[DealsModel locationUpdateNotificationName]
65                                             object:self];
66 }
67
68 - (CLLocation *)deviceLocation
69 {
70     if (_deviceLocation == nil)
71     {
72         _deviceLocation = [[CLLocation alloc] initWithLatitude:0.0 longitude:0.0];
73     }
74     return _deviceLocation;
75 }
76
77 - (NSMutableDictionary *)images
78 {
79     if (_images == nil)
80     {
81         _images = [[NSMutableDictionary alloc] init];
82     }
83     return _images;
84 }
85
86
87 @end
88
```

```
1 //
2 // DealsModel.h
3 // NearbyDeals
4 //
5 // Created by Radu-Tudor Ionescu on 4/17/12.
6 // Copyright (c) 2012 __MyCompanyName__. All rights reserved.
7 //
8
9 #import <Foundation/Foundation.h>
10 #import <CoreLocation/CoreLocation.h>
11
12 @interface DealsModel : NSObject
13
14 @property (nonatomic, strong) NSArray *nearbyDeals;
15 @property (nonatomic, strong) CLLocation *deviceLocation;
16 @property (nonatomic, strong) NSMutableDictionary *images;
17
18 + (DealsModel *)sharedModel;
19 + (NSString *)locationUpdateNotificationName;
20 + (NSString *)dealsUpdateNotificationName;
21
22 @end
23
```

Note that images is nil, but it should be initialized at this moment.

Implement the images dictionary getter and lazily initialize the instance variable when it's nil.

Task 3

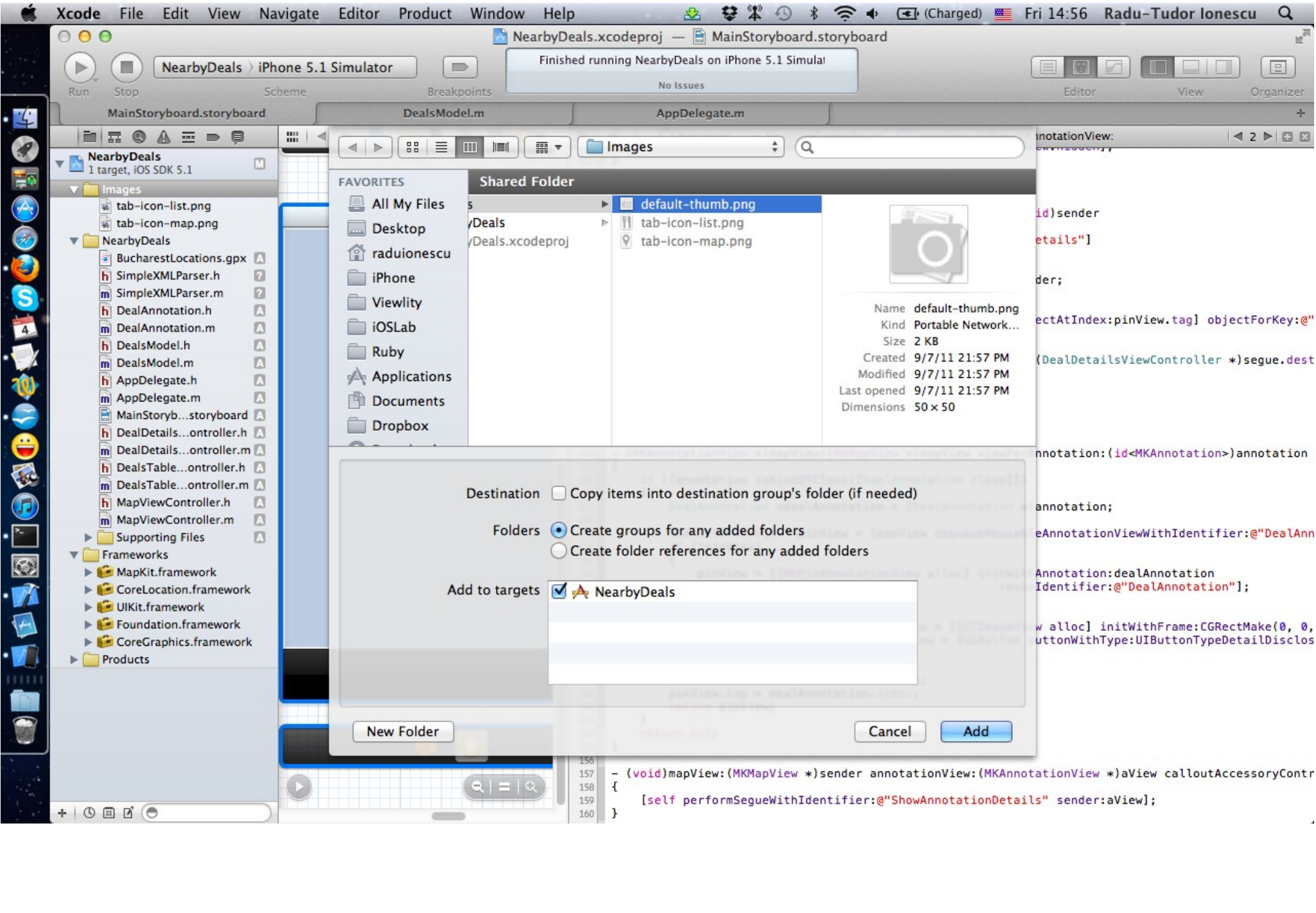
Task: Add thumbnail images to the pin callout views on the map.

5. The images are downloaded asynchronously. But before assigning them as the left accessory of callout views, we have to consider the situation when the callout appears before the application has a chance to download them. It would be nice to display a default thumbnail in this case.

Let's copy and paste the “default-thumb.png” image to our Images subfolder in our Project's folder.

6. Open Project Navigator.
7. Right-click on the Images folder and select the “Add Files to NearbyDeals ...” option.
8. Navigate to the Images subfolder and choose the file named “default-thumb.png”.
9. Click Add to make this image visible to our Project.

The following screenshot gives you a hint.



NearbyDeals iPhone 5.1 Simulator
Run Stop Scheme Breakpoints

Finished running NearbyDeals on iPhone 5.1 Simulator
No Issues

Editor View Organizer

MainStoryboard.storyboard

DealsModel.m

AppDelegate.m

- NearbyDeals
 - 1 target, iOS SDK 5.1
 - Images
 - tab-icon-list.png
 - tab-icon-map.png
 - NearbyDeals
 - BucharestLocations.gpx
 - SimpleXMLParser.h
 - SimpleXMLParser.m
 - DealAnnotation.h
 - DealAnnotation.m
 - DealsModel.h
 - DealsModel.m
 - AppDelegate.h
 - AppDelegate.m
 - MainStoryboard...storyboard
 - DealDetails...ontroller.h
 - DealDetails...ontroller.m
 - DealsTable...ontroller.h
 - DealsTable...ontroller.m
 - MapViewController.h
 - MapViewController.m
 - Supporting Files
 - Frameworks
 - MapKit.framework
 - CoreLocation.framework
 - UIKit.framework
 - Foundation.framework
 - CoreGraphics.framework
 - Products

FAVORITES

- All My Files
- Desktop
- raduionescu
- iPhone
- Viewlity
- iOSLab
- Ruby
- Applications
- Documents
- Dropbox

Shared Folder

- default-thumb.png
- tab-icon-list.png
- tab-icon-map.png



Name default-thumb.png
 Kind Portable Network...
 Size 2 KB
 Created 9/7/11 21:57 PM
 Modified 9/7/11 21:57 PM
 Last opened 9/7/11 21:57 PM
 Dimensions 50 x 50

Destination Copy items into destination group's folder (if needed)

Folders Create groups for any added folders
 Create folder references for any added folders

Add to targets NearbyDeals

New Folder Cancel Add

```

notationView:
...
(id)sender
etails"]
der;
ectAtIndex:pinView.tag] objectForKey:@"
(DealDetailsViewController *)segue.dest
notation:(id<MKAnnotation>)annotation
annotation;
eAnnotationViewWithIdentifier:@"DealAnn
Annotation:dealAnnotation
Identifier:@"DealAnnotation";
w alloc] initWithFrame:CGRectMake(0, 0,
uttonWithType:UIButtonTypeDetailDisclos

```

```

156 - (void)mapView:(MKMapView *)sender annotationView:(MKAnnotationView *)aView calloutAccessoryContr
157 {
158     [self performSegueWithIdentifier:@"ShowAnnotationDetails" sender:aView];
159 }
160

```

Task 3

Task: Add thumbnail images to the pin callout views on the map.

10. We have everything that we need in order to create the `leftCalloutAccessoryView` for each pin.

Hide Project Navigator to make more room for code.

11. Switch to the `MainStoryboard.storyboard` tab in Xcode.

12. Make sure the `MapViewController.m` is selected in Assistant Editor.

13. Let's add implementation to the `mapView:viewForAnnotation:delegate` method. Initialize an `UIImageView` object using `initWithFrame:` and pass the `(0, 0, 30, 30)` frame as argument. Note that the `image @property` will be `nil` (since we only set the frame).

Set the `leftCalloutAccessoryView` to this image view.

Look over the next screenshot to see what needs to be done for this step.

Xcode interface showing the development of a map application. The left pane displays the iPhone 5.1 Simulator with an MKMapView. The right pane shows the source code for MapViewController.m.

```
113 #pragma mark - Storyboard segues
114
115 - (void)prepareForSegue:(UIStoryboardSegue *)segue sender:(id)sender
116 {
117     if ([segue.identifier isEqualToString:@"ShowAnnotationDetails"]
118         && [sender isKindOfClass:[MKAnnotationView class]])
119     {
120         MKAnnotationView *pinView = (MKAnnotationView *)sender;
121
122         DealsModel *sharedModel = [DealsModel sharedModel];
123         NSString *urlString = [[sharedModel.nearbyDeals objectAtIndex:pinView.tag] objectForKey:@"url"];
124         NSURL *dealURL = [NSURL URLWithString:urlString];
125
126         DealDetailsViewController *detailsViewController = (DealDetailsViewController *)segue.destinationViewController;
127         detailsViewController.dealURL = dealURL;
128     }
129 }
130
131 #pragma mark - Map View delegate methods
132
133 - (MKAnnotationView *)mapView:(MKMapView *)mapView viewForAnnotation:(id<MKAnnotation>)annotation
134 {
135     if ([annotation isKindOfClass:[DealAnnotation class]])
136     {
137         DealAnnotation *dealAnnotation = (DealAnnotation *)annotation;
138
139         MKAnnotationView *pinView = [mapView dequeueReusableAnnotationViewWithIdentifier:@"DealAnnotation"];
140         if (!pinView)
141         {
142             pinView = [[MKPinAnnotationView alloc] initWithAnnotation:dealAnnotation
143                     reuseIdentifier:@"DealAnnotation"];
144
145             pinView.canShowCallout = YES;
146             pinView.leftCalloutAccessoryView = [[UIImageView alloc] initWithFrame:CGRectMake(0, 0, 30, 30)];
147             pinView.rightCalloutAccessoryView = [UIButton buttonWithType:UIButtonTypeDetailDisclosure];
148         }
149
150         pinView.annotation = dealAnnotation;
151         pinView.tag = dealAnnotation.index;
152         return pinView;
153     }
154     return nil;
155 }
156
157 - (void)mapView:(MKMapView *)sender annotationView:(MKAnnotationView *)aView calloutAccessoryControlTapped:(UIControl *)control
158 {
159     [self performSegueWithIdentifier:@"ShowAnnotationDetails" sender:aView];
160 }
161
162 @end
163
```

Task 3

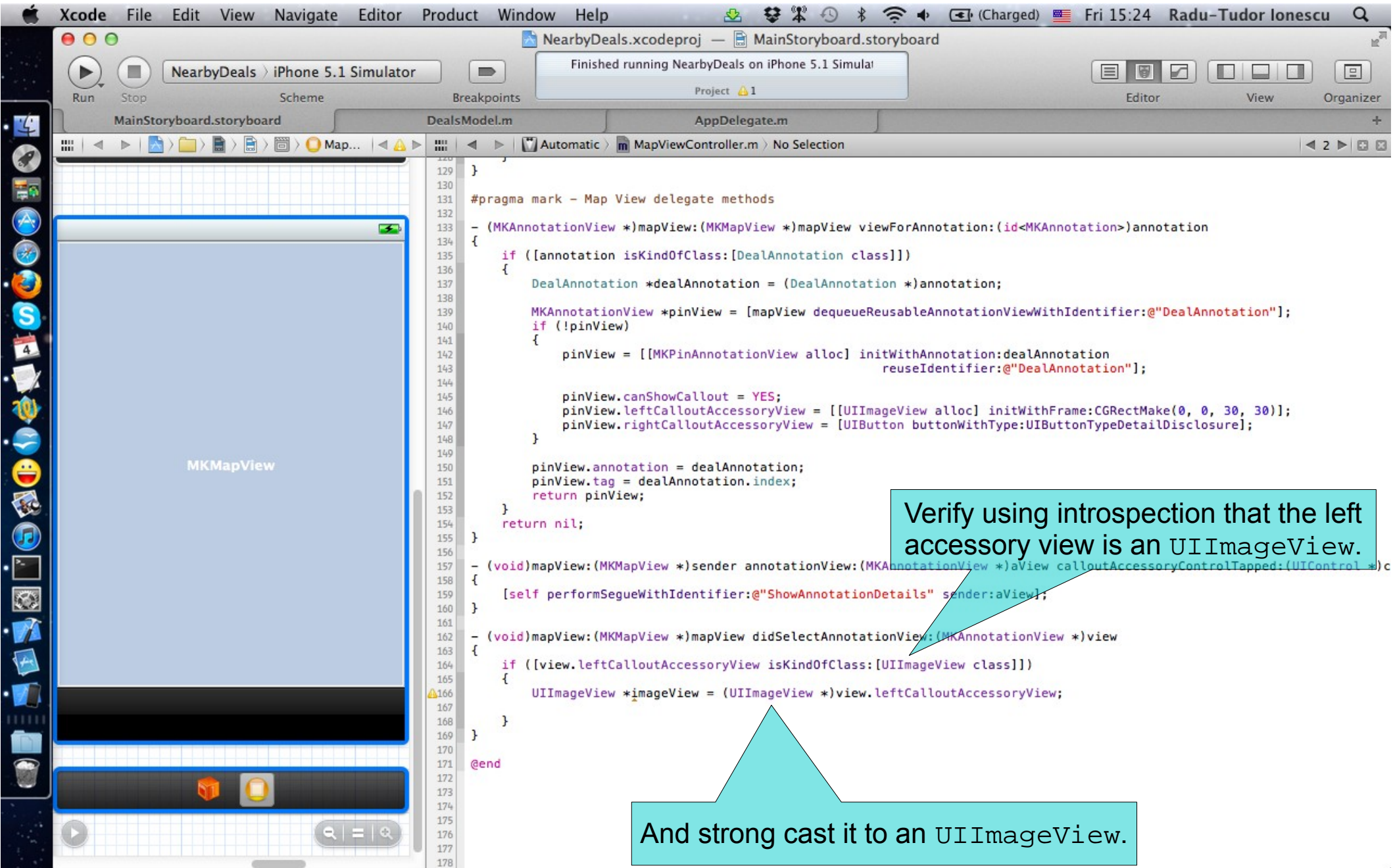
Task: Add thumbnail images to the pin callout views on the map.

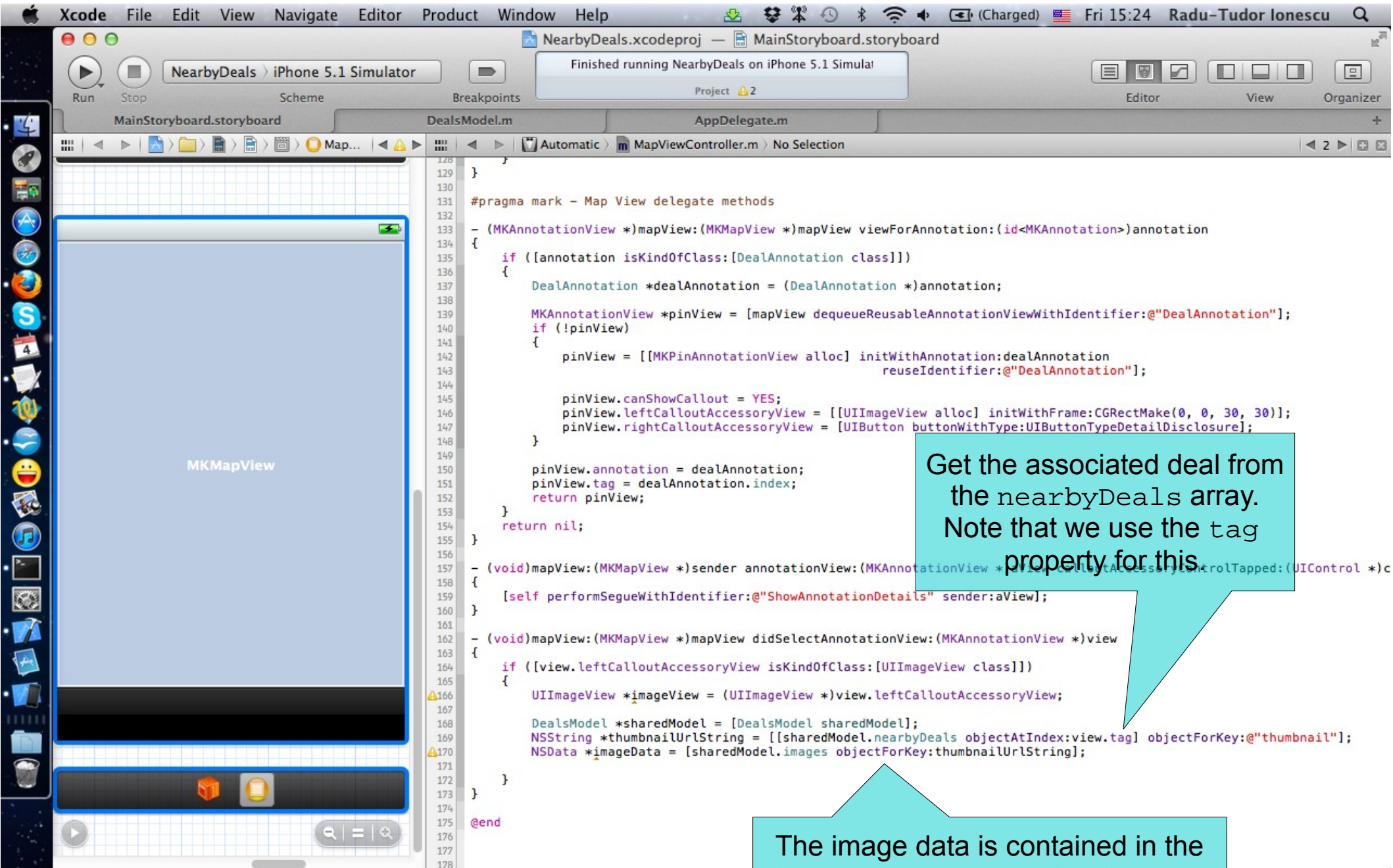
14. We created the left accessory view, but its `image` is `nil` (thus nothing to display in the callout view).

We should set the `image` to an actual image (default or downloaded asynchronously) before presenting the callout view. But we must do this as late as possible because we want to let as much time as possible for downloading the thumbnail images.

Let's implement the `mapView:didSelectAnnotationView:delegate` method and set the image there.

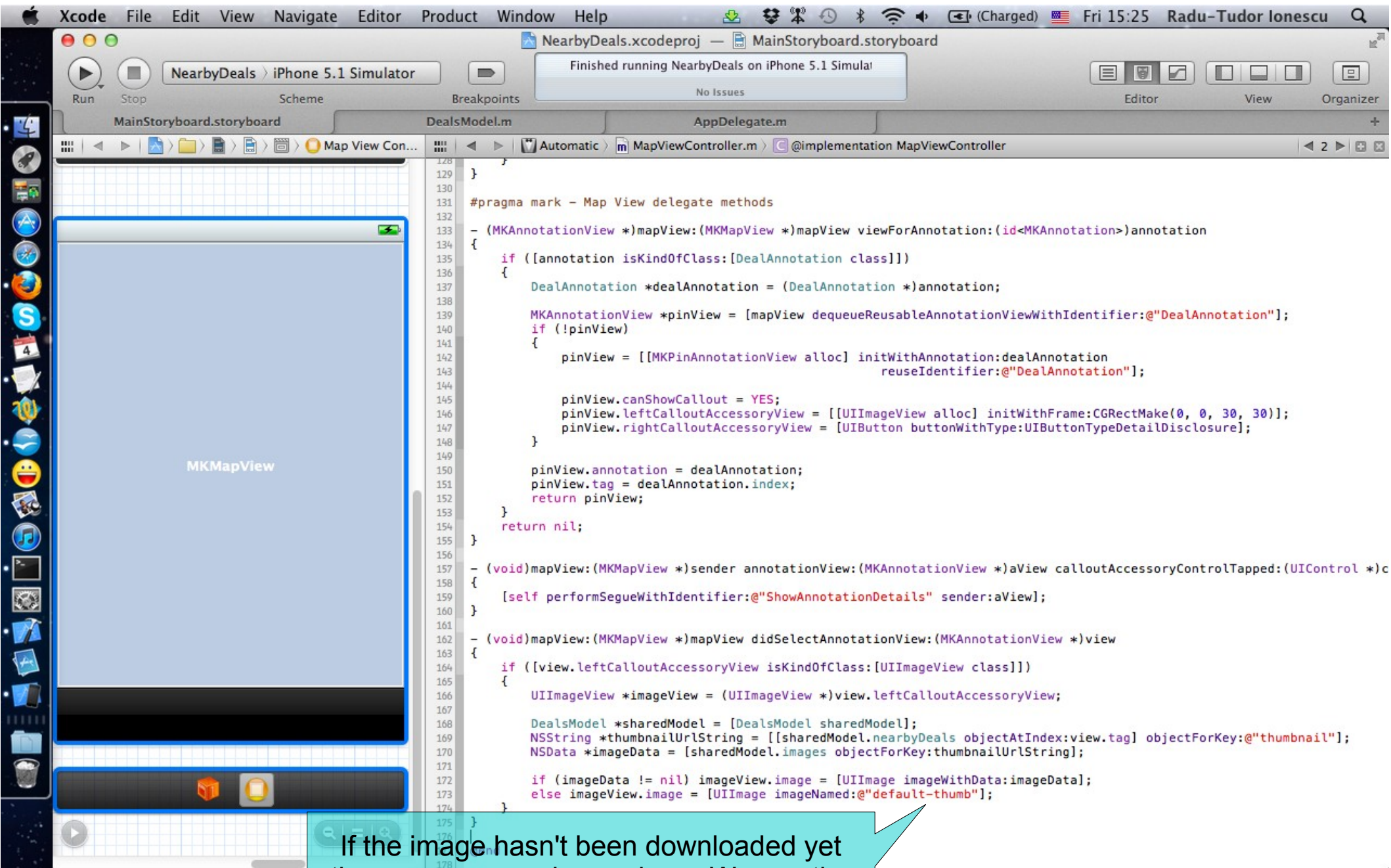
Look over the next slides for help in completing this step.





Get the associated deal from the nearbyDeals array. Note that we use the tag property for this.

The image data is contained in the images NSMutableDictionary. We use objectForKey: to get it.



If the image hasn't been downloaded yet then `imageData` is `nil` here. We use the default thumbnail image in this case.

Task 3

Task: Add thumbnail images to the pin callout views on the map.

15. Run the application in iOS Simulator.
16. Simulate locations using the BucharestLocations GPX file.
17. Wait until you get nearby deals, then stop simulating location updates.
18. Navigate to the second tab of the application. Tap on a pin from the map to see its callout view. Notice the image on left side of the callout view.

Tap on the disclosure button to see details about that deal.

19. Note that you can return to the Map View using the “Back” button on the Navigation Bar.
20. All is good. Stop running the application.

Assignment 1

Assignment: Remove the duplicate code from the `MapViewController.m` implementation.

Hint: Declare a new private method called `addDealAnnotations` and put the duplicate code inside it. Look in the `viewDidLoad` and `addDealAnnotationsForNotification:` implementations to find the duplicate code.

Assignment 2

Assignment: Download the images only if they are not in the dictionary.

Hint: Before putting a task (to download a thumbnail image) in the `downloadQueue`, verify that the image is not already in the dictionary of images.

This may happen when the application receives a location update and it tries to request new nearby deals from the server. It may be that the server will return some of the older deals for the new location (if the two locations are not too far apart). The associated images of those deals should already be in the dictionary (no need to download them again).

Assignment 3*

Assignment: Use the asynchronously downloaded images for Table View cells to make the Table View more responsive.

Hints: You have to re-implement the Table View delegate method `tableView:cellForRowAtIndexPath:`. Stop downloading images synchronously and use the images dictionary of the `sharedModel` instead (in a similar way to the Map View Controller).

Think about what happens if the Table View loads before the images have a chance to be downloaded. You should send the `reloadData` message to the `tableView` after the images have been downloaded.

To find out when the images have downloaded add another task to the `downloadQueue` that posts a notification. Name this notification something like “`didDownloadImagesNotification`”.

The Table View Controller should observe this notification and reload the cells. You would probably have to declare another method (`reloadDataForNotification:`) that will get executed when the Table View Controller receives the notification.

Congratulations!