

[UIKit](#) / [...](#) / [Collection views](#) / Selecting multiple items with a two-finger pan gesture

Sample Code

Selecting multiple items with a two-finger pan gesture

Accelerate user selection of multiple items using the multiselect gesture on table and collection views.

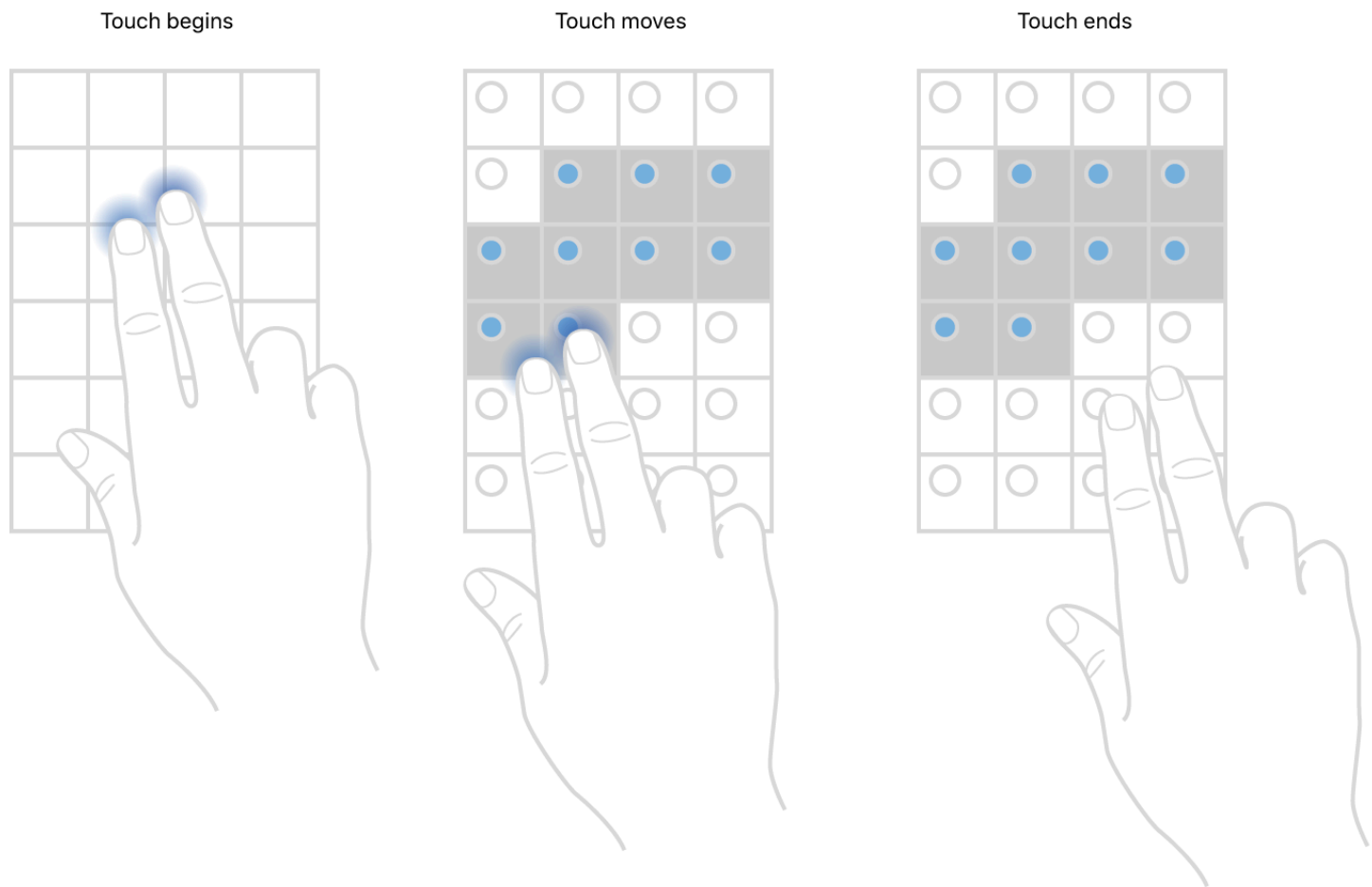
[Download](#)

iOS 15.0+ | iPadOS 15.0+ | Xcode 14.1+

Overview

In iOS 13 and later, you can provide users of your app with the ability to select multiple items using a two-finger pan gesture on table and collection views. Apps that opt-in to this feature make it possible for users to select multiple items quickly. For instance, when a table view recognizes the two-finger pan gesture, an app can automatically put the table view in edit mode, and the user doesn't need to tap an Edit or Select button.

To select multiple items, drag two fingers over the items you want to select. When the view recognizes the two-finger pan gesture, it changes to edit mode, allowing you to select more than one item.



The selected items don't have to be contiguous. Use two fingers to select a few items, scroll the view, and use your two fingers again to select more items. You can also use the same two-finger pan gesture to unselect multiple items. Just drag your two fingers over the selected items, and both the table and collection views will deselect the items.

This sample shows you how to support this feature in your app. The sample app displays a table view and a collection view. When the sample is run on iPad, the app displays the two views side-by-side using a split view. When you run the app on iPhone, the app displays a tab bar that lets you switch between the table and collection views.

Support multiple item selection in a table view

To enable the two-finger pan gesture in a table view, implement the delegate method `tableView(_:shouldBeginMultipleSelectionInteractionAt:)`, and return `true`. The table view calls this method when it detects the two-finger touch to determine whether the app supports the multiple-selection gesture.

```
override func tableView(_ tableView: UITableView, shouldBeginMultipleSelectionInteractionAt: IndexPath) -> Bool {
    return true
}
```

After returning `true`, the table view calls the `tableView(_:didBeginMultipleSelectionInteractionAt:)` delegate method. The sample app uses this opportunity to switch the table view into edit mode without requiring the user to tap the Edit button. The table view also selects the current row. The user pans their two fingers up or down on the table view to select additional rows.

```
override func tableView(_ tableView: UITableView, didBeginMultipleSelectionInteractionAt indexPath: IndexPath) {
    // Replace the Edit button with Done, and put the
    // table view into editing mode.
    self.setEditing(true, animated: true)
}
```

When the user lifts their two fingers off the device, the table view calls the `tableViewDidEndMultipleSelectionInteraction(_ :)` delegate method. This is the app's indication that the user is no longer using the two-finger pan gesture. The sample app's implementation of this method doesn't perform any action, which gives the user the opportunity to select more items using the two-finger pan gesture. The user can also select more items by moving a single finger along the edge of the table where it displays the selection checkboxes.

```
override func tableViewDidEndMultipleSelectionInteraction(_ tableView: UITableView) {
    print("\(#function)")
}
```

Support multiple item selection in a collection view

Providing the same multiselect behavior in a collection view is similar to the implementation for a table view. Start by implementing the collection view delegate method `collectionView(_:shouldBeginMultipleSelectionInteractionAt:)`, which determines whether the gesture should be available to the user. The sample app returns `true` in this method.

Next, implement the delegate method `collectionView(_:didBeginMultipleSelectionInteractionAt:)`. As with the table view delegate variant, the sample app implementation of this method puts the collection view into edit mode.

The third and last delegate method to implement is `collectionViewDidEndMultipleSelectionInteraction(_ :)`. Here the sample app doesn't perform any action so that the user can continue selecting items with either a tap or another pan gesture using two fingers.

```
func collectionView(_ collectionView: UICollectionView, shouldBeginMultipleSelectionInteractionAt indexPath: IndexPath) {
    // Returning `true` automatically sets `collectionView.isEditing`
    // to `true`. The app sets it to `false` after the user taps the Done button.
}
```

```

        return true
    }

    func collectionView(_ collectionView: UICollectionView, didBeginMultipleSelectionInteraction: UICollectionViewMultipleSelectionInteraction) {
        // Replace the Select button with Done, and put the
        // collection view into editing mode.
        setEditing(true, animated: true)
    }

    func collectionViewDidEndMultipleSelectionInteraction(_ collectionView: UICollectionView) {
        print("\(#function)")
    }
}

```

The user can also pan a single finger along the constrained axis to select more items. For instance, if the collection view scrolls vertically, the user can pan one finger horizontally to select more items.

See Also

Selection management

{} Changing the appearance of selected and highlighted cells

Provide visual feedback to the user about the state of a cell and the transition between states.