Article

Deleting Donated Shortcuts

Remove your donations from Siri.

Overview

If your app donated a shortcut containing information that the user has since deleted, your app should delete the donation. For example, let's say that a user sends a message to a contact using your messaging app. Your app donates the *send message* shortcut that includes information about the contact. Later, the user deletes that contact from the app. The app should delete the donations it made that included the contact information.

When a user deletes data from an app, they expect the data to be gone for good. This means removing the data from the app as well as from the system—if the system has knowledge of that data. It's important for your app to honor this unspoken agreement with the user: to maintain privacy and trust at all times.

There are other times your app should delete donations; for instance, when your app no longer supports the action. In general, whenever a donation is no longer appropriate for the user, your app should delete the donation.

Delete Interaction Donations

To delete donations made with interaction objects, use one of the `delete` class methods on `INInteraction` (see Deleting Interactions from the System). To delete specific donations, use `delete(with:completion:)`, passing in the list of `identifier` strings for each interaction to delete. Or use `delete(with:completion:)`, passing in the `groupIdentifier` string used to group of two or more interactions.

If you need to delete all interactions that your app donated—for example, when the user signs out of your app—use `deleteAll(completion:)`.

Delete User Activity Donations

The method you use to delete a donation made with `NSUserActivity` depends on how you set up the user activity.

Persisted Activities

If user activity's `isEligibleForPrediction` property is set to `true` and the `persistentIdentifier` property is set to a unique string identifying the activity, then:

- Use `deleteSavedUserActivities(withPersistentIdentifiers:completionHandler:)`, passing in the list of persistent identifiers, to delete individual activities. This method also deletes user activities stored by Core Spotlight that have a matching persistent identifier.
- Use `deleteAllSavedUserActivities(completionHandler:)` to delete all activities saved by your app; for example, after the user signs out of your app. This method also deletes all user activities created by your app and stored by Core Spotlight.

Spotlight Indexed Activities

If the user activity is indexed in Spotlight and the `relatedUniqueIdentifier` property is set to a unique string identifying the activity, then:

- Use `deleteSearchableItems(withIdentifiers:completionHandler:)` to delete specific Spotlight items.
- Use `deleteSearchableItems(withDomainIdentifiers:completionHandler:)` to delete Spotlight items grouped by domain identifiers.
- Use `deleteAllSearchableItems(completionHandler:)` to delete all searchable items that your app contributed to Spotlight from the index.

When you delete a Spotlight item, the system deletes the related user activity and its donations.

For more information about Spotlight indexing, see Core Spotlight.

See Also

Articles

Adding User Interactivity with Siri Shortcuts and the Shortcuts App

Add custom intents and parameters to help users interact more quickly and effectively with Siri and the Shortcuts app.

- Defining Relevant Shortcuts for the Siri Watch Face

  Inform Siri when your app's shortcuts may be useful to the user.

- Dispatching intents to handlers

  Provide SiriKit with an intent handler capable of handling a specific intent.

- Improving Siri Media Interactions and App Selection

  Fine-tune voice controls and improve Siri Suggestions by sharing app capabilities, customized names, and listening habits with the system.

- Improving interactions between Siri and your messaging app

  Donate app-specific content, use Siri's contact suggestions, and adopt the latest platform features to create a more consistent messaging experience.

- Registering Custom Vocabulary with SiriKit

  Register your app's custom terminology, and provide sample phrases for how to use your app with Siri.

- Confirming the Details of an Intent

  Perform final validation of the intent parameters and verify that your services are ready to fulfill the intent.

- Handling an Intent

  Fulfill the intent and provide feedback to SiriKit about what you did.

- Resolving the Parameters of an Intent

  Validate the parameters of an intent and make sure that you have the information you need to continue.

- Generating a List of Ride Options

  Generate ride options for Maps to display to the user.

- Handling the Ride-Booking Intents

  Support the different intent-handling sequences for booking rides with Shortcuts or Maps.

- Donating Reservations

  Inform Siri of reservations made from your app.

- Specifying Synonyms for Your App Name

  Provide alternative names for your app that are more familiar or easier for users to speak.

- Intent Phrases

The keys that you include in your global vocabulary file to show how users engage your app from Siri.

Localizing Your Vocabulary for Chinese Dialects

Apply emphasis markers to your pronunciation tips to assist Siri with Chinese dialects.