Table Views in iOS

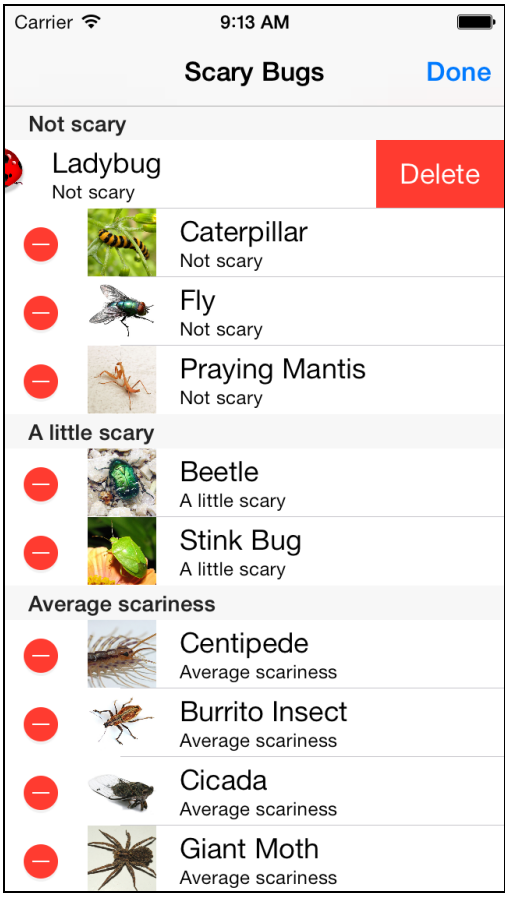
Hands-On Challenges

Table Views: Beginning to Advanced Hands-On Challenges

Challenge #4: Deleting Rows

I don't know about you, but when I see bugs, I'm not the type to kindly pick them up with a napkin and walk it outdoor – no, I just want to squash it!

So in this challenge, you will add the ability to delete rows from the table view to squash those pesky bugs.



See if you can do this on your own based on what you learned on the video. If you get stuck, follow along with the full walkthrough below!

Full Walkthrough

Open the Scary Bugs project where you left it off in the last challenge, or use the starter project provided by the instructor.

Open **BugTableViewController.swift** and add the following method to the bottom of the file:

```
override func tableView(tableView: UITableView, commitEditingStyle
  editingStyle: UITableViewCellEditingStyle, forRowAtIndexPath
  indexPath: NSIndexPath) {

  if editingStyle == .Delete {
    let bugSection = bugSections[indexPath.section]
    bugSection.bugs.removeAtIndex(indexPath.row)
    tableView.deleteRowsAtIndexPaths([indexPath],
      withRowAnimation: .Automatic)
  }
}
```

This is the method you need to implement to support deleting rows in your table view. Let's review this line by line:

1. First you need to check the editing style that is passed in. The options for editing style are none, insert or delete; you want delete here.

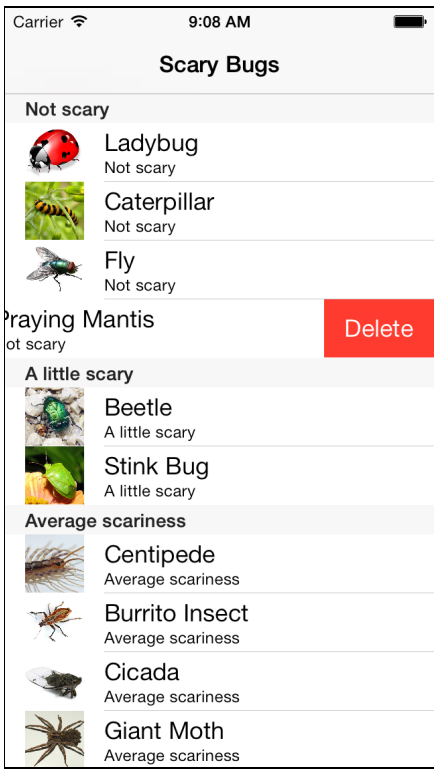
2. You look up the `BugSection` for the `indexPath.section` as usual.

3. There are two steps whenever you want to update a table view. First, you need to update the model appropriately. This is what you do here: you remove the appropriate bug from the `BugSection`.

4. Second, you need to tell the table view that you have deleted a row, so the visual representation of the model matches the model itself. The best way to do this is by calling `deleteRowsAtIndexPaths(_:withRowAnimation:)` like you see here, because it performs an animation on the row to show the user visually that it has been successfully deleted. Note that you could call `tableView.reloadData()` instead, but the disadvantage is you wouldn't get the nice animation like you do here.

5. You'll add some extra code here in a moment.

Build and run, and swipe left on a row to reveal a delete button:



Tap the button, and the bug will be removed from the list.

One last thing. Sometimes users do not realize they can delete bugs by swiping a cell to the left, so let's add a button to turn on editing mode on all cells to help out those users.

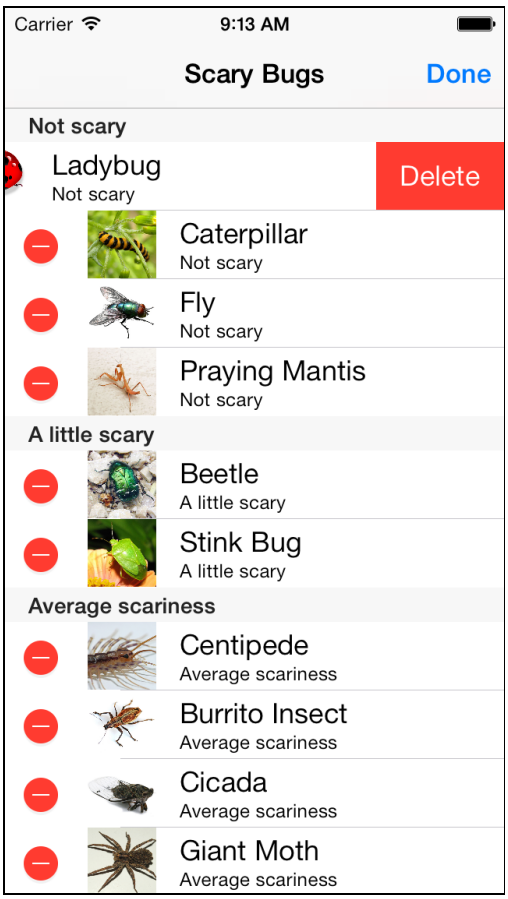
To do this, simply add this line of code to the end of `viewDidLoad`:

```
navigationItem.rightBarButtonItem = editButtonItem()
```

`UIViewController` comes with a built-in bar button item you can use to easily enable/disable editing mode on a view controller. Here you take that bar button item and place it in the upper right of the navigation bar.

And that's it! Because you're subclassing `UITableViewController`, `UITableViewController` is already configured to override `setEditing(_:animated:)` (which gets called when the user taps this button) to toggle editing mode on the table view controller.

Build and run, and tap the edit button to reveal buttons, which you can then tap to reveal the delete buttons:



Congratulations – now enjoy squashing some bugs!