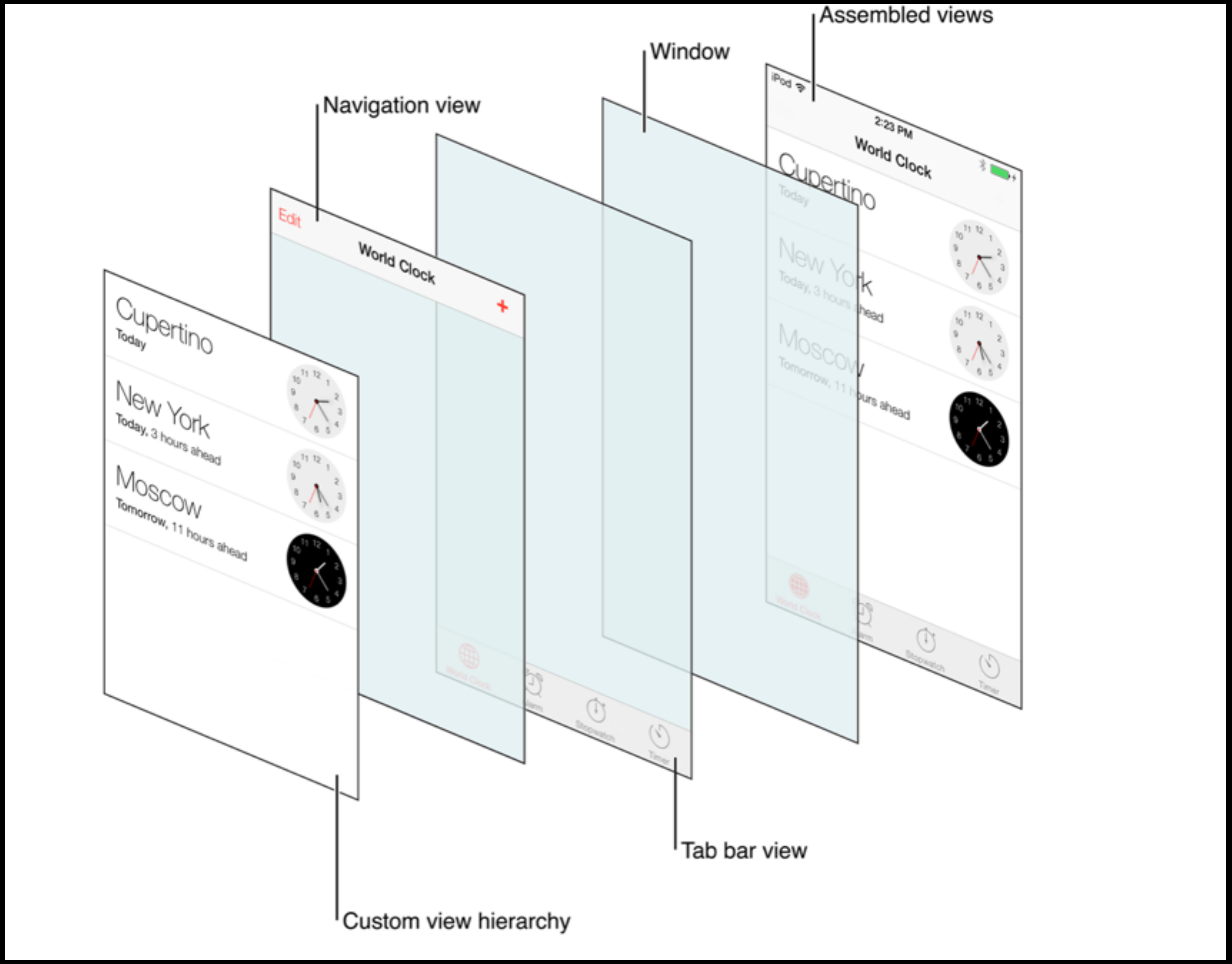
iOS Foundations II
Day 4

- Homework Review
- Navigation Controller
- Segues
- ‘Passing’ objects through segues

NavigationControllers



- “A navigation controller manages a stack of view controllers to provide a drill-down interface for hierarchical content.”

NavigationControllers

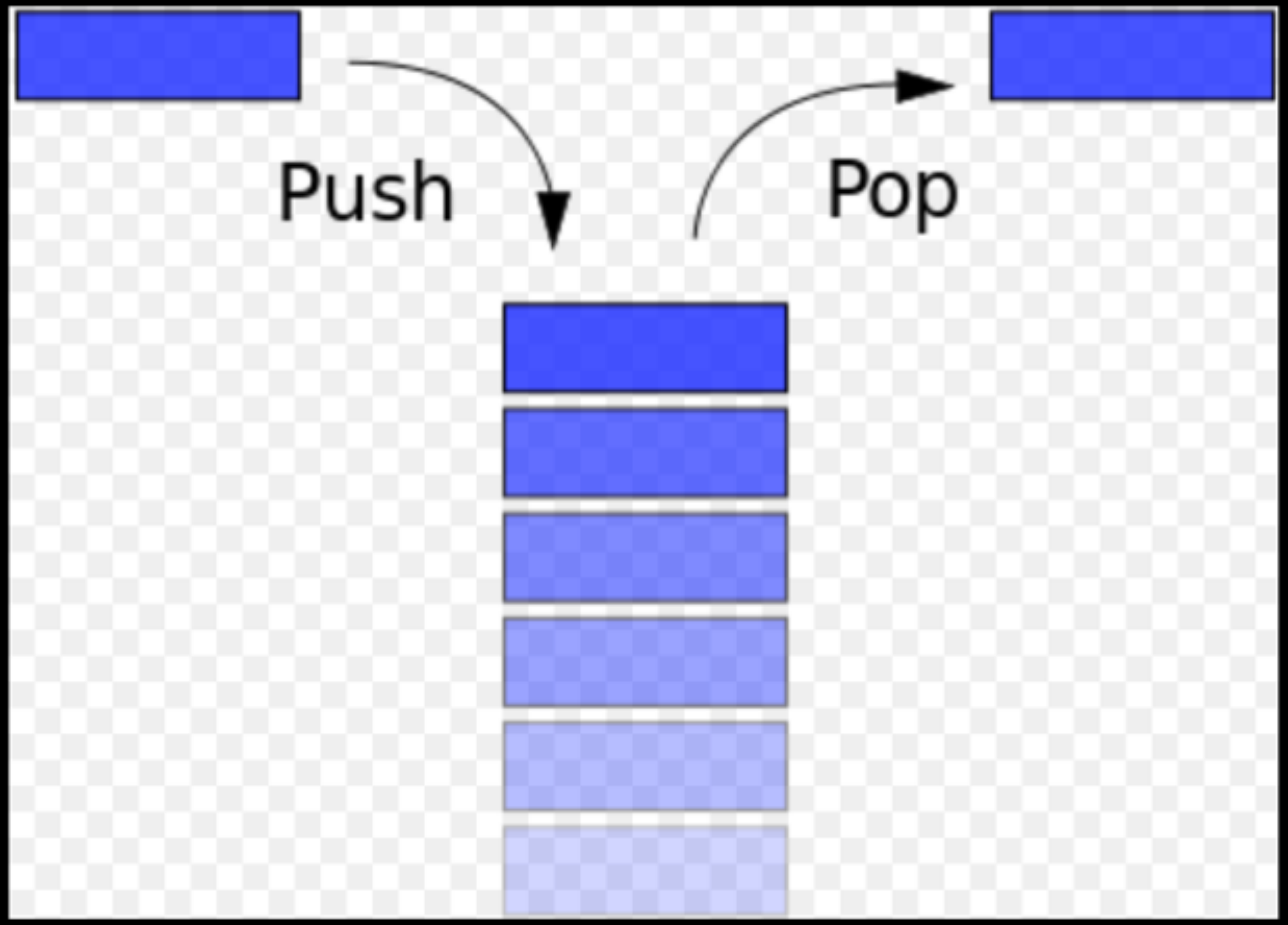
- “The navigation controller’s primary responsibility is to respond to user actions by pushing new content view controllers onto the stack or popping content view controllers off of the stack”
- The first ViewController you push onto the stack becomes the rootViewController and is never popped off because then no view would be on screen
- Nav Controllers have a property to the topViewController and an array property for all its viewControllers currently on the stack.
- The Nav bar up top can be customized or hidden.

`NavigationControllers`

- Storyboards make navigation controllers extremely easy to install into your app. Here's all the methods you need to do it in code without the storyboard:
- `init(rootViewController:)` UINavigationController is initialized with a rootViewController.
- `pushViewController(animated:)` To add or 'push' a view controller onto the stack.
- `popViewController(animated:)` To remove or 'pop' a view controller from the stack.

Stack Data Structure

- “A stack is particular kind of abstract data type or collection in which the only operations on the collection is adding (push) or removal (pop).” - Wikipedia
- LIFO : Last-In-First-Out. The last item added is the first to be removed.



Demo

Segues

- Segues are provided by the storyboard to help you easily transition from one view controller to another.
- There are 2 primary segues that are used : Show and Present
- Show refers to pushing a view controller onto the navigation stack, it usually will slide in from the right or left.
- Present refers to modally presenting a view controller, it usually slides up from the bottom.
- You can create your own custom segue to customize the behavior to match your needs.

Segues

- 2 primary methods for dealing with segues in code:
    1. performSegueWithIdentifier:
    2. prepareForSegue:sender:

`performSegueWithIdentifier`

- Call this method in your View Controller to trigger a segue in code.
- If you your segue isn't hooked up to be triggered by an action in your interface, you will need to call this method to trigger the segue.

`prepareForSegue:sender:`

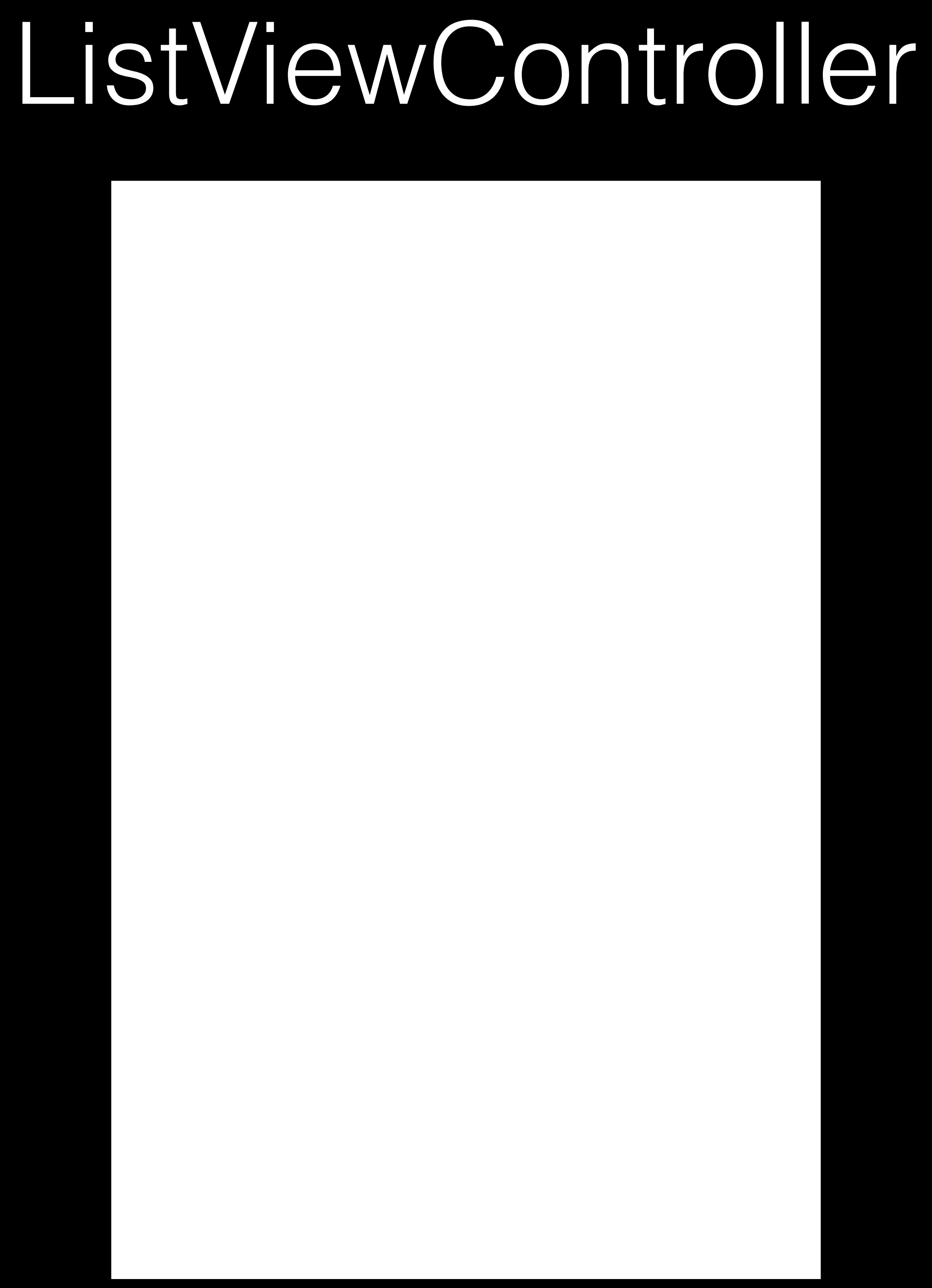
- This method is called on the source view controller of the segue, right before the segue is actually performed.
- The first paramater is the segue itself, which is an instance of the UISegue class.
- The most important property of a UISegue instance is the destinationViewController property, which gives you a reference to the view controller you are about to segue to.
- This is a great spot to pass information to the next screen.

Demo

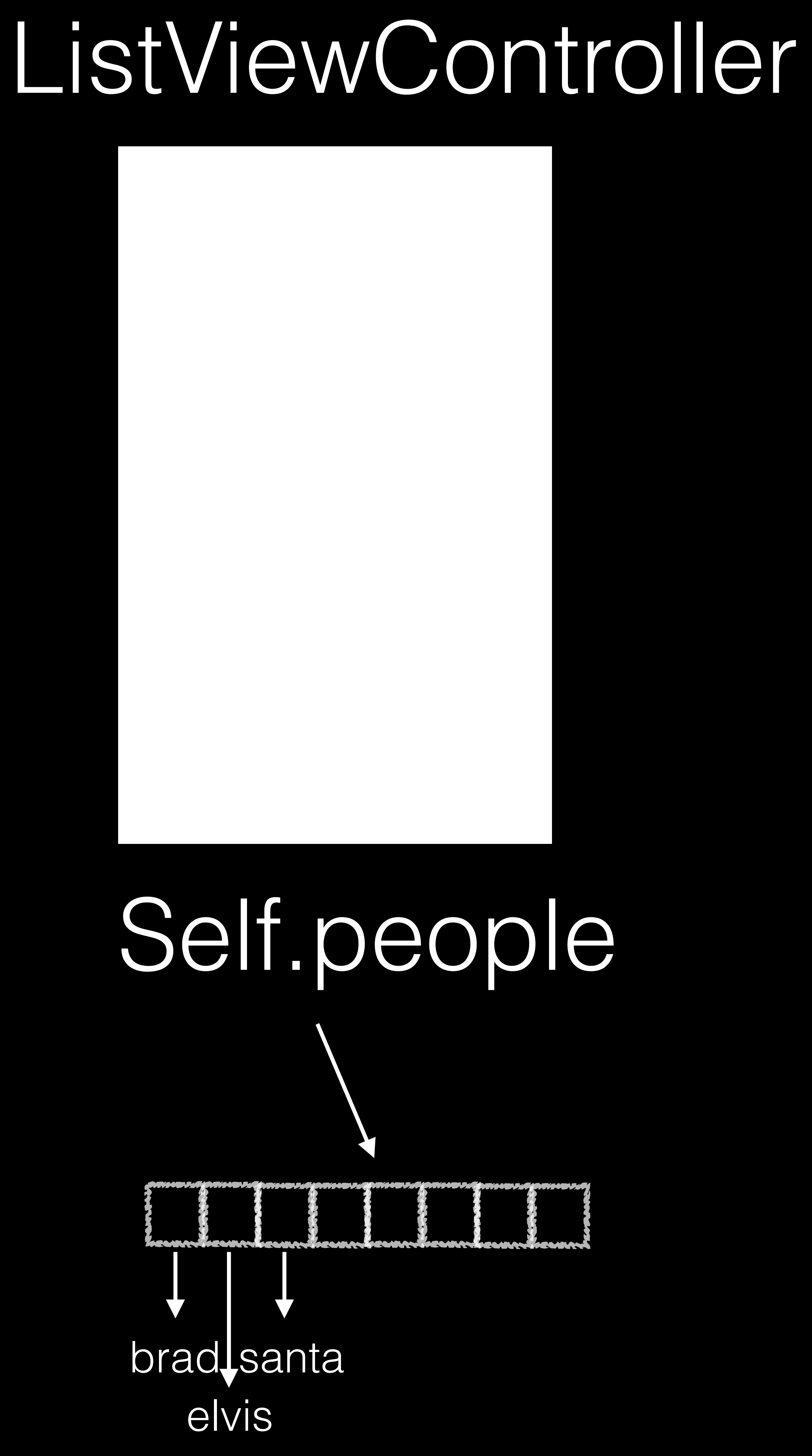
Passing data to View Controllers

- There are many patterns we can use to pass data around our app:
  - Delegation
  - Notification Center
  - Singletons
- But for now, we can do something as simple as passing a reference directly to the new view controller.

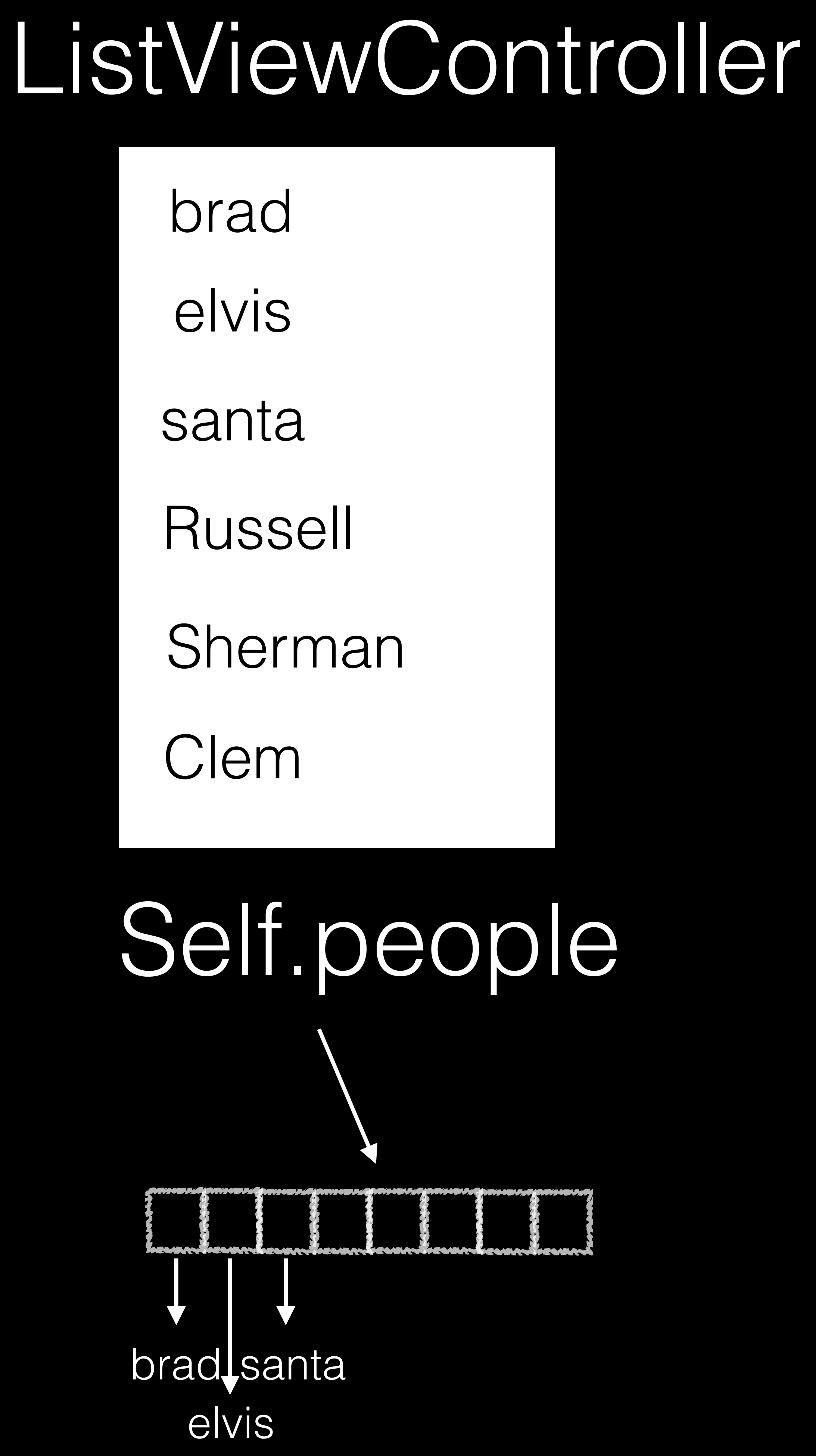
Passing data to View Controllers



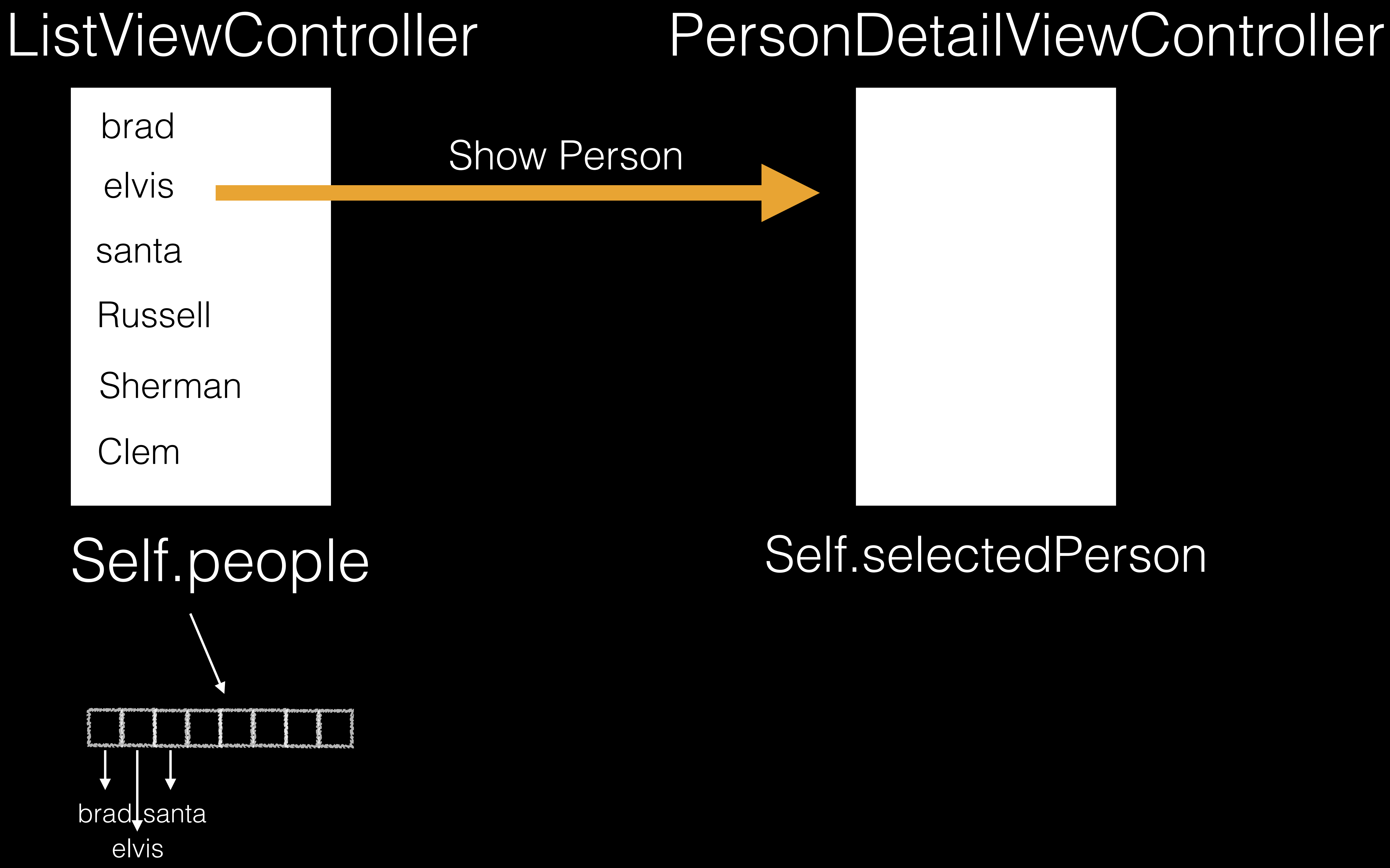
ListViewController creates an array of Person objects and sets it people property ’s value to that array



ListViewController, as the datasource of the tableview, uses the self.people array as the backing array for the tableview



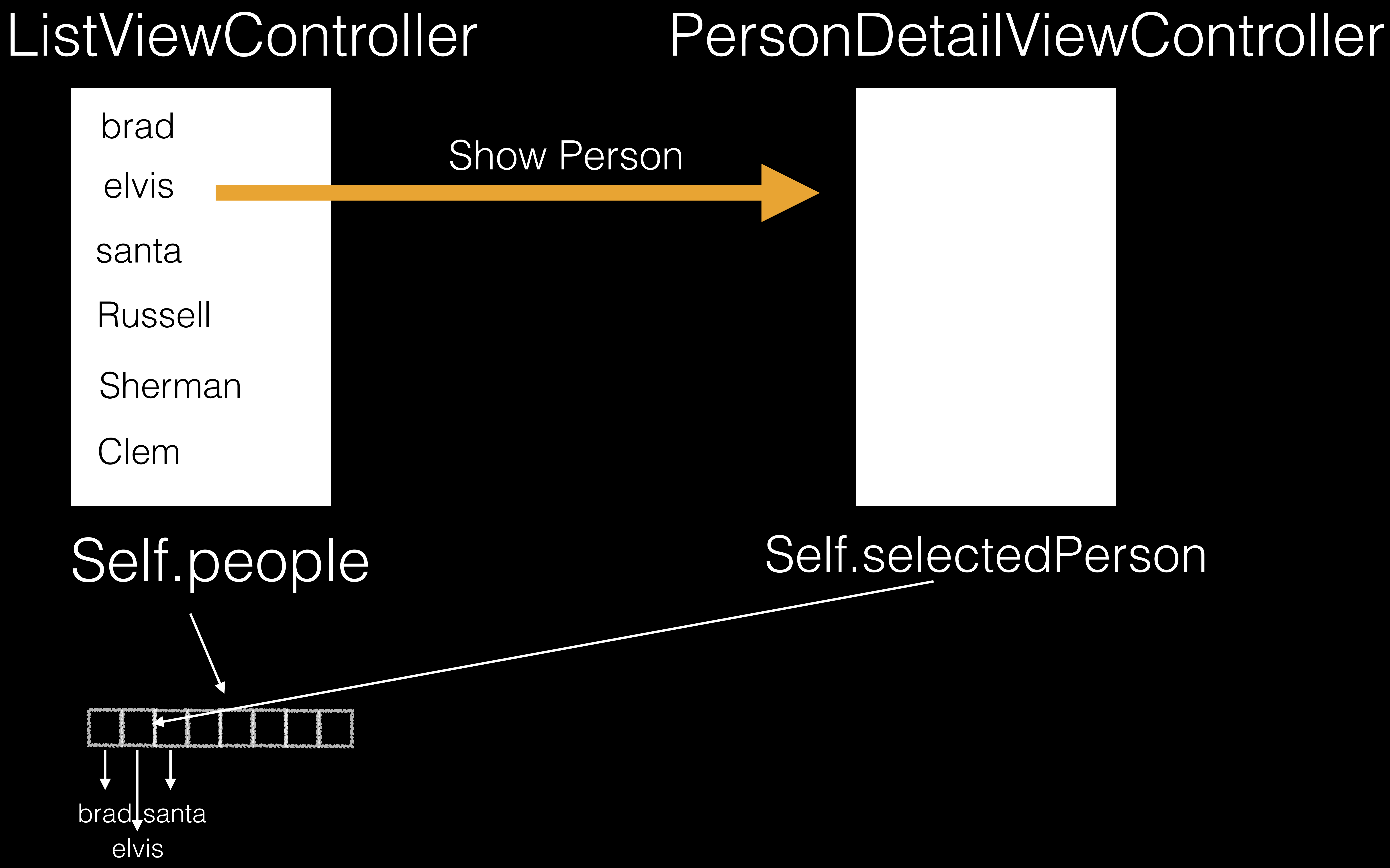
The User clicks on a cell, triggering our ShowPerson Segue. The PersonDetailViewController is initialized.



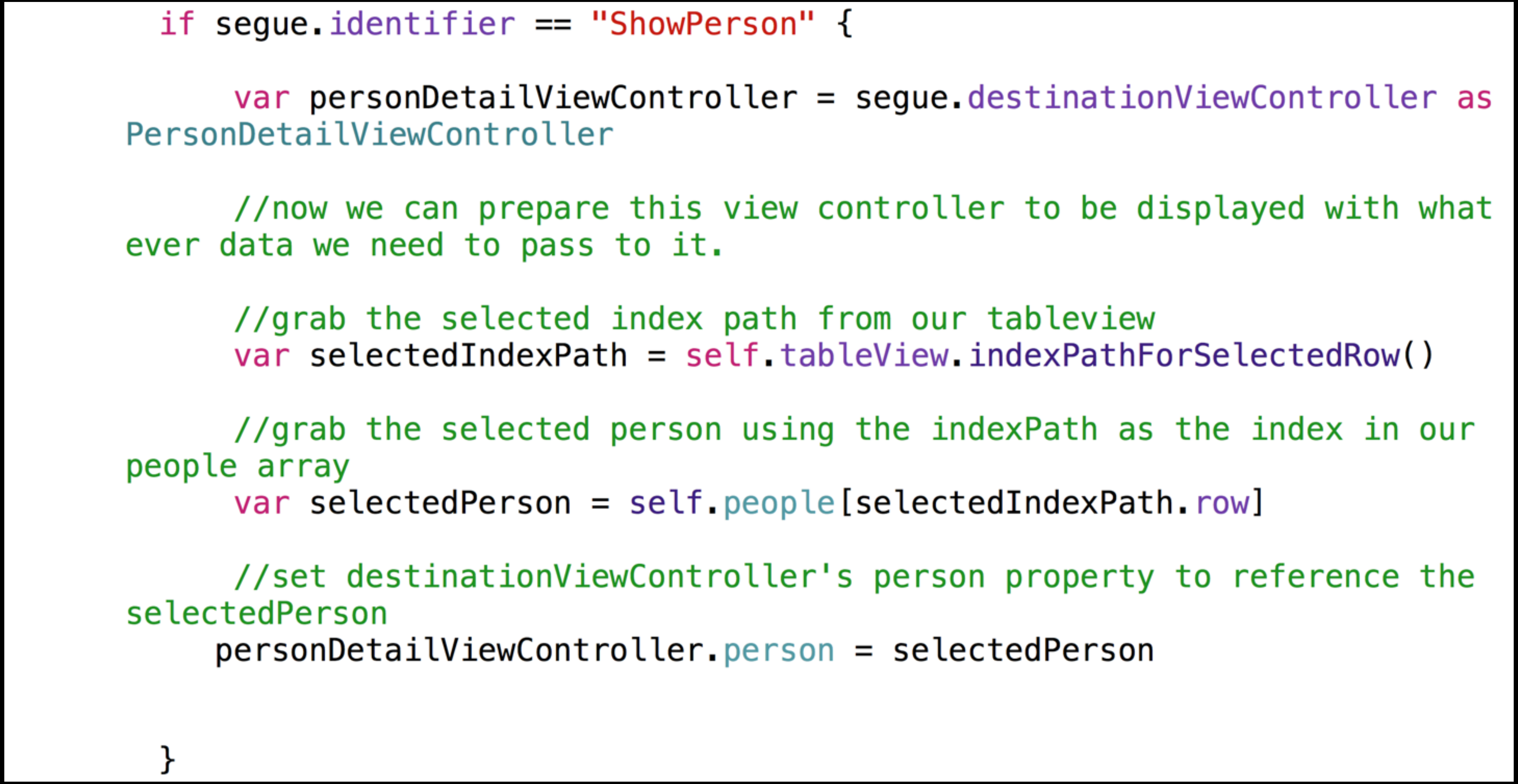
In Prepare for Segue, the ListViewController intercepts the segue and grabs a reference to the destination view controller, which is the PersonDetailViewController.

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

    if segue.identifier == "ShowPerson" {

        var personDetailViewController = segue.destinationViewController as
  PersonDetailViewController

        //now we can prepare this view controller to be displayed with what
  ever data we need to pass to it.
    }
}
```

ListViewController sets PersonDetailViewController's selectedPerson property to reference the person that was clicked from the list.

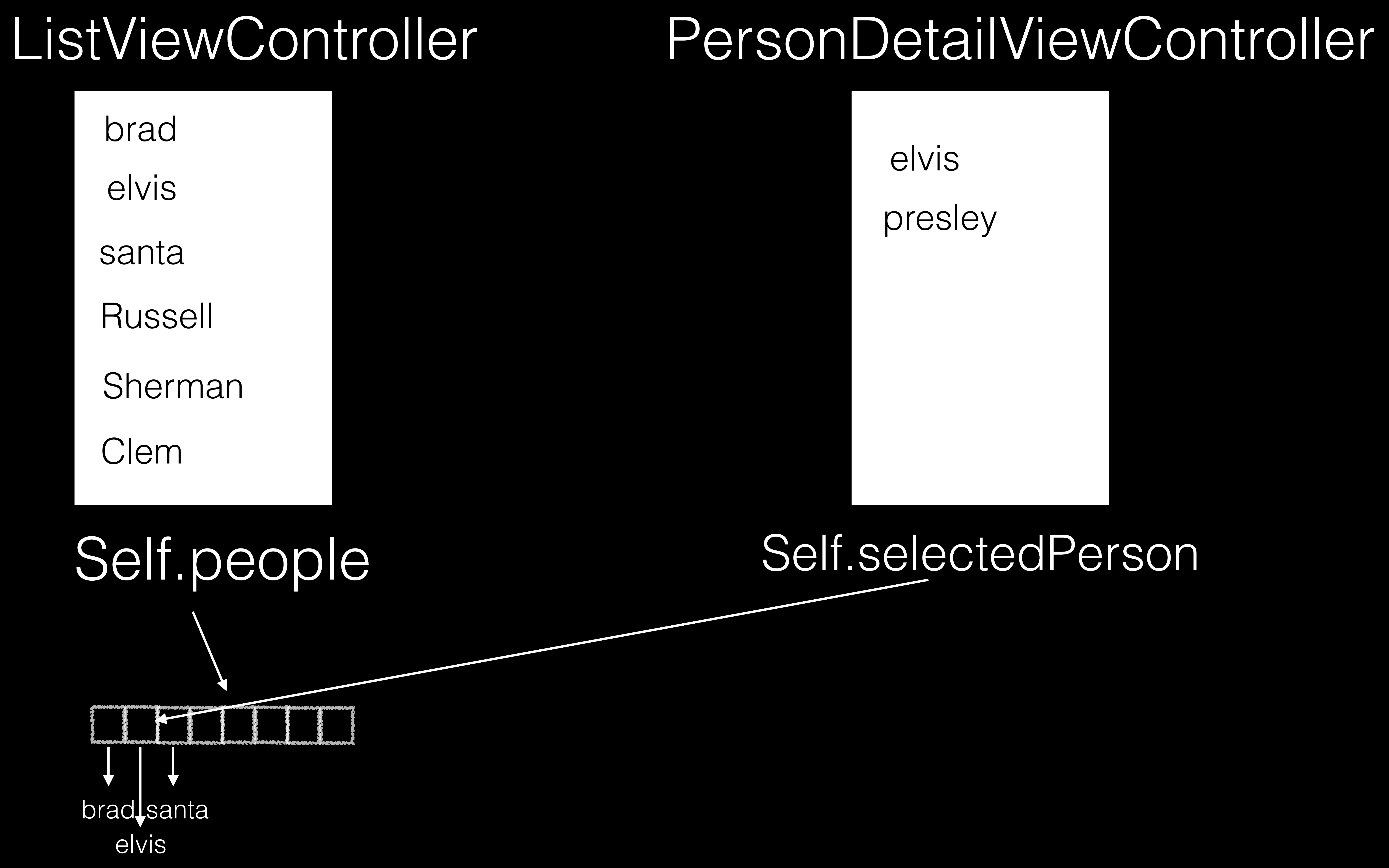


ListViewController sets PersonDetailViewController’s selectedPerson property to reference the person that was clicked from the list.



```
    if segue.identifier == "ShowPerson" {

        var personDetailViewController = segue.destinationViewController as
  PersonDetailViewController

        //now we can prepare this view controller to be displayed with what
  ever data we need to pass to it.

        //grab the selected index path from our tableview
        var selectedIndexPath = self.tableView.indexPathForSelectedRow()

        //grab the selected person using the indexPath as the index in our
  people array
        var selectedPerson = self.people[selectedIndexPath.row]

        //set destinationViewController's person property to reference the
  selectedPerson
      personDetailViewController.person = selectedPerson


    }
```

Now PersonDetailViewController can use it's selectedPerson property to fill out his interface with that Person's name properties. Any changes he makes to this Person object apply to the person object inside the original array!


ListViewController
PersonDetailViewController
brad
elvis
santa
Russell
Sherman
Clem
elvis
presley
Self.people
Self.selectedPerson
brad
santa
elvis