Class

HKHealthStore

The access point for all data managed by HealthKit.

iOS 8.0+ | iPadOS 8.0+ | Mac Catalyst 13.0+ | macOS 13.0+ | visionOS 1.0+ | watchOS 2.0+

```
class HKHealthStore
```

Mentioned in

- Executing Observer Queries
- Reading data from HealthKit

Overview

Use a `HKHealthStore` object to request permission to share or read HealthKit data. After you have permission, you can use the HealthKit store to save new samples to the store, or to manage the samples that your app saved. Additionally, you can use the HealthKit store to start, stop, and manage queries.

For more information, see Setting up HealthKit.

Topics

Accessing HealthKit

```
func authorizationStatus(for: HKObjectType) -> HKAuthorizationStatus
```

Returns the app's authorization status for sharing the specified data type.

`enum HKAuthorizationStatus`

Constants indicating the authorization status for a particular data type.

`func getRequestStatusForAuthorization(toShare: Set<HKSampleType>, read: Set<HKObjectType>, completion: (HKAuthorizationRequestStatus, (any Error)?) -> Void)`

Indicates whether the system presents the user with a permission sheet if your app requests authorization for the provided types.

`enum HKAuthorizationRequestStatus`

Values that indicate whether your app needs to request authorization from the user.

`class func isHealthDataAvailable() -> Bool`

Returns a Boolean value that indicates whether HealthKit is available on this device.

`func supportsHealthRecords() -> Bool`

Returns a Boolean value that indicates whether the current device supports clinical records.

`func requestAuthorization(toShare: Set<HKSampleType>?, read: Set<HKObjectType>?, completion: (Bool, (any Error)?) -> Void)`

Requests permission to save and read the specified data types.

`func requestAuthorization(toShare: Set<HKSampleType>, read: Set<HKObjectType>) async throws`

Asynchronously requests permission to save and read the specified data types.

`func requestPerObjectReadAuthorization(for: HKObjectType, predicate: NSPredicate?, completion: (Bool, (any Error)?) -> Void)`

Asynchronously requests permission to read a data type that requires per-object authorization (such as vision prescriptions).

`func handleAuthorizationForExtension(completion: (Bool, (any Error)?) -> Void)`

Requests permission to save and read the data types specified by an extension.

`var authorizationViewControllerPresenter: UIViewController?`

The view controller that presents HealthKit authorization sheets.

Querying HealthKit data

`func execute(HKQuery)`

Starts executing the provided query.

`func stop(HKQuery)`

Stops a long-running query.

Reading characteristic data

`func biologicalSex() throws -> HKBiologicalSexObject`

Reads someone’s biological sex from the HealthKit store.

`func bloodType() throws -> HKBloodTypeObject`

Reads the user’s blood type from the HealthKit store.

~~`func dateOfBirth() throws -> Date`~~

Reads the user’s date of birth from the HealthKit store as a date value.

Deprecated

`func dateOfBirthComponents() throws -> DateComponents`

Reads the user’s date of birth from the HealthKit store as date components.

`func fitzpatrickSkinType() throws -> HKFitzpatrickSkinTypeObject`

Reads the user’s Fitzpatrick Skin Type from the HealthKit store.

`func wheelchairUse() throws -> HKWheelchairUseObject`

Reads the user’s wheelchair use from the HealthKit store.

Working with HealthKit objects

`func delete(HKObject, withCompletion: (Bool, (any Error)?) -> Void)`

Deletes the specified object from the HealthKit store.

`func delete([HKObject], withCompletion: (Bool, (any Error)?) -> Void)`

Deletes the specified objects from the HealthKit store.

```
func deleteObjects(of: HKObjectType, predicate: NSPredicate, with
Completion: (Bool, Int, (any Error)?) -> Void)
```

Deletes objects saved by this application that match the provided type and predicate.

`func earliestPermittedSampleDate() -> Date`

Returns the earliest date permitted for samples.

```
func save(HKObject, withCompletion: (Bool, (any Error)?) -> Void)
```

Saves the provided object to the HealthKit store.

```
func save([HKObject], withCompletion: (Bool, (any Error)?) -> Void)
```

Saves an array of objects to the HealthKit store.

Accessing the preferred units

```
func preferredUnits(for: Set<HKQuantityType>, completion: ([HKQuantity
Type : HKUnit], (any Error)?) -> Void)
```

Returns the user’s preferred units for the given quantity types.

```
static let HKUserPreferencesDidChange: NSNotification.Name
```

Notifies observers whenever the user changes his or her preferred units.

Managing background delivery

```
func enableBackgroundDelivery(for: HKObjectType, frequency: HKUpdate
Frequency, withCompletion: (Bool, (any Error)?) -> Void)
```

Enables the delivery of updates to an app running in the background.

```
com.apple.developer.healthkit.background-delivery
```

A Boolean value that indicates whether observer queries receive updates while running in the background.

```
enum HKUpdateFrequency
```

Constants that determine how often the system launches your app in response to changes to HealthKit data.

```
func disableBackgroundDelivery(for: HKObjectType, withCompletion: (Bool
, (any Error)?) -> Void)
```

Disables background deliveries of update notifications for the specified data type.

```
func disableAllBackgroundDelivery(completion: (Bool, (any Error)?) ->
Void)
```

Disables all background deliveries of update notifications.

Managing workouts

~~`func splitTotalEnergy(HKQuantity, start: Date, end: Date, resultsHandler: (HKQuantity?, HKQuantity?, (any Error)?) -> Void)`~~

Calculates the active and resting energy burned based on the total energy burned over the given duration.

Deprecated

`func recoverActiveWorkoutSession(completion: (HKWorkoutSession?, (any Error)?) -> Void)`

Recovers an active workout session.

Managing workout sessions

`var workoutSessionMirroringStartHandler: ((HKWorkoutSession) -> Void)?`

A block that the system calls when it starts a mirrored workout session.

`func startWatchApp(with: HKWorkoutConfiguration, completion: (Bool, (any Error)?) -> Void)`

Launches or wakes the companion watchOS app to create a new workout session.

`func pause(HKWorkoutSession)`

Pauses the provided workout session.

`func resumeWorkoutSession(HKWorkoutSession)`

Resumes the provided workout session.

Managing estimates

`func recalibrateEstimates(sampleType: HKSampleType, date: Date, completion: (Bool, (any Error)?) -> Void)`

Recalibrates the prediction algorithm used to calculate the specified sample type.

Accessing the move mode

`func activityMoveMode() throws -> HKActivityMoveModeObject`

Returns the activity move mode for the current user.

`static let HKUserPreferencesDidChange: NSNotification.Name`

Notifies observers whenever the user changes his or her preferred units.

Deprecated symbols

~~`func add([HKSample], to: HKWorkout, completion: (Bool, (any Error)?) -> Void)`~~

Associates the provided samples with the specified workout.

Deprecated

~~`func start(HKWorkoutSession)`~~

Starts a workout session for the current app.

Deprecated

~~`func end(HKWorkoutSession)`~~

Ends a workout session for the current app.

Deprecated

Instance Methods

`func relateWorkoutEffortSample(HKSample, with: HKWorkout, activity: HKWorkoutActivity?, completion: (Bool, (any Error)?) -> Void)`

`func unrelateWorkoutEffortSample(HKSample, from: HKWorkout, activity: HKWorkoutActivity?, completion: (Bool, (any Error)?) -> Void)`

Relationships

Inherits From

`NSObject`

Conforms To

`CVarArg`
`CustomDebugStringConvertible`
`CustomStringConvertible`
`Equatable`
`Hashable`
`NSObjectProtocol`

```
Sendable
SendableMetatype
```

See Also

Health data

- Saving data to HealthKit
  Create and share HealthKit samples.
- Reading data from HealthKit
  Use queries to request sample data from HealthKit.
- Creating a Mobility Health App
  Create a health app that allows a clinical care team to send and receive mobility data.
- Data types
  Specify the kind of data used in HealthKit.
- Samples
  Create and save health and fitness samples.
- Queries
  Query health and fitness data.
- Visualizing HealthKit State of Mind in visionOS
  Incorporate HealthKit State of Mind into your app and visualize the data in visionOS.
- Logging symptoms associated with a medication
  Fetch medications and dose events from the HealthKit store, and create symptom samples to associate with them.