Swift - How creating custom viewForHeaderInSection, Using a XIB file?

I can create simple custom viewForHeaderInSection in programmatically like below. But I want to do much more complex things maybe connection with a different class and reach their properties like a tableView cell. Simply, I want to see what I do.

```
func tableView(tableView: UITableView, viewForHeaderInSection section: Int) ->
UIView? {

    if(section == 0) {

        let view = UIView() // The width will be the same as the cell, and the
height should be set in tableView:heightForRowAtIndexPath:
        let label = UILabel()
        let button   = UIButton(type: UIButtonType.System)

        label.text="My Details"
        button.setTitle("Test Title", forState: .Normal)
        // button.addTarget(self, action: Selector("visibleRow:"),
forControlEvents:.TouchUpInside)

        view.addSubview(label)
        view.addSubview(button)

        label.translatesAutoresizingMaskIntoConstraints = false
        button.translatesAutoresizingMaskIntoConstraints = false

        let views = ["label": label, "button": button, "view": view]

        let horizontallayoutContraints =
NSLayoutConstraint.constraintsWithVisualFormat("H:|-10-[label]-60-[button]-10-|",
options: .AlignAllCenterY, metrics: nil, views: views)
        view.addConstraints(horizontallayoutContraints)

        let verticalLayoutContraint = NSLayoutConstraint(item: label, attribute:
.CenterY, relatedBy: .Equal, toItem: view, attribute: .CenterY, multiplier: 1,
constant: 0)
        view.addConstraint(verticalLayoutContraint)

        return view
    }

    return nil
}


func tableView(tableView: UITableView, heightForHeaderInSection section: Int) ->
CGFloat {
    return 50
}
```

Is there anyone to explain how can I create a custom tableView header view using xib? I have encountered with old Obj-C topics but I'm new with Swift language. If someone explain as detailed, It would be great.

> **1.issue:** Button @IBAction doesn't connect with my ViewController. **(Fixed)**

Solved with File's Owner, ViewController base class (clicked left outline menu.)

> **2.issue:** Header height problem **(Fixed)**

Solved adding **headerView.clipsToBounds = true** in **viewForHeaderInSection**: method.

For constraint warnings this answer solved my problems:

When I added ImageView even same height constraint with this method in viewController, it flow over tableView rows look like picture.

```
 func tableView(tableView: UITableView, heightForHeaderInSection section: Int) ->
CGFloat {
    return 120
}
```

If I use, automaticallyAdjustsScrollViewInsets in viewDidLoad, In this case image flows under navigationBar. -fixed-

```
self.automaticallyAdjustsScrollViewInsets = false
```

> **3.issue:** If button under View **(Fixed)**

```
@IBAction func didTapButton(sender: AnyObject) {
    print("tapped")

    if let upView = sender.superview {
        if let headerView = upView?.superview as? CustomHeader {
            print("in section \(headerView.sectionNumber)")
        }

    }
}
```

ios swift uitableview xib

edited May 23 '17 at 11:33
Community ♦
1 1

asked Apr 28 '16 at 22:33
Burak
898 3 12 36

Try looking here: stackoverflow.com/questions/9219234/… – Curmudgeonlybumbly Apr 28 '16 at 22:49

FWIW, the accepted answer to that other question is a kludgy solution that was used before iOS offered dequeue capabilities for header/footer views. – Rob Apr 29 '16 at 6:41

1 Answer

The typical process for NIB based headers would be:

1. Create `UITableViewHeaderFooterView` subclass with, at the least, an outlet for your label. You might want to also give it some identifier by which you can reverse engineer to which section this header corresponds. Likewise, you may want to specify a protocol by which the header can inform the view controller of events (like the tapping of the button). Thus, in Swift 3:

   ```
   // if you want your header to be able to inform view controller of key events,
   create protocol

   protocol CustomHeaderDelegate: class {
       func didTapButton(in section: Int)
   }

   // define CustomHeader class with necessary `delegate`, `@IBOutlet` and
   `@IBAction`:

   class CustomHeader: UITableViewHeaderFooterView {
       weak var delegate: CustomHeaderDelegate?

       @IBOutlet weak var customLabel: UILabel!

       var sectionNumber: Int!  // you don't have to do this, but it can be useful to
   have reference back to the section number so that when you tap on a button, you
   know which section you came from

       @IBAction func didTapButton(_ sender: AnyObject) {
           delegate?.didTapButton(in: sectionNumber)
       }

   }
   ```

2. Create NIB. Personally, I give the NIB the same name as the base class to simplify management of my files in my project and avoid confusion. Anyway, the key steps include:
   - Create view NIB, or if you started with an empty NIB, add view to the NIB;
   - Set the base class of the view to be whatever your `UITableViewHeaderFooterView` subclass was (in my example, `CustomHeader`);
   - Add your controls and constraints in IB;
   - Hook up `@IBOutlet` references to outlets in your Swift code;
   - Hook up the button to the `@IBAction`
3. In the `viewDidLoad` in the view controller, register the NIB. In Swift 3:

   ```
   override func viewDidLoad() {
       super.viewDidLoad()
   ```

```
    tableView.register(UINib(nibName: "CustomHeader", bundle: nil),
forHeaderFooterViewReuseIdentifier: "CustomHeader")
}
```

Or in Swift 2:

```
override func viewDidLoad() {
    super.viewDidLoad()

    tableView.registerNib(UINib(nibName: "CustomHeader", bundle: nil),
forHeaderFooterViewReuseIdentifier: "CustomHeader")
}
```

4. In `viewForHeaderInSection`, dequeue a reusable view using the same identifier you specified in the prior step. Having done that, you can now use your outlet, you don't have to do anything with programmatically created constraints, etc. The only think you need to do (for the protocol for the button to work) is to specify its delegate. For example, in Swift 3:

```
override func tableView(_ tableView: UITableView, viewForHeaderInSection section:
Int) -> UIView? {
    let headerView = tableView.dequeueReusableHeaderFooterView(withIdentifier:
"CustomHeader") as! CustomHeader

    headerView.customLabel.text = content[section].name  // set this however is
appropriate for your app's model
    headerView.sectionNumber = section
    headerView.delegate = self

    return headerView
}

override func tableView(_ tableView: UITableView, heightForHeaderInSection section:
Int) -> CGFloat {
    return 44  // or whatever
}
```

Or, in Swift 2:

```
override func tableView(tableView: UITableView, viewForHeaderInSection section:
Int) -> UIView? {
    let headerView =
tableView.dequeueReusableHeaderFooterViewWithIdentifier("CustomHeader") as!
CustomHeader

    headerView.customLabel.text = content[section].name
    headerView.sectionNumber = section
    headerView.delegate = self

    return headerView
}

override func tableView(tableView: UITableView, heightForHeaderInSection section:
Int) -> CGFloat {
    return 44  // or whatever
}
```

5. Obviously, if you're going to specify the view controller as the `delegate` for the button in the header view, you have to conform to that protocol:

```
extension ViewController: CustomHeaderDelegate {
    func didTapButton(in section: Int) {
        print("\(section)")
    }
}
```

This all sounds confusing when I list all the steps involved, but it's really quite simple once you've done it once or twice. I think it's simpler than building the header view programmatically.

edited Jan 30 '17 at 8:30

answered Apr 29 '16 at 6:29
Rob
263k 40 480 623

Appreciate your detailed answer, I have just started to follow your steps and approved as an answer. Later I can leave some comments here as small question. Thanks for all. – Burak Apr 29 '16 at 9:39

I added all steps, header looks well. But I couldn't connect button with @IBAction in my ViewController. What I'm missing, Need I add another thing for my tableView? – Burak Apr 29 '16 at 11:04

2 Did you set the File Owner of the NIB to be your view controller class? If so, when you select the button, the view controller should be one of the two classes that are available under the "Automatic" setting in the assistant editor. – Rob Apr 29 '16 at 13:00

Ok, now I cannot handle section number, doesn't enter 'if let headerView = sender.superview as? CustomHeader {..' looks only 'tapped' on the xcode console screen. Also I have tried 'if let _ = buttonNIB.superview as? CustomHeader {' in CustomHeader subclass again, it didn' response. – Burak Apr 29 '16 at 13:41

1 I'd suggest you examine your view hierarchy (eg using the view debugger) and just see what the button's `superview` is. Did you, for example, put the button within a container view of the `CustomHeader`? – Rob Apr 29 '16 at 14:19

OK. When I drag my button to first hierarchy in CustomHeader, it works now. What if I have another button in the another view, how Can I reach parent CustomHeader to provide section number again? – Burak Apr 29 '16 at 14:27

Alternatively I think that using `headerView.goBtn.setTitle("sectionBtn", forState: .Normal)` like this and then handle it in the subclass. But I doesn't give me section number. – Burak Apr 29 '16 at 14:36

I understand, I need to two times different cast if my button in the another view, I have edited my question all problems encountered to be helpful everyone. Thanks again Rob. – Burak Apr 29 '16 at 14:58

1 I think it is worth mentioning: add 'self.tableView.sectionHeaderHeight = UITableViewAutomaticDimension' and 'self.tableView.estimatedSectionHeaderHeight = 70' to 'viewDidLoad'. If your constraints are pining each subview on all sides, you get Self-Sized headers – Torsten B Aug 5 '16 at 13:31

1 Thank you for detailed answer! ;) – Tom Calmon Aug 23 '16 at 19:34

somehow NIB view bg color doesn't work, I have to set it in viewForHeaderInSection – djdance Jun 14 '17 at 12:54

Very Impressed and appreciate your answers, it helps me lot. Thanks – Abhishek Mitra Jul 5 '17 at 8:31

Without you Mr @Rob I would be lost in this swift world :) – Annjawn Dec 23 '17 at 3:58

To restate what @Rob said in an above comment, ensure that when you hook your IBOutlet to your xib that you select the name of the custom class, not `File Owner`. – Jacob Davis 12 mins ago edit