Subclassing UITableViewCell-07

In this assignment you will create your own custom table view cell for use in the table view in your HomeOwner app.

1. Ensure you commit and push, then submit your commit ID for Lab 7 before moving on with this lab. Also, ensure you create a new repository for this lab, remove your remote for lab 7, add a new remote for lab 8, then do a push (also don't forget to add collaborators).

2. Create a new Swift file named ItemCell. In ItemCell.swift, define ItemCell as a UITableViewCell subclass.

3. The easiest way to configure a UITableViewCell subclass is through a storyboard. In your previous lab, you saw that storyboards for table view controllers have a Prototype Cells section. This is where you will lay out the content for the ItemCell.

4. Open Main.storyboard and select the UITableViewCell in the document outline. Open its attributes inspector, change the Style to Custom, and change the Identifier to ItemCell.

5. Now open its identity inspector. In the Class field, enter ItemCell.

6. Change the height of the prototype cell to be about 65 points tall. You can change it either on the canvas or by selecting the table view cell and changing the Row Height from its size inspector.

7. An ItemCell will display three text elements, so drag three UILabel objects onto the cell. Configure them as shown below. Make the text of the bottom label a slightly smaller font in a light shade of gray. Make sure that the labels do not overlap at all.

8. Add constraints to these three labels as follows.
   a) Select the top-left label and open the Auto Layout Add New Constraints menu. Select the top and left strut and then click Add 2 Constraints.
   b) You want the bottom-left label to always be aligned with the top-left label. Control-drag from the bottom-left label to the top-left label and select Leading.
   c) With the bottom-left label still selected, open the Add New Constraints menu, select the bottom strut, and then click Add 1 Constraint.
   d) Select the right label and Control-drag from this label to its superview on its right side. Select both Trailing Space to Container Margin and Center Vertically in Container.
   e) Select the bottom-left label and open its size inspector. Find the Vertical Content Hugging Priority and lower it to 250. Lower the Vertical Content Compression Resistance Priority to 749. You will learn what these Auto Layout properties do in Chapter 13.

f) Your frames might be misplaced, so select the three labels and click the Update Frames button.

9. In your ItemCell.swift file, create 3 IBOutlets to the labels you created in your storyboard.

10. In your ItemsViewController.swift file, in the viewDidLoad, set the table view row height.

11. Update your cellForRowAt method to use your new ItemCell (set the name, serial number, and value in dollars lables).

12. Open Main.storyboard and update the labels to use the Dynamic Type text styles instead of fixed fonts. Select the nameLabel and valueLabel and open the attributes inspector. Click on the text icon to the right of Font. For Font, choose Text Styles - Body. Repeat the same steps for the serialNumberLabel, choosing the Caption 1 text style.

13. Now let's change the preferred font size. You do this through the Settings application. Build and run the application. From the simulator's Hardware menu, select Home. Next, on the simulator's Home screen, open the Settings application. Choose General, then Accessibility, and then Larger Text. (On an actual device, this menu can also be accessed in Settings via Display & Brightness → Text Size.) Drag the slider all the way to the left to set the font size to the smallest value.

14. Open ItemCell.swift and override awakeFromNib() to have the labels automatically adjust.

Final App