Developing Applications for iOS

Lab 7:
Nearby Deals (3 of 6)

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

Task 1

Task: Add a Model that will hold the nearby deals received from the server.

1. Launch Xcode and go to “File > Open” and select the Xcode project (.xcodeproj) inside the “NearbyDeals(2of6)” folder. You can also double-click on the .xcodeproj file to open it in Xcode.

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. We start by adding a new class to our Project. This class will be the Model of our application and it will be used by both list and map Views.

Open Project Navigator and right-click on the NearbyDeals group. Select the “New File...” option.

5. Select Objective-C Class from the pop-up window and click Next.

Task 1

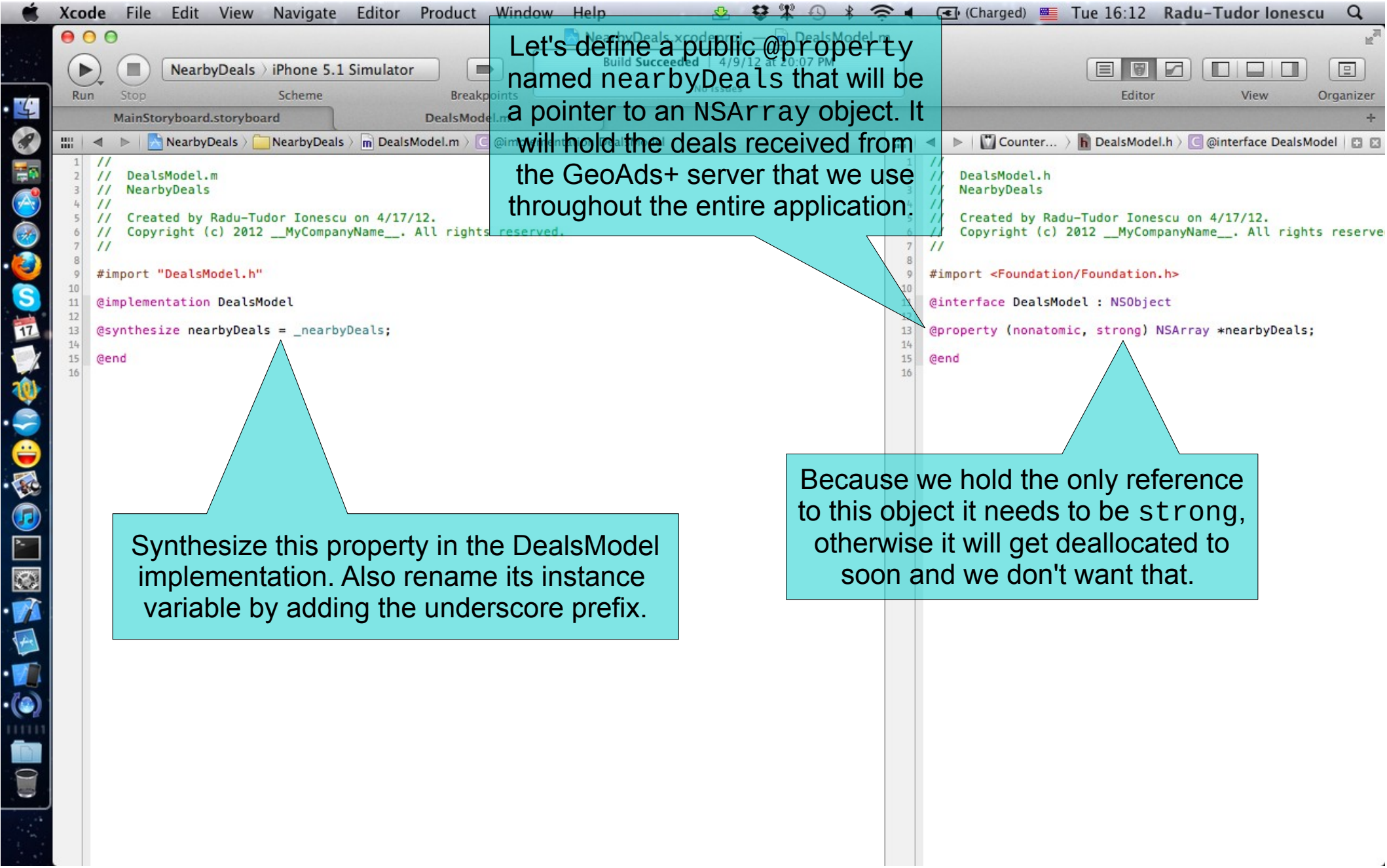
Task: Add a Model that will hold the nearby deals received from the server.

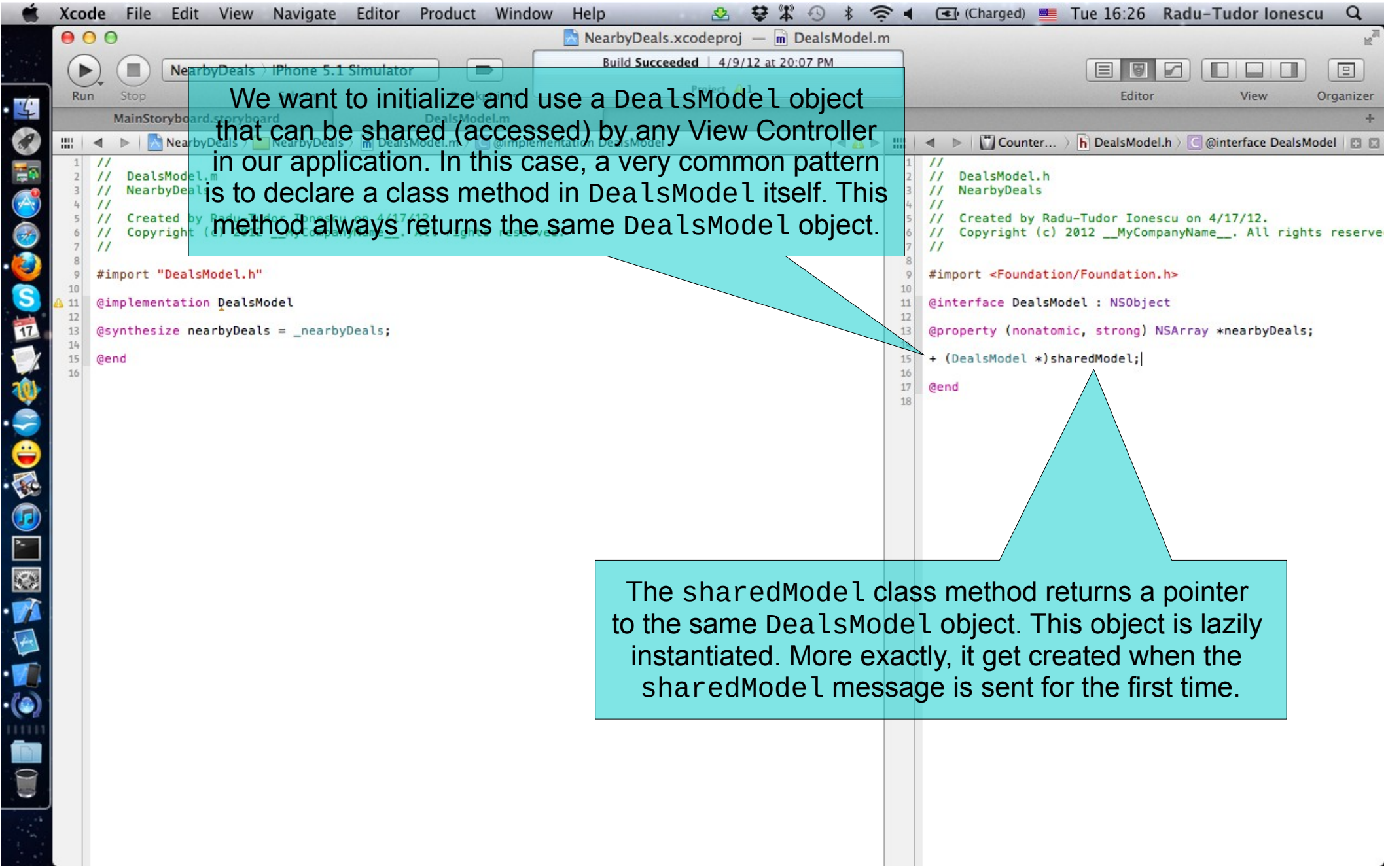
6. Name the class DealsModel and set it as a subclass of `NSObject`. Then click Next.

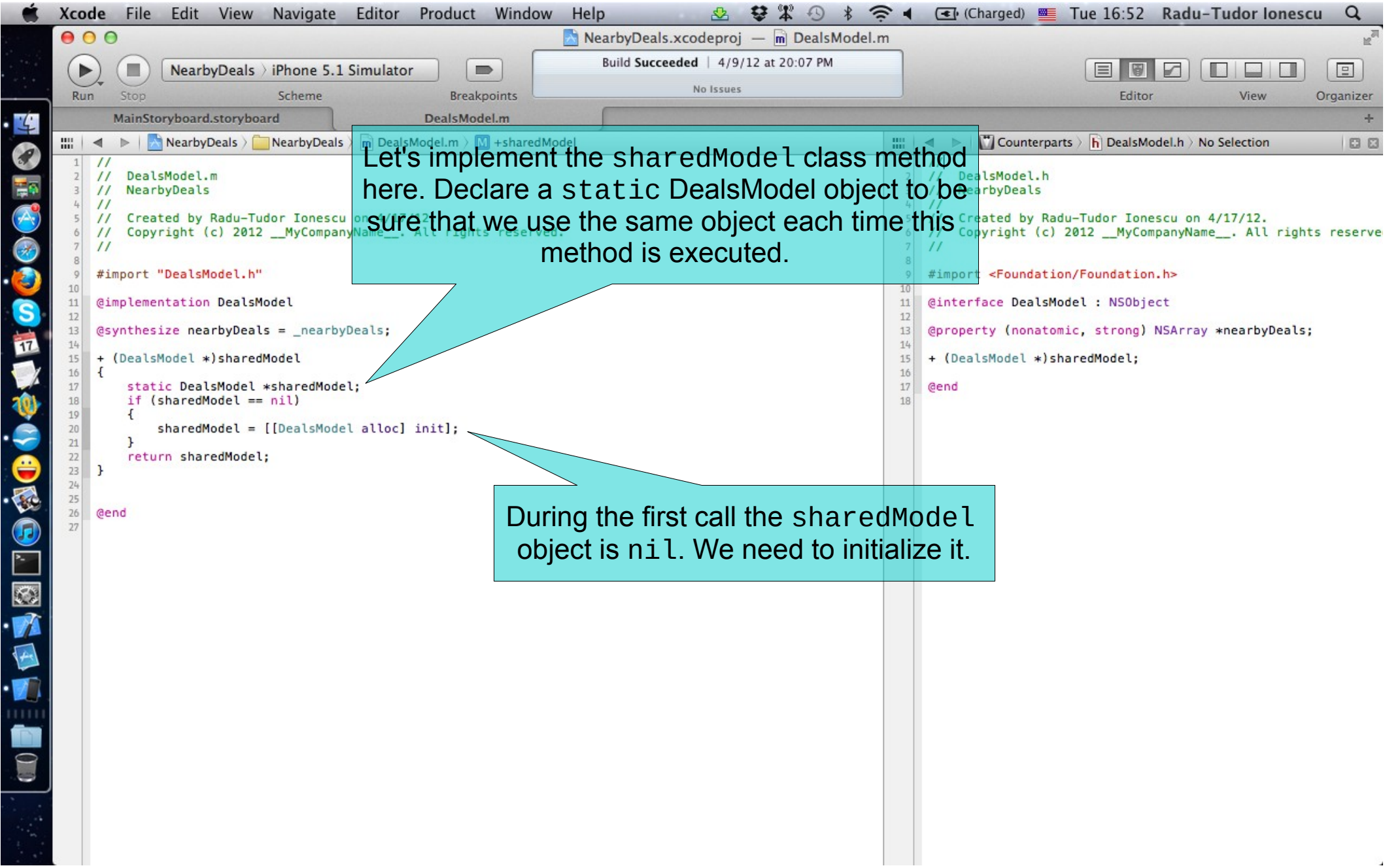
7. Choose the NearbyDeals subfolder to place the .h and .m files inside it. Click Create.

8. Create a new Tab (use the CMD + T shortcut for this) and make sure Assistant Editor is opened. Then, click on DealsModel.m in Project Navigator to open it on the left side of the Assistant Editor. DealsModels.h should be automatically selected (if Assistant Editor is in automatic mode) on the right side.

9. Hide Projector Navigator and let's create our Model. Follow the steps from the following slides to accomplish this task.


Let's define a public @property named nearbyDeals that will be a pointer to an NSArray object. It will hold the deals received from the GeoAds+ server that we use throughout the entire application.
Synthesize this property in the DealsModel implementation. Also rename its instance variable by adding the underscore prefix.
Because we hold the only reference to this object it needs to be strong, otherwise it will get deallocated to soon and we don't want that.


Xcode File Edit View Navigate Editor Product Window Help
NearbyDeals.xcodeproj — DealsModel.m
Build Succeeded | 4/9/12 at 20:07 PM
Run
Stop
Editor
View
Organizer
We want to initialize and use a DealsModel object that can be shared (accessed) by any View Controller in our application. In this case, a very common pattern is to declare a class method in DealsModel itself. This method always returns the same DealsModel object.
//
// DealsModel.m
// NearbyDeals
//
// Created by Radu-Tudor Ionescu on 4/17/12.
// Copyright (c) 2012 __MyCompanyName__. All rights reserved.
//
#import "DealsModel.h"
@implementation DealsModel
@synthesize nearbyDeals = _nearbyDeals;
@end
//
// DealsModel.h
// NearbyDeals
//
// Created by Radu-Tudor Ionescu on 4/17/12.
// Copyright (c) 2012 __MyCompanyName__. All rights reserved.
//
#import <Foundation/Foundation.h>
@interface DealsModel : NSObject
@property (nonatomic, strong) NSArray *nearbyDeals;
+ (DealsModel *)sharedModel;
@end
The sharedModel class method returns a pointer to the same DealsModel object. This object is lazily instantiated. More exactly, it get created when the sharedModel message is sent for the first time.




```
//
//  DealsModel.m
//  NearbyDeals
//
//  Created by Radu-Tudor Ionescu on 4/17/12.
//  Copyright (c) 2012 __MyCompanyName__. All rights reserved.
//

#import "DealsModel.h"

@implementation DealsModel

@synthesize nearbyDeals = _nearbyDeals;

+ (DealsModel *)sharedModel
{
    static DealsModel *sharedModel;
    if (sharedModel == nil)
    {
        sharedModel = [[DealsModel alloc] init];
    }
    return sharedModel;
}


@end
```

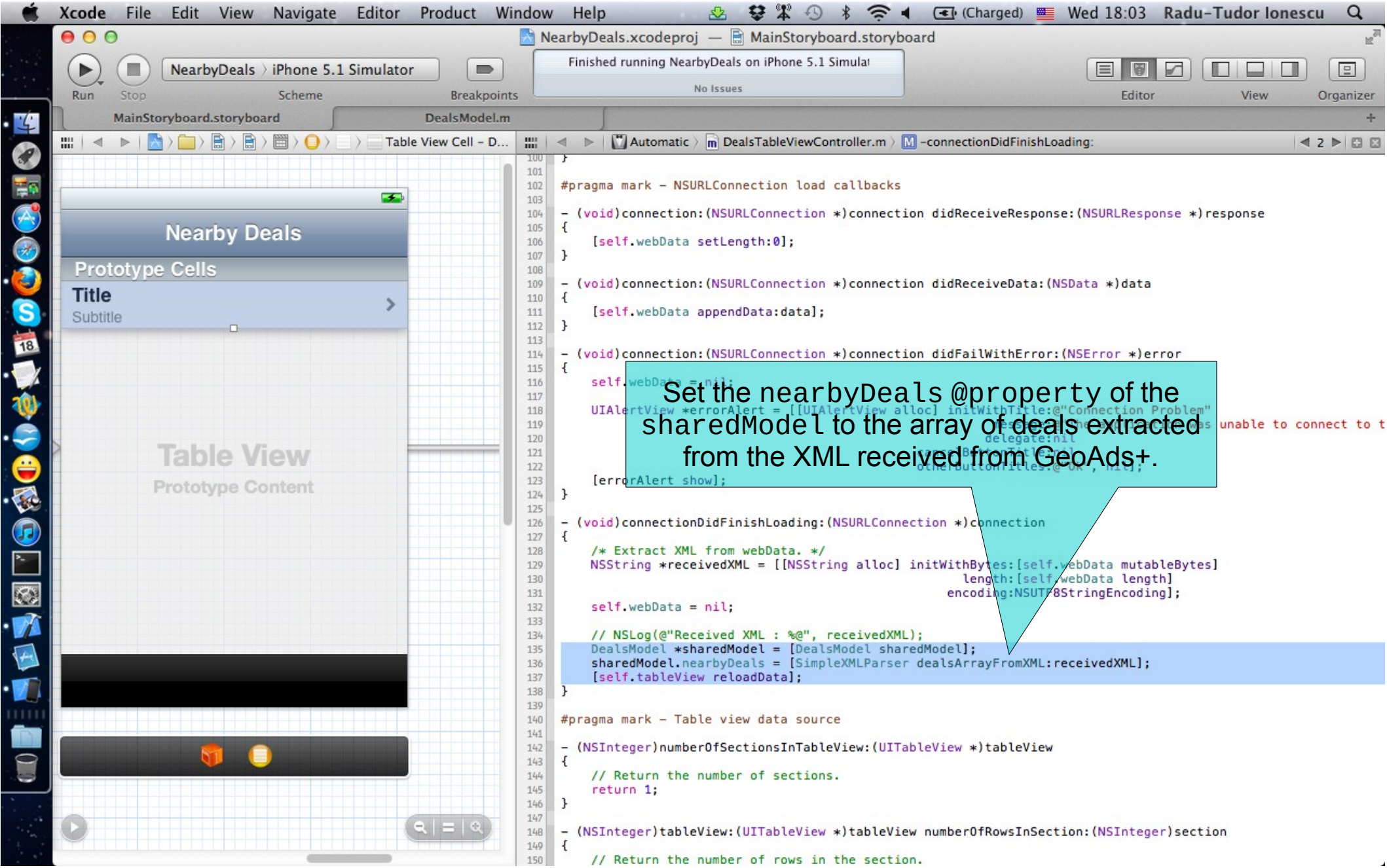
```
//
//  DealsModel.h
//  NearbyDeals
//
//  Created by Radu-Tudor Ionescu on 4/17/12.
//  Copyright (c) 2012 __MyCompanyName__. All rights reserved.
//

#import <Foundation/Foundation.h>

@interface DealsModel : NSObject

@property (nonatomic, strong) NSArray *nearbyDeals;

+ (DealsModel *)sharedModel;

@end
```

Task 2

Task: Change the DealsTableViewController class so that it uses the new shared Model.

1. Switch to the MainStoryboard.storyboard tab in Xcode.
2. Click on the Deals Table View Controller in Interface Builder and select DealsTableViewController.m file in Assistant Editor.
3. In oder to use the shared Model of the application we need to `#import` the "`DealsModel.h`" header file.
4. Scroll down to the `connectionDidFinishLoading:` method and modify it so that it sets the shared Model of the application. The `nearbyDeals` array will not be used anymore. Instead, the Model of our Table View Controller will be the `NSArray` of deals from the `sharedModel`. Note that we need to tell the `tableView` to `reloadData` in this method (and every time the `sharedModel` changes the list of `nearbyDeals`).

Look at the next screenshot to see how to implement the `connectionDidFinishLoading:` method.


Xcode File Edit View Navigate Editor Product Window Help
Wed 18:03 Radu-Tudor Ionescu
NearbyDeals.xcodeproj — MainStoryboard.storyboard
Run Stop Scheme Breakpoints
NearbyDeals iPhone 5.1 Simulator
Finished running NearbyDeals on iPhone 5.1 Simulat
No Issues
Editor View Organizer
MainStoryboard.storyboard
DealsModel.m
Table View Cell – D...
Automatic DealsTableViewController.m -connectionDidFinishLoading:
Nearby Deals
Prototype Cells
Title
Subtitle
Table View
Prototype Content
Set the nearbyDeals @property of the sharedModel to the array of deals extracted from the XML received from GeoAds+.
#pragma mark - NSURLConnection load callbacks
- (void)connection:(NSURLConnection *)connection didReceiveResponse:(NSURLResponse *)response
{
[self.webData setLength:0];
}
- (void)connection:(NSURLConnection *)connection didReceiveData:(NSData *)data
{
[self.webData appendData:data];
}
- (void)connection:(NSURLConnection *)connection didFailWithError:(NSError *)error
{
UIAlertView *errorAlert = [[UIAlertView alloc] initWithTitle:@"Connection Problem"
delegate:nil
[errorAlert show];
}
- (void)connectionDidFinishLoading:(NSURLConnection *)connection
{
/* Extract XML from webData. */
NSString *receivedXML = [[NSString alloc] initWithBytes:[self.webData mutableBytes]
length:[self.webData length]
encoding:NSUTF8StringEncoding];
self.webData = nil;
// NSLog(@"Received XML : %@", receivedXML);
DealsModel *sharedModel = [DealsModel sharedModel];
sharedModel.nearbyDeals = [SimpleXMLParser dealsArrayFromXML:receivedXML];
[self.tableView reloadData];
}
#pragma mark - Table view data source
- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
{
// Return the number of sections.
return 1;
}
- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
{
// Return the number of rows in the section.

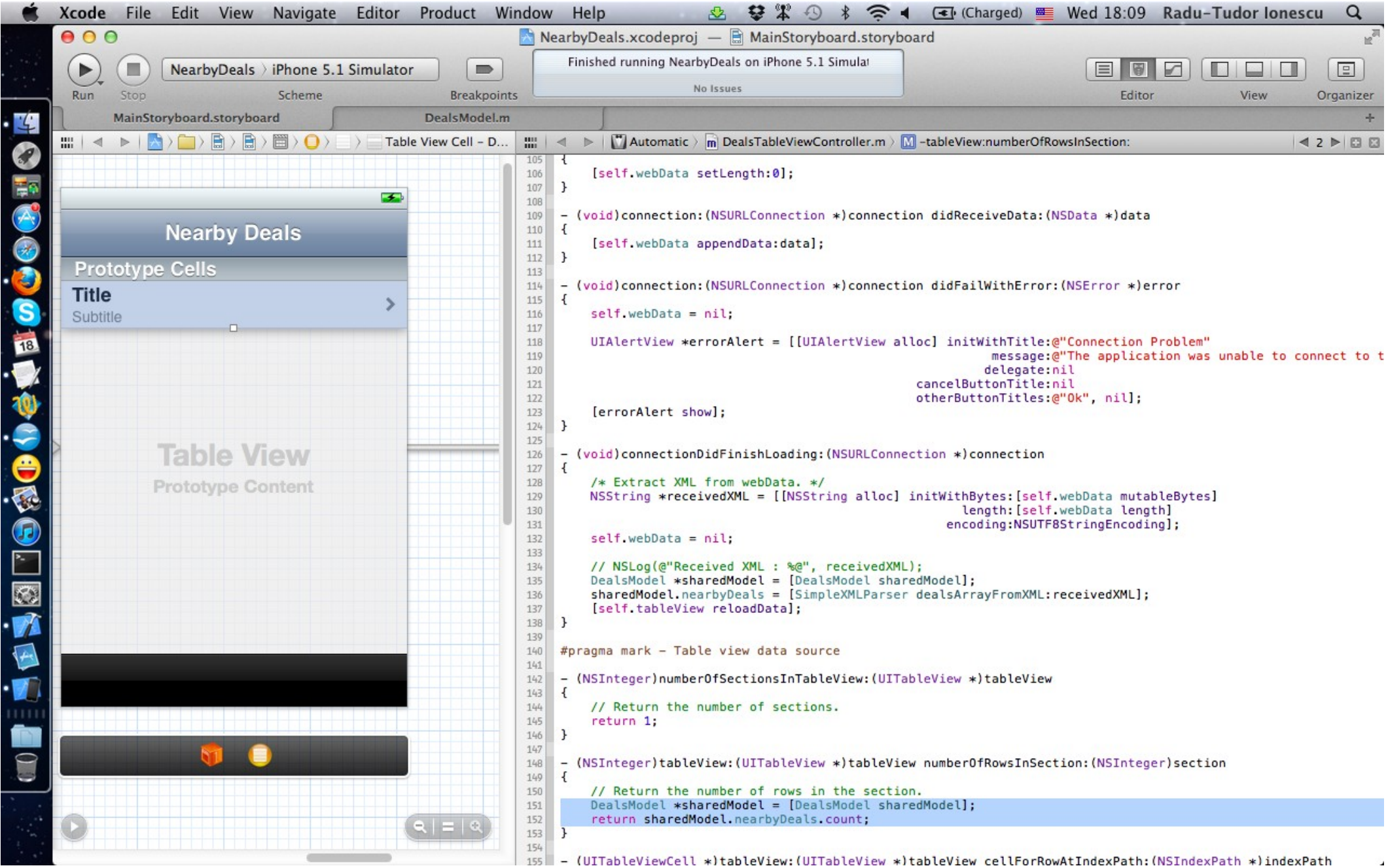
Task 2

Task: Change the DealsTableViewController class so that it uses the new shared Model.

5. Next, we have to adjust the Table View `dataSource` methods so that we use the deals array from the new `sharedModel`.

The `tableView:numberOfRowsInSection:` method should return the number of deals inside the `sharedModel`.

Look at the next screenshot to see how to re-implement this method.


Xcode File Edit View Navigate Editor Product Window Help
(Charged) Wed 18:09 Radu-Tudor Ionescu
NearbyDeals.xcodeproj — MainStoryboard.storyboard
Run Stop NearbyDeals iPhone 5.1 Simulator Scheme Breakpoints
Finished running NearbyDeals on iPhone 5.1 Simula
No Issues
Editor View Organizer
MainStoryboard.storyboard
DealsModel.m
Table View Cell – D...
Nearby Deals
Prototype Cells
Title
Subtitle
Table View
Prototype Content
Automatic DealsTableViewController.m -tableView:numberOfRowsInSection:
{
[self.webData setLength:0];
}
- (void)connection:(NSURLConnection *)connection didReceiveData:(NSData *)data
{
[self.webData appendData:data];
}
- (void)connection:(NSURLConnection *)connection didFailWithError:(NSError *)error
{
self.webData = nil;
UIAlertView *errorAlert = [[UIAlertView alloc] initWithTitle:@"Connection Problem"
message:@"The application was unable to connect to t
delegate:nil
cancelButtonTitle:nil
otherButtonTitles:@"Ok", nil];
[errorAlert show];
}
- (void)connectionDidFinishLoading:(NSURLConnection *)connection
{
/* Extract XML from webData. */
NSString *receivedXML = [[NSString alloc] initWithBytes:[self.webData mutableBytes]
length:[self.webData length]
encoding:NSUTF8StringEncoding];
self.webData = nil;
// NSLog(@"Received XML : %@", receivedXML);
DealsModel *sharedModel = [DealsModel sharedModel];
sharedModel.nearbyDeals = [SimpleXMLParser dealsArrayFromXML:receivedXML];
[self.tableView reloadData];
}
#pragma mark - Table view data source
- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
{
// Return the number of sections.
return 1;
}
- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
{
// Return the number of rows in the section.
DealsModel *sharedModel = [DealsModel sharedModel];
return sharedModel.nearbyDeals.count;
}
- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath

Task 2

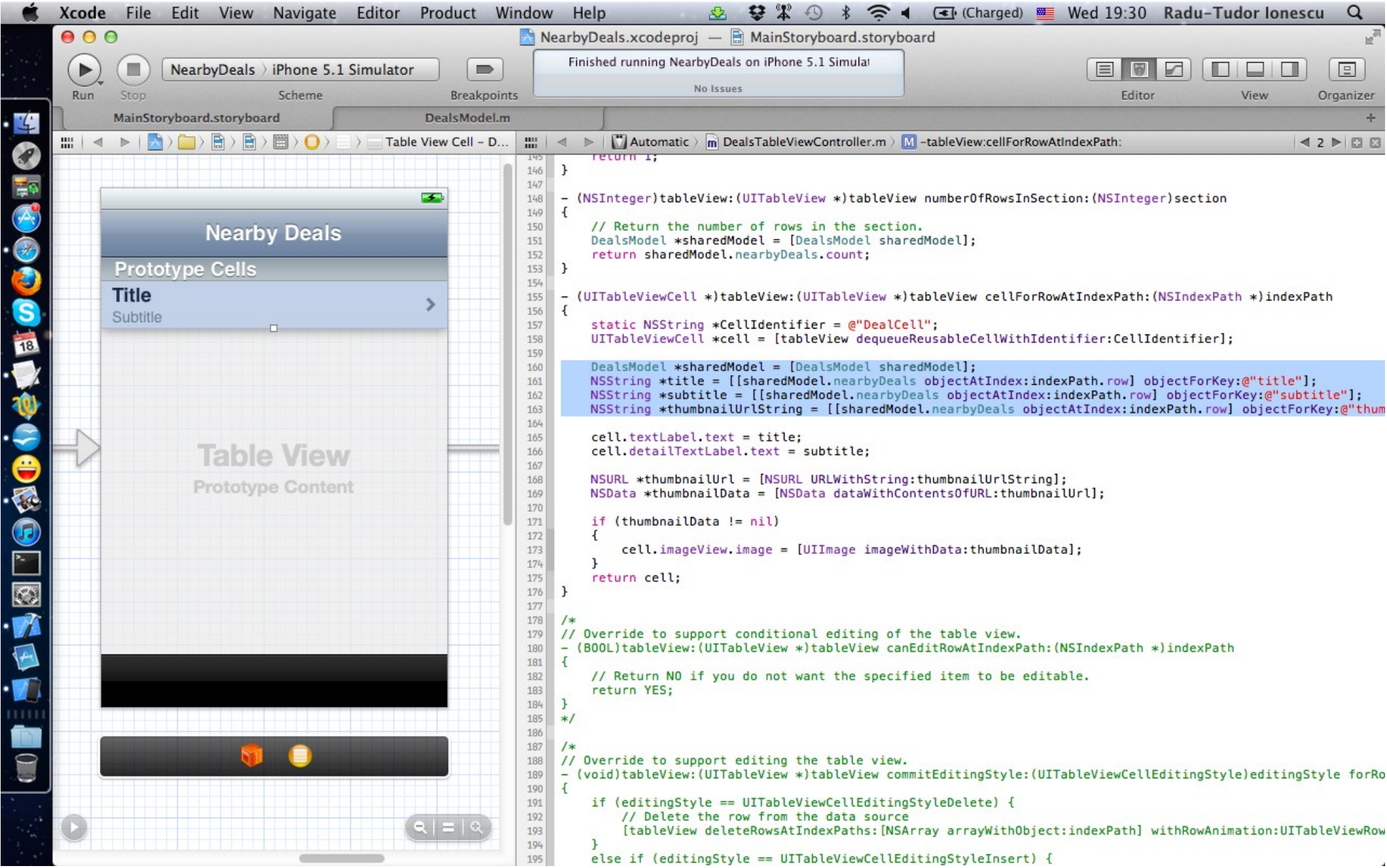
Task: Change the DealsTableViewController class so that it uses the new shared Model.

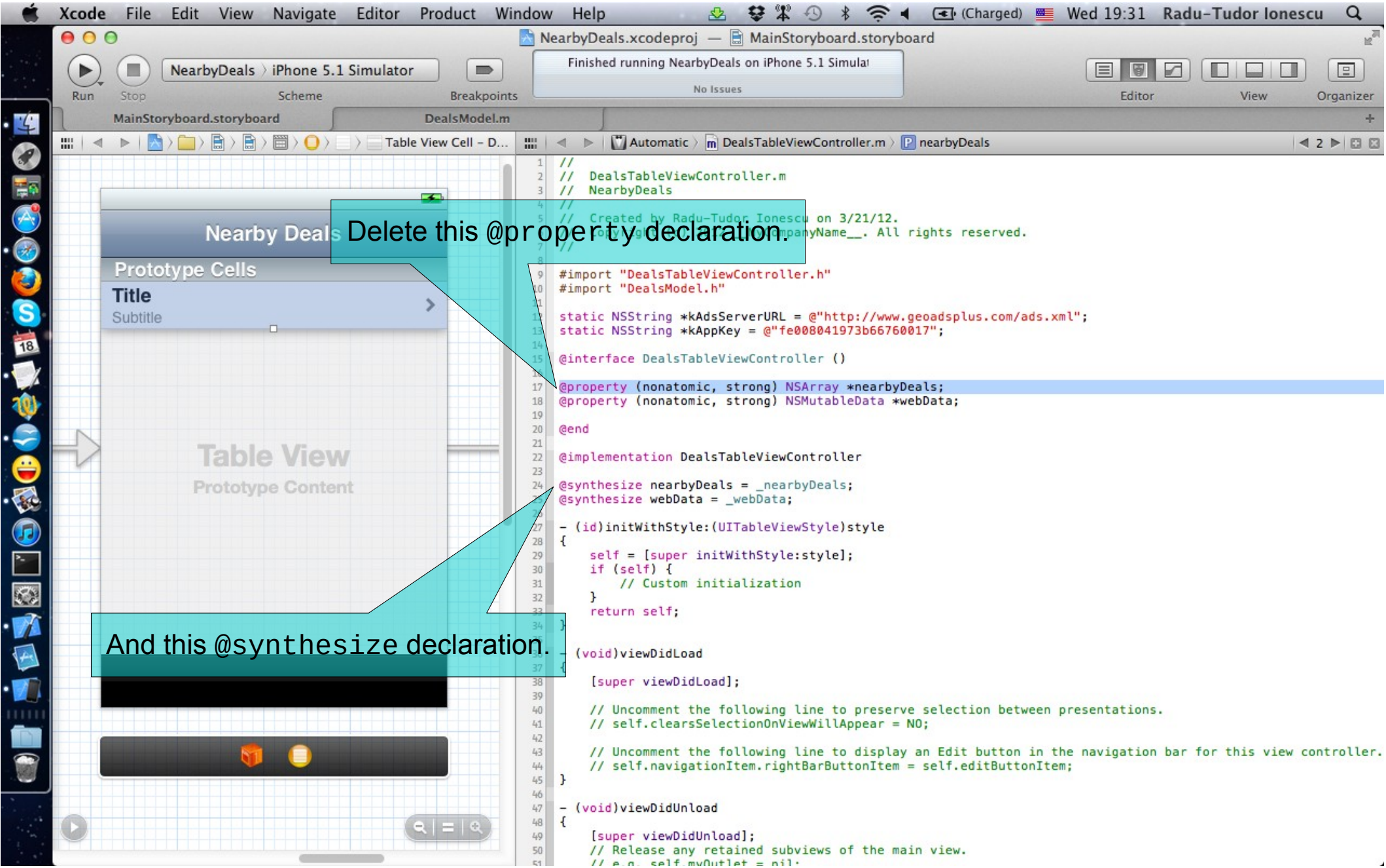
6. Build the `UITableView` cells from the `sharedModel` of the application.

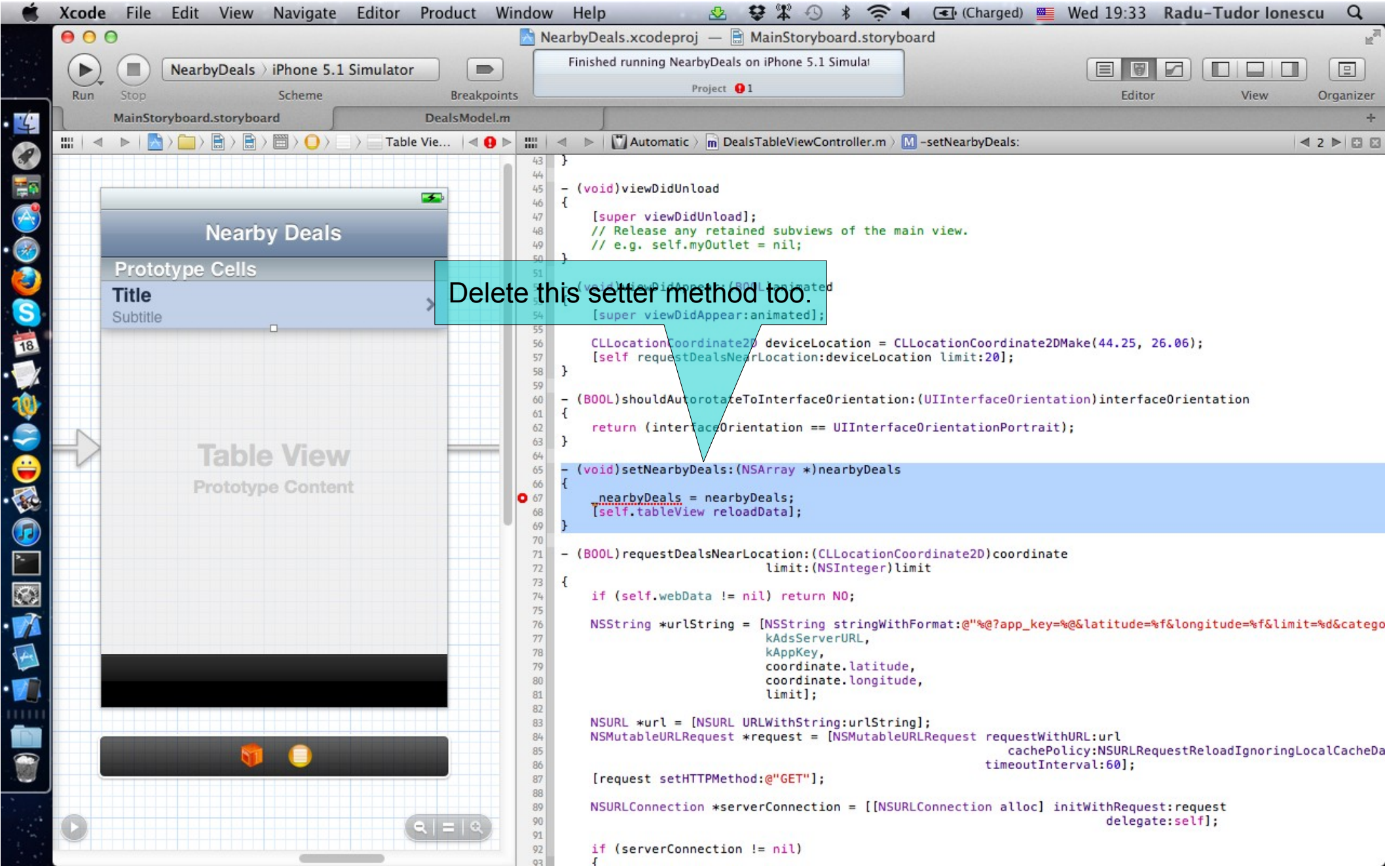
The method `tableView:cellForRowAtIndexPath:` needs to be re-implemented so that it returns `UITableViewCells` configured using the `sharedModel`'s deals array.

Look at the next screenshot to see how to re-implement this method.

7. Delete the `nearbyDeals` private `@property` from the Table View Controller since we don't use it anymore. Don't forget to delete the `@synthesize` declaration and its setter too.


Xcode File Edit View Navigate Editor Product Window Help
(Charged) Wed 19:30 Radu-Tudor Ionescu
NearbyDeals.xcodeproj — MainStoryboard.storyboard
Run Stop NearbyDeals iPhone 5.1 Simulator Scheme Breakpoints
Finished running NearbyDeals on iPhone 5.1 Simula
No Issues
Editor View Organizer
MainStoryboard.storyboard
DealsModel.m
Table View Cell – D...
Automatic DealsTableViewController.m -tableView:cellForRowAtIndexPath:
Nearby Deals
Prototype Cells
Title
Subtitle
Table View
Prototype Content
146 }
148 - (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
149 {
150 // Return the number of rows in the section.
151 DealsModel *sharedModel = [DealsModel sharedModel];
152 return sharedModel.nearbyDeals.count;
153 }
155 - (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath
156 {
157 static NSString *CellIdentifier = @"DealCell";
158 UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier];
160 DealsModel *sharedModel = [DealsModel sharedModel];
161 NSString *title = [[sharedModel.nearbyDeals objectAtIndex:indexPath.row] objectForKey:@"title"];
162 NSString *subtitle = [[sharedModel.nearbyDeals objectAtIndex:indexPath.row] objectForKey:@"subtitle"];
163 NSString *thumbnailUrlString = [[sharedModel.nearbyDeals objectAtIndex:indexPath.row] objectForKey:@"thum
165 cell.textLabel.text = title;
166 cell.detailTextLabel.text = subtitle;
168 NSURL *thumbnailUrl = [NSURL URLWithString:thumbnailUrlString];
169 NSData *thumbnailData = [NSData dataWithContentsOfURL:thumbnailUrl];
171 if (thumbnailData != nil)
172 {
173 cell.imageView.image = [UIImage imageWithData:thumbnailData];
174 }
175 return cell;
176 }
178 /*
179 // Override to support conditional editing of the table view.
180 - (BOOL)tableView:(UITableView *)tableView canEditRowAtIndexPath:(NSIndexPath *)indexPath
181 {
182 // Return NO if you do not want the specified item to be editable.
183 return YES;
184 }
185 */
187 /*
188 // Override to support editing the table view.
189 - (void)tableView:(UITableView *)tableView commitEditingStyle:(UITableViewCellEditingStyle)editingStyle forRo
190 {
191 if (editingStyle == UITableViewCellEditingStyleDelete) {
192 // Delete the row from the data source
193 [tableView deleteRowsAtIndexPaths:[NSArray arrayWithObject:indexPath] withRowAnimation:UITableViewRow
194 }
195 else if (editingStyle == UITableViewCellEditingStyleInsert) {


Delete this @property declaration.
And this @synthesize declaration.
Nearby Deals
Prototype Cells
Title
Subtitle
Table View
Prototype Content
NearbyDeals.xcodeproj — MainStoryboard.storyboard
Finished running NearbyDeals on iPhone 5.1 Simulat
No Issues
MainStoryboard.storyboard
DealsModel.m
DealsTableViewController.m
nearbyDeals
@property (nonatomic, strong) NSArray *nearbyDeals;
@property (nonatomic, strong) NSMutableData *webData;
@synthesize nearbyDeals = _nearbyDeals;
@synthesize webData = _webData;


Xcode File Edit View Navigate Editor Product Window Help
(Charged) Wed 19:33 Radu-Tudor Ionescu
NearbyDeals.xcodeproj — MainStoryboard.storyboard
Run Stop NearbyDeals iPhone 5.1 Simulator Scheme Breakpoints
Finished running NearbyDeals on iPhone 5.1 Simula
Project 1
Editor View Organizer
MainStoryboard.storyboard
DealsModel.m
Table Vie...
Automatic DealsTableViewController.m -setNearbyDeals:
Nearby Deals
Prototype Cells
Title
Subtitle
Table View
Prototype Content
Delete this setter method too.
43 }
45 - (void)viewDidUnload
46 {
47 [super viewDidUnload];
48 // Release any retained subviews of the main view.
49 // e.g. self.myOutlet = nil;
50 }
54 [super viewDidAppear:animated];
56 CLLocationCoordinate2D deviceLocation = CLLocationCoordinate2DMake(44.25, 26.06);
57 [self requestDealsNearLocation:deviceLocation limit:20];
58 }
60 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
61 {
62 return (interfaceOrientation == UIInterfaceOrientationPortrait);
63 }
65 - (void)setNearbyDeals:(NSArray *)nearbyDeals
66 {
67 _nearbyDeals = nearbyDeals;
68 [self.tableView reloadData];
69 }
71 - (BOOL)requestDealsNearLocation:(CLLocationCoordinate2D)coordinate
72 limit:(NSInteger)limit
73 {
74 if (self.webData != nil) return NO;
76 NSString *urlString = [NSString stringWithFormat:@"%@?app_key=%@&latitude=%f&longitude=%f&limit=%d&catego
77 kAdsServerURL,
78 kAppKey,
79 coordinate.latitude,
80 coordinate.longitude,
81 limit];
83 NSURL *url = [NSURL URLWithString:urlString];
84 NSMutableURLRequest *request = [NSMutableURLRequest requestWithURL:url
85 cachePolicy:NSURLRequestReloadIgnoringLocalCacheDa
86 timeoutInterval:60];
87 [request setHTTPMethod:@"GET"];
89 NSURLConnection *serverConnection = [[NSURLConnection alloc] initWithRequest:request
90 delegate:self];
92 if (serverConnection != nil)
93 {

Task 2

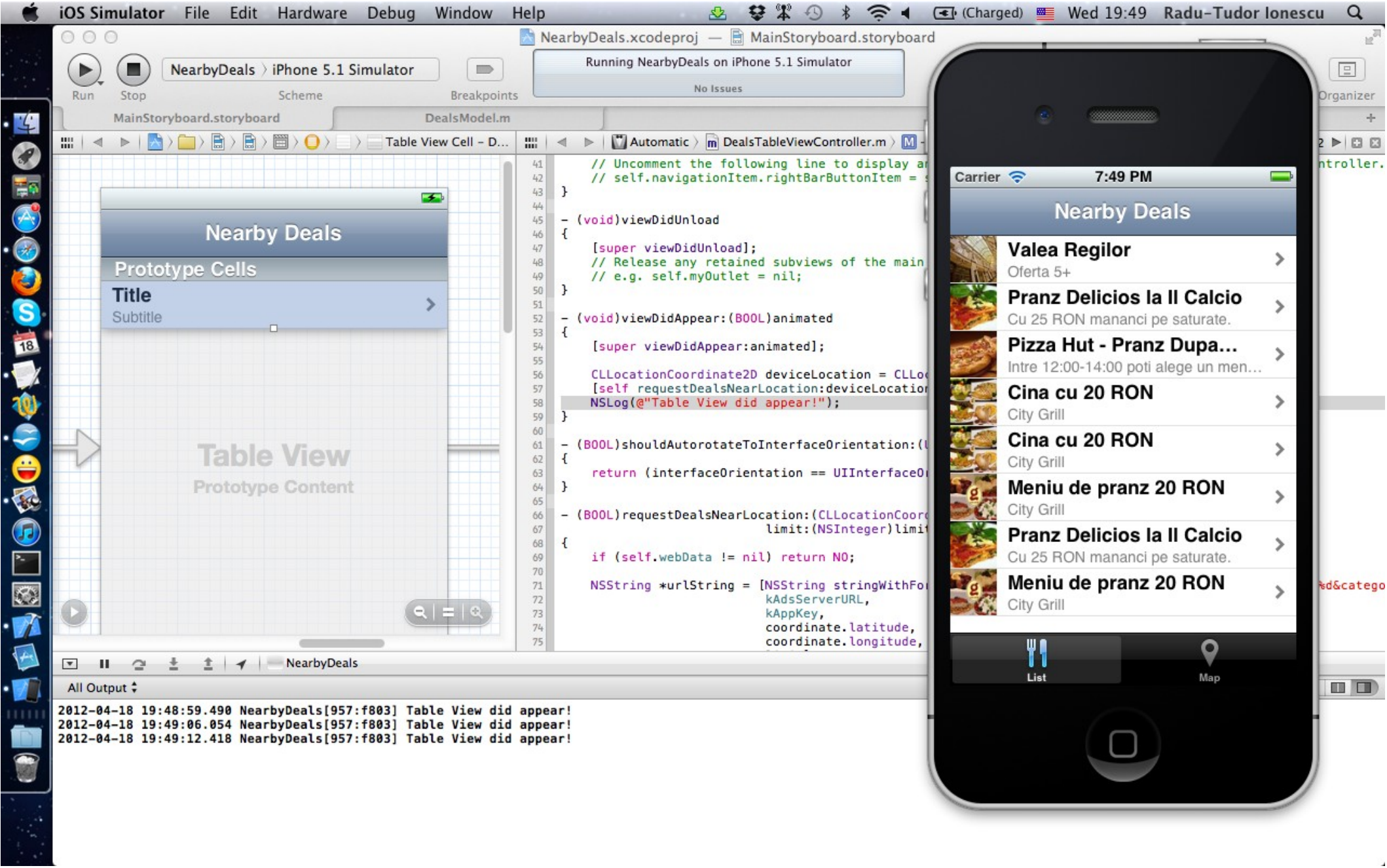
Task: Change the DealsTableViewController class so that it uses the new shared Model.

8. We have a small problem with our `sharedModel` that we need to solve. The `sharedModel` is reloaded each time the Table View Controller appears on screen. This happens when our Controller receives the `viewDidAppear:` message and makes a request to the GeoAds+ API with this line of code:

```
[self requestDealsNearLocation:deviceLocation
                         limit:20];
```

Let's put an `NSLog` in the `viewDidAppear:` method implementation to see when it gets executed. Check out the next slide for this.

9. Run the application in iOS Simulator. Change between the tabs of the application. Also try to check out details about a deal when you are in list View. Go back from the details View. Notice in the console that is `viewDidAppear:` executed several times. We don't want make server requests each time the Table View appears on screen.


iOS Simulator File Edit Hardware Debug Window Help
(Charged) Wed 19:49 Radu-Tudor Ionescu
NearbyDeals.xcodeproj — MainStoryboard.storyboard
Run Stop NearbyDeals 〉 iPhone 5.1 Simulator Scheme Breakpoints
Running NearbyDeals on iPhone 5.1 Simulator
No Issues
Organizer
MainStoryboard.storyboard
DealsModel.m
Table View Cell – D...
Automatic 〉 DealsTableViewController.m
Nearby Deals
Prototype Cells
Title
Subtitle
Table View
Prototype Content
// Uncomment the following line to display an
// self.navigationItem.rightBarButtonItem =
}
- (void)viewDidUnload
{
[super viewDidUnload];
// Release any retained subviews of the main
// e.g. self.myOutlet = nil;
}
- (void)viewDidAppear:(BOOL)animated
{
[super viewDidAppear:animated];
CLLocationCoordinate2D deviceLocation = CLLo
[self requestDealsNearLocation:deviceLocatio
NSLog(@"Table View did appear!");
}
- (BOOL)shouldAutorotateToInterfaceOrientation:(
{
return (interfaceOrientation == UIInterfaceO
}
- (BOOL)requestDealsNearLocation:(CLLocationCoor
limit:(NSInteger)limi
{
if (self.webData != nil) return NO;
NSString *urlString = [NSString stringWithFo
kAdsServerURL,
kAppKey,
coordinate.latitude,
coordinate.longitude,
NearbyDeals
All Output
2012-04-18 19:48:59.490 NearbyDeals[957:f803] Table View did appear!
2012-04-18 19:49:06.054 NearbyDeals[957:f803] Table View did appear!
2012-04-18 19:49:12.418 NearbyDeals[957:f803] Table View did appear!
Carrier 7:49 PM
Nearby Deals
Valea Regilor
Oferta 5+
Pranz Delicios la Il Calcio
Cu 25 RON mananci pe saturate.
Pizza Hut - Pranz Dupa...
Intre 12:00-14:00 poti alege un men...
Cina cu 20 RON
City Grill
Cina cu 20 RON
City Grill
Meniu de pranz 20 RON
City Grill
Pranz Delicios la Il Calcio
Cu 25 RON mananci pe saturate.
Meniu de pranz 20 RON
City Grill
List
Map

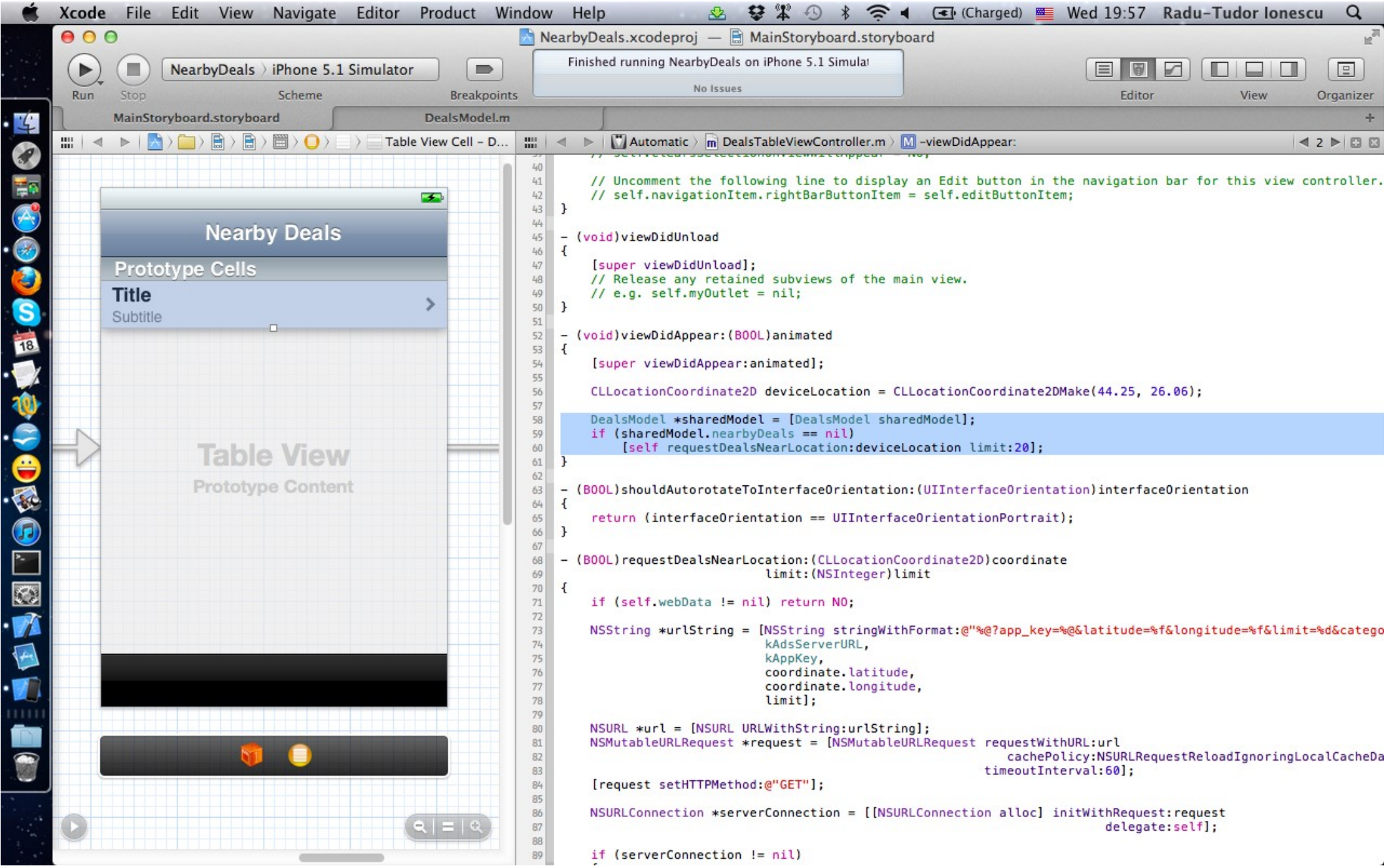
Task 2

Task: Change the DealsTableViewController class so that it uses the new shared Model.

10. Change the `viewDidAppear:` implementation in order to make server requests only when the `nearbyDeals NSArray` of the `sharedModel` is `nil`.

The idea is that we want to receive deals from the GeoAds+ server only when our local list is empty. We don't want to make server requests that are not necessary.

This method implementation can be seen on the next slide.


Xcode File Edit View Navigate Editor Product Window Help
(Charged) Wed 19:57 Radu-Tudor Ionescu
NearbyDeals.xcodeproj — MainStoryboard.storyboard
Run Stop NearbyDeals iPhone 5.1 Simulator Scheme Breakpoints
Finished running NearbyDeals on iPhone 5.1 Simula
No Issues
Editor View Organizer
MainStoryboard.storyboard
DealsModel.m
Table View Cell – D...
Automatic DealsTableViewController.m -viewDidAppear:
Nearby Deals
Prototype Cells
Title
Subtitle
Table View
Prototype Content
// Uncomment the following line to display an Edit button in the navigation bar for this view controller.
// self.navigationItem.rightBarButtonItem = self.editButtonItem;
}
- (void)viewDidUnload
{
[super viewDidUnload];
// Release any retained subviews of the main view.
// e.g. self.myOutlet = nil;
}
- (void)viewDidAppear:(BOOL)animated
{
[super viewDidAppear:animated];
CLLocationCoordinate2D deviceLocation = CLLocationCoordinate2DMake(44.25, 26.06);
DealsModel *sharedModel = [DealsModel sharedModel];
if (sharedModel.nearbyDeals == nil)
[self requestDealsNearLocation:deviceLocation limit:20];
}
- (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
{
return (interfaceOrientation == UIInterfaceOrientationPortrait);
}
- (BOOL)requestDealsNearLocation:(CLLocationCoordinate2D)coordinate
limit:(NSInteger)limit
{
if (self.webData != nil) return NO;
NSString *urlString = [NSString stringWithFormat:@"%@?app_key=%@&latitude=%f&longitude=%f&limit=%d&catego
kAdsServerURL,
kAppKey,
coordinate.latitude,
coordinate.longitude,
limit];
NSURL *url = [NSURL URLWithString:urlString];
NSMutableURLRequest *request = [NSMutableURLRequest requestWithURL:url
cachePolicy:NSURLRequestReloadIgnoringLocalCacheDa
timeoutInterval:60];
[request setHTTPMethod:@"GET"];
NSURLConnection *serverConnection = [[NSURLConnection alloc] initWithRequest:request
delegate:self];
if (serverConnection != nil)

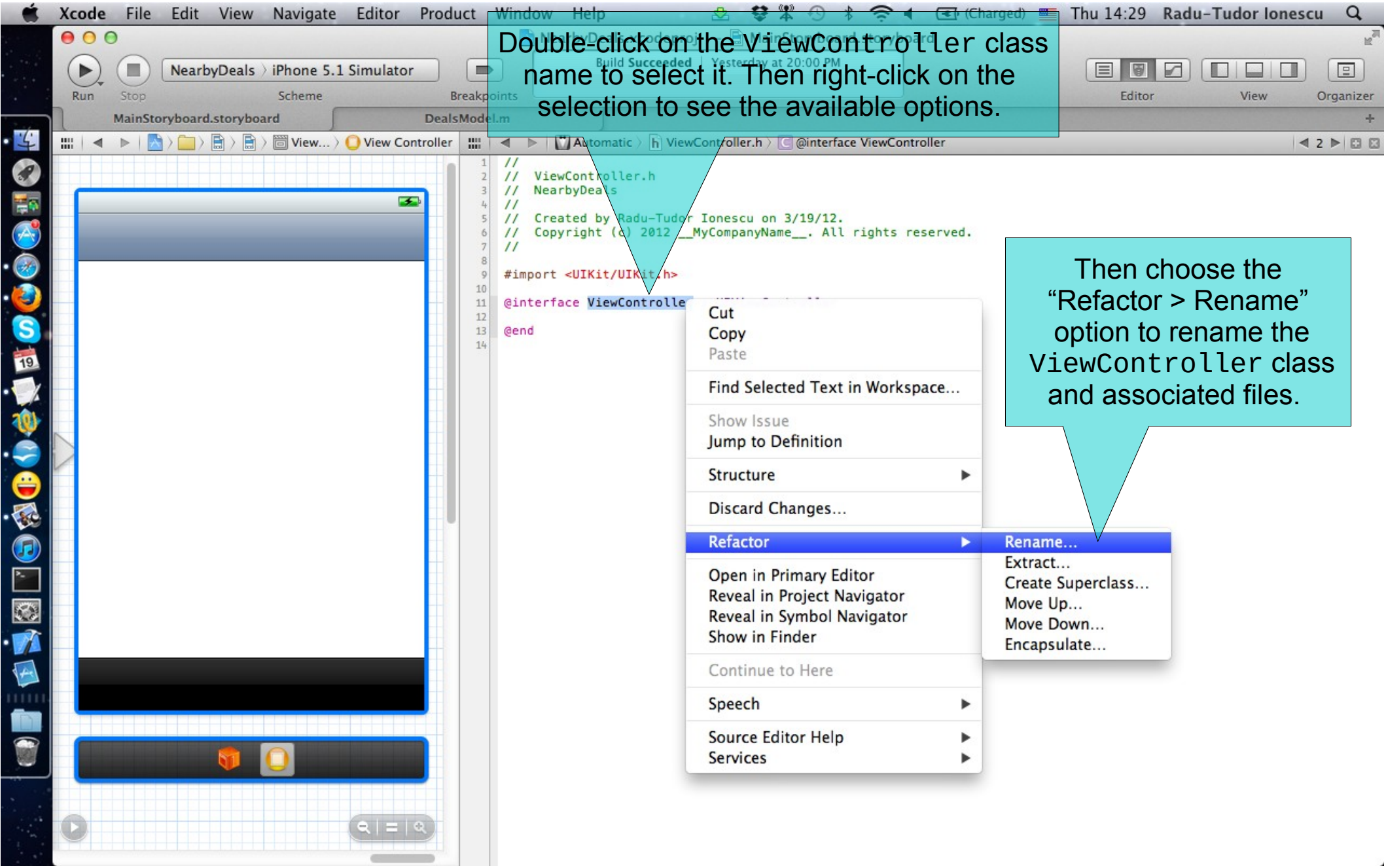
Task 3

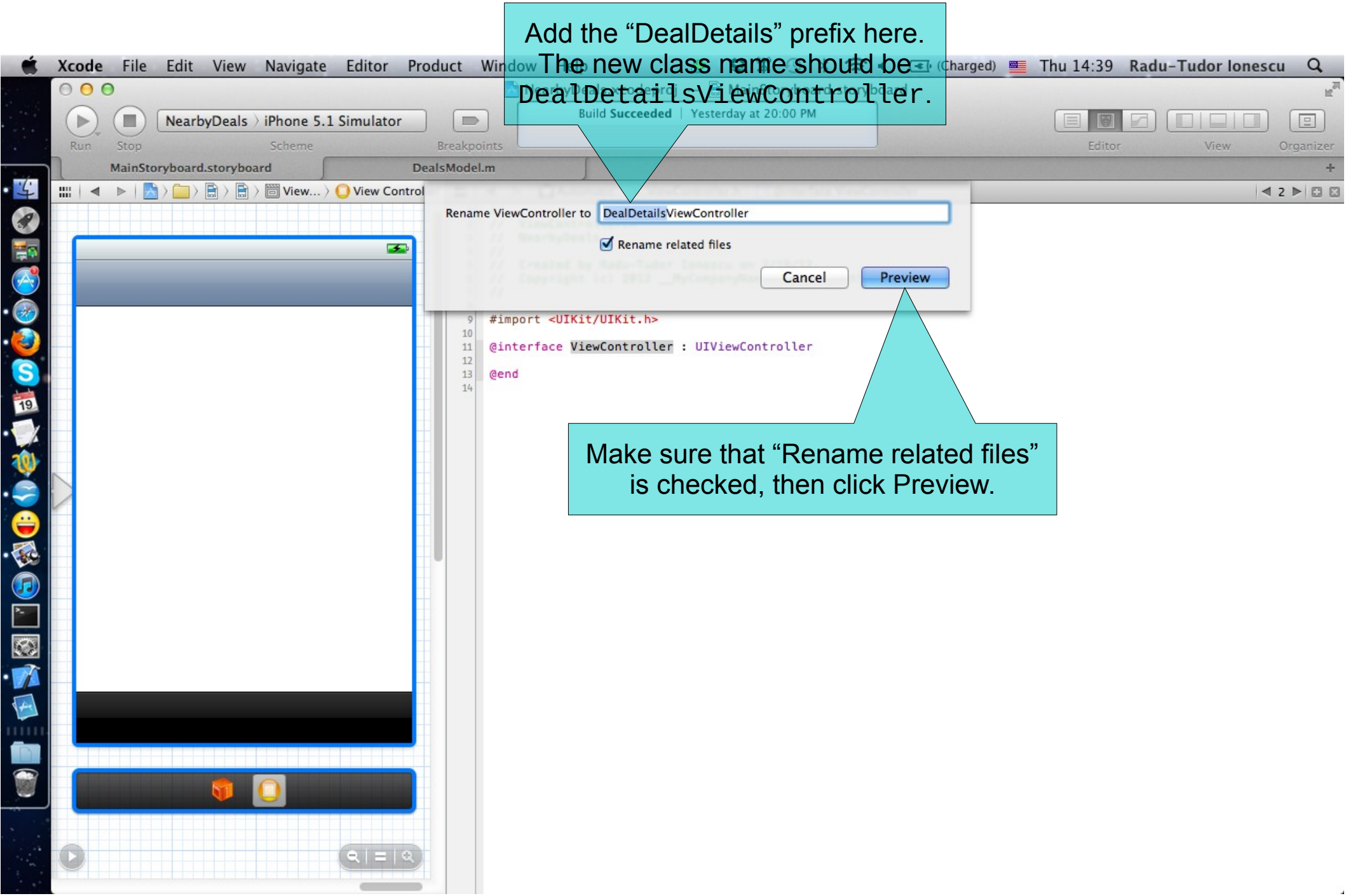
Task: Configure the View Controller that presents deal details.

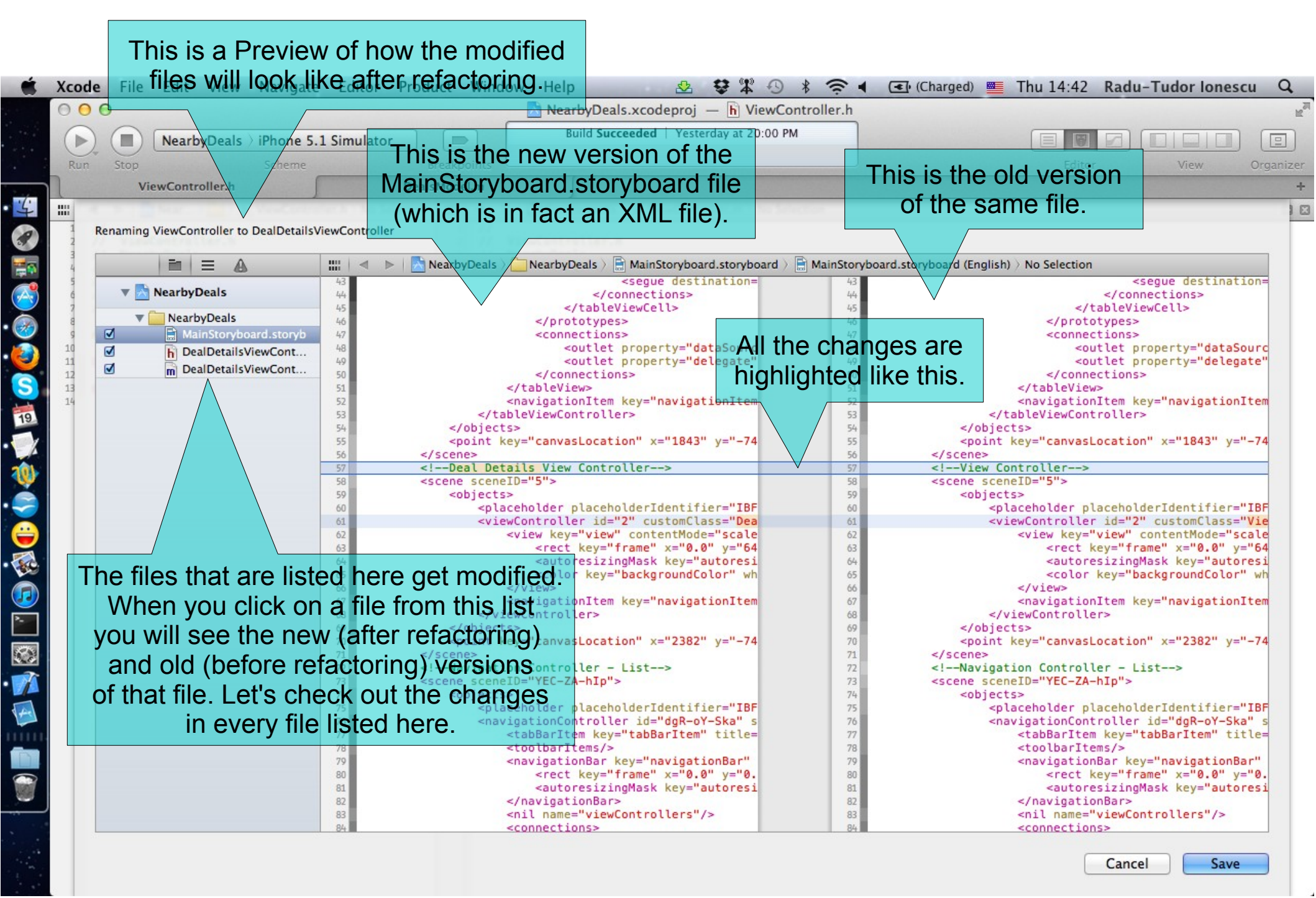
1. Our Table View Controller segues to a View Controller that presents details about a deal. This View Controller from Interface Builder is associated with the `ViewController` class. The name of this class is too generic. Let's add the “DealDetails” prefix to make it more specific.

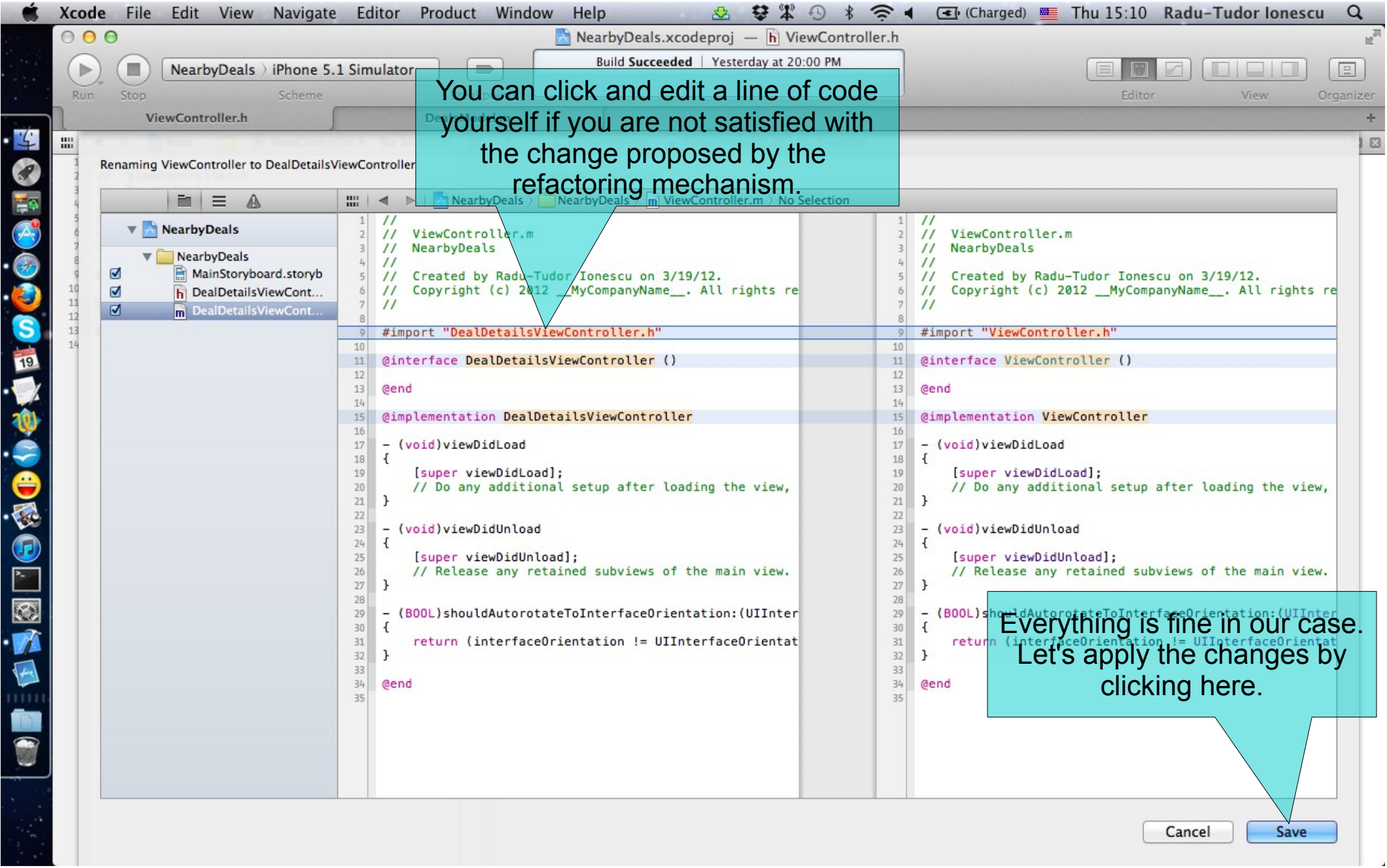
Switch to the MainStoryboard.storyboard tab in Xcode. Select the View Controller that should display details about a deal.

2. Follow the steps from the following slides to understand how to refactor the `ViewController` class.


Double-click on the ViewController class name to select it. Then right-click on the selection to see the available options.
Then choose the "Refactor > Rename" option to rename the ViewController class and associated files.
Cut
Copy
Paste
Find Selected Text in Workspace...
Show Issue
Jump to Definition
Structure
Discard Changes...
Refactor
Open in Primary Editor
Reveal in Project Navigator
Reveal in Symbol Navigator
Show in Finder
Continue to Here
Speech
Source Editor Help
Services
Rename...
Extract...
Create Superclass...
Move Up...
Move Down...
Encapsulate...


Add the “DealDetails” prefix here.
The new class name should be DealDetailsViewController.
Rename ViewController to DealDetailsViewController
Rename related files
Cancel
Preview
Make sure that “Rename related files” is checked, then click Preview.
#import <UIKit/UIKit.h>
@interface ViewController : UIViewController
@end


This is a Preview of how the modified files will look like after refactoring.
This is the new version of the MainStoryboard.storyboard file (which is in fact an XML file).
This is the old version of the same file.
All the changes are highlighted like this.
The files that are listed here get modified. When you click on a file from this list you will see the new (after refactoring) and old (before refactoring) versions of that file. Let's check out the changes in every file listed here.
Renaming ViewController to DealDetailsViewController
NearbyDeals.xcodeproj — ViewController.h
Build Succeeded | Yesterday at 20:00 PM
Cancel
Save


You can click and edit a line of code yourself if you are not satisfied with the change proposed by the refactoring mechanism.
Everything is fine in our case. Let's apply the changes by clicking here.
Renaming ViewController to DealDetailsViewController
Cancel
Save

Task 3

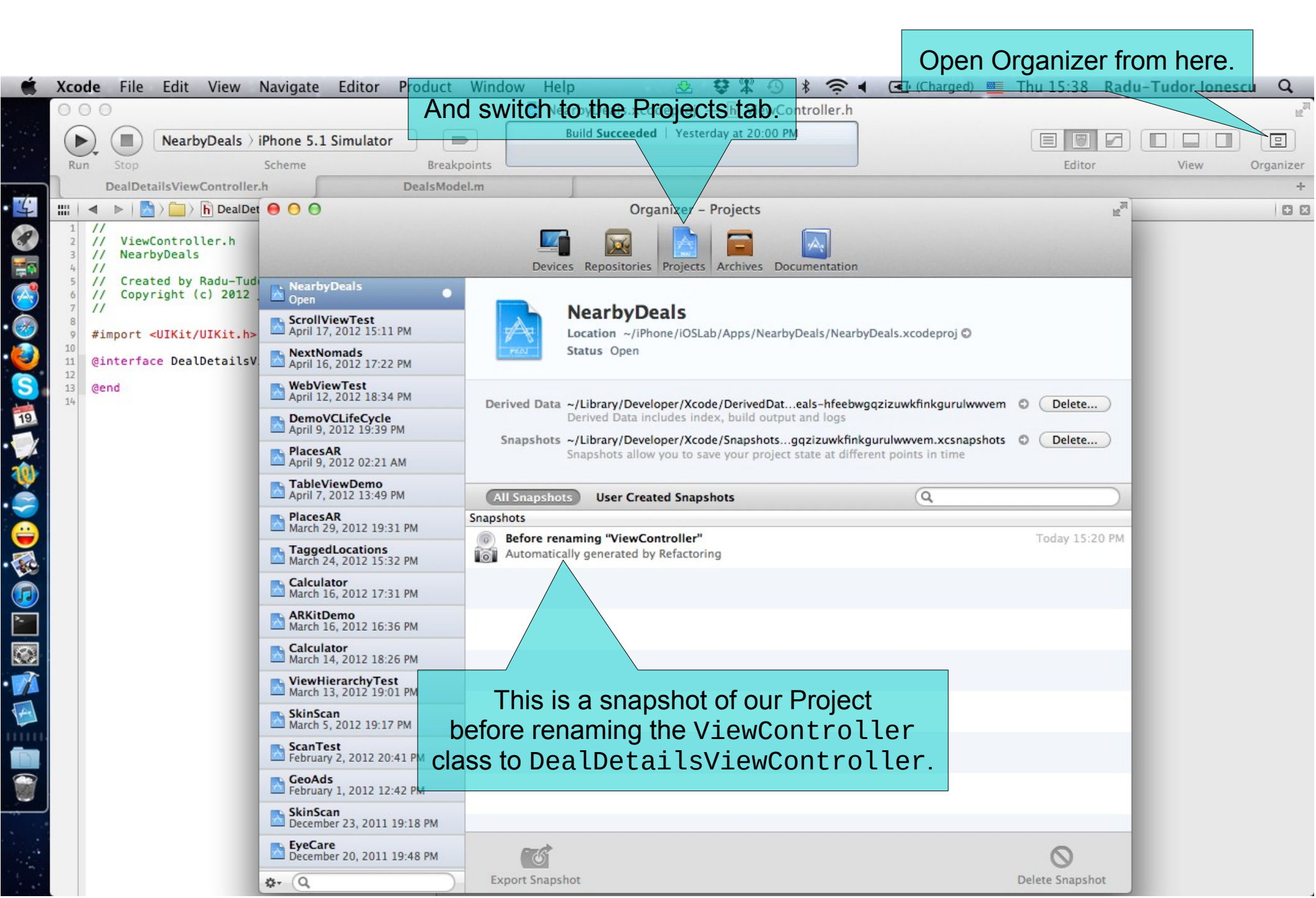
Task: Configure the View Controller that presents deal details.

3. You may be asked to create a **snapshot** of your Project. Click Enable and continue. By default Xcode creates a snapshot automatically before a major change such as refactoring your code or executing a find & replace operation. In general, you can change this option from "File > Project Settings (or Workspace Settings)".

Note that you can also create a snapshot manually by choosing "File > Create Snapshot". To see the snapshots for a project or workspace, click the project in the Projects pane of the Organizer window.

To restore a snapshot, choose Restore Snapshot from the File menu and select the snapshot to restore. When you click Restore, Xcode replaces the current version of the project with the version in the snapshot. Xcode makes a snapshot of the current version before replacing it.

4. Let's open Organizer and see our Project's Snapshots.


Open Organizer from here.
And switch to the Projects tab.
Xcode File Edit View Navigate Editor Product Window Help
Run Stop Scheme Breakpoints Editor View Organizer
NearbyDeals iPhone 5.1 Simulator
Build Succeeded Yesterday at 20:00 PM
DealDetailsViewController.h
DealsModel.m
Organizer – Projects
Devices Repositories Projects Archives Documentation
NearbyDeals
Open
ScrollViewTest
April 17, 2012 15:11 PM
NextNomads
April 16, 2012 17:22 PM
WebViewTest
April 12, 2012 18:34 PM
DemoVCLifeCycle
April 9, 2012 19:39 PM
PlacesAR
April 9, 2012 02:21 AM
TableViewDemo
April 7, 2012 13:49 PM
PlacesAR
March 29, 2012 19:31 PM
TaggedLocations
March 24, 2012 15:32 PM
Calculator
March 16, 2012 17:31 PM
ARKitDemo
March 16, 2012 16:36 PM
Calculator
March 14, 2012 18:26 PM
ViewHierarchyTest
March 13, 2012 19:01 PM
SkinScan
March 5, 2012 19:17 PM
ScanTest
February 2, 2012 20:41 PM
GeoAds
February 1, 2012 12:42 PM
SkinScan
December 23, 2011 19:18 PM
EyeCare
December 20, 2011 19:48 PM
NearbyDeals
Location ~/iPhone/iOSLab/Apps/NearbyDeals/NearbyDeals.xcodeproj
Status Open
Derived Data ~/Library/Developer/Xcode/DerivedDat...eals-hfeebwgqzizuwkfinkgurulwwvem
Derived Data includes index, build output and logs
Delete...
Snapshots ~/Library/Developer/Xcode/Snapshots...gqzizuwkfinkgurulwwvem.xcsnapshots
Snapshots allow you to save your project state at different points in time
Delete...
All Snapshots User Created Snapshots
Snapshots
Before renaming "ViewController"
Automatically generated by Refactoring
Today 15:20 PM
Export Snapshot
Delete Snapshot
This is a snapshot of our Project before renaming the ViewController class to DealDetailsViewController.

Task 3

Task: Configure the View Controller that presents deal details.

5. Close Organizer and continue with the configuration of the Deal Details View Controller from Interface Builder.

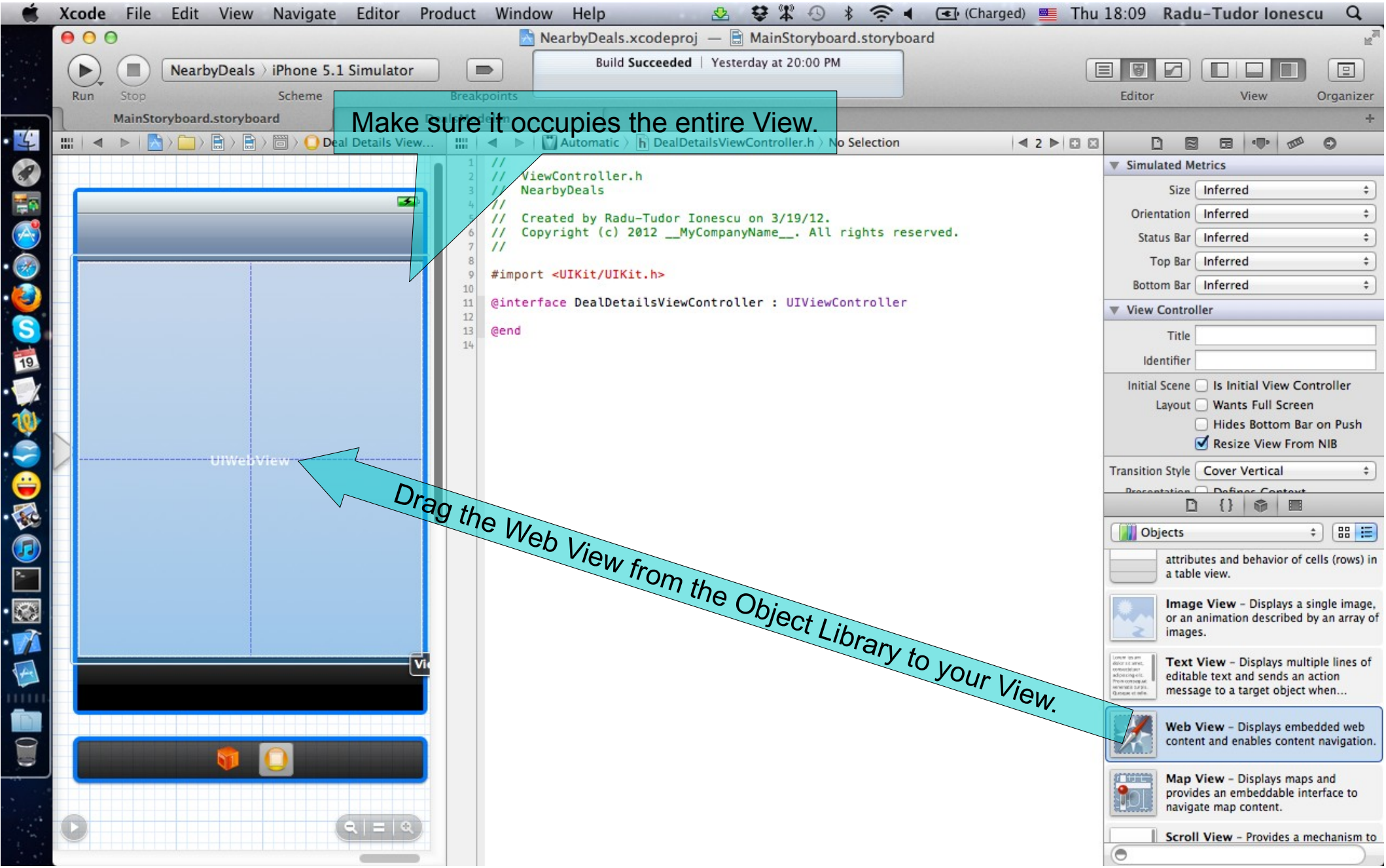
Select the MainStoryboard.storyboard file (if it's not already selected) to make some changes.

6. The URL of a deal opens a HTML page with details about that deal. This is exactly what we need. We will add an `UIWebView` to our View Controller and open the URL when the View appears on screen.

7. Open Utilities area.

8. Drag and drop an `UIWebView` from Object Library. Make sure it occupies the entire View before you drop it.

Check out the next screenshot to see how the `UIWebView` should look like.


Make sure it occupies the entire View.
Drag the Web View from the Object Library to your View.
UIWebView
Web View – Displays embedded web content and enables content navigation.

Task 3

Task: Configure the View Controller that presents deal details.

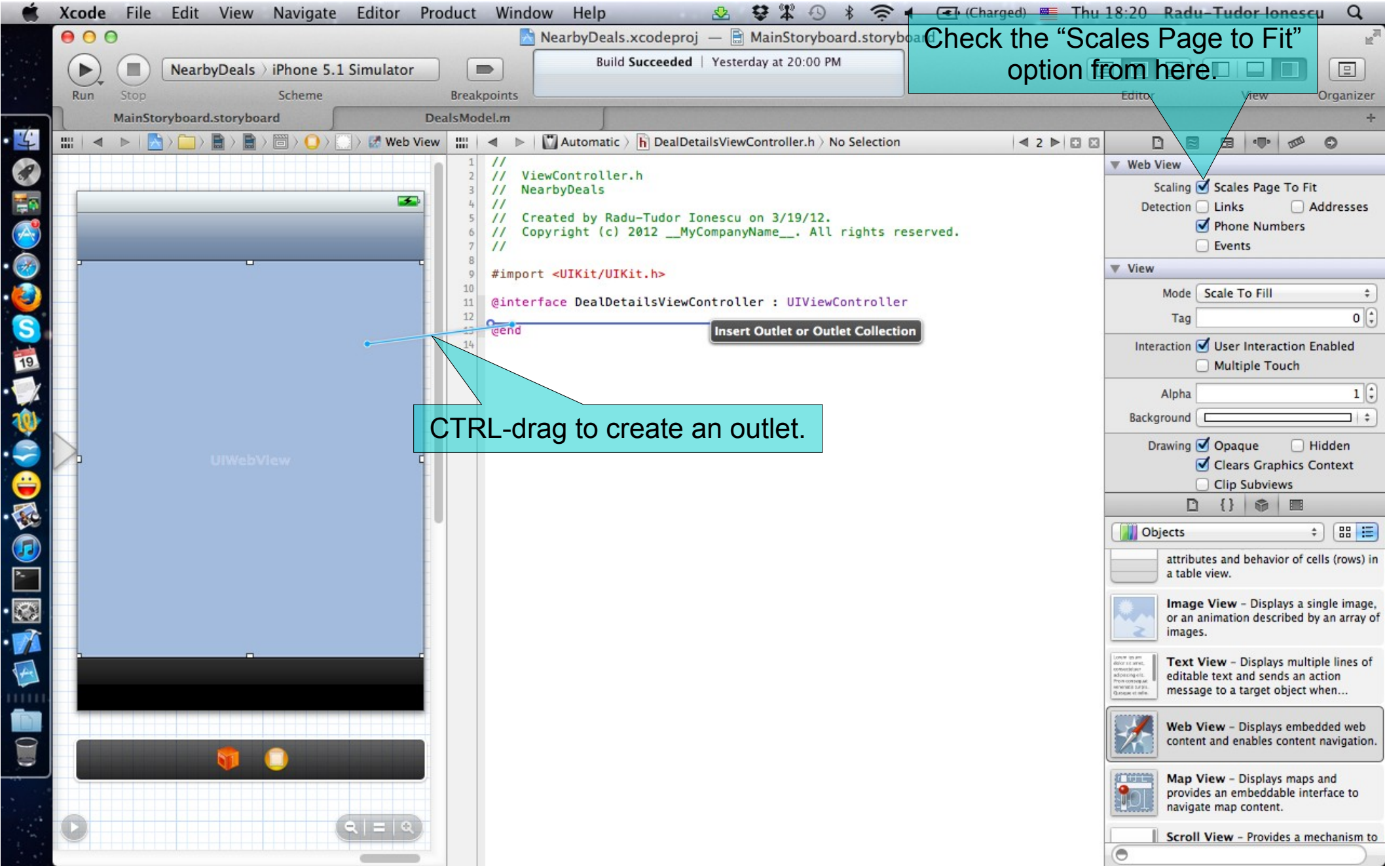
9. Check the “Scales Page to Fit” option in Attributes Inspector.

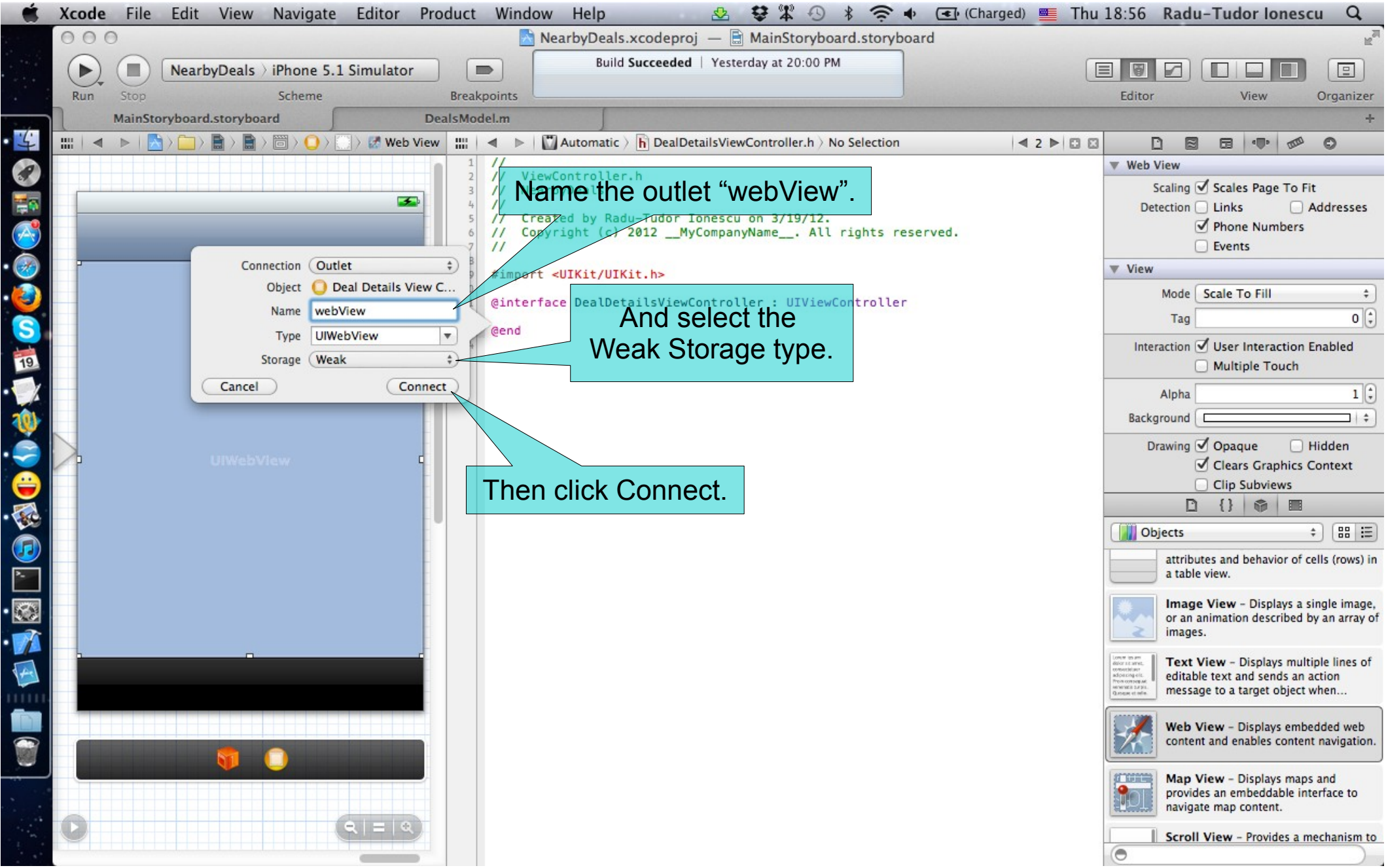
This will set the `scalesPageToFit` property to `YES`. In this case, the webpage is scaled to fit the View and the user can zoom in and zoom out. If `scalesPageToFit` is `NO`, user zooming is disabled. The default value is `NO`.

10. Select DealDetailsViewController.h in Assistant Editor.

11. CTRL-drag from the `UIWebView` to the `@interface` block inside DealDetailsViewController.h to create an outlet for this Web View.

Follow the steps from the next slides to finish adding the outlet.


Check the “Scales Page to Fit” option from here.
CTRL-drag to create an outlet.
Insert Outlet or Outlet Collection
Xcode File Edit View Navigate Editor Product Window Help
NearbyDeals.xcodeproj — MainStoryboard.storyboard
Build Succeeded | Yesterday at 20:00 PM
Run Stop Scheme Breakpoints
MainStoryboard.storyboard
DealsModel.m
Web View
Automatic DealDetailsViewController.h No Selection
UIWebView
// ViewController.h
// NearbyDeals
// Created by Radu-Tudor Ionescu on 3/19/12.
// Copyright (c) 2012 __MyCompanyName__. All rights reserved.
#import <UIKit/UIKit.h>
@interface DealDetailsViewController : UIViewController
@end
Web View
Scaling Scales Page To Fit
Detection Links Addresses
Phone Numbers
Events
View
Mode Scale To Fill
Tag 0
Interaction User Interaction Enabled
Multiple Touch
Alpha 1
Background
Drawing Opaque Hidden
Clears Graphics Context
Clip Subviews
Objects
attributes and behavior of cells (rows) in a table view.
Image View – Displays a single image, or an animation described by an array of images.
Text View – Displays multiple lines of editable text and sends an action message to a target object when...
Web View – Displays embedded web content and enables content navigation.
Map View – Displays maps and provides an embeddable interface to navigate map content.
Scroll View – Provides a mechanism to


Name the outlet “webView”.
And select the Weak Storage type.
Then click Connect.
Connection Outlet
Object Deal Details View C...
Name webView
Type UIWebView
Storage Weak
Cancel
Connect
UIWebView
Web View
Scaling Scales Page To Fit
Detection Links Addresses
Phone Numbers
Events
View
Mode Scale To Fill
Tag 0
Interaction User Interaction Enabled
Multiple Touch
Alpha 1
Background
Drawing Opaque Hidden
Clears Graphics Context
Clip Subviews
Objects
Image View – Displays a single image, or an animation described by an array of images.
Text View – Displays multiple lines of editable text and sends an action message to a target object when...
Web View – Displays embedded web content and enables content navigation.
Map View – Displays maps and provides an embeddable interface to navigate map content.
Scroll View – Provides a mechanism to

Task 3

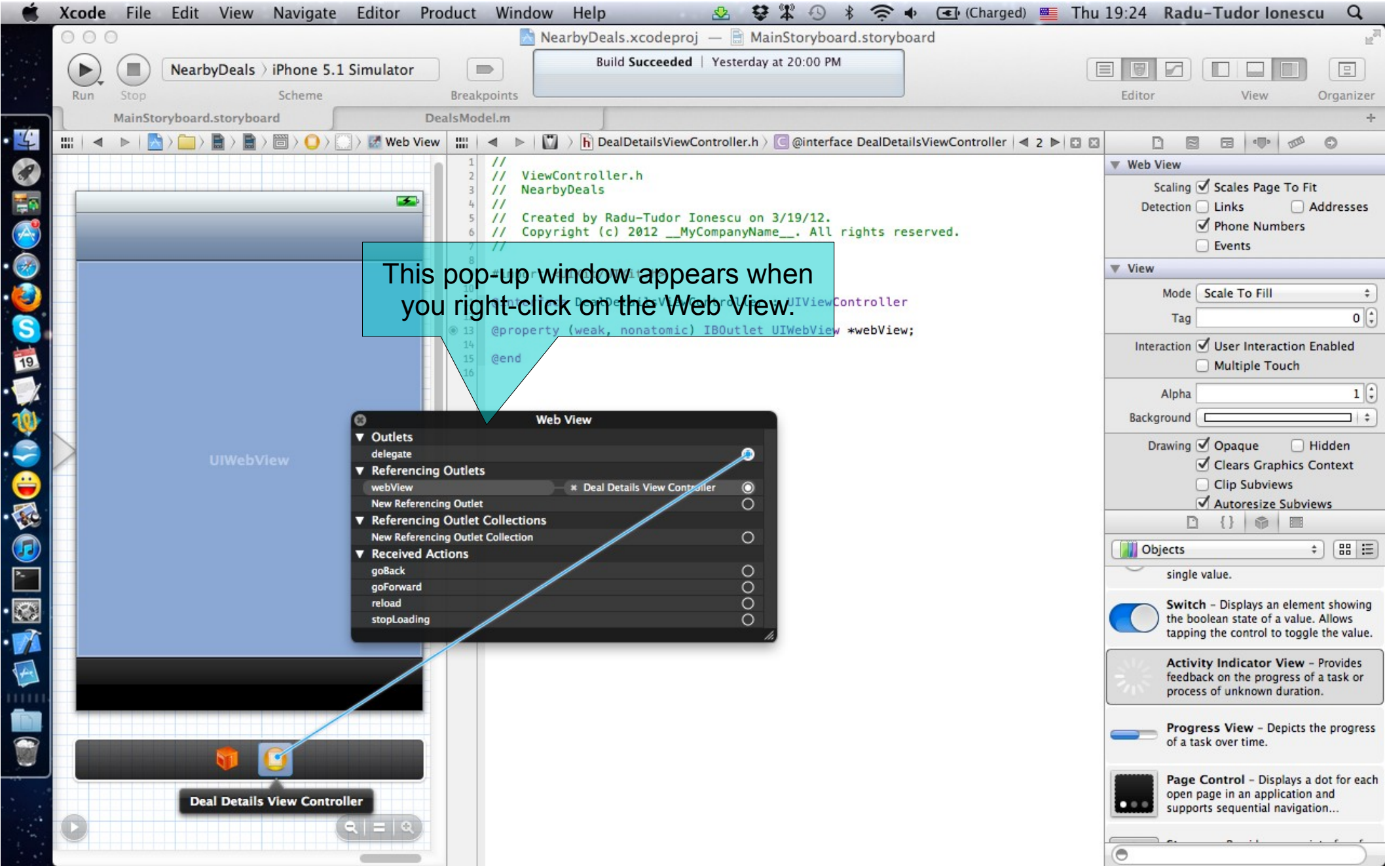
Task: Configure the View Controller that presents deal details.

12. We want to load a webpage inside our Web View from the deal's URL address. To let the user know that we are about to load a webpage in our View Controller, we will use an Activity Indicator. While we are loading the webpage, this Activity Indicator will animate a spinning wheel. We have to stop this animation when the page is loaded. Therefore, our View Controller should be set as the `UIWebView`'s delegate object and it must conform to the `UIWebViewDelegate` protocol (we want to add custom behavior to our Web View).

In Interface Builder right-click on the Web View.

13. Drag from the delegate outlet to the View Controller icon placed under the View of this View Controller.

Look at the next screenshot to understand exactly how to do this.


This pop-up window appears when you right-click on the Web View.
Web View
Outlets
delegate
Referencing Outlets
webView
Deal Details View Controller
New Referencing Outlet
Referencing Outlet Collections
New Referencing Outlet Collection
Received Actions
goBack
goForward
reload
stopLoading
UIWebView
Deal Details View Controller

Task 3

Task: Configure the View Controller that presents deal details.

14. Declare that the Deal Details View Controller conforms to the `UIWebViewDelegate` protocol right after the `UIViewController` superclass in the `@interface` declaration.

We will implement the methods from the `UIWebViewDelegate` protocol in a moment.

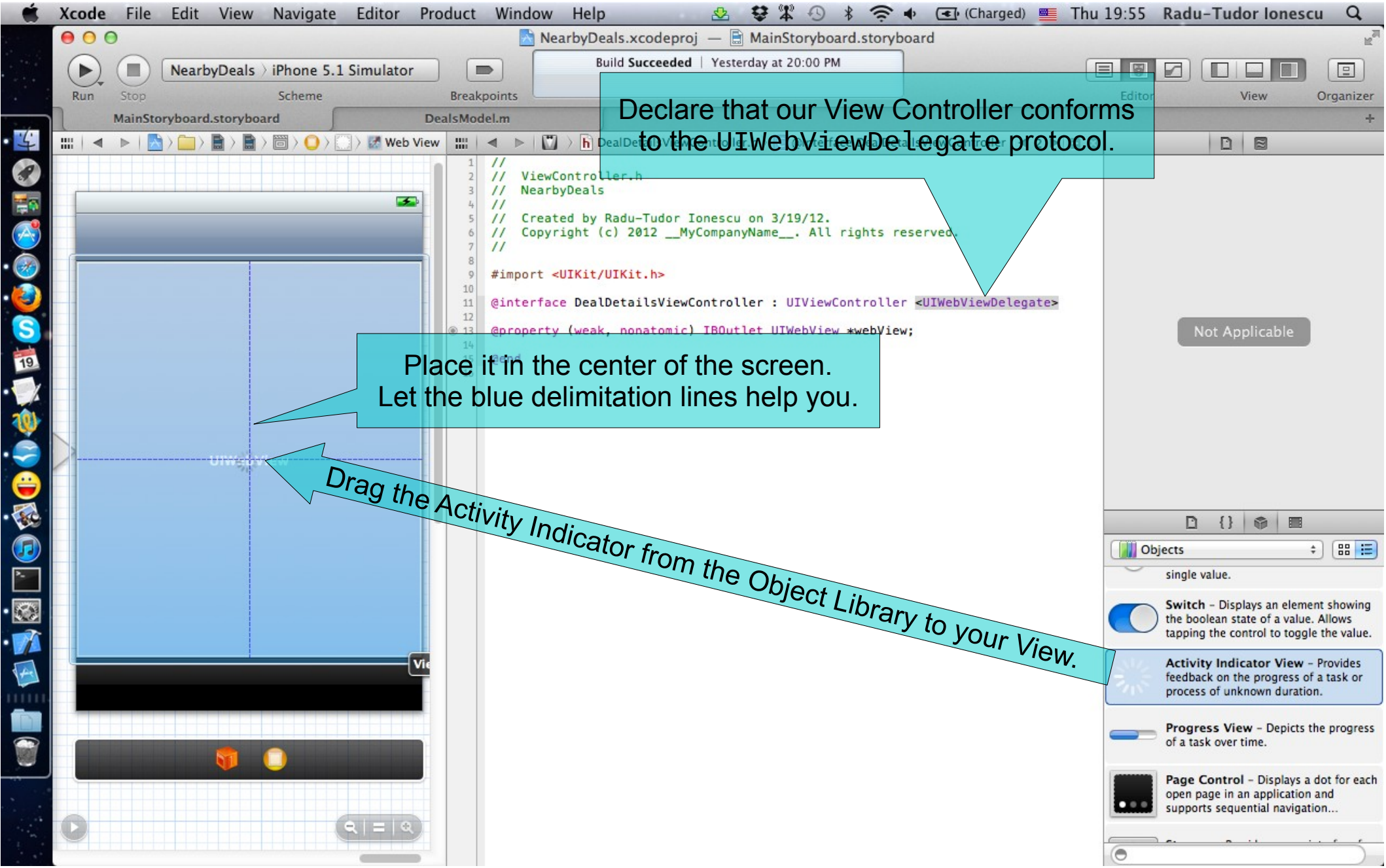
15. Drag and drop an Activity Indicator View from Object Library. Place it over the Web View in the center of the screen.

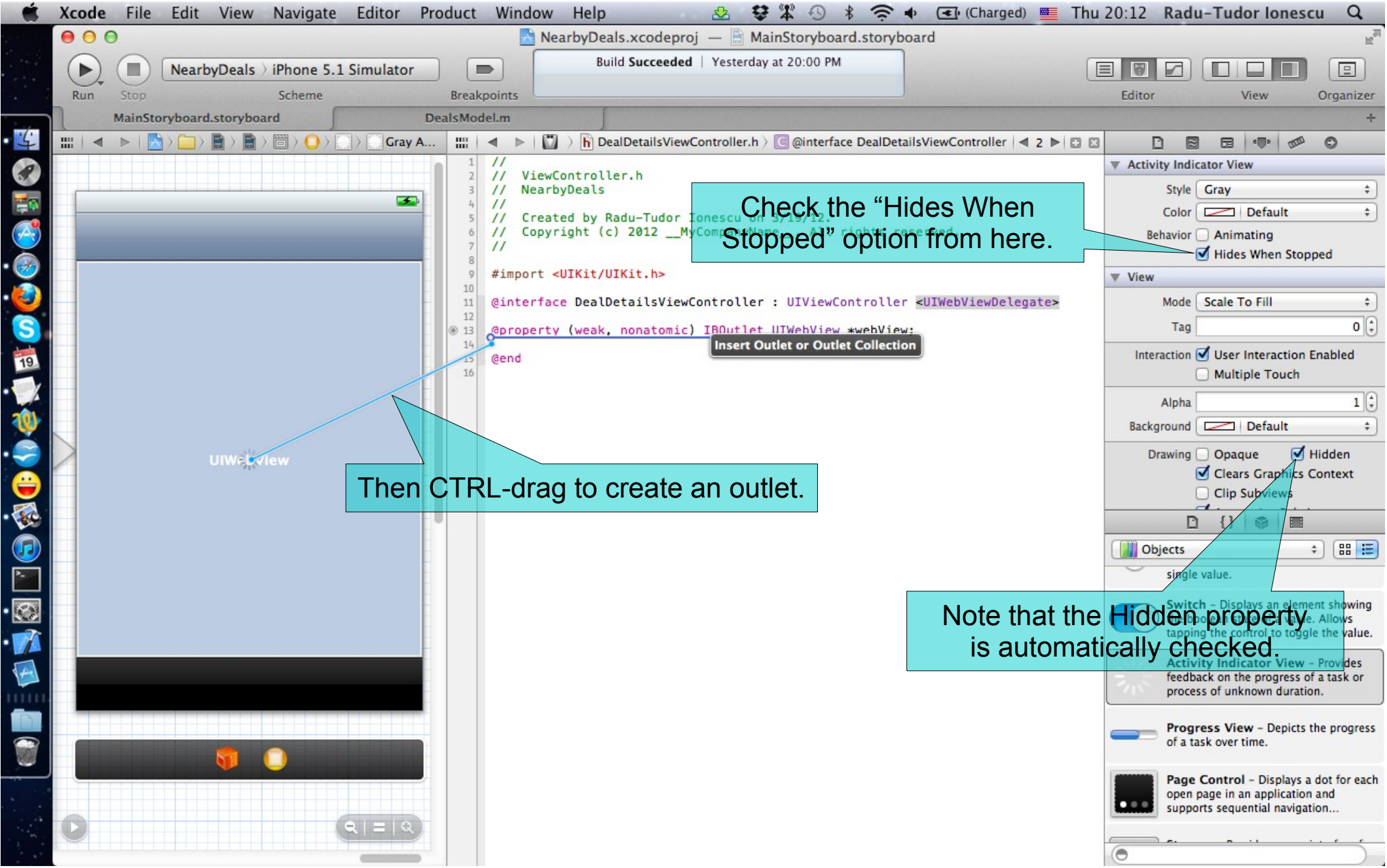
16. Check the “Hides When Stopped” option. Watch what happens with the Hidden property of the Activity Indicator.

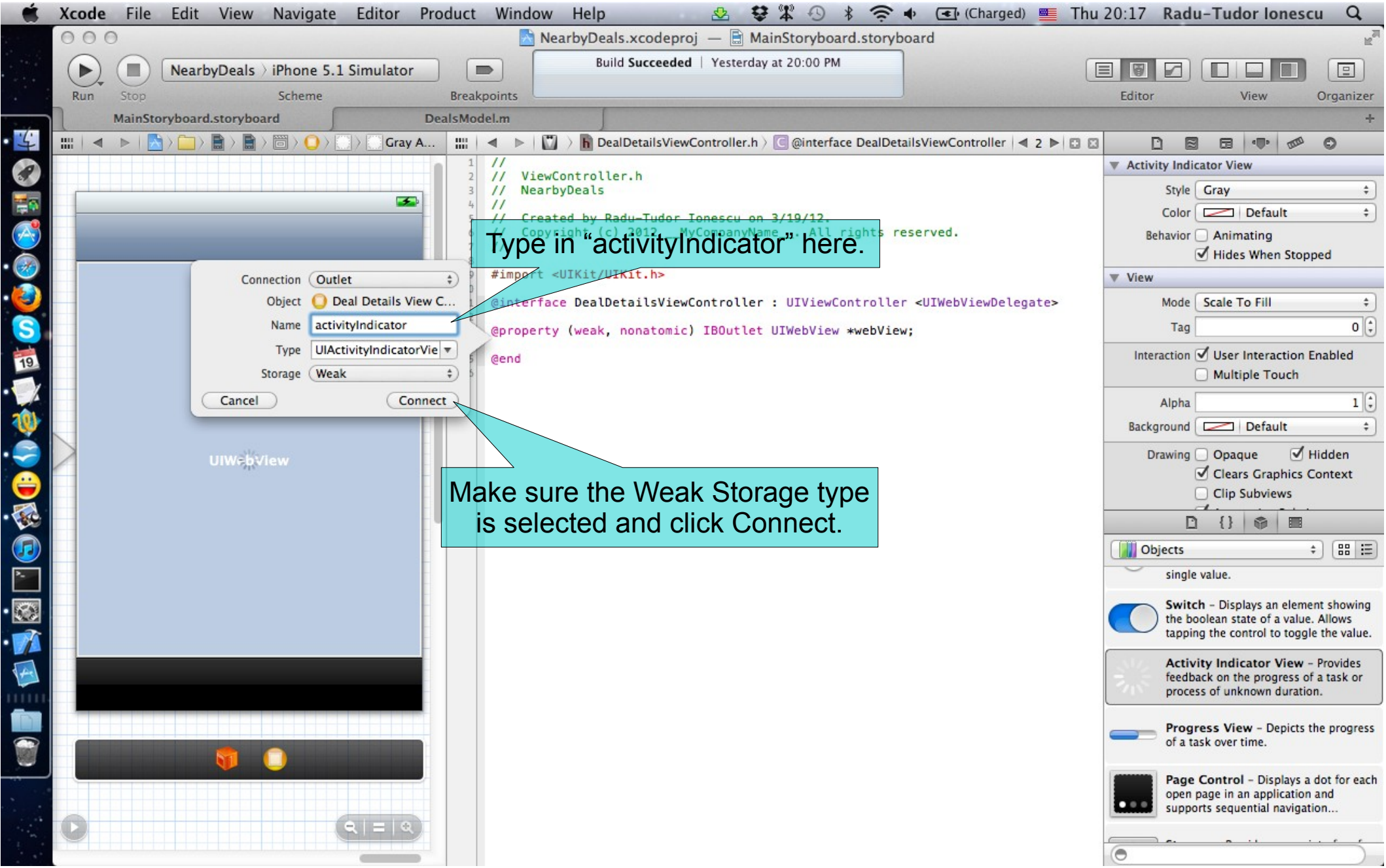
17. CTRL-drag from the `UIActivityIndicatorView` to the DealDetailsViewController `@interface` to create an outlet.

18. Name this outlet “activityIndicator” and select the Weak Storage.

Look over the next slides for hints.


Declare that our View Controller conforms to the UIWebViewDelegate protocol.
Place it in the center of the screen.
Let the blue delimitation lines help you.
Drag the Activity Indicator from the Object Library to your View.
Not Applicable
Objects
Switch – Displays an element showing the boolean state of a value. Allows tapping the control to toggle the value.
Activity Indicator View – Provides feedback on the progress of a task or process of unknown duration.
Progress View – Depicts the progress of a task over time.
Page Control – Displays a dot for each open page in an application and supports sequential navigation...


Xcode File Edit View Navigate Editor Product Window Help
NearbyDeals.xcodeproj — MainStoryboard.storyboard
Build Succeeded | Yesterday at 20:00 PM
MainStoryboard.storyboard
DealsModel.m
DealDetailsViewController.h
@interface DealDetailsViewController
Insert Outlet or Outlet Collection
Activity Indicator View
Style Gray
Color Default
Behavior Animating
Hides When Stopped
View
Mode Scale To Fill
Tag 0
Interaction User Interaction Enabled
Multiple Touch
Alpha 1
Background Default
Drawing Opaque Hidden
Clears Graphics Context
Clip Subviews
Objects
Check the “Hides When Stopped” option from here.
Then CTRL-drag to create an outlet.
Note that the Hidden property is automatically checked.


Type in “activityIndicator” here.
Make sure the Weak Storage type is selected and click Connect.
Connection Outlet
Object Deal Details View C...
Name activityIndicator
Type UIActivityIndicatorVie
Storage Weak
Cancel
Connect
UIWebView
Activity Indicator View
Style Gray
Color Default
Behavior Animating
Hides When Stopped
View
Mode Scale To Fill
Tag 0
Interaction User Interaction Enabled
Multiple Touch
Alpha 1
Background Default
Drawing Opaque Hidden
Clears Graphics Context
Clip Subviews
Objects
Switch – Displays an element showing the boolean state of a value. Allows tapping the control to toggle the value.
Activity Indicator View – Provides feedback on the progress of a task or process of unknown duration.
Progress View – Depicts the progress of a task over time.
Page Control – Displays a dot for each open page in an application and supports sequential navigation...

Task 3

Task: Configure the View Controller that presents deal details.

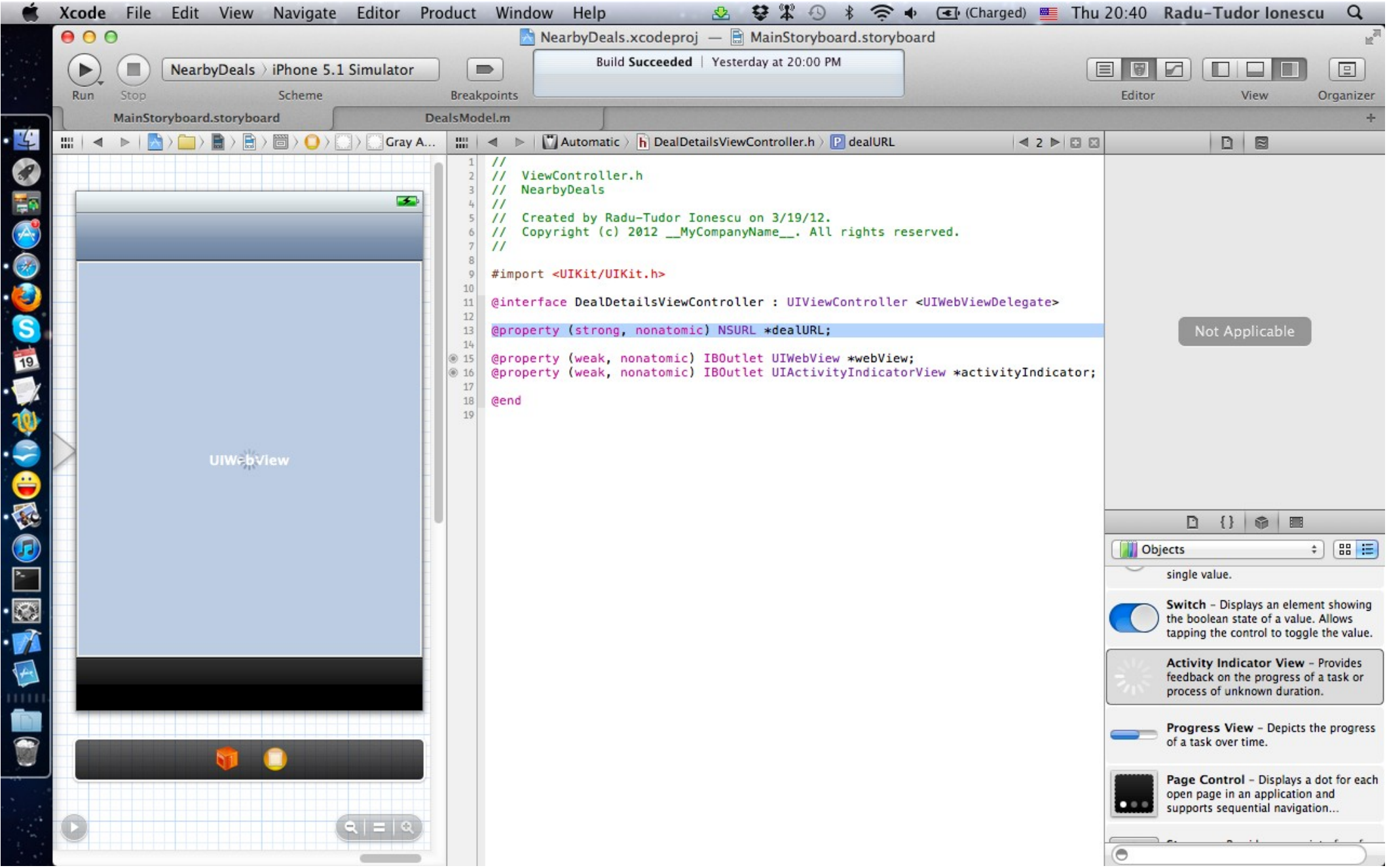
19. The Table View Controller will pass the URL address corresponding to the deal selected by the user to the Deal Details View Controller. This View Controller should have a public `NSURL` `@property` that can be set by the Table View Controller when the segue starts.

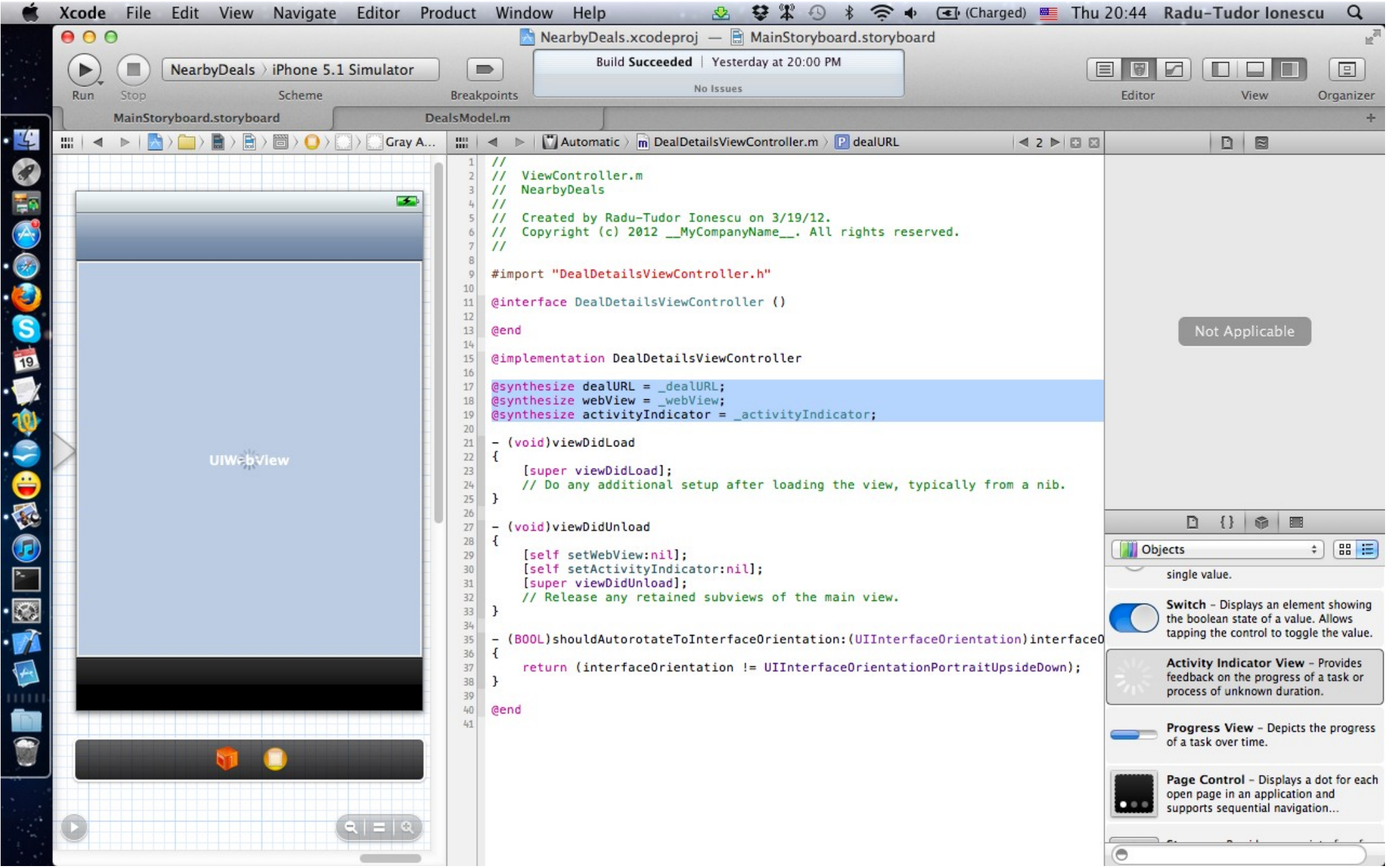
Let's declare this `@property` with the `strong` storage type and name it `dealURL`.

20. Choose DealDetailsViewController.m in Assistant Editor and `@synthesize` this property right after the `@implementation` directive. Also rename it's instance variable to `_dealURL`.

21. It's a good time to also rename the instance variables of the `webView` and `activityIndicator` `@propertys` by prefixing them with underscore.

Look over the next slides for help.


Xcode File Edit View Navigate Editor Product Window Help
(Charged) Thu 20:40 Radu-Tudor Ionescu
NearbyDeals.xcodeproj — MainStoryboard.storyboard
Run Stop NearbyDeals iPhone 5.1 Simulator Scheme Breakpoints
Build Succeeded | Yesterday at 20:00 PM
Editor View Organizer
MainStoryboard.storyboard
DealsModel.m
Gray A...
Automatic DealDetailsViewController.h dealURL
UIWebView
//
// ViewController.h
// NearbyDeals
//
// Created by Radu-Tudor Ionescu on 3/19/12.
// Copyright (c) 2012 __MyCompanyName__. All rights reserved.
//
#import <UIKit/UIKit.h>
@interface DealDetailsViewController : UIViewController <UIWebViewDelegate>
@property (strong, nonatomic) NSURL *dealURL;
@property (weak, nonatomic) IBOutlet UIWebView *webView;
@property (weak, nonatomic) IBOutlet UIActivityIndicatorView *activityIndicator;
@end
Not Applicable
Objects
single value.
Switch – Displays an element showing the boolean state of a value. Allows tapping the control to toggle the value.
Activity Indicator View – Provides feedback on the progress of a task or process of unknown duration.
Progress View – Depicts the progress of a task over time.
Page Control – Displays a dot for each open page in an application and supports sequential navigation...


Xcode File Edit View Navigate Editor Product Window Help
(Charged) Thu 20:44 Radu-Tudor Ionescu
NearbyDeals.xcodeproj — MainStoryboard.storyboard
Run Stop NearbyDeals iPhone 5.1 Simulator Scheme Breakpoints
Build Succeeded | Yesterday at 20:00 PM
No Issues
Editor View Organizer
MainStoryboard.storyboard
DealsModel.m
Gray A...
Automatic DealDetailsViewController.m dealURL
UIWebView
//
// ViewController.m
// NearbyDeals
//
// Created by Radu-Tudor Ionescu on 3/19/12.
// Copyright (c) 2012 __MyCompanyName__. All rights reserved.
//
#import "DealDetailsViewController.h"
@interface DealDetailsViewController ()
@end
@implementation DealDetailsViewController
@synthesize dealURL = _dealURL;
@synthesize webView = _webView;
@synthesize activityIndicator = _activityIndicator;
- (void)viewDidLoad
{
[super viewDidLoad];
// Do any additional setup after loading the view, typically from a nib.
}
- (void)viewDidUnload
{
[self setWebView:nil];
[self setActivityIndicator:nil];
[super viewDidUnload];
// Release any retained subviews of the main view.
}
- (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceO
{
return (interfaceOrientation != UIInterfaceOrientationPortraitUpsideDown);
}
@end
Not Applicable
Objects
single value.
Switch - Displays an element showing the boolean state of a value. Allows tapping the control to toggle the value.
Activity Indicator View - Provides feedback on the progress of a task or process of unknown duration.
Progress View - Depicts the progress of a task over time.
Page Control - Displays a dot for each open page in an application and supports sequential navigation...

Task 3

Task: Configure the View Controller that presents deal details.

22. Close the Utilities area.

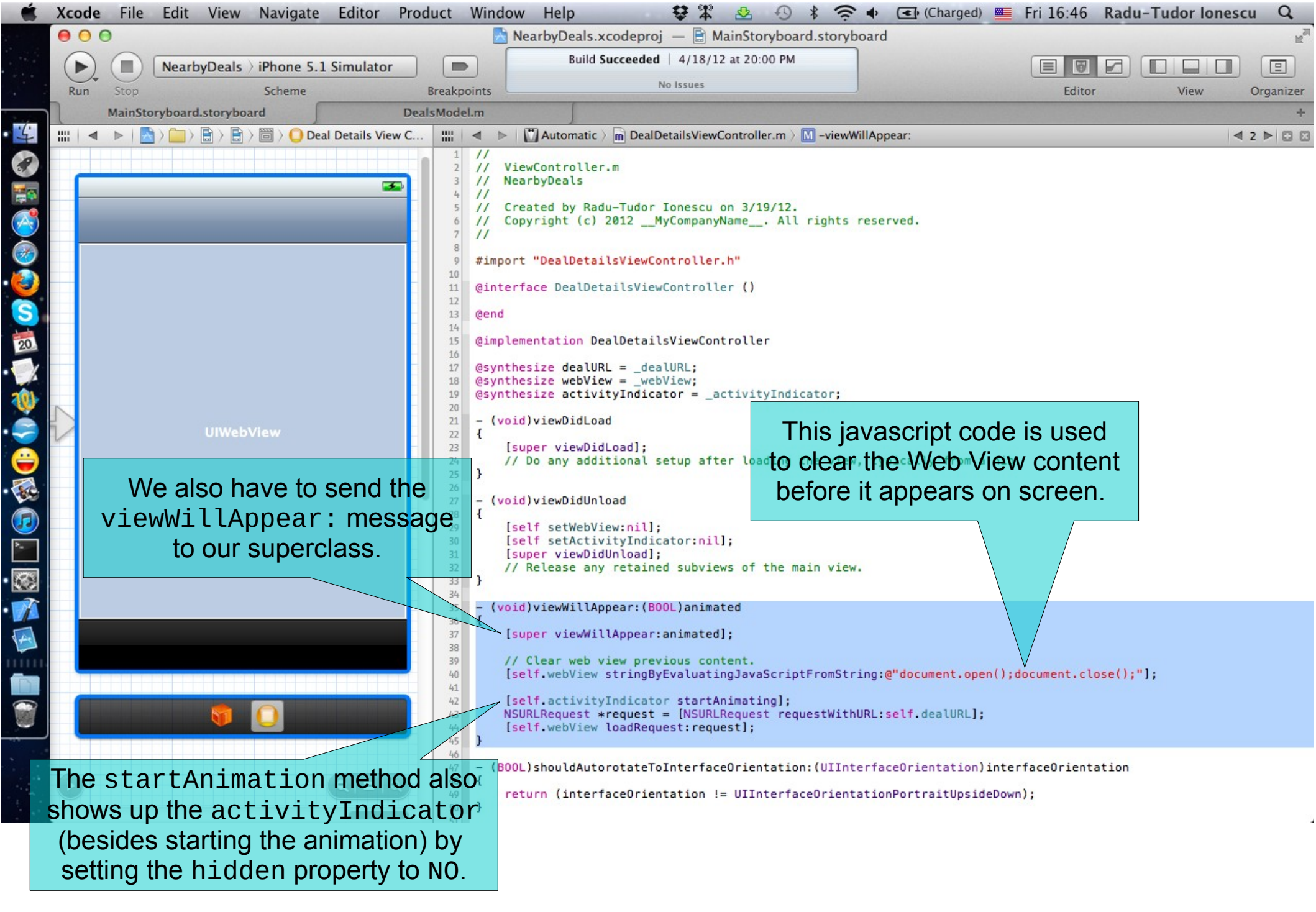
23. Let's start loading the webpage in the `viewWillAppear:` method that gets executed right before the View appears on screen.

The first thing to do in the `viewWillAppear:` method is to clear the previous webpage from the Web View. The Web View may have a webpage loaded if we previously shown details about another deal. We use some javascript code to clear it.

Right before loading the `dealURL` in our `webView`, we will tell the `activityIndicator` to `startAnimating` the spinning wheel.

Then we load an `NSURLRequest` (that was initialized with the `dealURL`) in the `webView`.

Look over the next slide for this method implementation.


Xcode File Edit View Navigate Editor Product Window Help
Fri 16:46 Radu-Tudor Ionescu
NearbyDeals.xcodeproj — MainStoryboard.storyboard
Build Succeeded | 4/18/12 at 20:00 PM
No Issues
Run Stop Scheme Breakpoints Editor View Organizer
MainStoryboard.storyboard
DealsModel.m
Deal Details View C...
Automatic DealDetailsViewController.m -viewWillAppear:
UIWebView
//
// ViewController.m
// NearbyDeals
//
// Created by Radu-Tudor Ionescu on 3/19/12.
// Copyright (c) 2012 __MyCompanyName__. All rights reserved.
//
#import "DealDetailsViewController.h"
@interface DealDetailsViewController ()
@end
@implementation DealDetailsViewController
@synthesize dealURL = _dealURL;
@synthesize webView = _webView;
@synthesize activityIndicator = _activityIndicator;
- (void)viewDidLoad
{
[super viewDidLoad];
// Do any additional setup after loading the view.
}
- (void)viewDidUnload
{
[self setWebView:nil];
[self setActivityIndicator:nil];
[super viewDidUnload];
// Release any retained subviews of the main view.
}
- (void)viewWillAppear:(BOOL)animated
{
[super viewWillAppear:animated];
// Clear web view previous content.
[self.webView stringByEvaluatingJavaScriptFromString:@"document.open();document.close();"];
[self.activityIndicator startAnimating];
NSURLRequest *request = [NSURLRequest requestWithURL:self.dealURL];
[self.webView loadRequest:request];
}
- (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
return (interfaceOrientation != UIInterfaceOrientationPortraitUpsideDown);
We also have to send the viewWillAppear: message to our superclass.
This javascript code is used to clear the Web View content before it appears on screen.
The startAnimation method also shows up the activityIndicator (besides starting the animation) by setting the hidden property to NO.

Task 3

Task: Configure the View Controller that presents deal details.

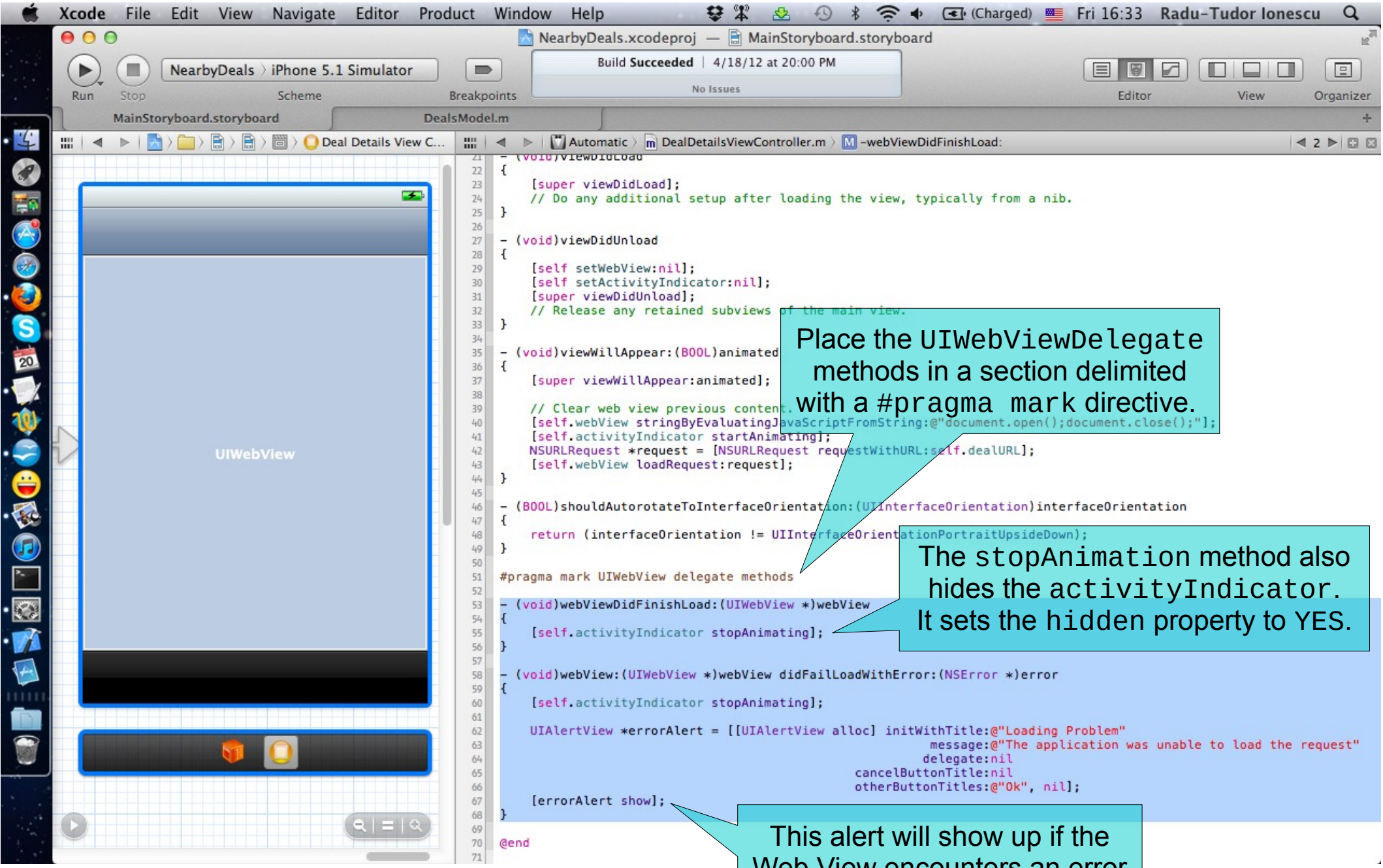
24. Next, we should implement the `UIWebViewDelegate`'s methods. When the Web View finishes loading the request with success, the `activityIndicator` must stop the animation. Implement the `webViewDidFinishLoad:` method to work as described.

25. If the Web View fails to load the request, we should display an Alert View with an error message to the user in order to let him know about the problem. Again, the `activityIndicator` must stop the animation.

Implement the `webView:didFailLoadWithError:` method to work as described above.

Note that we should stop loading the Web View's request and the `activityIndicator` animation if the View goes off the screen. This is left as an assignment.

Look over the next slide for the `UIWebViewDelegate`'s methods implementation.


Xcode File Edit View Navigate Editor Product Window Help
Fri 16:33 Radu-Tudor Ionescu
NearbyDeals.xcodeproj — MainStoryboard.storyboard
Run Stop Scheme Breakpoints
NearbyDeals iPhone 5.1 Simulator
Build Succeeded 4/18/12 at 20:00 PM
No Issues
Editor View Organizer
MainStoryboard.storyboard
DealsModel.m
Deal Details View C...
Automatic DealDetailsViewController.m -webViewDidFinishLoad:
UIWebView
- (void)viewDidLoad
{
[super viewDidLoad];
// Do any additional setup after loading the view, typically from a nib.
}
- (void)viewDidUnload
{
[self setWebView:nil];
[self setActivityIndicator:nil];
[super viewDidUnload];
// Release any retained subviews of the main view.
}
- (void)viewWillAppear:(BOOL)animated
{
[super viewWillAppear:animated];
// Clear web view previous content.
[self.webView stringByEvaluatingJavaScriptFromString:@"document.open();document.close();"];
[self.activityIndicator startAnimating];
NSURLRequest *request = [NSURLRequest requestWithURL:self.dealURL];
[self.webView loadRequest:request];
}
- (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
{
return (interfaceOrientation != UIInterfaceOrientationPortraitUpsideDown);
}
#pragma mark UIWebView delegate methods
- (void)webViewDidFinishLoad:(UIWebView *)webView
{
[self.activityIndicator stopAnimating];
}
- (void)webView:(UIWebView *)webView didFailLoadWithError:(NSError *)error
{
[self.activityIndicator stopAnimating];
UIAlertView *errorAlert = [[UIAlertView alloc] initWithTitle:@"Loading Problem"
message:@"The application was unable to load the request"
delegate:nil
cancelButtonTitle:nil
otherButtonTitles:@"Ok", nil];
[errorAlert show];
}
@end
Place the UIWebViewDelegate methods in a section delimited with a #pragma mark directive.
The stopAnimation method also hides the activityIndicator. It sets the hidden property to YES.
This alert will show up if the Web View encounters an error when loading the request.

Task 3

Task: Configure the View Controller that presents deal details.

26. The Deal Details View Controller is configured, but we still have to pass the deal URL address when the Table View Controller performs the segue.

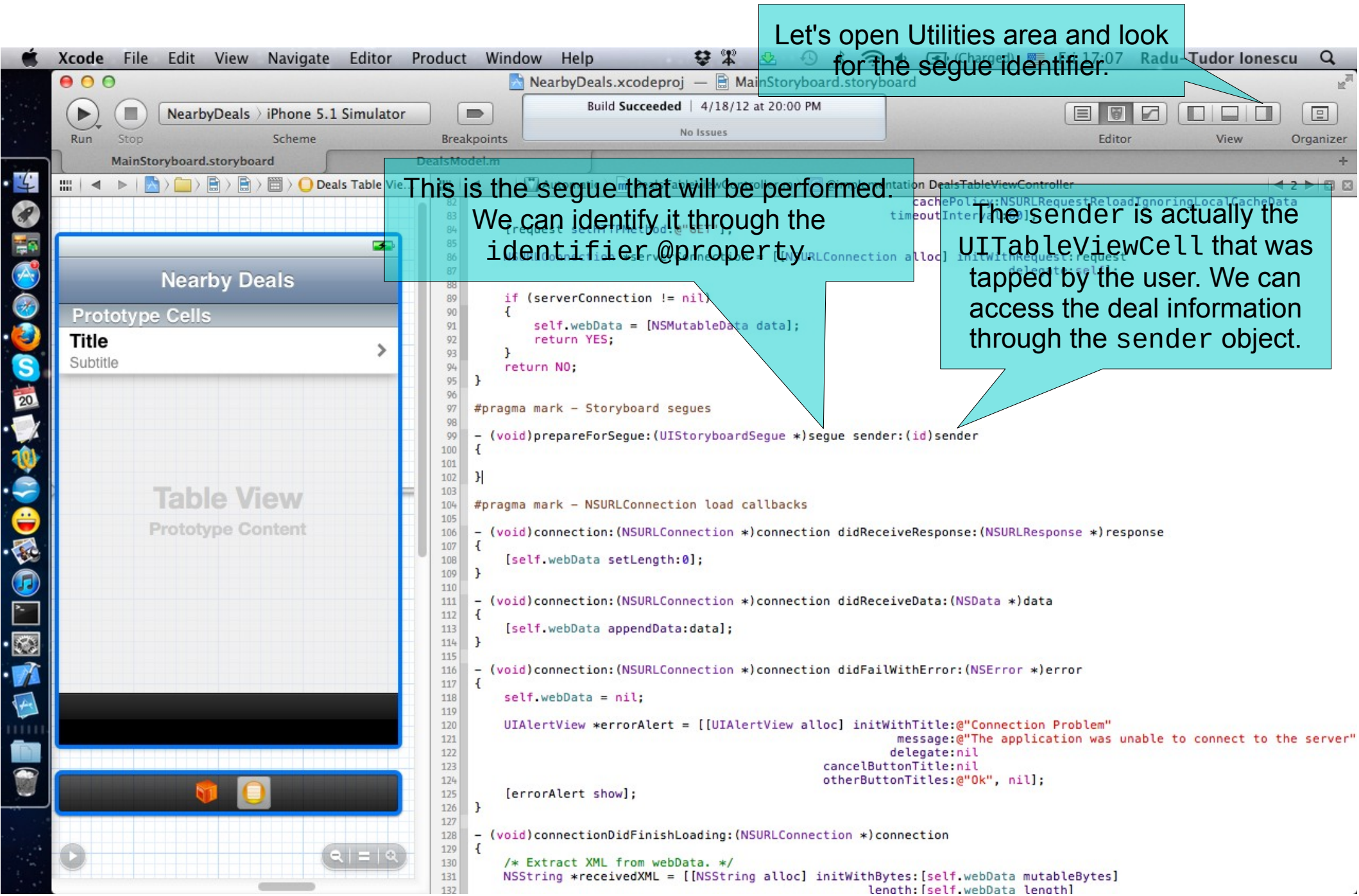
Let's scroll to and click on the Table View Controller in Interface Builder.

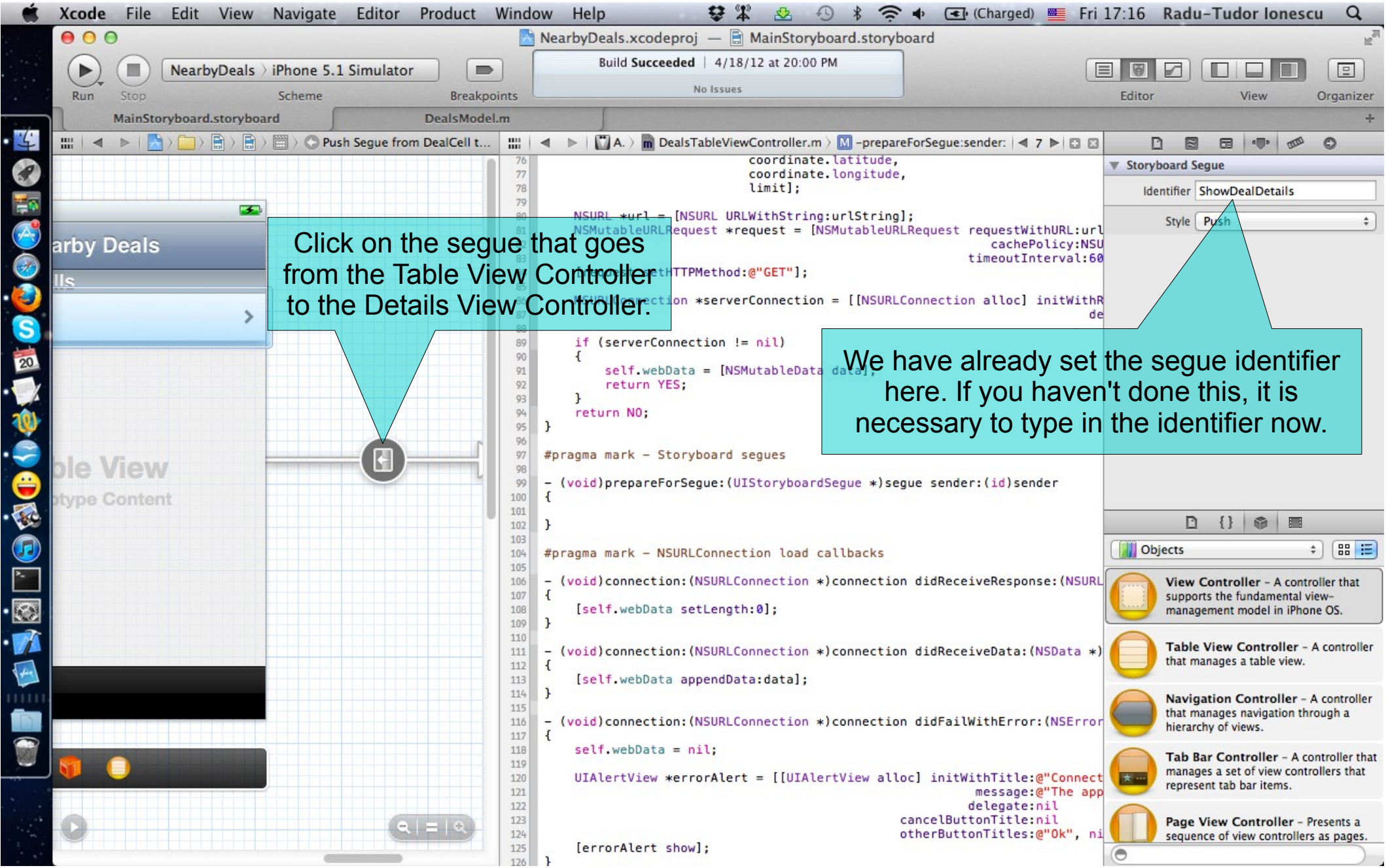
27. Select the DealsTableViewController implementation file in Assistant Editor.

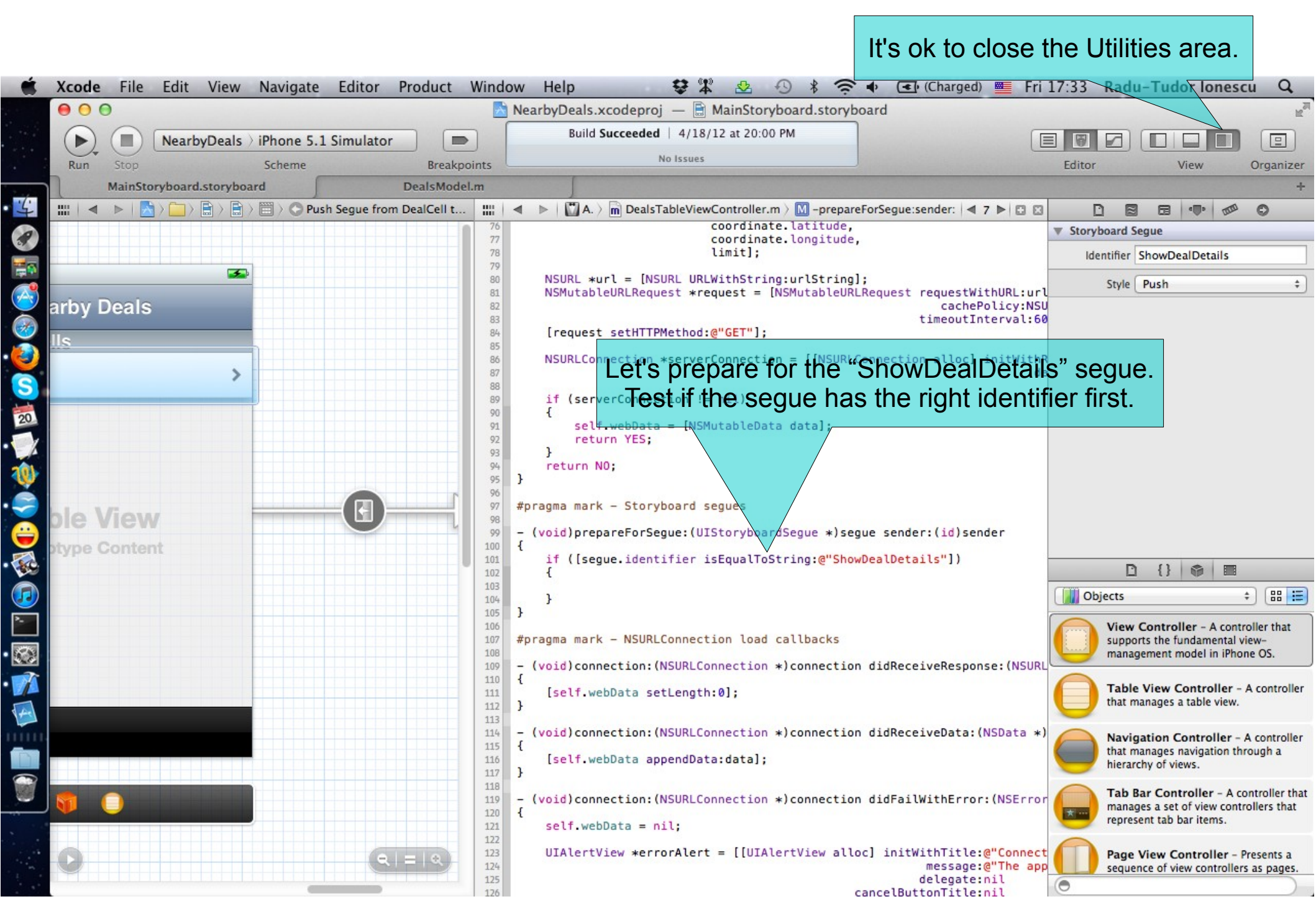
28. Let's add a new section of code and name it “Storyboard segues” using the `#pragma mark` compiler directive right before the “NSURLConnection load callbacks” section.

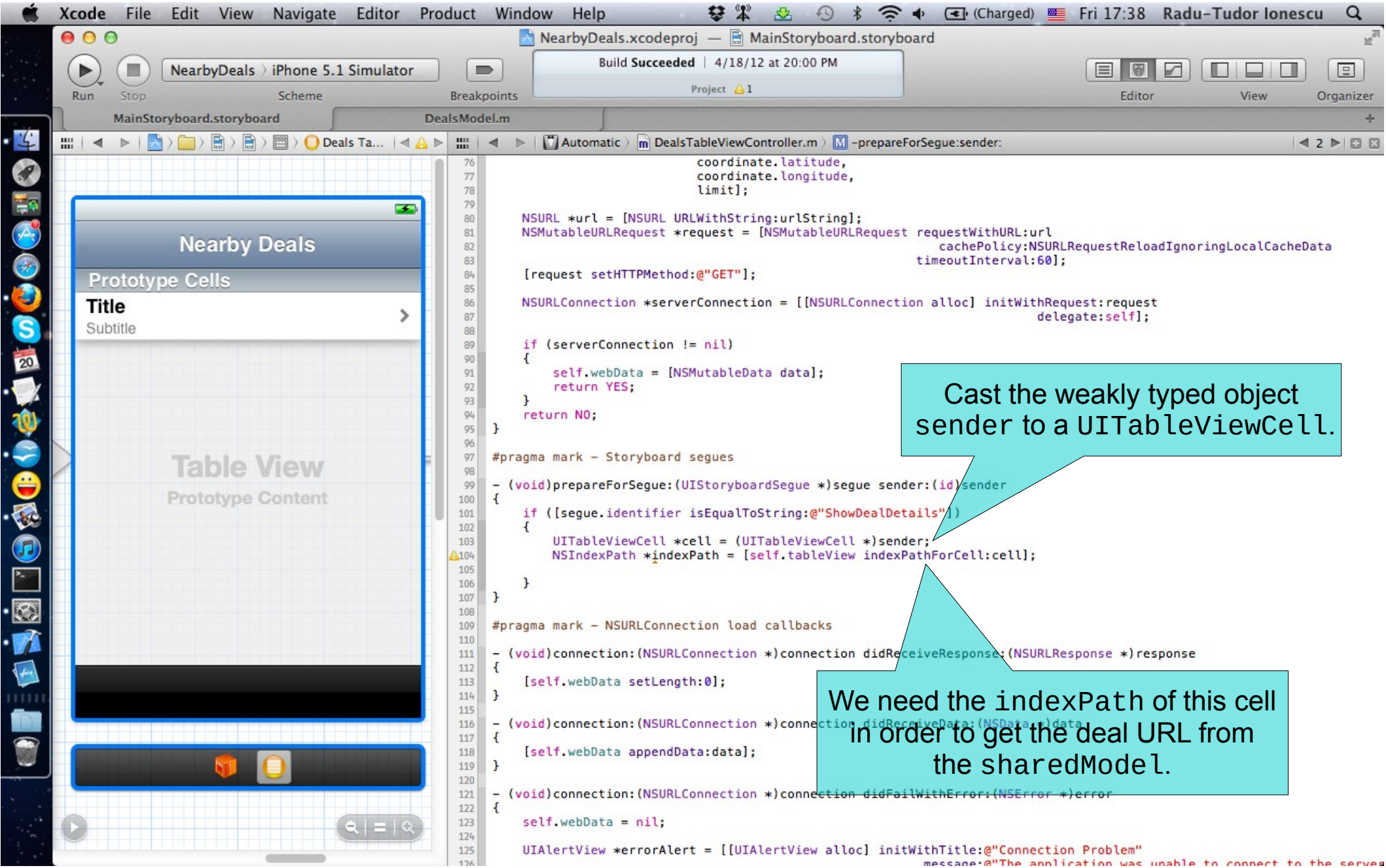
29. We have a chance to pass the deal URL to the Deal Details View Controller in the `prepareForSegue:sender:` method. Let's add an implementation to this method in the “Storyboard segues” section.

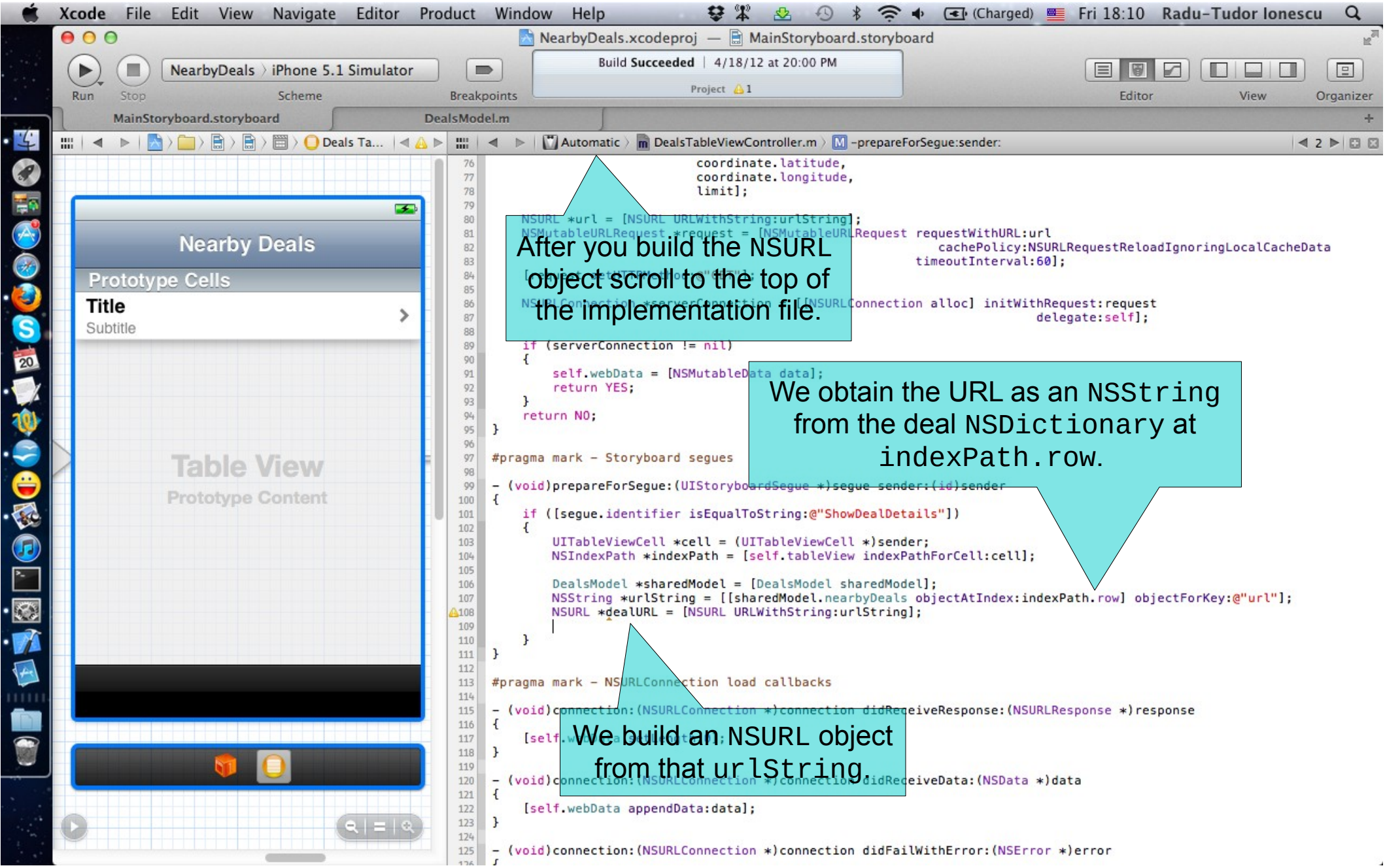
Look over the next slides to finish this method's implementation.


Let's open Utilities area and look for the segue identifier.
This is the segue that will be performed. We can identify it through the identifier @property.
The sender is actually the UITableViewCell that was tapped by the user. We can access the deal information through the sender object.
Nearby Deals
Prototype Cells
Title
Subtitle
Table View
Prototype Content
- (void)prepareForSegue:(UIStoryboardSegue *)segue sender:(id)sender
{
}
#pragma mark - Storyboard segues
#pragma mark - NSURLConnection load callbacks
- (void)connection:(NSURLConnection *)connection didReceiveResponse:(NSURLResponse *)response
- (void)connection:(NSURLConnection *)connection didReceiveData:(NSData *)data
- (void)connection:(NSURLConnection *)connection didFailWithError:(NSError *)error
- (void)connectionDidFinishLoading:(NSURLConnection *)connection


Click on the segue that goes from the Table View Controller to the Details View Controller.
We have already set the segue identifier here. If you haven't done this, it is necessary to type in the identifier now.
Storyboard Segue
Identifier ShowDealDetails
Style Push
Objects
View Controller – A controller that supports the fundamental view-management model in iPhone OS.
Table View Controller – A controller that manages a table view.
Navigation Controller – A controller that manages navigation through a hierarchy of views.
Tab Bar Controller – A controller that manages a set of view controllers that represent tab bar items.
Page View Controller – Presents a sequence of view controllers as pages.


It's ok to close the Utilities area.
Let's prepare for the "ShowDealDetails" segue.
Test if the segue has the right identifier first.
Xcode File Edit View Navigate Editor Product Window Help
NearbyDeals.xcodeproj — MainStoryboard.storyboard
Build Succeeded | 4/18/12 at 20:00 PM
No Issues
Run Stop Scheme Breakpoints Editor View Organizer
MainStoryboard.storyboard
DealsModel.m
Push Segue from DealCell t...
DealsTableViewController.m
-prepareForSegue:sender:
Storyboard Segue
Identifier ShowDealDetails
Style Push
Objects
View Controller – A controller that supports the fundamental view-management model in iPhone OS.
Table View Controller – A controller that manages a table view.
Navigation Controller – A controller that manages navigation through a hierarchy of views.
Tab Bar Controller – A controller that manages a set of view controllers that represent tab bar items.
Page View Controller – Presents a sequence of view controllers as pages.


Nearby Deals
Prototype Cells
Title
Subtitle
Table View
Prototype Content
Cast the weakly typed object sender to a UITableViewCell.
We need the indexPath of this cell in order to get the deal URL from the sharedModel.


After you build the NSURL object scroll to the top of the implementation file.
We obtain the URL as an NSString from the deal NSDictionary at indexPath.row.
We build an NSURL object from that urlString.

Next, we are going to set the `dealURL` `@property` of our destination View Controller. The Table View Controller has to know about the Deal Details View Controller, so let's `#import` the "DealDetailsViewController.h" header here.


```
//
//  DealsTableViewController.m
//  NearbyDeals
//
//  Created by Radu-Tudor Ionescu on 3/21/12.
//  Copyright (c) 2012 __MyCompanyName__. Al
//

#import "DealsTableViewController.h"
#import "DealDetailsViewController.h"
#import "DealsModel.h"

static NSString *kAdsServerURL = @"http://www
static NSString *kAppKey = @"fe008041973b6676

@interface DealsTableViewController ()

@property (nonatomic, strong) NSMutableData *

@end

@implementation DealsTableViewController

@synthesize webData = _webData;

- (id)initWithStyle:(UITableViewStyle)style
{
    self = [super initWithStyle:style];
    if (self) {
        // Custom initialization
    }
    return self;
}

- (void)viewDidLoad
{
    [super viewDidLoad];

    // Uncomment the following line to preserve selection between presentations.
    // self.clearsSelectionOnViewWillAppear = NO;

    // Uncomment the following line to display an Edit button in the navigation bar for this view controller.
    // self.navigationItem.rightBarButtonItem = self.editButtonItem;
}

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

@interface DealsTableViewController()
webData
@implementation DealsTableViewController
webData
-initWithStyle:
-viewDidLoad
-viewDidUnload
-viewDidAppear:
-shouldAutorotateToInterfaceOrientation:
-requestDealsNearLocation:limit:

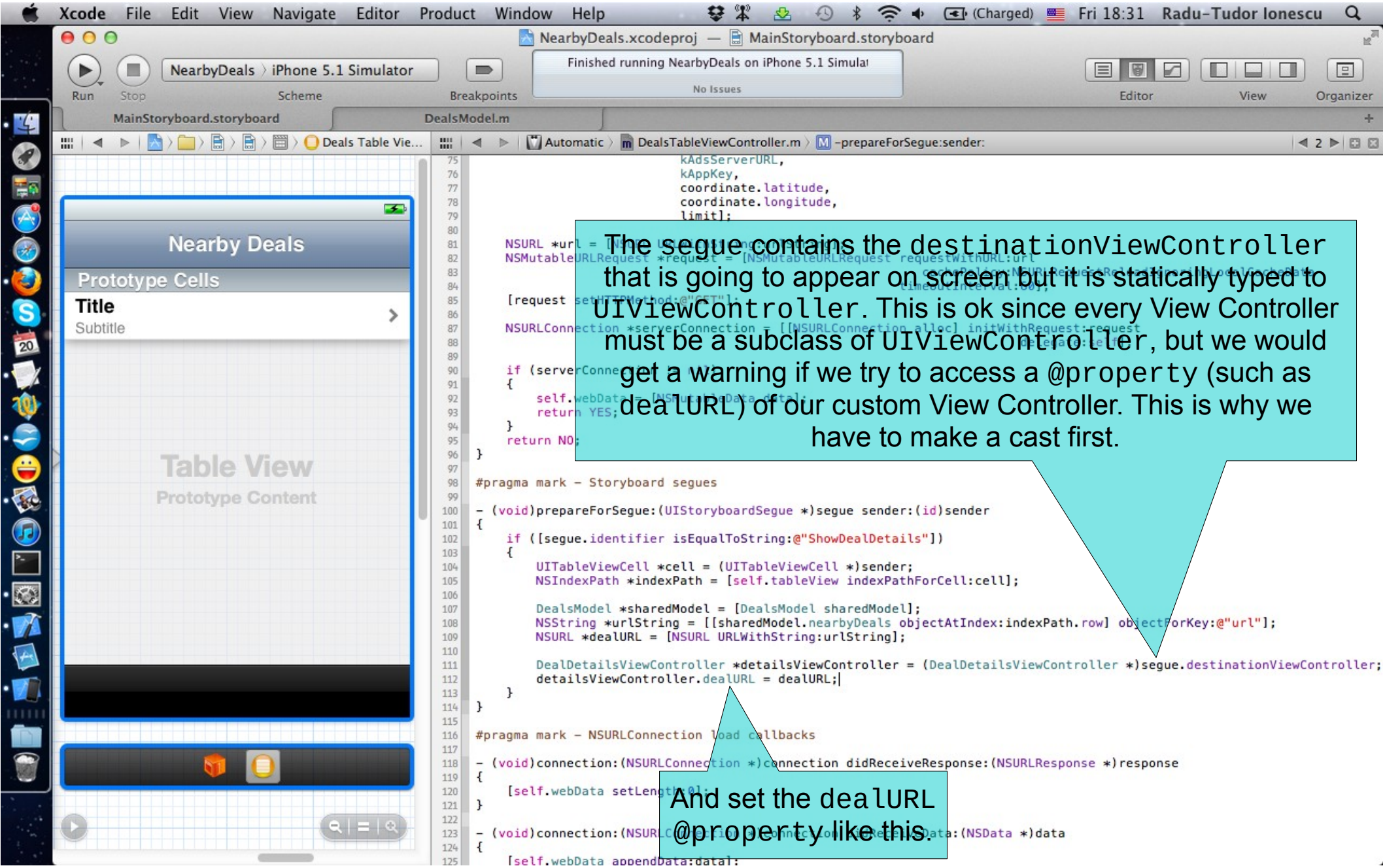
Storyboard segues
-prepareForSegue:sender:

NSURLConnection load callbacks
-connection:didReceiveResponse:
-connection:didReceiveData:
-connection:didFailWithError:
-connectionDidFinishLoading:

Table view data source
-numberOfSectionsInTableView:

Table view delegate
-tableView:didSelectRowAtIndexPath:

Then go back to the `prepareForSegue:sender:` method to finish it's implementation.


Nearby Deals
Prototype Cells
Title
Subtitle
Table View
Prototype Content
The segue contains the destinationViewController that is going to appear on screen but it is statically typed to UIViewController. This is ok since every View Controller must be a subclass of UIViewController, but we would get a warning if we try to access a @property (such as dealURL) of our custom View Controller. This is why we have to make a cast first.
And set the dealURL @property like this.

Task 3

Task: Configure the View Controller that presents deal details.

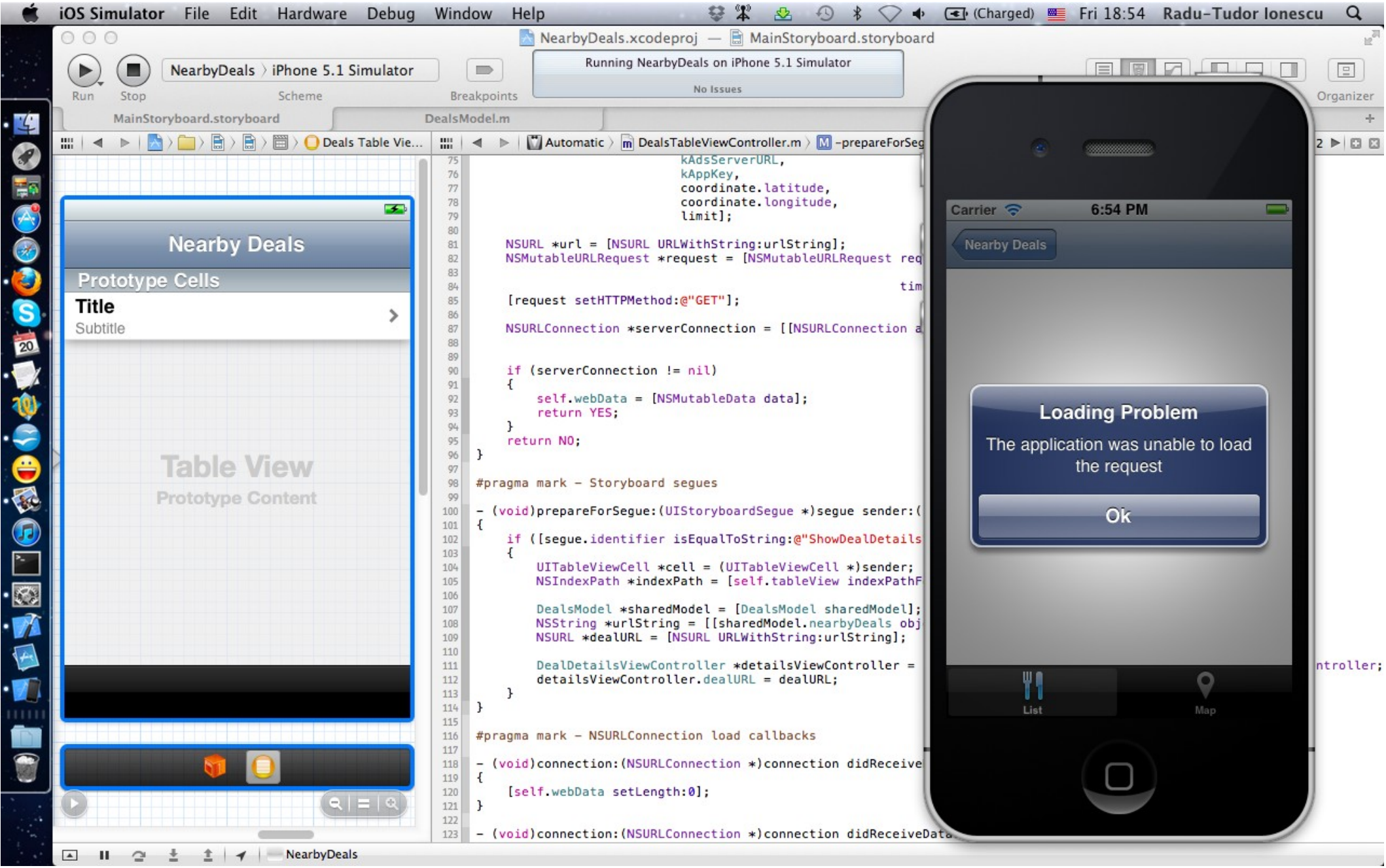
30. Run the application in iOS Simulator. Try to view the details for a few deals after the application loads them inside the Table View.

31. Unplug your Internet cable to test what happens when the device has lost its Internet connection. Now try to view details about a nearby deal.

An Alert View like the one from the next slide should appear on screen. If you press the “Ok” button the Alert View is dismissed, but the Deal Details View remains on screen.

There is no reason to let the Deal Details View on screen. The user's only action is to go back from this View and maybe try to see details about another deal. We can anticipate the user's action and automatically put him back on the Table View when he presses the “Ok” button.

32. Stop running the application and plug your Internet cable back.


iOS Simulator File Edit Hardware Debug Window Help
(Charged) Fri 18:54 Radu-Tudor Ionescu
NearbyDeals.xcodeproj — MainStoryboard.storyboard
Run Stop NearbyDeals 〉 iPhone 5.1 Simulator Scheme Breakpoints
Running NearbyDeals on iPhone 5.1 Simulator
No Issues
Organizer
MainStoryboard.storyboard
DealsModel.m
Deals Table Vie...
Automatic 〉 DealsTableViewController.m 〉 -prepareForSeg
Nearby Deals
Prototype Cells
Title
Subtitle
Table View
Prototype Content
kAdsServerURL,
kAppKey,
coordinate.latitude,
coordinate.longitude,
limit];
NSURL *url = [NSURL URLWithString:urlString];
NSMutableURLRequest *request = [NSMutableURLRequest req
[request setHTTPMethod:@"GET"];
NSURLConnection *serverConnection = [[NSURLConnection a
if (serverConnection != nil)
{
self.webData = [NSMutableData data];
return YES;
}
return NO;
}
#pragma mark - Storyboard segues
- (void)prepareForSegue:(UIStoryboardSegue *)segue sender:(
{
if ([segue.identifier isEqualToString:@"ShowDealDetails
{
UITableViewCell *cell = (UITableViewCell *)sender;
NSIndexPath *indexPath = [self.tableView indexPathF
DealsModel *sharedModel = [DealsModel sharedModel];
NSString *urlString = [[sharedModel.nearbyDeals obj
NSURL *dealURL = [NSURL URLWithString:urlString];
DealDetailsViewController *detailsViewController =
detailsViewController.dealURL = dealURL;
}
}
#pragma mark - NSURLConnection load callbacks
- (void)connection:(NSURLConnection *)connection didReceive
{
[self.webData setLength:0];
}
- (void)connection:(NSURLConnection *)connection didReceiveData
Carrier 6:54 PM
Nearby Deals
Loading Problem
The application was unable to load the request
Ok
List
Map
NearbyDeals

Task 3

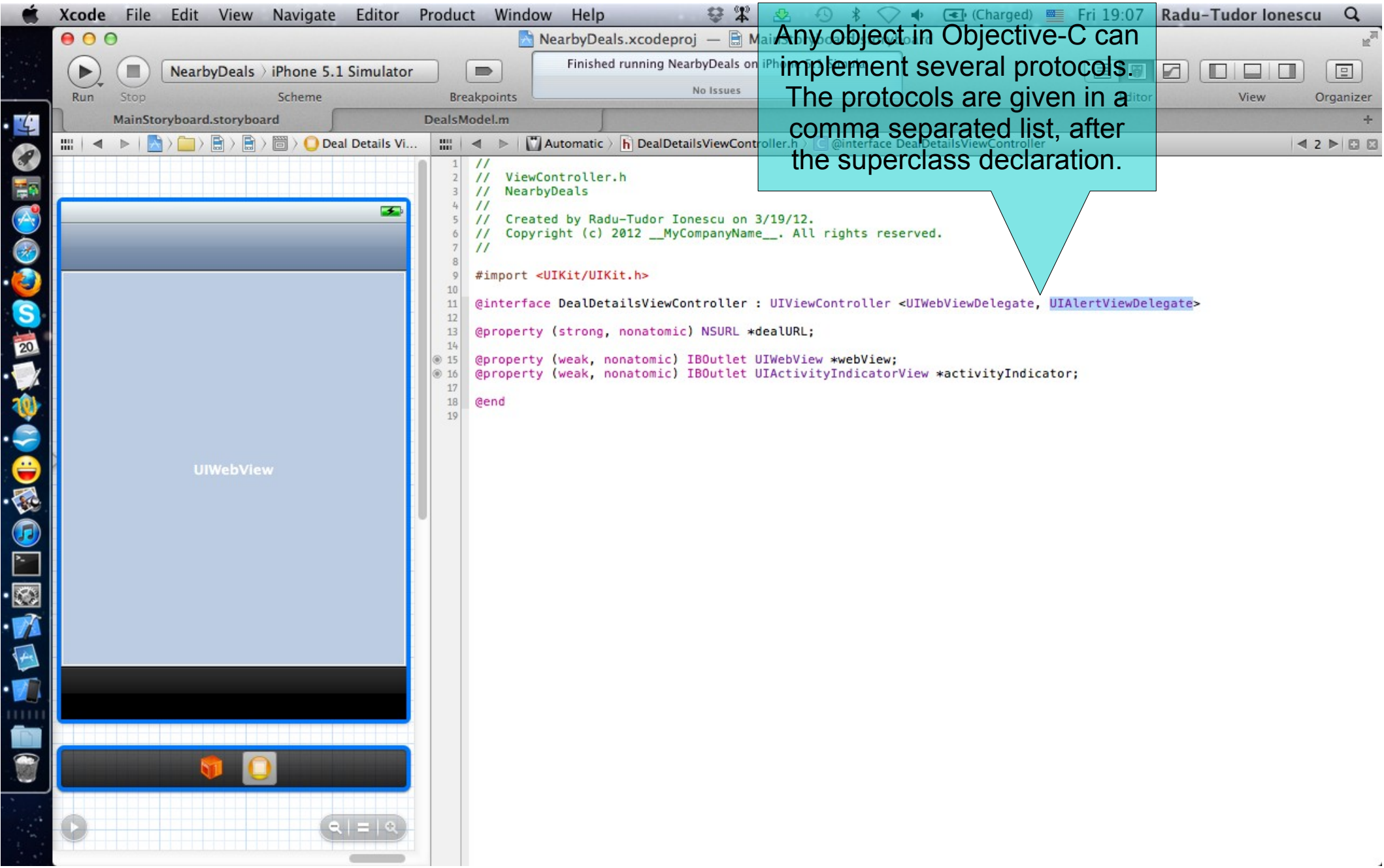
Task: Configure the View Controller that presents deal details.

33. Click on the Deal Details View Controller in Interface Builder to see its associated files in Assistant Editor. Make sure the header file is selected in Assistant Editor.

34. We can assign a `delegate` object to the `UIAlertView` that shows the “Loading Problem” error message. The `delegate` object will receive an event when the user presses the “Ok” button of the Alert View. We want to pop our View Controller from the navigation stack in this case.

We must declare the `UIAlertViewDelegate` protocol and set this View Controller to be the Alert View delegate.

The next screenshot shows how to declare this `@protocol`.


Any object in Objective-C can implement several protocols. The protocols are given in a comma separated list, after the superclass declaration.
Xcode File Edit View Navigate Editor Product Window Help
Radu-Tudor Ionescu
NearbyDeals.xcodeproj
NearbyDeals iPhone 5.1 Simulator
Run Stop Scheme Breakpoints
Finished running NearbyDeals on
No Issues
View Organizer
MainStoryboard.storyboard
DealsModel.m
Deal Details Vi...
Automatic DealDetailsViewController.h @interface DealDetailsViewController
UIWebView
//
// ViewController.h
// NearbyDeals
//
// Created by Radu-Tudor Ionescu on 3/19/12.
// Copyright (c) 2012 __MyCompanyName__. All rights reserved.
//
#import <UIKit/UIKit.h>
@interface DealDetailsViewController : UIViewController <UIWebViewDelegate, UIAlertViewDelegate>
@property (strong, nonatomic) NSURL *dealURL;
@property (weak, nonatomic) IBOutlet UIWebView *webView;
@property (weak, nonatomic) IBOutlet UIActivityIndicatorView *activityIndicator;
@end

Task 3

Task: Configure the View Controller that presents deal details.

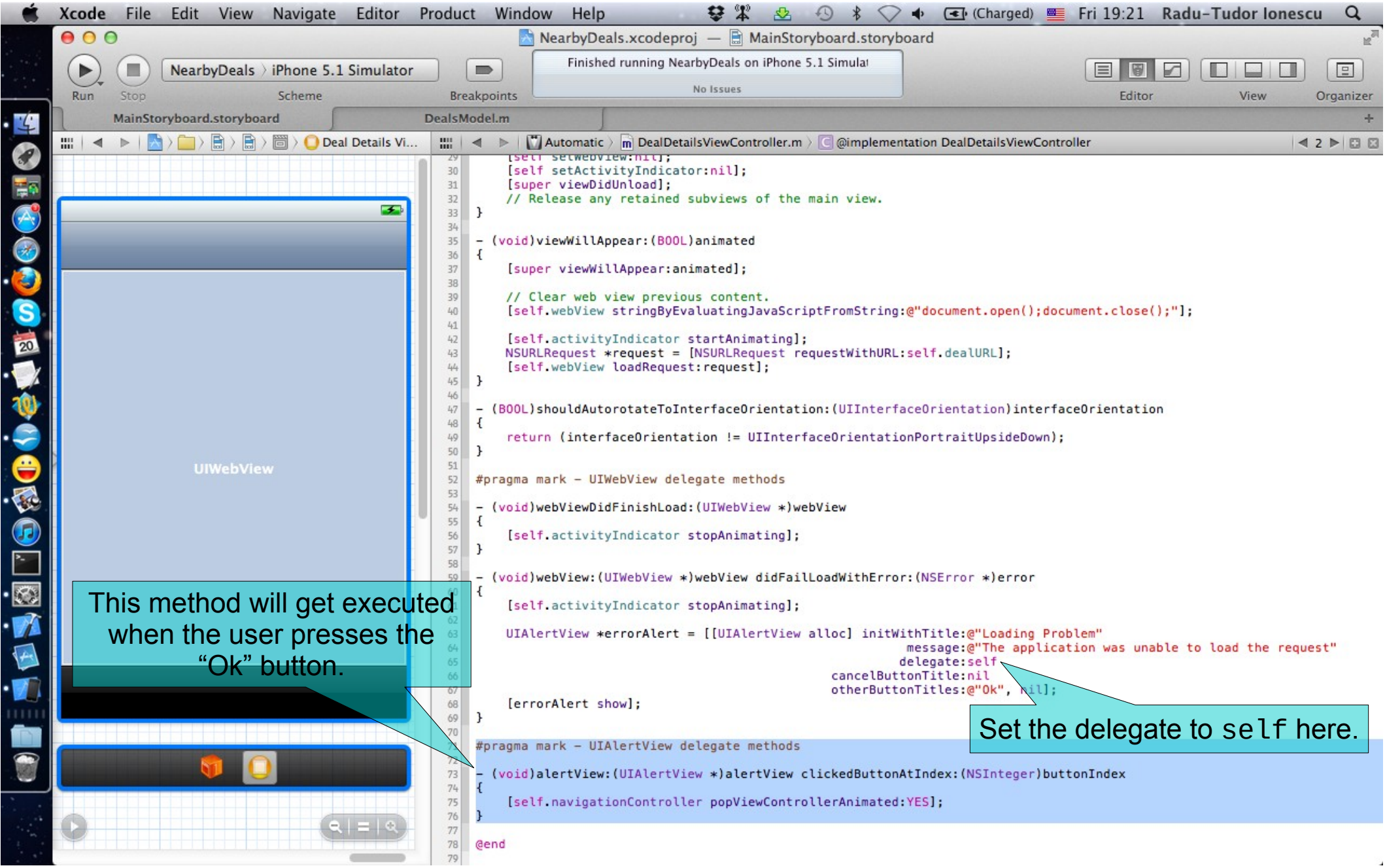
35. Select the DealDetailsViewController.m implementation file in Assistant Editor.

36. Set the `delegate` object of the `UIAlertView` to `self` when you initialize it in the `webView:didFailLoadWithError:` method.

37. Add a `#pragma mark` declaration for the methods of the `UIAlertViewDelegate` protocol.

38. Implement the `alertView:clickedButtonAtIndex:` method and pop the View Controller from the navigation stack. You have to send the `popViewControllerAnimated:` message to the `self.navigationController` object.

Look over the next screenshot for help.


Xcode File Edit View Navigate Editor Product Window Help
Fri 19:21 Radu-Tudor Ionescu
NearbyDeals.xcodeproj — MainStoryboard.storyboard
Run Stop NearbyDeals iPhone 5.1 Simulator Scheme Breakpoints
Finished running NearbyDeals on iPhone 5.1 Simula
No Issues
Editor View Organizer
MainStoryboard.storyboard
DealsModel.m
Deal Details Vi...
Automatic DealDetailsViewController.m @implementation DealDetailsViewController
UIWebView
This method will get executed when the user presses the “Ok” button.
Set the delegate to self here.
[self setWebView:nil];
[self setActivityIndicator:nil];
[super viewDidUnload];
// Release any retained subviews of the main view.
}
- (void)viewWillAppear:(BOOL)animated
{
[super viewWillAppear:animated];
// Clear web view previous content.
[self.webView stringByEvaluatingJavaScriptFromString:@"document.open();document.close();"];
[self.activityIndicator startAnimating];
NSURLRequest *request = [NSURLRequest requestWithURL:self.dealURL];
[self.webView loadRequest:request];
}
- (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
{
return (interfaceOrientation != UIInterfaceOrientationPortraitUpsideDown);
}
#pragma mark - UIWebView delegate methods
- (void)webViewDidFinishLoad:(UIWebView *)webView
{
[self.activityIndicator stopAnimating];
}
- (void)webView:(UIWebView *)webView didFailLoadWithError:(NSError *)error
{
[self.activityIndicator stopAnimating];
UIAlertView *errorAlert = [[UIAlertView alloc] initWithTitle:@"Loading Problem"
message:@"The application was unable to load the request"
delegate:self
cancelButtonTitle:nil
otherButtonTitles:@"Ok", nil];
[errorAlert show];
}
#pragma mark - UIAlertView delegate methods
- (void)alertView:(UIAlertView *)alertView clickedButtonAtIndex:(NSInteger)buttonIndex
{
[self.navigationController popViewControllerAnimated:YES];
}
@end

Task 3

Task: Configure the View Controller that presents deal details.

39. Run the application in iOS Simulator. Wait until the application loads the nearby deals inside the Table View.

40. Unplug your Internet cable and try to view details about a deal.

The Alert View should appear on screen. If you press the “Ok” button the Alert View is dismissed and the user is put back on the Table View. The problem is fixed!

41. Stop running the application and plug your Internet cable again.

Assignment 1

Assignment: The Table View cells are very tight together. Set the cell height to 70 pixels to add some white space and make them look nice.

Hint: Open the Utilities area and select the Prototype Cell in Interface Builder. Look for the Row Height property in Size Inspector. Note that you have to check the “Custom” option.

Do the same thing for the Table View. It also has a Row Height property that needs to be set.

Assignment 2

Assignment: Set the title of Deal Details View Controller to “Deal Details”.

Hint: Do this in Interface Builder.

Assignment 3

Assignment: It would be nice to hide the bottom Tab Bar when the Deal Details View Controller is pushed on screen.

Hint: Select the Deal Details View in Interface Builder. In Attributes Inspector search for a property that suggests it would hide the Tab Bar.

You can also solve this assignment programmatically by setting the `hidesBottomBarWhenPushed` `@property` of the View Controller when its View loads (i.e. in the `viewDidLoad` method).

Assignment 4

Assignment: Implement the `viewDidDisappear:` method of the Deal Details View Controller so that it stops loading webpage requested by the Web View.

Hint: Note that if you stop loading the request programmatically, you also have to stop the animation of the Activity Indicator.

Congratulations!