Praktikum Entwicklung von Mediensystemen mit
iOS

Wintersemester 2012 / 2013

Prof. Heinrich Hußmann, Dr. Alexander De Luca, Fabius Steinberger

24.10.2012

Today

- Table View

- Navigation Controller

- Passing Data Between Scenes

- Tips & Tricks

- Your Favorite Apps

- Assignment 2

Navigation-based Apps (iPhone)

- **Master View Controller:** A Table View displays a list of table rows. Navigate to the Detail View by selecting a table row.

- **Detail View Controller:** Custom content. Navigate to Master by tapping the Back Button

Table View

Top Bar   Custom content   Back button
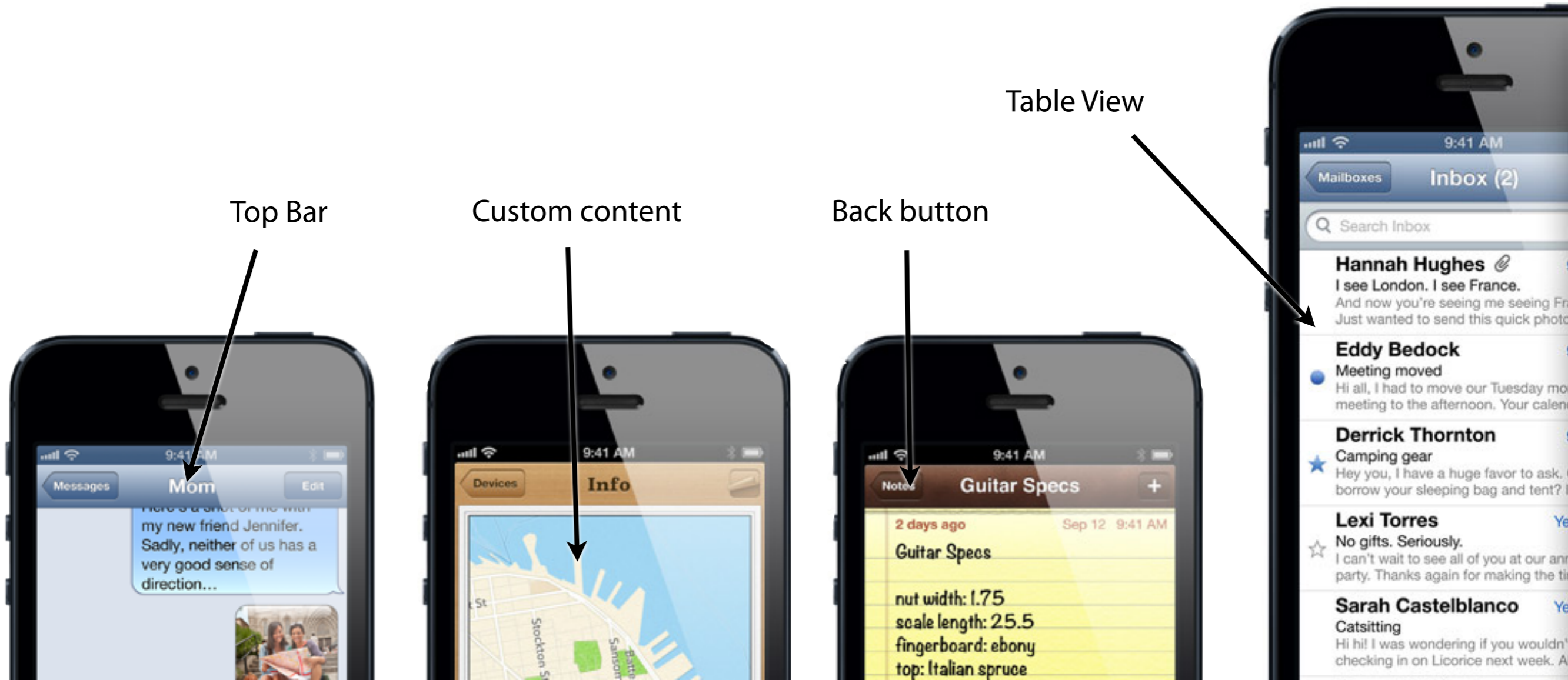
Navigation-based Apps (iPad)

- Split View: Master View and Detail View fit on one screen

Hello Table View

- Create a new XCode Project ("Single View Application") for iPhone. (Use Storyboards and ARC)

- Change the base class of ViewController to UITableViewController and make it UITableViewDelegate and UITableViewDataSource. Add an array in .h and fill it with data in viewDidLoad in .m:

```
@interface ViewController : UITableViewController<UITableViewDelegate, UITableViewDataSource>
@property(strong, nonatomic) NSArray* tableEntries;

self.tableEntries = @[@"Blur", @"Beatles", @"Stone Roses", @"Oasis", @"Velvet Underground"];
```

- In the Storyboard, delete the scene and create a new Table View Controller. Change its class to ViewController.

- Select the Table View Cell and change its Identifier to "Cell".
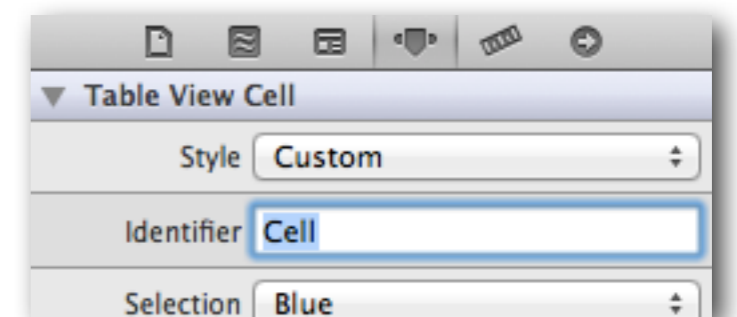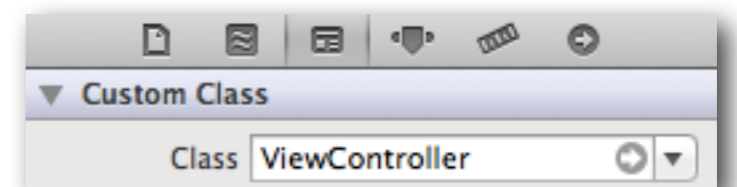
Hello Table View

- Our table rows need to be filled with the data stored in our array. For this, implement the following methods of the Table View Data Source and Table View Delegate protocols:

```objc
// number of sections in table
- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView {
    return 1;
}

// number of rows in our section
-(NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section {
    return [self.tableEntries count];
}

// fill table rows with content
- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath {

    UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:@"Cell" forIndexPath:indexPath];

    cell.textLabel.text = [self.tableEntries objectAtIndex:indexPath.row];
    return cell;
}
```

Position of array entry
= position of cell in Table View
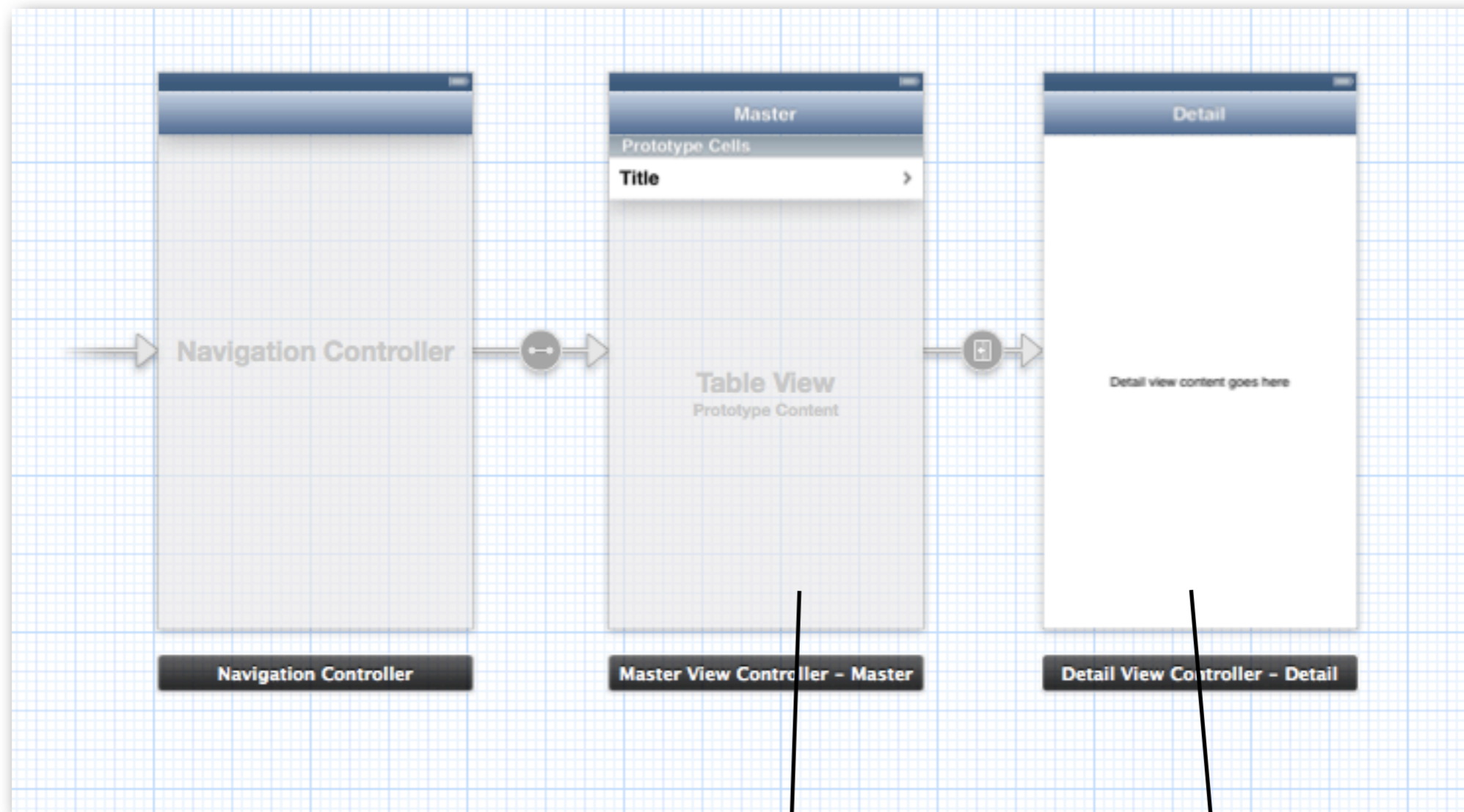
Hello Table View

- Handle row selection:

```objc
// handle user interaction (i.e. row selection)
- (void)tableView:(UITableView *)tableView didSelectRowAtIndexPath:(NSIndexPath *)indexPath
{
    UIAlertView *alert = [[UIAlertView alloc] initWithTitle:[self.tableEntries objectAtIndex:indexPath.row]
                                    message:nil
                                   delegate:nil
                          cancelButtonTitle:@"OK"
                          otherButtonTitles:nil, nil];

    [alert show];
}
```
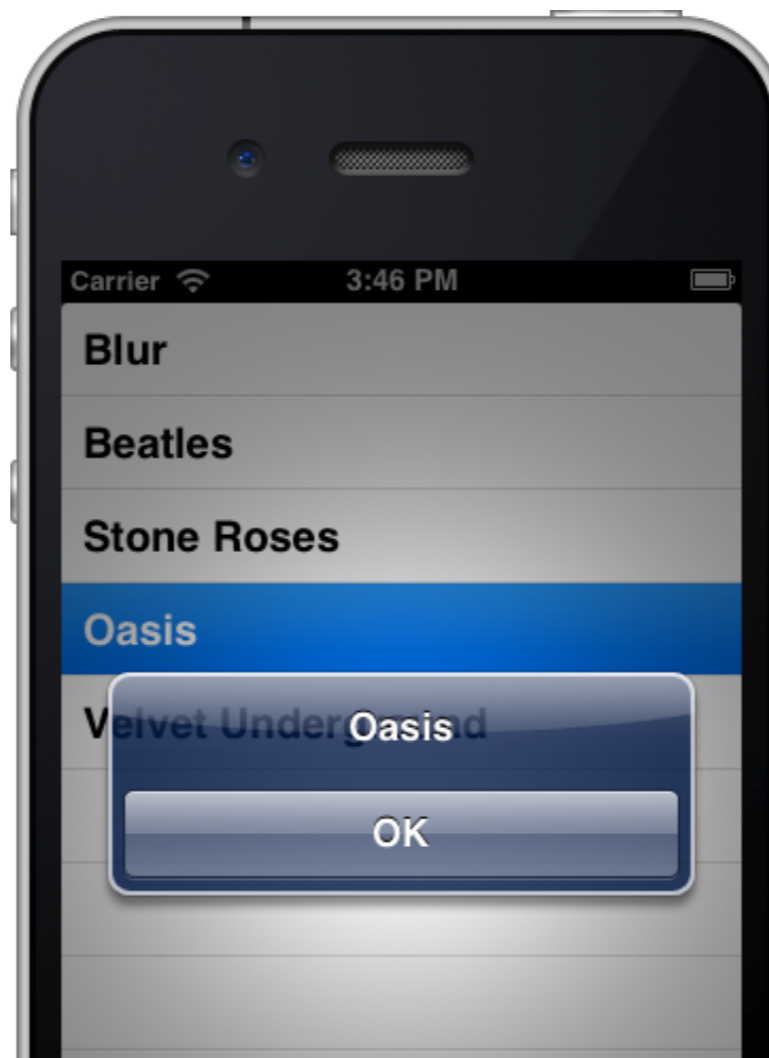
Hello Navigation Controller

- Create a new XCode Project ("Master Detail Application"). (Use Storyboards and ARC)


```
@interface
MasterViewController :
UITableViewController
```

```
@interface
DetailViewController :
UIViewController
```

Hello Navigation Controller

- Master View Controller: "+" button in Top Bar created in viewDidLoad:

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

self.navigationItem.rightBarButtonItem = addButton;
```
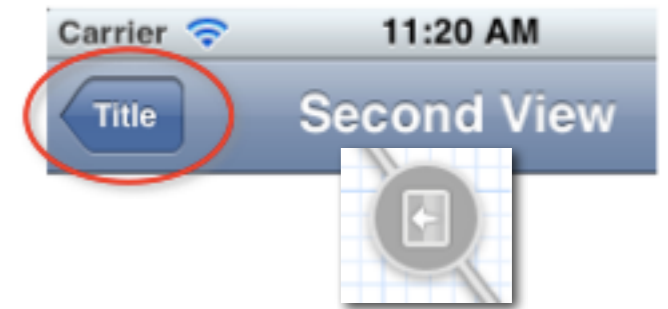
  - `target:self` which object receives the action?

  - `action:@selector(insertNewObject:)` what's the action? (this calls a method "insertNewObject:")

- Row selection by the user: The Detail View Controller is pushed as defined by the Storyboard Segue:
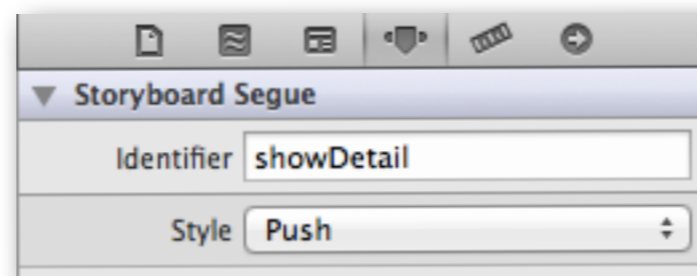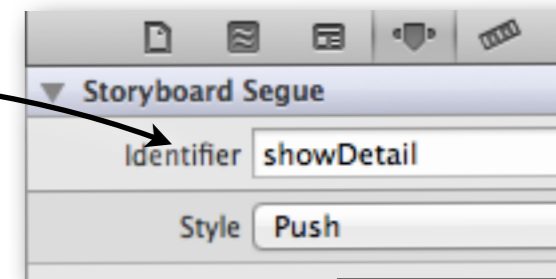
Passing Data Between Scenes (Push)

- For Push Segues: Update data in prepareForSegue.

  - In Master View Controller:

```
- (void)prepareForSegue:(UIStoryboardSegue *)segue sender:(id)sender
{
    if ([[segue identifier] isEqualToString:@"showDetail"]) {
        NSIndexPath *indexPath = [self.tableView indexPathForSelectedRow];
        NSDate *object = _objects[indexPath.row];
        [[segue destinationViewController] setDetailItem:object];
    }
}
```
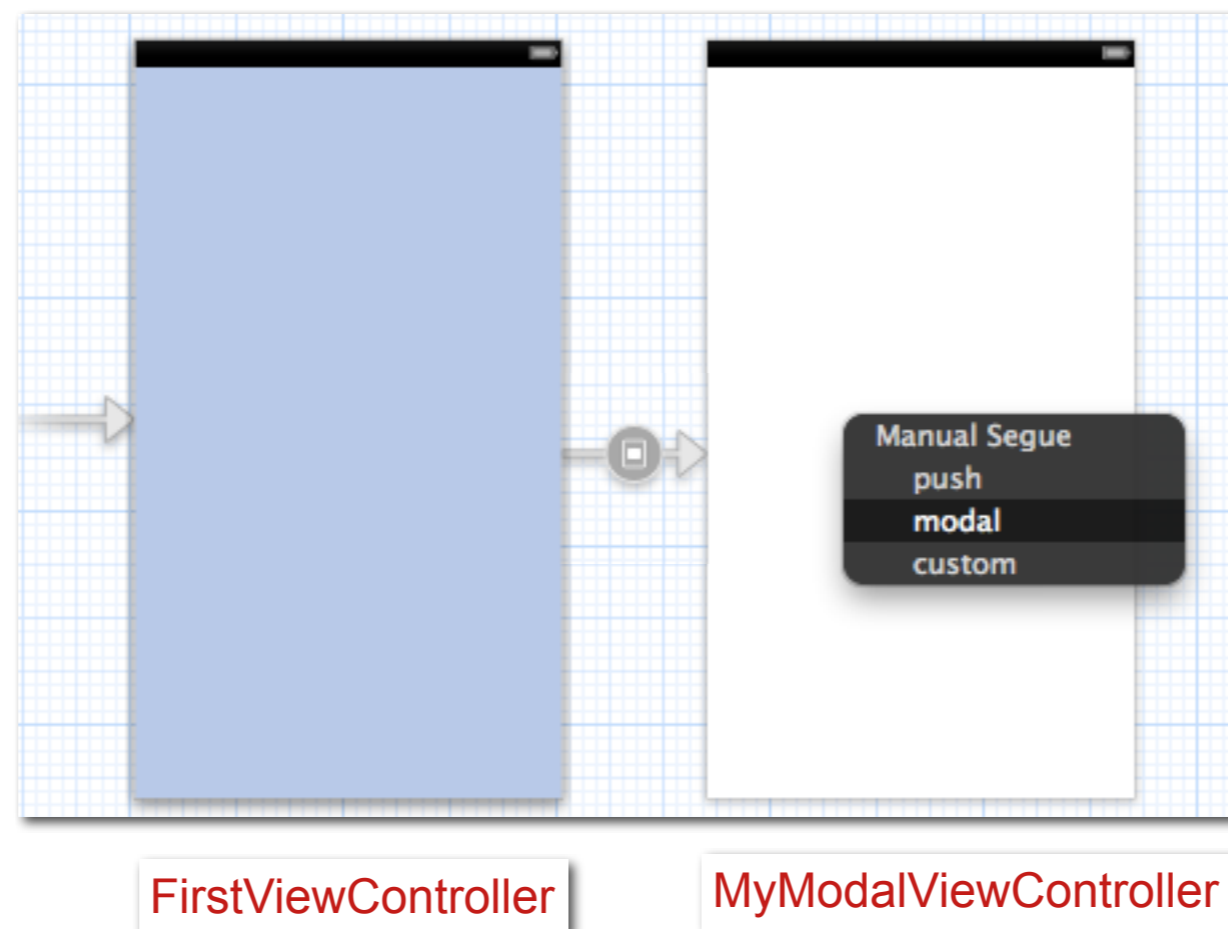
  - In Detail View Controller:

```
- (void)setDetailItem:(id)newDetailItem
{
    // Update the textlabel
}
```

- Push Segues add View Controllers on the Navigation Stack. Based on user interaction, the View Controllers are pushed or popped.

Passing Data Between Scenes (Modal)

- Modal Segues are used outside the context of Navigation Controllers.

- In the following example, My Modal View Controller is modally presented by First View Controller. First View Controller acts like a parent and is responsible for dismissing My Model View Controller.

FirstViewController   MyModalViewController

Passing Data Between Scenes (Modal)

- For Modal Segues: Use a delegate that processes user interaction (e.g. Done/Cancel) and dismisses the View Controller.
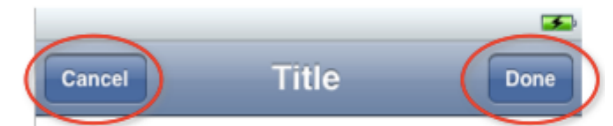
  MyModalViewController.h

  - Create delegate by defining a protocol:

    ```
    @protocol MyModalViewControllerDelegate
        - (void) myModalViewControllerDidCancel:(MyModalViewController*)controller;
        - (void) myModalViewControllerDidSave:(MyModalViewController*)controller;
    @end
    ```

  - Create a property for the delegate:

    MyModalViewController.h

    ```
    @interface MyModalViewController : UIViewController
        @property (nonatomic, strong) id <MyModalViewControllerDelegate> delegate;
        - (IBAction)cancel:(id)sender;
        - (IBAction)done:(id)sender;
    @end
    ```

  - Call the delegate on user interaction:

    ```
    - (IBAction)cancel:(id)sender {
        [self.delegate myModalViewControllerDidCancel:self];
    }
    - (IBAction)done:(id)sender {
        [self.delegate myModalViewControllerDidSave:self];
    }
    ```

    MyModalViewController.m

Passing Data Between Scenes (Modal)

- First View Controller implements our delegate protocol:

```
#import "MyModalViewController.h"
@interface FirstViewController : UIViewController <MyModalViewControllerDelegate>
@end
```

- Set First View Controller as delegate of the My Modal View Controller:

```
- (void) prepareForSegue:(UIStoryboardSegue*)segue sender:(id)sender {
    if ([[segue identifier] isEqualToString:@"FirstViewController_To_MyModelViewController"]) {
        MyModalViewController *myModalViewController = [segue destinationViewController];
        myModalViewController.delegate = self;
    }
}
```

- Handle user interaction (dismiss My Modal View Controller):

```
- (void) myModalViewControllerDidCancel:(MyModalViewController*)controller {
    [controller dismissModalViewControllerAnimated:YES];
}

- (void) myModalViewControllerDidSave:(MyModalViewController*)controller{
    [controller dismissModalViewControllerAnimated:YES];
}
```

Passing Data Between Scenes (Global)

- Use the Application Delegate for global data:

```
#import "AppDelegate.h"

AppDelegate *appDelegate = (AppDelegate*) [UIApplication sharedApplication].delegate;
NSArray *data = appDelegate.data;
```

- sharedApplication is a singleton (i.e. an instance that exists only once) that can be accessed from anywhere within the application.

Top 3 Resources

1 iOS Developer Library

Organizer   Documentation   or https://developer.apple.com/library/ios
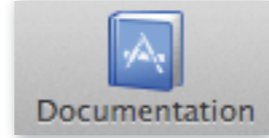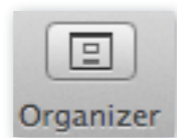
2 RAYWENDERLICH
Tutorials for iPhone / iOS Developers and Gamers

http://www.raywenderlich.com/tutorials

3 stackoverflow

Google   stackoverflow ios app delegate

Assignment 2

- Individual assignment

- Navigation-based app

- Due next Wednesday 12:00, upload to Uniworx

- Questions?