Sample Code

Synchronizing a local store to the cloud

Share data between a user's devices and other iCloud users.

Download

iOS 16.0+ | iPadOS 16.0+ | Mac Catalyst 16.0+ | Xcode 14.1+

Overview

Note

This sample code project is associated with the WWDC22 session 10119: Optimize your use of Core Data and CloudKit .

Configure the sample code project

Before you run the sample code project in Xcode, do the following:

1. In Xcode's Project navigator, select the project, and click the Signing & Capabilities tab.
2. From the Team pop-up menu, choose your developer team.
3. In the Bundle Identifier field, enter a new bundle ID for the CoreDataCloudKitDemo target. The bundle identifier for the project has an associated App ID, so you need a unique identifier to create your own App ID. Use a reverse-DNS format for your identifier, as Preparing Your App for Distribution describes.
4. In the iCloud Capability Section, click the + button to create a new iCloud Container, or click a the checkbox next to an existing container you would like to use.
5. Select the CoreDataCloudKitDemoUnitTests target and use the Team pop-up menu to choose your developer team.

Xcode automatically generates provisioning profiles as needed. You can now build and run the CoreDataCloudKitDemo app or tests.

Run the CoreDataCloudKitDemo app

1. Select the CoreDataCloudKitDemo scheme.
2. Choose a destination. The CoreDataCloudKitDemo app supports the following destinations:
    - Any iOS simulator
    - Any iOS device
    - My Mac (Mac Catalyst)

Configuration options

To facilitate testing, the app supports the following configuration options the `AppDelegate` class parses into properties:

- `-CDCKDTesting`
    - Set to `1` to store files in the special directory `TestStores`, so that tests don't overwrite user data.
- `-CDCKDAllowCloudKitSync`
    - Set to `0` to disable CloudKit sync during testing.
- `com.apple.CoreData.ConcurrencyDebug`
    - Enable Core Data multithreading assertions to verify all Core Data operations use the correct queue.

See Also

CloudKit mirroring

Mirroring a Core Data store with CloudKit

Back user interfaces with a local replica of a CloudKit private database.

`class` `NSPersistentCloudKitContainer`

A container that encapsulates the Core Data stack in your app, and mirrors select persistent stores to a CloudKit private database.

`class` `NSPersistentCloudKitContainerOptions`

An object that customizes how a store description aligns with a CloudKit database.

{} Sharing Core Data objects between iCloud users

Use Core Data and CloudKit to synchronize data between devices of an iCloud user and share data between different iCloud users.