Lecture 6

Segues

Announcements

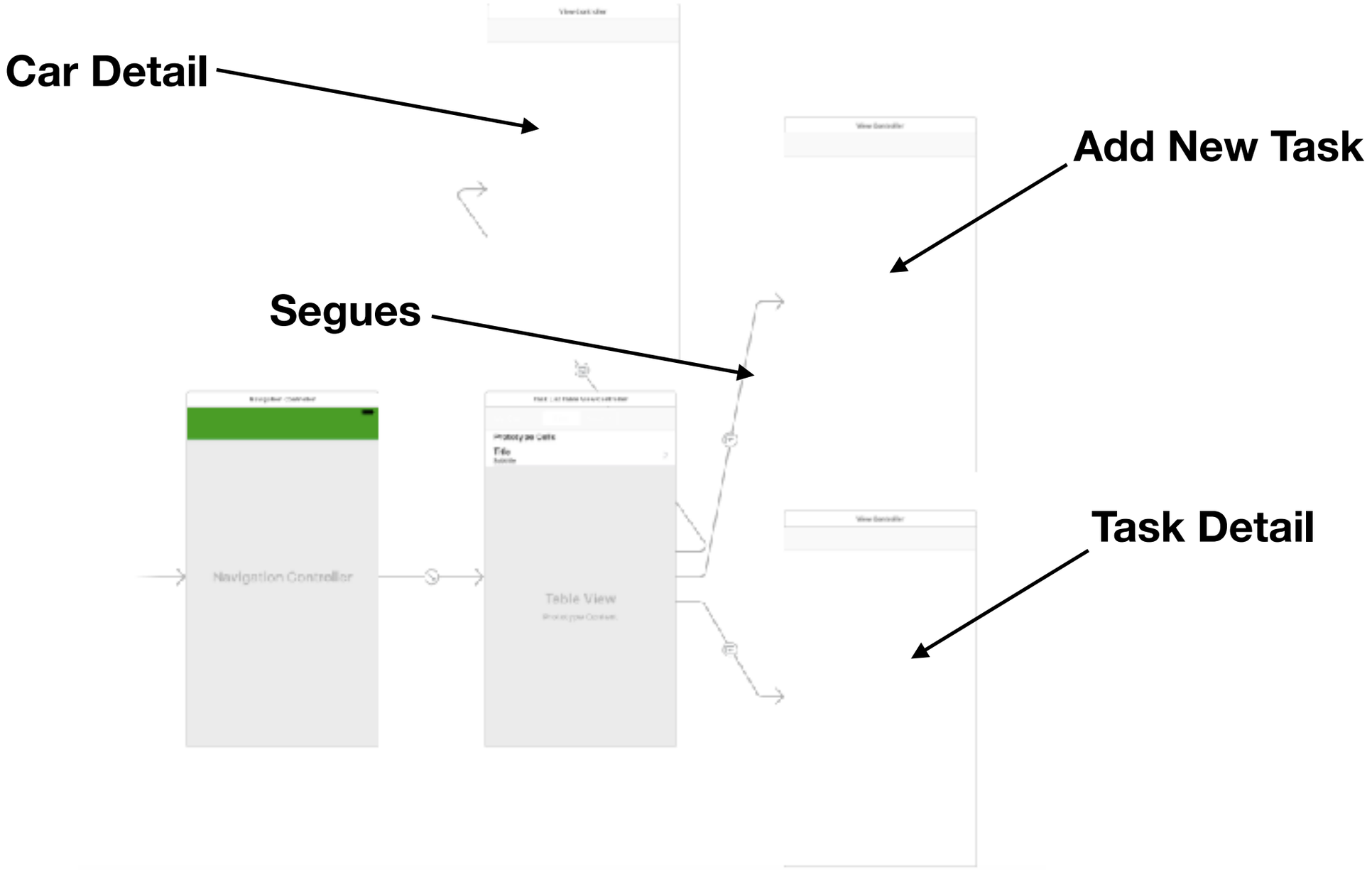
- Test on Workshops - Tuesday 27th Feb 1-2 **Brookfield** G. 01

- Test on material presented at lectures Thursday 8th March 1-2 **Brookfield** G.10

- Assignment for iOS component will be issued over the weekend

Segues

- We can now retrieve data from the server and display it in our Task List. But we must add supporting screens to show car details, add new tasks and show task details.



Segues

- A segues defines a transition between two view controllers in our storyboard
- Segues have a start point and an end point. The start point can be some control:
    - Buttons, table cell, or other controls, or can be triggered manually.
- The end point is always another view controller
- Segues can be configured to have a name and a type:
    - The name allows us intercept the segue programatically
    - The type determines how the transition will be handled
- We can use segues as portals between view controllers, allowing us to configure the destination, for example, by passing information from source to destination
- More at https://developer.apple.com/library/content/featuredarticles/ViewControllerPGforiPhoneOS/UsingSegues.html

Segue Types

Segue type	Behavior
Show (Push)	This segue displays the new content using the `showViewController:sender:` method of the target view controller. For most view controllers, this segue presents the new content modally over the source view controller. Some view controllers specifically override the method and use it to implement different behaviors. For example, a navigation controller pushes the new view controller onto its navigation stack. UIKit uses the `targetViewControllerForAction:sender:` method to locate the source view controller.
Show Detail (Replace)	This segue displays the new content using the `showDetailViewController:sender:` method of the target view controller. This segue is relevant only for view controllers embedded inside a `UISplitViewController` object. With this segue, a split view controller replaces its second child view controller (the detail controller) with the new content. Most other view controllers present the new content modally. UIKit uses the `targetViewControllerForAction:sender:` method to locate the source view controller.
Present Modally	This segue displays the view controller modally using the specified presentation and transition styles. The view controller that defines the appropriate presentation context handles the actual presentation.
Present as Popover	In a horizontally regular environment, the view controller appears in a popover. In a horizontally compact environment, the view controller is displayed using a full-screen modal presentation.

Creating a Segue

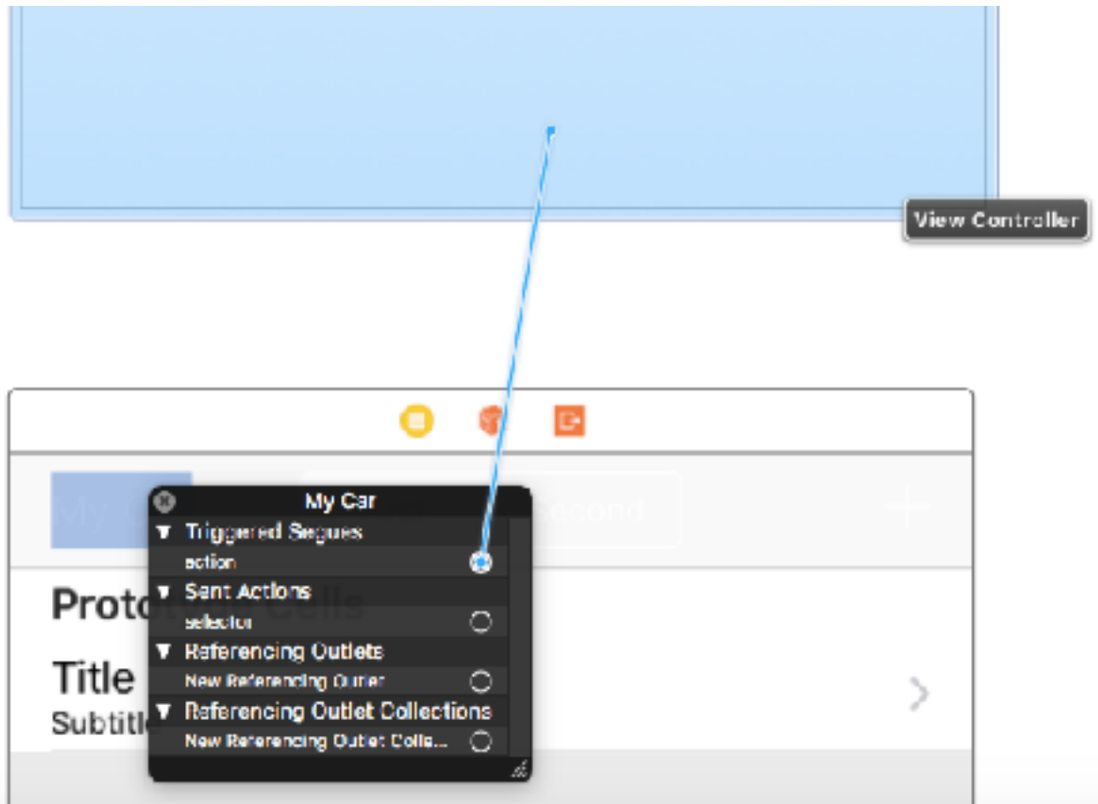
1. Right click on the storyboard element that you wish to segue from

2. From the segue option, drag to the control

3. Select segue type

4. Configure segue

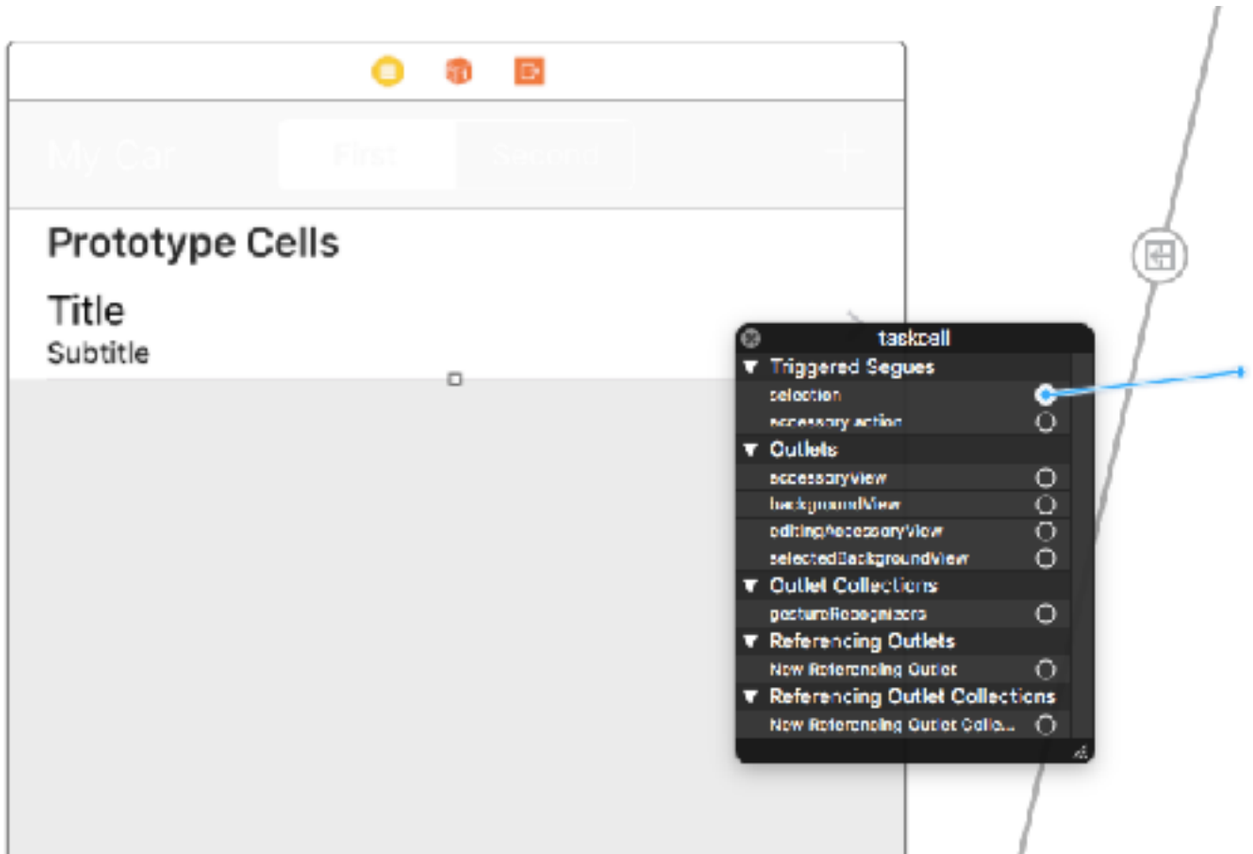
1.



2.

Segue from Table

- We can invoke two separate segues form a table cell:
  - From the cell itself (selection option)
  - From the cell accessory
- These are configured in the same way as before and invoked when the row or accessory is selected.

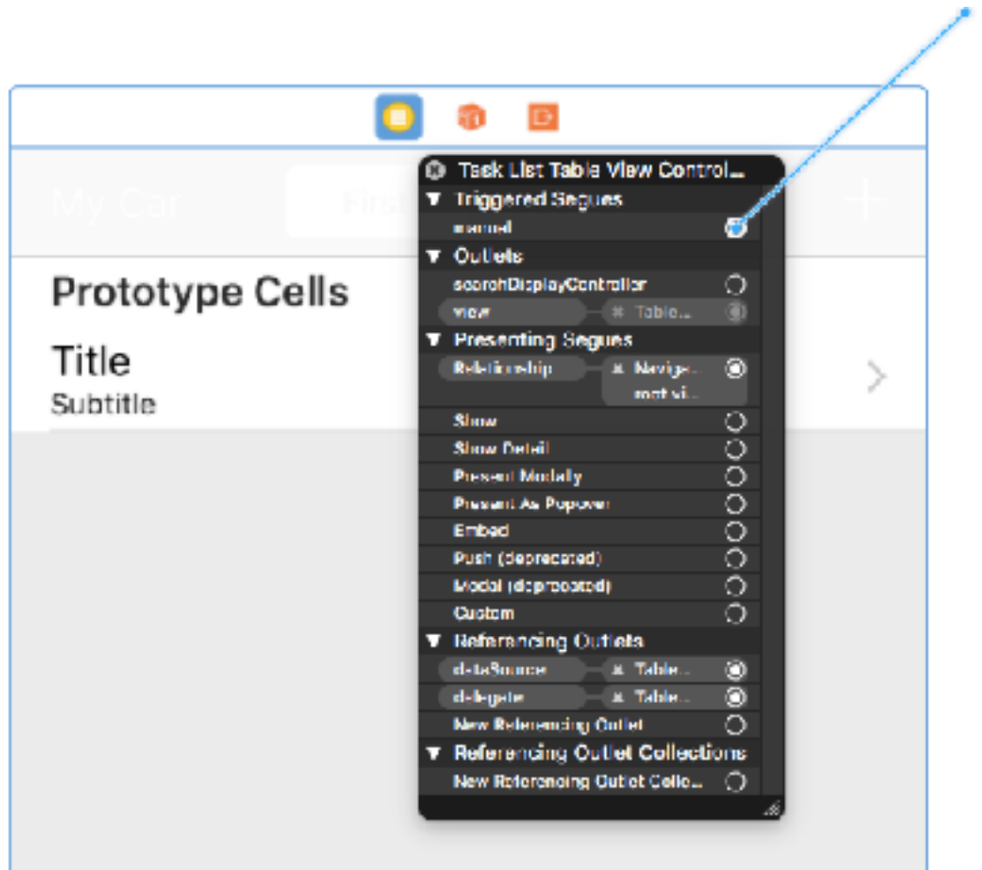


Manual Segue

- Occasionally we may want to have a general segue between two view controllers. These are not associated with a control, but can be invoked programatically.

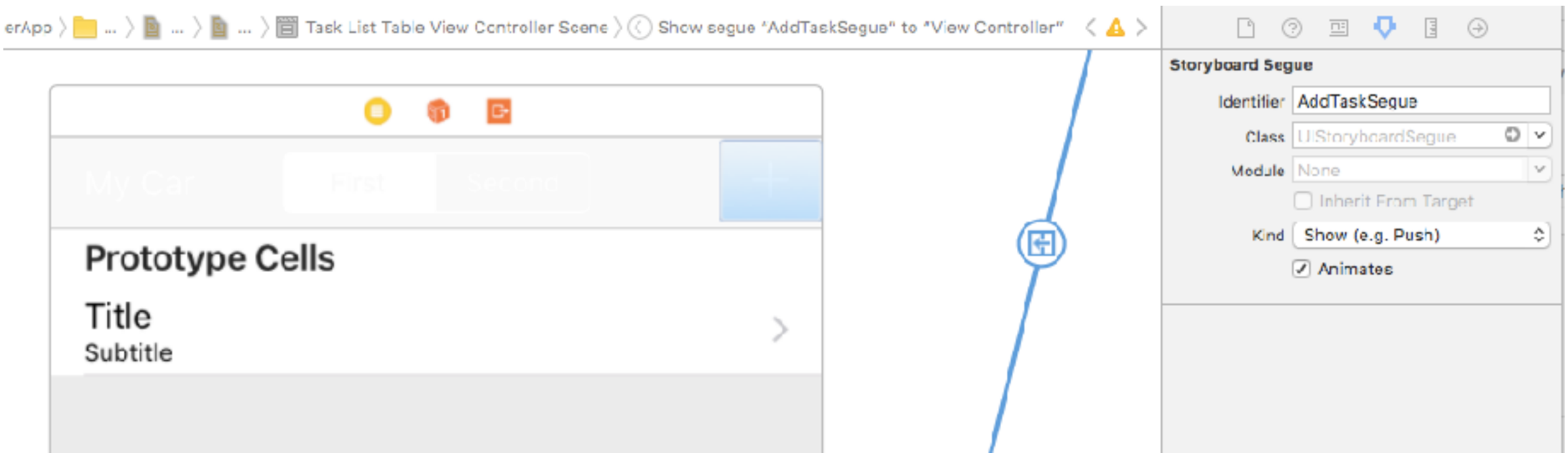
  - This allows logic in the view controller to determine when the segue is performed.

  - To invoke, select the view controller and select manual segue in the usual way

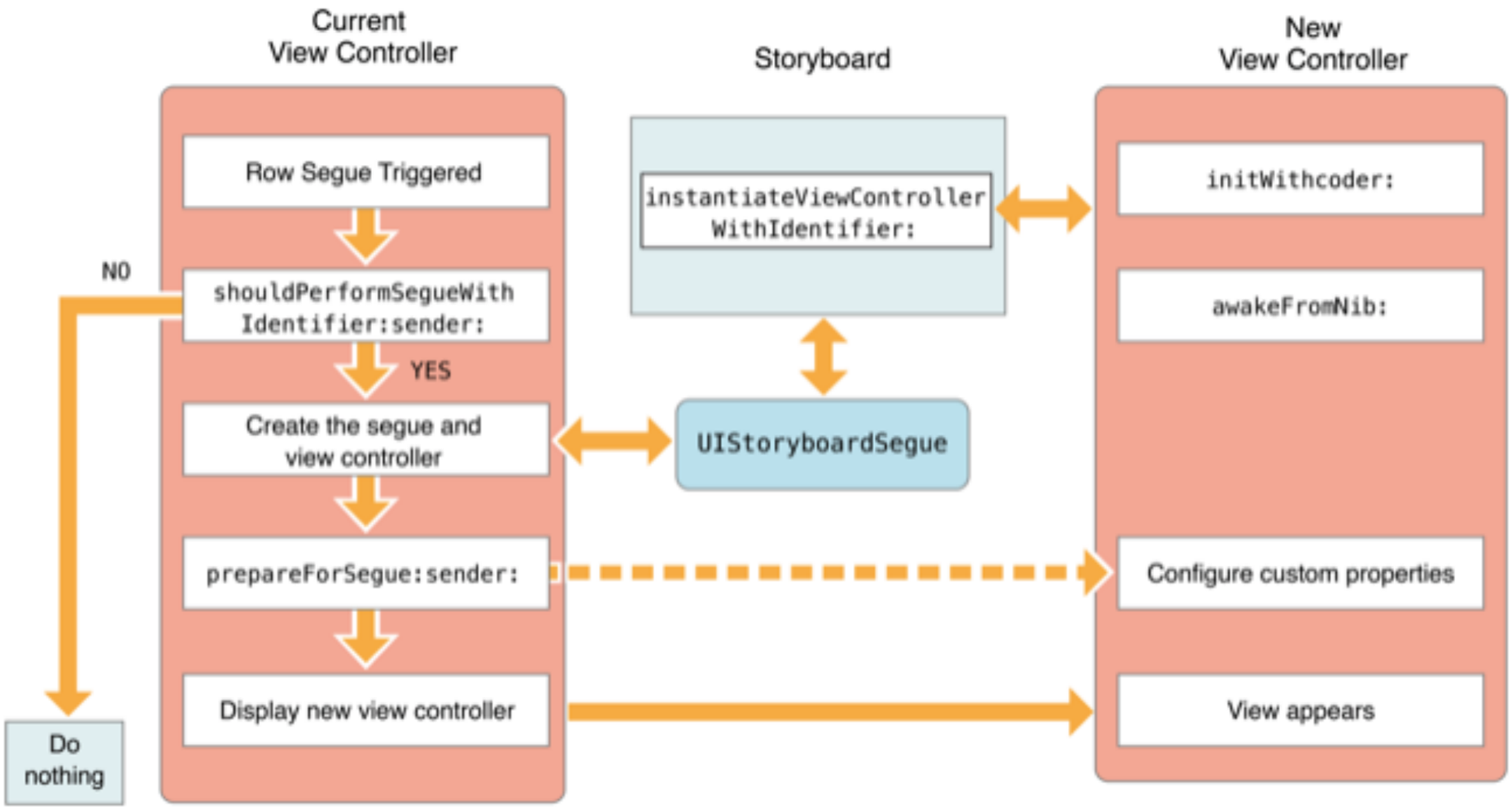


Configuring the Segue

- We can configure the segue by selecting it and examining its attribute inspector.
    - We can change the segue's type here (though usually we will have selected correctly at creation type)
    - Most importantly we can give the segue a name to allow us access it programatically.



Lifecycle of a Segue



Programatically Invoking a Segue

- Segues can be invoked by interaction with a control (pressing a button, selecting a row in a table)

- We can also programatically invoke a segue using the following code:

```
performSegue(withIdentifier: "SomeName", sender: self)
```

Passing Information Across a Segue

- Once we reach the performSegue method, we can intercept the segue and use it to configure the destination.

    - Assume we have created a ViewController for the detail view

    - Also assume that we have added a variable ‘task’ to this view controller

```
import UIKit

class MyNextViewController:UIViewController{

    var somevalue = 0
```

Passing Information Across a Segue

- Back in the TaskTableViewController, we can now intercept our segue and configure the task value at the destination

```
override fun prepare(for segue: UIStoryboardSegue, sender: Any?){
    if segue.identifier == “myfirstsegue”{
       //Do something for this segue
    }
    else if segue.identifier == “mysecondsegue”{
        let destination = segue.destination as! MyNextViewController
        destination.somevalue = 100
    }
    else….
}
```