Article

Testing the launch experience of your App Clip

Debug App Clip invocations, test the launch experience, and verify the configuration of your released App Clip.

Overview

A fast launch experience is crucial to the experience your App Clip provides. As a result, you'll spend a significant amount of time choosing invocation URLs, configuring App Clip experiences in App Store Connect, and writing code to handle the invocation URL.

To ensure your App Clip offers a smooth launch experience and that you properly configured your App Clip experiences, use the following tests and verifications:

- During development, use the `_XCAppClipURL` environment variable to configure an invocation URL for debugging.
- Verify the design of the App Clip card and the launch experience of your App Clip from invocations by creating a local experience on your test device.
- When you're ready to share a beta version with others, create an App Clip experience for testers in TestFlight.
- When you've published your App Clip on the App Store, verify your App Clip configuration by using the App Clip diagnostics on iPhone.

Debug your App Clip

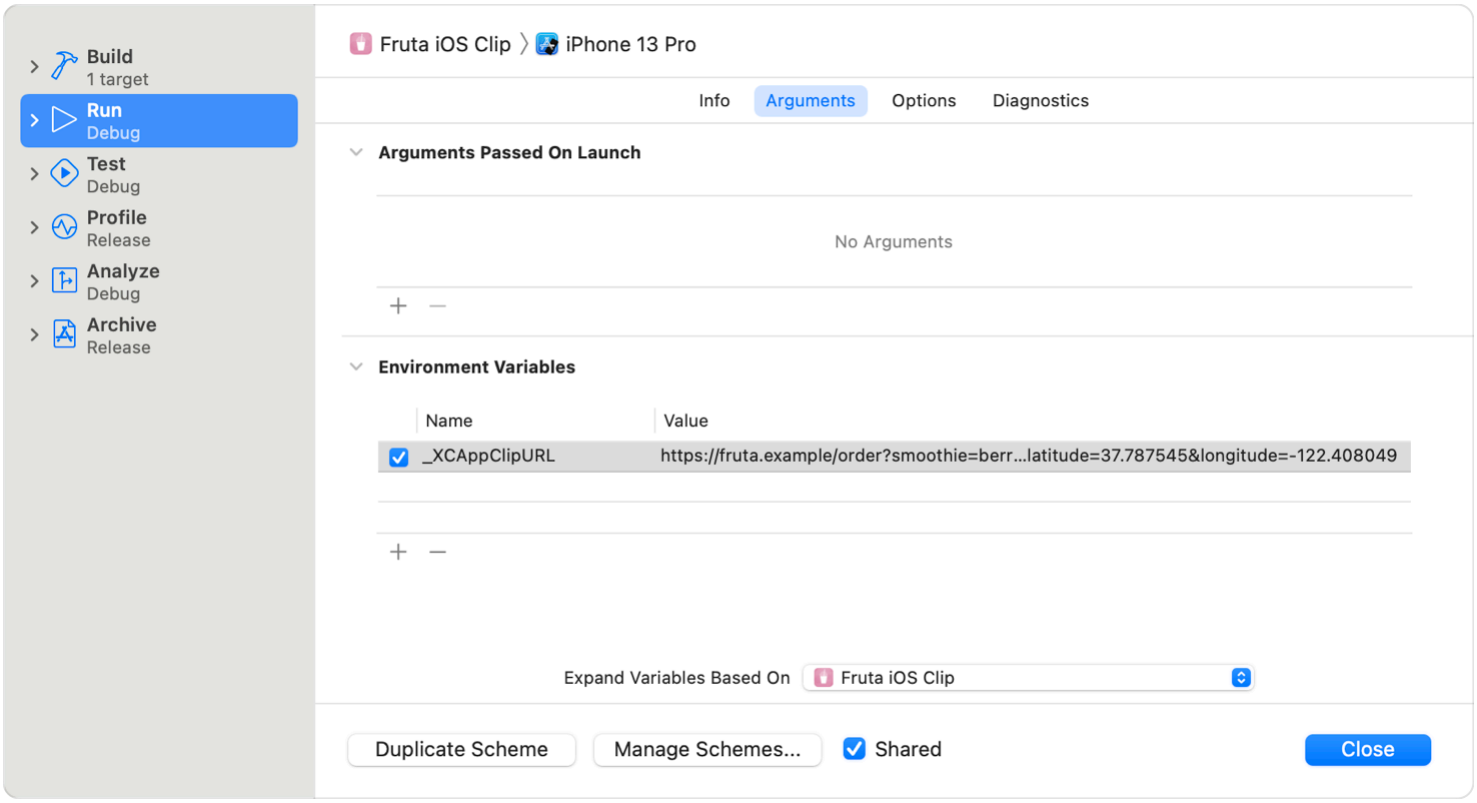
The invocation URL plays a key role to offering a fast and instant launch experience for your App Clip because you can use it to update the user interface of your App Clip to match the user's context. Testing this code is especially important. In Xcode, provide an invocation URL for

debugging. Then, make sure your code handles your invocation URLs — including unexpected invocation URLs — and updates the user interface of your App Clip to match the user's context.

To configure an invocation URL for debugging:

1. In Xcode, choose Product > Scheme > Edit Scheme and select the scheme for your App Clip.
2. Select the Run action.
3. In the Arguments tab, check whether the `_XCAppClipURL` environment variable is present. When you add an App Clip target to your project, Xcode adds it for you. If it's missing, add the environment variable.
4. Set the environment variable's value to the invocation URL you want to test.
5. Enable the variable by selecting the checkbox next to it.
6. Build and run the App Clip to access the test URL you configured from an `NSUserActivity` object. For more information on accessing the invocation URL, refer to Responding to invocations.

The following screenshot shows the sheet to configure an App Clip target's Run action with a value for the `_XCAppClipURL` environment variable:



When you debug your App Clip with Xcode, the App Clip launches right away with the value you set for the `_XCAppClipURL` variable. Note that the App Clip card doesn't appear. To see the App Clip card on invocation when testing and test your entire launch experience, register a local experience on your test device.

Test App Clip invocations with a local experience

Leveraging the `_XCAppClipURL` environment variable is helpful when you debug the code that handles the invocation URL. However, you need to ensure your App Clip provides a fast and reliable launch experience from various invocations. In addition, exploring imagery, text, and a call-to-action verb for the App Clip card is especially important because the App Clip card is the user's first interaction when they launch your App Clip.

> **Note**
>
> Local experiences are a great way to test the launch experience of your App Clip during development. However, you can use local experiences to test the scenario in which a person has installed the full app and an invocation launches the app and not the App Clip. To verify that the invocation URL can launch your full app, use universal links. For more information, refer to Test the launch experience of the full app during development below.

To test invocations and explore the design of your App Clip card during development, configure a local experience on your test device. With a local experience, you can launch your App Clip by:

- Scanning an App Clip Code, QR code, or NFC tag you've created to launch the local experience. For information on creating App Clip Codes for testing, refer to Creating App Clip Codes with the App Clip Code Generator. To create a QR code or write an NFC tag, use your favorite tool.
- Tapping a Smart Banner you added to your website or an App Clip card that appears in Safari or an `SFSafariViewController`.
- Sharing a link to the website that displays a Smart App Banner. Make sure to add the sender of the message as a contact on the test device.

Note that you don't need to configure the `Associated Domains Entitlement` or associate the App Clip with your website to launch a local experience from App Clip Codes, QR codes, or NFC tags. However, testing invocations from a Smart App Banner or the App Clip card in Safari comes with prerequisites; for example, you need to associate your website with your App Clip, submit a version of your app with an App Clip for review to App Store Connect, and release it in the App Store after approval. To learn more about when the Smart App Banner appears and for information on supporting invocations from your website or the Messages app, refer to Supporting invocations from your website and the Messages app. To verify your configured App Clip experiences for a released app and App Clip, refer to Verify the configuration of your released App Clip below.

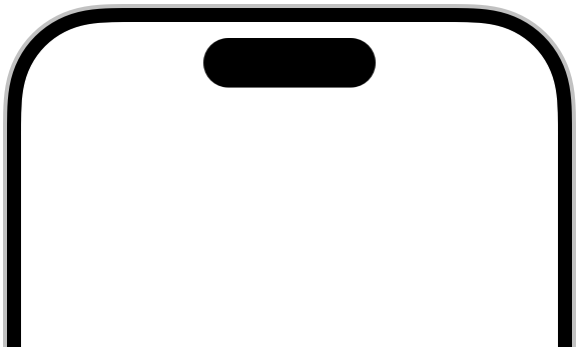
> **Tip**
>
> Add the Code Scanner and the NFC Tag Reader to Control Center by opening the Settings app and selecting Control Center.

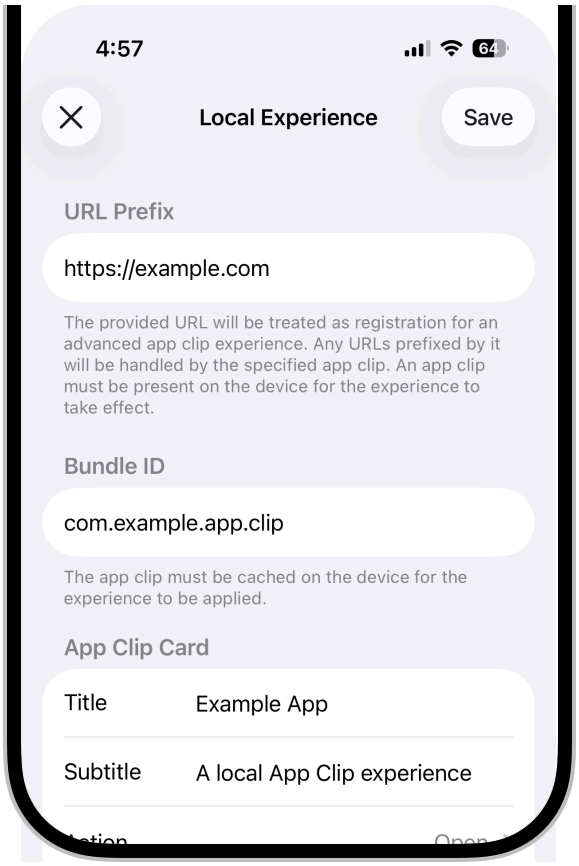
To test your launch experience with a local experience:

1. Build and run your App Clip on your test device to make sure it's cached on the test device. For example, with the Fruta sample code project, follow the instructions to configure it, then run the `Fruta iOS Clip` scheme. To download the sample code project, visit Fruta: Building a feature-rich app with SwiftUI.
2. On your test device, open the Settings app, navigate to the iOS developer settings by choosing Developer > Local Experiences, and select Register Local Experience.
3. Enter the invocation URL you want to test. In some cases, it may be a simple URL prefix like `https://fruta.example.com`. It can also be a longer invocation URL with path and query parameters.
4. Enter the bundle ID of your App Clip.
5. Provide a title and a subtitle. For the Fruta sample code project, you could enter `Order a smoothie.` as the title and `It's yummy!` as the subtitle.
6. Select a call-to-action; for example `Open`.
7. Choose a photo you want to use as the App Clip card's header image.

> **Note**
>
> In addition to building and running your App Clip on your test device to cache it on the device, you can also manually add the `.ipa` file of your App Clip to the device. First, archive the full app that corresponds with your App Clip. Then, in Xcode, export the App Clip for Ad Hoc or Development distribution. Finally, drag the exported `.ipa` file to a connected device that appears in the Devices and Simulators window.

The following screenshot shows the interface you use to configure a local experience on iPhone:





When you configure a local experience on a device, the local experience takes precedence over App Clip experiences you configure in App Store Connect. However, local experiences only launch an App Clip that's signed for Development, Ad Hoc, or TestFlight distribution. They don't launch an App Clip or full app that's published on the App Store. Remember to remove the local experience before testing App Clip experiences you configure in App Store Connect.

Test the launch experience of the full app during development

Local App Clip invocations help you test the launch experience of your App Clip while you develop it in Xcode. However, local invocations don't allow you to test the launch experience of the full app from an App Clip invocation. The best way to verify that your full app can handle App Clip invocations as you would expect is to use universal links and the invocation URL of your App Clip as the link's URL. Both App Clips and universal links make the launch URL available as part of the user activity object. If a Universal Link opens your app to the App Clip experience you expect, you can be sure that the launch experience from the App Clip card to your full app works correctly.

For more information about debugging and testing universal links, refer to TN3155: Debugging universal links. For general information about universal links, refer to Supporting universal links in your app.

Create App Clip experiences for testers in TestFlight

A reliable user experience is crucial to App Clips and requires spending time to test all user flows and supported invocations. To help make sure your App Clip works as expected, you can make your App Clip available to testers in TestFlight. First, upload an app with an App Clip to App Store Connect. Then, in App Store Connect, navigate to the uploaded build. In the App Clip section, you can configure up to three different App Clip experiences for testing. Note that the App Clip card doesn't appear for App Clip experiences you create in TestFlight.

> **Important**
>
> Unlike local experiences, creating an App Clip experience for testers in TestFlight requires you to associate your App Clip with your website. For more information on configuring associated domains, refer to Associating your App Clip with your website.

Users don't install App Clips, and App Clips don't appear on the Home Screen. Similarly, testers don't install the beta version of your App Clip, and it also doesn't appear on the Home Screen either. Instead, testers launch the App Clip experiences you configure for testing through the TestFlight app on their device.

For more information on configuring experiences for testing in App Store Connect, refer to Testing an App Clip Experience.

Testers can also configure a local experience to launch the App Clip you distribute with TestFlight. However, you must still associate your App Clip with your website so testers can launch it from the TestFlight app. In addition, testers must launch the App Clip from an App Clip experience you configure for testing in App Store Connect at least once to ensure that the App Clip is cached on the device.

Remove cached advanced App Clip experiences

When you create a new advanced App Clip experience in App Store Connect, the new experience may not work right away because iOS caches App Clip experiences. To make sure an invocation launches a new App Clip experience, open the Settings app, navigate to the Developer section, and choose Clear Experience Cache.

Verify the configuration of your released App Clip

When you've uploaded your app version with an App Clip, it has passed App Store review, and you've released the version in the App Store, verify the configuration of your App Clip experiences to be sure that your website displays the Smart App Banner and that the App Clip appears on users' devices:

1. Configure a default and optional advanced App Clip experiences in App Store Connect and release the version of your app that contains the App Clip on the App Store.
2. On iPhone, open the Settings app, and navigate to the iOS developer settings by choosing Developer.
3. Choose Diagnostics in the App Clips Testing section.
4. Enter the URL of the website you associated with your App Clip and that you use for your App Clip experiences — for example, `https://fruta.example.com`.

When you enter a URL, the system verifies your App Clip configuration and indicates whether you:

- Registered an advanced App Clip experience
- Published your App Clip on the App Store
- Associated your App Clip with the entered URL
- Picked an invocation URL that fits into an App Clip Code
- Added a Smart App Banner to your website

If the system found issues with your configuration, the App Clip diagnostics functionality on iPhone indicates errors and offers links to App Clip documentation to help resolve issues. Note that advanced App Clip experiences and displaying a Smart App Banner on your website are optional but recommended steps that help users discover your App Clip.