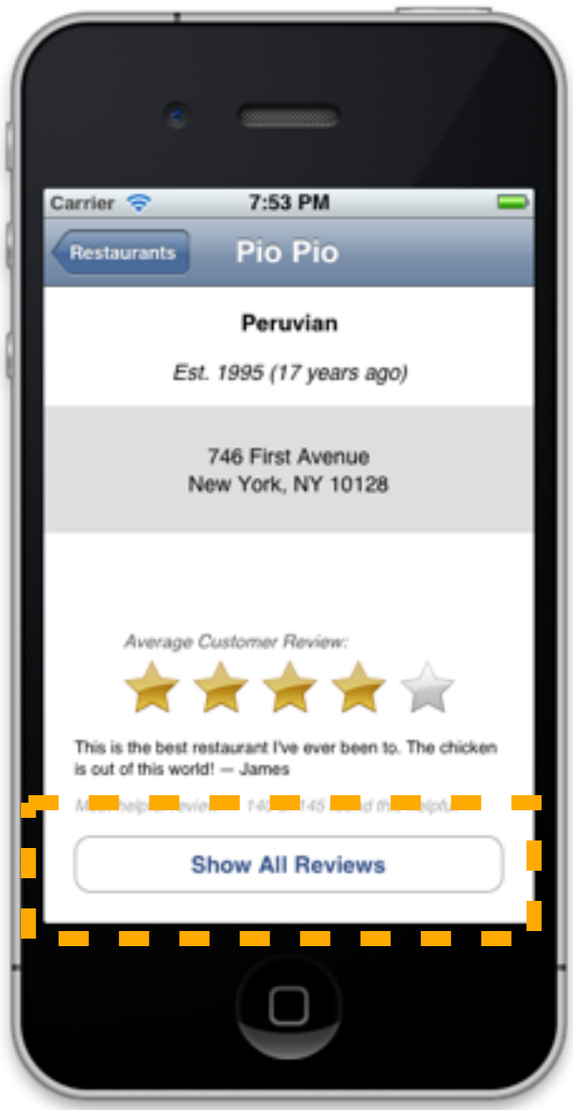
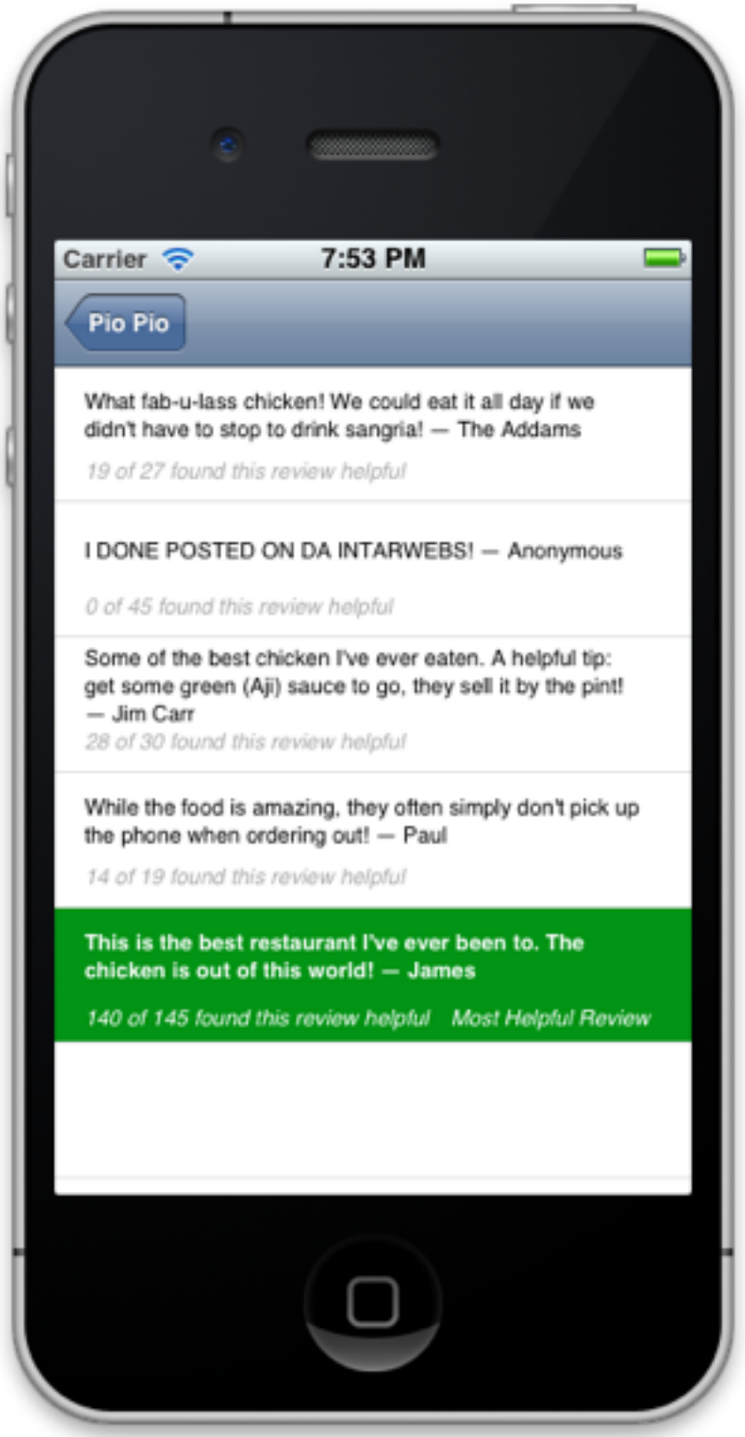
Assignment 5

Custom Table Cells

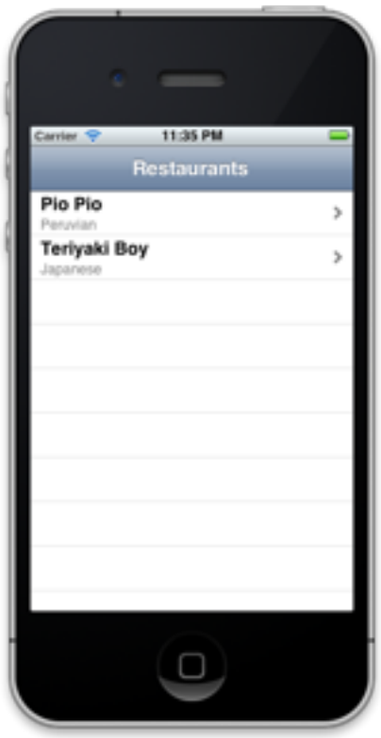
Goal



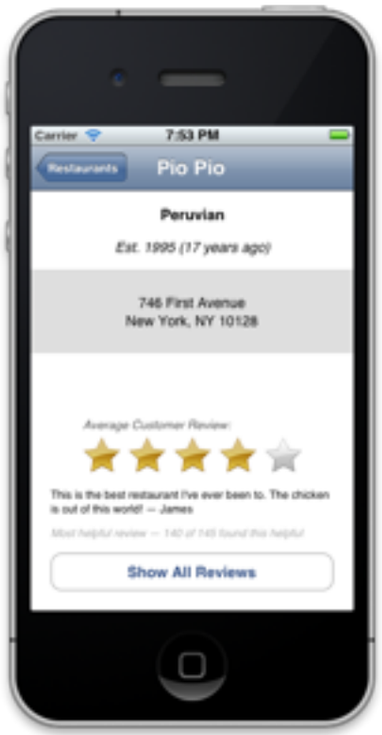


- This is very similar to the last assignment
- We'll be creating another table, this time to show all the reviews of the restaurant

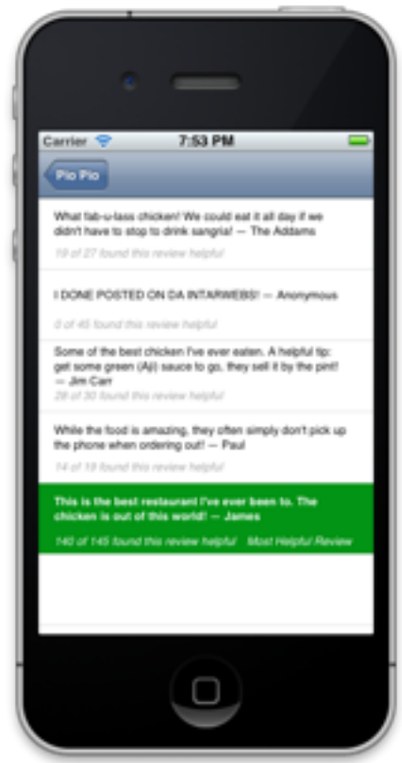
Goal

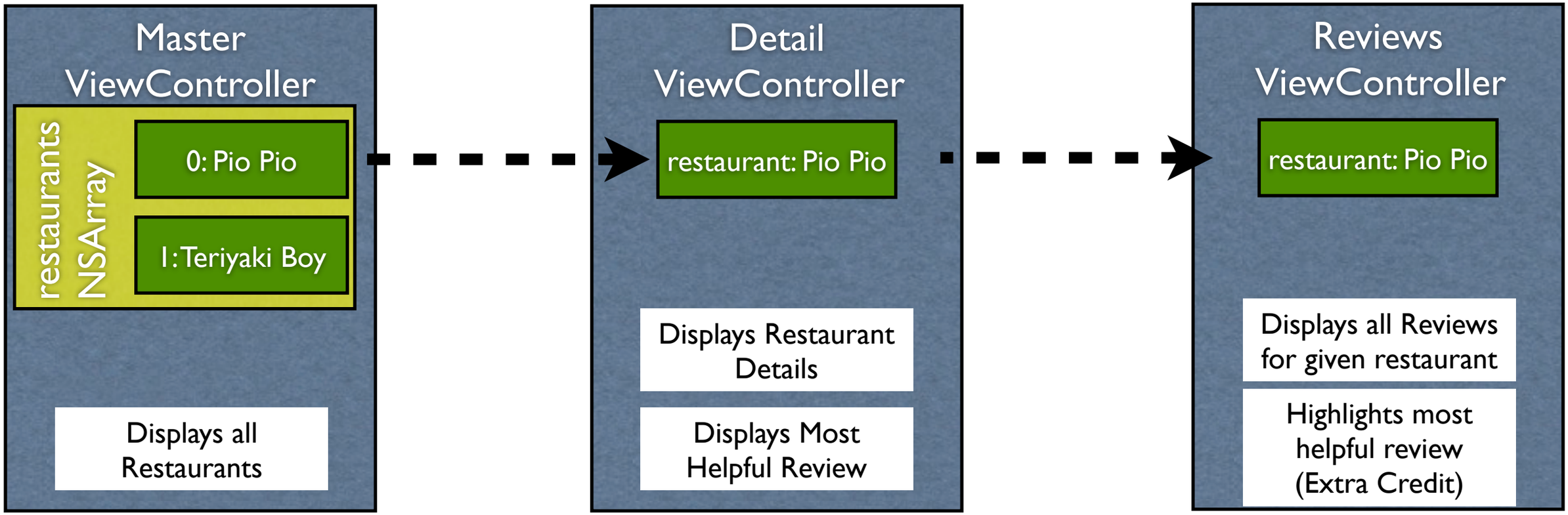


Segue



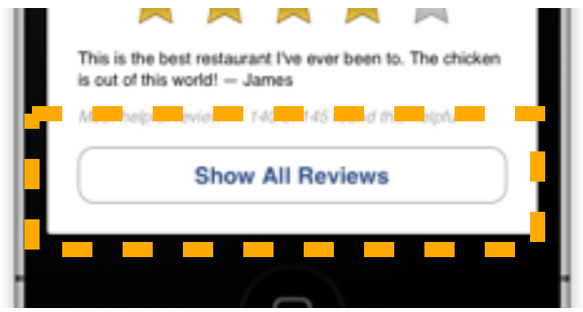
Segue





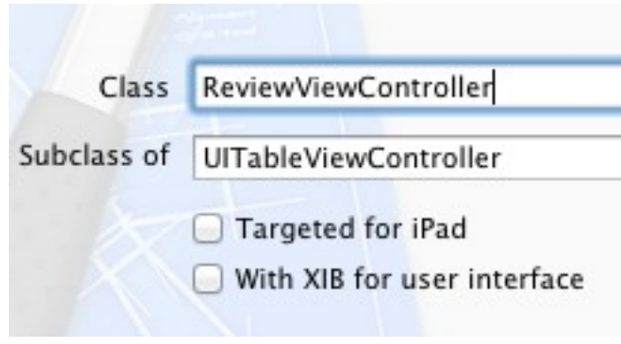
Creating another View

- In your storyboard, drag a new **Table View Controller** out onto the storyboard from the library onto the canvas.
- Drag a rounded rect button onto the detail view that says "Show All Reviews"
- Create a segue. Right click and drag from the button to the new table view.
- Build and run, and you should be able to see a new empty table appear from tapping the "Show All Reviews" button.



Creating a View Controller Subclass

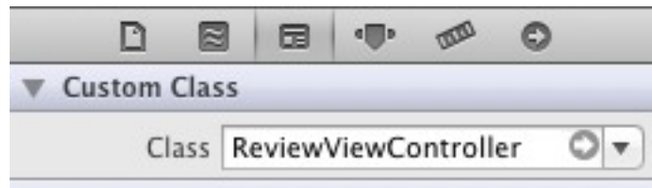
- We now need to create another file for displaying the review.
- In your project navigator, right click "Restaurants" and choose "New File...".
- Then pick "Objective-C class" under the "Cocoa Touch" section.
- On the next screen, name your class "ReviewViewController, and make it a subclass of "UITableViewController". Leave both checkboxes unchecked.
- Then, specify that the file goes in the same location on your computer as MasterViewController and DetailViewController.



Hint: You may want to drag the ReviewViewController file up to be alongside the Master and Detail view controllers.

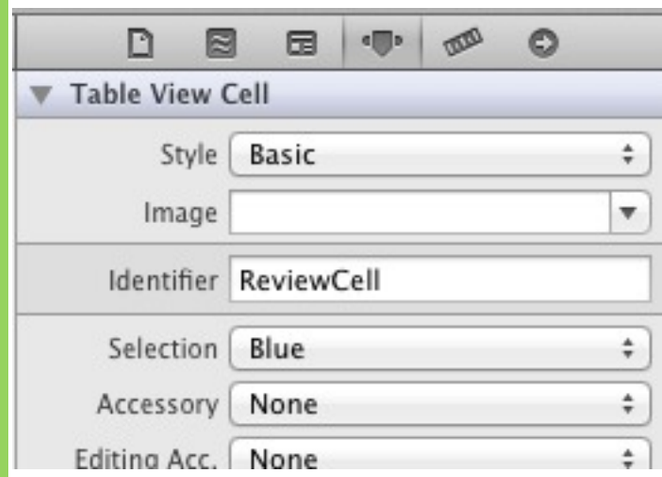
Creating a View Controller Subclass

- Now, we need to tell your storyboard that the controller controlling your UITableView is actually your ReviewViewController.
- In your storyboard, choose your new view. Then, in the gutter below the view, choose your Table View Controller.
- In the utilities panel on the left, change the the "Identity" tab and select "ReviewViewController" from the **Class** list.



Setting test properties on the cell

- Next, we need to configure the prototype cell to be able to be found by our view.
- Add an identifier of "ReviewCell" to the cell
- Change it's style to be Basic. This will allow us to test, we'll change this to be more complex in a moment.



We should now have the table and it's controller connected.

Testing the cell

We're now going to see if our ViewController and TableView are connected.

- In in ReviewViewController.m:
- Change `- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView` to return 1
- Change `- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section` to return 1
- In `- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath`, configure the cell to edit the textLabel's text property to say "test", and change the CellIdentifier to be "ReviewCell"

Build and run, and if everything works, sync with GitHub.

Adding a restaurant property to ReviewViewController

We're going to perform a segue just like we did from Master to Detail.

Because of that, we are going to need to pass the restaurant from the Detail to the Reviews view controller, so we need to expose it as a property.

- In ReviewViewController.h
  - #import the restaurant class
  - Add a instance variable for "restaurant" (type: Restaurant*)
  - Add a property for restaurant
- In ReviewViewController.m
  - Synthesize the property

This should all be very familiar.

Passing the restaurant to the Reviews controller

- In DetailViewController.m
  - Add an import for ReviewViewController
  - Inside: `-(void)prepareForSegue:(UIStoryboardSegue *)segue sender:(id)sender`
    - Get the VC from the segue: `ReviewViewController* reviewVC = (ReviewViewController*)[segue destinationViewController];`
    - Pass the restaurant using your newly created property: `reviewVC.restaurant = restaurant;`

Test the segue

- In ReviewViewController.m
  - Inside `- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section`
    - Return the count of the reviews property from the restaurant
- Test your code to ensure that you are showing the correct number of cells in the table view (there should be one per review, though they should still all say “test”)

Create a Custom Cell

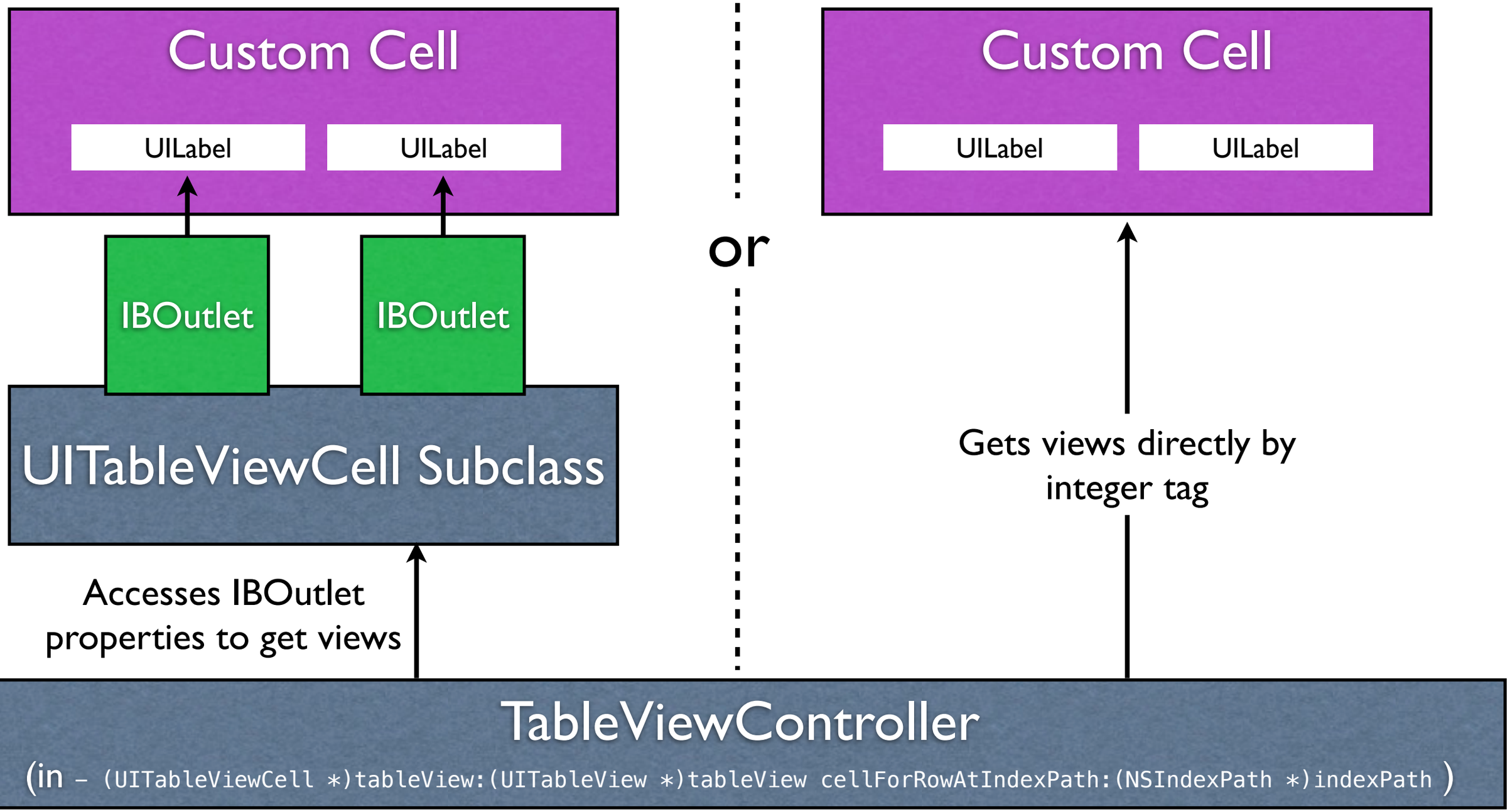
Now we need to populate each cell with the text of our review. But we want these cells laid out in a custom way. This is where using prototype cells is helpful.

- In your Storyboard, Select your cell and change it's style to "custom".
- Copy the 2 labels from your detail view and paste them into the cell. It should look something like the image on the right.

Setting Values on a Custom Cell

We need to make a connection now between our view controller (which decides what to put in each cell) and our newly designed custom cell.

There are 2 ways to accomplish this.



Setting tags on views

We'll go with the simpler approach. We'll set a tag on each view we want to access.

A tag is a simple int that allows us access to a view. If we ask the UITableViewCell for a view with a certain tag, we will get the correct view back.

This approach makes sense for simpler views. For more complex views, IBOutlets make things more clear.

- In your Storyboard, Select your review text label and set it's tag to 1.
- Do the same for the helpful review label, but set it's tag to 2.

Getting views using tags

```
- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:
(NSIndexPath *)indexPath
{
    NSString *CellIdentifier = @"ReviewCell";
    UITableViewCell *cell = [tableView
dequeueReusableCellWithIdentifier:CellIdentifier];
    UILabel* reviewTextLabel = (UILabel*)[cell viewWithTag:1];
    UILabel* reviewHelpfulnessLabel = (UILabel*)[cell viewWithTag:2];

continued, next page...
```

Getting the review and setting the labels

```
from previous page...

   Review* reviewForIndexPath = [restaurant.reviews objectAtIndex:indexPath.row];
    reviewTextLabel.text = reviewForIndexPath.text;
    reviewHelpfulnessLabel.text = [NSString stringWithFormat:@"%i of %i found
this review helpful",reviewForIndexPath.numberOfHelpfulReviews,
reviewForIndexPath.numberOfUnhelpfulReviews +
reviewForIndexPath.numberOfHelpfulReviews];
    return cell;
}
```

This should seem very familiar. We did the exact same thing when we displayed the restaurant in the MasterViewController, except that this time we simply ask for the review out of the already existing reviews array on the restaurant.

Assignment 5 Completed!

- Extra Credit:
  - Highlight the cell with the most helpful review.
  - You should try formatting this cell differently than the other cells.
  - You will need to make an additional Prototype cell to accomplish this elegantly