

[AppKit](#) / [Writing Tools](#) / Customizing Writing Tools behavior for AppKit views

Article

Customizing Writing Tools behavior for AppKit views

Modify the behavior of Writing Tools in standard macOS text views, and adjust your app's behavior while the feature is active.

Overview

The [NSTextView](#) and [NSTextField](#) classes have built-in support for Writing Tools, but you can customize how the feature works for your interface. You might choose to change how someone experiences Writing Tools in your app, or disable the feature for specific types of content. For example, you might disable Writing Tools in a view that you use to display code listings. You can also customize the type of content Writing Tools generates for your text view.

Note

If you create a custom text view, instead of using the standard system views, use the available APIs to add Writing Tools support to your view. For more information, see [Adding Writing Tools support to a custom AppKit view](#).

Specifying the Writing Tools UI experience you want

Writing Tools supports both limited and complete versions of the Writing Tools experience. The limited experience keeps changes in the Writing Tools interface until the person approves them. The complete experience displays the changes inline with the rest of your view's content. Both experiences give people the ability to accept or reject changes, but the complete experience offers a more interactive way to view those changes.

To specify the experience you want, change the value in your view's [writingToolsBehavior](#) property. The system applies the best available experience to views by default, but you can specify

the experience you prefer in your views. You can also disable the feature if you don't want Writing Tools to be active in your view. To disable the feature, set the value of this property to `NSWritingToolsBehavior.none`.

Modify your app's behavior when Writing Tools is active

It takes time for the large language models (LLMs) of Writing Tools to suggest changes. It also takes time for someone to review the changes and approve them. While those things happen, Writing Tools might make temporary changes to your view's content. During that time, disable any app-specific features that might interfere with the Writing Tools operation. For example, you might choose to disable cloud-based updates of your text while Writing Tools is active.

In a `NSTextView` you can determine when Writing Tools is active using your delegate object. The system calls the delegate's `textViewWritingToolsWillBegin(_ :)` method before Writing Tools makes any modifications, and it calls the `textViewWritingToolsDidEnd(_ :)` method after incorporating any final changes. Use those methods to turn off any features that might interfere with Writing Tools while it's active.

Specify ranges of text for Writing Tools to ignore

If you don't want Writing Tools to modify portions of your `NSTextView` object, implement the `textView(_ :writingToolsIgnoredRangesInEnclosingRange:)` method in your delegate. Writing Tools calls that method for each operation, providing you with the range of text it's considering. Use your implementation of the method to specify any subranges of text you want Writing Tools to ignore. You might use this method to prevent Writing Tools from changing code listings, proper names, or direct quotes embedded in the text.

See Also

Configuration

`enum` `NSWritingToolsBehavior`

Constants that specify the Writing Tools experience for the underlying view.

`struct` `NSWritingToolsResultOptions`

Constants to specify what type of content to allow in Writing Tools suggestions or rewrites.