

# Developing Applications for iOS



## Lab 6: Nearby Deals (2 of 6)

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

# Nearby Deals

## Description:

We are going to build a new application that will show deals from nearby restaurants and bars. The application will display the deals in two modes: a list view (using a `UITableViewController`) and a map view (using a `MKMapView`). We will request the deals from a server ([www.geoadsplus.com](http://www.geoadsplus.com) to be more precise). We will use XML to communicate with this server. Note that XML and JSON are standard ways of communicating with a server.

We have to pass the device location (latitude, longitude) to the server so that it gives us nearby deals. Thus, we will need to use Location Services to determine the device location.

We will offer details about our deals. We are going to use a navigation controller to navigate between the list View and the details View.

# Task 1

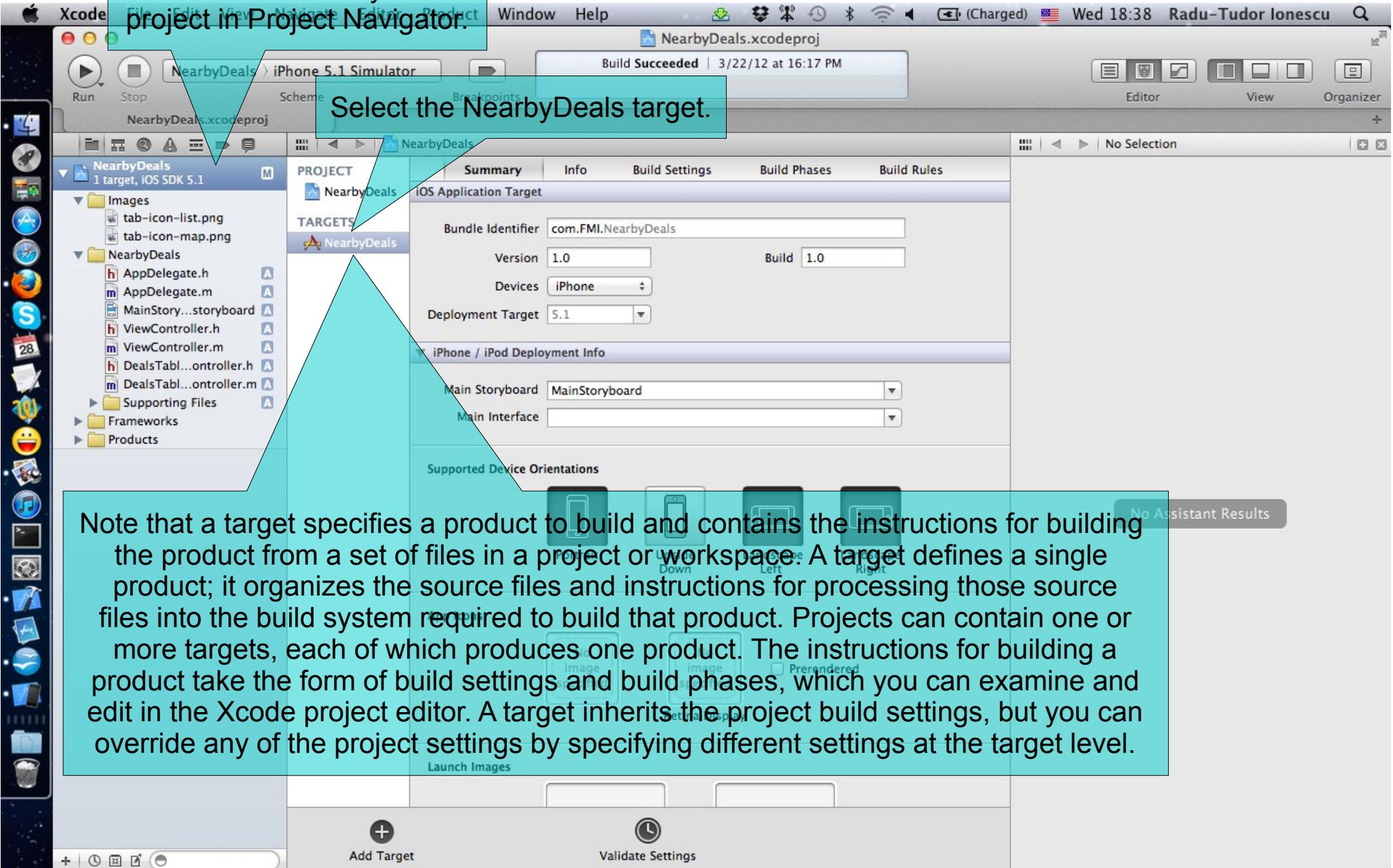
**Task: Add the CoreLocation framework to your project.**

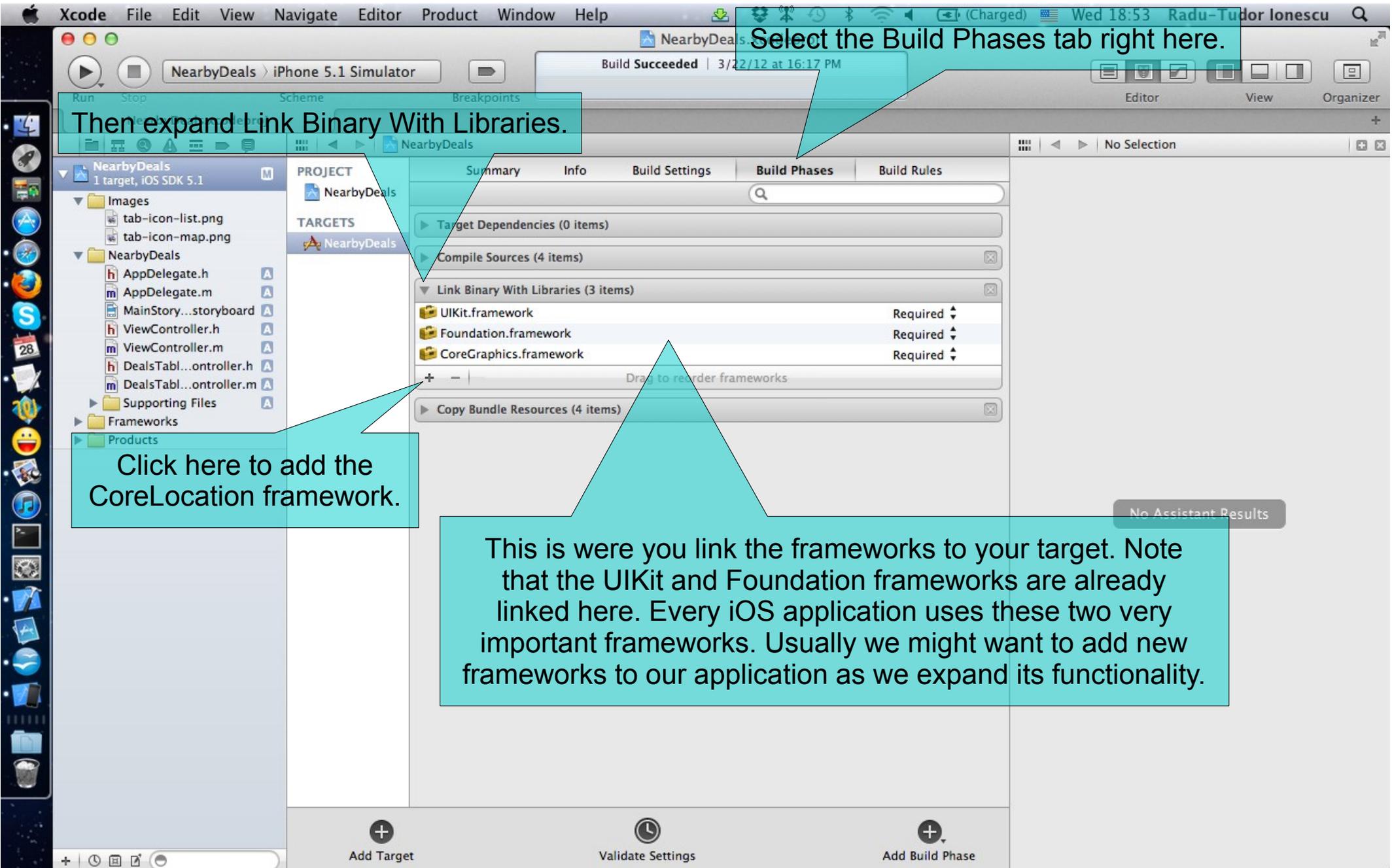
1. Launch Xcode and go to “File > Open” and select the Xcode project (.xcodproj) inside the “NearbyDeals(1of6)” folder. You can also double-click on the .xcodproj file to open it in Xcode.
2. Run the application in iOS Simulator and take a look to remember what was done last time.
3. Stop running the application.
4. Open Project Navigator and follow the steps in the next slides to add the CoreLocation framework to your project.

Click on the NearbyDeals project in Project Navigator.

Select the NearbyDeals target.

Note that a target specifies a product to build and contains the instructions for building the product from a set of files in a project or workspace. A target defines a single product; it organizes the source files and instructions for processing those source files into the build system required to build that product. Projects can contain one or more targets, each of which produces one product. The instructions for building a product take the form of build settings and build phases, which you can examine and edit in the Xcode project editor. A target inherits the project build settings, but you can override any of the project settings by specifying different settings at the target level.





Select the Build Phases tab right here.

Then expand Link Binary With Libraries.

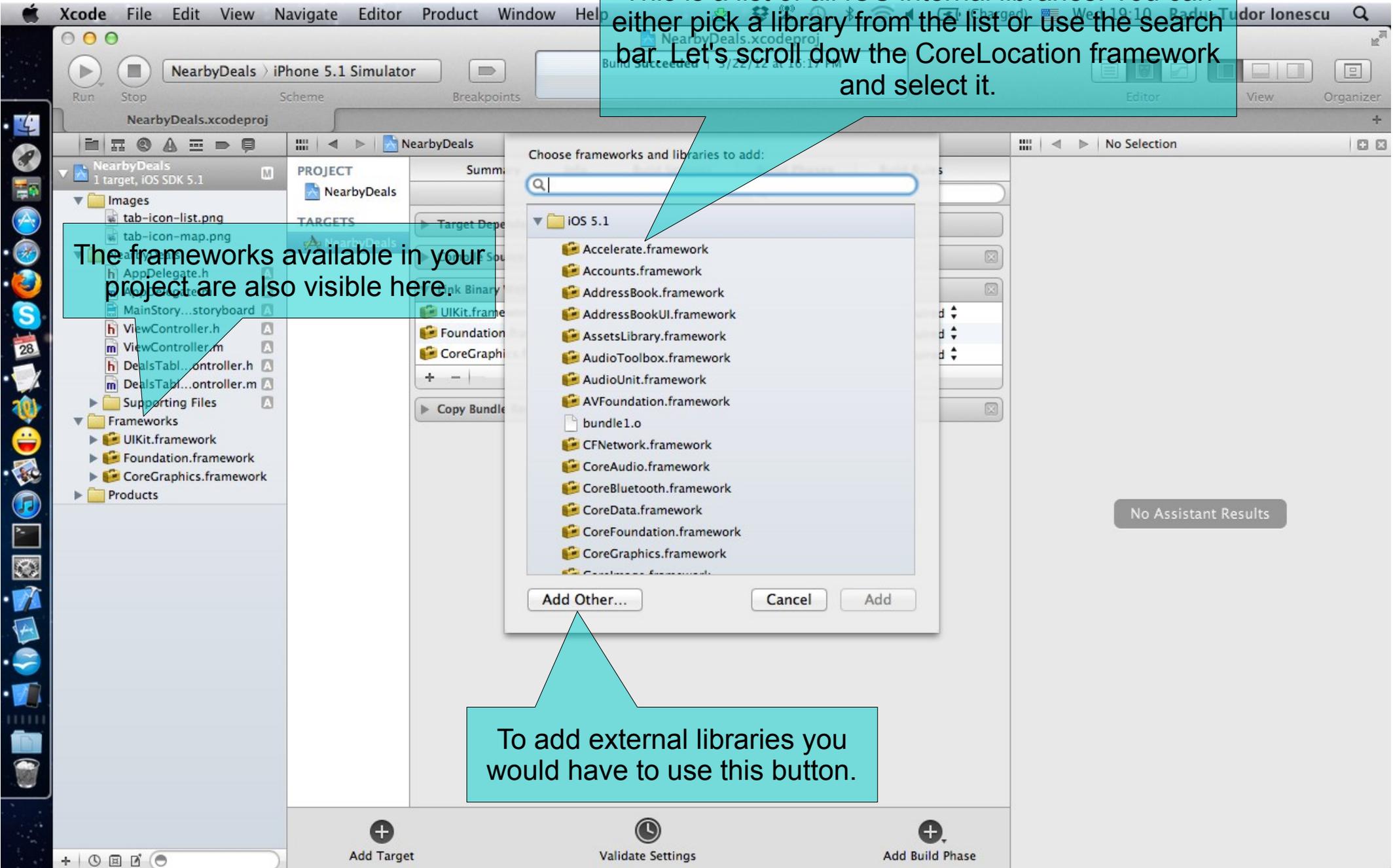
Click here to add the CoreLocation framework.

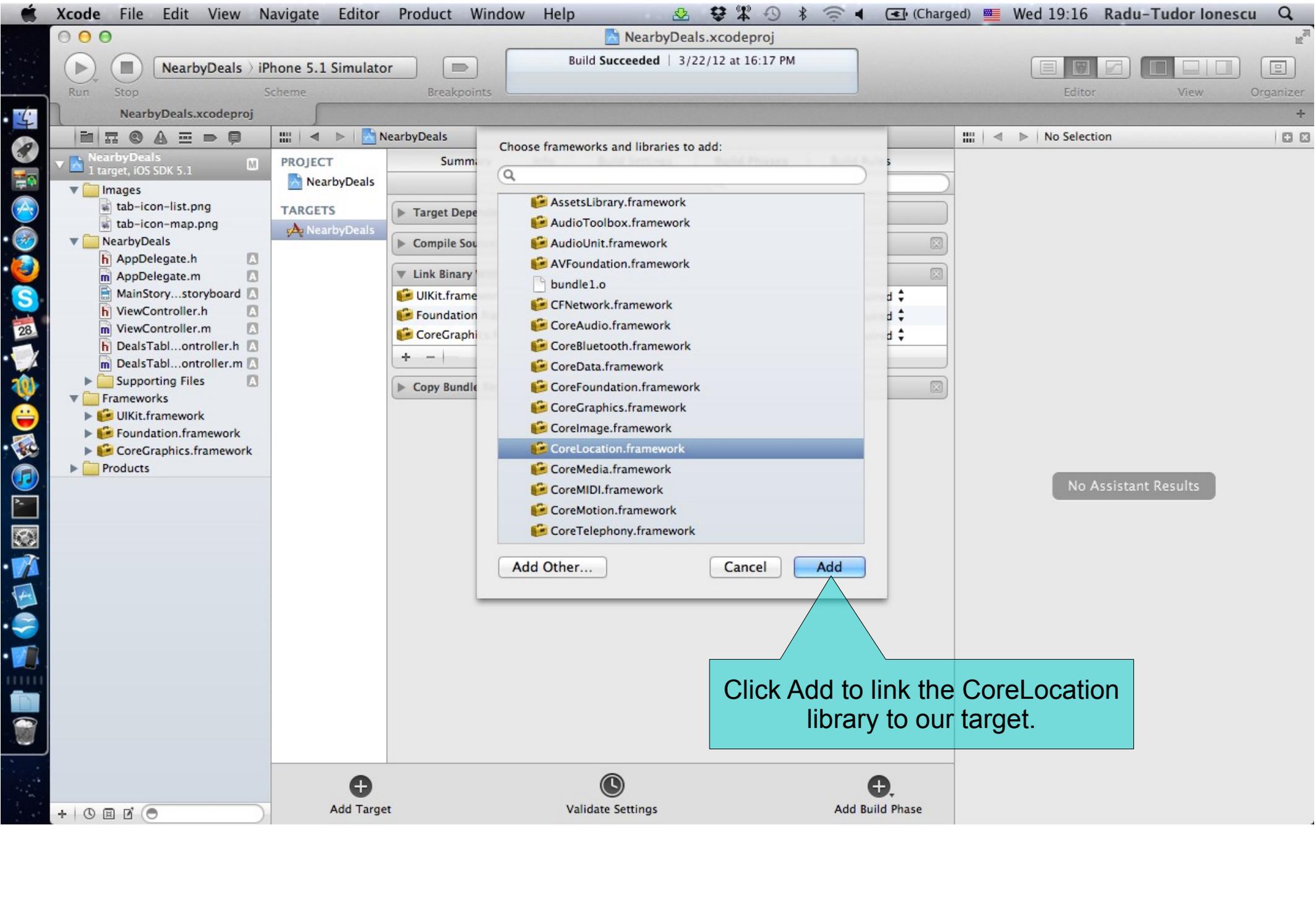
This is were you link the frameworks to your target. Note that the UIKit and Foundation frameworks are already linked here. Every iOS application uses these two very important frameworks. Usually we might want to add new frameworks to our application as we expand its functionality.

This is a list of all iOS internal libraries. You can either pick a library from the list or use the search bar. Let's scroll down the CoreLocation framework and select it.

The frameworks available in your project are also visible here.

To add external libraries you would have to use this button.





Choose frameworks and libraries to add:

- AssetsLibrary.framework
- AudioToolbox.framework
- AudioUnit.framework
- AVFoundation.framework
- bundle1.o
- CFNetwork.framework
- CoreAudio.framework
- CoreBluetooth.framework
- CoreData.framework
- CoreFoundation.framework
- CoreGraphics.framework
- CoreImage.framework
- CoreLocation.framework
- CoreMedia.framework
- CoreMIDI.framework
- CoreMotion.framework
- CoreTelephony.framework

Add Other...

Cancel

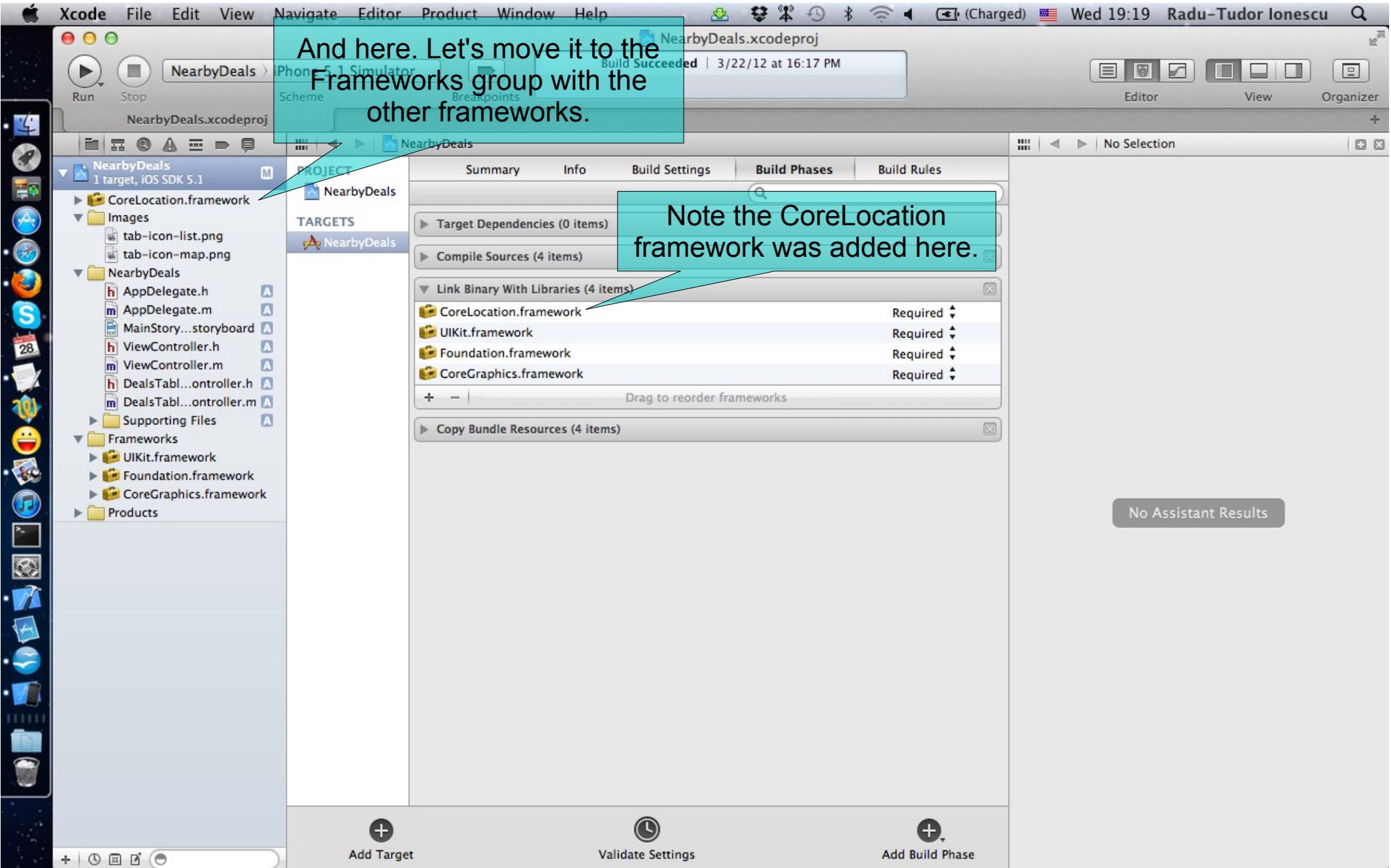
Add

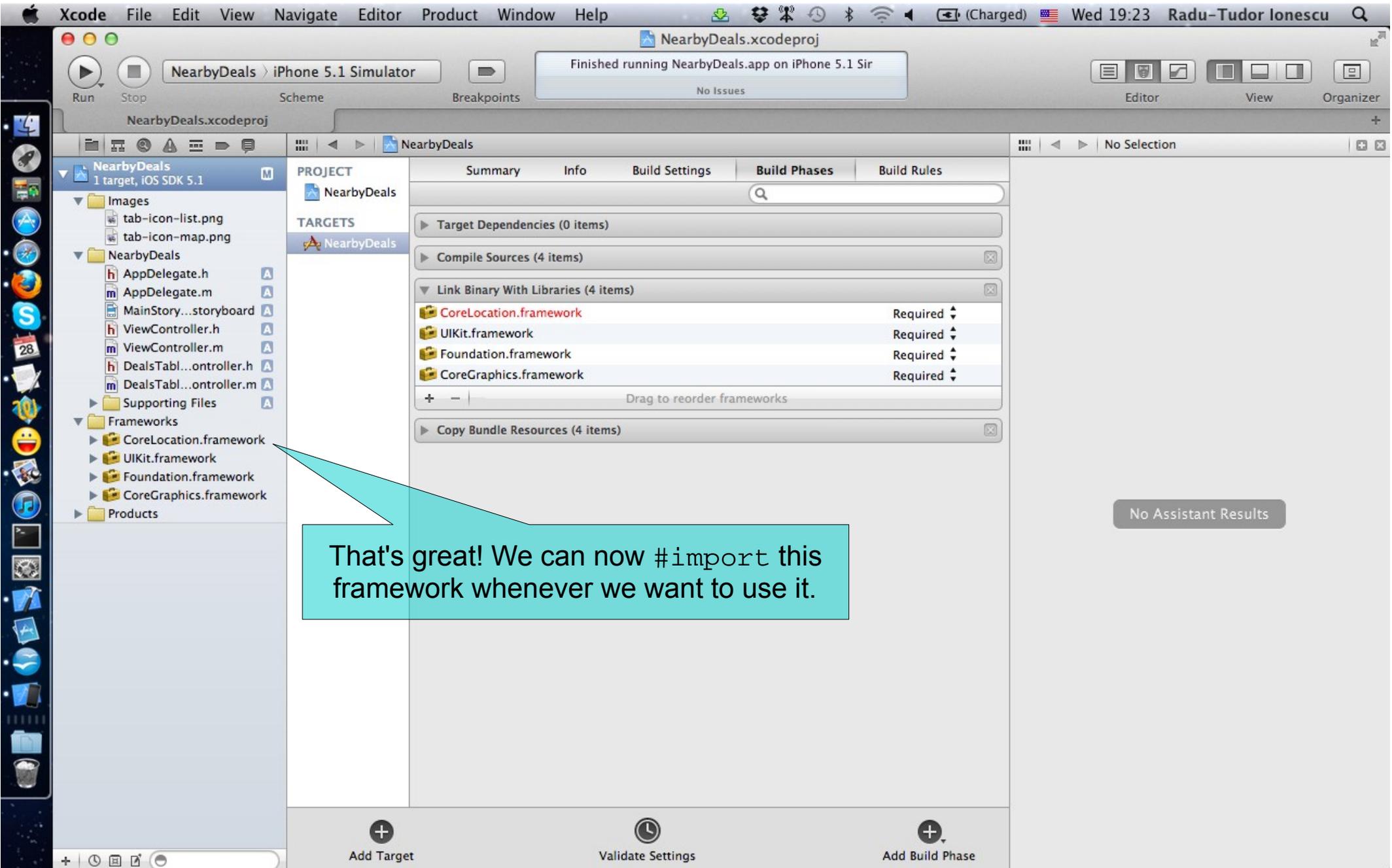
Click Add to link the CoreLocation library to our target.

+ Add Target

⌚ Validate Settings

+ Add Build Phase





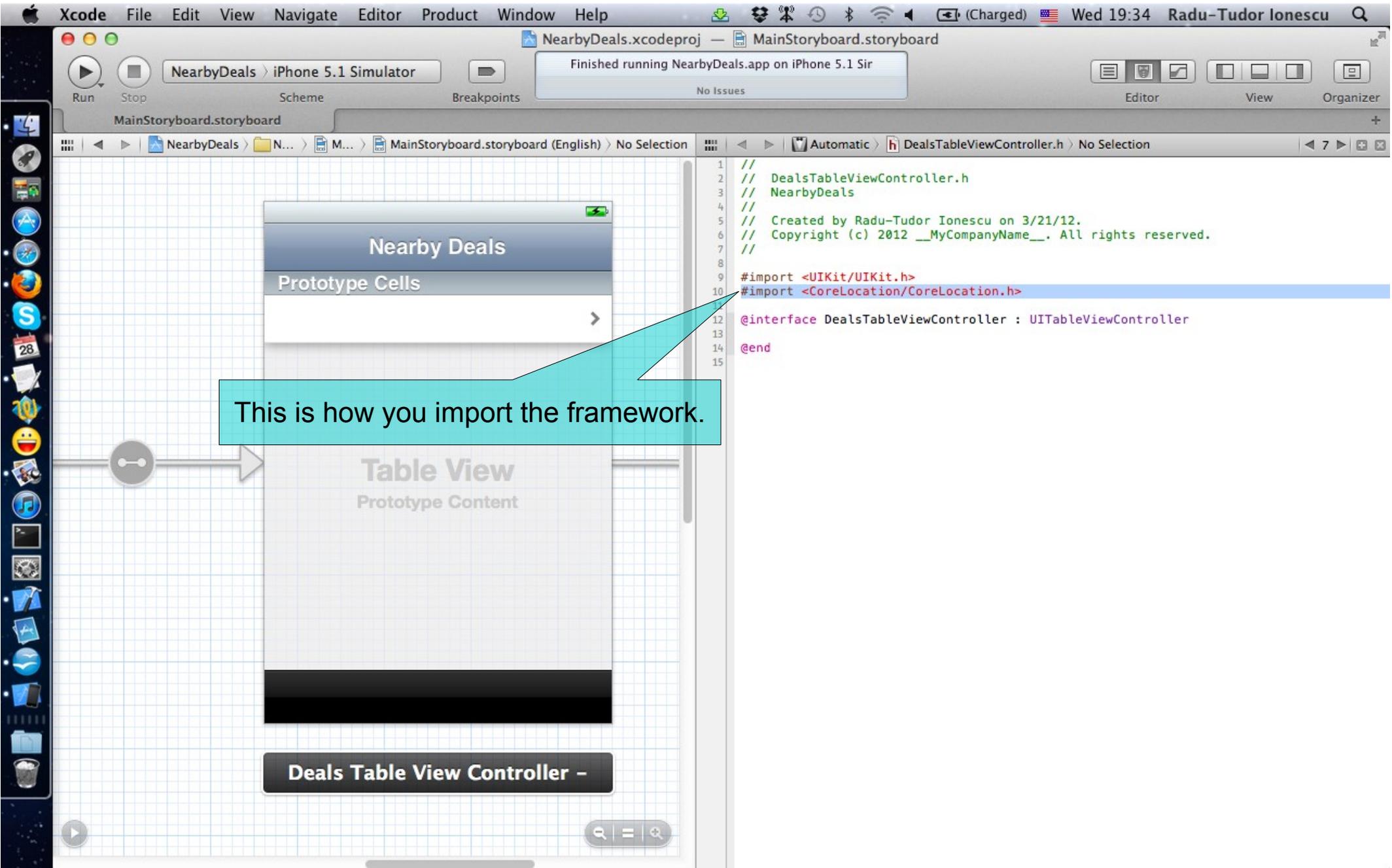
That's great! We can now #import this framework whenever we want to use it.

# Task 1

Task: Add the CoreLocation framework to your project.

5. Select the MainStoryboard.storyboard file.
6. Hide Project Navigator and let's `#import` the CoreLocation framework inside the DealsTableViewController.

We need this framework because we will declare a method soon that has a `CLLocationCoordinate2D` argument. This is a C struct from the CoreLocation library that contains two properties: `latitude` and `longitude`.



## Task 2

Task: Create a request using the GeoAds+ API that will return deals from nearby Bars or Restaurants.

1. Before creating a request from code, we should check out the GeoAds+ API in a web browser. Let's open the API documentation first: [http://www.geoadsplus.com/api\\_documentation](http://www.geoadsplus.com/api_documentation) in Safari.
2. Note that you need an app\_key to make a request to the GeoAds+ API. We will use this one: fe008041973b66760017.
3. Let's try the following request in Safari and see what we get:

```
http://www.geoadsplus.com/ads.xml?  
app\_key=fe008041973b66760017&latitude=44.25&longitude=26.06  
&limit=20&category=Restaurants,Bars
```

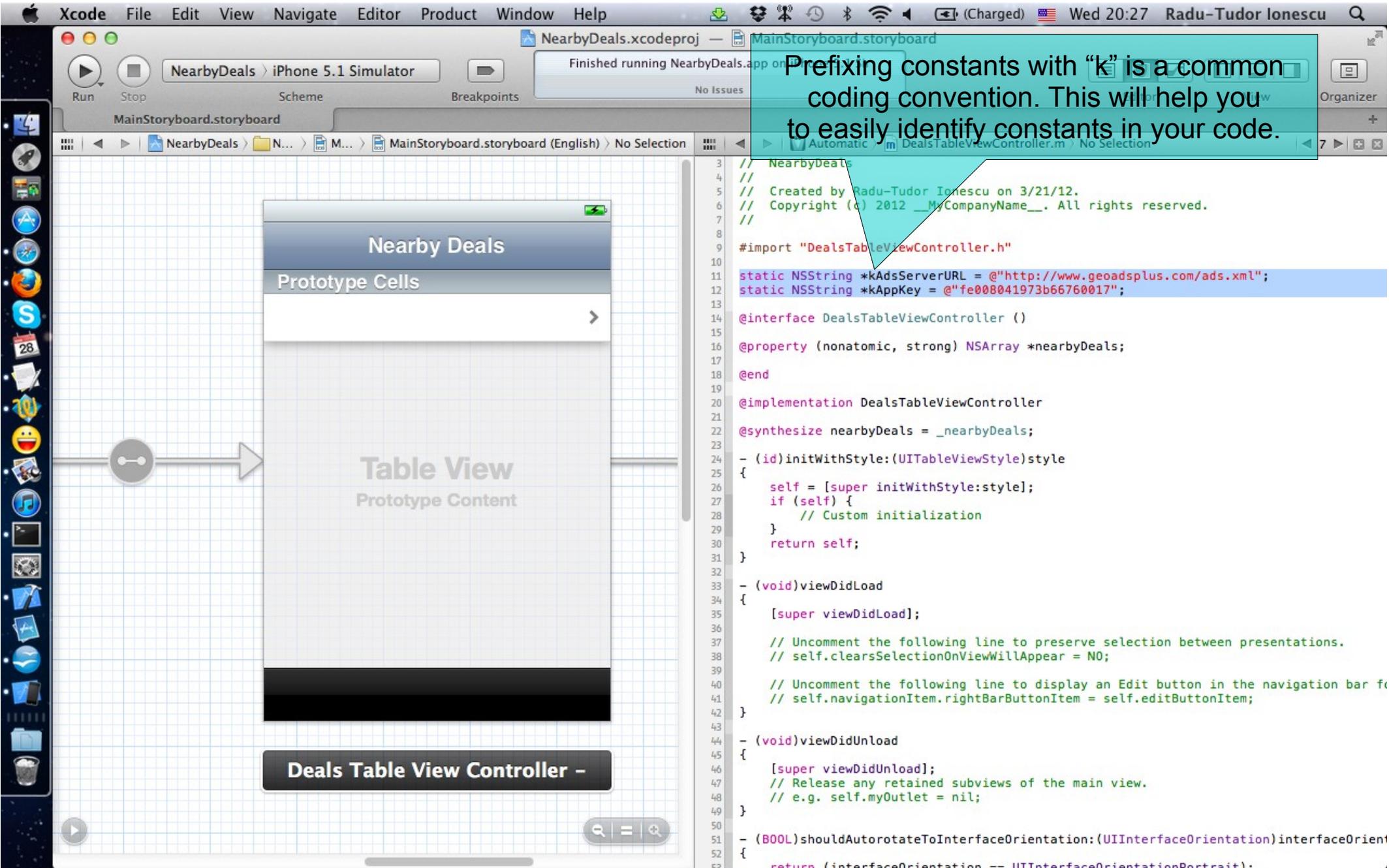
4. Note that you should right-click in Safari and select View Source to see the XML returned by the GeoAds+ API.

## Task 2

Task: Create a request using the GeoAds+ API that will return deals from nearby Bars or Restaurants.

5. Let's return to our project in Xcode. We will declare two constant strings for the GeoAds+ URL and for the app key.
6. Select DealsTableViewController.m in Assistant Editor.
7. Right after the `#import` declare a static `NSString` that holds this URL: <http://www.geoadsplus.com/ads.xml>.
8. In a similar way, declare another static `NSString` to hold our app key: `fe008041973b66760017`.

The next screenshot shows you how to declare these string constants.



## Task 2

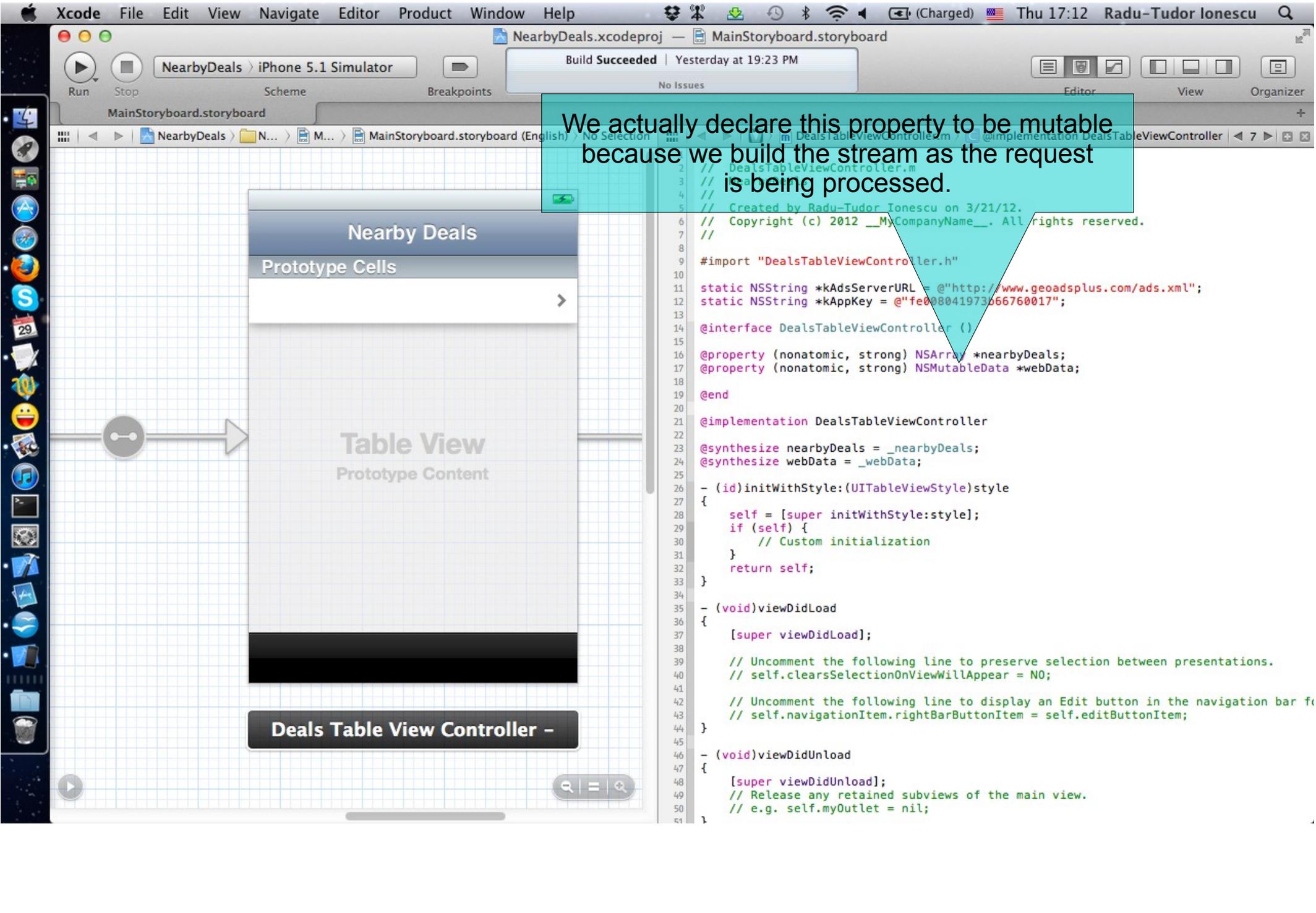
**Task:** Create a request using the GeoAds+ API that will return deals from nearby Bars or Restaurants.

9. Next, we are going to make an asynchronous request to the GeoAds+ API. For this, we implement a method that starts the request by using an `NSURLConnection` object. While this request is processed by the server it can send some messages to its delegate.

We will declare a private `@property` that will contain the stream of bytes (`NSData`) that the delegate receives through the `NSURLConnection` object. Name this property `webData`. Because we (the Controller) have the only reference to it, we use the `strong` storage type.

10. Synthesize this property and prefix its instance variable with underscore.

The next screenshot shows you how to do these steps.



We actually declare this property to be mutable because we build the stream as the request is being processed.

```
1 //
2 // DealsTableViewController.m
3 //
4 //
5 // Created by Radu-Tudor Ionescu on 3/21/12.
6 // Copyright (c) 2012 __MyCompanyName__. All rights reserved.
7 //
8
9 #import "DealsTableViewController.h"
10
11 static NSString *kAdsServerURL = @"http://www.geoadsplus.com/ads.xml";
12 static NSString *kAppKey = @"fe008041973b66760017";
13
14 @interface DealsTableViewController ()
15
16 @property (nonatomic, strong) NSArray *nearbyDeals;
17 @property (nonatomic, strong) NSMutableData *webData;
18
19 @end
20
21 @implementation DealsTableViewController
22
23 @synthesize nearbyDeals = _nearbyDeals;
24 @synthesize webData = _webData;
25
26 - (id)initWithStyle:(UITableViewStyle)style
27 {
28     self = [super initWithStyle:style];
29     if (self) {
30         // Custom initialization
31     }
32     return self;
33 }
34
35 - (void)viewDidLoad
36 {
37     [super viewDidLoad];
38
39     // Uncomment the following line to preserve selection between presentations.
40     // self.clearsSelectionOnViewWillAppear = NO;
41
42     // Uncomment the following line to display an Edit button in the navigation bar for
43     // self.navigationItem.rightBarButtonItem = self.editButtonItem;
44 }
45
46 - (void)viewDidUnload
47 {
48     [super viewDidUnload];
49     // Release any retained subviews of the main view.
50     // e.g. self.myOutlet = nil;
51 }
```

## Task 2

Task: Create a request using the GeoAds+ API that will return deals from nearby Bars or Restaurants.

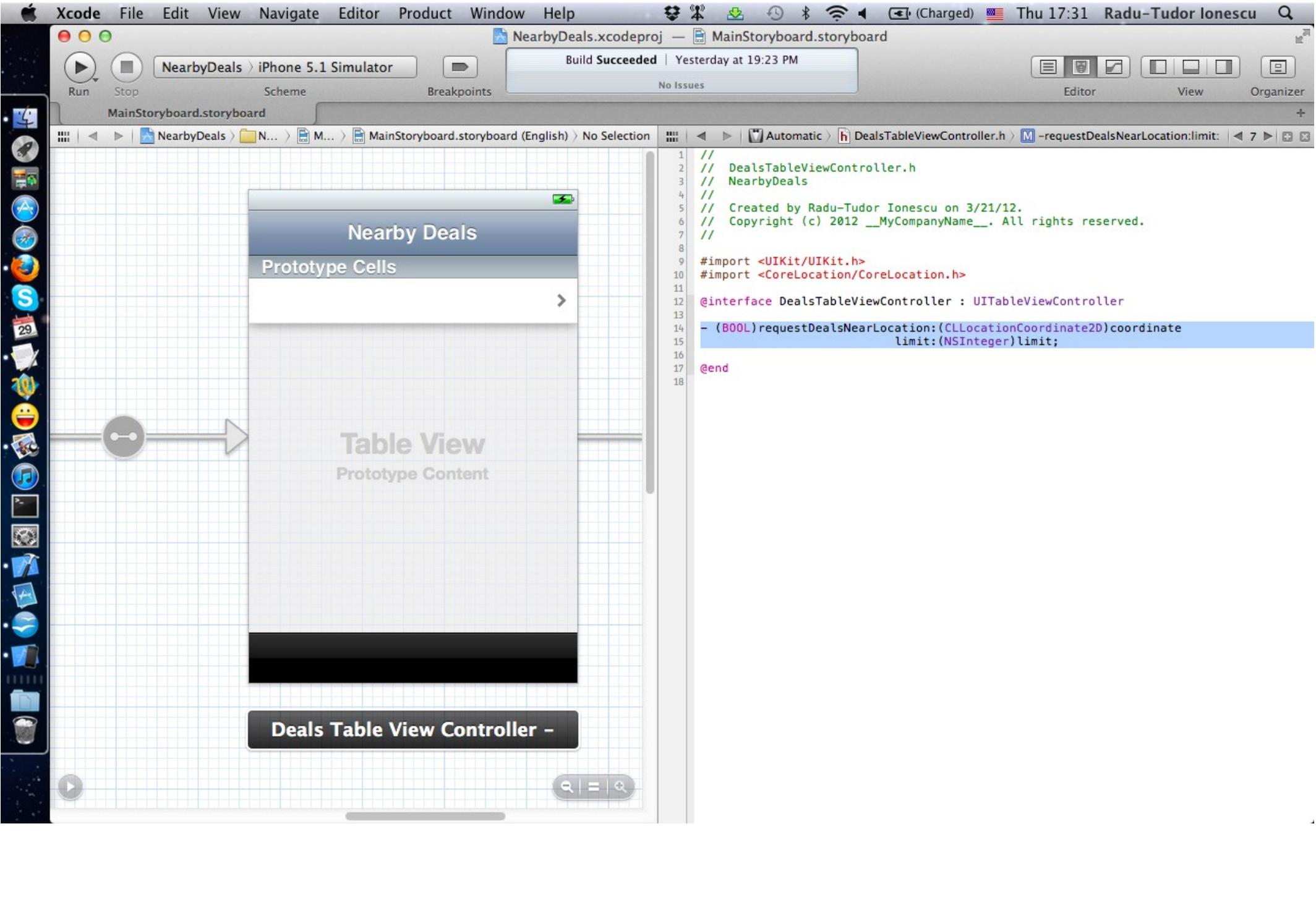
11. Let's declare and implement the method that makes the request for nearby deals to the GeoAds+ API.

Open DealsTableViewController.h in Assistant Editor.

12. Declare the `requestDealsNearLocation:limit:` method that has two arguments: the location (GPS coordinates) of the device and the maximum number of deals to be returned by the server.

This method will return a `BOOL` value that indicates whether the request was successfully started or not.

The next screenshot shows you how to do declare this method.



Run Stop Scheme Breakpoints

Build Succeeded | Yesterday at 19:23 PM  
No Issues

Editor View Organizer

MainStoryboard.storyboard  
NearbyDeals > N... > M... > MainStoryboard.storyboard (English) > No Selection

Automatic > DealsTableViewController.h > -requestDealsNearLocation:limit: < 7 >



```
1 //  
2 // DealsTableViewController.h  
3 // NearbyDeals  
4 //  
5 // Created by Radu-Tudor Ionescu on 3/21/12.  
6 // Copyright (c) 2012 __MyCompanyName__. All rights reserved.  
7 //  
8  
9 #import <UIKit/UIKit.h>  
10 #import <CoreLocation/CoreLocation.h>  
11  
12 @interface DealsTableViewController : UITableViewController  
13  
14 - (BOOL)requestDealsNearLocation:(CLLocationCoordinate2D)coordinate  
15     limit:(NSInteger)limit;  
16  
17 @end  
18
```

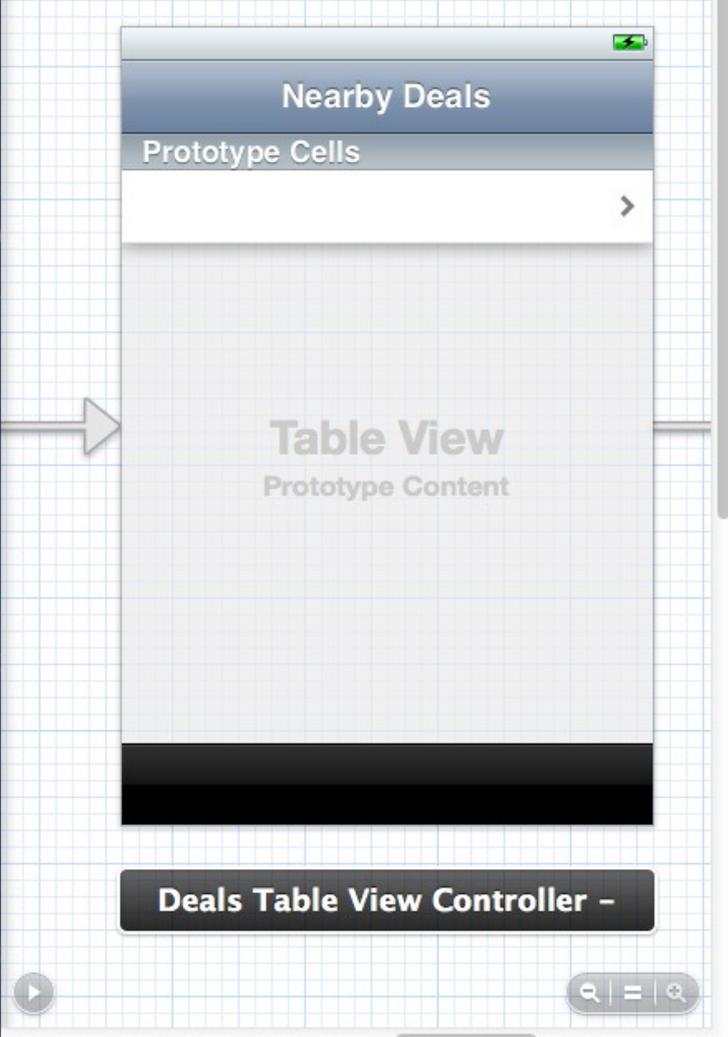
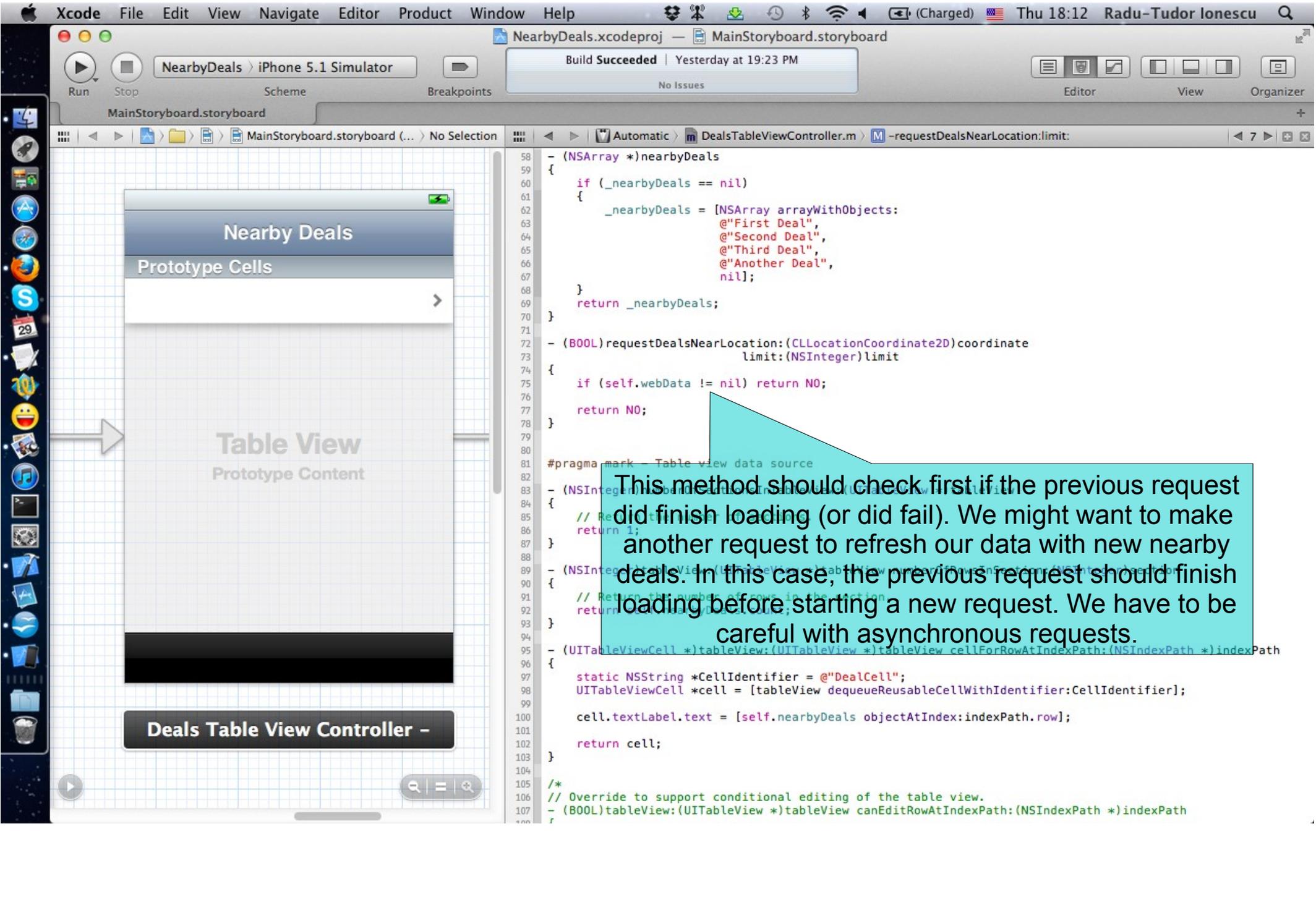
## Task 2

Task: Create a request using the GeoAds+ API that will return deals from nearby Bars or Restaurants.

13. Go back to DealsTableViewController.m in Assistant Editor.

14. Implement the `requestDealsNearLocation:limit:` method after the `nearbyDeals` getter.

The next screenshots guide you through the implementation of this method.

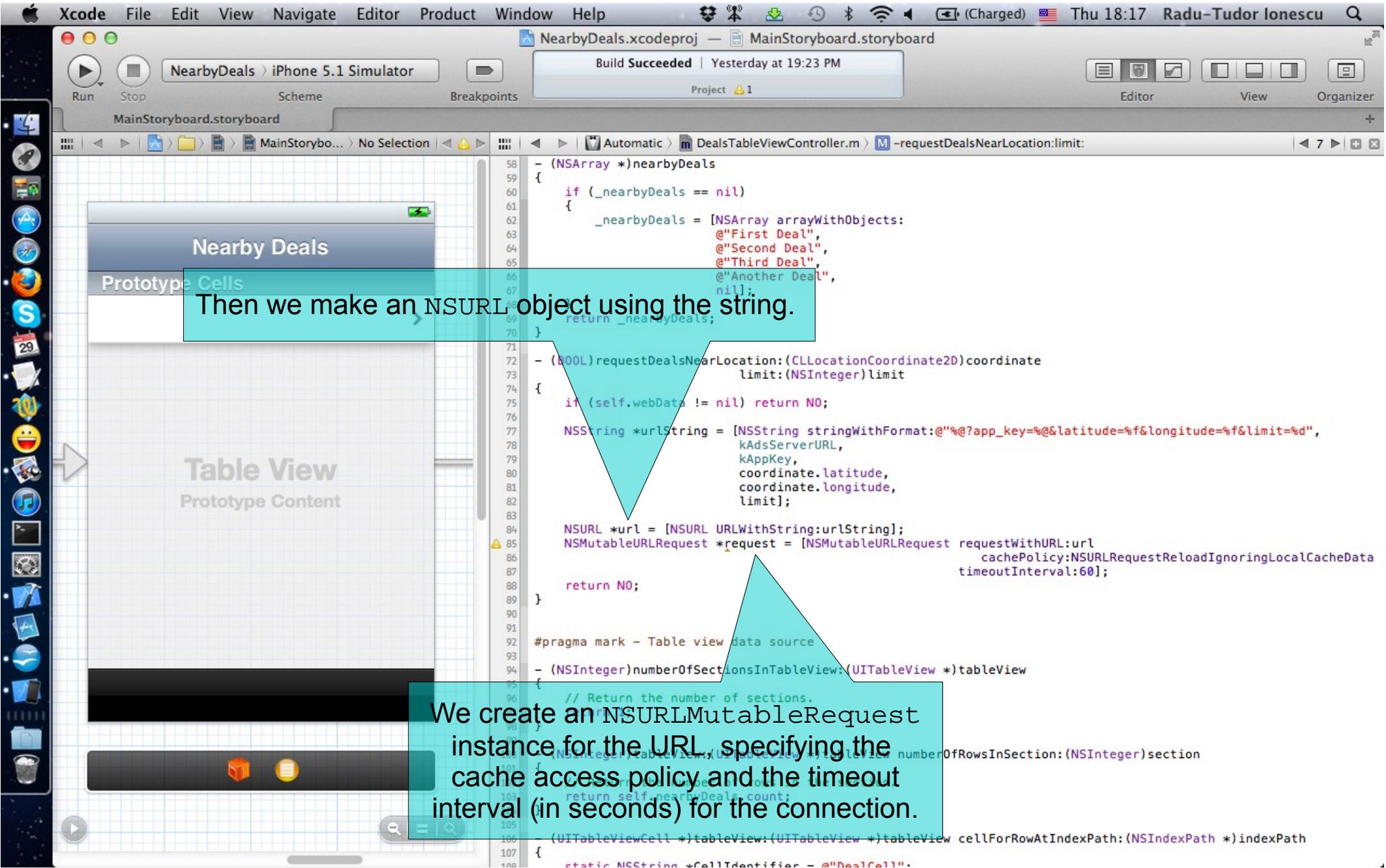


```
58 - (NSArray *)nearbyDeals
59 {
60     if (_nearbyDeals == nil)
61     {
62         _nearbyDeals = [NSArray arrayWithObjects:
63             @"First Deal",
64             @"Second Deal",
65             @"Third Deal",
66             @"Another Deal",
67             nil];
68     }
69     return _nearbyDeals;
70 }
71
72 - (BOOL)requestDealsNearLocation:(CLLocationCoordinate2D)coordinate
73     limit:(NSInteger)limit
74 {
75     if (self.webData != nil) return NO;
76     return NO;
77 }
78
79
80
81 #pragma mark - Table view data source
82
83 - (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
84 {
85     // Return the number of sections in the table view.
86     return 1;
87 }
88
89 - (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
90 {
91     // Return the number of rows in the section.
92     return [_nearbyDeals count];
93 }
94
95 - (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath
96 {
97     static NSString *CellIdentifier = @"DealCell";
98     UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier];
99
100    cell.textLabel.text = [self.nearbyDeals objectAtIndex:indexPath.row];
101
102    return cell;
103 }
104
105 /*
106 // Override to support conditional editing of the table view.
107 - (BOOL)tableView:(UITableView *)tableView canEditRowAtIndexPath:(NSIndexPath *)indexPath
108 {
109     // Return NO if you do not want the specified item to be editable.
110     return NO;
111 }
112 */
```

This method should check first if the previous request did finish loading (or did fail). We might want to make another request to refresh our data with new nearby deals. In this case, the previous request should finish loading before starting a new request. We have to be careful with asynchronous requests.

We make the URL string using the coordinate and limit arguments of this method.

```
58 - (NSArray *)nearbyDeals
59 {
60     if (_nearbyDeals == nil)
61     {
62         _nearbyDeals = [NSArray arrayWithObjects:
63             @"First Deal",
64             @"Second Deal",
65             @"Third Deal",
66             @"Another Deal",
67             nil];
68     }
69     return _nearbyDeals;
70 }
71
72 - (BOOL)requestDealsNearLocation:(CLLocationCoordinate2D)coordinate
73     limit:(NSInteger)limit
74 {
75     if (self.webData != nil) return NO;
76
77     NSString *urlString = [NSString stringWithFormat:@"%s?app_key=%s&latitude=%f&longitude=%f&limit=%d",
78         kAdsServerURL,
79         kAppKey,
80         coordinate.latitude,
81         coordinate.longitude,
82         limit];
83
84     return NO;
85 }
86
87 #pragma mark - Table view data source
88
89 - (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
90 {
91     // Return the number of sections.
92     return 1;
93 }
94
95 - (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
96 {
97     // Return the number of rows in the section.
98     return self.nearbyDeals.count;
99 }
100
101 - (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath
102 {
103     static NSString *CellIdentifier = @"DealCell";
104     UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier];
105
106     cell.textLabel.text = [self.nearbyDeals objectAtIndex:indexPath.row];
107 }
```



Then we make an NSURL object using the string.

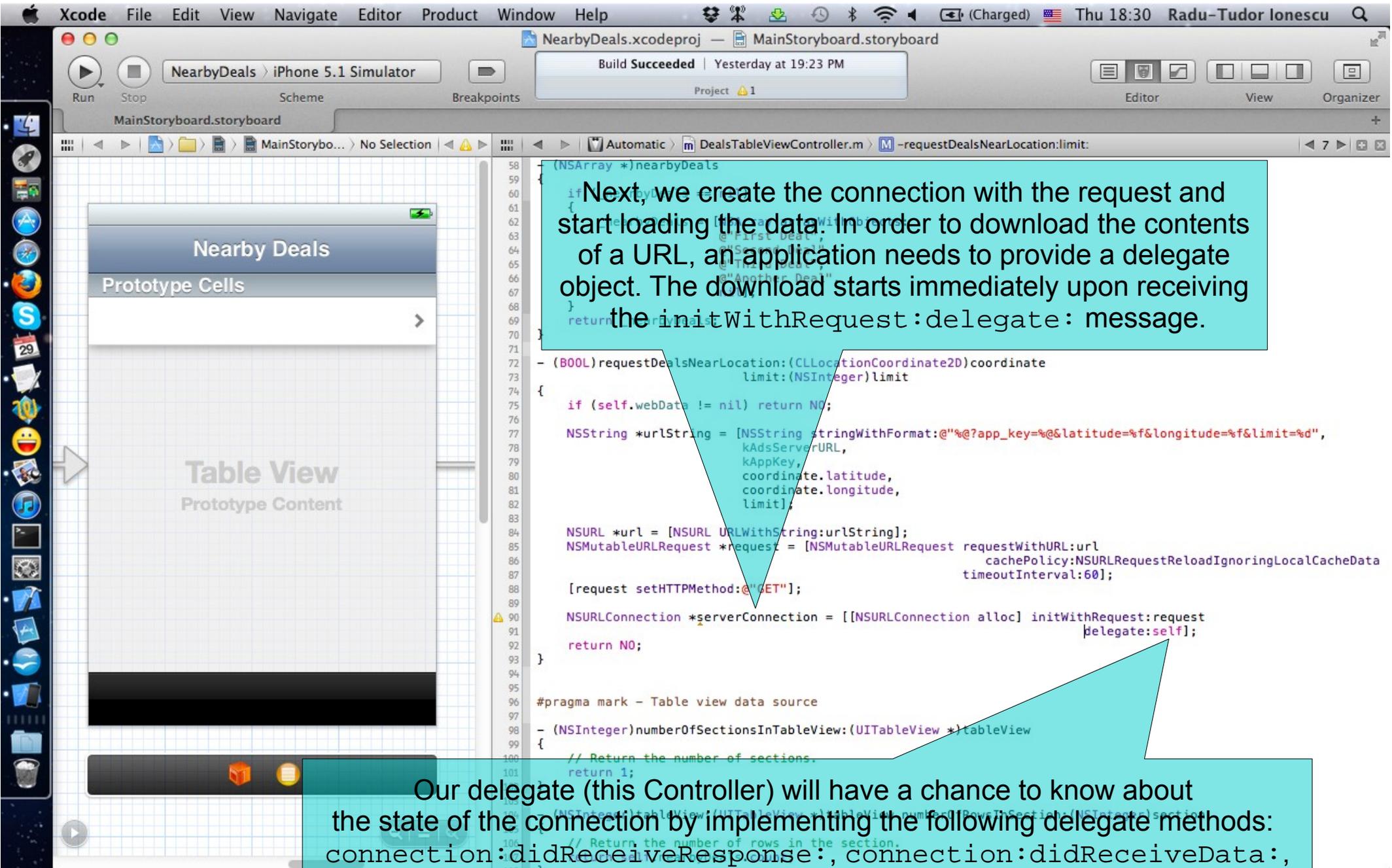
We create an NSMutableURLRequest instance for the URL, specifying the cache access policy and the timeout interval (in seconds) for the connection.

```
58 - (NSArray *)nearbyDeals
59 {
60     if (_nearbyDeals == nil)
61     {
62         _nearbyDeals = [NSArray arrayWithObjects:
63             @"First Deal",
64             @"Second Deal",
65             @"Third Deal",
66             @"Another Deal",
67             nil];
68     }
69     return _nearbyDeals;
70 }
71
72 - (BOOL)requestDealsNearLocation:(CLLocationCoordinate2D)coordinate
73     limit:(NSInteger)limit
74 {
75     if (self.webData != nil) return NO;
76
77     NSString *urlString = [NSString stringWithFormat:@"%s?app_key=%s&latitude=%f&longitude=%f&limit=%d",
78         kAdsServerURL,
79         kAppKey,
80         coordinate.latitude,
81         coordinate.longitude,
82         limit];
83
84     NSURL *url = [NSURL URLWithString:urlString];
85     NSMutableURLRequest *request = [NSMutableURLRequest requestWithURL:url
86         cachePolicy:NSURLRequestReloadIgnoringLocalCacheData
87         timeoutInterval:60];
88
89     return NO;
90 }
91
92 #pragma mark - Table view data source
93
94 - (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
95 {
96     // Return the number of sections.
97     return 1;
98 }
99
100 - (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
101 {
102     return self.nearbyDeals.count;
103 }
104
105 - (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath
106 {
107     static NSString *cellIdentifier = @"DealCell";
108 }
```

The screenshot shows the Xcode IDE with the 'Nearby Deals' app project open. The interface is split into two main panes. On the left is the Storyboard, showing a 'Nearby Deals' app with a title bar and a 'Table View' containing 'Prototype Content'. On the right is the code editor, displaying the implementation of the 'DealsTableViewController.m' file. The code includes a method to return a list of nearby deals and a method to request deals near a location. A blue callout box points to the 'requestDealsNearLocation' method in the code.

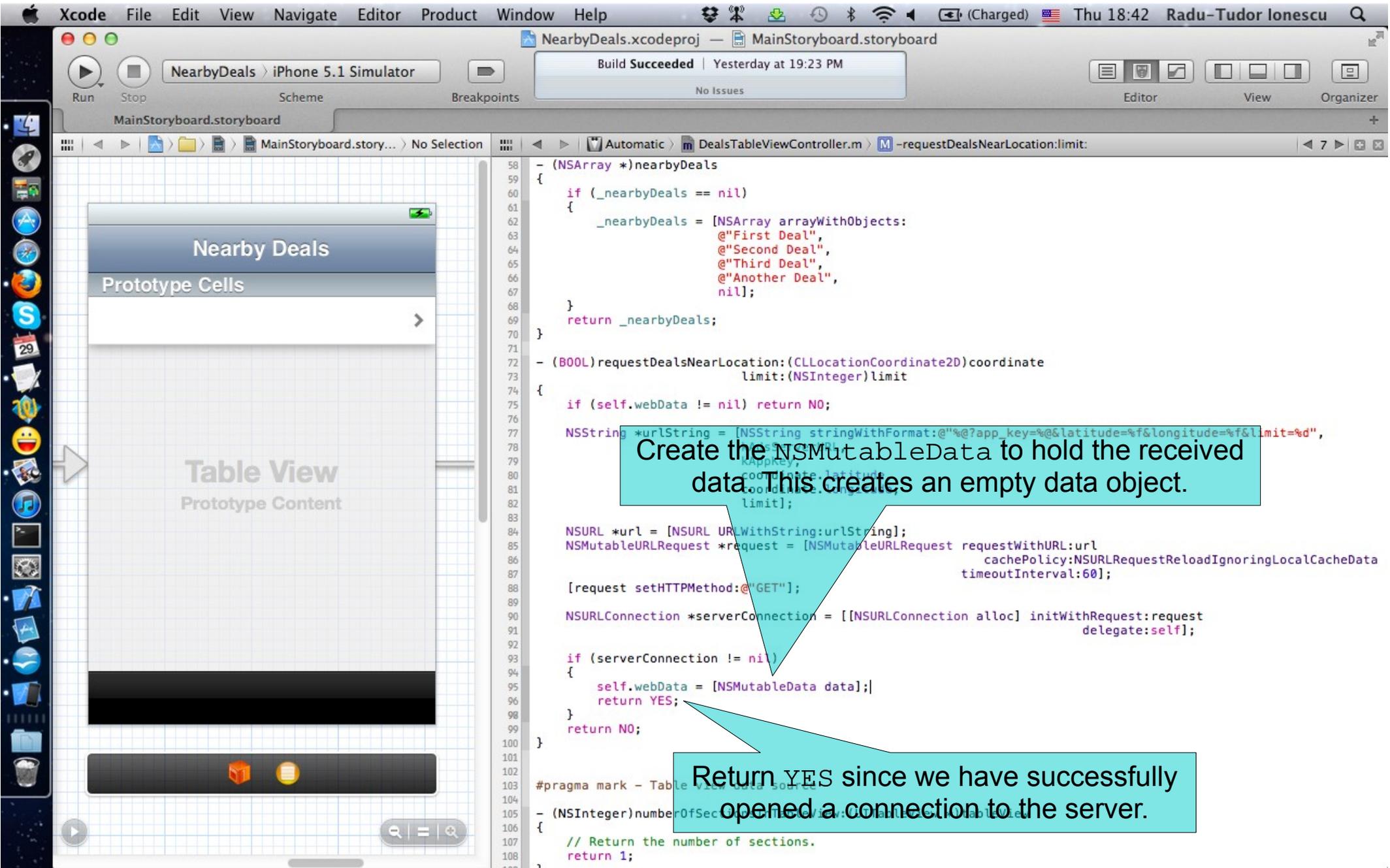
```
58 - (NSArray *)nearbyDeals
59 {
60     if (_nearbyDeals == nil)
61     {
62         _nearbyDeals = [NSArray arrayWithObjects:
63             @"First Deal",
64             @"Second Deal",
65             @"Third Deal",
66             @"Another Deal",
67             nil];
68     }
69     return _nearbyDeals;
70 }
71
72 - (BOOL)requestDealsNearLocation:(CLLocationCoordinate2D)coordinate
73     limit:(NSInteger)limit
74 {
75     if (self.webData != nil) return NO;
76
77     NSString *urlString = [NSString stringWithFormat:@"%s?app_key=%s&latitude=%f&longitude=%f&limit=%d",
78         kAdsServerURL,
79         kAppKey,
80         coordinate.latitude,
81         coordinate.longitude,
82         limit];
83
84     NSURL *url = [NSURL URLWithString:urlString];
85     NSMutableURLRequest *request = [NSMutableURLRequest requestWithURL:url
86         cachePolicy:NSURLRequestReloadIgnoringLocalCacheData
87         timeoutInterval:60];
88
89     [request setHTTPMethod:@"GET"];
90     return NO;
91 }
92
93
94 #pragma mark - Table view data source
95
96 - (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
97 {
98     // Return the number of sections.
99     return 1;
100 }
101
102 - (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
103 {
104     // Return the number of rows in the section.
105     return self.nearbyDeals.count;
106 }
107
108 - (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath
```

NSMutableURLRequest objects accept both standard HTTP methods: GET and POST. In our case we only need to GET some data from the server. If we would have to upload (write) data on a server we should use POST instead.



Next, we create the connection with the request and start loading the data. In order to download the contents of a URL, an application needs to provide a delegate object. The download starts immediately upon receiving the initWithRequest:delegate: message.

Our delegate (this Controller) will have a chance to know about the state of the connection by implementing the following delegate methods: connection:didReceiveResponse:, connection:didReceiveData:, connection:didFailWithError: and connectionDidFinishLoading:.



```
58 - (NSArray *)nearbyDeals
59 {
60     if (_nearbyDeals == nil)
61     {
62         _nearbyDeals = [NSArray arrayWithObjects:
63             @"First Deal",
64             @"Second Deal",
65             @"Third Deal",
66             @"Another Deal",
67             nil];
68     }
69     return _nearbyDeals;
70 }
71
72 - (BOOL)requestDealsNearLocation:(CLLocationCoordinate2D)coordinate
73     limit:(NSInteger)limit
74 {
75     if (self.webData != nil) return NO;
76
77     NSString *urlString = [NSString stringWithFormat:@"%s?app_key=%s&latitude=%f&longitude=%f&limit=%d",
78         kAppKey,
79         coordinate.latitude,
80         coordinate.longitude,
81         limit];
82
83     NSURL *url = [NSURL URLWithString:urlString];
84     NSMutableURLRequest *request = [NSMutableURLRequest requestWithURL:url
85         cachePolicy:NSURLRequestReloadIgnoringLocalCacheData
86         timeoutInterval:60];
87
88     [request setHTTPMethod:@"GET"];
89
90     NSURLConnection *serverConnection = [[NSURLConnection alloc] initWithRequest:request
91         delegate:self];
92
93     if (serverConnection != nil)
94     {
95         self.webData = [NSMutableData data];
96         return YES;
97     }
98     return NO;
99 }
100
101
102
103 #pragma mark - Table View Data Source
104
105 - (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
106 {
107     // Return the number of sections.
108     return 1;
109 }
```

Create the NSMutableData to hold the received data. This creates an empty data object.

Return YES since we have successfully opened a connection to the server.

## Task 2

Task: Create a request using the GeoAds+ API that will return deals from nearby Bars or Restaurants.

15. We should continue by implementing the methods for the `NSURLConnection` delegate. Let's mark the section of code that will contain these delegate methods using the `#pragma mark` compiler directive.
16. Name this section "NSURLConnection load callbacks".

The next screenshot shows where to add this section.

Xcode File Edit View Navigate Editor Product Window Help

NearbyDeals.xcodeproj — MainStoryboard.storyboard

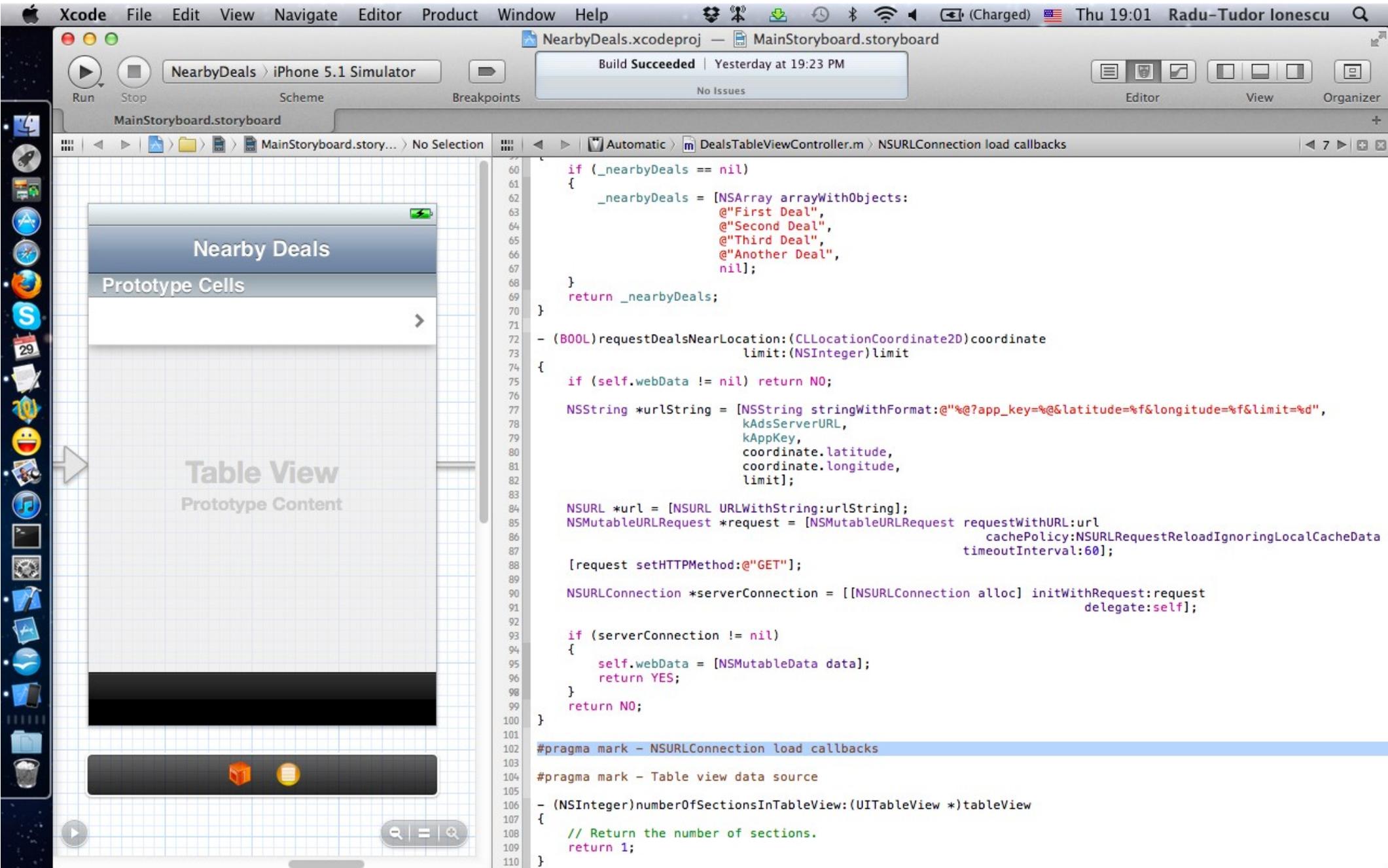
Build Succeeded | Yesterday at 19:23 PM  
No Issues

NearbyDeals > iPhone 5.1 Simulator

MainStoryboard.storyboard

MainStoryboard.story... > No Selection

Automatic > DealsTableViewCell.m > NSURLConnection load callbacks



```
60 if (_nearbyDeals == nil)
61 {
62     _nearbyDeals = [NSArray arrayWithObjects:
63         @"First Deal",
64         @"Second Deal",
65         @"Third Deal",
66         @"Another Deal",
67         nil];
68 }
69 return _nearbyDeals;
70 }
71
72 - (BOOL)requestDealsNearLocation:(CLLocationCoordinate2D)coordinate
73     limit:(NSInteger)limit
74 {
75     if (self.webData != nil) return NO;
76
77     NSString *urlString = [NSString stringWithFormat:@"%s?app_key=%s&latitude=%f&longitude=%f&limit=%d",
78         kAdsServerURL,
79         kAppKey,
80         coordinate.latitude,
81         coordinate.longitude,
82         limit];
83
84     NSURL *url = [NSURL URLWithString:urlString];
85     NSMutableURLRequest *request = [NSMutableURLRequest requestWithURL:url
86                                     cachePolicy:NSURLRequestReloadIgnoringLocalCacheData
87                                     timeoutInterval:60];
88     [request setHTTPMethod:@"GET"];
89
90     NSURLConnection *serverConnection = [[NSURLConnection alloc] initWithRequest:request
91                                         delegate:self];
92
93     if (serverConnection != nil)
94     {
95         self.webData = [NSMutableData data];
96         return YES;
97     }
98     return NO;
99 }
100
101 #pragma mark - NSURLConnection load callbacks
102
103 #pragma mark - Table view data source
104
105 - (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
106 {
107     // Return the number of sections.
108     return 1;
109 }
110
```

## Task 2

**Task:** Create a request using the GeoAds+ API that will return deals from nearby Bars or Restaurants.

17. When the server has provided sufficient data to create an `NSURLResponse` object, the delegate receives the following message: `connection:didReceiveResponse:`.

You should be prepared for your delegate to receive the `connection:didReceiveResponse: message` multiple times for a single connection. This message can be sent due to server redirects, or in rare cases multi-part MIME documents. Each time the delegate receives the `connection:didReceiveResponse: message`, it should reset any progress indication and discard all previously received data.

The next screenshot shows how to implement this callback method.

Xcode File Edit View Navigate Editor Product Window Help

NearbyDeals.xcodeproj — MainStoryboard.storyboard

Build Succeeded | Yesterday at 19:23 PM

No Issues

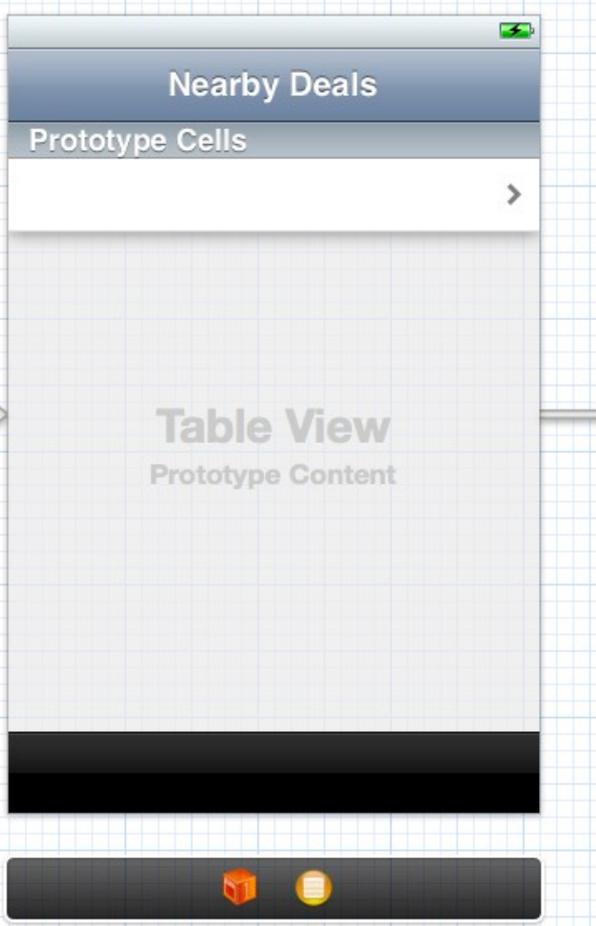
NearbyDeals > iPhone 5.1 Simulator

Run Stop Scheme Breakpoints Editor View Organizer

MainStoryboard.storyboard

MainStoryboard.story... > No Selection

Automatic > DealsTableViewController.m > M -connection:didReceiveResponse:



```
95     self.webData = [NSMutableArray data];
96     return YES;
97 }
98     return NO;
99 }
100 }
101 }
102 #pragma mark - NSURLConnection load callbacks
103 }
104 - (void)connection:(NSURLConnection *)connection didReceiveResponse:(NSURLResponse *)response
105 {
106     [self.webData setLength:0];
107 }
108 }
109 #pragma mark - Table view data source
110 }
111 - (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
112 {
113     // Return the number of sections.
114     return 1;
115 }
116 }
117 - (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
118 {
119     // Return the number of rows in the section.
120     return self.nearbyDeals.count;
121 }
122 }
123 - (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath
124 {
125     static NSString *CellIdentifier = @"DealCell";
126     UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier];
127
128     cell.textLabel.text = [self.nearbyDeals objectAtIndex:indexPath.row];
129
130     return cell;
131 }
132 }
133 /*
134 // Override to support conditional editing of the table view.
135 - (BOOL)tableView:(UITableView *)tableView canEditRowAtIndexPath:(NSIndexPath *)indexPath
136 {
137     // Return NO if you do not want the specified item to be editable.
138     return YES;
139 }
140 */
141 /*
142 // Override to support editing the table view.
143 - (void)tableView:(UITableView *)tableView commitEditingStyle:(UITableViewCellEditingStyle)editingStyle forRowAtIndexPath:(NSIndexPath *)indexPath
144 {
145     // ...
146 }
```

## Task 2

Task: Create a request using the GeoAds+ API that will return deals from nearby Bars or Restaurants.

18. Our Table View Controller (the delegate) is periodically sent `connection:didReceiveData:` messages as the data is received. The delegate implementation is responsible for storing the newly received data.

We should append the new data to `webData` in this method.

Note that you can also use this method to provide an indication of the connection's progress to the user. This is useful when we transfer large files from/to a server.

The next screenshot shows how to implement this callback method.

Xcode File Edit View Navigate Editor Product Window Help

NearbyDeals.xcodeproj — MainStoryboard.storyboard

Build Succeeded | Yesterday at 19:23 PM  
No Issues

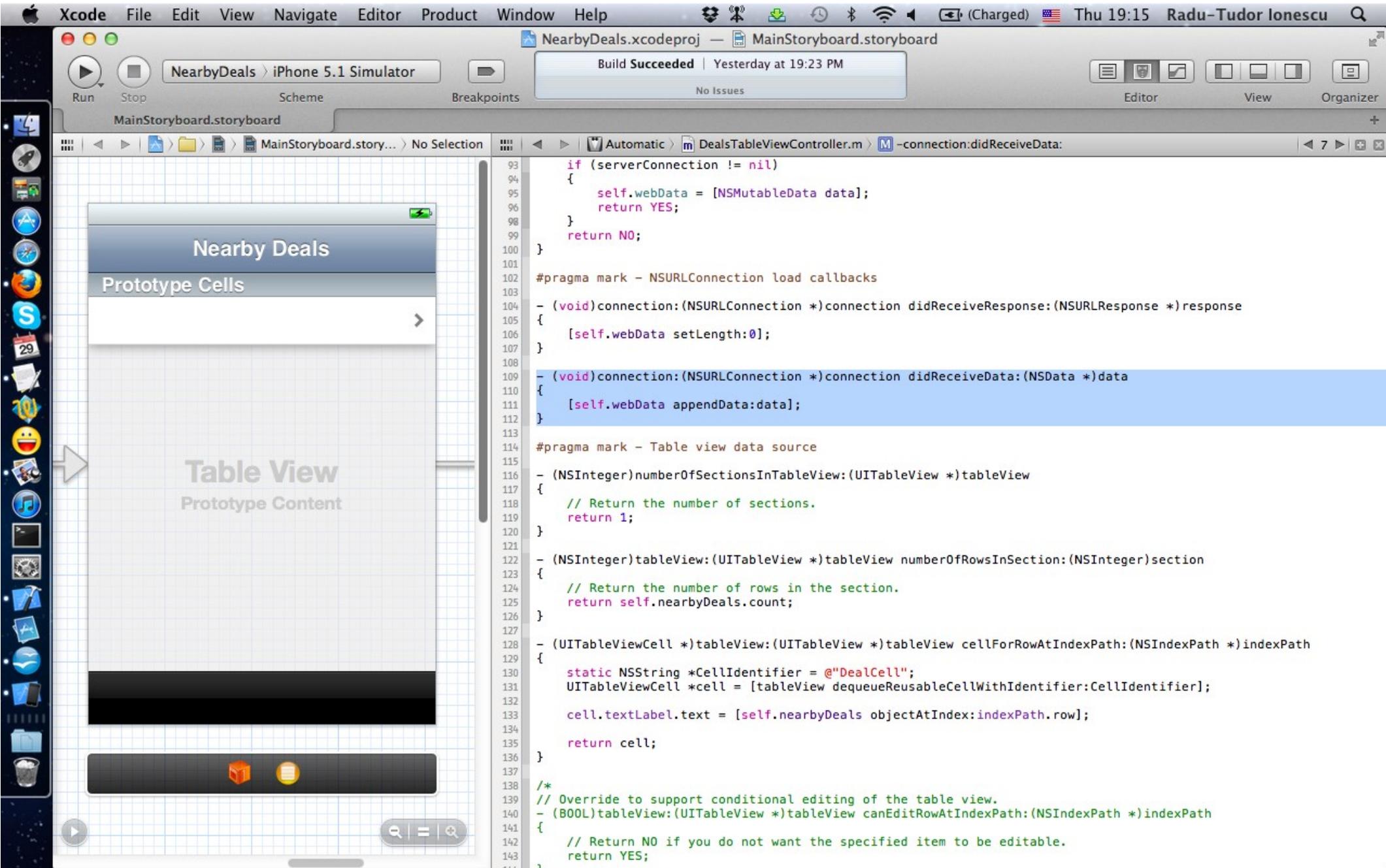
NearbyDeals > iPhone 5.1 Simulator

Run Stop Scheme Breakpoints Editor View Organizer

MainStoryboard.storyboard

MainStoryboard.story... > No Selection

Automatic > DealsTableViewController.m > M -connection:didReceiveData:



```
93     if (serverConnection != nil)
94     {
95         self.webData = [NSMutableData data];
96         return YES;
97     }
98     return NO;
99 }
100 }
101
102 #pragma mark - NSURLConnection load callbacks
103
104 - (void)connection:(NSURLConnection *)connection didReceiveResponse:(NSURLResponse *)response
105 {
106     [self.webData setLength:0];
107 }
108
109 - (void)connection:(NSURLConnection *)connection didReceiveData:(NSData *)data
110 {
111     [self.webData appendData:data];
112 }
113
114 #pragma mark - Table view data source
115
116 - (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
117 {
118     // Return the number of sections.
119     return 1;
120 }
121
122 - (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
123 {
124     // Return the number of rows in the section.
125     return self.nearbyDeals.count;
126 }
127
128 - (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath
129 {
130     static NSString *CellIdentifier = @"DealCell";
131     UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier];
132
133     cell.textLabel.text = [self.nearbyDeals objectAtIndex:indexPath.row];
134
135     return cell;
136 }
137
138 /*
139 // Override to support conditional editing of the table view.
140 - (BOOL)tableView:(UITableView *)tableView canEditRowAtIndexPath:(NSIndexPath *)indexPath
141 {
142     // Return NO if you do not want the specified item to be editable.
143     return YES;
144 }
```

## Task 2

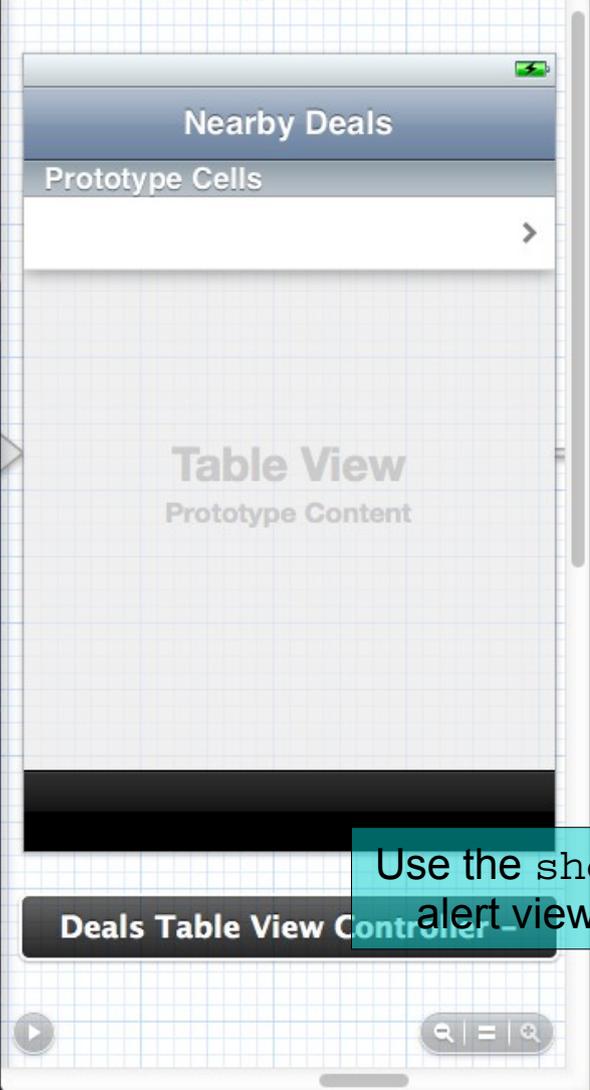
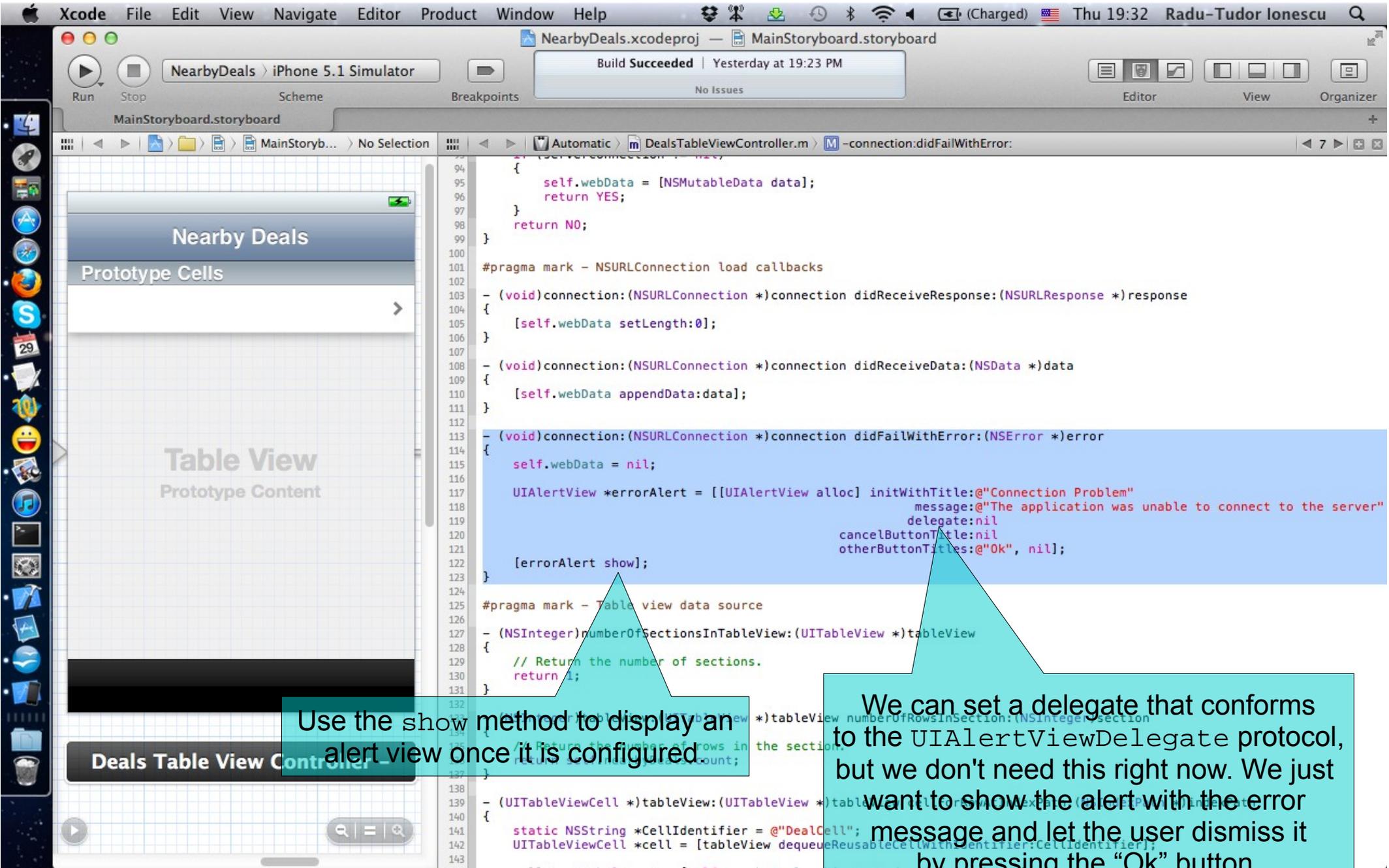
Task: Create a request using the GeoAds+ API that will return deals from nearby Bars or Restaurants.

19. If an error is encountered during the download, the delegate receives a `connection:didFailWithError:` message. The `NSError` object passed as the parameter specifies the details of the error. After our delegate object receives the `connection:didFailWithError:` message, it receives no further delegate messages for the specified connection.

We set `webData` to `nil` in this method because we have finished the request (without success).

20. It would be nice to let the user know that we have a connection problem. A standard way to present ad-hoc messages to the user is to use an `UIAlertView` object. We are going to build an alert programmatically that will display an error message on the screen. It will have a dismiss button too.

The next screenshot shows how to implement this callback method.



```
94 {
95     self.webData = [NSMutableData data];
96     return YES;
97 }
98 return NO;
99 }
100
101 #pragma mark - NSURLConnection load callbacks
102
103 - (void)connection:(NSURLConnection *)connection didReceiveResponse:(NSURLResponse *)response
104 {
105     [self.webData setLength:0];
106 }
107
108 - (void)connection:(NSURLConnection *)connection didReceiveData:(NSData *)data
109 {
110     [self.webData appendData:data];
111 }
112
113 - (void)connection:(NSURLConnection *)connection didFailWithError:(NSError *)error
114 {
115     self.webData = nil;
116
117     UIAlertView *errorAlert = [[UIAlertView alloc] initWithTitle:@"Connection Problem"
118                                                                    message:@"The application was unable to connect to the server"
119                                                                    delegate:nil
120                                                                    cancelButtonTitle:nil
121                                                                    otherButtonTitles:@"Ok", nil];
122
123     [errorAlert show];
124 }
125
126 #pragma mark - Table view data source
127
128 - (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
129 {
130     // Return the number of sections.
131     return 1;
132 }
133
134 - (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
135 {
136     // Return the number of rows in the section.
137     return 1;
138 }
139
140 - (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath
141 {
142     static NSString *CellIdentifier = @"DealCell";
143     UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier];
144 }
```

Use the show method to display an alert view once it is configured.

We can set a delegate that conforms to the UIAlertViewDelegate protocol, but we don't need this right now. We just want to show the alert with the error message and let the user dismiss it by pressing the "Ok" button.

## Task 2

**Task:** Create a request using the GeoAds+ API that will return deals from nearby Bars or Restaurants.

21. Finally, if the connection succeeds in downloading the request, the delegate receives the `connectionDidFinishLoading:` message. The delegate will receive no further messages for the connection.

Let's transform the received bytes into a string and print it with an `NSLog` for now.

This represents the simplest implementation of a client using `NSURLConnection`. Additional delegate methods provide the ability to customize the handling of server redirects, authorization requests and caching of the response.

The next screenshot shows how to implement this callback method.

22. Run the application in iOS Simulator and check out the console to see if it gets the XML from the server.

Xcode File Edit View Navigate Editor Product Window Help

NearbyDeals.xcodeproj — MainStoryboard.storyboard

Build Succeeded | Yesterday at 19:23 PM

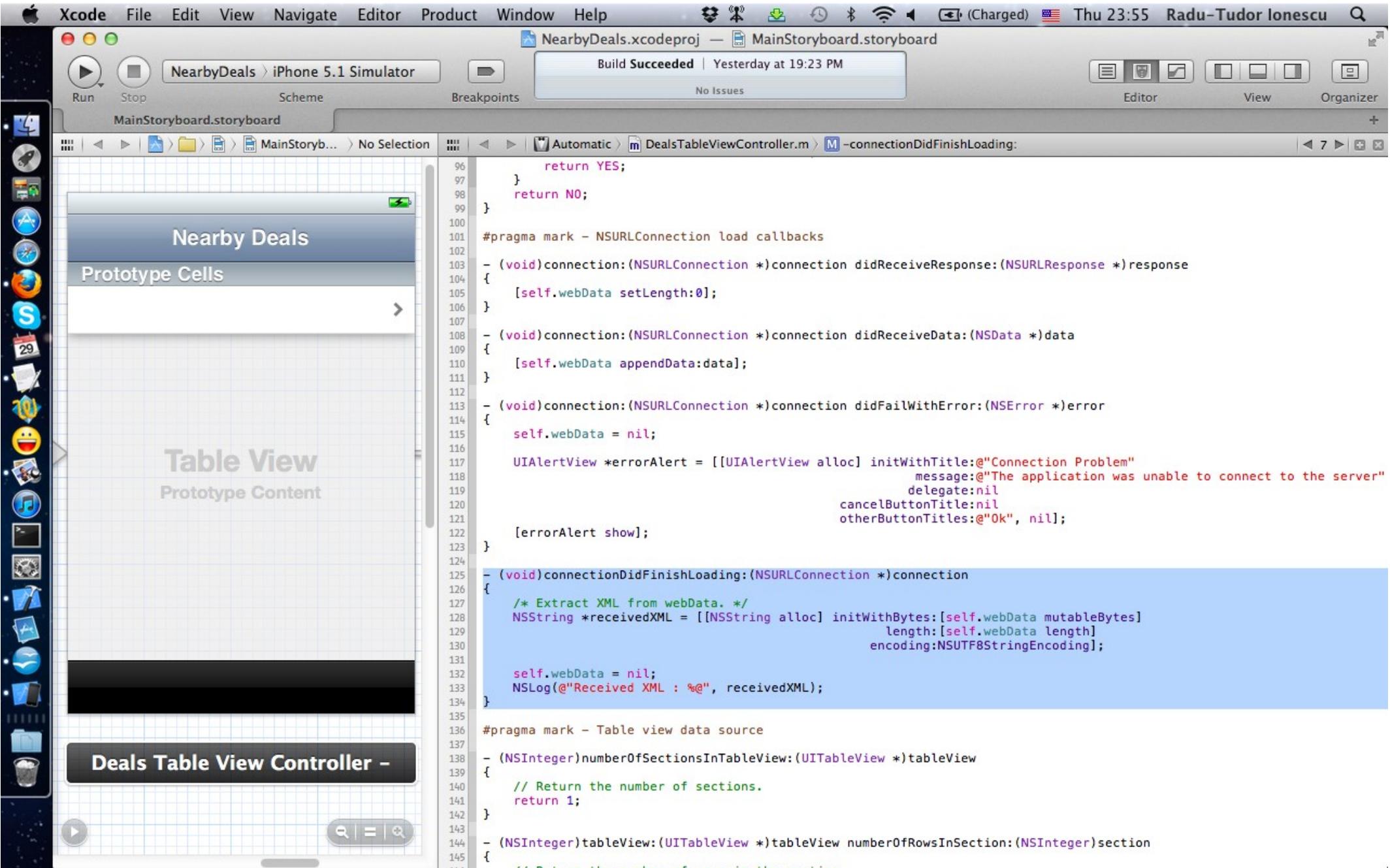
No Issues

NearbyDeals > iPhone 5.1 Simulator

Run Stop Scheme Breakpoints Editor View Organizer

MainStoryboard.storyboard

MainStoryb... > No Selection Automatic > DealsTableViewController.m > M -connectionDidFinishLoading:



```
96     return YES;
97   }
98   return NO;
99 }
100
101 #pragma mark - NSURLConnection load callbacks
102
103 - (void)connection:(NSURLConnection *)connection didReceiveResponse:(NSURLResponse *)response
104 {
105     [self.webData setLength:0];
106 }
107
108 - (void)connection:(NSURLConnection *)connection didReceiveData:(NSData *)data
109 {
110     [self.webData appendData:data];
111 }
112
113 - (void)connection:(NSURLConnection *)connection didFailWithError:(NSError *)error
114 {
115     self.webData = nil;
116
117     UIAlertView *errorAlert = [[UIAlertView alloc] initWithTitle:@"Connection Problem"
118                                     message:@"The application was unable to connect to the server"
119                                     delegate:nil
120                                     cancelButtonTitle:nil
121                                     otherButtonTitles:@"Ok", nil];
122
123     [errorAlert show];
124 }
125
126 - (void)connectionDidFinishLoading:(NSURLConnection *)connection
127 {
128     /* Extract XML from webData. */
129     NSString *receivedXML = [[NSString alloc] initWithBytes:[self.webData mutableBytes]
130                                                         length:[self.webData length]
131                                                         encoding:NSUTF8StringEncoding];
132
133     self.webData = nil;
134     NSLog(@"Received XML : %@", receivedXML);
135 }
136
137 #pragma mark - Table view data source
138
139 - (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
140 {
141     // Return the number of sections.
142     return 1;
143 }
144
145 - (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
146 {
147     // Return the number of rows in the section.
```

## Task 2

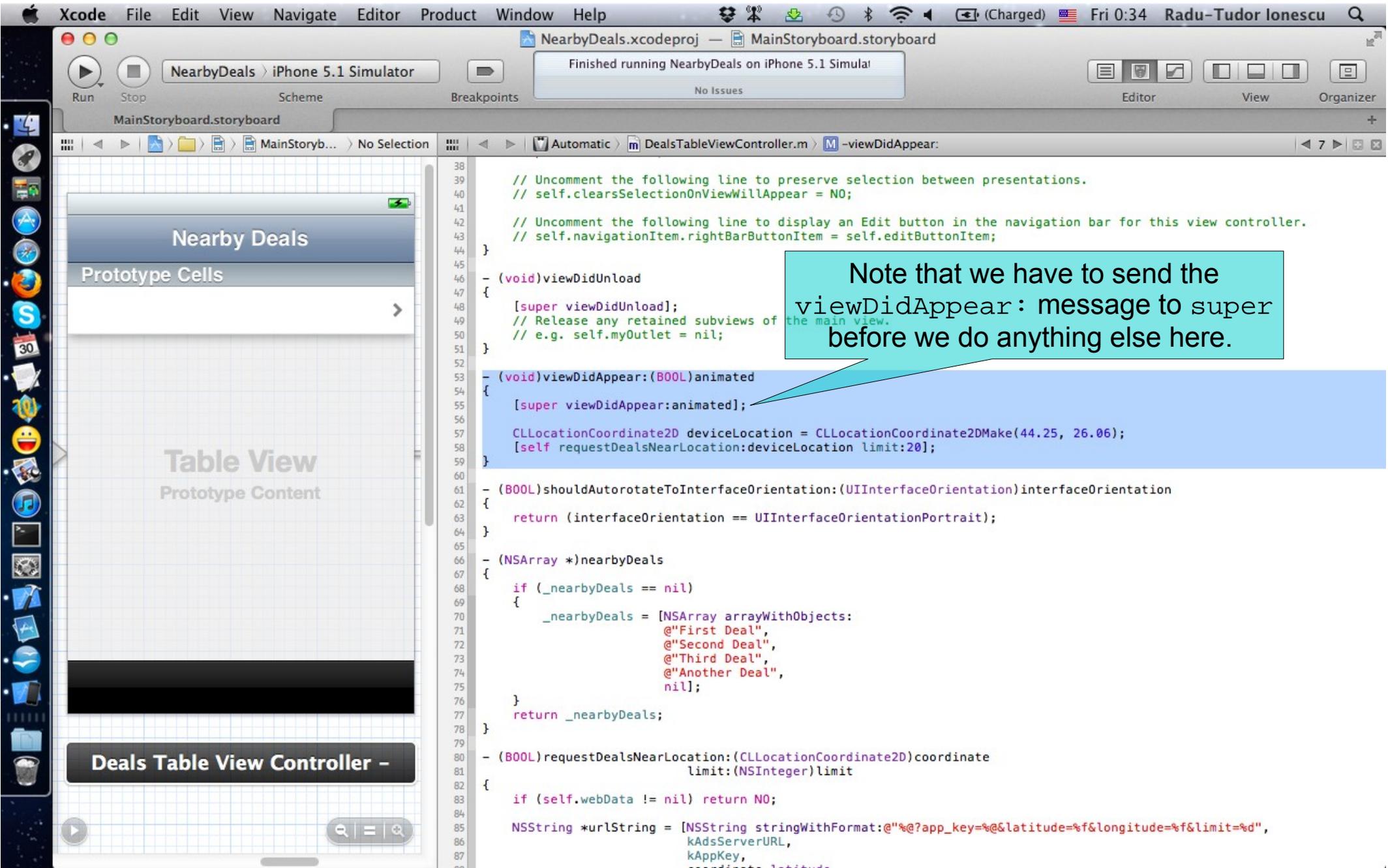
Task: Create a request using the GeoAds+ API that will return deals from nearby Bars or Restaurants.

22. We want to make this request right when our Table View appears on screen. Note the each View Controller has a lifecycle that starts with its creation. We will discuss this later in detail, but for now you should know that you (the Controller) get notified when your View appears on screen.

If we want to do something custom (such as making a request to a server) when it appears on screen we should implement the `viewDidAppear:` method.

We are going to make a request from a fake location that we create using the `CLLocationCoordinate2DMake` helper C function. Later we will use the GPS location of the device.

The next screenshot shows how to implement this callback method.



Note that we have to send the viewDidLoad:animated: message to super before we do anything else here.

```
38
39 // Uncomment the following line to preserve selection between presentations.
40 // self.clearsSelectionOnViewWillAppear = NO;
41
42 // Uncomment the following line to display an Edit button in the navigation bar for this view controller.
43 // self.navigationItem.rightBarButtonItem = self.editButtonItem;
44
45 }
46 - (void)viewDidLoad
47 {
48     [super viewDidLoad];
49     // Release any retained subviews of the main view.
50     // e.g. self.myOutlet = nil;
51 }
52
53 - (void)viewDidLoad:(BOOL)animated
54 {
55     [super viewDidLoad:animated];
56
57     CLLocationCoordinate2D deviceLocation = CLLocationCoordinate2DMake(44.25, 26.06);
58     [self requestDealsNearLocation:deviceLocation limit:20];
59 }
60
61 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
62 {
63     return (interfaceOrientation == UIInterfaceOrientationPortrait);
64 }
65
66 - (NSArray *)nearbyDeals
67 {
68     if (_nearbyDeals == nil)
69     {
70         _nearbyDeals = [NSArray arrayWithObjects:
71             @"First Deal",
72             @"Second Deal",
73             @"Third Deal",
74             @"Another Deal",
75             nil];
76     }
77     return _nearbyDeals;
78 }
79
80 - (BOOL)requestDealsNearLocation:(CLLocationCoordinate2D)coordinate
81     limit:(NSInteger)limit
82 {
83     if (self.webData != nil) return NO;
84
85     NSString *urlString = [NSString stringWithFormat:@"%@?app_key=%@&latitude=%f&longitude=%f&limit=%d",
86         kAdsServerURL,
87         kAppKey,
88         coordinate.latitude
89     ];
90 }
```

## Task 2

Task: Create a request using the GeoAds+ API that will return deals from nearby Bars or Restaurants..

23. Run the application in iOS Simulator and check out the console to see if it gets the XML from the server.

24. Stop running the application.

25. Disconnect you computer from the Internet (just pull out the cable) and lets see what happens with our request.

Run the application again. This time it should display the error message.

26. Stop running the application.

27. Don't forget to put back your Internet cable.

## Task 3

Task: Add a new class to your project with helper methods for XML parsing.

1. Usually we use the `NSXMLParser` class to parse XML documents. This class parses XML files in an event-driven manner as it notifies its delegate about the items (elements, attributes) that it encounters as it processes an XML document. This is very efficient when we want to parse large XML files, but in our case we need something simple. Thus, we are going to parse the XML file ourselves and build an `NSArray` with the nearby deals.

Actually, we are going to add `SimpleXMLParser.h/m` to our project. This class already contains some helper class methods for parsing XML documents.

Open Project Navigator.

2. Copy and paste `SimpleXMLParser.h` and `SimpleXMLParser.m` to the `NearbyDeals` folder within your project folder. Remember that you can right-click on the `NearbyDeals` group in Project Navigator and select “Show in Finder”.

## Task 3

Task: Add a new class to your project with helper methods for XML parsing.

3. Right-click on the NearbyDeals group in Project Navigator and select “Add files to NearbyDeals...”.
4. Select the SimpleXMLParser.h and SimpleXMLParser.m from the NearbyDeals folder and click Add.
5. We should add some functionality to the `SimpleXMLParser` class in order to parse the XML with nearby deals. It's good if we do this in another tab. Go to “File > New > Tab” in Xcode menu or use the `CMD+T` shortcut keys to create the new tab.
6. Open `SimpleXMLParser.m` on the left-side of the Editor, and `SimpleXMLParser.h` on the right-side. The header file should automatically appear on the right-side if you are on automatic mode.
7. Take a look at the implementation of the following methods:  
`convertSpecialCharactersToUnicodeInXML:`  
`contentOfFirstTagWithName:fromXML:`

## Task 3

Task: Add a new class to your project with helper methods for XML parsing.

8. Let's declare and implement another class method in the `SimpleXMLParser` that will parse the XML document with nearby deals received from the server.

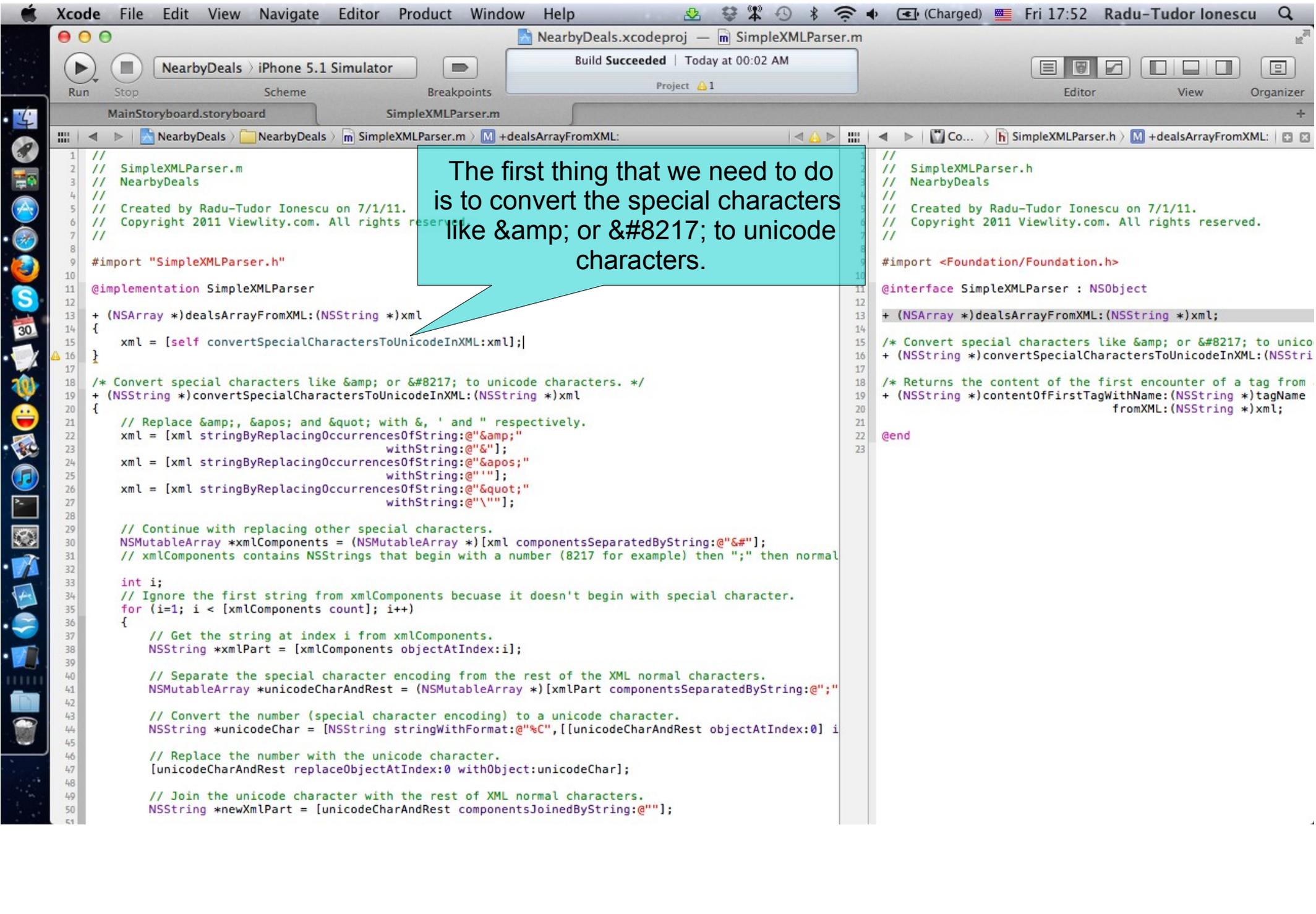
It will build an array of deals. Each deal will be represented by an `NSDictionary` that will store the deal's title, subtitle, description, etc.

Follow the steps from the next slides to implement this method.

```
Xcode File Edit View Navigate Editor Product Window Help
NearbyDeals.xcodeproj — SimpleXMLParser.m
Build Succeeded | Today at 00:02 AM
Project 1
NearbyDeals > iPhone 5.1 Simulator
MainStoryboard.storyboard SimpleXMLParser.m
NearbyDeals > NearbyDeals > SimpleXMLParser.m > +contentOfFirstTagWithName:fromXML:
SimpleXMLParser.h > +dealArrayFromXML:

1 //
2 // SimpleXMLParser.m
3 // NearbyDeals
4 //
5 // Created by Radu-Tudor Ionescu on 7/1/11.
6 // Copyright 2011 Viewlity.com. All rights reserved.
7 //
8
9 #import "SimpleXMLParser.h"
10
11 @implementation SimpleXMLParser
12
13 /* Convert special characters like & or &#8217; to unicode characters. */
14 + (NSString *)convertSpecialCharactersToUnicodeInXML:(NSString *)xml
15 {
16     // Replace & , &apos; and &quot; with & , ' and " respectively.
17     xml = [xml stringByReplacingOccurrencesOfString:@"&"
18         withString:@"&"];
19     xml = [xml stringByReplacingOccurrencesOfString:@"&apos;"
20         withString:@"'"];
21     xml = [xml stringByReplacingOccurrencesOfString:@"&quot;"
22         withString:@""];
23
24     // Continue with replacing other special characters.
25     NSMutableArray *xmlComponents = (NSMutableArray *)xml.componentsSeparatedByString:@" ";
26     // xmlComponents contains NSStrings that begin with a number (8217 for example) then ";" then normal
27
28     int i;
29     // Ignore the first string from xmlComponents because it doesn't begin with special character.
30     for (i=1; i < [xmlComponents count]; i++)
31     {
32         // Get the string at index i from xmlComponents.
33         NSString *xmlPart = [xmlComponents objectAtIndex:i];
34
35         // Separate the special character encoding from the rest of the XML normal characters.
36         NSMutableArray *unicodeCharAndRest = (NSMutableArray *)xmlPart.componentsSeparatedByString:@" ";
37
38         // Convert the number (special character encoding) to a unicode character.
39         NSString *unicodeChar = [NSString stringWithFormat:@"%C", [[unicodeCharAndRest objectAtIndex:0] intValue]];
40
41         // Replace the number with the unicode character.
42         [unicodeCharAndRest replaceObjectAtIndex:0 withObject:unicodeChar];
43
44         // Join the unicode character with the rest of XML normal characters.
45         NSString *newXmlPart = [unicodeCharAndRest componentsJoinedByString:@" "];
46
47         // Replace the old xmlPart with the new one that begins with a unicode character instead of spec
48         [xmlComponents replaceObjectAtIndex:i withObject:newXmlPart];
49     }
50
51     // Join the xmlComponents into an XML string that has only unicode characters.
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573

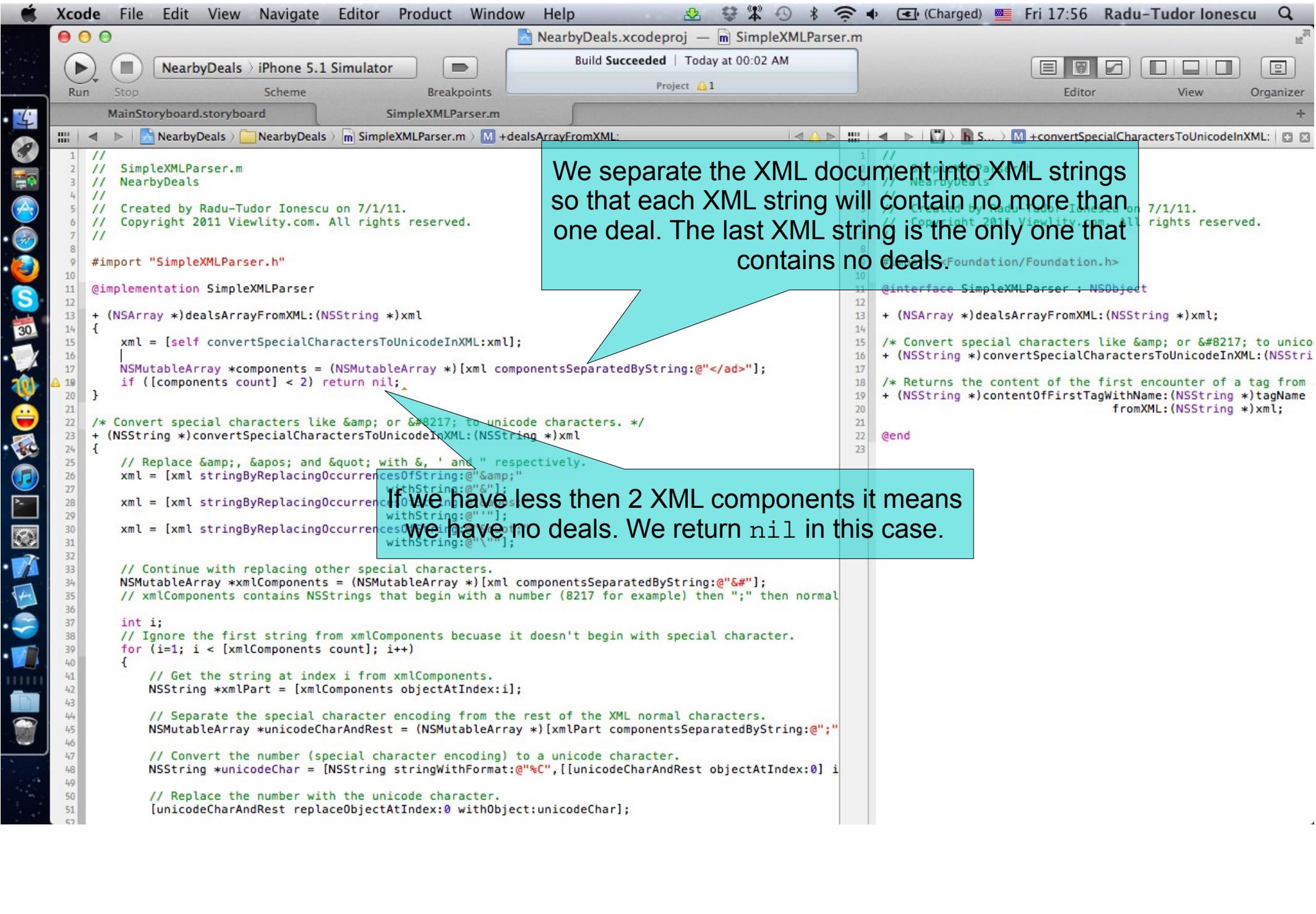
```



The first thing that we need to do is to convert the special characters like & or &#8217; to unicode characters.

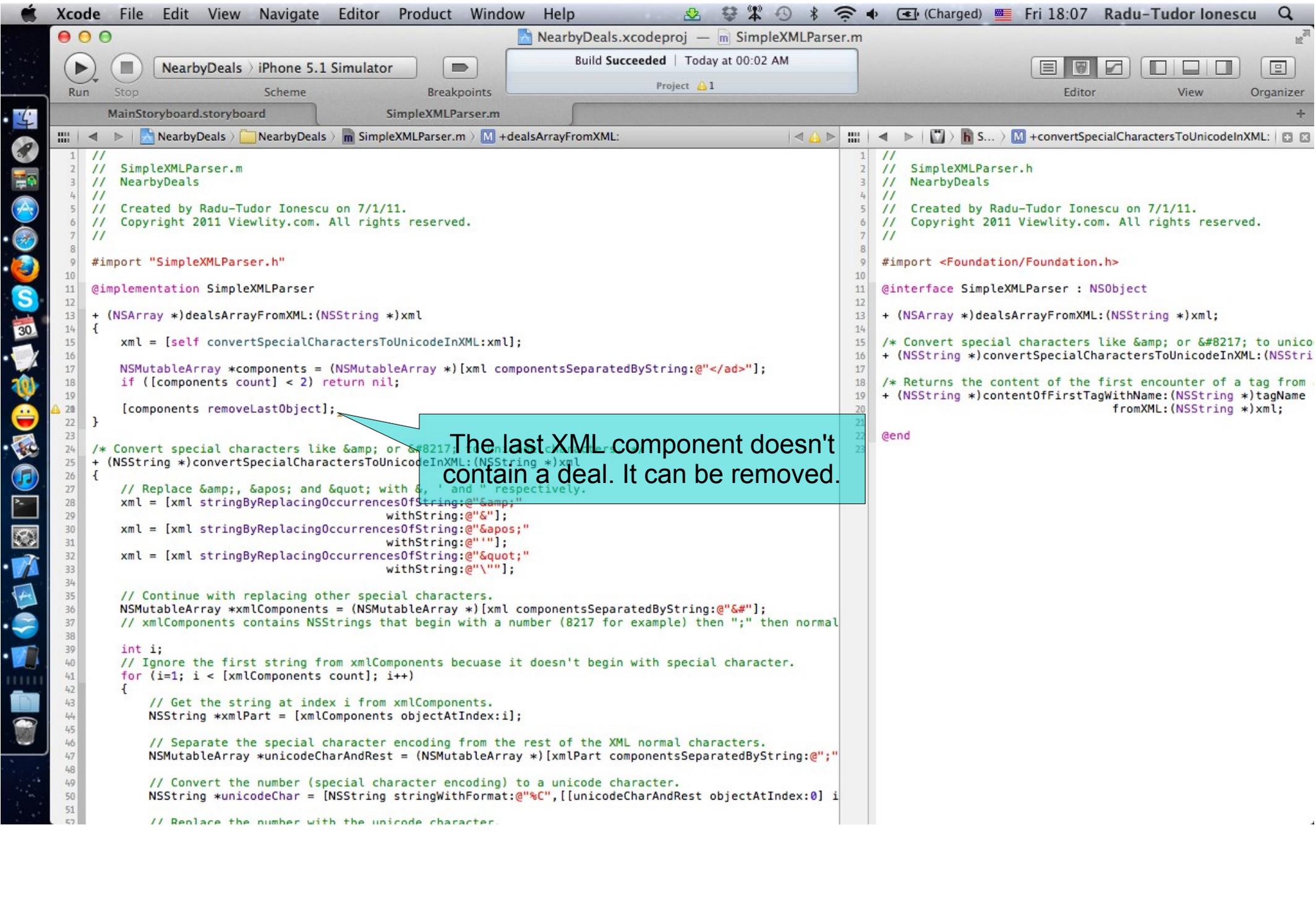
```
1 //
2 // SimpleXMLParser.m
3 // NearbyDeals
4 //
5 // Created by Radu-Tudor Ionescu on 7/1/11.
6 // Copyright 2011 Viewlity.com. All rights reserved.
7 //
8
9 #import "SimpleXMLParser.h"
10
11 @implementation SimpleXMLParser
12
13 + (NSArray *)dealsArrayFromXML:(NSString *)xml
14 {
15     xml = [self convertSpecialCharactersToUnicodeInXML:xml];
16 }
17
18 /* Convert special characters like & or &#8217; to unicode characters. */
19 + (NSString *)convertSpecialCharactersToUnicodeInXML:(NSString *)xml
20 {
21     // Replace &, &apos; and &quot; with &, ' and " respectively.
22     xml = [xml stringByReplacingOccurrencesOfString:@"&"
23         withString:@"&"];
24     xml = [xml stringByReplacingOccurrencesOfString:@"&apos;"
25         withString:@"'"];
26     xml = [xml stringByReplacingOccurrencesOfString:@"&quot;"
27         withString:@"\""];
28
29     // Continue with replacing other special characters.
30     NSMutableArray *xmlComponents = (NSMutableArray *)[xml componentsSeparatedByString:@"&#"];
31     // xmlComponents contains NSStrings that begin with a number (8217 for example) then ";" then normal
32
33     int i;
34     // Ignore the first string from xmlComponents because it doesn't begin with special character.
35     for (i=1; i < [xmlComponents count]; i++)
36     {
37         // Get the string at index i from xmlComponents.
38         NSString *xmlPart = [xmlComponents objectAtIndex:i];
39
40         // Separate the special character encoding from the rest of the XML normal characters.
41         NSMutableArray *unicodeCharAndRest = (NSMutableArray *)[xmlPart componentsSeparatedByString:@";"];
42
43         // Convert the number (special character encoding) to a unicode character.
44         NSString *unicodeChar = [NSString stringWithFormat:@"%C", [[unicodeCharAndRest objectAtIndex:0] intValue]];
45
46         // Replace the number with the unicode character.
47         [unicodeCharAndRest replaceObjectAtIndex:0 withObject:unicodeChar];
48
49         // Join the unicode character with the rest of XML normal characters.
50         NSString *newXmlPart = [unicodeCharAndRest componentsJoinedByString:@""];
51     }
52 }
```

```
1 //
2 // SimpleXMLParser.h
3 // NearbyDeals
4 //
5 // Created by Radu-Tudor Ionescu on 7/1/11.
6 // Copyright 2011 Viewlity.com. All rights reserved.
7 //
8
9 #import <Foundation/Foundation.h>
10
11 @interface SimpleXMLParser : NSObject
12
13 + (NSArray *)dealsArrayFromXML:(NSString *)xml;
14
15 /* Convert special characters like & or &#8217; to unicode characters. */
16 + (NSString *)convertSpecialCharactersToUnicodeInXML:(NSString *)xml;
17
18 /* Returns the content of the first encounter of a tag from the XML. */
19 + (NSString *)contentOfFirstTagWithName:(NSString *)tagName
20     fromXML:(NSString *)xml;
21
22 @end
23
```



We separate the XML document into XML strings so that each XML string will contain no more than one deal. The last XML string is the only one that contains no deals.

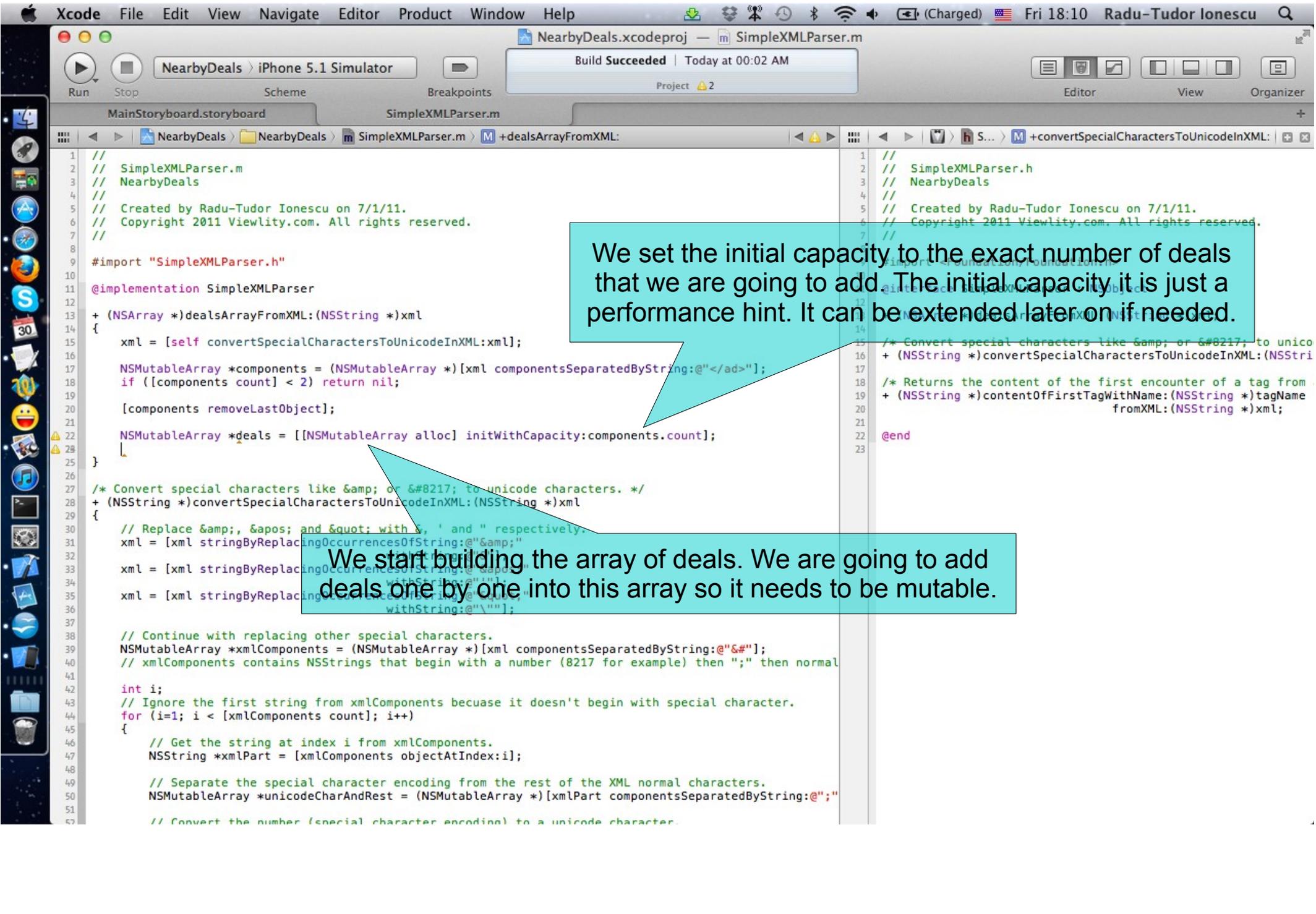
If we have less then 2 XML components it means we have no deals. We return nil in this case.



```
1 //
2 // SimpleXMLParser.m
3 // NearbyDeals
4 //
5 // Created by Radu-Tudor Ionescu on 7/1/11.
6 // Copyright 2011 Viewlity.com. All rights reserved.
7 //
8
9 #import "SimpleXMLParser.h"
10
11 @implementation SimpleXMLParser
12
13 + (NSArray *)dealsArrayFromXML:(NSString *)xml
14 {
15     xml = [self convertSpecialCharactersToUnicodeInXML:xml];
16
17     NSMutableArray *components = (NSMutableArray *)[xml componentsSeparatedByString:@"</ad>"];
18     if ([components count] < 2) return nil;
19
20     [components removeLastObject];
21
22 }
23
24 /* Convert special characters like &amp; or &#8217; to unicode characters.
25 + (NSString *)convertSpecialCharactersToUnicodeInXML:(NSString *)xml
26 {
27     // Replace &amp;, &apos; and &quot; with &, ' and " respectively.
28     xml = [xml stringByReplacingOccurrencesOfString:@"&amp;"
29         withString:@"&"];
30     xml = [xml stringByReplacingOccurrencesOfString:@"&apos;"
31         withString:@"'"];
32     xml = [xml stringByReplacingOccurrencesOfString:@"&quot;"
33         withString:@"\""];
34
35     // Continue with replacing other special characters.
36     NSMutableArray *xmlComponents = (NSMutableArray *)[xml componentsSeparatedByString:@"&#"];
37     // xmlComponents contains NSStrings that begin with a number (8217 for example) then ";" then normal
38
39     int i;
40     // Ignore the first string from xmlComponents because it doesn't begin with special character.
41     for (i=1; i < [xmlComponents count]; i++)
42     {
43         // Get the string at index i from xmlComponents.
44         NSString *xmlPart = [xmlComponents objectAtIndex:i];
45
46         // Separate the special character encoding from the rest of the XML normal characters.
47         NSMutableArray *unicodeCharAndRest = (NSMutableArray *)[xmlPart componentsSeparatedByString:@";"];
48
49         // Convert the number (special character encoding) to a unicode character.
50         NSString *unicodeChar = [NSString stringWithFormat:@"%C", [[unicodeCharAndRest objectAtIndex:0] i
51
52         // Replace the number with the unicode character.
```

The last XML component doesn't contain a deal. It can be removed.

```
1 //
2 // SimpleXMLParser.h
3 // NearbyDeals
4 //
5 // Created by Radu-Tudor Ionescu on 7/1/11.
6 // Copyright 2011 Viewlity.com. All rights reserved.
7 //
8
9 #import <Foundation/Foundation.h>
10
11 @interface SimpleXMLParser : NSObject
12
13 + (NSArray *)dealsArrayFromXML:(NSString *)xml;
14
15 /* Convert special characters like &amp; or &#8217; to unicode
16 + (NSString *)convertSpecialCharactersToUnicodeInXML:(NSString *)xml;
17
18 /* Returns the content of the first encounter of a tag from
19 + (NSString *)contentOfFirstTagWithName:(NSString *)tagName
20     fromXML:(NSString *)xml;
21
22 @end
```



We set the initial capacity to the exact number of deals that we are going to add. The initial capacity it is just a performance hint. It can be extended later on if needed.

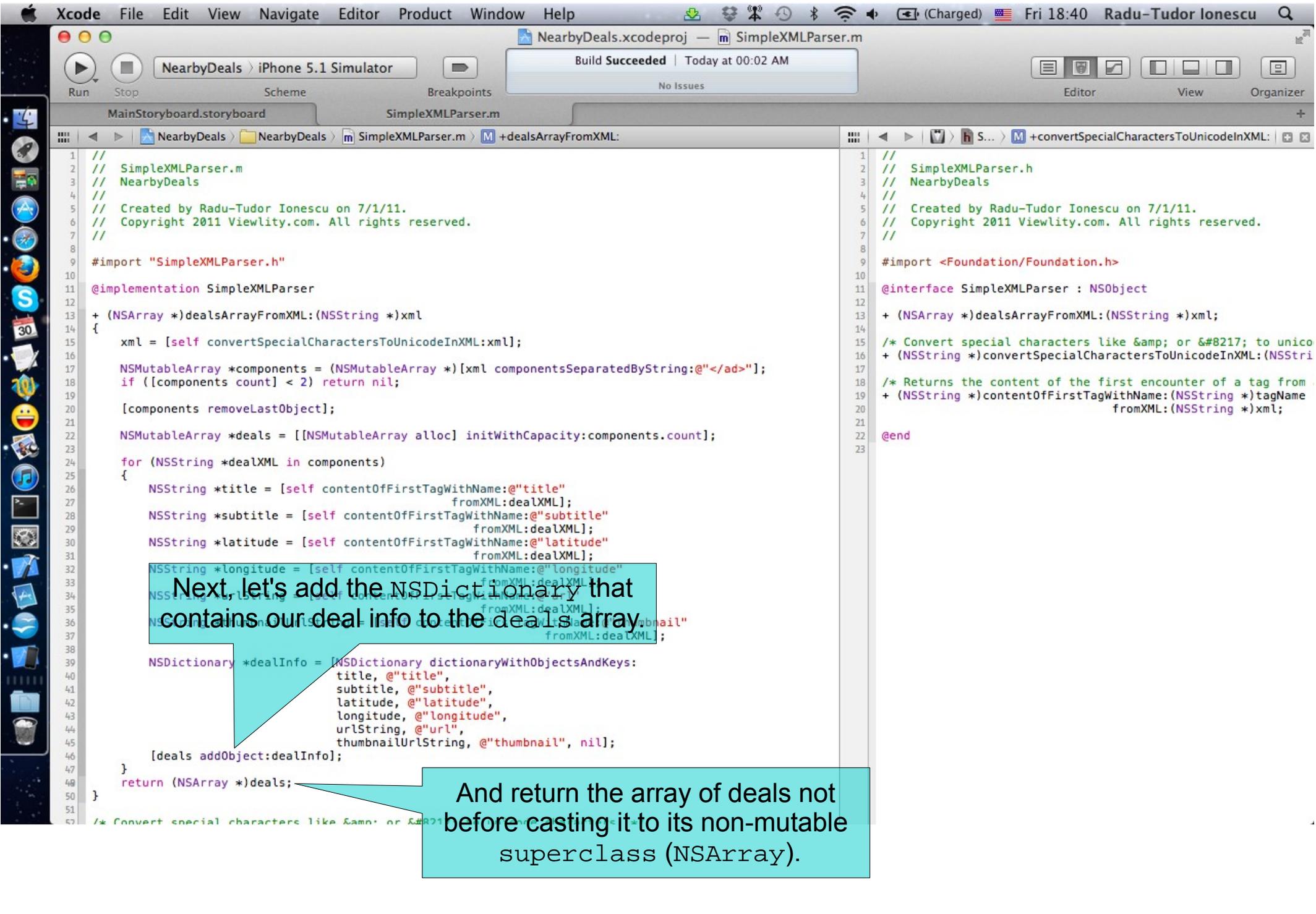
We start building the array of deals. We are going to add deals one by one into this array so it needs to be mutable.

```
Xcode File Edit View Navigate Editor Product Window Help
NearbyDeals.xcodeproj — SimpleXMLParser.m
Build Succeeded | Today at 00:02 AM
Project 8
NearbyDeals > iPhone 5.1 Simulator
MainStoryboard.storyboard SimpleXMLParser.m
NearbyDeals > NearbyDeals > SimpleXMLParser.m > +dealsArrayFromXML:
1 //
2 // SimpleXMLParser.m
3 // NearbyDeals
4 //
5 // Created by Radu-Tudor Ionescu on 7/1/11.
6 // Copyright 2011 Viewlity.com. All rights reserved.
7 //
8
9 #import "SimpleXMLParser.h"
10
11 @implementation SimpleXMLParser
12
13 + (NSArray *)dealsArrayFromXML:(NSString *)xml
14 {
15     xml = [self convertSpecialCharactersToUnicodeInXML:xml];
16
17     NSMutableArray *components = (NSMutableArray *)xml.componentsSeparatedByString:@"<deal>";
18     if ([components count] < 2) return nil;
19
20     [components removeLastObject];
21
22     NSMutableArray *deals = [[NSMutableArray alloc] initWithCapacity:components.count];
23
24     for (NSString *dealXML in components)
25     {
26         NSString *title = [self contentOfFirstTagWithName:@"title"
27                             fromXML:dealXML];
28         NSString *subtitle = [self contentOfFirstTagWithName:@"subtitle"
29                               fromXML:dealXML];
30         NSString *latitude = [self contentOfFirstTagWithName:@"latitude"
31                               fromXML:dealXML];
32         NSString *longitude = [self contentOfFirstTagWithName:@"longitude"
33                                fromXML:dealXML];
34         NSString *urlString = [self contentOfFirstTagWithName:@"url"
35                                fromXML:dealXML];
36         NSString *thumbnailUrlString = [self contentOfFirstTagWithName:@"thumbnail"
37                                         fromXML:dealXML];
38     }
39 }
40
41
42
43 /* Convert special characters like &amp; or &#8217; to unicode characters. */
44 + (NSString *)convertSpecialCharactersToUnicodeInXML:(NSString *)xml
45 {
46     // Replace &amp;, &apos; and &quot; with &, ' and " respectively.
47     xml = [xml stringByReplacingOccurrencesOfString:@"&amp;"
48                                                  withString:@"&"];
49     xml = [xml stringByReplacingOccurrencesOfString:@"&apos;"
50                                                  withString:@"'"];
51     xml = [xml stringByReplacingOccurrencesOfString:@"&quot;"
52                                                  withString:@"\""];
53 }
54
55
56
57
58 //
59 // SimpleXMLParser.h
60 // NearbyDeals
61 //
62 // Created by Radu-Tudor Ionescu on 7/1/11.
63 // Copyright 2011 Viewlity.com. All rights reserved.
64 //
65 //
66 #import <Foundation/Foundation.h>
67
68 @interface SimpleXMLParser : NSObject
69
70 + (NSArray *)dealsArrayFromXML:(NSString *)xml;
71
72 /* Convert special characters like &amp; or &#8217; to unicode characters. */
73 + (NSString *)convertSpecialCharactersToUnicodeInXML:(NSString *)xml;
74
75 /* Returns the content of the first encounter of a tag from the XML. */
76 + (NSString *)contentOfFirstTagWithName:(NSString *)tagName
77     fromXML:(NSString *)xml;
78
79 @end
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

Let's extract the deal data from each XML component. We do this using a for-in structure. Note that we convert the objects from the components array to NSString because we know they are NSString objects.

We use the contentOfFirstTagWithName: helper method to extract the content of the tags we are interested in. We want to configure our Table View Cells using the title, subtitle and the thumbnail photo. We are going to use the latitude and longitude to present the deals on the map later. We display the deal details in a UIWebView that opens the deal URL. This is all the information we need.

```
Xcode File Edit View Navigate Editor Product Window Help
NearbyDeals.xcodeproj — SimpleXMLParser.m
Build Succeeded | Today at 00:02 AM
Project 3
NearbyDeals > iPhone 5.1 Simulator
MainStoryboard.storyboard SimpleXMLParser.m
NearbyDeals > NearbyDeals > SimpleXMLParser.m > +dealsArrayFromXML:
1 //
2 // SimpleXMLParser.m
3 // NearbyDeals
4 //
5 // Created by Radu-Tudor Ionescu on 7/1/11.
6 // Copyright 2011 Viewlity.com. All rights reserved.
7 //
8
9 #import "SimpleXMLParser.h"
10
11 @implementation SimpleXMLParser
12
13 + (NSArray *)dealsArrayFromXML:(NSString *)xml
14 {
15     xml = [self convertSpecialCharactersToUnicodeInXML:xml];
16
17     NSMutableArray *components = (NSMutableArray *)[xml componentsSeparatedByString:@"</ad>"];
18     if ([components count] < 2) return nil;
19
20     [components removeLastObject];
21
22     NSMutableArray *deals = [[NSMutableArray alloc] initWithCapacity:components.count];
23
24     for (NSString *dealXML in components)
25     {
26         NSString *title = [self contentOfFirstTagWithName:@"title"
27                             fromXML:dealXML];
28         NSString *subtitle = [self contentOfFirstTagWithName:@"subtitle"
29                               fromXML:dealXML];
30         NSString *latitude = [self contentOfFirstTagWithName:@"latitude"
31                               fromXML:dealXML];
32         NSString *longitude = [self contentOfFirstTagWithName:@"longitude"
33                                fromXML:dealXML];
34         NSString *urlString = [self contentOfFirstTagWithName:@"url"
35                                fromXML:dealXML];
36         NSString *thumbnailUrlString = [self contentOfFirstTagWithName:@"thumbnail"
37                                         fromXML:dealXML];
38
39         NSDictionary *dealInfo = [NSDictionary dictionaryWithObjectsAndKeys:
40                                 title, @"title",
41                                 subtitle, @"subtitle",
42                                 latitude, @"latitude",
43                                 longitude, @"longitude",
44                                 urlString, @"url",
45                                 thumbnailUrlString, @"thumbnail", nil];
46
47     }
48 }
49
50 /* Convert special characters like & or &#8217; to unicode characters. */
51 + (NSString *)convertSpecialCharactersToUnicodeInXML:(NSString *)xml
52 {
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2
```



```
1 //  
2 // SimpleXMLParser.m  
3 // NearbyDeals  
4 //  
5 // Created by Radu-Tudor Ionescu on 7/1/11.  
6 // Copyright 2011 Viewlity.com. All rights reserved.  
7 //  
8  
9 #import "SimpleXMLParser.h"  
10  
11 @implementation SimpleXMLParser  
12  
13 + (NSArray *)dealFromArrayFromXML:(NSString *)xml  
14 {  
15     xml = [self convertSpecialCharactersToUnicodeInXML:xml];  
16  
17     NSMutableArray *components = (NSMutableArray *) [xml componentsSeparatedByString:@"</ad>"];  
18     if ([components count] < 2) return nil;  
19  
20     [components removeLastObject];  
21  
22     NSMutableArray *deals = [[NSMutableArray alloc] initWithCapacity:components.count];  
23  
24     for (NSString *dealXML in components)  
25     {  
26         NSString *title = [self contentOfFirstTagWithName:@"title"  
27                             fromXML:dealXML];  
28         NSString *subtitle = [self contentOfFirstTagWithName:@"subtitle"  
29                               fromXML:dealXML];  
30         NSString *latitude = [self contentOfFirstTagWithName:@"latitude"  
31                               fromXML:dealXML];  
32         NSString *longitude = [self contentOfFirstTagWithName:@"longitude"  
33                               fromXML:dealXML];  
34         NSString *urlString = [self contentOfFirstTagWithName:@"url"  
35                               fromXML:dealXML];  
36         NSString *thumbnailUrlString = [self contentOfFirstTagWithName:@"thumbnail"  
37                                         fromXML:dealXML];  
38  
39         NSDictionary *dealInfo = [NSDictionary dictionaryWithObjectsAndKeys:  
40             title, @"title",  
41             subtitle, @"subtitle",  
42             latitude, @"latitude",  
43             longitude, @"longitude",  
44             urlString, @"url",  
45             thumbnailUrlString, @"thumbnail", nil];  
46  
47         [deals addObject:dealInfo];  
48     }  
49     return (NSArray *)deals;  
50 }  
51  
52 /* Convert special characters like &amp; or &#8217;
```

```
1 //  
2 // SimpleXMLParser.h  
3 // NearbyDeals  
4 //  
5 // Created by Radu-Tudor Ionescu on 7/1/11.  
6 // Copyright 2011 Viewlity.com. All rights reserved.  
7 //  
8  
9 #import <Foundation/Foundation.h>  
10  
11 @interface SimpleXMLParser : NSObject  
12  
13 + (NSArray *)dealFromArrayFromXML:(NSString *)xml;  
14  
15 /* Convert special characters like &amp; or &#8217; to unic  
16 + (NSString *)convertSpecialCharactersToUnicodeInXML:(NSString *)xml;  
17  
18 /* Returns the content of the first encounter of a tag from  
19 + (NSString *)contentOfFirstTagWithName:(NSString *)tagName  
20                               fromXML:(NSString *)xml;  
21  
22 @end  
23
```

Next, let's add the NSDictionary that contains our deal info to the deals array.

And return the array of deals not before casting it to its non-mutable superclass (NSArray).

## Task 4

Task: Use the `SimpleXMLParser` class methods to parse the XML with nearby deals and display them in your Table View.

1. Switch to the `MainStoryboard.storyboard` tab in Xcode.
2. Open `DealsTableViewController.h` in Assistant Editor.
3. We want to use the `SimpleXMLParser` `dealsArrayFromXML:` class method to obtain an `NSArray` with the nearby deals from the XML file we received from the server.

The first thing to do is to `#import` the `SimpleXMLParser` header file into our Table View Controller so that we can use its methods.

The next screenshot show you how to `#import` this header file.

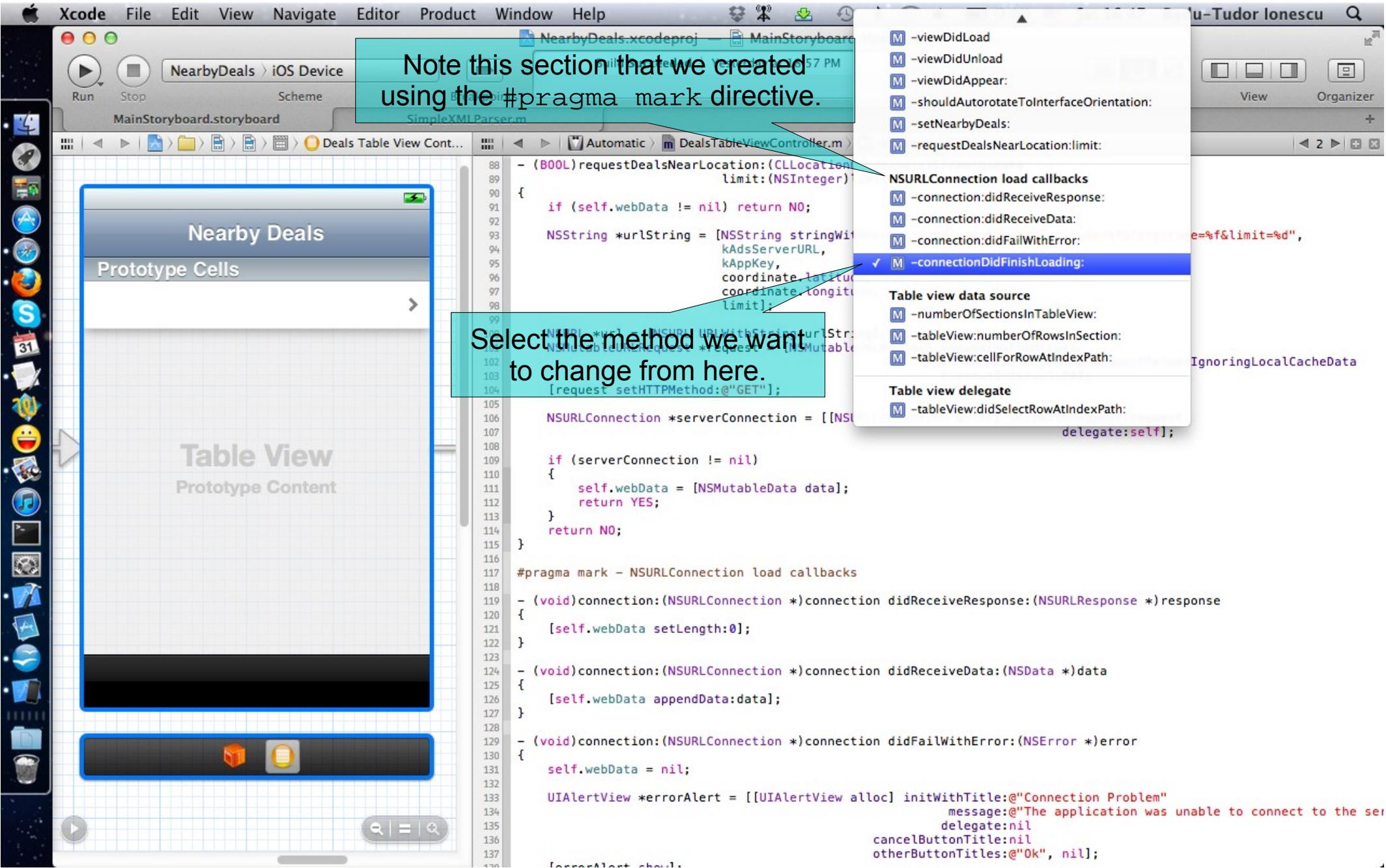


## Task 4

Task: Use the `SimpleXMLParser` class methods to parse the XML with nearby deals and display them in your Table View.

4. Open `DealsTableViewController.m` in Assistant Editor.
5. Scroll to the `connectionDidFinishLoading:` method we implemented earlier. Comment the `NSLog` that prints the XML document to the console.
6. Parse the `receivedXML` using the `dealsArrayFromXML:` class method and store the result into the `nearbyDeals` Model.

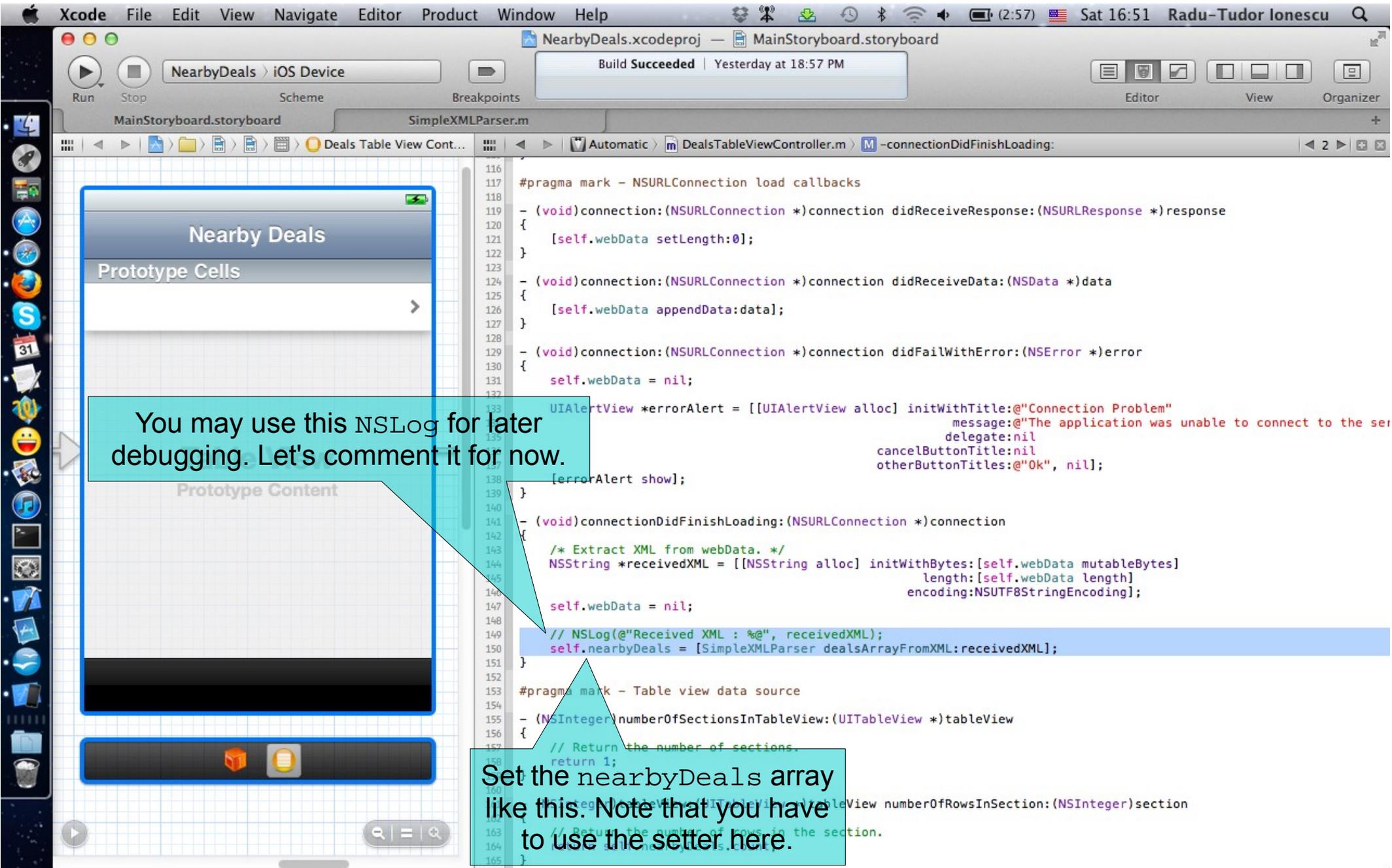
The next screenshots show you how to perform these steps.



Note this section that we created using the #pragma mark directive.

Select the method we want to change from here.

- viewDidLoad
- viewDidUnload
- viewDidAppear:
- shouldAutorotateToInterfaceOrientation:
- setNearbyDeals:
- requestDealsNearLocation:limit:
- NSURLConnection load callbacks**
- connection:didReceiveResponse:
- connection:didReceiveData:
- connection:didFailWithError:
- connectionDidFinishLoading:**
- Table view data source**
- numberOfSectionsInTableView:
- tableView:numberOfRowsInSection:
- tableView:cellForRowAtIndexPath:
- Table view delegate**
- tableView:didSelectRowAtIndexPath:



You may use this NSLog for later debugging. Let's comment it for now.

Set the nearbyDeals array like this. Note that you have to use the setter here.

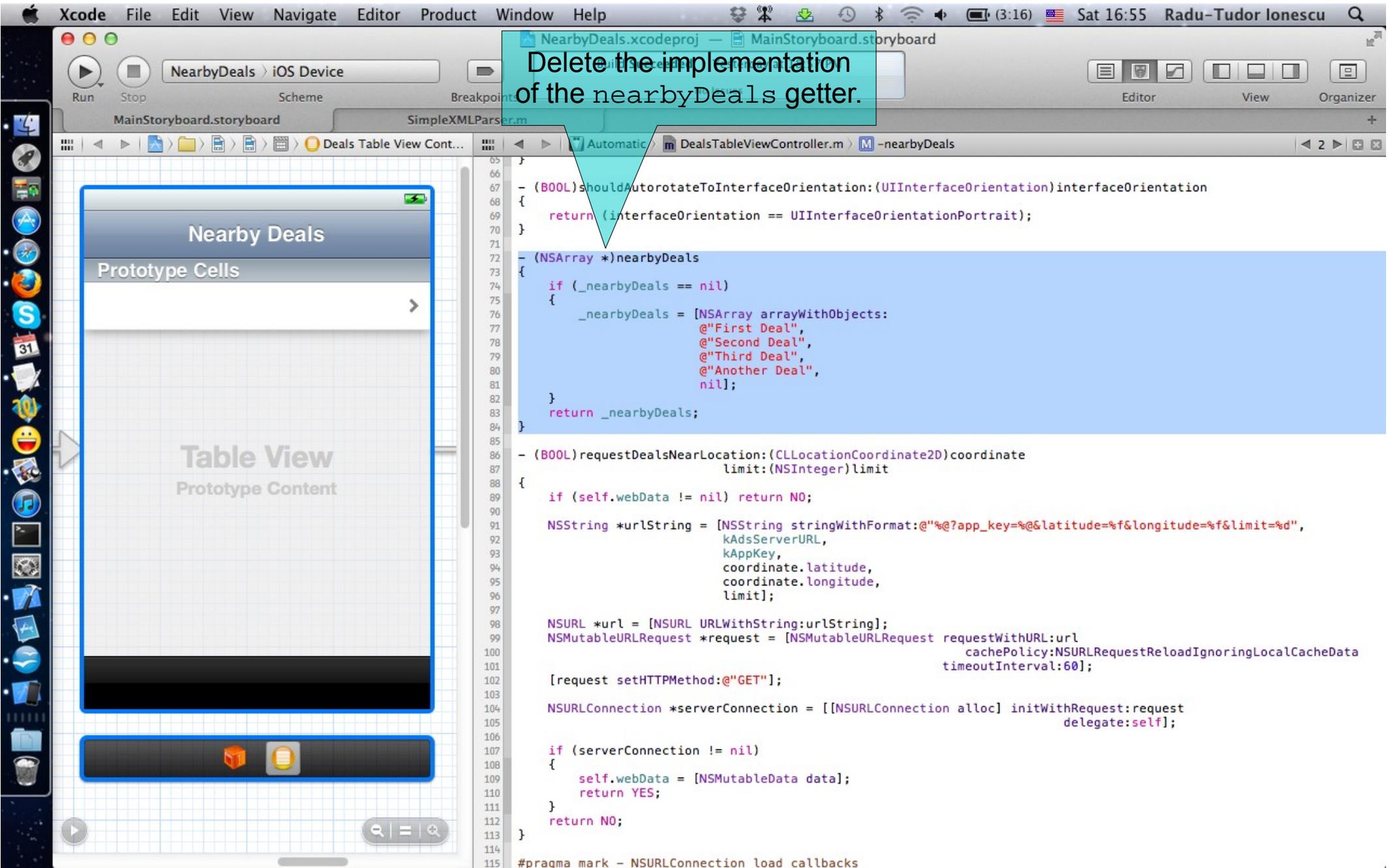
```
116
117 #pragma mark - NSURLConnection load callbacks
118
119 - (void)connection:(NSURLConnection *)connection didReceiveResponse:(NSURLResponse *)response
120 {
121     [self.webData setLength:0];
122 }
123
124 - (void)connection:(NSURLConnection *)connection didReceiveData:(NSData *)data
125 {
126     [self.webData appendData:data];
127 }
128
129 - (void)connection:(NSURLConnection *)connection didFailWithError:(NSError *)error
130 {
131     self.webData = nil;
132 }
133
134 UIAlertView *errorAlert = [[UIAlertView alloc] initWithTitle:@"Connection Problem"
135     message:@"The application was unable to connect to the server"
136     delegate:nil
137     cancelButtonTitle:nil
138     otherButtonTitles:@"Ok", nil];
139 [errorAlert show];
140 }
141
142 - (void)connectionDidFinishLoading:(NSURLConnection *)connection
143 {
144     /* Extract XML from webData. */
145     NSString *receivedXML = [[NSString alloc] initWithBytes:[self.webData mutableBytes]
146         length:[self.webData length]
147         encoding:NSUTF8StringEncoding];
148     self.webData = nil;
149     // NSLog(@"Received XML : %@", receivedXML);
150     self.nearbyDeals = [SimpleXMLParser dealsFromArrayFromXML:receivedXML];
151 }
152
153 #pragma mark - Table view data source
154
155 - (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
156 {
157     // Return the number of sections.
158     return 1;
159 }
160
161 - (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
162 {
163     // Return the number of rows in the section.
164     return [self.nearbyDeals count];
165 }
```

## Task 4

Task: Use the `SimpleXMLParser` class methods to parse the XML with nearby deals and display them in your Table View.

7. We set the `nearbyDeals @property` when we receive the XML from the server. We no longer need the lazy instantiation mechanism for `nearbyDeals` because we instantiate it in the `connectionDidFinishLoading:` method. Thus, we can delete the getter implementation.

The next screenshot shows the code that needs to be deleted.



Delete the implementation of the nearbyDeals getter.

```
65 }
66
67 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
68 {
69     return (interfaceOrientation == UIInterfaceOrientationPortrait);
70 }
71
72 - (NSArray *)nearbyDeals
73 {
74     if (_nearbyDeals == nil)
75     {
76         _nearbyDeals = [NSArray arrayWithObjects:
77             @"First Deal",
78             @"Second Deal",
79             @"Third Deal",
80             @"Another Deal",
81             nil];
82     }
83     return _nearbyDeals;
84 }
85
86 - (BOOL)requestDealsNearLocation:(CLLocationCoordinate2D)coordinate
87     limit:(NSInteger)limit
88 {
89     if (self.webData != nil) return NO;
90
91     NSString *urlString = [NSString stringWithFormat:@"%s?app_key=%s&latitude=%f&longitude=%f&limit=%d",
92         kAdsServerURL,
93         kAppKey,
94         coordinate.latitude,
95         coordinate.longitude,
96         limit];
97
98     NSURL *url = [NSURL URLWithString:urlString];
99     NSMutableURLRequest *request = [NSMutableURLRequest requestWithURL:url
100         cachePolicy:NSURLRequestReloadIgnoringLocalCacheData
101         timeoutInterval:60];
102
103     [request setHTTPMethod:@"GET"];
104
105     NSURLConnection *serverConnection = [[NSURLConnection alloc] initWithRequest:request
106         delegate:self];
107
108     if (serverConnection != nil)
109     {
110         self.webData = [NSMutableData data];
111         return YES;
112     }
113     return NO;
114 }
115 #pragma mark - NSURLConnection load callbacks
```

## Task 4

Task: Use the `SimpleXMLParser` class methods to parse the XML with nearby deals and display them in your Table View.

8. We changed our Model in the `connectionDidFinishLoading:` method. We have to let the Table View know that we changed the data so that it has a chance to reload the new data. We do this by sending the `reloadData` message to the `tableView`.

Note that `tableView` is a `@property` inherited by our Table View Controller from `UITableViewController`. It is an outlet of our Table View.

The best place to message the Table View with `reloadData` is the setter of the `nearbyDeals`. If we always instantiate our Model through the setter, we make sure that the Table View always knows about this change as soon as possible.

Let's override the `nearbyDeals` setter with our own implementation and send the `reloadData` to the Table View there.

The next screenshot shows you how to do this.

Xcode File Edit View Navigate Editor Product Window Help

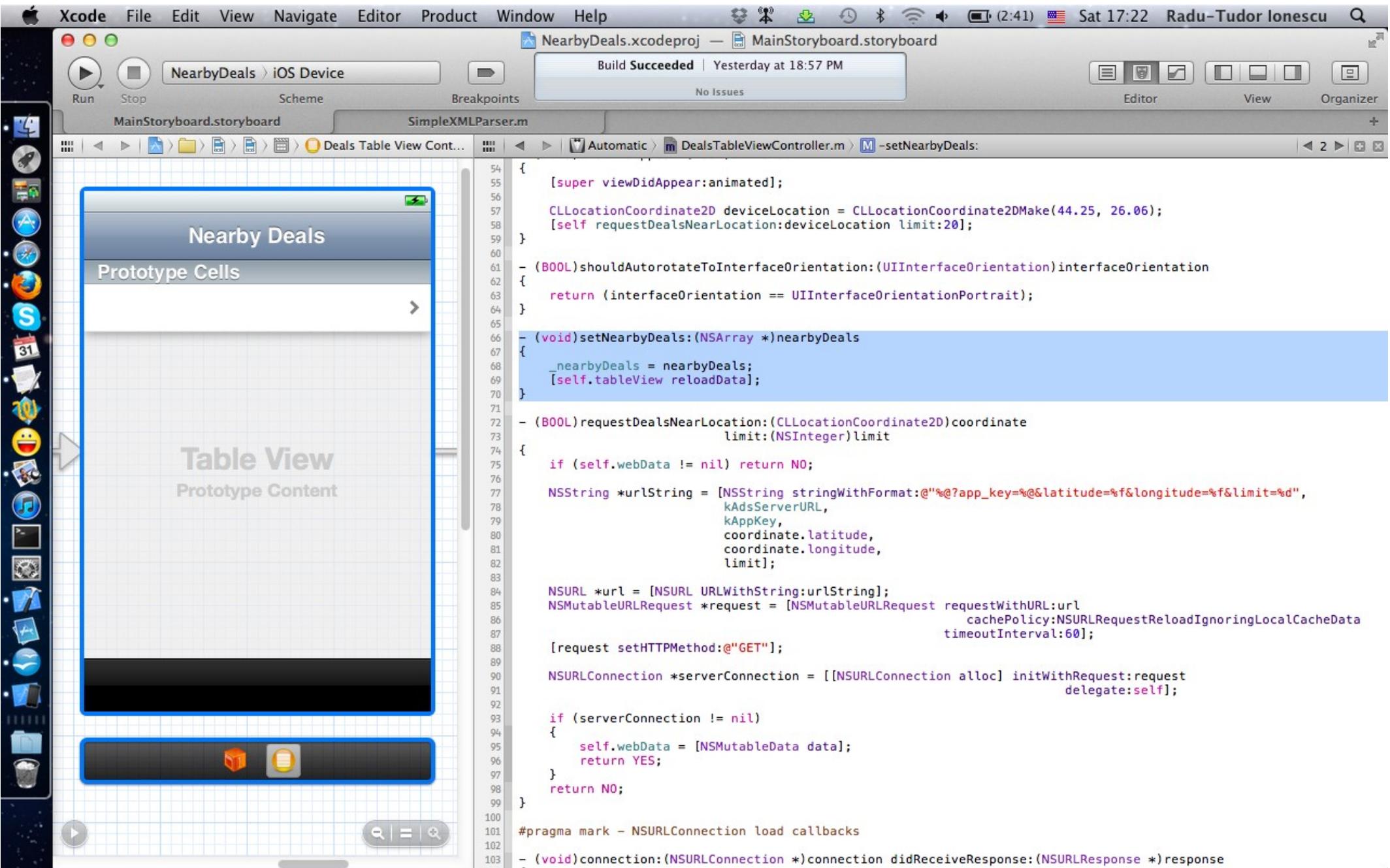
NearbyDeals.xcodeproj — MainStoryboard.storyboard

Build Succeeded | Yesterday at 18:57 PM  
No Issues

NearbyDeals > iOS Device

Run Stop Scheme Breakpoints Editor View Organizer

MainStoryboard.storyboard SimpleXMLParser.m Deals Table View Cont... Automatic DealsTableViewController.m -setNearbyDeals:



```
54 {
55     [super viewDidLoad];
56
57     CLLocationCoordinate2D deviceLocation = CLLocationCoordinate2DMake(44.25, 26.06);
58     [self requestDealsNearLocation:deviceLocation limit:20];
59 }
60
61 - (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
62 {
63     return (interfaceOrientation == UIInterfaceOrientationPortrait);
64 }
65
66 - (void)setNearbyDeals:(NSArray *)nearbyDeals
67 {
68     _nearbyDeals = nearbyDeals;
69     [self.tableView reloadData];
70 }
71
72 - (BOOL)requestDealsNearLocation:(CLLocationCoordinate2D)coordinate
73     limit:(NSInteger)limit
74 {
75     if (self.webData != nil) return NO;
76
77     NSString *urlString = [NSString stringWithFormat:@"%s?app_key=%s&latitude=%f&longitude=%f&limit=%d",
78         kAdsServerURL,
79         kAppKey,
80         coordinate.latitude,
81         coordinate.longitude,
82         limit];
83
84     NSURL *url = [NSURL URLWithString:urlString];
85     NSMutableURLRequest *request = [NSMutableURLRequest requestWithURL:url
86         cachePolicy:NSURLRequestReloadIgnoringLocalCacheData
87         timeoutInterval:60];
88
89     [request setHTTPMethod:@"GET"];
90
91     NSURLConnection *serverConnection = [[NSURLConnection alloc] initWithRequest:request
92         delegate:self];
93
94     if (serverConnection != nil)
95     {
96         self.webData = [NSMutableData data];
97         return YES;
98     }
99     return NO;
100 }
101 #pragma mark - NSURLConnection load callbacks
102
103 - (void)connection:(NSURLConnection *)connection didReceiveResponse:(NSURLResponse *)response
```

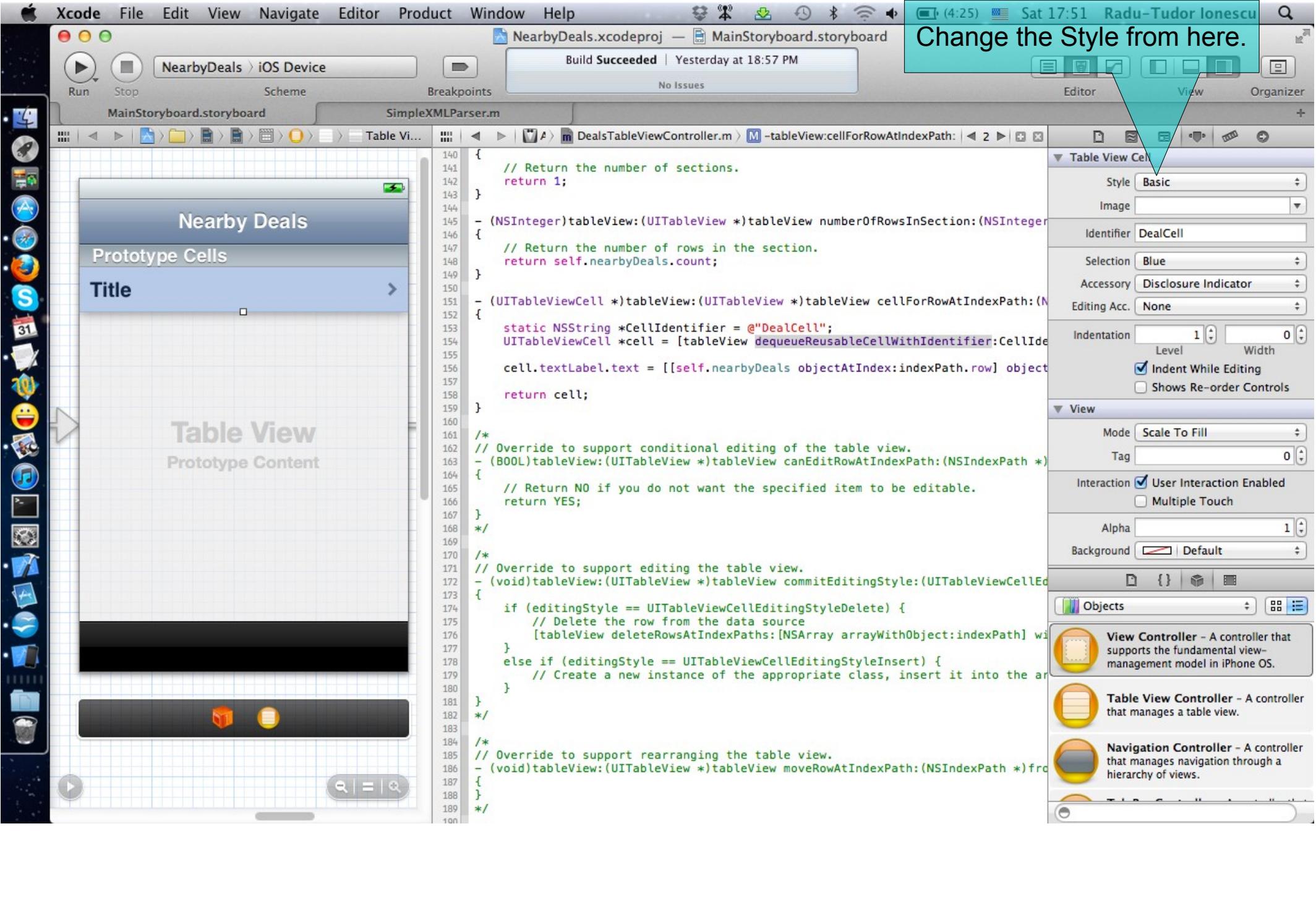
## Task 4

**Task:** Use the `SimpleXMLParser` class methods to parse the XML with nearby deals and display them in your Table View.

9. Let's show up the Utilities area.
10. Click on the Prototype Cell in Interface Builder to see its attributes in Inspector.
11. Change the Prototype Cell's Style to Basic in Attributes Inspector.

Look on the next screenshot for help.

12. Hide back the Utilities area.



Change the Style from here.

```
140 {
141     // Return the number of sections.
142     return 1;
143 }
144
145 - (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)
146 {
147     // Return the number of rows in the section.
148     return self.nearbyDeals.count;
149 }
150
151 - (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)
152 {
153     static NSString *CellIdentifier = @"DealCell";
154     UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier];
155
156     cell.textLabel.text = [[self.nearbyDeals objectAtIndex:indexPath.row] object];
157
158     return cell;
159 }
160
161 /*
162 // Override to support conditional editing of the table view.
163 - (BOOL)tableView:(UITableView *)tableView canEditRowAtIndexPath:(NSIndexPath *)
164 {
165     // Return NO if you do not want the specified item to be editable.
166     return YES;
167 }
168 */
169
170 /*
171 // Override to support editing the table view.
172 - (void)tableView:(UITableView *)tableView commitEditingStyle:(UITableViewCellEditingStyle)
173 {
174     if (editingStyle == UITableViewCellEditingStyleDelete) {
175         // Delete the row from the data source
176         [tableView deleteRowsAtIndexPaths:[NSArray arrayWithObject:indexPath] withRowAnimation];
177     }
178     else if (editingStyle == UITableViewCellEditingStyleInsert) {
179         // Create a new instance of the appropriate class, insert it into the array, and
180         // add a new row to the table view.
181     }
182 }
183 */
184
185 /*
186 // Override to support rearranging the table view.
187 - (void)tableView:(UITableView *)tableView moveRowAtIndexPath:(NSIndexPath *)fromIndexPath:
188 {
189 }
190 */
```

**Table View Cell**

Style: Basic

Image: [dropdown]

Identifier: DealCell

Selection: Blue

Accessory: Disclosure Indicator

Editing Acc.: None

Indentation: 1 (Level) 0 (Width)

Indent While Editing

Shows Re-order Controls

**View**

Mode: Scale To Fill

Tag: 0

Interaction:  User Interaction Enabled

Multiple Touch

Alpha: 1

Background: Default

**Objects**

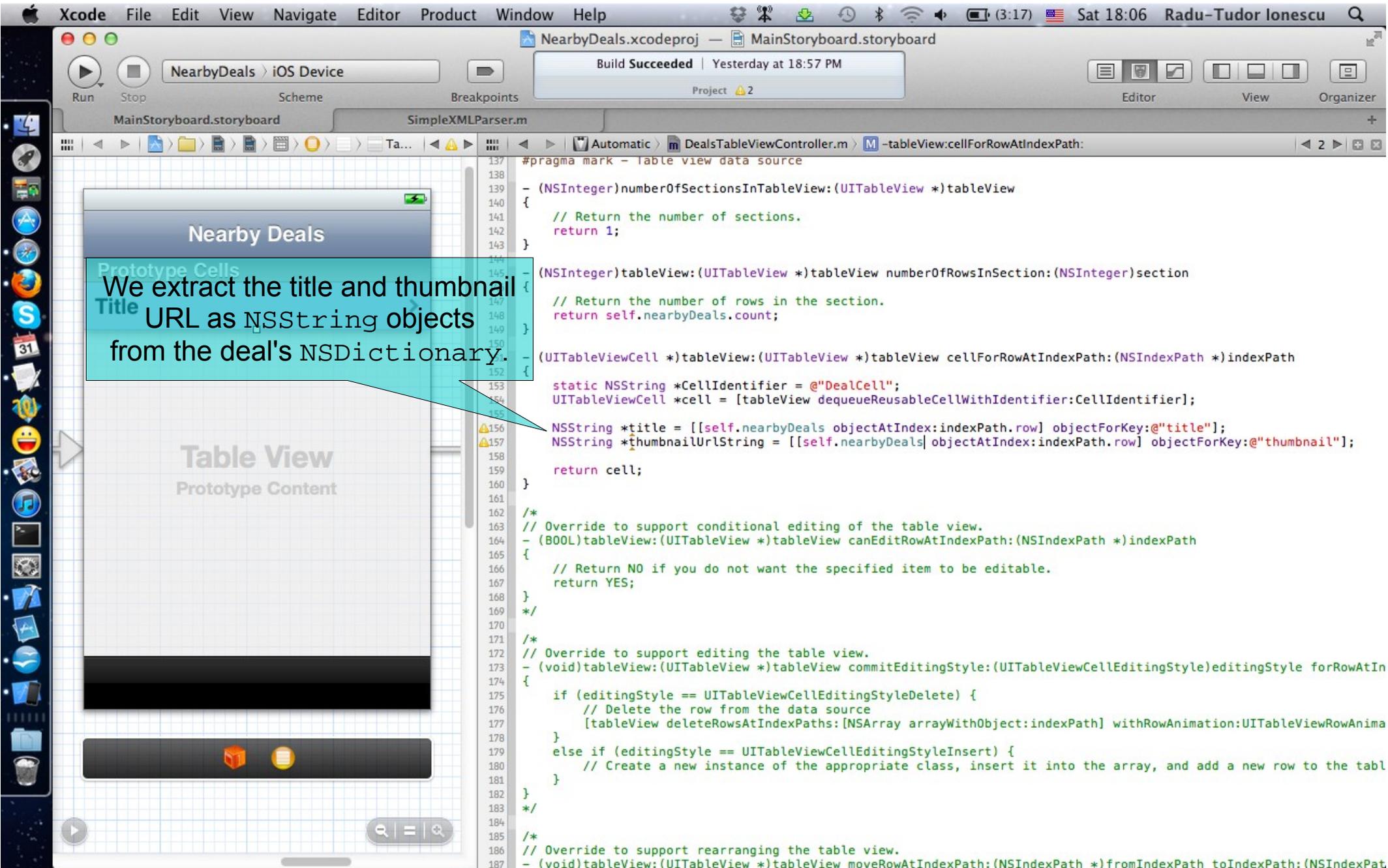
- View Controller** - A controller that supports the fundamental view-management model in iPhone OS.
- Table View Controller** - A controller that manages a table view.
- Navigation Controller** - A controller that manages navigation through a hierarchy of views.

## Task 4

Task: Use the `SimpleXMLParser` class methods to parse the XML with nearby deals and display them in your Table View.

13. Re-implement the `tableView:cellForRowAtIndexPath:` method to use the data from the new Model (that we received from the GeoAds+ server). Each Table View Cell will display information about a deal. For now, we want to present the deal title and its thumbnail photo. We extract this information from the `NSDictionary` at `indexPath.row` inside the `nearbyDeals` array.

The next slides will guide you through the re-implementation of this method.



We extract the title and thumbnail URL as NSString objects from the deal's NSDictionary.

```
137 #pragma mark - table view data source
138
139 - (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
140 {
141     // Return the number of sections.
142     return 1;
143 }
144
145 - (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
146 {
147     // Return the number of rows in the section.
148     return self.nearbyDeals.count;
149 }
150
151 - (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath
152 {
153     static NSString *CellIdentifier = @"DealCell";
154     UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier];
155
156     NSString *title = [[self.nearbyDeals objectAtIndex:indexPath.row] objectForKey:@"title"];
157     NSString *thumbnailUrlString = [[self.nearbyDeals objectAtIndex:indexPath.row] objectForKey:@"thumbnail"];
158
159     return cell;
160 }
161
162 /*
163  * Override to support conditional editing of the table view.
164  * - (BOOL)tableView:(UITableView *)tableView canEditRowAtIndexPath:(NSIndexPath *)indexPath
165  * {
166  *     // Return NO if you do not want the specified item to be editable.
167  *     return YES;
168  * }
169  */
170
171 /*
172  * Override to support editing the table view.
173  * - (void)tableView:(UITableView *)tableView commitEditingStyle:(UITableViewCellEditingStyle)editingStyle forRowAtIndexPath:(NSIndexPath *)indexPath
174  * {
175  *     if (editingStyle == UITableViewCellEditingStyleDelete) {
176  *         // Delete the row from the data source
177  *         [tableView deleteRowsAtIndexPaths:[NSArray arrayWithObject:indexPath] withRowAnimation:UITableViewRowAnimationAutomatic]
178  *     }
179  *     else if (editingStyle == UITableViewCellEditingStyleInsert) {
180  *         // Create a new instance of the appropriate class, insert it into the array, and add a new row to the table
181  *     }
182  * }
183  */
184
185 /*
186  * Override to support rearranging the table view.
187  * - (void)tableView:(UITableView *)tableView moveRowAtIndexPath:(NSIndexPath *)fromIndexPath toIndexPath:(NSIndexPath *)toIndexPath
```

Xcode File Edit View Navigate Editor Product Window Help

NearbyDeals.xcodeproj — MainStoryboard.storyboard

Build Succeeded | Yesterday at 18:57 PM

Project 1

Run Stop Scheme Breakpoints Editor View Organizer

MainStoryboard.storyboard SimpleXMLParser.m

Automatic DealsTableViewCell.m tableView:cellForRowAtIndexPath:

137 #pragma mark - table view data source  
138  
139 - (NSInteger)numberOfSectionsInTableView:(UITableView \*)tableView  
140 {  
141 // Return the number of sections.  
142 return 1;  
143 }  
144  
145 - (NSInteger)tableView:(UITableView \*)tableView numberOfRowsInSection:(NSInteger)section  
146 {  
147 // Return the number of rows in the section.  
148 return self.nearbyDeals.count;  
149 }  
150  
151 - (UITableViewCell \*)tableView:(UITableView \*)tableView cellForRowAtIndexPath:(NSIndexPath \*)indexPath  
152 {  
153 static NSString \*cellIdentifier = @"DealCell";  
154 UITableViewCell \*cell = [tableView dequeueReusableCellWithIdentifier:cellIdentifier];  
155  
156 NSString \*title = [[self.nearbyDeals objectAtIndex:indexPath.row] objectForKey:@"title"];  
157 NSString \*thumbnailUrlString = [[self.nearbyDeals objectAtIndex:indexPath.row] objectForKey:@"thumbnail"];  
158  
159 cell.textLabel.text = title;  
160  
161 return cell;  
162 }  
163  
164 /\*  
165 // Override to support conditional editing of the table view.  
166 - (BOOL)tableView:(UITableView \*)tableView canEditRowAtIndexPath:(NSIndexPath \*)indexPath  
167 {  
168 // Return NO if you do not want the specified item to be editable.  
169 return YES;  
170 }  
171 \*/  
172  
173 /\*  
174 // Override to support editing the table view.  
175 - (void)tableView:(UITableView \*)tableView commitEditingStyle:(UITableViewCellEditingStyle)editingStyle forRowAtIndexPath:  
176 {  
177 if (editingStyle == UITableViewCellEditingStyleDelete) {  
178 // Delete the row from the data source  
179 [tableView deleteRowsAtIndexPaths:[NSArray arrayWithObject:indexPath] withRowAnimation:UITableViewRowAnimationAutomatic]  
180 }  
181 else if (editingStyle == UITableViewCellEditingStyleInsert) {  
182 // Create a new instance of the appropriate class, insert it into the array, and add a new row to the table  
183 }  
184 }  
185 \*/  
186  
187 /\*

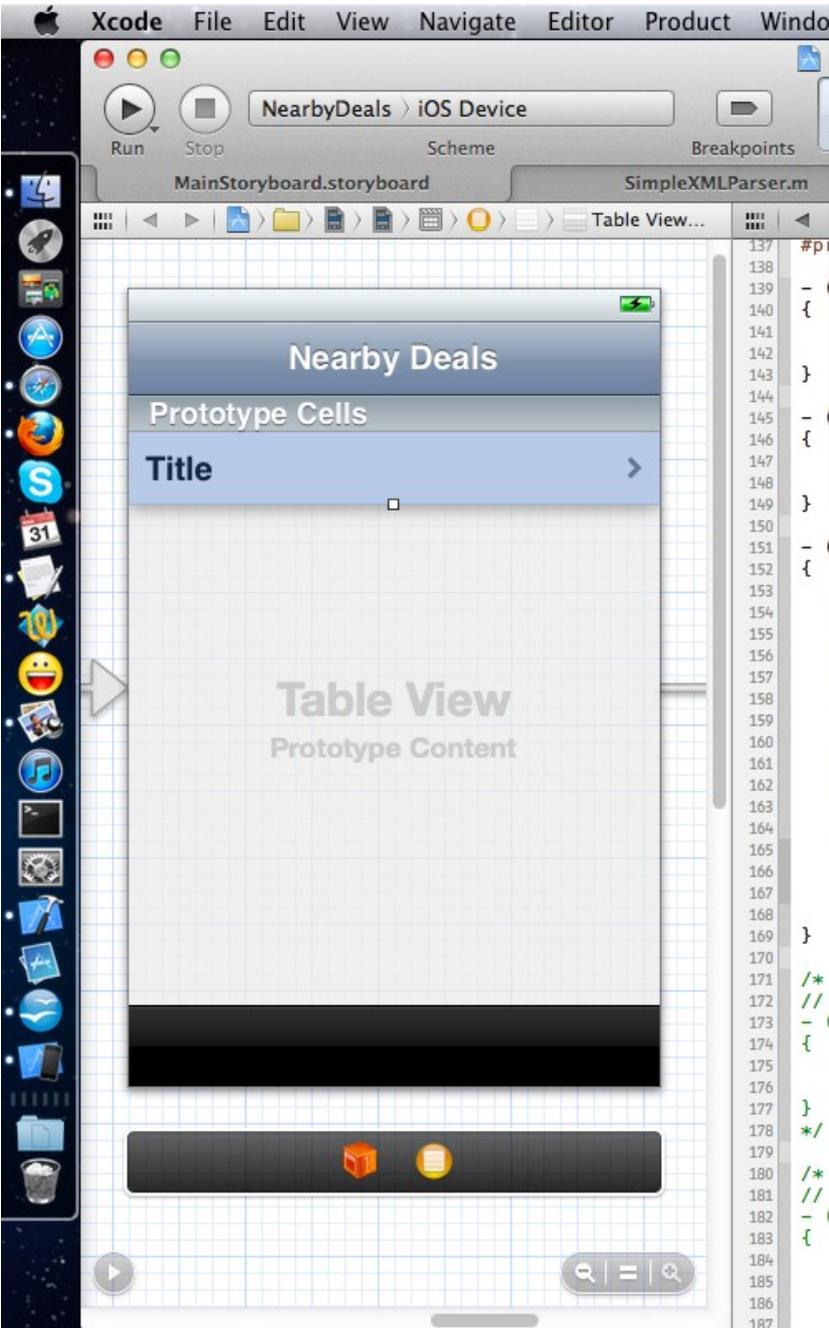
Nearby Deals

Prototype Cells

Title

Table View  
Prototype Content

Set the cell text to be the deal's title.



Send synchronous request to download the thumbnail image. This request will block this method until the image data is downloaded from the URL specified by `thumbnailUrl`. Because of this, the Table View will load very slow and it may be unpleasant for the user. It is better to use an asynchronous request in this case. The asynchronous request will not block this method (because it will execute on another thread) and the Table View loading will look really smooth. We are going to request the thumbnails asynchronously later. For now, we stick to the synchronous request which is very easy to implement in one line of code.

```
137 #pragma mark - Table view data source
138
139 - (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
140 {
141     // Return the number of sections.
142     return 1;
143 }
144
145 - (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
146 {
147     // Return the number of rows in the section.
148     return self.nearbyDeals.count;
149 }
150
151 - (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath
152 {
153     static NSString *CellIdentifier = @"DealCell";
154     UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier];
155
156     NSString *title = [[self.nearbyDeals objectAtIndex:indexPath.row] objectForKey:@"title"];
157     NSString *thumbnailUrlString = [[self.nearbyDeals objectAtIndex:indexPath.row] objectForKey:@"thumbnail"];
158
159     cell.textLabel.text = title;
160
161     NSURL *thumbnailUrl = [NSURL URLWithString:thumbnailUrlString];
162     NSData *thumbnailData = [NSData dataWithContentsOfURL:thumbnailUrl];
163
164     if (thumbnailData != nil)
165     {
166         cell.imageView.image = [UIImage imageWithData:thumbnailData];
167     }
168     return cell;
169 }
170
171 /*
172 // Override to support conditional editing of the table view.
173 - (BOOL)tableView:(UITableView *)tableView canEditRowAtIndexPath:(NSIndexPath *)indexPath
174 {
175     // Return NO if you do not want the specified item to be editable.
176     return YES;
177 }
178 */
179
180 /*
181 // Override to support editing the table view.
182 - (void)tableView:(UITableView *)tableView commitEditingStyle:(UITableViewCellEditingStyle)editingStyle forRowAtIndexPath:(NSIndexPath *)indexPath
183 {
184     if (editingStyle == UITableViewCellEditingStyleDelete)
185     {
186         // Delete the row from the data source.
187         [tableView deleteRowsAtIndexPaths:[indexPath] withRowAnimation:UITableViewRowAnimationDefault];
188     }
189 }
```

Build an UIImage from the thumbnailData and set it to the imageView of the Table View Cell. First we make sure that our synchronous request returns something not nil.

# Assignment 1

Assignment: Adjust the GeoAds+ URL string that is constructed inside the `requestDealsNearLocation:limit:` method in order to obtain nearby deals only for Restaurants and Bars.

Hint: Look at the request to GeoAds+ API we made from Safari.

# Assignment 2

Assignment: Add the deal's subtitle to the Table View Cell.

Hint: You have to change the Prototype Cell Style to Subtitle and set the cell's `detailTextLabel` programmatically.

**Congratulations!**