# Basic Exercises Part 4.2 Passing data. Segues.

## A 🡪 B

* Use segues to define the flow of your app’s interface. A segue defines a transition between two view controllers in your app’s storyboard file. The starting point of a segue is the button, table row, or gesture recognizer that initiates the segue. The end point of a segue is the view controller you want to display. A segue always presents a new view controller, but you can also use an *unwind segue* to dismiss a view controller.
* You do not need to trigger segues programmatically. At runtime, UIKit loads the segues associated with a view controller and connects them to the corresponding elements. When the user interacts with the element, UIKit loads the appropriate view controller, notifies your app that the segue is about to occur, and executes the transition. You can use the notifications sent by UIKit to pass data to the new view controller or prevent the segue from happening altogether.
* In other words, a segue configures a future situation, when the segue will be triggered. At that time, one view controller that already exists will cause the instantiation of another, bringing the latter into existence. For example:
* A screenshot of a social media post

  Description automatically generated

## Using segues (A🡪B). Forward.

### **1.1 Create a new project**

Create a basic Single View. We will create a new project on each approach, so we only going to write the steps once.

### **1.2 Add two views**

Open the Library (also: Shift + Cmd + L) and search for: View Controller. Click and drag it onto your view. Next, do the same thing with a second View Controller. Search for it and then drag it onto your view.

### **1.3 Create their classes**

Create a new file (File →New →File… →Cocoa Touch Class →Next →choose UIViewController in Subclass of: → name it whatever you like for example ViewControllerA →Next →Create) for your second view controller. Do the same for the second view controller.

Assign each file to the respective view controller we just created in the Identity Inspector

*A screenshot of a cell phone

Description automatically generated*

Additionally, add three labels and a button to make it easier to distinguish the two controllers. Create the outlets and actions.

### **1.4 Storyboard ID**

Remember the Identity Inspector where we assign the new file to the view controller we create? Set the Storyboard ID in the inspector section. For example:

A screenshot of a cell phone

Description automatically generated

Do the same for both View controllers. Now, in the second view controller add a property:

**var** passTextSegue = String?

And in viewDidLoad:

myLabel.text = passTextSegue

### **1.5 Creating a Segues between view controllers**

To create a segue between view controllers in the same storyboard file, Control-click an appropriate element in the first view controller and drag to the target view controller. The starting point of a segue must be a view or object with a defined action, such as a control, bar button item, or gesture recognizer. You can also create segues from cell-based views such as tables and collection views.

Some elements support multiple segues. For example, a table row lets you configure different segues for taps in the row’s accessory button and taps in the rest of the row.

When you release the mouse button, Interface Builder prompts you to select the type of relationship you want to create between the two view controllers, as shown in figure. Select the segue that corresponds to the transition you want.

A screen shot of a computer

Description automatically generated

After creating a segue, select the segue object and assign an identifier to it using the attributes inspector. During a segue, you can use the identifier to determine which segue was triggered, which is especially useful if your view controller supports multiple segues

When selecting the relationship type for your segue, select an adaptive segue whenever possible. Adaptive segues (push, replace, present modally, present as popover) adjust their behavior automatically based on the current environment. For example, the behavior of a Show segue changes based on the presenting view controller. Nonadaptive segues are provided for apps that must also run on iOS 7, which does not support adaptive segues

### **1.6 Combination Button-Segue-Label**

In the first view controller Ctrl + drag the line/segue from the button we created to the View Controller … not the label, I repeat, not the label!.

Choose “show”, now there should be a line between your two view controllers.

Override the prepareForSegue method. Inside the method check for the segue you just created by its identifier. Cast the destination view controller and pass data to it by setting properties on the downcast view controller. Add the next code on your first view controller:

**override** **func** prepare(for segue: UIStoryboardSegue, sender: Any?) {

**let** segueDestination = segue.destination **as**! ViewControllerB

segueDestination.passTextSegue = "Hello World"

}

### **1.7 Avoid crashes**

In order to prevent any crash situation, add an identifier to all your segues. Select your segue and set a name, for example:

A screenshot of a cell phone

Description automatically generated

Segues can be performed programatically or using button action event set in the storyboard by ctrl+drag to destination view controller. You can call for a segue programatically, when needed, using segue identifier in the view controller:

**Objective-C**

- (void)showDetail {

[self performSegueWithIdentifier:@"showDetailingSegue" sender:self];

}

**Swift**

func showDetail() {

self.performSegue(withIdentifier: "showDetailingSegue", sender: self)

}

You can configure segue payload in the overrided version of prepareForSegue method. You can set required properties before destination view controller is loaded.

**Objective-C**

- (void)prepareForSegue:(UIStoryboardSegue \*)segue sender:(id)sender {

if([segue.identifier isEqualToString:@"showDetailingSegue"]){

DetailViewController \*controller = (DetailViewController \*)segue.destinationViewController;

controller.isDetailingEnabled = YES;

}

}

**Swift**

override func prepare(for segue: UIStoryboardSegue, sender: Any?) {

if segue.identifier == "showDetailingSegue" {

let controller = segue.destinationViewController as! DetailViewController

controller.isDetailingEnabled = true

}

}

**1.7 Switch language**

We always need to try the same behavior on Objective C.