CS 696 Mobile Application Development
Fall Semester, 2010
Doc 14 Tables & Navigation
Oct 14, 2010

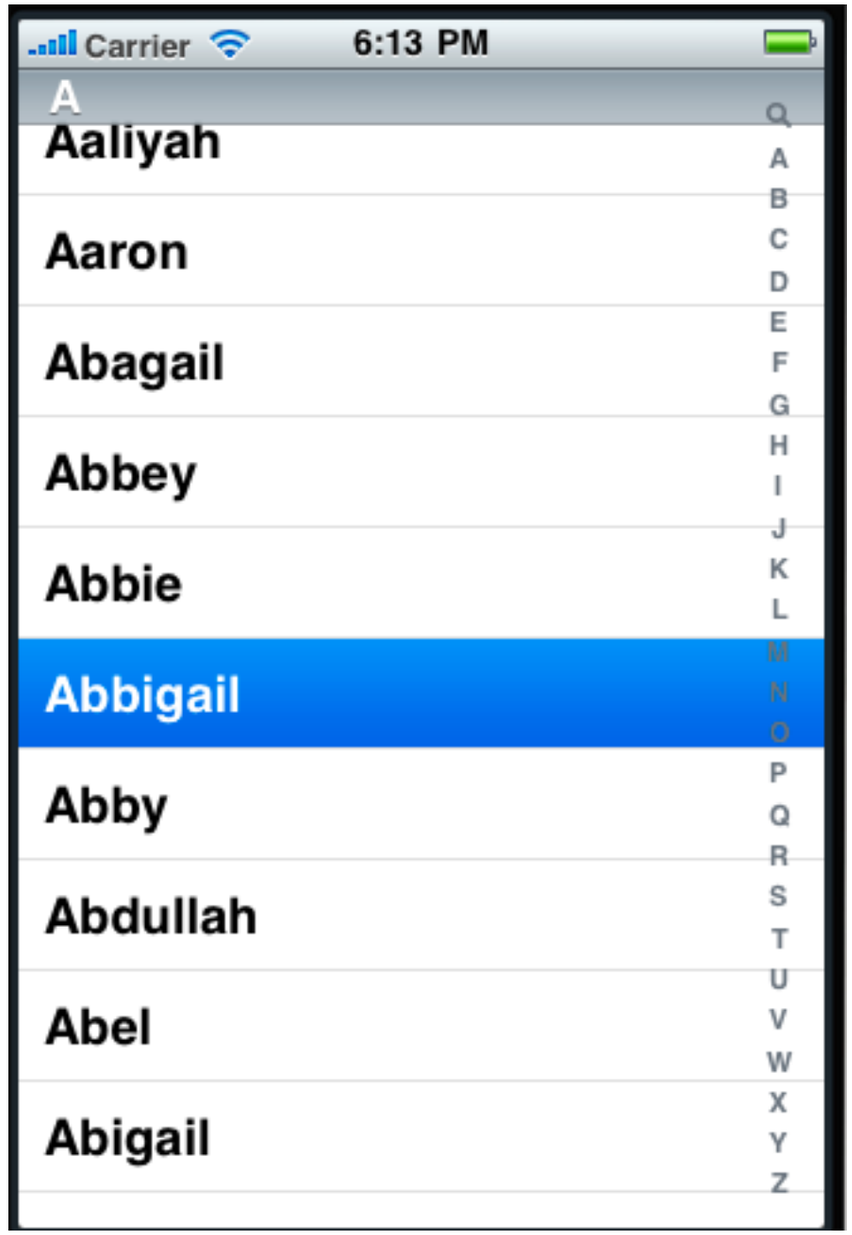
References

Stanford iPhone Course, Winter 2010, CS193P - Lecture 7

Beginning iPhone 3 Development: Exploring the iPhone SDK by Jeff LaMarche, and David Mark

Table Basics

UITableViewDelegate
UITableViewDataSource
NSIndexPath



UITableViewDataSource

Table only asks for data needed for current view

- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
Optional

- (NSInteger)tableView:(UITableView *)tableView
numberOfRowsInSection:(NSInteger)section

- (UITableViewCell *)tableView:(UITableView *)tableView
cellForRowAtIndexPath:(NSIndexPath *)indexPath

Data & Memory

Table does not need all the same time

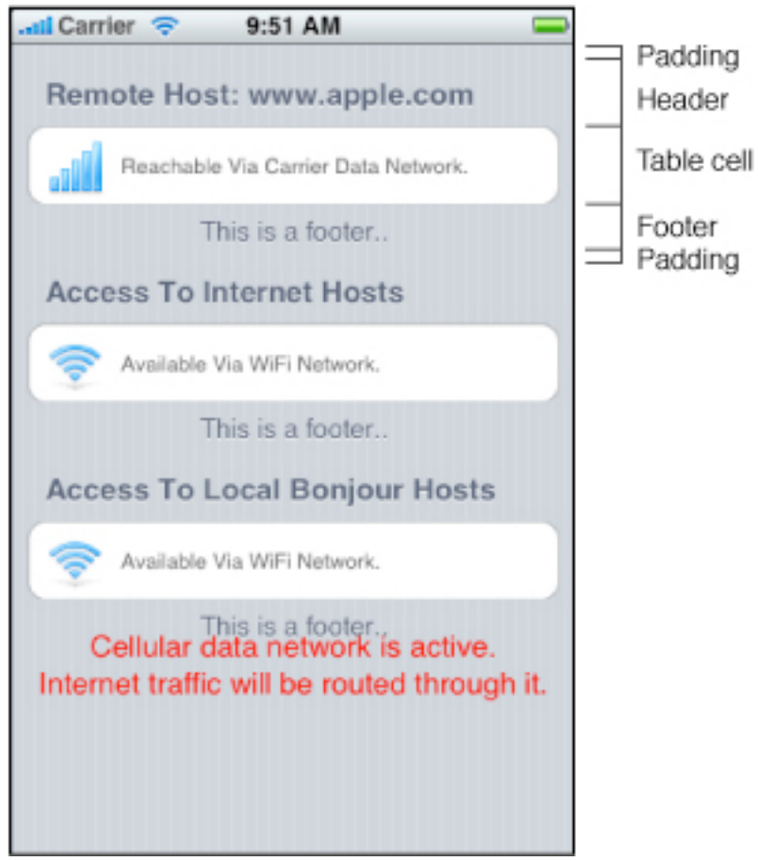
For large datasets your program does not need to load all data

Scrolling slows down when reading data

Section Header & Footer

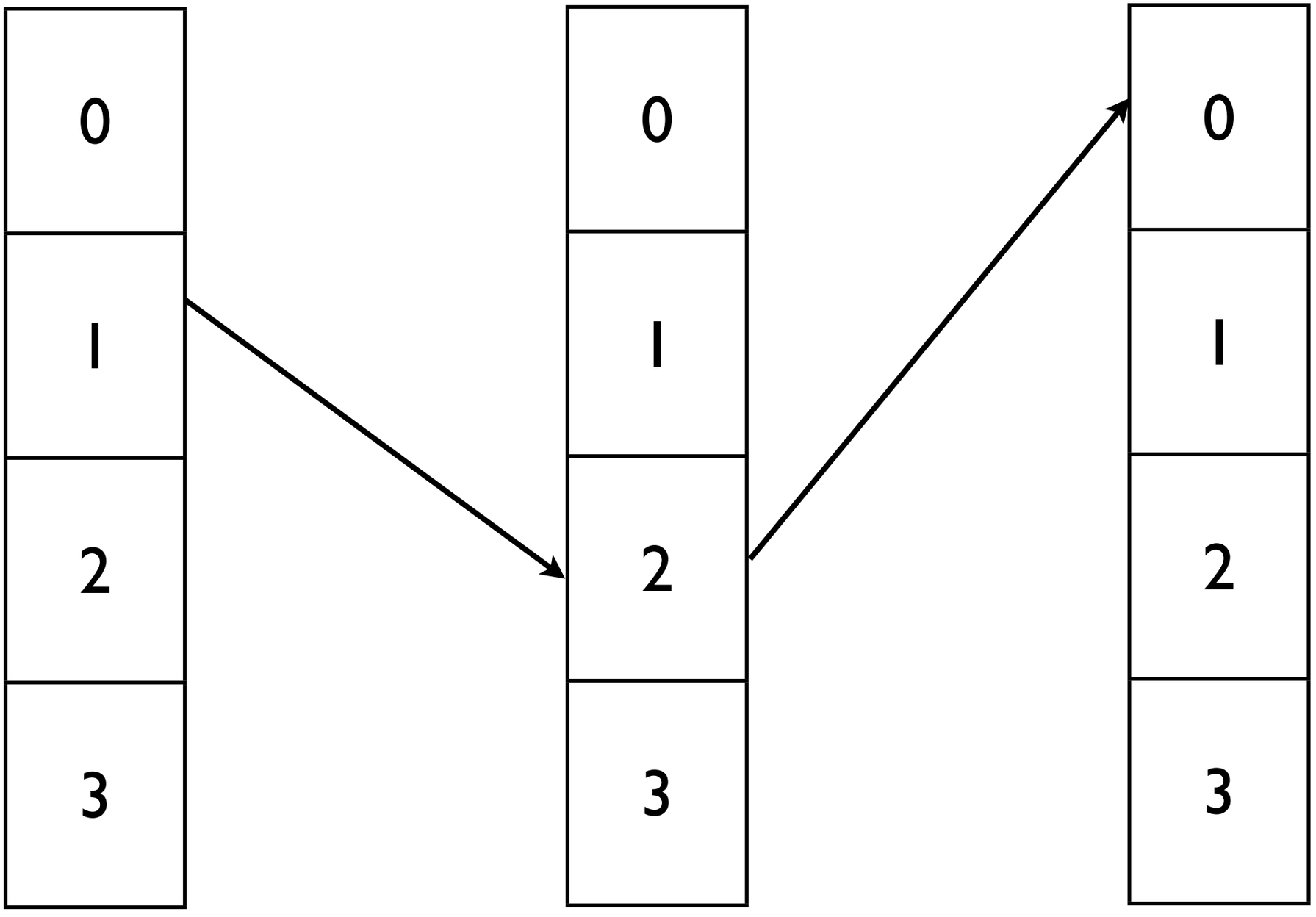
- (NSString *)tableView:(UITableView *)tableView
titleForHeaderInSection:(NSInteger)section

- (NSString *)tableView:(UITableView *)tableView
titleForFooterInSection:(NSInteger)section



NSIndexPath

Path in a tree of nested arrays



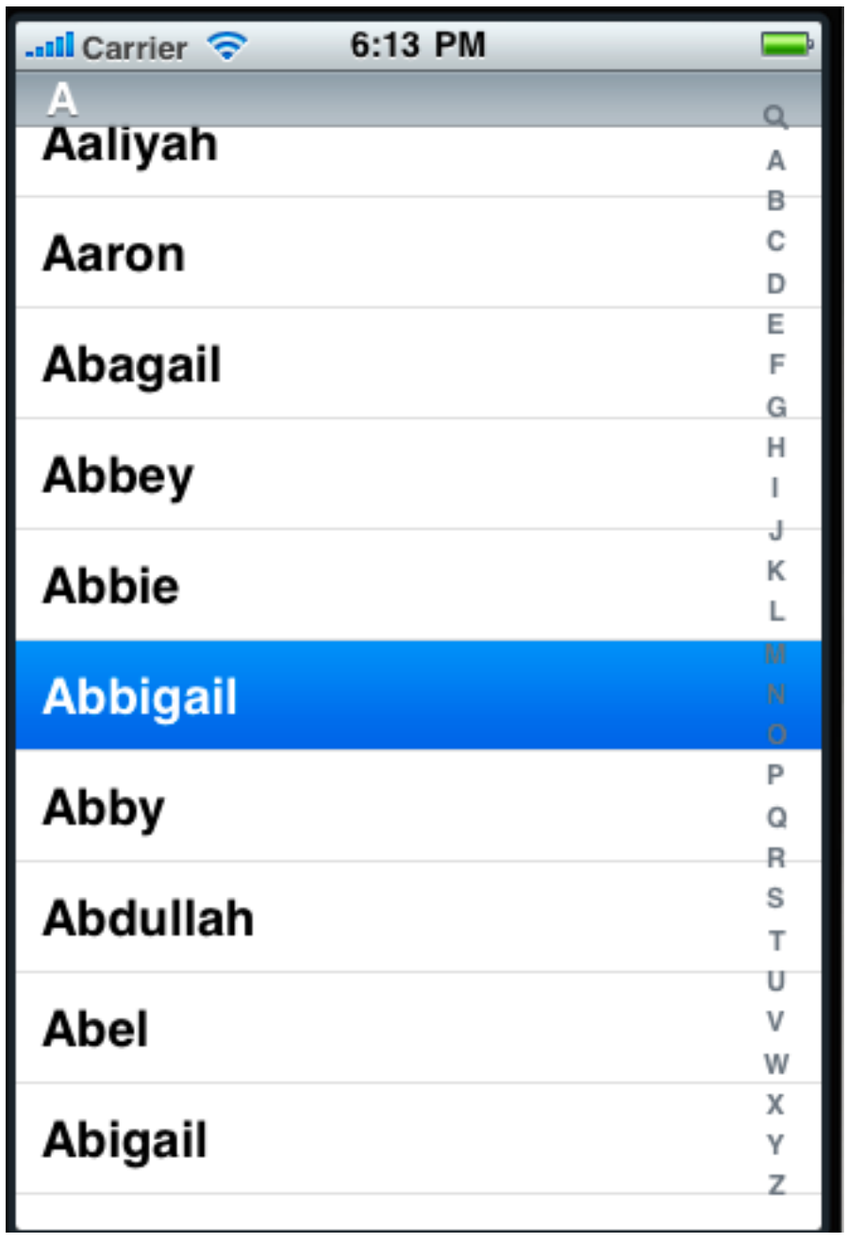
Category with helper methods

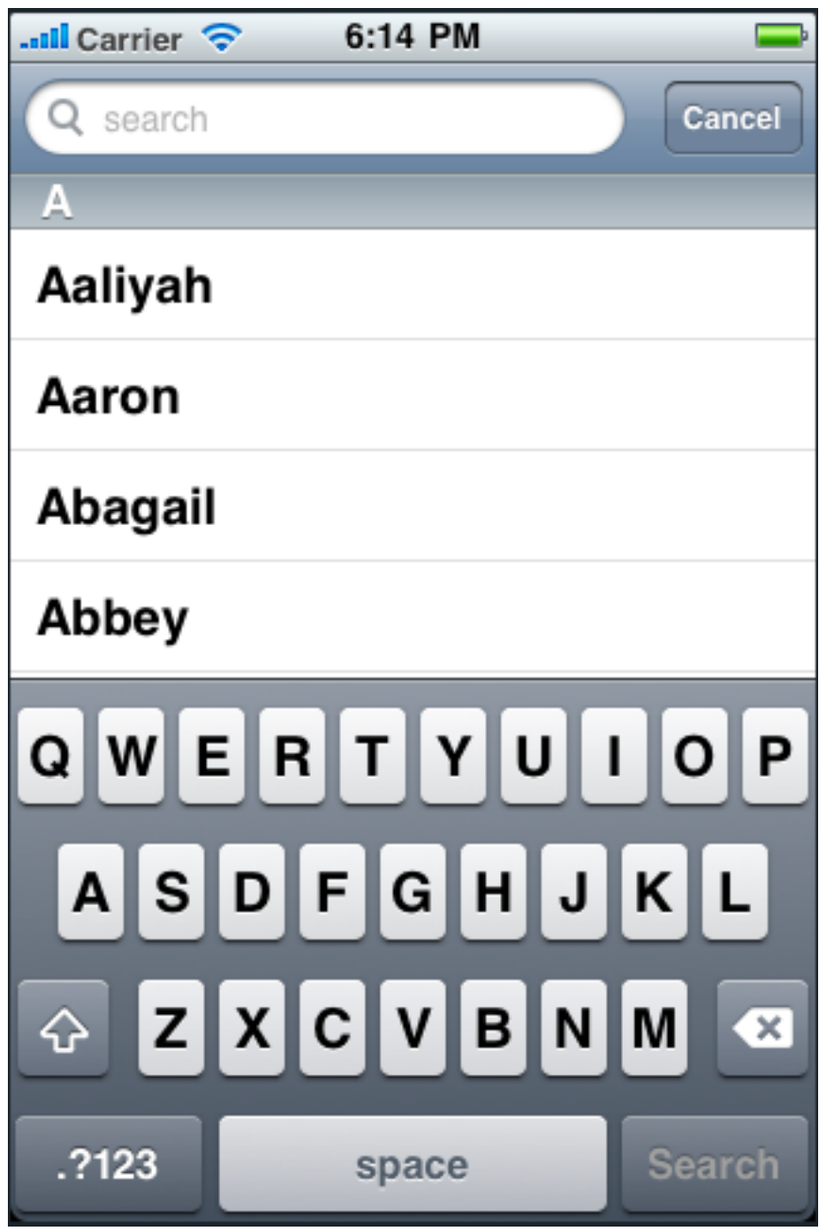
```
@interface NSIndexPath (UITableView)
 + (NSIndexPath *)indexPathForRow:(NSUInteger)row
         inSection:(NSUInteger)section;

@property(nonatomic,readonly) NSUInteger section;
@property(nonatomic,readonly) NSUInteger row;
@end
```

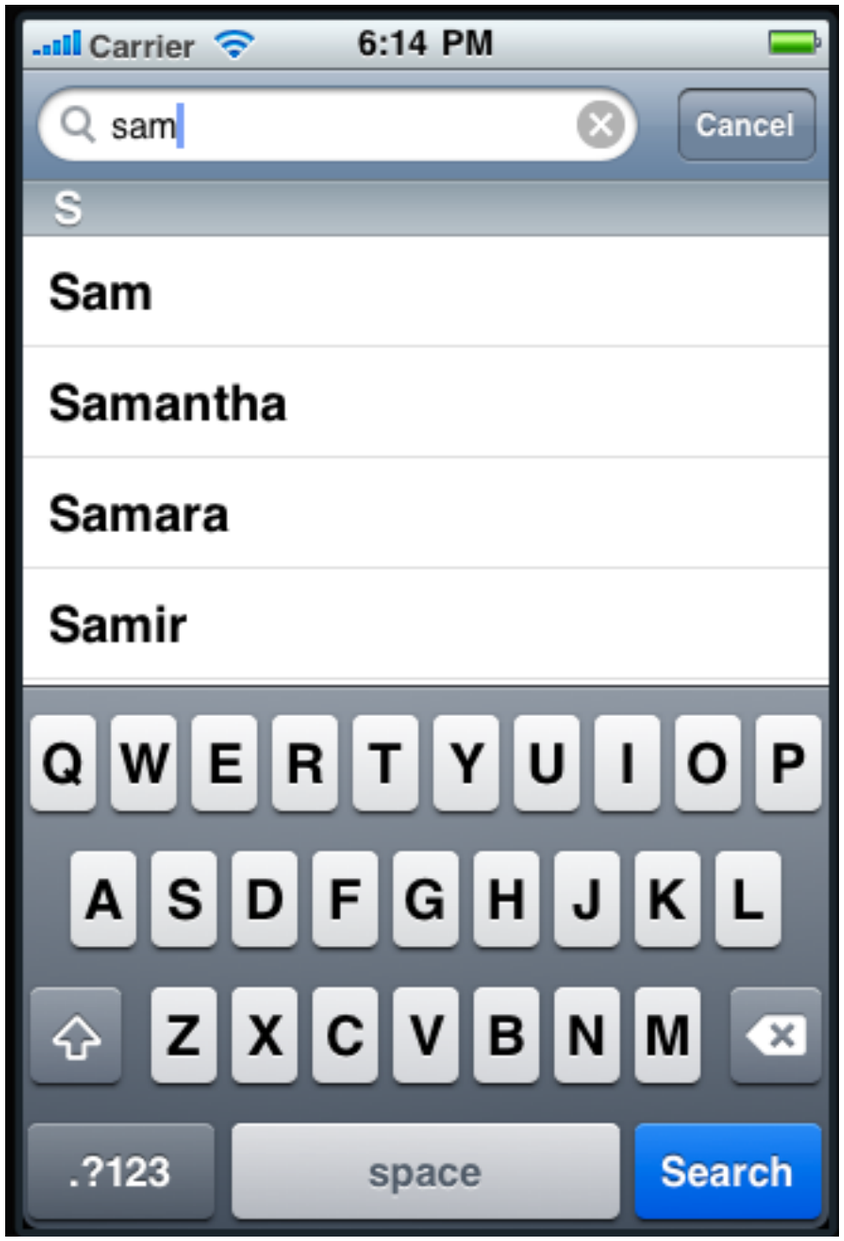
Reason For Helper methods

```
- (UITableViewCell *)tableView:(UITableView *)tableView
        cellForRowAtIndexPath:(NSIndexPath *)indexPath {
```

Tables & Search





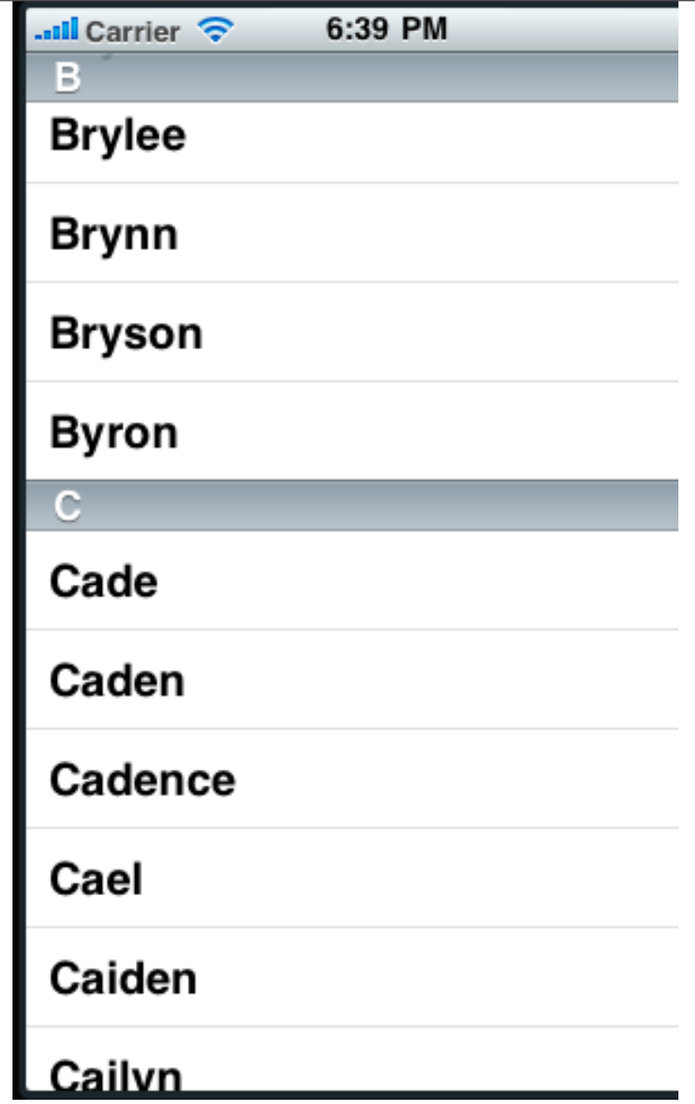


Sections

TableView Data Source methods

- (NSInteger) numberOfSectionsInTableView:
    (UITableView *) tableView

- (NSInteger) tableView: (UITableView *) tableView
    numberOfRowsInSection: (NSInteger) section



- (NSArray *) sectionIndexTitlesForTableView: (UITableView *) tableView

- (NSString *) tableView: (UITableView *) tableView
    titleForHeaderInSection: (NSInteger) section

Organizing the data

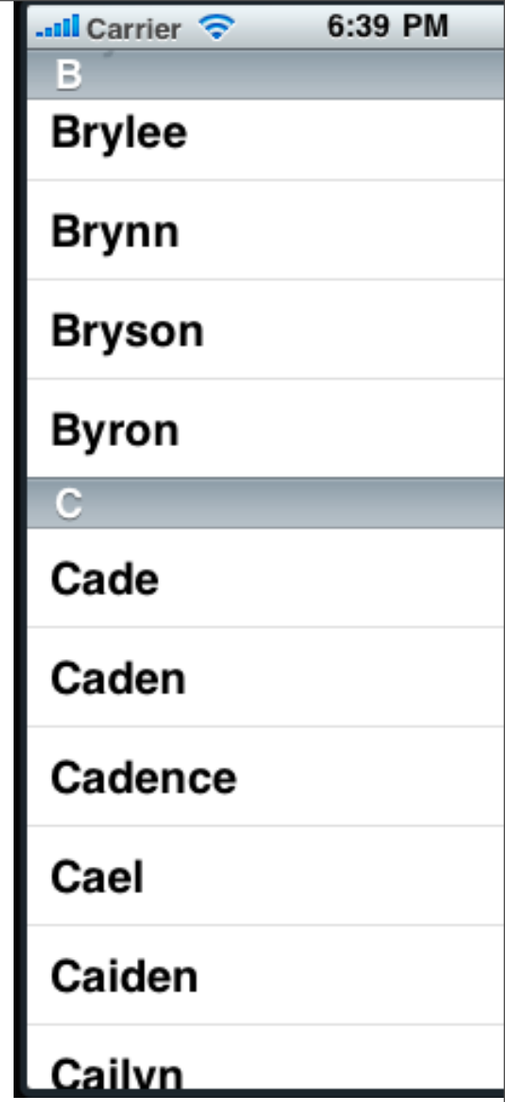
Depends on your data

Dictionary

Key: Section label (letters in alphabet)
Value: Array of item in section (array of names)

Sections

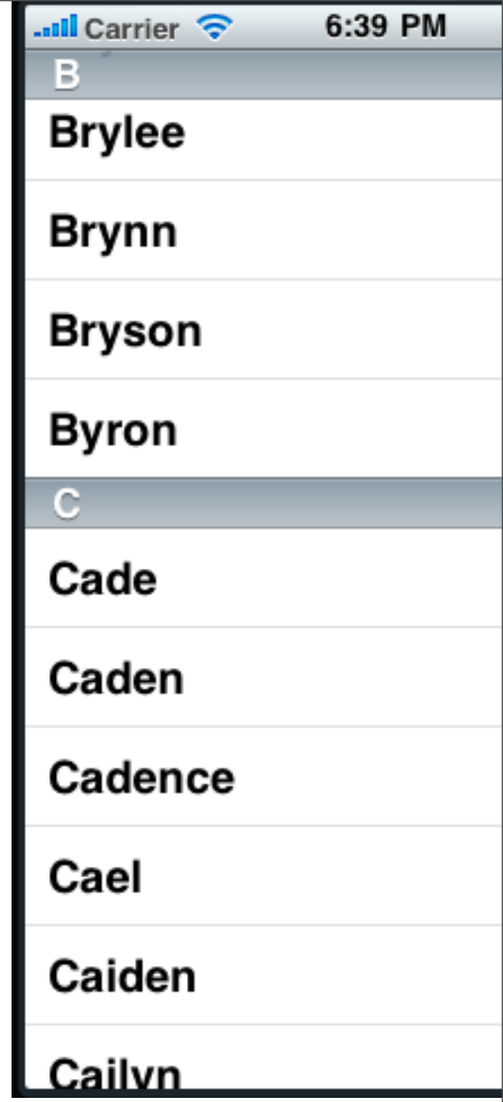
```
- (NSInteger) numberOfSectionsInTableView:
    (UITableView *) tableView

        return [[dataDictionary allKeys] count]

- (NSString *) tableView: (UITableView *) tableView
     titleForHeaderInSection: (NSInteger) section

          return [[dataDictionary allKeys] objectAtIndex: section]
```




```
- (NSInteger) tableView: (UITableView *) tableView
     numberOfRowsInSection: (NSInteger) section

          keys = [[dataDictionary allKeys]
     sectionData = [[dataDictionary allKeys] objectAtIndex: section]
     return [sectionData count]
```

Sections

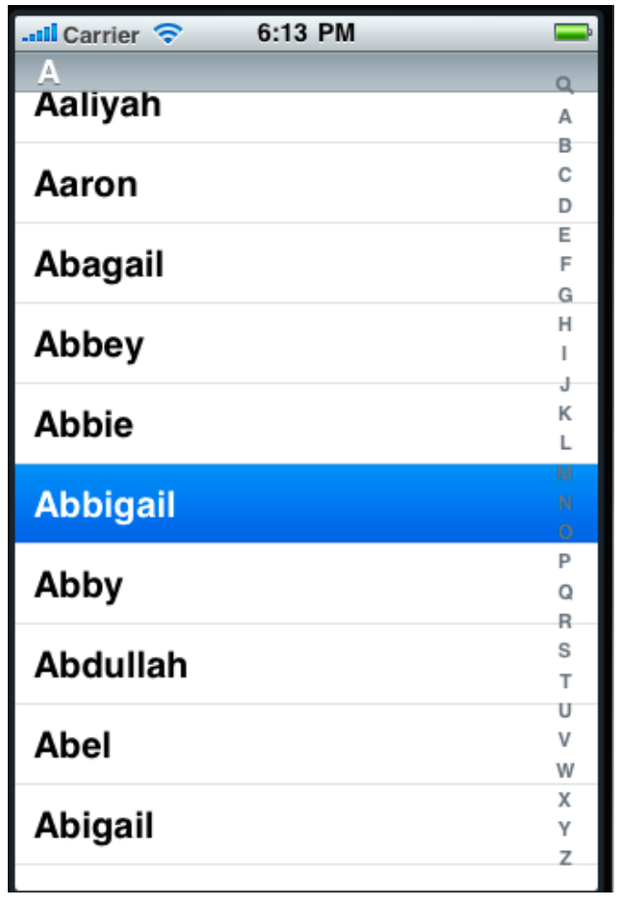
```
- (NSInteger) tableView: (UITableView *) tableView
      numberOfRowsInSection: (NSInteger) section

                keys = [[dataDictionary allKeys]
          sectionData = [[dataDictionary allKeys] objectAtIndex: section]
          return [sectionData count]
```



Index

TableView needs to know

- Index labels
- Number of rows per index




```
- (NSArray *) sectionIndexTitlesForTableView: (UITableView *) tableView

        return [dataDictionary allKeys]
```

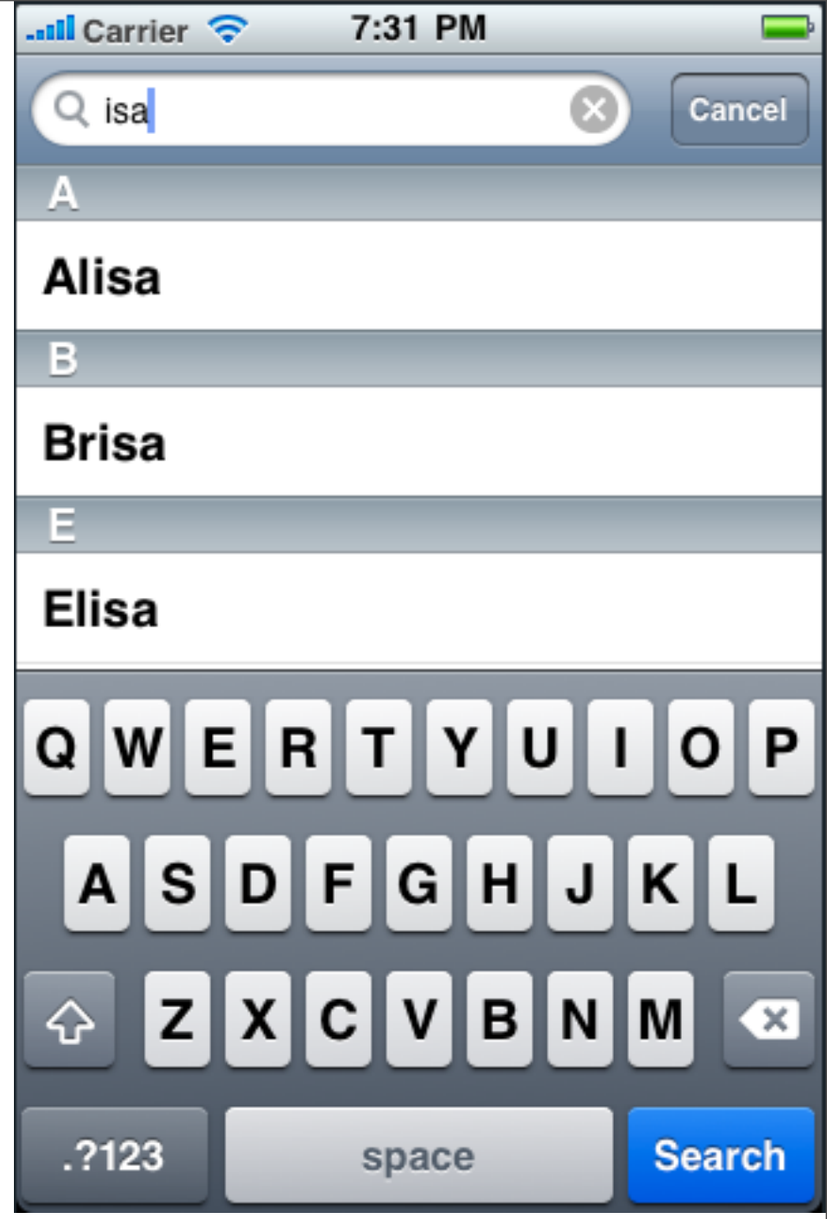
Search - Restricting data

As user types

- Remove items that do not match
- Update table to show restricted data

Use mutable copy of data

So can remove data that does not match



Mutable Deep Copy

```
@implementation NSDictionary(DeepMutableCopy)
-(NSMutableDictionary *)mutableDeepCopy
{
    NSMutableDictionary *ret = [[NSMutableDictionary alloc] initWithCapacity:[self count]];
    NSArray *keys = [self allKeys];
    for (id key in keys)
    {
        id oneValue = [self valueForKey:key];
        id oneCopy = nil;

        if ([oneValue respondsToSelector:@selector(mutableDeepCopy)])
            oneCopy = [oneValue mutableDeepCopy];
        else if ([oneValue respondsToSelector:@selector(mutableCopy)])
            oneCopy = [oneValue mutableCopy];
        if (oneCopy == nil)
            oneCopy = [oneValue copy];
        [ret setValue:oneCopy forKey:key];
    }
    return ret;
}
@end
```

Search Bar delegate Methods

- (void)searchBarSearchButtonClicked:(UISearchBar *)searchBar
    NSString *searchTerm = [searchBar text];
    restrict data, update table view

- (void)searchBarTextDidBeginEditing:(UISearchBar *)searchBar
    update table view

- (void)searchBar:(UISearchBar *)searchBar textDidChange:(NSString *)searchTerm
    restrict data, update table view

- (void)searchBarCancelButtonClicked:(UISearchBar *)searchBar
    Reset initialize data to show all, update table view

Updating Table

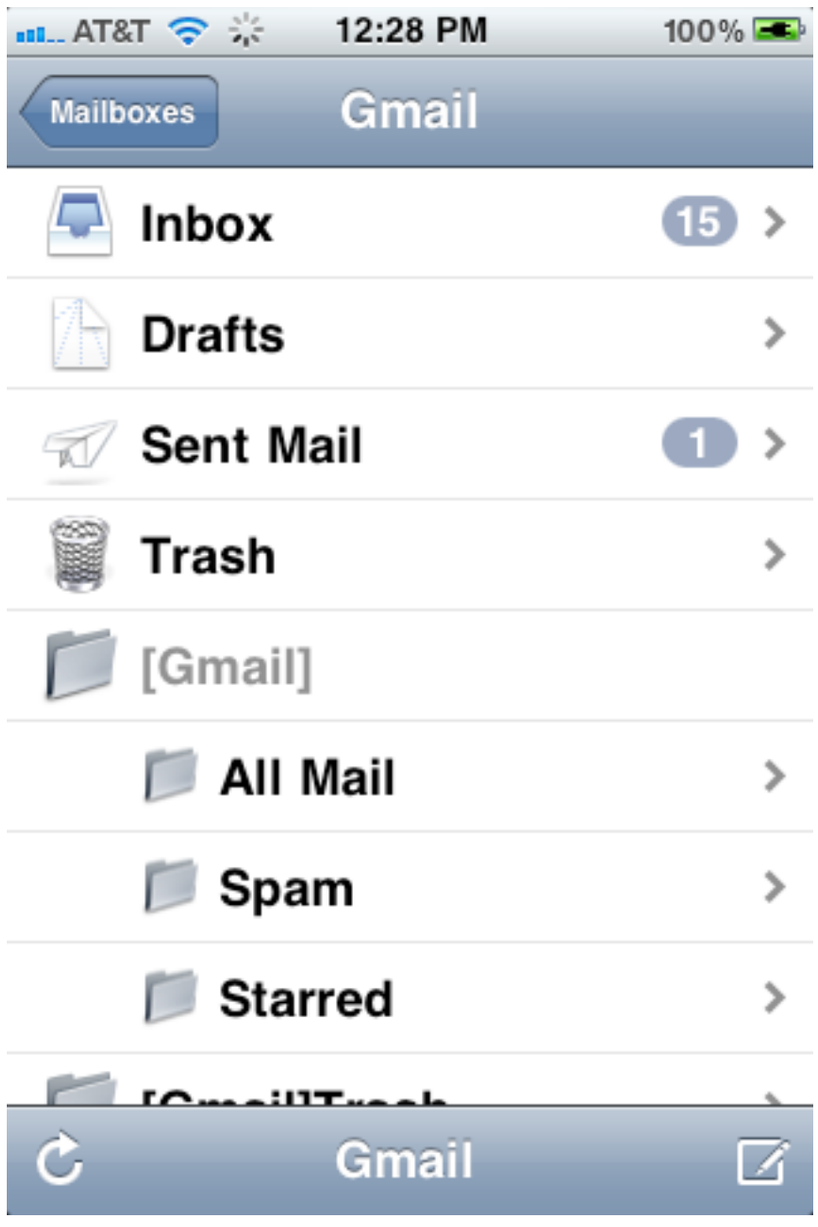
[table reloadData];

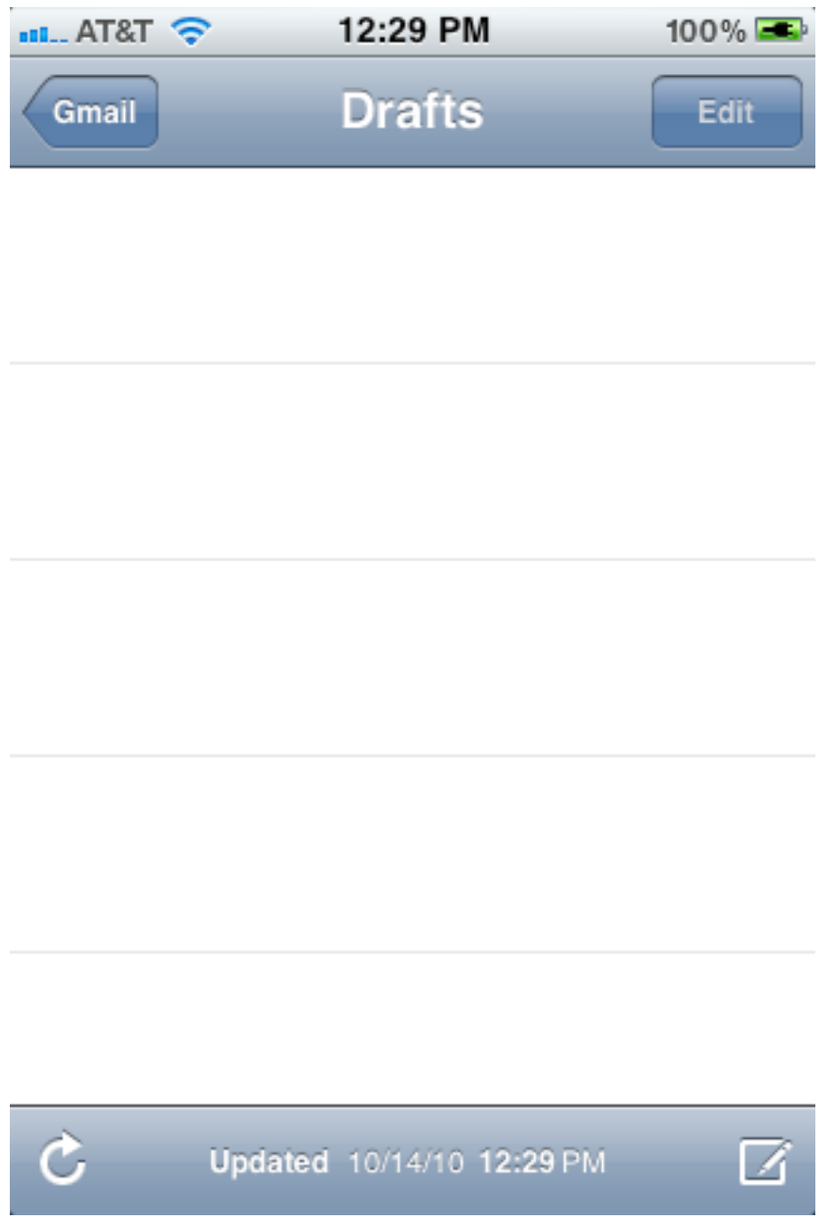
Table then calls TableViewDelegate methods to get current data

Navigation

Navigation


AT&T
12:28 PM
100%
Mailboxes
All Inboxes
117
POP:whitney@c...
102
Gmail
15
rewhitney@me.com
Accounts
POP:whitney@c...
102
Gmail
15
rewhitney@me.com
Updated 10/14/10 12:26 PM


AT&T
12:28 PM
100%
Mailboxes
Gmail
Inbox
15
Drafts
Sent Mail
1
Trash
[Gmail]
All Mail
Spam
Starred
Gmail


AT&T
12:29 PM
100%
Gmail
Drafts
Edit
Updated 10/14/10 12:29 PM

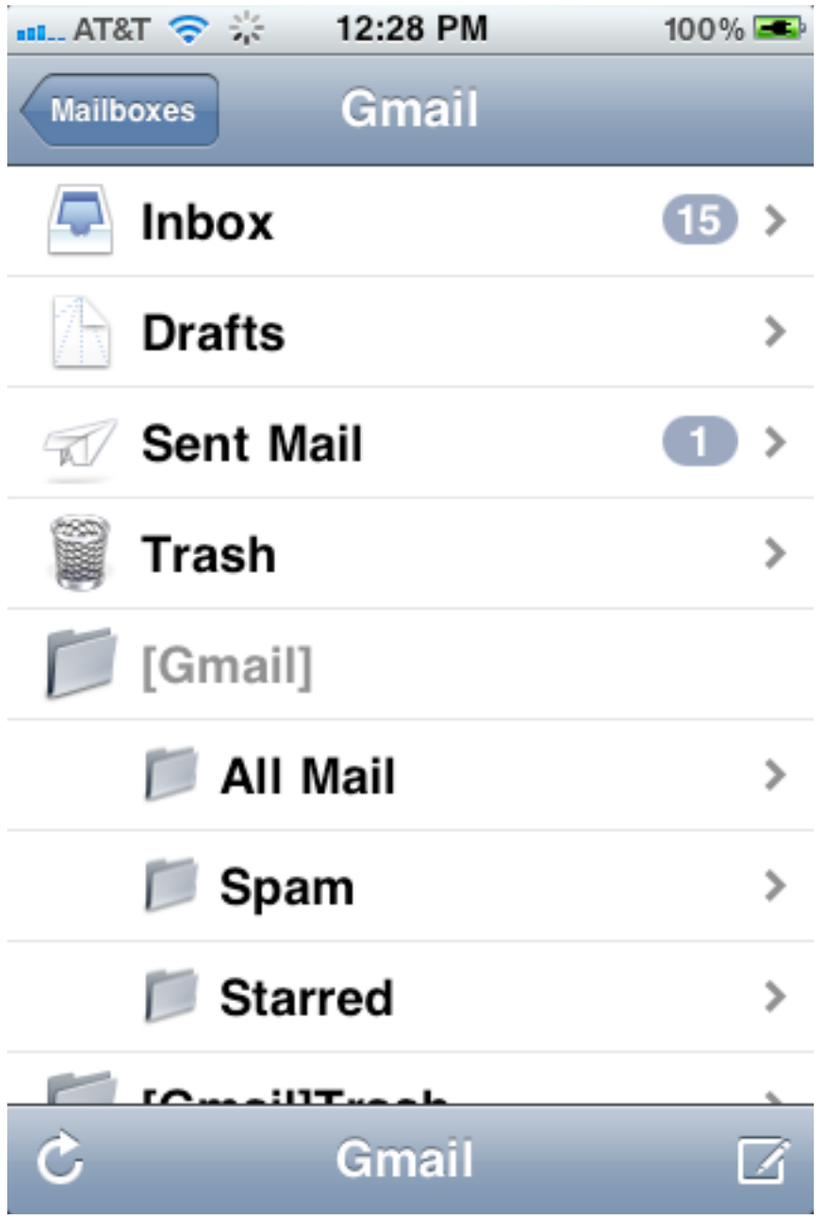
UINavigationController

Navigation Bar

Stack of views with UITableViewControllers

Push & pop controllers

NavigationController -> Mailboxes -> Gmail



Pushing & Popping -UINavigationController

- (void)pushViewController:(UIViewController *)viewController
    animated:(BOOL)animated;

In your code (UITableViewController subclass)

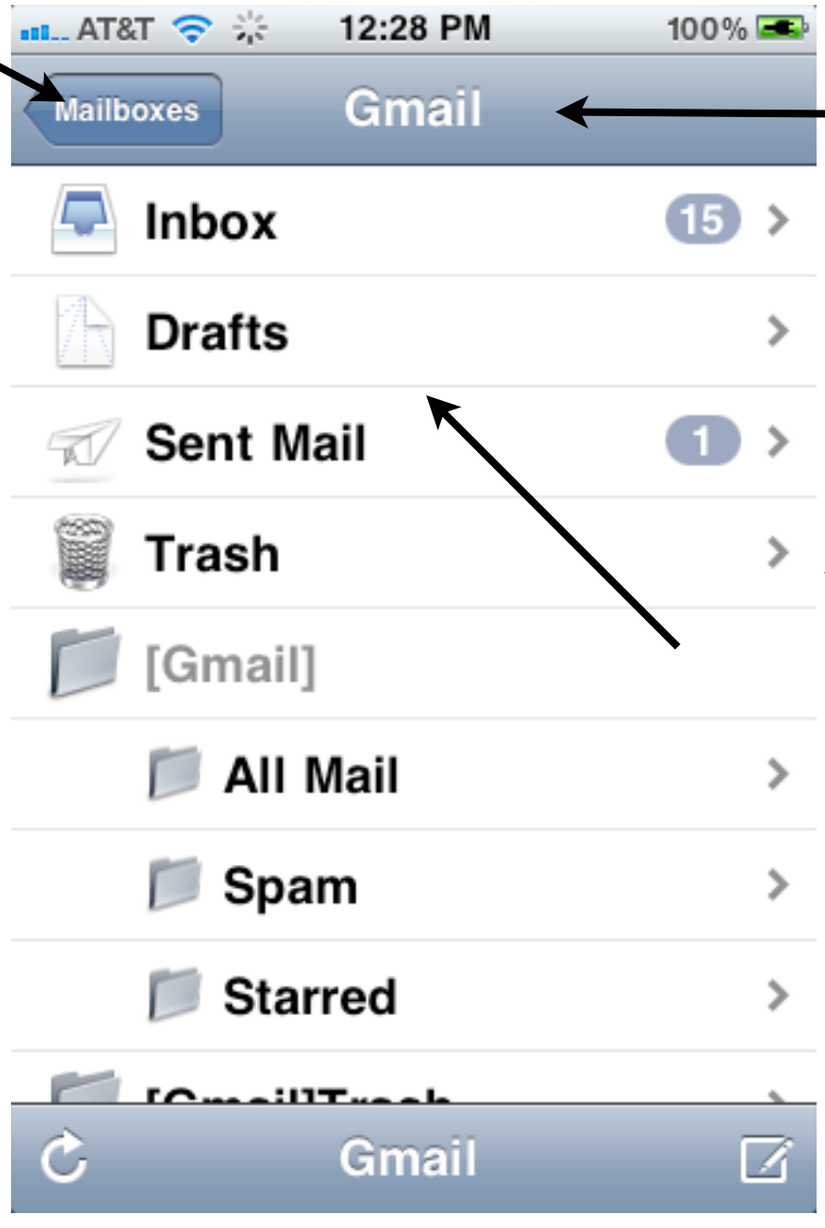
[self.navigationController pushViewController:nextController
    animated:YES];

- (UIViewController *)popViewControllerAnimated:(BOOL)animated;

Done for you went Back button pressed

The Parts

Previous view controller's title



Top view controller's title

Top view controller's view

Top view controller's toolbar

optional

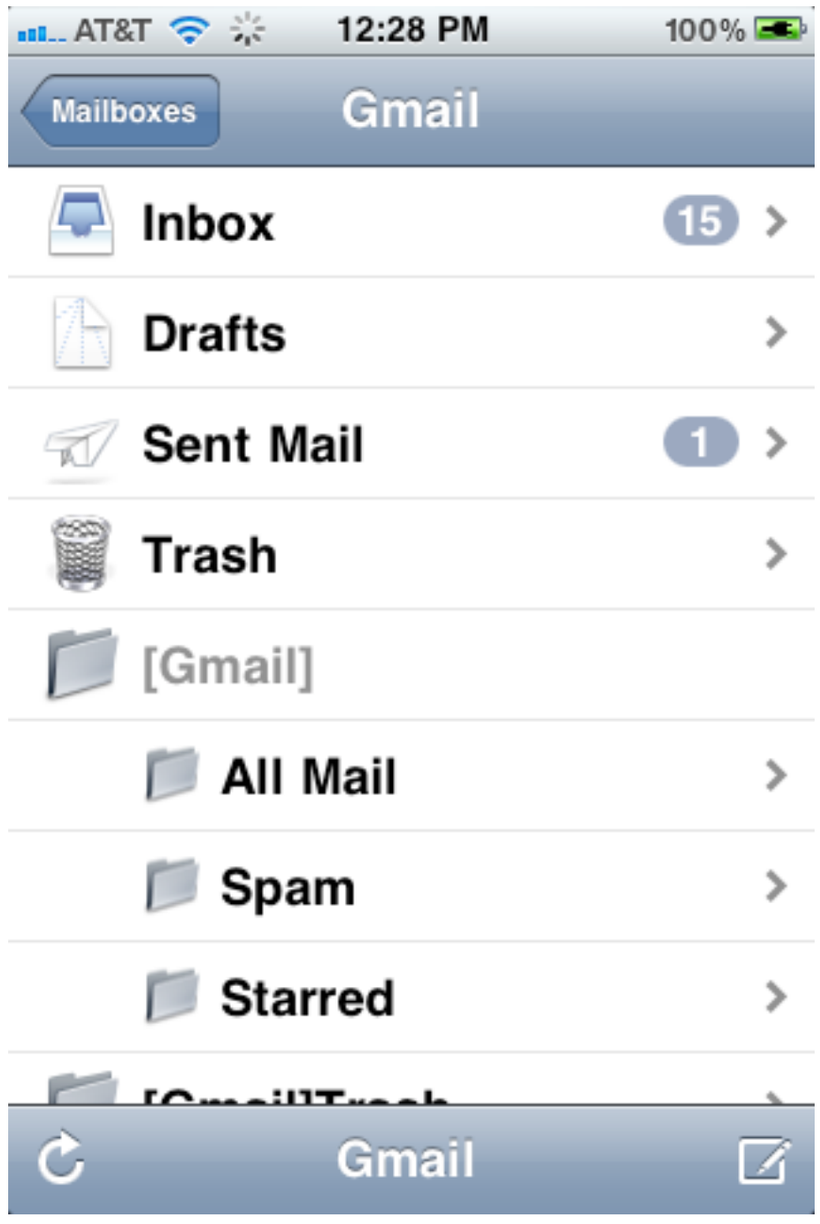
Titles

UITableViewController title property is used for

View title (current view Controller)

Back button title

(Previous view controller)



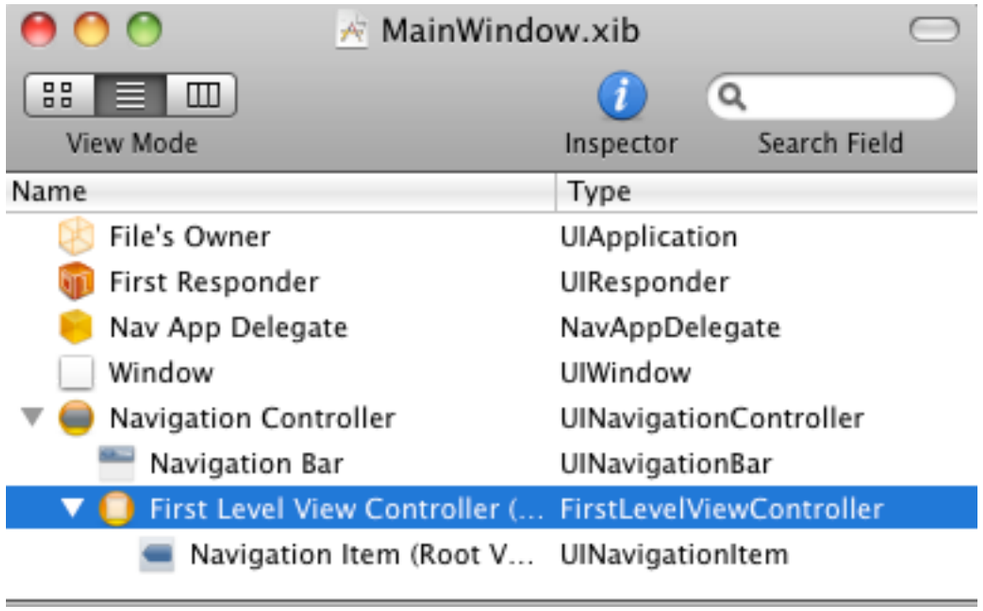

```
- (void)viewDidLoad {
  self.title = @"Gmail";
```

Navigation Controller

Needs view controller

Set in nib file

Or code



Customizing Navigation

Navigation bar has

leftBarButtonItem (backbarButtonItem)

Default - title of previous controller

titleView

Default - title of current controller

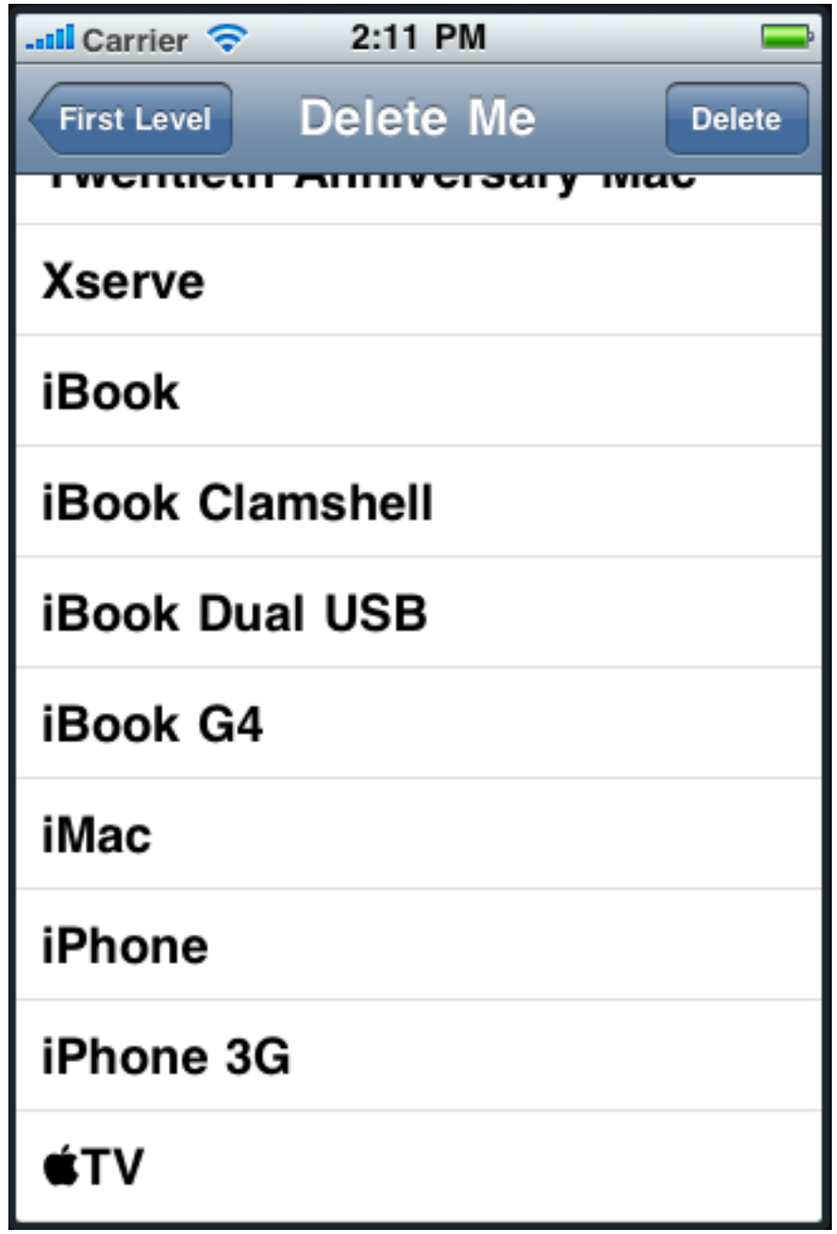
rightBarButtonItem

Default - empty

Create

Delete button

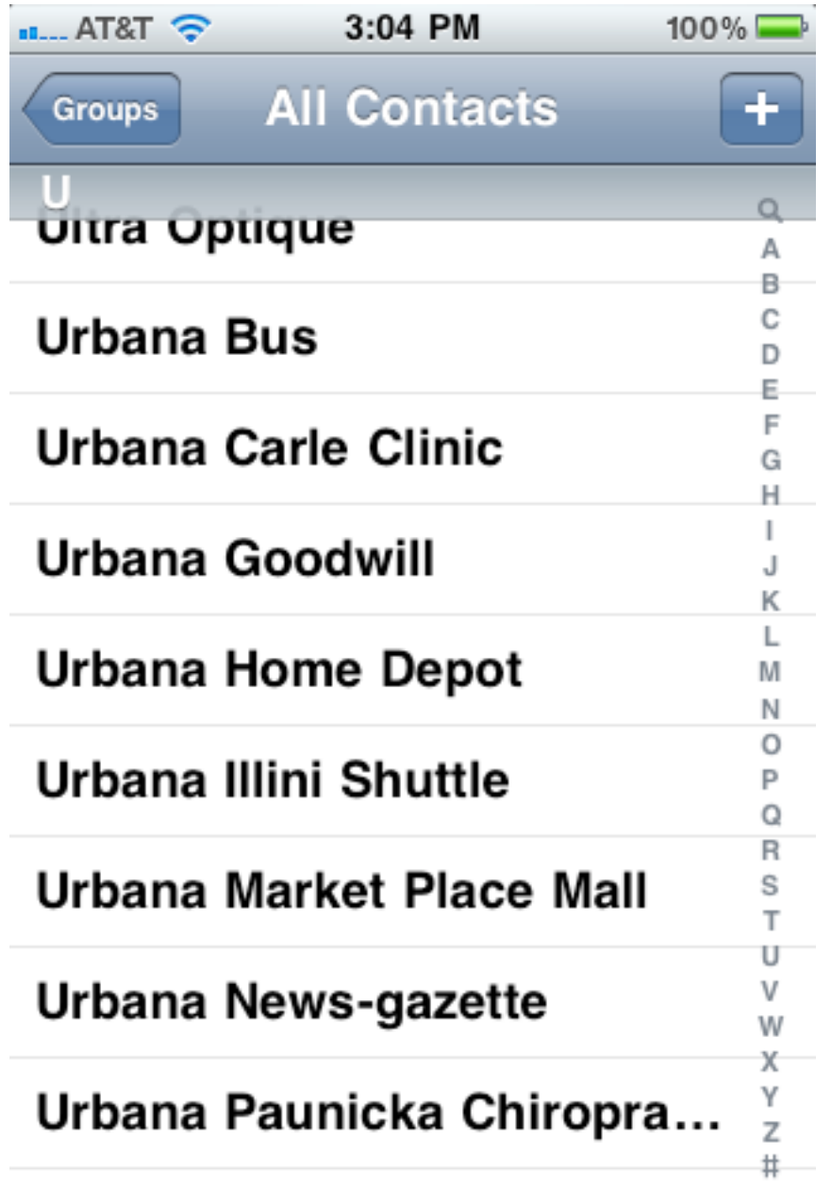
```
UIBarButtonItem *editButton = [[UIBarButtonItem alloc]
                               initWithTitle:@"Delete"
                               style:UIBarButtonItemStyleBordered
                               target:self
                               action:@selector(toggleEdit:)];
self.navigationItem.rightBarButtonItem = editButton;
[editButton release];
```



Edit/Done Button

Common Pattern

Shortcut setup

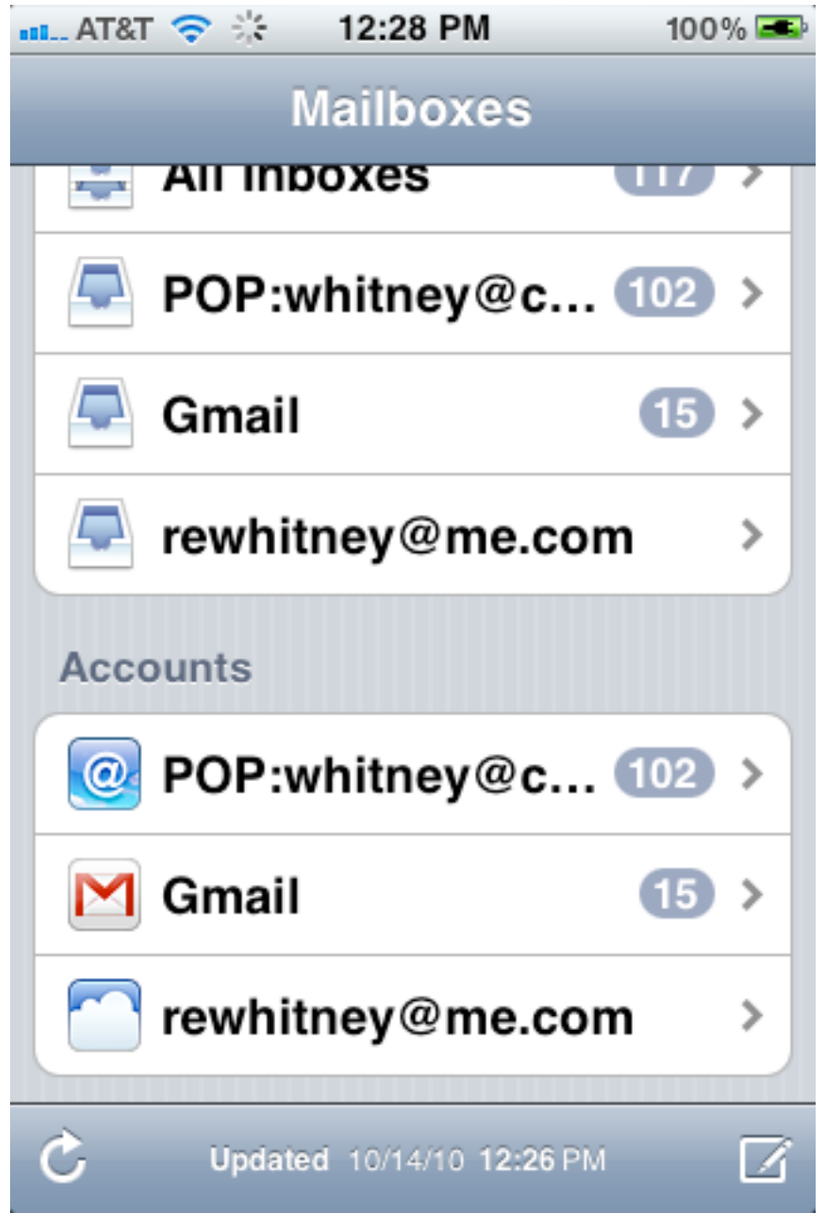



```
self.navigationItem.leftBarButtonItem =
    self.editButtonItem;


// Called when the user toggles the edit/done button
- (void)setEditing:(BOOL)editing animated:(BOOL)animated {


}
```

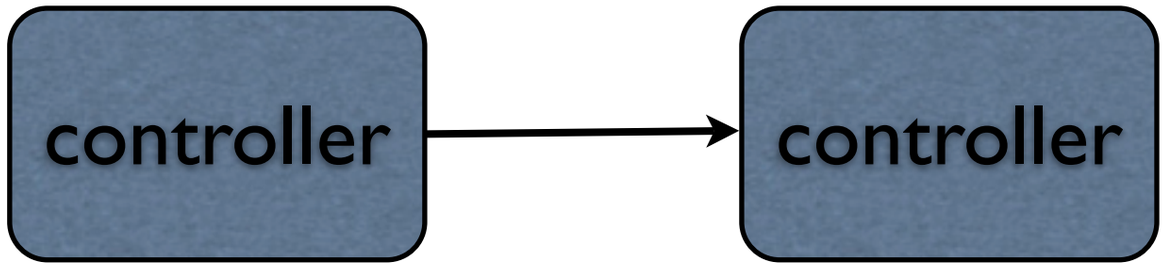
TabBar

If managed by a navigation controller
Use them to set toolbar items

- (void)setToolbarItems:(NSArray *)toolbarItems
    animated:(BOOL)animated



Passing Data



When one controller push another on stack
New controller may need information from first controller
Pass it to the new controller

```
[nextController dataINeed: [self selectedItem]

[self.navigationController pushViewController: nextController
    animated:YES];
```

More Details

View Controller Programming Guide for iOS Chapter 3 Navigation

UINavigationController class documentation