

[StoreKit](#) / Message

## Structure

# Message

An instance for receiving and displaying App Store messages in your app.

iOS 16.0+ | iPadOS 16.0+ | Mac Catalyst 16.0+ | visionOS 1.0+

```
struct Message
```

## Mentioned in

- 📄 Merchandising win-back offers in your app
- 📄 Testing win-back offers in the sandbox environment
- 📄 Choosing a StoreKit API for In-App Purchases
- 📄 Testing failing subscription renewals and In-App Purchases

## Overview

A StoreKit message represents a sheet that appears over your app to display important information from the App Store. Messages have a reason, which the `reason` value indicates. StoreKit retrieves any messages from the App Store each time your app launches, and presents them by default.

### Note

StoreKit displays messages from the App Store regardless of the SDK version you use to build your app. Apps built for iOS 16 and later can implement a message listener and delay or suppress messages.

You can optionally use the Message API to control message presentation by delaying or suppressing messages. Your app can listen for messages with the `messages` asynchronous sequence, and can display them at a particular time by calling `display(in:)`, or `DisplayMessageAction` for SwiftUI views. For example, you may choose to delay messages in views where an interrupting sheet might confuse someone, such as in the middle of an onboarding flow, or if your app is providing real-time instructions.

StoreKit presents message sheets only if a message is still relevant. For example, if a person resolves the issue outside your app before it calls `display(in:)`, StoreKit doesn't present the message. StoreKit ensures that it presents each unique message once, even if the app asks to display messages multiple times.

## Listen for and display messages

If you want to defer or suppress App Store messages, set up the message listener in your app when your app launches.

To control when a message may display, call `display(in:)` or `DisplayMessageAction` when your app's ready to have StoreKit present the message. If your app doesn't call either of these APIs after it listens for messages, it suppresses the messages. The following example is for apps that use `UIKit`:

```
// Listen for App Store messages.
for await message in StoreKit.Message.messages {
    // Call display on the message when the app is ready.
}

// Indicate the app is ready to display the message.
guard let windowScene = self.view.window?.windowScene else {
    fatalError("Could not get window scene.")
}
try? message.display(in: windowScene)
```

For a code example that uses SwiftUI, see [DisplayMessageAction](#).

---

## Topics

### Getting messages and message reasons

```
static var messages: Message.Messages
```

The asynchronous sequence that sends a message when the App Store creates it.

```
let reason: Message.Reason
```

The reason that the App Store sends the message.

```
struct Messages
```

An asynchronous sequence of messages from the App Store.

```
struct Reason
```

Reasons for the App Store messages.

## Displaying messages

```
func display(in: UIWindowScene) throws
```

Requests the system to display the App Store message in the window scene.

---

# Relationships

## Conforms To

Copyable

Equatable

Hashable

Sendable

SendableMetatype

---

## See Also

## Messages

```
struct Reason
```

Reasons for the App Store messages.

```
struct DisplayMessageAction
```

An instance that asks StoreKit to display an App Store message, if appropriate.