

[Foundation](#) / NSUbiquitousKeyValueStore

## Class

# NSUbiquitousKeyValueStore

An iCloud-based container of key-value pairs you use to share data among instances of your app running on a user's connected devices.

iOS 5.0+ | iPadOS 5.0+ | Mac Catalyst 13.1+ | macOS 10.7+ | tvOS 9.0+ | visionOS 1.0+ | watchOS 9.0+

```
class NSUbiquitousKeyValueStore
```

## Overview

Use the iCloud key-value store to make preference, configuration, and app-state data available to every instance of your app on every device connected to a user's iCloud account. You can store scalar values such as `BOOL`, as well as values containing any of the property list object types: [NSNumber](#), [NSString](#), [NSDate](#), [NSData](#), [NSArray](#), and [NSDictionary](#).

Changes your app writes to the key-value store object are initially held in memory, then written to disk by the system at appropriate times. If you write to the key-value store object when the user is not signed into an iCloud account, the data is stored locally until the next synchronization opportunity. When the user signs into an iCloud account, the system automatically reconciles your local, on-disk keys and values with those on the iCloud server.

Any device running your app, and attached to the same iCloud account, can upload key-value changes to iCloud. To keep track of such changes, register for the [didChangeExternallyNotification](#) notification during app launch. Then, obtain the keys and values from iCloud (which may be newer than those that are local) by calling the [synchronize\(\)](#) method. You need not call the [synchronize\(\)](#) method again during your app's life cycle, unless your app design requires fast-as-possible upload to iCloud after you change a value.

For more information on adopting key-value storage in your app, see [Designing for Key-Value Data in iCloud](#) in [iCloud Design Guide](#).

Avoid using this class for data that is essential to your app's behavior when offline; instead, store such data directly into the local user defaults database.

The total amount of space available in your app's key-value store, for a given user, is 1 MB. There is a per-key value size limit of 1 MB, and a maximum of 1024 keys. If you attempt to write data that exceeds these quotas, the write attempt fails and no change is made to your iCloud key-value storage. In this scenario, the system posts the didChangeExternallyNotification notification with a change reason of NSUbiquitousKeyValueStoreQuotaViolationChange.

The maximum length for key strings for the iCloud key-value store is 64 bytes using UTF8 encoding. Attempting to write a value to a longer key name results in a runtime error.

To use this class, you must distribute your app through the App Store or Mac App Store, and you must request the `com.apple.developer.ubiquity-kvstore-identifier` entitlement in your Xcode project. For more on this, see Configuring Common Key-Value Storage for Multiple Apps in iCloud Design Guide.

This class is not meant to be subclassed.

---

## Topics

### Getting the Shared Instance

```
class var `default`: NSUbiquitousKeyValueStore
```

Returns the shared iCloud key-value store object.

### Getting Values

```
func array(forKey: String) -> [Any]?
```

Returns the array associated with the specified key.

```
func bool(forKey: String) -> Bool
```

Returns the Boolean value associated with the specified key.

```
func data(forKey: String) -> Data?
```

Returns the data object associated with the specified key.

```
func dictionary(forKey: String) -> [String : Any]?
```

Returns the dictionary object associated with the specified key.

```
func double(forKey: String) -> Double
```

Returns the double value associated with the specified key.

```
func longLong(forKey: String) -> Int64
```

Returns the long long value associated with the specified key.

```
func object(forKey: String) -> Any?
```

Returns the object associated with the specified key.

```
func string(forKey: String) -> String?
```

Returns the string associated with the specified key.

## Setting Values

```
func set([Any]?, forKey: String)
```

Sets an array object for the specified key in the key-value store.

```
func set(Bool, forKey: String)
```

Sets a Boolean value for the specified key in the key-value store.

```
func set(Data?, forKey: String)
```

Sets a data object for the specified key in the key-value store.

```
func set([String : Any]?, forKey: String)
```

Sets a dictionary object for the specified key in the key-value store.

```
func set(Double, forKey: String)
```

Sets a double value for the specified key in the key-value store.

```
func set(Int64, forKey: String)
```

Sets a long long value for the specified key in the key-value store.

```
func set(Any?, forKey: String)
```

Sets an object for the specified key in the key-value store.

```
func set(String?, forKey: String)
```

Sets a string object for the specified key in the key-value store.

## Explicitly Synchronizing In-Memory Key-Value Data to Disk

```
func synchronize() -> Bool
```

Explicitly synchronizes in-memory keys and values with those stored on disk.

## Removing Keys

```
func removeObject(forKey: String)
```

Removes the value associated with the specified key from the key-value store.

## Retrieving the Current Keys and Values

```
var dictionaryRepresentation: [String : Any]
```

A dictionary containing all of the key-value pairs in the key-value store.

## Constants

⌵ Notification Keys

These keys can be included in the user info dictionary of the didChangeExternallyNotification notification.

⌵ Change Reason Values

Possible values associated with the NSUbiquitousKeyValueStoreChangeReasonKey key.

## Notifications

```
class let didChangeExternallyNotification: NSNotification.Name
```

Posted when the value of one or more keys in the local key-value store changed due to incoming data pushed from iCloud.

---

## Relationships

### Inherits From

NSObject

### Conforms To

CVarArg

CustomDebugStringConvertible

CustomStringConvertible  
Equatable  
Hashable  
NSObjectProtocol