Article

Making network requests in a widget extension

Update your widget with new information you fetch with a network request.

Overview

Widgets use a timeline mechanism to update their content. In addition to providing a timeline with new local data, you can initiate a network request that provides updated information for your widget from your server. Network requests are possible while the widget extension is active, such as when it provides a snapshot, or a new timeline. For example, the Emoji Rangers widget of the Emoji Rangers: Supporting Live Activities, interactivity, and animations sample code project loads updated leaderboard data in its `getTimeline(in:completion:)` implementation.

In general, making a network request from a widget extension is similar to making a network request in your app: you can initiate inline or background requests. However, widgets can only access limited resources and a request may not have enough time to complete before the system halts the widget extension. To update a widget with data from your server, use background requests similar to the process described in Downloading files in the background.

Load data with a background network request

When a widget initiates a background network request, the system delivers events related to the request directly to the widget extension instead of the containing app. To process them, do the following:

1. Add the `onBackgroundURLSessionEvents(matching:_:)` modifier to your `WidgetConfiguration` implementation.
2. Maintain a reference to the `URLSession` object of your background network request.
3. Keep a reference to the `matching` parameter to identify the session.

4. Store a reference to the `onBackgroundURLSessionEvents(matching:_:)` modifier's completion handler. You'll invoke it after the system delivers all events for the network request.

If the system terminates your widget extension before all events complete, use the stored `matching` identifier to check if a corresponding `URLSession` object exists. If no session object exists, create a new session using the identifier.

> **Tip**
>
> Consider initializing URLSession objects lazily and caching them in a central location so that your code works regardless of whether your extension remains active, is suspended, or is terminated.

Update your widget after background network requests complete

After invoking `onBackgroundURLSessionEvents(matching:_:)`, the system calls the `urlSession(_:downloadTask:didFinishDownloadingTo:)` method of the `URLSessionDelegate` you supplied to the `URLSession`. When the system has delivered all events, it calls the delegate's `urlSessionDidFinishEvents(forBackgroundURLSession:)` method.

To refresh your widget's timeline after the network request completes, call the needed `WidgetCenter` methods from your implementation of `SessionDidFinishEvents(forBackgroundURLSession:)`. Once you finish handling the events, call the completion handler of on `BackgroundURLSessionEvents(matching:_:)` that you previously stored.

See Also

Capabilities

- Accessing location information in widgets

  Incorporate location information into your widget presentation to make it more relevant and contextual.