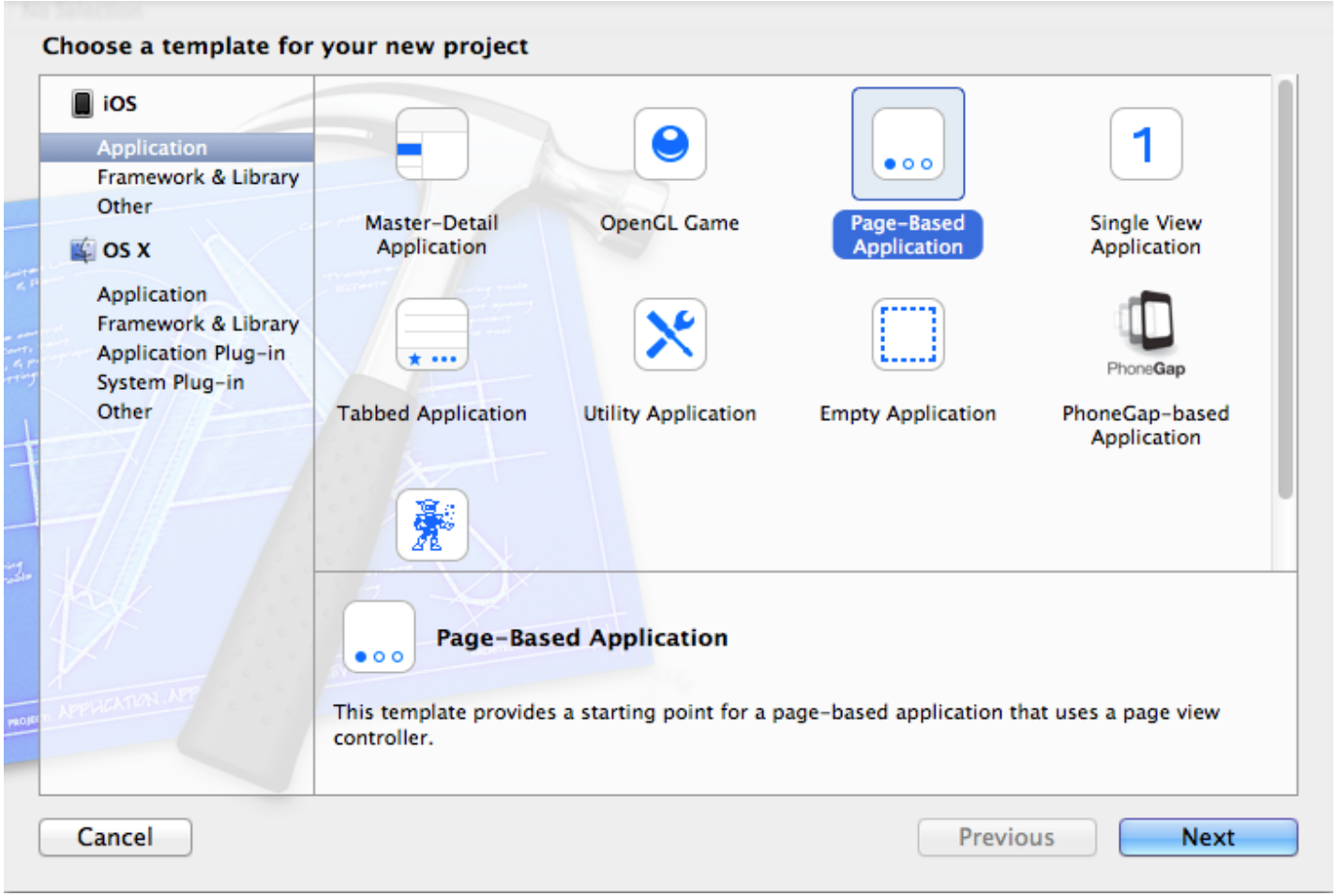
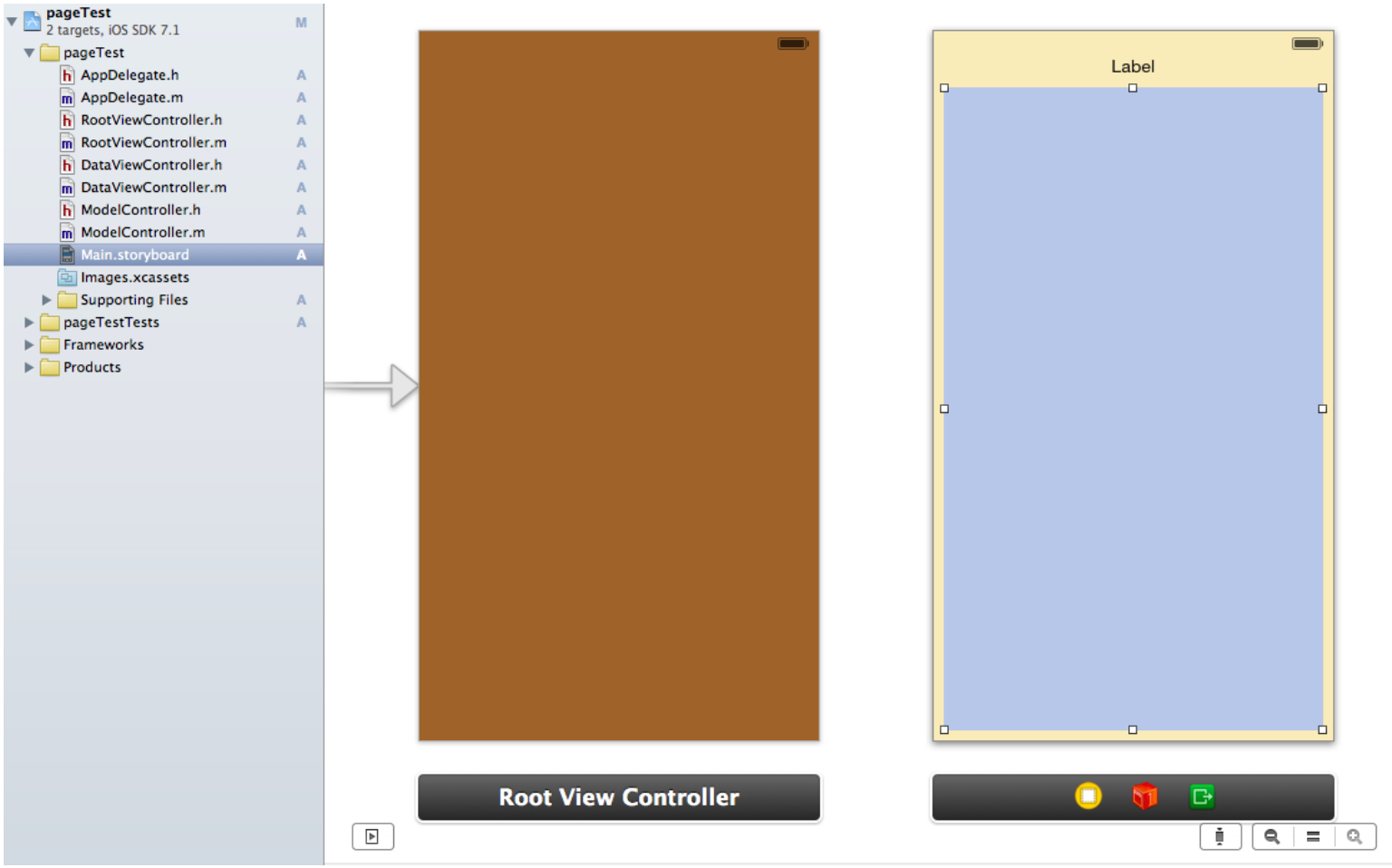
iPhone Apps Development using Objective C & Xcode (Lesson 5)

By Dannis Mok

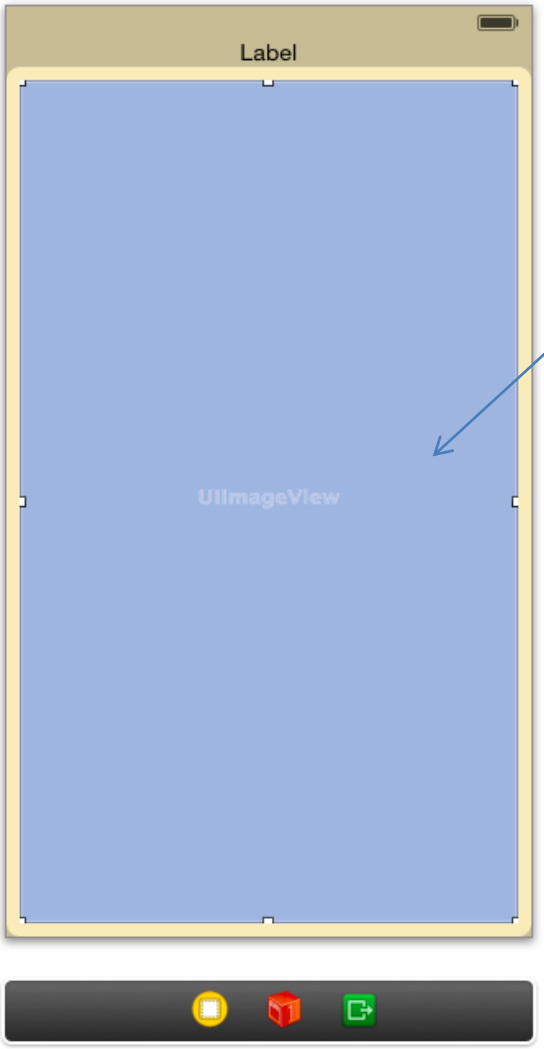
Page based Application



Page based Application



Page based Application




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

@interface DataViewController : UIViewController

@property (strong, nonatomic) IBOutlet UILabel *dataLabel;
@property (strong, nonatomic) id dataObject;

@property (weak, nonatomic) IBOutlet UIImageView *
    myImageView;


@end
```

Add an ImageView into the DataViewController as an outlet

Page based Application

```
@interface ModelController()
@property (readonly, strong, nonatomic) NSArray *pageData;
@end

@implementation ModelController

- (id)init
{
    self = [super init];
    if (self) {
        // Create the data model.
        NSDateFormatter *dateFormatter = [[NSDateFormatter alloc] init];
        _pageData = [NSArray
            arrayWithObjects:@"flower1.jpeg",@"flower2.jpg",@"flower3.jpg",
            @"flower4.jpeg",@"flower5.jpeg",nil];
    }
    return self;
}
```

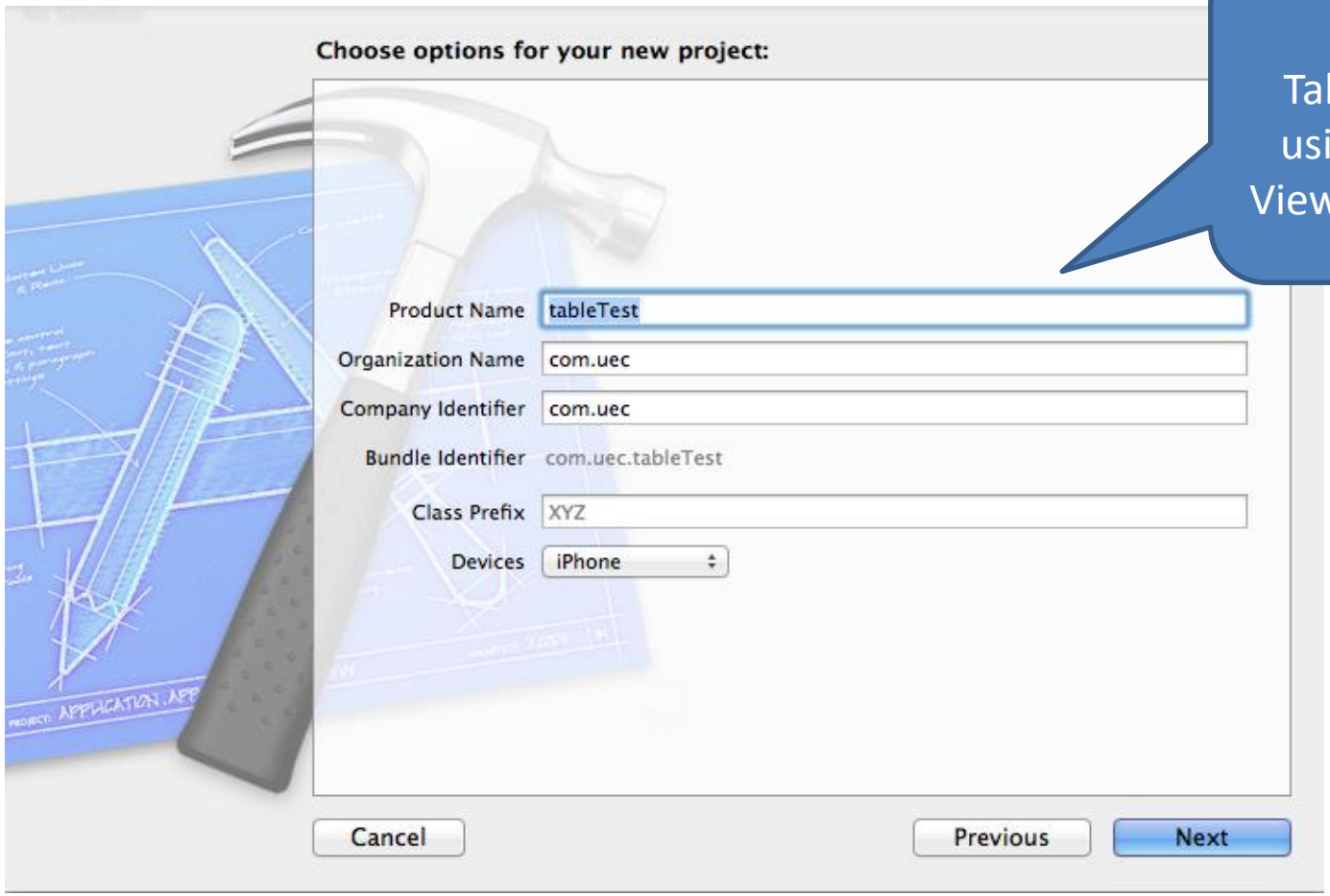
Update the _pageData array in the ModelController to contain the file names of those photos added. These photos can be displayed one by one

Page based Application

```
#import "DataViewController.h"

@interface DataViewController ()

@end

@implementation DataViewController

- (void)viewDidLoad
{
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a
        nib.
}

- (void)didReceiveMemoryWarning
{
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

- (void)viewWillAppear:(BOOL)animated
{
    [super viewWillAppear:animated];
    self.dataLabel.text = [self.dataObject description];

    [self.myImageView setImage:[UIImage imageNamed:self.dataObject]];

}

@end
```

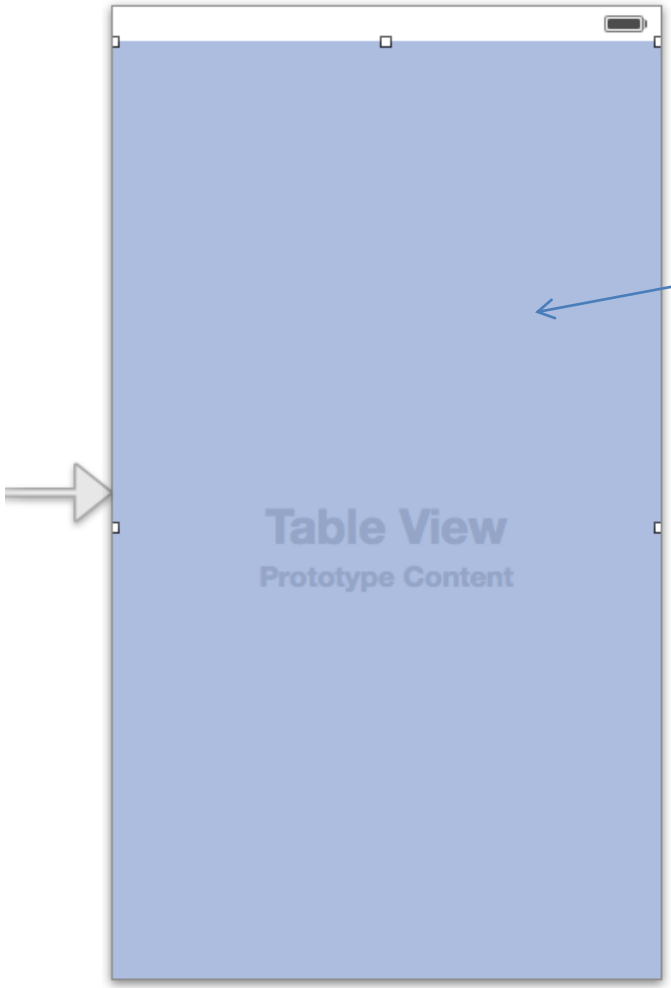
Add this statement to let the photo to display in the ImageView whenever the view appear

TableView



Create a TableView by using a Single View Application

TableView




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

@interface ViewController : UIViewController <
    UITableViewDelegate, UITableViewDataSource>


@property (weak, nonatomic) IBOutlet UITableView *
    myTableView;


@end
```

TableView needs 2 protocol to operate.

1) TableViewDataSource
2) TableViewDelegate

TableView

```
@implementation ViewController

- (void)viewDidLoad
{
    [super viewDidLoad];

    self.myTableView.dataSource = self;

    self.myTableView.delegate = self;

}

-(NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section {

    return 10;
}
```

Set the tableView's dataSource and delegate providers to be the ViewController itself

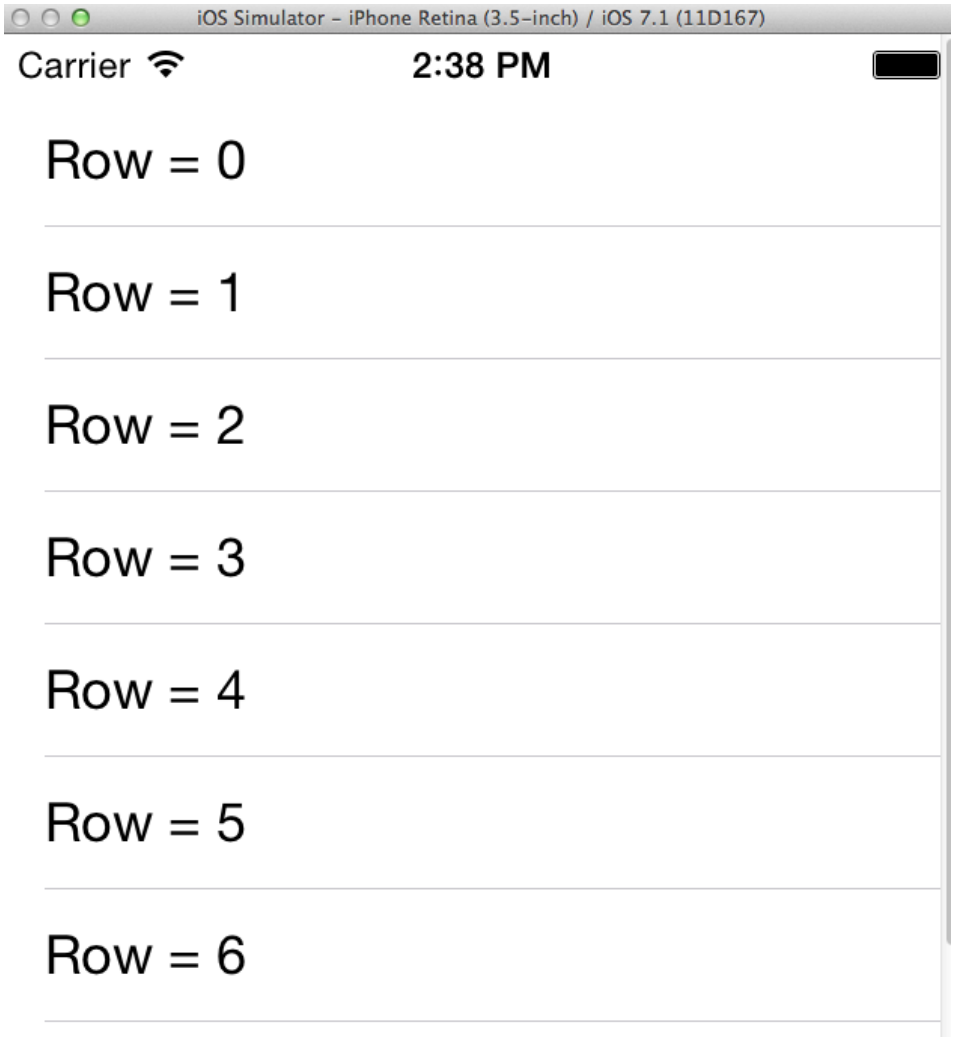
Set the number of rows in the table

TableView

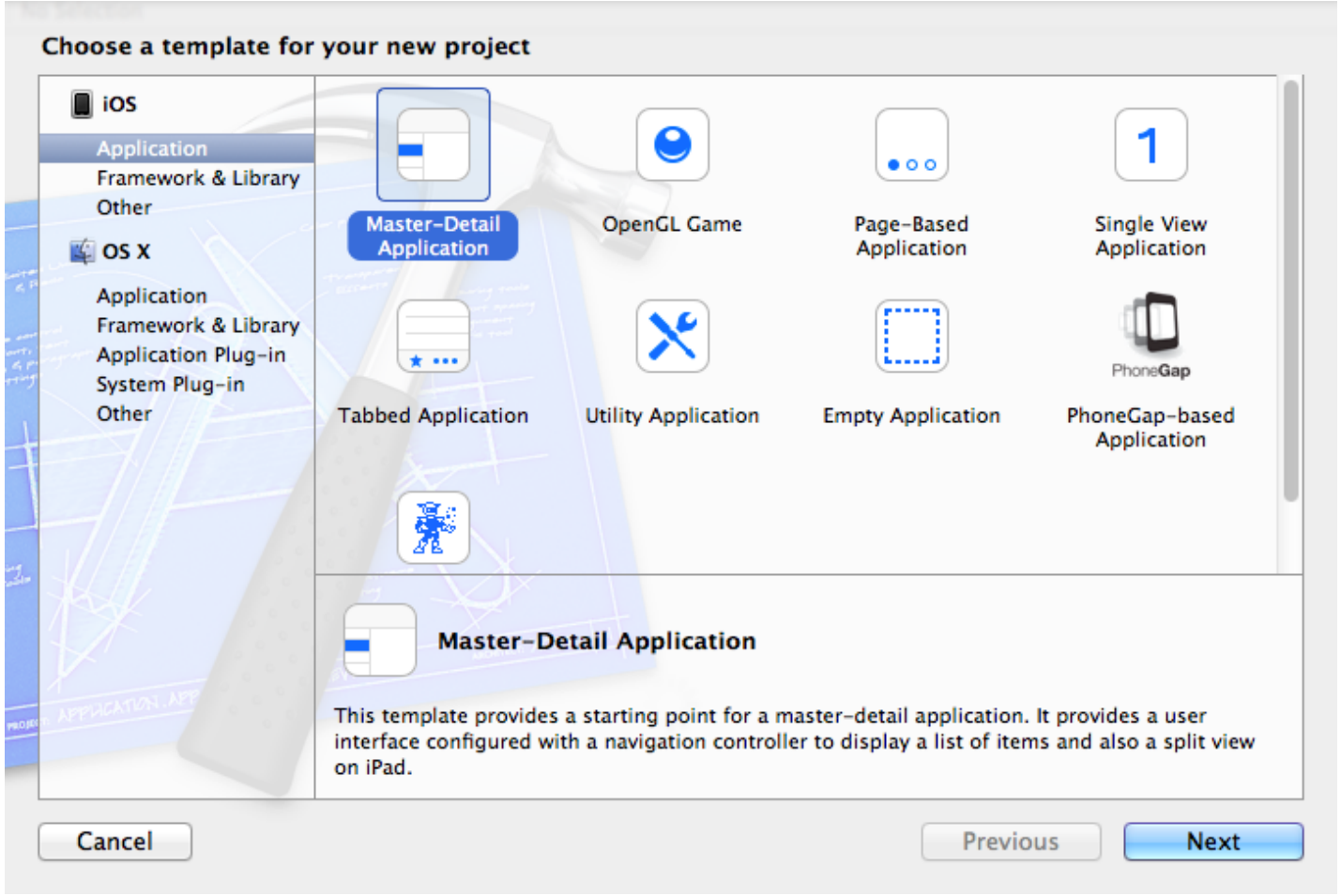
Below method is used to return the content of each cell

```
-(UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)
    indexPath {

    static NSString *CellIdentifier = @"Cell";

    UITableViewCell *cell=[tableView dequeueReusableCellWithIdentifier:CellIdentifier];

    if(cell == nil) {

        cell = [[UITableViewCell alloc]initWithStyle:UITableViewCellStyleDefault reuseIdentifier:
            CellIdentifier];

    }

    cell.textLabel.text = [NSString stringWithFormat:@"Row = %d",indexPath.row];

    return cell;

}
```

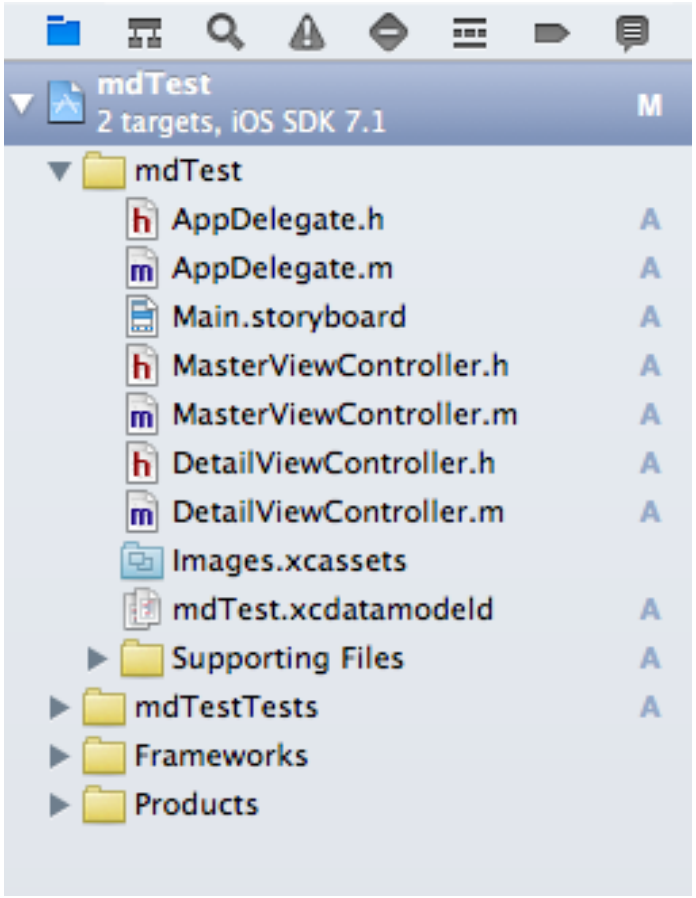
TableView

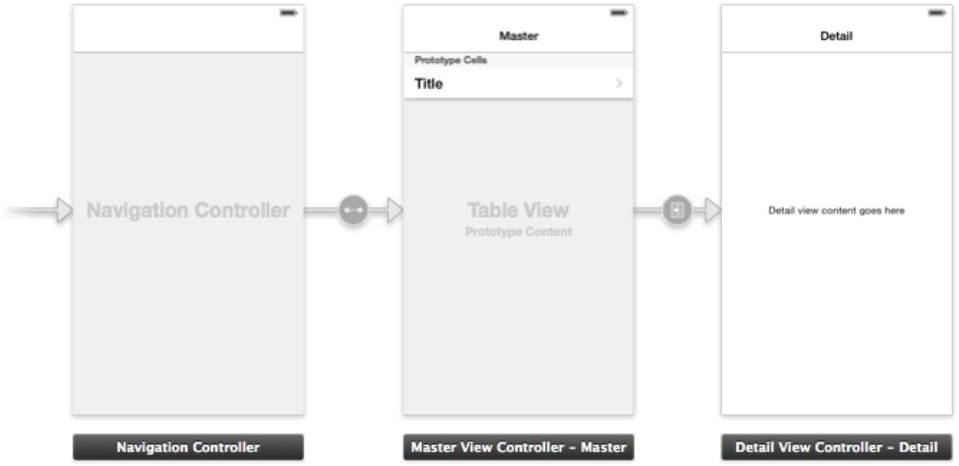

iOS Simulator - iPhone Retina (3.5-inch) / iOS 7.1 (11D167)
Carrier
2:38 PM
Row = 0
Row = 1
Row = 2
Row = 3
Row = 4
Row = 5
Row = 6

Master-Detail Application



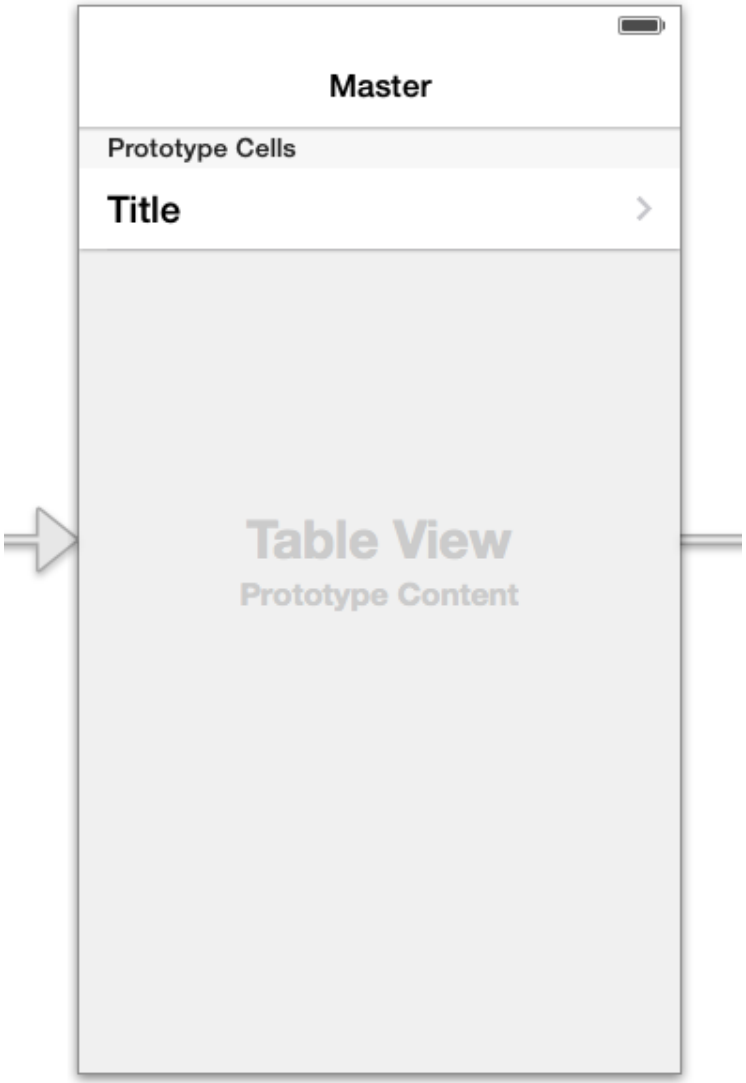
Master-Detail Application

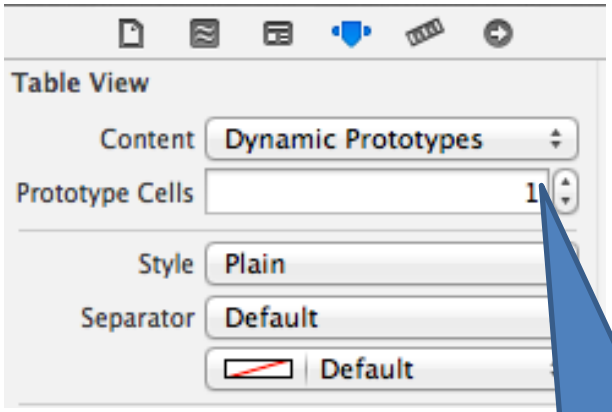




A Navigation Controller and TableView Controller will be created by default

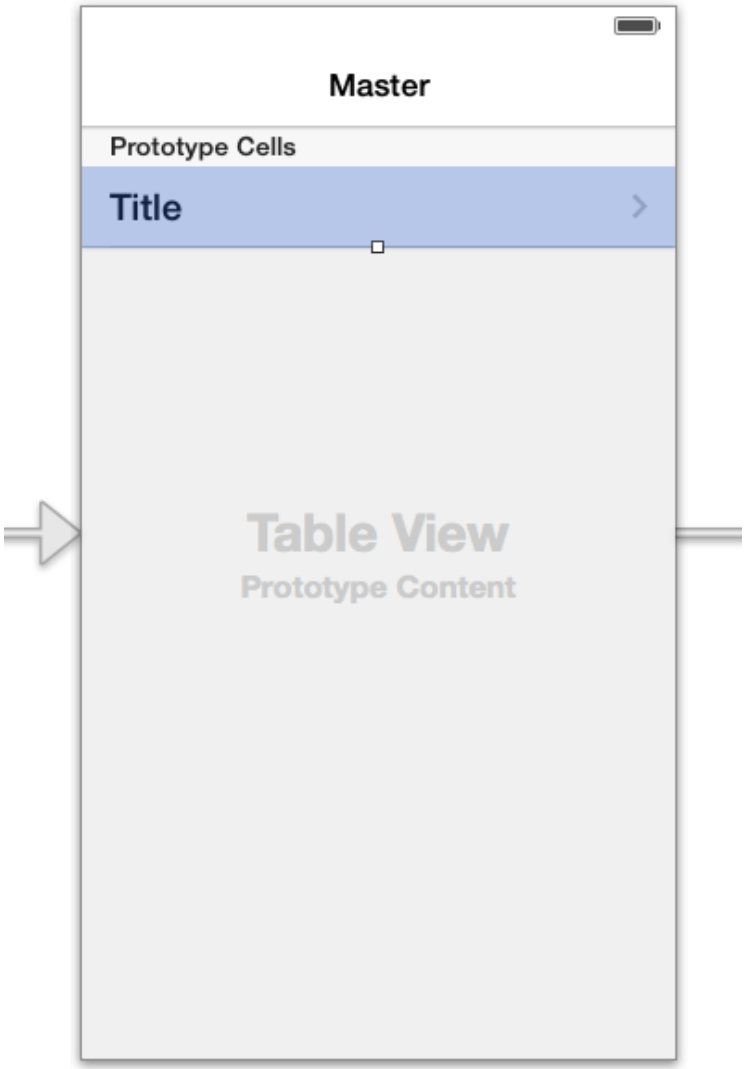
Master-Detail Application

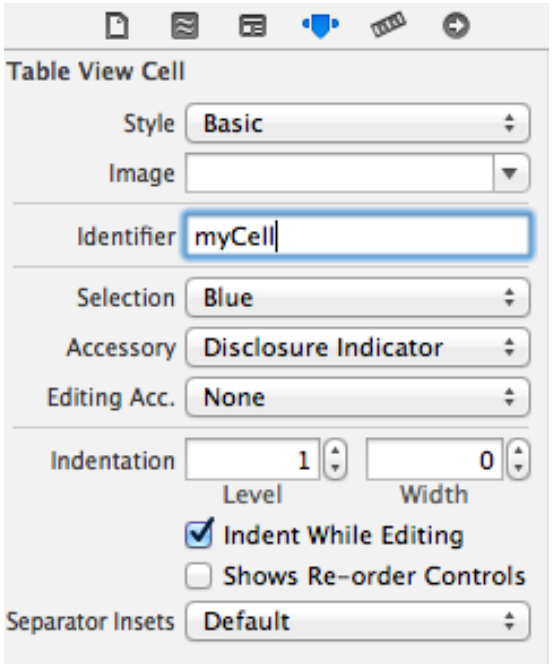




Make sure the TableView's Content is set to Dynamic Prototypes

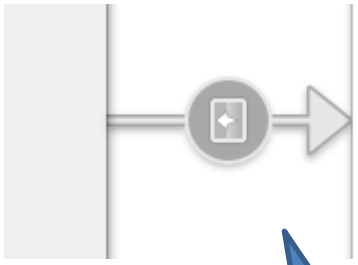
Master-Detail Application

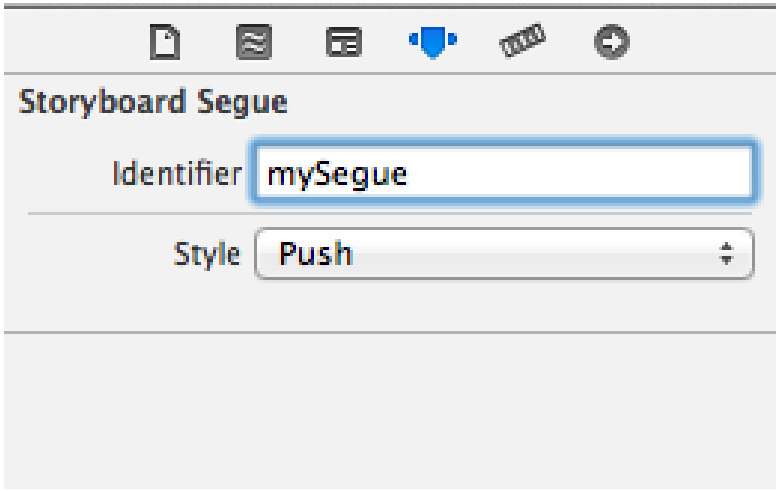




Change the Table View Cell's name to myCell

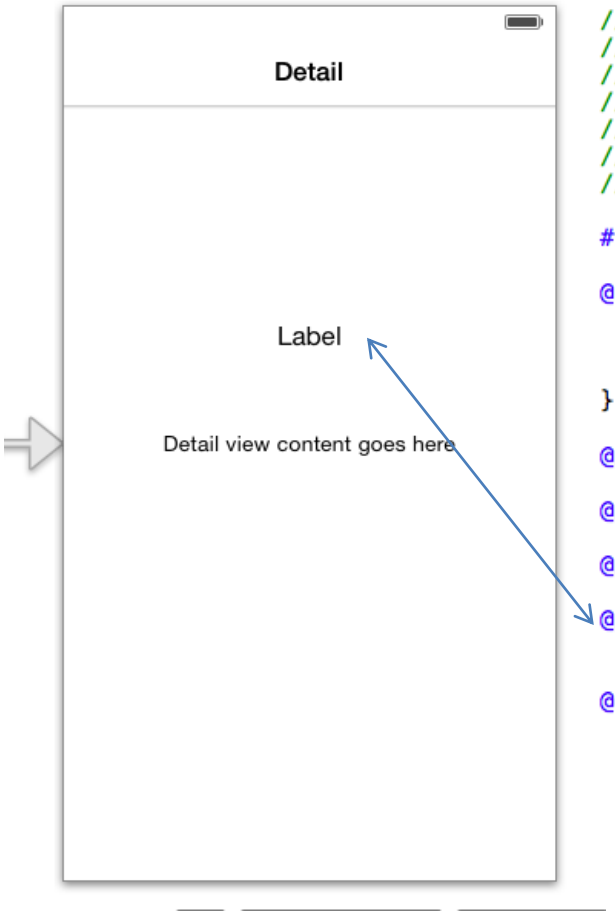
Master-Detail Application





Change the Segue's identifer to mySegue

Master-Detail Application




```
//
//  DetailViewController.h
//  mdTest
//
//  Created by Dannis Mok on 16/9/14.
//  Copyright (c) 2014 com.uec. All rights reserved.
//

#import <UIKit/UIKit.h>

@interface DetailViewController : UIViewController {

    NSString *myStr;

}

@property NSString *myStr;

@property (strong, nonatomic) id detailItem;

@property (weak, nonatomic) IBOutlet UILabel *detailDescriptionLabel;

@property (weak, nonatomic) IBOutlet UILabel *myLabel;

@end
```

Add a variable myStr to show the value pass from the MasterViewController

Master-Detail Application

```
//
//  DetailViewController.m
//  mdTest
//
//  Created by Dannis Mok on 16/9/14.
//  Copyright (c) 2014 com.uec. All rights reserved.
//

#import "DetailViewController.h"

@interface DetailViewController ()
- (void)configureView;
@end

@implementation DetailViewController

@synthesize myLabel,myStr;


- (void)viewDidLoad
{
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a
        nib.
    [self configureView];

    self.myLabel.text = myStr;

}
```

Synthesize the variables

Show the variable content

Master-Detail Application

```
@interface DetailViewController ()
- (void)configureView;
@end

@implementation DetailViewController

#pragma mark - Managing the detail item

- (void)setDetailItem:(id)newDetailItem
{
    if (_detailItem != newDetailItem) {
        _detailItem = newDetailItem;

        // Update the view.
        [self configureView];
    }
}

- (void)configureView
{
    // Update the user interface for the detail item.

    if (self.detailItem) {
        self.detailDescriptionLabel.text = [[self.detailItem valueForKey:@"timeStamp"] description];
    }
}

- (void)viewDidLoad
{
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.
    [self configureView];

    self.myLabel.text = myStr;

}
```

Show the variable content in the Label

Master-Detail Application

```
//
//  MasterViewController.h
//  mdTest
//
//  Created by Dannis Mok on 16/9/14.
//  Copyright (c) 2014 com.uec. All rights reserved.
//

#import <UIKit/UIKit.h>

#import <CoreData/CoreData.h>

@interface MasterViewController : UITableViewController <
    NSFetchedResultsControllerDelegate> {

    NSArray *myData;

}

@property (strong, nonatomic) NSFetchedResultsController *
    fetchedResultsController;
@property (strong, nonatomic) NSManagedObjectContext *managedObjectContext;

@end
```

Add a myData array in the MasterViewController

Master-Detail Application

```
@implementation MasterViewController

- (void)awakeFromNib
{
    [super awakeFromNib];
}

- (void)viewDidLoad
{
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a
        nib.
    self.navigationItem.leftBarButtonItem = self.editButtonItem;

    UIBarButtonItem *addButton = [[UIBarButtonItem alloc]
        initWithBarButtonSystemItem:UIBarButtonSystemItemAdd target:self
        action:@selector(insertNewObject:)];
    self.navigationItem.rightBarButtonItem = addButton;

    self.title = @"Student";

    myData = [[NSArray alloc] initWithObjects:@"Peter", @"Mary", @"Tom",
        @"Dicky", nil];


}
```

Initialize the array

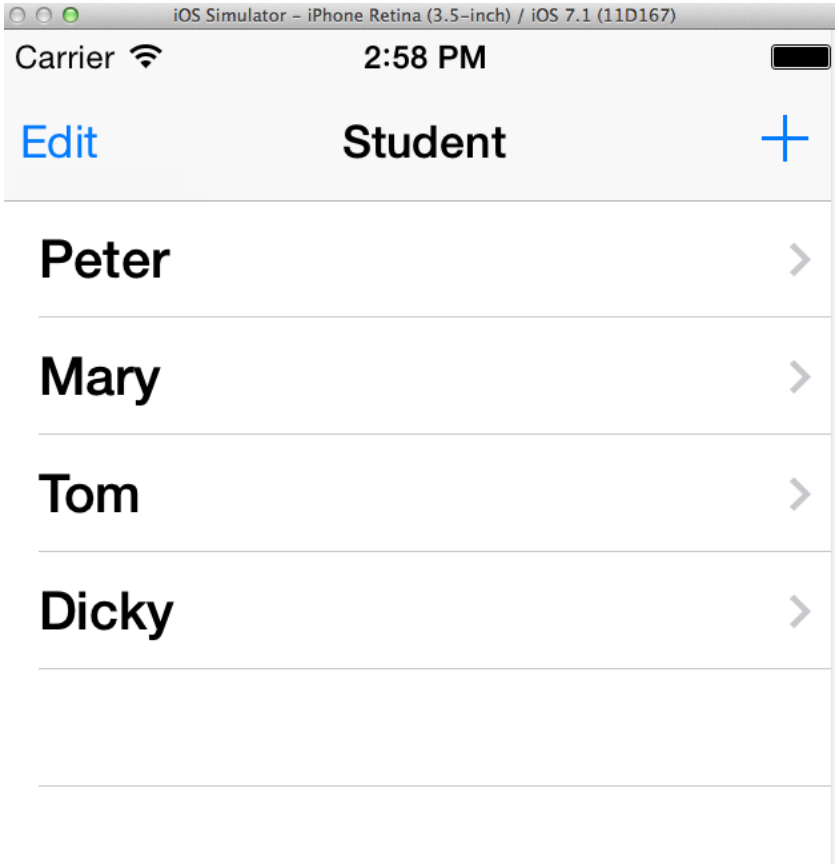
Master-Detail Application

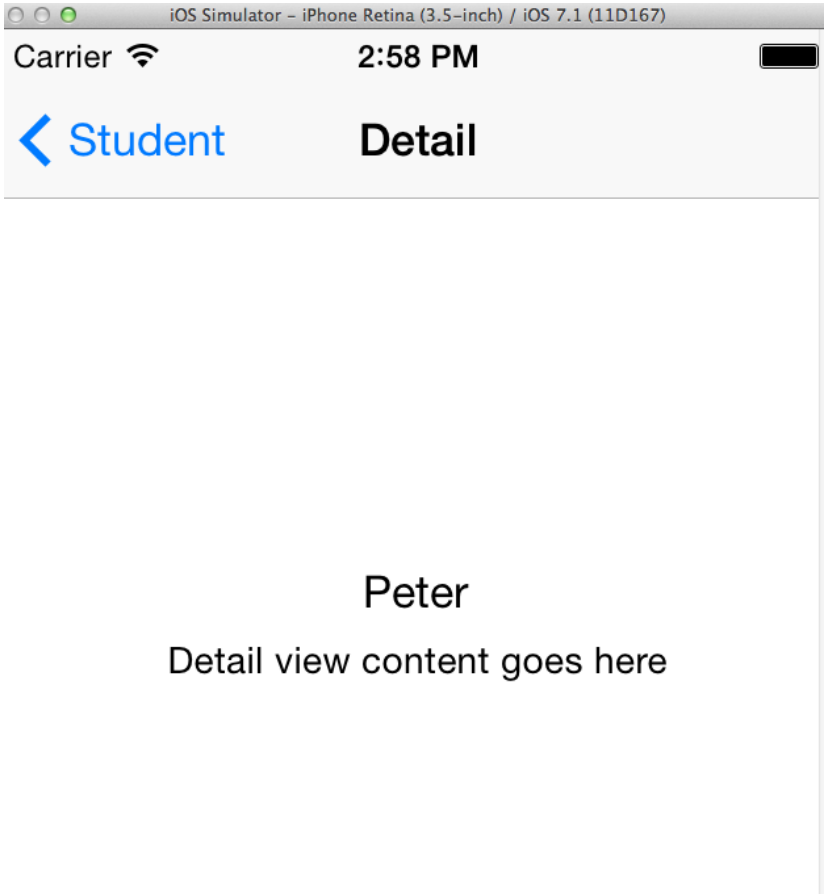
```
#pragma mark - Table View

- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
{
    return [[self.fetchedResultsController sections] count];
}

- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:
    (NSInteger)section
{
    return [myData count];
}

- (UITableViewCell *)tableView:(UITableView *)tableView
    cellForRowAtIndexPath:(NSIndexPath *)indexPath
{
    static NSString *CellIdentifier = @"myCell";

    UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:
        CellIdentifier];

    if(cell == nil) {

        cell = [[UITableViewCell alloc]initWithStyle:
            UITableViewCellStyleDefault reuseIdentifier:CellIdentifier];

    }

    cell.textLabel.text = [myData objectAtIndex:indexPath.row];


    return cell;
}
```

Return the number of rows

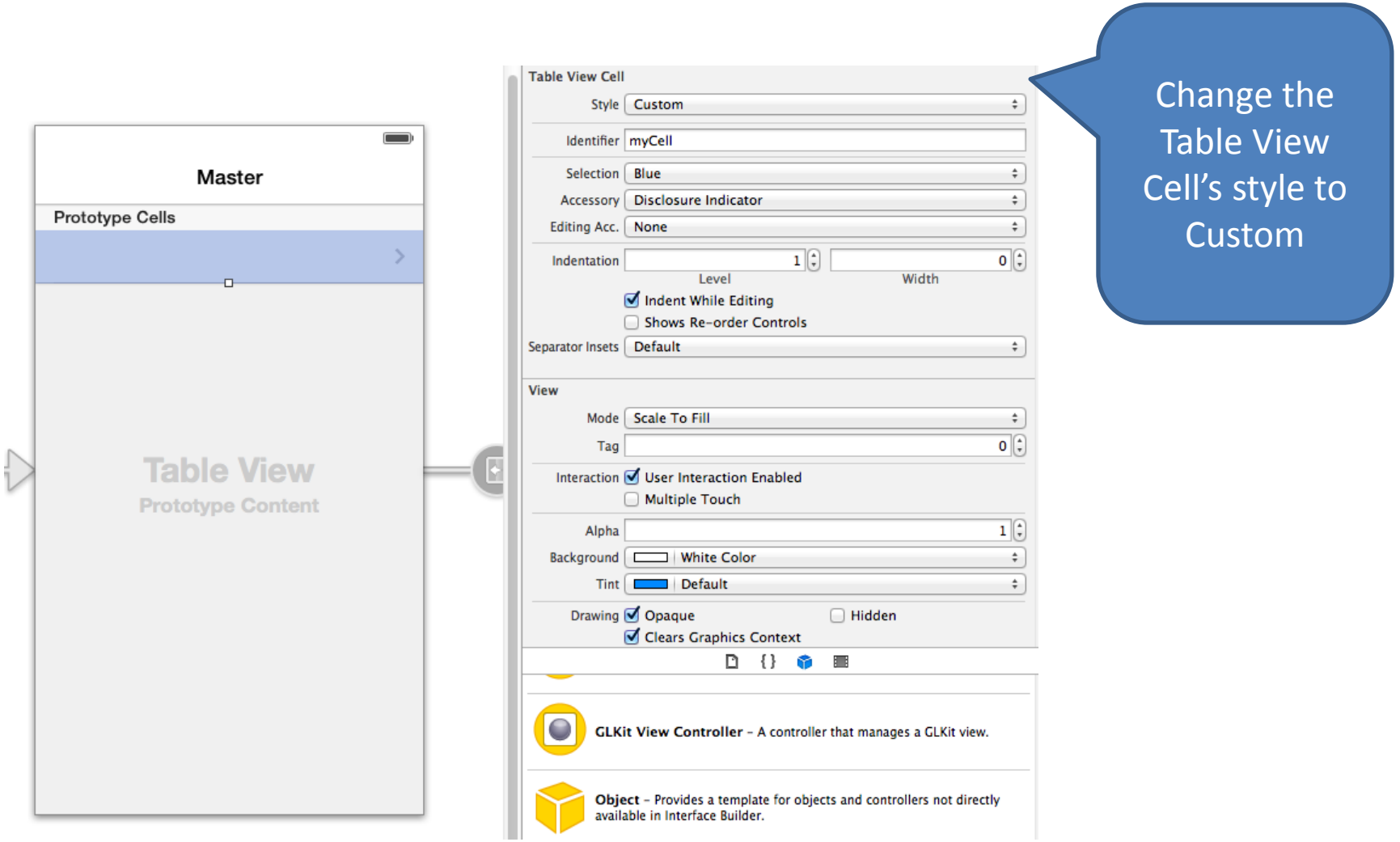
Return the cell content

Master-Detail Application



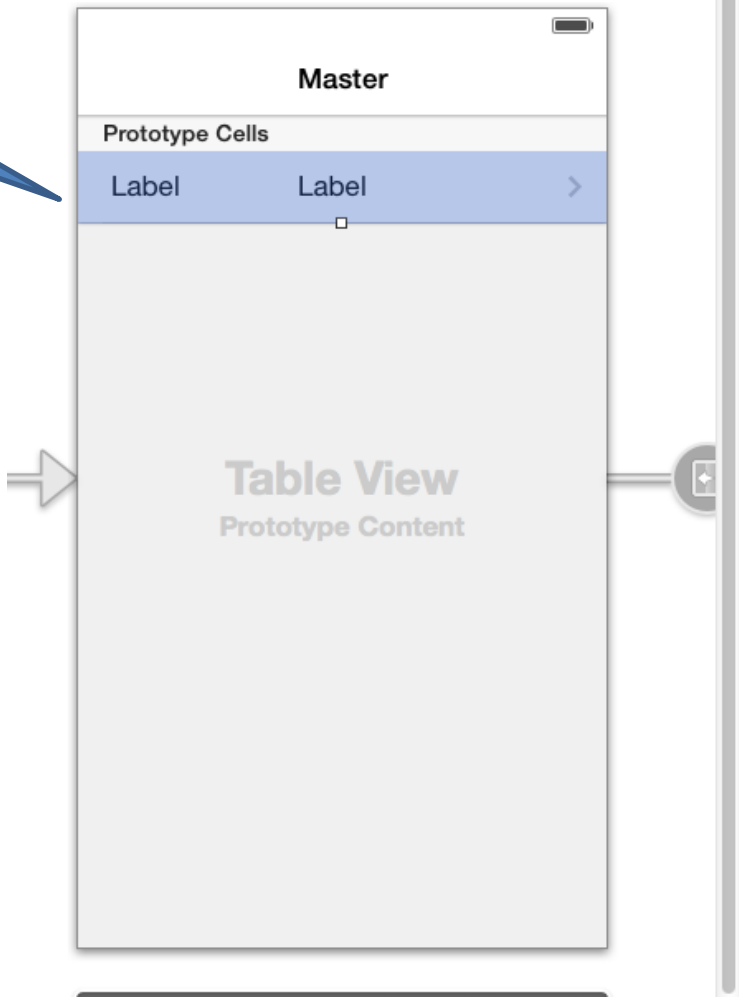


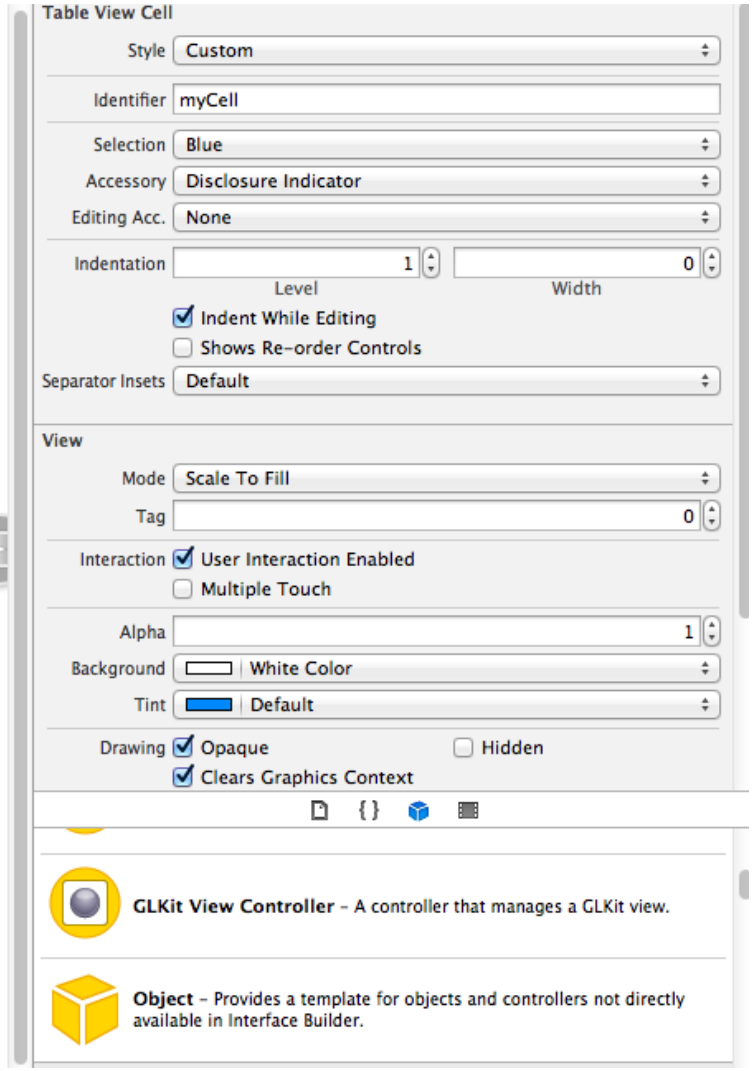
Customize the TableView Cell



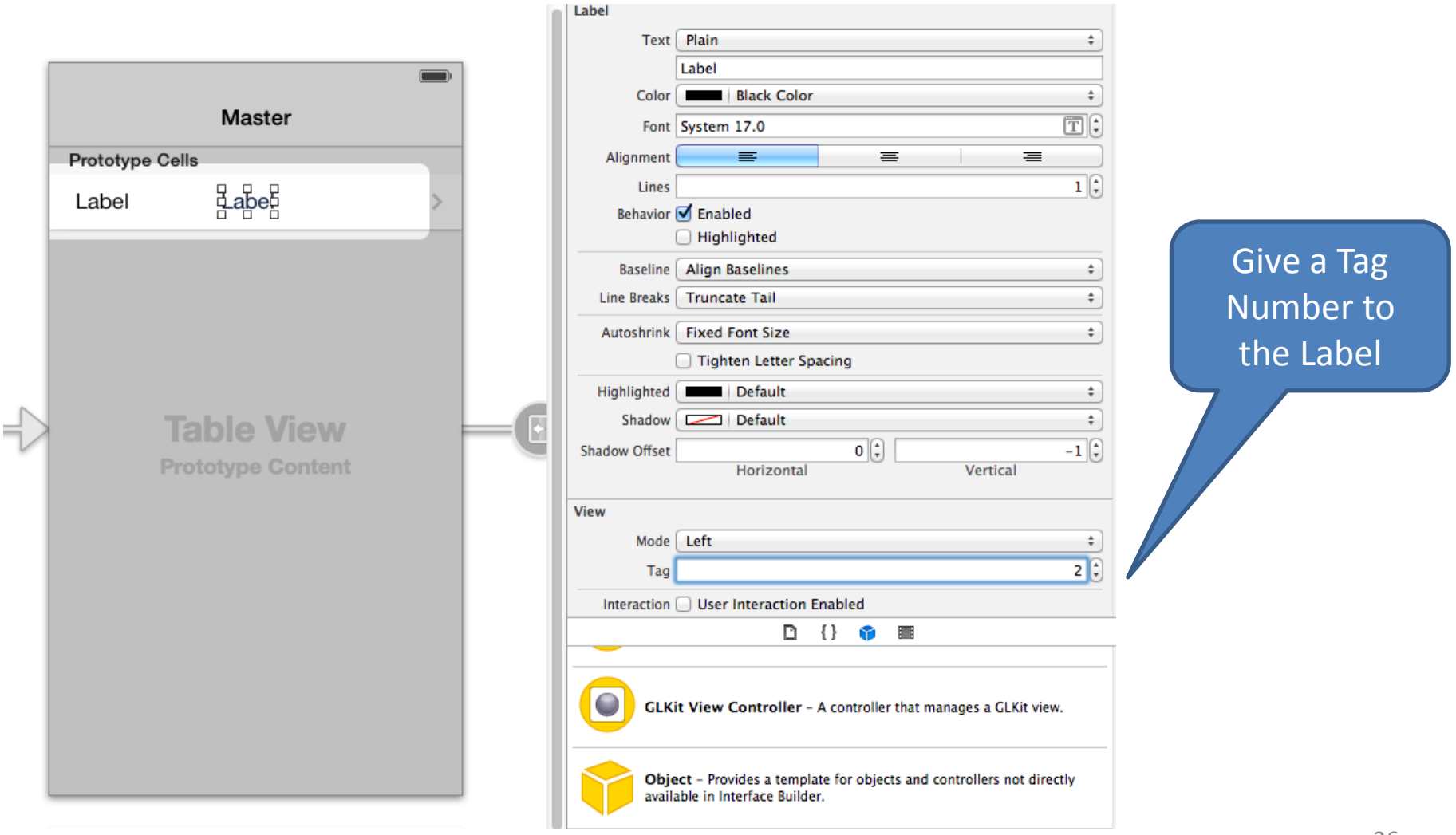
Customize the TableView Cell

Add the Label in the Cell





Customize the TableView Cell



Customize the TableView Cell

```
- (UITableViewCell *)tableView:(UITableView *)tableView
    cellForRowAtIndexPath:(NSIndexPath *)indexPath
{
    static NSString *CellIdentifier = @"myCell";

    UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:
        CellIdentifier];

    if(cell == nil) {

        cell = [[UITableViewCell alloc]initWithStyle:
            UITableViewCellStyleDefault reuseIdentifier:CellIdentifier];

    }

    UILabel *label1 = (UILabel *)[cell viewWithTag:1];

    UILabel *label2 = (UILabel *)[cell viewWithTag:2];

    label1.text = @"Label 1";

    label2.text = @"Label 2";


    return cell;
}
```

Update the Label Content for each cell