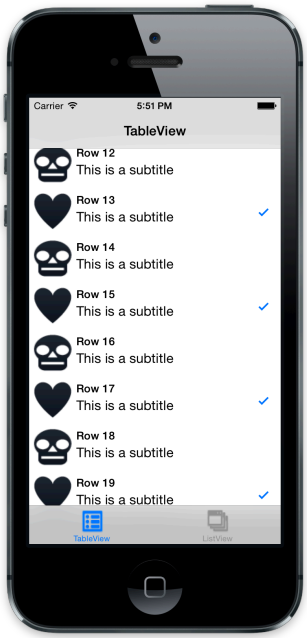
04 - TableViews and ListViews

Mission: In this lab, you will implement both a TableView and a ListView to compare features, ease of use, and performance.



Specification: To successfully complete this activity, you will:

- Complete a two-tabbed application that implements both TableView and ListView
- Manage eventing with both components
- Compare the performance of the TableView and ListView components

1. Download the starting file set 04_ListViews and import the resulting files as a new Titanium Mobile project. Update the SDK and other details of the tiapp.xml as necessary.
2. Build the project to view its starting state. The app features two tabs, one with a TableView and the other with a ListView. Both tables are empty at this point.
3. Working first with the tabletab controller, implement these features:
    a. Create a loop that creates 50 table rows (use anonymous objects), creating a new instance of the tablerow controller for each. Pass data to createController() so that:
        i. The heading of each row is "Row #" where # is the row number
        ii. The subheading is "This is a subtitle"
        iii. Each row has a custom property named i equal to the loop counter value (logic in tablerow.js will create the alternating left image)
    b. Add a click event listener to your table. On each click:
        i. Log a message to the console stating that "Row # was clicked" where # is the row number
        ii. If the user clicks on the image, change the image to show the /dark_star.png graphic
4. Build and test your table and event listener. It should match the demo your instructor presented.
5. Working with the listtab controller, implement these features:
    a. Create a loop that creates 50 objects, each with the following details:
        i. a heading object, whose text property is "Item #" where # is the row number
        ii. a subheading, whose text property is "This is a subtitle"
        iii. a leftimage object, whose image property is "/dark_heart.png" for odd rows and "/dark_skull.png" for even rows
        iv. a template property, whose value is "odd" for odd rows and "even" for even rows
    b. Add an itemclick event listener to your list. On each click:
        i. Log a message to the console stating that "Item # was clicked" where # is the row number
        ii. If the user clicks on the image, change the image to show the /dark_star.png graphic. You will need a reference to the item, to update its leftimage.image property. Don't forget to update the item by calling the section's updateItemAt() method
6. Build and test your list and event listener. It should match the demo your instructor presented.
7. Increase the number of rows created to 5000. Test your app, preferably

on a device. Do you notice any difference in how long the app takes to load? Is scrolling performance on the table different?

8. Test your app, preferably on a device. Do you notice any difference in how long the app takes to load? Is scrolling performance on the table different?
9. Increase the number of list items created to 5000. Test your app again. Do you notice any difference in how long the app takes to load? Is scrolling performance on the list different? Is scrolling performance similar or different for the table and list?

Summary

In this lab, you implemented both a TableView and a ListView to compare features, ease of use, and performance.

Resources

API docs: ListView: http://docs.appcelerator.com/titanium/latest/#!/api/Titanium.UI.ListView
API docs: TableView: http://docs.appcelerator.com/titanium/latest/#!/api/Titanium.UI.TableView
Guides: ListViews: http://docs.appcelerator.com/titanium/latest/#!/guide/ListViews
Finished code: 04_ListView