Class

NSTextField

Text the user can select or edit to send an action message to a target when the user presses the Return key.

macOS

```
@MainActor
class NSTextField
```

Mentioned in

- Adding Writing Tools support to a custom AppKit view
- Adopting the system text cursor in custom text views
- Customizing Writing Tools behavior for AppKit views

Overview

The `NSTextField` class uses the `NSTextFieldCell` class to implement its user interface. Text fields display text either as a static label or as an editable input field. The content of a text field is either plain text or a rich-text attributed string. Text fields also support line wrapping to display multiline text, and a variety of truncation styles if the content doesn't fit the available space.

The parent class, `NSControl`, provides the methods for setting the values of the text field, such as `stringValue` and `doubleValue`. There are corresponding methods to retrieve values.

In macOS 12 and later, if you explicitly call the `layoutManager` property on your text field, the framework will revert to a compatibility mode that uses `NSLayoutManager`. The text view also switches to this compatibility mode when it encounters text content that's not yet supported.

Topics

Creating Text Fields

`convenience init(labelWithAttributedString: NSAttributedString)`

Creates a text field for use as a static label that displays styled text, doesn’t wrap, and doesn’t have selectable text.

`convenience init(labelWithString: String)`

Initializes a text field for use as a static label that uses the system default font, doesn’t wrap, and doesn’t have selectable text.

`convenience init(string: String)`

Initializes a single-line editable text field for user input using the system default font and standard visual appearance.

`convenience init(wrappingLabelWithString: String)`

Initializes a text field for use as a multiline static label with selectable text that uses the system default font.

Controlling Selection and Editing

`var isSelectable: Bool`

A Boolean value that determines whether the user can select the content of the text field.

`var isEditable: Bool`

A Boolean value that controls whether the user can edit the value in the text field.

Controlling Rich Text Behavior

`var allowsEditingTextAttributes: Bool`

A Boolean value that controls whether the user can change font attributes of the text field’s string.

`var importsGraphics: Bool`

A Boolean value that controls whether the user can drag image files into the text field.

Setting Placeholder Text

`var placeholderString: String?`

The string the text field displays when empty to help the user understand the text field's purpose.

`var placeholderAttributedString: NSAttributedString?`

The attributed string the text field displays when empty to help the user understand the text field's purpose.

Configuring Line Wrapping

`var lineBreakStrategy: NSParagraphStyle.LineBreakStrategy`

The strategy that the system uses to break lines when laying out multiple lines of text.

`var allowsDefaultTighteningForTruncation: Bool`

A Boolean value that controls whether single-line text fields tighten intercharacter spacing before truncating the text.

`var maximumNumberOfLines: Int`

The maximum number of lines a wrapping text field displays before clipping or truncating the text.

Sizing with Auto Layout

`var preferredMaxLayoutWidth: CGFloat`

The maximum width of the text field's intrinsic content size.

Setting the Text Color

`var textColor: NSColor?`

The color of the text field's content.

Controlling the Background

`var backgroundColor: NSColor?`

The color of the background the text field's cell draws behind the text.

`var drawsBackground: Bool`

A Boolean value that controls whether the text field's cell draws a background color behind the text.

`var isBezeled: Bool`

A Boolean value that controls whether the text field draws a bezeled background around its contents.

`var bezelStyle: NSTextField.BezelStyle`

The text field’s bezel style, square or rounded.

`enum BezelStyle`

The style of bezel the text field displays.

Setting a Border

`var isBordered: Bool`

A Boolean value that controls whether the text field draws a solid black border around its contents.

Selecting the Text

`func selectText(Any?)`

Ends editing in the text field and, if it’s selectable, selects the entire text content.

Working with the Responder Chain

`var acceptsFirstResponder: Bool`

A Boolean value that indicates whether the text field is editable and accepts first responder status.

Using Keyboard Interface Control

`var allowsCharacterPickerTouchBarItem: Bool`

A Boolean value that controls whether the Touch Bar displays the character picker item for rich text fields.

Supporting Text Completion and Suggestions

`var isAutomaticTextCompletionEnabled: Bool`

A Boolean value that indicates whether the text field automatically completes text as the user types.

`protocol NSTextSuggestionsDelegate`

A protocol for suggestion delegates of text fields to conform to in order to provide text suggestions in response to the user typing.

`struct NSSuggestionItem`

The items that appear in suggestion menus.

`struct NSSuggestionItemResponse`

Describes the result of a batch of suggestion items from a search

`struct NSSuggestionItemSection`

Describes a section of suggestions items in a suggestions menu

Setting the Delegate

`var delegate: (any NSTextFieldDelegate)?`

The text field’s delegate.

Implementing Delegate Methods

`func textShouldBeginEditing(NSText) -> Bool`

Requests permission to begin editing a text object.

`func textDidBeginEditing(Notification)`

Posts a notification to the default notification center that the text is about to go into edit mode.

`func textDidChange(Notification)`

Posts a notification when the text changes, and forwards the message to the text field’s cell if it responds.

`func textShouldEndEditing(NSText) -> Bool`

Performs validation on the text field’s new value.

`func textDidEndEditing(Notification)`

Posts a notification when the text is no longer in edit mode.

Instance Properties

`var allowsCharacterPickerTouchBarItem: Bool`

A Boolean value that controls whether the Touch Bar displays the character picker item for rich text fields.

`var allowsWritingTools: Bool`

`var allowsWritingToolsAffordance: Bool`

`var placeholderAttributedStrings: [NSAttributedString]`

`var placeholderStrings: [String]`

`var resolvesNaturalAlignmentWithBaseWritingDirection: Bool`

Specifies the behavior for resolving `NSTextAlignment.natural` to the visual alignment.

`var suggestionsDelegate: (any NSTextSuggestionsDelegate)?`

The delegate that provides text suggestions for the receiving text field and responds to the user highlighting and selecting items.

Relationships

Inherits From

`NSControl`

Inherited By

`NSComboBox`
`NSSearchField`
`NSSecureTextField`
`NSTokenField`

Conforms To

`CVarArg`
`CustomDebugStringConvertible`
`CustomStringConvertible`
`Equatable`
`Hashable`
`NSAccessibilityElementProtocol`
`NSAccessibilityNavigableStaticText`
`NSAccessibilityProtocol`

```
NSAccessibilityStaticText
NSAnimatablePropertyContainer
NSAppearanceCustomization
NSCoding
NSDraggingDestination
NSObjectProtocol
NSStandardKeyBindingResponding
NSTextContent
NSTouchBarProvider
NSUserActivityRestoring
NSUserInterfaceItemIdentification
NSUserInterfaceValidations
Sendable
SendableMetatype
```

See Also

Text views

`protocol NSTextFieldDelegate`

A protocol that a text field delegate can use to control its field editor action menu.

`class NSTextView`

A view that draws text and handles user interactions with that text.

`protocol NSTextViewDelegate`

A set of optional methods that text view delegates can use to manage selection, set text attributes, work with the spell checker, and more.

`protocol NSTextDelegate`

A set of optional methods implemented by the delegate of an `NSText` object to edit text and change text formats.

`class NSText`

The most general programmatic interface for objects that manage text.