Article

Previewing widgets and Live Activities in Xcode

Use Xcode previews to iteratively develop, fine-tune, and troubleshoot widgets and Live Activities.

Overview

With Xcode previews, you can make changes to your app's code and see the results quickly in the preview canvas. Using the `#Preview` macros for widgets, you can preview widgets and Live Activities. The macros allow you to provide timeline entries and content states and click through timeline updates and content changes. As a result, Xcode previews help you quickly and iteratively develop, review, and fine-tune functionality and animations.

For general information about Xcode previews, refer to Previewing your app's interface in Xcode.

> **Note**
>
> Xcode previews support for iOS and watchOS widgets, and Live Activities. Debug your macOS widgets as described in Debugging widgets.

Provide timeline entries to widget previews

With Xcode previews, you can preview each widget for context and appearance to adjust your widget layouts as needed and iteratively refine their appearance. For example, create a preview for you `WidgetFamily.systemSmall` widget and view it on the preview canvas in light mode, dark mode, in StandBy, in NightMode, and on the iPad Lock Screen.

Using one of the `#Preview` macros, provide Xcode with the widget and its widget families:

- `Preview(_:as:widget:timelineProvider:)`

- `Preview(_:as:using:widget:timelineProvider:)`
- `Preview(_:as:using:widget:timelineProvider:)`
- `Preview(_:as:widget:timeline:)`

Note that you can create a timeline provider to supply the preview with timeline entries, or create a new timeline with entries specific to the widget. For either option, Xcode creates a preview for each timeline entry and shows transitions for each data update when you navigate from entry to entry in the preview canvas. When you provide a `timelineProvider`, you can easily look for issues as your widget updates and identify bugs and things to polish as data changes.

In some cases, you might need to fix an animation that occurs later in the timeline. Creating a specific timeline is especially useful because it can help you fix the issue faster by focusing on a specific transition between two timeline entries. For example, you might supply the preview with a timeline provider that has 10 entries and identify an optimization for the transition from entry 9 to 10. To focus on optimizing this specific transition without going through entries 1 to 8, you create a preview with a `timeline` specific to this case that only contains entries 9 and 10.

The following example shows a preview for the medium system family widget of the Emoji Rangers: Supporting Live Activities, interactivity, and animations sample code project that uses a timeline with two entries.

```
#Preview(as: .systemMedium, widget: {
    EmojiRangerWidget()
}, timeline: {
    let date = Date()
    SimpleEntry(date: date, relevance: nil, hero: .spouty)
    SimpleEntry(date: date.addingTimeInterval(60), relevance: nil, hero: .spook)
})
```

Tip

You can click the pin button in the upper left of the canvas to keep the preview active even when navigating to another file. Then, press the preview's play button in the timeline and the loop button. This process allows you to make code changes in one half of the editor — for example, fixing a transition in one file — while you view the updates live in the preview canvas on the second half of the editor without navigating back and forth from the preview canvas.

Create previews for your Live Activities

Previewing Live Activities with Xcode works similar to previewing widgets. The only notable difference is that ActivityKit doesn't use timelines to update your Live Activities. As a result, the `Preview( :as:using:widget:contentStates:)` macro for Live Activities let you provide `contentStates` to preview content updates for your Live Activity previews.

See Also

Previews and debugging

`struct WidgetPreviewContext`

A specification for the context of a widget preview.

Preview macros

Use Swift macros to create widget previews in Xcode.