UITableViewDataSource Protocol Reference

User Experience: Tables

2010-05-24

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010

Simultaneously published in the United States and Canada.

Contents

UITableViewDataSource Protocol Reference

Conforms to	NSObject
Framework	/System/Library/Frameworks/UIKit.framework
Availability	Available in iOS 2.0 and later.
Companion guide	Table View Programming Guide for iOS
Declared in	UITableView.h
Related sample code	GKRocket

Overview

The `UITableViewDataSource` protocol is adopted by an object that mediates the application's data model for a `UITableView` object. The data source provides the table-view object with the information it needs to construct and modify a table view.

As a representative of the data model, the data source supplies minimal information about the table view's appearance. The table-view object's delegate—an object adopting the `UITableViewDelegate` protocol—provides that information.

The required methods of the protocol provide the cells to be displayed by the table-view as well as inform the `UITableView` object about the number of sections and the number of rows in each section. The data source may implement optional methods to configure various aspects of the table view and to insert, delete, and reorder rows.

Note: To enable the swipe-to-delete feature of table views (wherein a user swipes horizontally across a row to display a Delete button), you must implement the `tableView:commitEditingStyle:forRowAtIndexPath:` (page 9) method.

Many methods take `NSIndexPath` objects as parameters. `UITableView` declares a category on `NSIndexPath` that enables you to get the represented row index (`row` property) and section index (`section` property), and to construct an index path from a given row index and section index (`indexPathForRow:inSection:` class method). (The first index in each index path identifies the section and the next identifies the row.)

Tasks

Configuring a Table View

- `tableView:cellForRowAtIndexPath:` (page 8) *required method*
  Asks the data source for a cell to insert in a particular location of the table view. (required)
- `numberOfSectionsInTableView:` (page 6)
  Asks the data source to return the number of sections in the table view.
- `tableView:numberOfRowsInSection:` (page 10) *required method*
  Tells the data source to return the number of rows in a given section of a table view. (required)
- `sectionIndexTitlesForTableView:` (page 7)
  Asks the data source to return the titles for the sections for a table view.
- `tableView:sectionForSectionIndexTitle:atIndex:` (page 11)
  Asks the data source to return the index of the section having the given title and section title index.
- `tableView:titleForHeaderInSection:` (page 12)
  Asks the data source for the title of the header of the specified section of the table view.
- `tableView:titleForFooterInSection:` (page 11)
  Asks the data source for the title of the footer of the specified section of the table view.

Inserting or Deleting Table Rows

- `tableView:commitEditingStyle:forRowAtIndexPath:` (page 9)
  Asks the data source to commit the insertion or deletion of a specified row in the receiver.
- `tableView:canEditRowAtIndexPath:` (page 7)
  Asks the data source to verify that the given row is editable.

Reordering Table Rows

- `tableView:canMoveRowAtIndexPath:` (page 8)
  Asks the data source whether a given row can be moved to another location in the table view.
- `tableView:moveRowAtIndexPath:toIndexPath:` (page 10)
  Tells the data source to move a row at a specific location in the table view to another location.

Instance Methods

numberOfSectionsInTableView:

Asks the data source to return the number of sections in the table view.

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

Parameters

tableView

An object representing the table view requesting this information.

Return Value

The number of sections in *tableView*. The default value is 1.

Availability

Available in iOS 2.0 and later.

See Also

- tableView:numberOfRowsInSection: (page 10)

Declared In

UITableView.h

sectionIndexTitlesForTableView:

Asks the data source to return the titles for the sections for a table view.

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

Parameters

tableView

The table-view object requesting this information.

Return Value

An array of strings that serve as the title of sections in the table view and appear in the index list on the right side of the table view. The table view must be in the plain style (UITableViewStylePlain). For example, for an alphabetized list, you could return an array containing strings "A" through "Z".

Availability

Available in iOS 2.0 and later.

See Also

- tableView:sectionForSectionIndexTitle:atIndex: (page 11)

Declared In

UITableView.h

tableView:canEditRowAtIndexPath:

Asks the data source to verify that the given row is editable.

```
- (BOOL)tableView:(UITableView *)tableView canEditRowAtIndexPath:(NSIndexPath
    *)indexPath
```

Parameters

tableView

The table-view object requesting this information.

indexPath

An index path locating a row in *tableView*.

Return Value
`YES` if the row indicated by *`indexPath`* is editable; otherwise, `NO`.

Discussion
The method permits the delegate to exclude individual rows from being treated as editable. Editable rows display the insertion or deletion control in their cells. If this method is not implemented, all rows are assumed to be editable. Rows that are not editable ignore the `editingStyle` property of a `UITableViewCell` object and do no indentation for the deletion or insertion control. Rows that are editable, but that do not want to have an insertion or remove control shown, can return `UITableViewCellEditingStyleNone` from the `tableView:editingStyleForRowAtIndexPath:` delegate method.

Availability
Available in iOS 2.0 and later.

Declared In
`UITableView.h`

tableView:canMoveRowAtIndexPath:

Asks the data source whether a given row can be moved to another location in the table view.

```
- (BOOL)tableView:(UITableView *)tableView canMoveRowAtIndexPath:(NSIndexPath
    *)indexPath
```

Parameters

`tableView`
The table-view object requesting this information.

`indexPath`
An index path locating a row in *`tableView`*.

Return Value
`YES` if the row can be moved; otherwise `NO`.

Discussion
This method allows the delegate to specify that the reordering control for a the specified row not be shown. By default, the reordering control is shown if the data source implements the `tableView:moveRowAtIndexPath:toIndexPath:` (page 10) method.

Availability
Available in iOS 2.0 and later.

Declared In
`UITableView.h`

tableView:cellForRowAtIndexPath:

Asks the data source for a cell to insert in a particular location of the table view. (required)

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

Parameters

tableView

A table-view object requesting the cell.

indexPath

An index path locating a row in *tableView*.

Return Value

An object inheriting from `UITableViewCell` that the table view can use for the specified row. An assertion is raised if you return `nil`.

Discussion

The returned `UITableViewCell` object is frequently one that the application reuses for performance reasons. You should fetch a previously created cell object that is marked for reuse by sending a `dequeueReusableCellWithIdentifier:` message to *tableView*. The identifier for a reusable cell object is assigned when the delegate initializes the cell object by calling the `initWithStyle:reuseIdentifier:` method of `UITableViewCell`. Various attributes of a table cell are set automatically based on whether the cell is a separator and on information the data source provides, such as for accessory views and editing controls.

Availability

Available in iOS 2.0 and later.

Declared In

`UITableView.h`

tableView:commitEditingStyle:forRowAtIndexPath:

Asks the data source to commit the insertion or deletion of a specified row in the receiver.

```
- (void)tableView:(UITableView *)tableView
    commitEditingStyle:(UITableViewCellEditingStyle)editingStyle
    forRowAtIndexPath:(NSIndexPath *)indexPath
```

Parameters

tableView

The table-view object requesting the insertion or deletion.

editingStyle

The cell editing style corresponding to a insertion or deletion requested for the row specified by *indexPath*. Possible editing styles are `UITableViewCellEditingStyleInsert` or `UITableViewCellEditingStyleDelete`.

indexPath

An index path locating the row in *tableView*.

Discussion

When users tap the insertion (green plus) control or Delete button associated with a `UITableViewCell` object in the table view, the table view sends this message to the data source, asking it to commit the change. (If the user taps the deletion (red minus) control, the table view then displays the Delete button to get confirmation.) The data source commits the insertion or deletion by invoking the `UITableView` methods `insertRowsAtIndexPaths:withRowAnimation:` or `deleteRowsAtIndexPaths:withRowAnimation:`, as appropriate.

To enable the swipe-to-delete feature of table views (wherein a user swipes horizontally across a row to display a Delete button), you must implement this method.

You should not call `setEditing:animated:` within an implementation of this method. If for some reason you must, invoke it after a delay by using the `performSelector:withObject:afterDelay:` method.

Availability
Available in iOS 2.0 and later.

Declared In
`UITableView.h`

tableView:moveRowAtIndexPath:toIndexPath:

Tells the data source to move a row at a specific location in the table view to another location.

```
- (void)tableView:(UITableView *)tableView moveRowAtIndexPath:(NSIndexPath
    *)fromIndexPath toIndexPath:(NSIndexPath *)toIndexPath
```

Parameters

tableView
The table-view object requesting this action.

fromIndexPath
An index path locating the row to be moved in *tableView*.

toIndexPath
An index path locating the row in *tableView* that is the destination of the move.

Discussion
The `UITableView` object sends this message to the data source when the user presses the reorder control in *fromRow*.

Availability
Available in iOS 2.0 and later.

See Also
- `tableView:commitEditingStyle:forRowAtIndexPath:` (page 9)

Declared In
`UITableView.h`

tableView:numberOfRowsInSection:

Tells the data source to return the number of rows in a given section of a table view. (required)

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

Parameters

tableView
The table-view object requesting this information.

section
An index number identifying a section in *tableView*.

Return Value
The number of rows in `section`.

Availability
Available in iOS 2.0 and later.

See Also
- `numberOfSectionsInTableView:` (page 6)

Declared In
`UITableView.h`

tableView:sectionForSectionIndexTitle:atIndex:

Asks the data source to return the index of the section having the given title and section title index.

```
- (NSInteger)tableView:(UITableView *)tableView sectionForSectionIndexTitle:(NSString
    *)title atIndex:(NSInteger)index
```

Parameters

tableView
The table-view object requesting this information.

title
The title as displayed in the section index of *tableView*.

index
An index number identifying a section title in the array returned by `sectionIndexTitlesForTableView:` (page 7).

Return Value
An index number identifying a section.

Discussion
This method is passed the index number and title of an entry in the section index list and should return the index of the referenced section. To be clear, there are two index numbers in play here: an index to an section index title in the array returned by `sectionIndexTitlesForTableView:`, and an index to a section of the table view; the former is passed in, and the latter is returned. You implement this method only for table views with a section index list—which can only be table views created in the plain style (`UITableViewStylePlain`). Note that the array of section titles returned by `sectionIndexTitlesForTableView:` can have fewer items than the actual number of sections in the table view.

Availability
Available in iOS 2.0 and later.

See Also
- `numberOfSectionsInTableView:` (page 6)

Declared In
`UITableView.h`

tableView:titleForFooterInSection:

Asks the data source for the title of the footer of the specified section of the table view.

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

Parameters

tableView

The table-view object asking for the title.

section

An index number identifying a section of *tableView*.

Return Value

A string to use as the title of the section footer. If you return `nil`, the section will have no title.

Discussion

The table view uses a fixed font style for section footer titles. If you want a different font style, return a custom view (for example, a `UILabel` object) in the delegate method `tableView:viewForFooterInSection:` instead.

Availability

Available in iOS 2.0 and later.

See Also

`- tableView:titleForHeaderInSection:` (page 12)

Declared In

`UITableView.h`

tableView:titleForHeaderInSection:

Asks the data source for the title of the header of the specified section of the table view.

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

Parameters

tableView

The table-view object asking for the title.

section

An index number identifying a section of *tableView*.

Return Value

A string to use as the title of the section header. If you return `nil`, the section will have no title.

Discussion

The table view uses a fixed font style for section header titles. If you want a different font style, return a custom view (for example, a `UILabel` object) in the delegate method `tableView:viewForHeaderInSection:` instead.

Availability

Available in iOS 2.0 and later.

See Also

`- tableView:titleForFooterInSection:` (page 11)

Declared In

`UITableView.h`

Document Revision History

This table describes the changes to *UITableViewDataSource Protocol Reference*.

Date	Notes
2010-05-24	Corrected typo.
2010-01-18	Made minor corrections.
2009-05-19	Removed discussion of tableView:didUpdateTextFieldForRowAtIndexPath: withValue:.
2009-03-08	Describes the tableView:didUpdateTextFieldForRowAtIndexPath: withValue: method, added for iOS 3.0.
2008-10-15	Added note about implementing tableView:commitEditingStyle:forRowAtIndexPath: as prerequisite for swipe-to-delete feature.
2008-09-09	Corrected description of commitEditingStyle:forRowAtIndexPath: method.
2008-06-23	New document that describes the protocol for mediating an application's data model for a UITableView object.

Document Revision History