

iOS Apprentice Review 2

Part 2: Demo Instructions

Copyright © 2014 Razeware LLC.

All rights reserved. No part of this book or corresponding materials (such as text, images, or source code) may be reproduced or distributed by any means without prior written per- mission of the copyright owner.

This book and all corresponding materials (such as source code) are provided on an "as is" basis, without warranty of any kind, express or implied, including but not limited to the warranties of merchantability, fitness for a particular purpose, and noninfringement. In no event shall the authors or copyright holders be liable for any claim, damages or other liability, whether in action of contract, tort or otherwise, arising from, out of or in connec- tion with the software or the use or other dealings in the software.

All trademarks and registered trademarks appearing in this book are the property of their respective owners.

# Part 1: Creating the Model

In this demo, you will convert the Story Time app to use to use MVC as well as create additional view controllers

## Step 1: Create the Story File

With the project StoryTime project open, select **File / Swift File**. Give it the name **Story.swift**

## Step 2: Define the Story types

At the top of the file, add the following code to designate the two types of stories. Place this at the top of the file

enum StoryType {

case zombies, vampires

}

## Step 3: Create the Story class

Create the class to contain the story object.

class Story {

var title : String

var name : String?

var verb : String?

var number : Int?

var winStory: String

var loseStory: String

var type: StoryType

}

## Step 4: Add the initializer

Add init() just underneath all of the properties. Compiler errors, begone!

init(title: String, winStory:String, loseStory: String, type:

StoryType) {

self.title = title

self.winStory = winStory

self.loseStory = loseStory

self.type = type

}

## Step 5: Create a private helper method.

Generating a story requires text replacement so provide a quick helper method to do this.

private func replaceText(needle: String, haystack: String, text: String) -> String {

return text.stringByReplacingOccurrencesOfString(needle, withString: haystack, options: NSStringCompareOptions.LiteralSearch, range: nil)

}

## Step 6: Write the story generator method.

func generateStory(monstersWin: Bool) -> String {

var storyText = loseStory

if monstersWin {

storyText = winStory

}

var monsters = "zombies"

if type == .vampires {

monsters = "vampires"

}

if verb != nil {

storyText = replaceText("<verb>", haystack: verb!, text: storyText)

}

if number != nil {

storyText = replaceText("<number>", haystack: String(number!), text: storyText)

}

if name != nil {

storyText = replaceText("<name>", haystack: name!, text: storyText)

}

storyText = replaceText("<monsters>", haystack: monsters, text: storyText)

return storyText

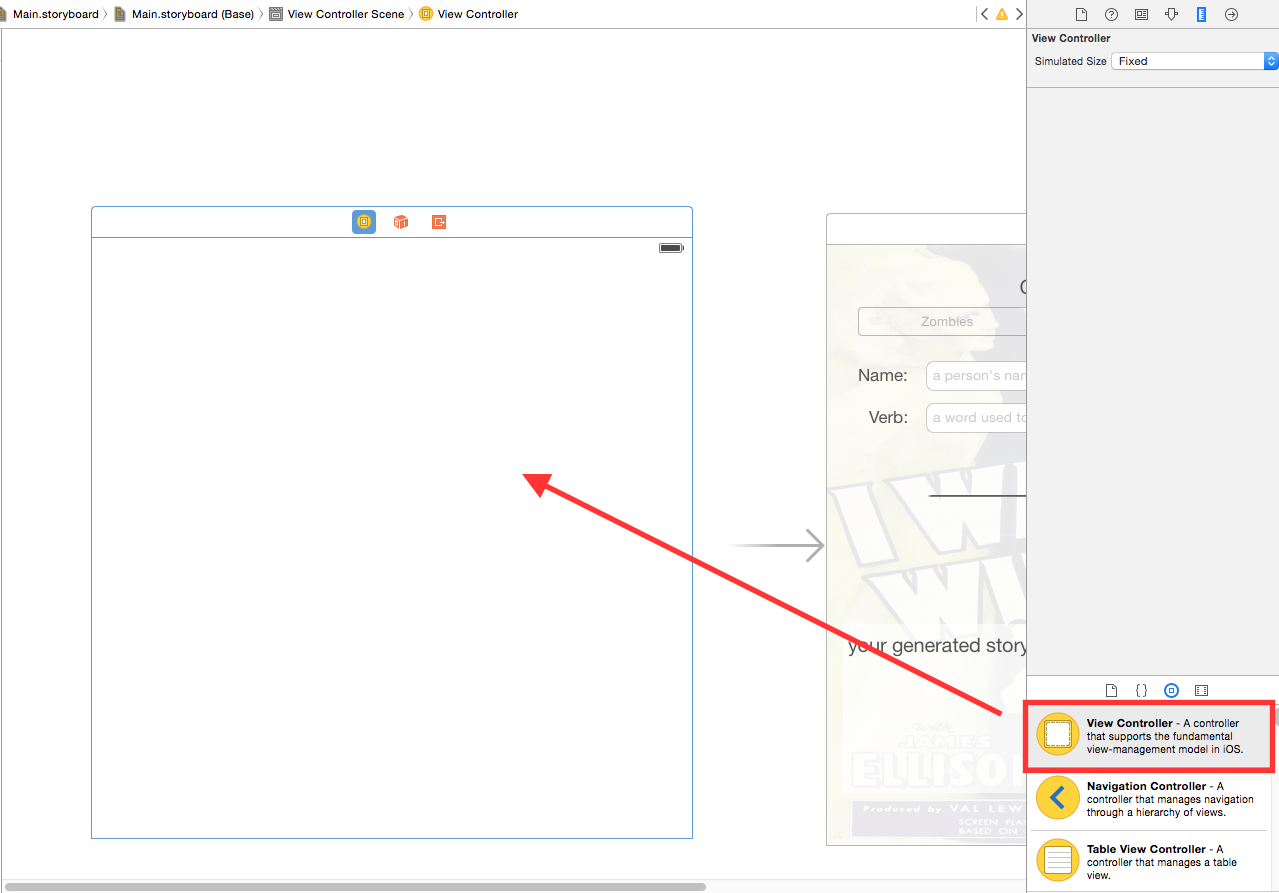
}

# Part 2: Updating the View

With the model set up, it’s time to update the view to add an additional view controller to list all the stories

## Step 1: Add a New View Controller

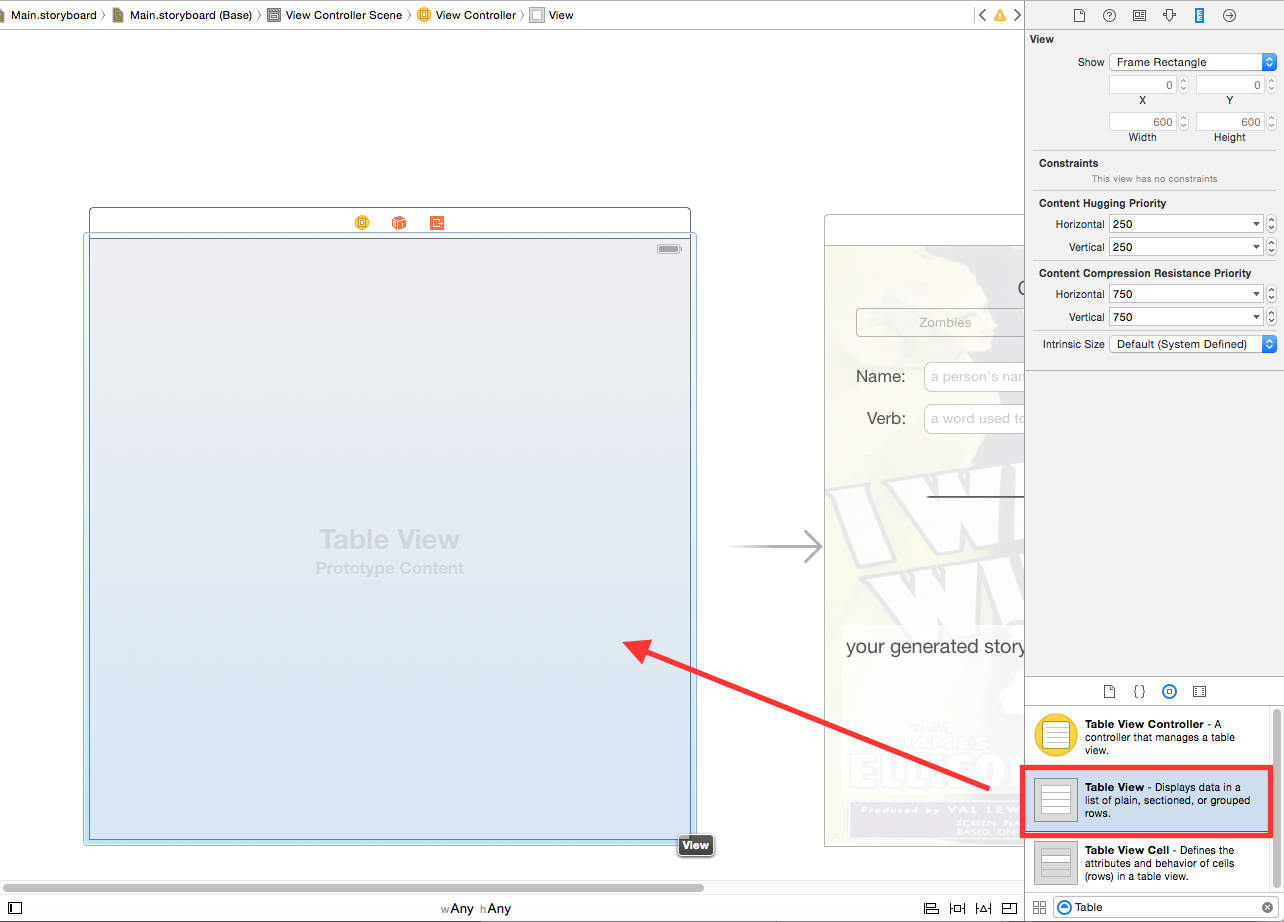
In your view controller, open **Main.Storyboard.** In your Object Library, find a View Controller and drag it onto the scene.



## Step 2: Add a Table View onto the Controller

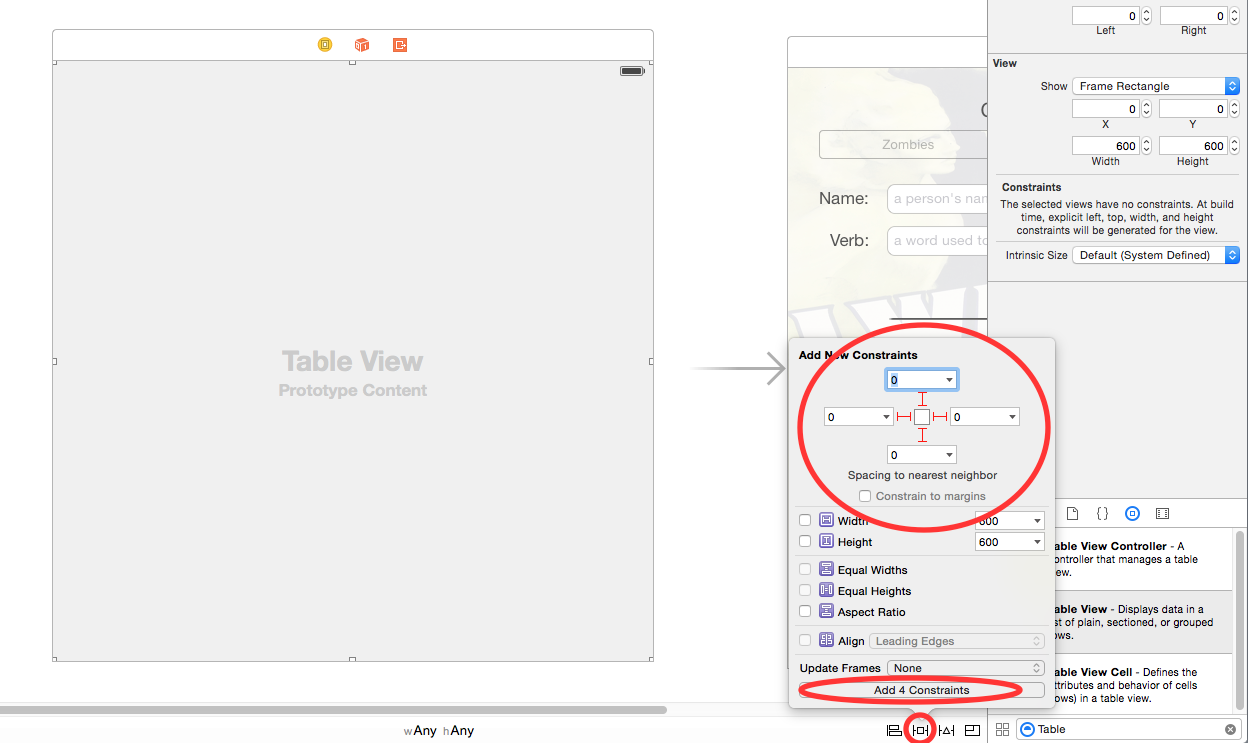
While you can always use a Table View Controller to do the same thing, it’s just as important to understand how to configure Table Views.

Drag a Table View from the Object Library onto your new View Contrioller.



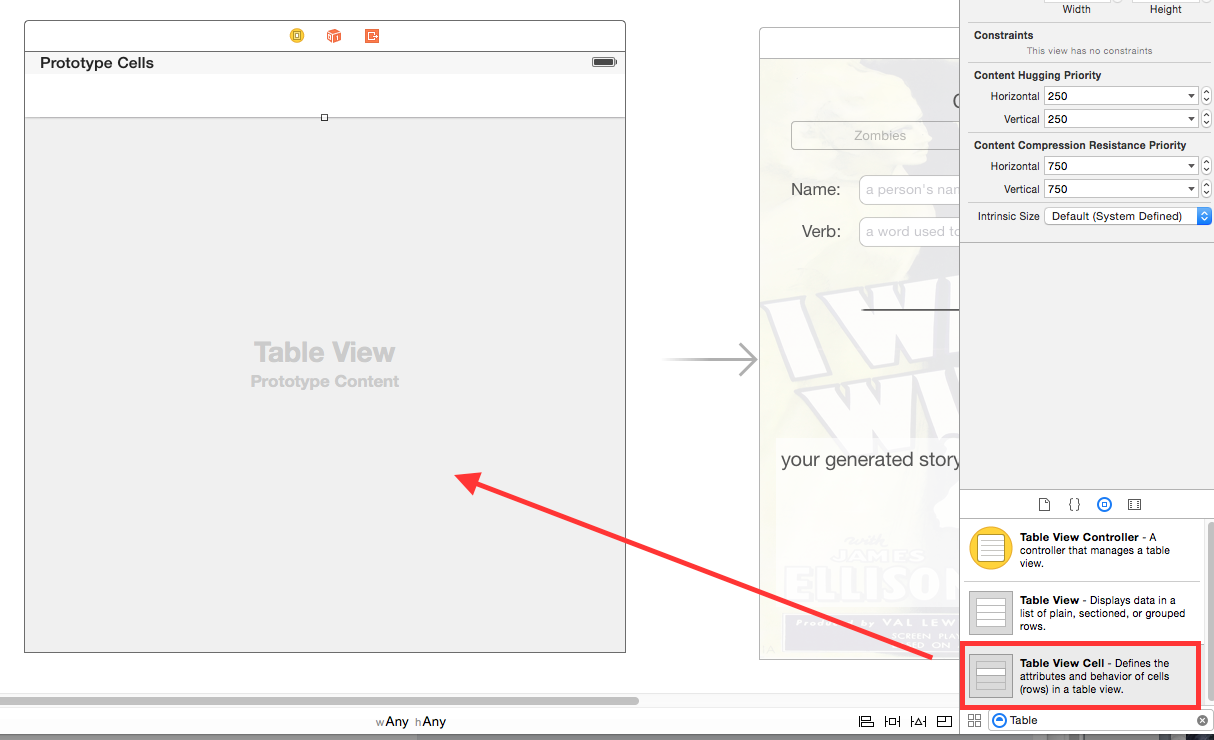
## Step 3: Add the Constraints

Select the Table View, then click the **Pin** button. Make sure to **uncheck** “**Constrain to margins**” and select all the neighboring constraints. Set them to **0**.



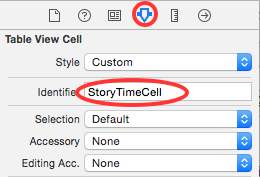
## Step 4: Add a Cell

From the Object Library, drag a Table View Cell into the Table View.

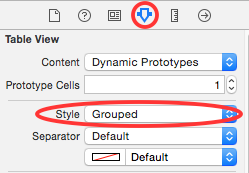


## Step 5: Configure the Cell and Table View

Select the cell. In the **Attributes Inspector**, search for the **Identifier** field. Provide the name: “**StoryTimeCell**”.



Next, select the **Table View** and in the **Attributes Inspector**, set the style to be **Grouped**.



## Step 6: Create the View Controller Class

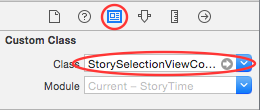
From the menu, select **File / New**. Under iOS, select **Source**. Choose **Cocoa Touch Class**, click Next. Name the file “**StorySelectionViewController**”. Make it a subclass of **UIViewController**. Click **Next**, then **Create.**

Open **StorySelectionViewController.swift**, and change the class definition to look like the following:

class StorySelectionViewController: UIViewController, UITableViewDataSource, UITableViewDelegate {

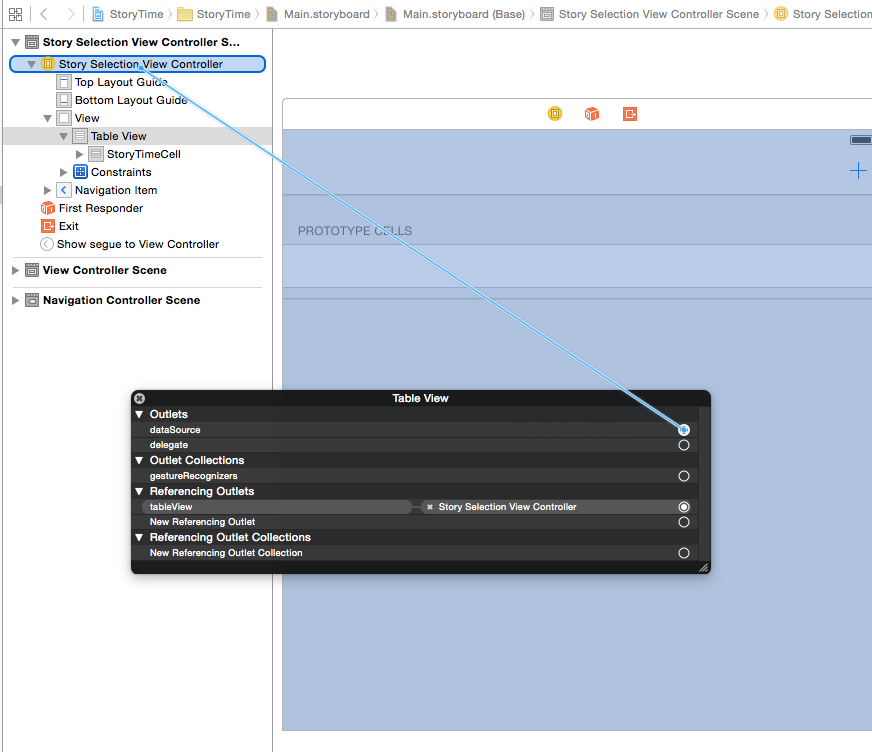
## Step 7: Provide an Identity

Return back to your storyboard. Select your new View Controller, and select the **Identity Inspector**. From the class dropdown, select **StorySelectionViewController.**



## Step 8: Setup the Delegate and the Data Source

Finally, setup your delegate and data source by right clicking or control clicking the Table View. From the dialog pop-up, first drag the dataSource to the owning View Controller. Next, do the same to delegate.



## Step 9: Create an outlet for the Table View

Select the Table View, open the Assistant Editor, and control drag to the code to create a new IBOutlet. Name the new outlet: **tableView**.

# 

# Part 3: MVC Realized

At this pont, you have a view to show your data, a model to contain your data, and a controller to manage the both of them. In this following section, you will put them all together.

## Step 1: Setting up the Model

At this point, you will create two arrays to store the model objects. Typically, you would use just one array, but for this demonstration, you will use two of them to understand how tableViews work in practice.

Open **StorySelectionViewController.swift**. At the top of the class, just underneath the newly created IBOutlet, add the following:

var zombieStories: NSMutableArray!

var vampireStories: NSMutableArray!

## Step 2: Creating the Model Objects

At the following inside of viewDidLoad():

override func viewDidLoad() {

super.viewDidLoad()

zombieStories = NSMutableArray()

vampireStories = NSMutableArray()

title = "Story Time!"

var winStory = "<name> entered the room and saw <number> <monsters>! <name> ran down the hall. Sadly, <name> was <verb> by all the <monsters>! \n\nPoor <name>. Better luck next time!"

var loseStory = "<name> entered the room and saw <number> <monsters>! Without missing a beat, <name> <verb> all of the <monsters>! \n\nPoor <monsters>. Fantastic! \n\n<name> will live to fight another day."

let story = Story(title: "Attack of the Zombies", winStory: winStory, loseStory: loseStory, type: .zombies)

zombieStories.addObject(story)

winStory = "<name> entered the room and saw <number> <monsters>! <name> ran down the hall. Sadly, <name> was <verb> by all the <monsters>! \n\nPoor <name>. Better luck next time!"

loseStory = "<name> entered the room and saw <number> <monsters>! Without missing a beat, <name> <verb> all of the <monsters>! \n\nPoor <monsters>. Fantastic! \n\n<name> will live to fight another day."

let anotherStory = Story(title: "Attack of the Vampires", winStory: winStory, loseStory: loseStory, type: .vampires)

vampireStories.addObject(anotherStory)  
}

## Step 3: Setting up the Table View

First, set the amount of sections.

func numberOfSectionsInTableView(tableView: UITableView) -> Int {

return 2

}

Second, let the table how many rows for each section.

func tableView(tableView: UITableView, numberOfRowsInSection section: Int) -> Int {

switch(section) {

case 0:

return zombieStories.count

case 1:

return vampireStories.count

default:

return 0

}

}

Third, give each of the sections a title.

func tableView(tableView: UITableView, titleForHeaderInSection section: Int) -> String? {

if section == 0 {

return "Zombie Stories"

}

return "Vampire Stories"

}

Finally, create the cells.

func tableView(tableView: UITableView, cellForRowAtIndexPath

indexPath: NSIndexPath) -> UITableViewCell {

let cell = tableView.dequeueReusableCellWithIdentifier

("StoryTimeCell", forIndexPath: indexPath) as

UITableViewCell

var story: Story!

if indexPath.section == 0 {

story = zombieStories[indexPath.row] as Story

} else {

story = vampireStories[indexPath.row] as Story

}

cell.textLabel.text = story.title

return cell

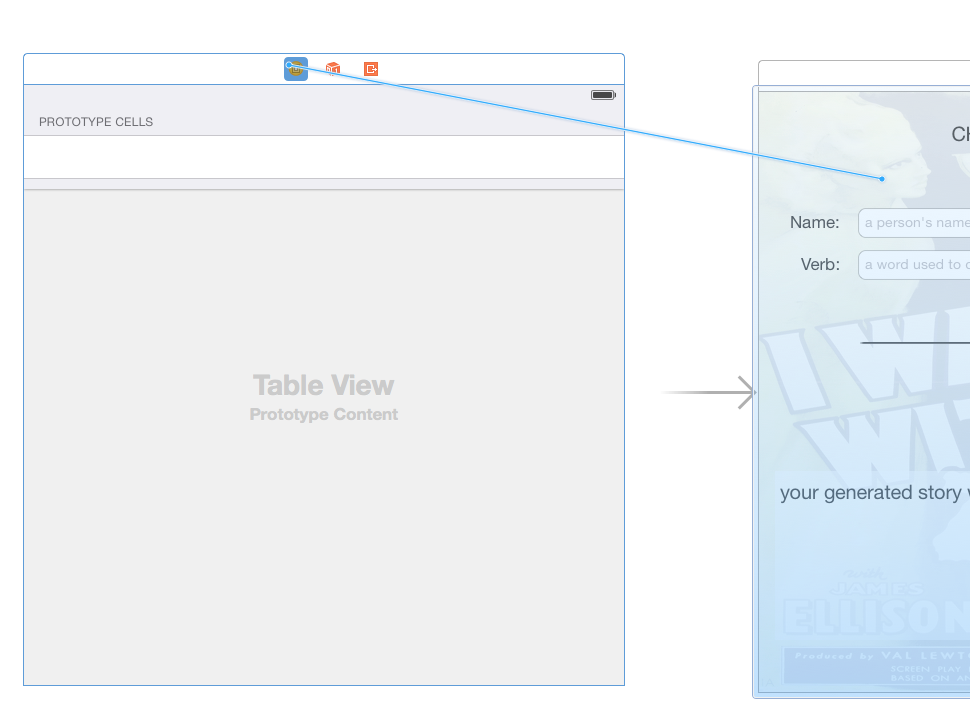
}

# Part 4: Putting It All Together

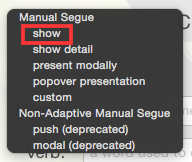
Now that your new view controller is completed, it’s time to put them all together.

## Step 1: Reconfiguring the Storyboard

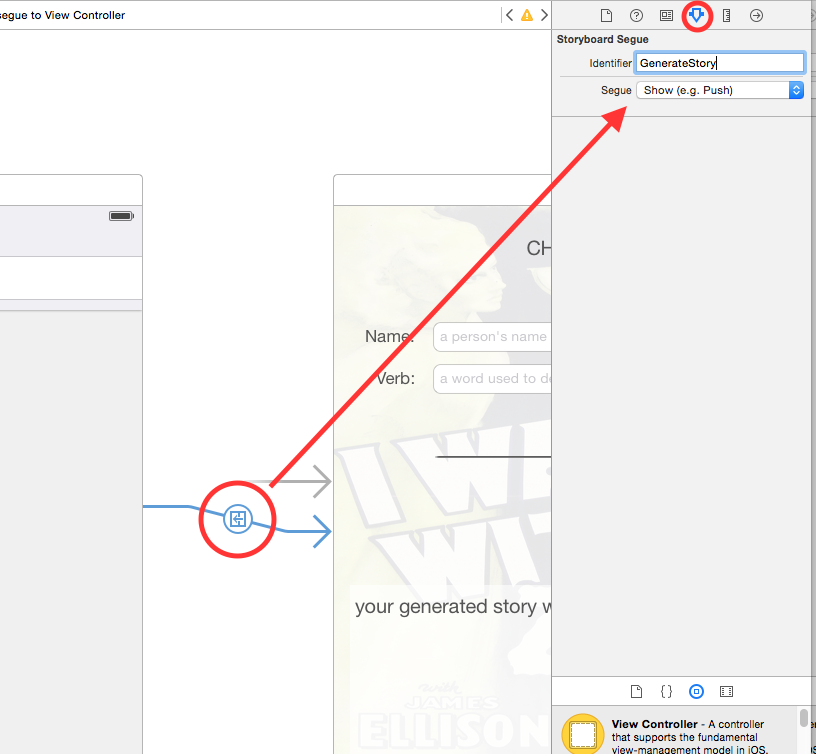
First, control click from the yellow icon of your View Controller and drag over to the existing View Controller.



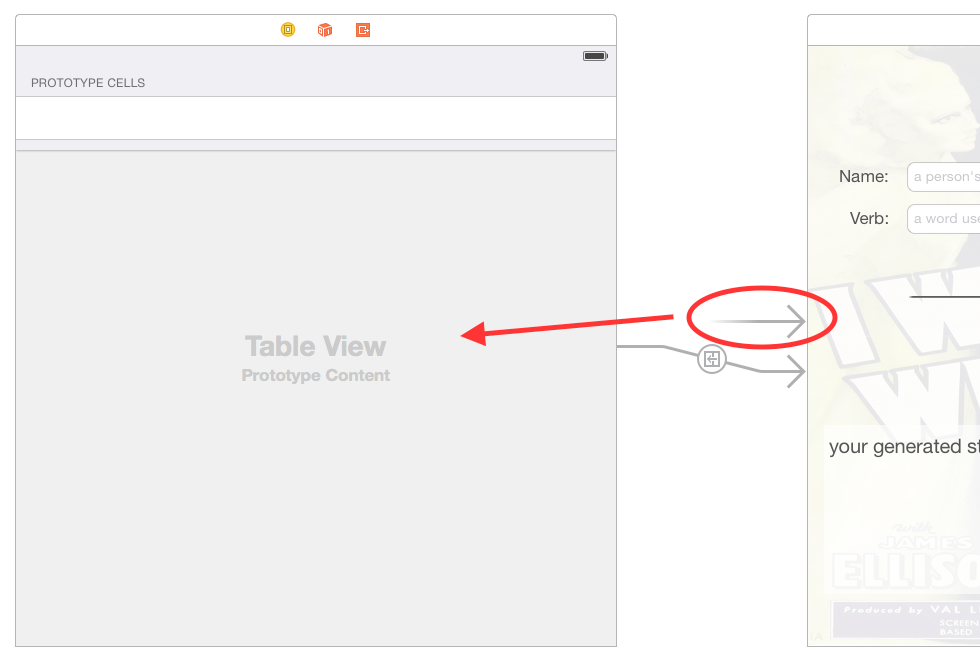
Select **Show** from the list of options.



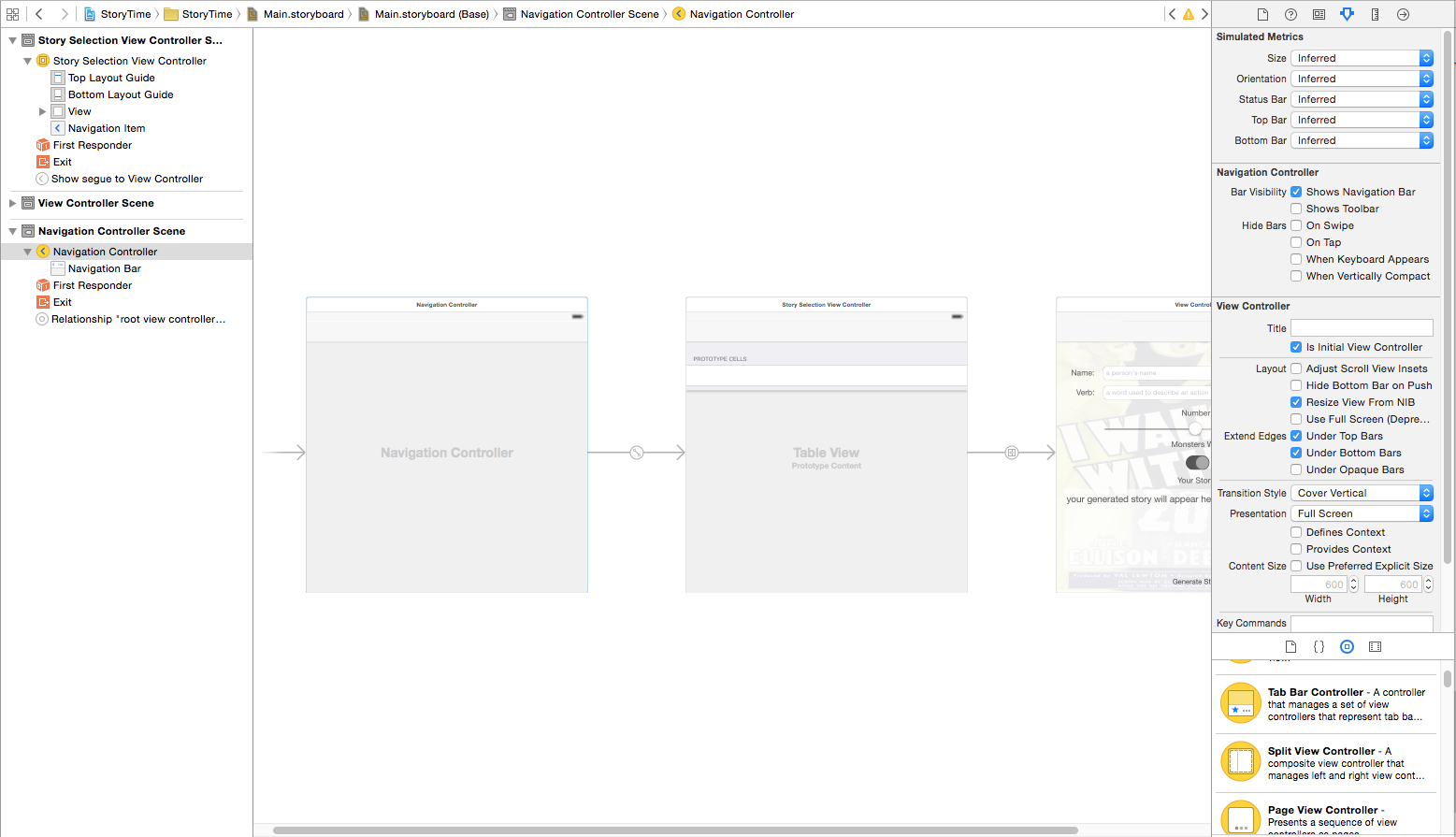
Next, select the newly created segue, and in the Attributes Inspector, give it the Identifier: **GenerateStory**.



Set your new View Controller to be the initial View Controller by dragging the Storyboard Entry Point (the arrow) to your View Controller.



Lastly, select the **StorySelectionViewController** and from the menu, select **Editor / Embed In / Navigation Controller.** Your storyboard should look like the following:



## Step 2: Setup the Existing View Controller to Use Your Model

Open **ViewController.swift** and add the following property with the existing properties.

var currentStory: Story?

In viewDidLoad(), add the following code just underneath the code that sets up the tableView.

if currentStory != nil {

var image: UIImage!

if currentStory?.type == StoryType.zombies {

image = UIImage(named: "zombies")

} else {

image = UIImage(named: "vampires")

}

backgroundImage.image = image

}

Finally replace populateStory() with the following code:

func populateStory() {

if currentStory != nil {

currentStory?.name = textField1.text

currentStory?.verb = textField2.text

currentStory?.number = Int(sliderControl.value)

textview.text = currentStory?.generateStory(switchControl.on)

}

}

## Step 3: Pass the Data between your View Controllers

Finally, pass the Story object between View Controllers. Open **StorySelectionViewController.swift** and add the following code.

override func prepareForSegue(segue: UIStoryboardSegue, sender: AnyObject?) {

if segue.identifier == "GenerateStory" {

if let indexPath = tableView.indexPathForSelectedRow() {

let storyViewController = segue.destinationViewController

as ViewController

if indexPath.section == 0 {

storyViewController.currentStory =

zombieStories[indexPath.row] as? Story

} else {

storyViewController.currentStory =

vampireStories[indexPath.row] as? Story  
 }

}

}

}