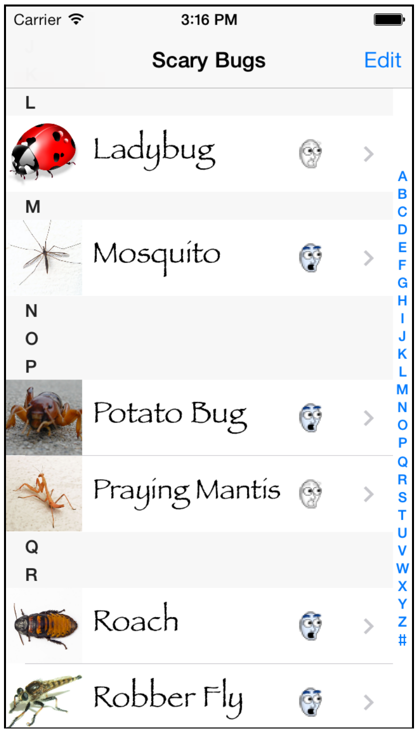
Table Views in iOS

Hands-On Challenges

Table Views:
Beginning to Advanced
Hands-On Challenges

Challenge #10: Indexing

As your list of bugs get larger and larger, it would be nice if you gave your users a way to quickly skip to different sections in your table. You can do this in iOS through a cool feature called indexing:



Your challenge is to add this feature to your Scary Bugs app. Note that this time, your starter project is **not the same as in the last challenge**. I have provided a new starter project for you that is a refactored version of the previous challenge; the big changes are the ability to move and add new cells have been removed, in preparation for the indexing feature you'll be adding in this challenge.

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

Note: You're getting more advanced at this point, so this time I am not listing out each and every step so it's a bit more of a challenge. If you get stuck, refer back to the video or check out the challenge solution. Good luck!

Open the starter project, open **BugTableViewController.m**, and replace `setupBugs` with the following:

```
- (void)setupBugs {

  // 1) Split bugs into sections
  NSInteger sectionTitlesCount = [[[UILocalizedIndexedCollation
    currentCollation] sectionTitles] count];
  NSMutableArray *allSections = [[NSMutableArray alloc]
    initWithCapacity:sectionTitlesCount];
  for (NSInteger i = 0; i < sectionTitlesCount; i++) {
    [allSections addObject:[NSMutableArray array]];
  }

  // 2) Add each object to appropriate section
  NSMutableArray *bugs = [ScaryBug bugs];
  for (ScaryBug *bug in bugs) {
    NSInteger sectionNumber = [[UILocalizedIndexedCollation
    currentCollation] sectionForObject:bug
    collationStringSelector:@selector(name)];
    [allSections[sectionNumber] addObject:bug];
  }

  // 3) Sort each section
  self.bugSections = [NSMutableArray array];
  for (NSInteger i = 0; i < sectionTitlesCount; i++) {
    NSArray *curIcons = allSections[i];
    [self.bugSections addObject:[curIcons
    sortedArrayUsingSelector:@selector(compare:)]];
  }

}
```

This is important to understand, so let's go over this section by section:

1. For each "letter in the alphabet" provided by the `UILocalizedIndexedCollation`, you'll want to store an array of bugs for that letter. So `allSections` is an array that contains an array for each letter, with the bugs for that letter.

2. Loop through all of the bugs, and use a helper method on `UILocalizedIndexedCollation` to find the appropriate section for that bug, based on its name. Add it to the appropriate array.

3. Loop through all of the arrays, and sort each array by the bug's name so they are all in alphabetical order. Set `bugSections` to the result.

For this to work, you need to add a `compare:` method to **ScaryBug.m**. See if you can do this yourself based on what you learned from the lecture.

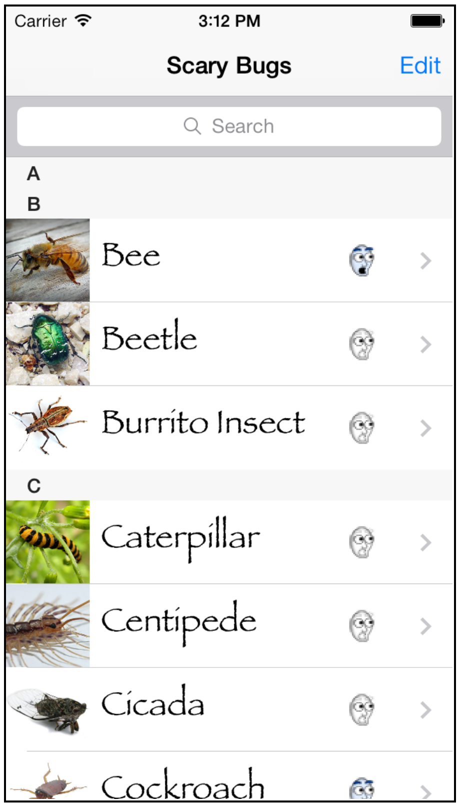
Back in **BugTableViewController.m**, replace `tableView:titleForHeaderInSection:` with the following:

```
- (NSString *)tableView:(UITableView *)tableView
  titleForHeaderInSection:(NSInteger)section {
  if (tableView == self.searchDisplayController.searchResultsTableView)
  {
    return nil;
  } else {
    return [[UILocalizedIndexedCollation currentCollation]
      sectionTitles][section];
  }
}
```

Before, each `BugSection` contained the name of the section, so you could just use that. But now you're not storing the letter associated with each section anywhere. However that's no problem, since the list of letters doesn't change – you just ask `UILocalizedIndexCollection` for the list, and index it by the section.

Next, you need to do a bit of refactoring. Anywhere that you index self.bugSections (i.e. self.bugSections[section]), you need to make sure that you cast the result as an `NSMutableArray`, not a `BugSection`. Similarly, you will want to make sure the code uses `bugSection` for the array of bugs, not `bugSection.bugs`. Go through the code and replace all instances of this.

Build and run, and if you got it right you should see the bugs are now split up by letter:



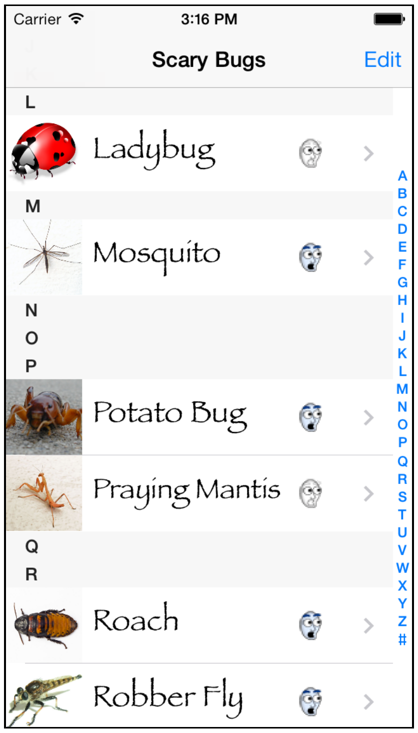
Now that you have this done, adding an index is easy! Just add these lines to **BugTableViewController.m**:

```
- (NSArray *)sectionIndexTitlesForTableView:(UITableView *)tableView {
  if (tableView == self.searchDisplayController.searchResultsTableView)
  {
    return nil;
  } else {
    return [[UILocalizedIndexedCollation currentCollation]
      sectionIndexTitles];
  }
}

- (NSInteger)tableView:(UITableView *)tableView
sectionForSectionIndexTitle:(NSString *)title
               atIndex:(NSInteger)index
{
  return [[UILocalizedIndexedCollation currentCollation]
    sectionForSectionIndexTitleAtIndex:index];
}
```

The first method just returns the indices to show in the sidebar, and the second method returns the section for a given index. Both of these are as simple as calling a method on the `UILocalizedIndexCollation`.

Build and run, and enjoy your new index!



Bonus #1: See if you can modify your table view to remove the empty sections, as shown in the video.

Bonus #2: See if you can add the search icon to the top of the index list, as shown in the video.