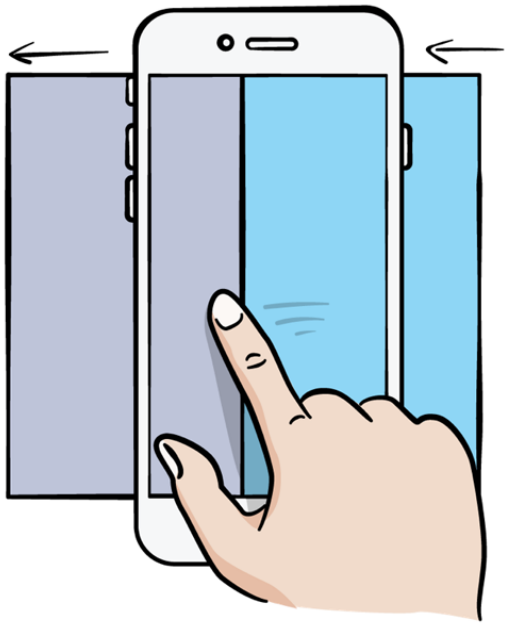
SCROLL VIEW SCHOOL



Scroll View School

Brian Moakley

Notice of Rights

Notice of Liability

Trademarks

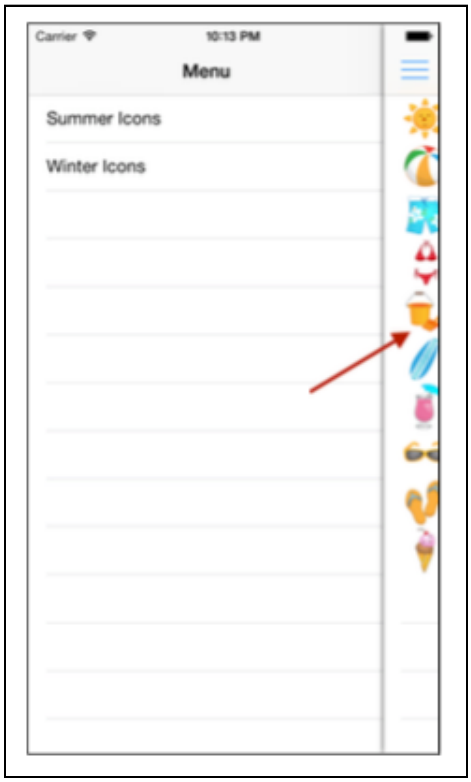
Table of Contents: Overview

Table of Contents: Extended

Scroll View School for Video 15: Sidebar 2

By Brian Moakley

When the sidebar is open, you can still tap on one of the rows in the table view and it will accept your touch and display an icon detail view in the main view.



Challenge

That's not what you want – the sidebar is open, after all! Most sidebar menu implementations will close the sidebar when you tap on that overlap area, so that's what you'll do in this challenge.

To start, add these lines to the end of `setupScrollView`. Just be warned, you'll get a few compile errors:

```
let tapRecognizer = UITapGestureRecognizer(target: self,
  action: #selector(viewTapped(tapRecognizer:)))
tapRecognizer.delegate = self
view.addGestureRecognizer(tapRecognizer)
```

This creates a tap gesture recognizer on the view and sets the view controller as its delegate.

You'll get an error on the delegate line since you haven't set this class as conforming to UIGestureRecognizerDelegate yet, but you'll fix that in a moment.

Next, add the implementation for the action method:

```
func viewTapped(tapRecognizer: UITapGestureRecognizer) {
  closeMenuAnimated(true)
}
```

This is pretty simple – when the user taps, you just close the menu.

You might have noticed a problem here – this gesture recognizer will fire on any tap on the entire screen! What you need is to only have the gesture recognizer run if the menu is open and the tap was in that overlap area.

Scroll to the very bottom of the file and add the following class extension and delegate method:

```
extension SidebarViewController: UIGestureRecognizerDelegate {
  func gestureRecognizer(_ gestureRecognizer:
    UIGestureRecognizer, shouldReceive touch: UITouch) -> Bool {

    let tapLocation = touch.location(in: view)
    let tapWasInOverlapArea = tapLocation.x >=
      view.bounds.width - overlap
    return tapWasInOverlapArea && leftMenuIsOpen()
  }
}
```

You're implementing `gestureRecognizer(_:shouldReceiveTouch:)` which lets you return a Boolean to say whether the recognizer should recognize the touch or not.

The two conditions are simple – the tap has to be in that overlap area to the right, and the left menu has to be open. If that's the case, then the gesture recognizer fires and the sidebar closes.

Build and run, and open the sidebar. You should be able to tap in that overlap area now to close the sidebar!

Über-h4xx0r Challenge: Right Sidebar

Your über challenge is to implement a right sidebar, using the techniques and theory you learned about in the video. Here are a few tips:

- Add a new parameter to to your initializer in `SidebarViewController` that takes a right view controller.
- In `setupViewControllers`, add the appropriate code and auto layout constraints for the right view controller.
- In `AppDelegate`, create another `MenuViewController` (and a navigation controller that contains it) and pass it to `SidebarViewController` as the right view controller.
- Set up the constraints of the scroll view in `SidebarViewController` so that it's width is the equal to the width of the main view, minus the gap.
- Set `clipsToBounds` on the scroll view to false.
- Set the `contentInset` on the scroll view properly so that you don't continue toscroll beyond the right hand side of the right view controller.
- Set `showsHorizontalScrollIndicator` on the scroll view to NO (it doesn't look nice).
- Add the scroll view's pan gesture recognizer to the main view, so that you can drag inside the "gap" area.
- Modify `gestureRecognizer(_:shouldReciveTouch:)` appropriately, since you now have a right sidebar as well. That means if there's a tap on the left side of the screen and the right menu is open, you should close the menu. You might want to add a `rightMenuIsOpen()` helper method too.

If you get this working, congratulations: you now have a solid understanding of `UIScrollView` and how it works! There are several variations to get it all working, so feel free to check out the full challenge solution to see one option.