Class

UICollectionViewTransitionLayout

A special type of layout object that lets you implement behaviors when changing from one layout to another in your collection view.

Declaration

```
class UICollectionViewTransitionLayout : UICollectionViewLayout
```

Overview

You can use `UICollectionViewTransitionLayout` as-is or subclass it to provide specialized behavior for your app. A common use for transition layouts is to create interactive transitions, such as those that are driven by gesture recognizers or touch events.

During a layout change, the collection view installs this layout object temporarily to manage the changeover. This layout object determines the layout of each item by interpolating between the layout values in the current and new layout objects. The interpolation is driven by the value in the `transitionProgress` property, which you update periodically from your code to drive the transition. For example, if you use this class in conjunction with a gesture recognizer, the handler for your gesture recognizer would update that property and invalidate the layout.

If you want to provide more than just a linear transition from the old to new layout over time, you need to subclass and provide the layout attributes for items yourself. Subclassing requires you to override all of the same methods you would override when subclassing `UICollectionViewLayout`. The difference is that your custom methods can work with your gesture recognizers or touch event code to change the layout based on input from the user. For example, you could use a custom layout object in conjunction with a gesture recognizer to make items track the location of the user's finger on the screen. You also need to implement the `collectionView(_:transitionLayoutForOldLayout:newLayout:)` method of your collection view delegate and return your custom layout object when asked for it.

Topics

Initializing the Transition Layout Object

`init(currentLayout: UICollectionViewLayout, nextLayout: UICollectionViewLayout)`

Initializes and returns a transition layout object.

`init?(coder: NSCoder)`

Updating the Transition Information

`var transitionProgress: CGFloat`

The completion percentage of the transition.

`func updateValue(CGFloat, forAnimatedKey: String)`

Sets the value for an animatable key.

`func value(forAnimatedKey: String) -> CGFloat`

Returns the most recently set value for the specified key.

Accessing the Layout Objects

`var currentLayout: UICollectionViewLayout`

The collection view’s current layout object.

`var nextLayout: UICollectionViewLayout`

The collection view’s new layout object.

Relationships

Inherits From

`UICollectionViewLayout`

Conforms To

`CVarArg`, `Equatable`, `Hashable`

See Also

Layouts

`class UICollectionViewLayout`

An abstract base class for generating layout information for a collection view.

`class UICollectionViewFlowLayout`

A concrete layout object that organizes items into a grid with optional header and footer views for each section.

`class UICollectionViewLayoutAttributes`

A layout object that manages the layout-related attributes for a given item in a collection view.

{}Customizing Collection View Layouts

Customize a view layout by changing the size of cells in the flow or implementing a mosaic style.