Sample Code

Accessing a User's Clinical Records

Request authorization to query HealthKit for a user's clinical records and display them in your app.

Download

iOS 12.0+ | iPadOS 12.0+ | Xcode 13.2+

Overview

This sample demonstrates how to request access to a user's clinical records through HealthKit. With the HealthKit framework's clinical record support, you can read Fast Healthcare Interoperability Resources (FHIR) objects from the HealthKit store. People must first download their clinical records from one of the supported healthcare institutions before the records appear in HealthKit.

Configure the Sample Code Project

To use HealthKit, you must first enable the HealthKit capability and include the `NSHealthShareUsageDescription` key in your app's `Info.plist` file, as described in Accessing Health Records. To access the clinical records, check the Clinical Health Records checkbox in the HealthKit capability and include the `NSHealthClinicalHealthRecordsShareUsageDescription` key in your app's `Info.plist` file.

The sample app enables the capability and provides the usage string.

Before building and running the app:

1. Set a valid signing team in the target's General pane so that Xcode can create a provisioning profile containing the HealthKit entitlement when you build the app for the first time.

2. Add sample data to the Health app by connecting a valid patient portal account from a supported healthcare institution. If you don't have such an account, you can add sample data within the Simulator as described in Accessing Sample Data in the Simulator.

When you first run the app, it hasn't requested permission to read or share any data in HealthKit. Tapping any items in the list results in an Authorization not Determined error. To authorize the app, scroll to the bottom of the list, and tap the Authorize button.

Define the Sample Types to Request

The app defines the clinical record sample types using the `HKClinicalTypeIdentifier` enumeration. The app must request permission to read all the types that it intends to use. Note that the app may define both clinical records and standard HealthKit sample types at the same time.

```
/// An enumeration that defines two categories of data types: Health Records and Fit
/// Health Records enumerates the clinical records the app would like to access and
/// fitness data types.
enum Section {
    case healthRecords
    case fitnessData

    var displayName: String {
        switch self {
        case .healthRecords:
            return "Health Records"
        case .fitnessData:
            return "Fitness Data"
        }
    }

    var types: [HKSampleType] {
        switch self {
        case .healthRecords:
            return [
                HKObjectType.clinicalType(forIdentifier: .allergyRecord)!,
                HKObjectType.clinicalType(forIdentifier: .vitalSignRecord)!,
                HKObjectType.clinicalType(forIdentifier: .conditionRecord)!,
                HKObjectType.clinicalType(forIdentifier: .immunizationRecord)!,
                HKObjectType.clinicalType(forIdentifier: .labResultRecord)!,
                HKObjectType.clinicalType(forIdentifier: .medicationRecord)!,
                HKObjectType.clinicalType(forIdentifier: .procedureRecord)!
```

```
            ]

        case .fitnessData:
            return [
                HKObjectType.quantityType(forIdentifier: .stepCount)!,
                HKObjectType.quantityType(forIdentifier: .distanceWalkingRunning)!
            ]
        }
    }
}
```

Request Authorization

The app may request authorization to access both clinical data and HealthKit data simultaneously.

```
/// Create an instance of the health store. Use the health store to request authoriz
/// HealthKit records and to query for the records.
let healthStore = HKHealthStore()

var sampleTypes: Set<HKSampleType> {
    return Set(Section.healthRecords.types + Section.fitnessData.types)
}

/// Before accessing clinical records and other health data from HealthKit, the app
/// authorization. The health store's getRequestStatusForAuthorization method allows
/// if user has already granted authorization. If the user hasn't granted authorizat
/// requests authorization from the person using the app.
@objc
func requestAuthorizationIfNeeded(_ sender: AnyObject? = nil) {
    healthStore.getRequestStatusForAuthorization(toShare: Set(), read: sampleTypes)
        if status == .shouldRequest {
            self.requestAuthorization(sender)
        } else {
            DispatchQueue.main.async {
                let message = "Authorization status has been determined, no need to
                self.present(message: message, titled: "Already Requested")
            }
        }
    }
}

/// The health store's requestAuthorization method presents a permissions sheet to t
```

```
/// choose what data they allow the app to access.
@objc
func requestAuthorization(_ sender: AnyObject? = nil) {
    healthStore.requestAuthorization(toShare: nil, read: sampleTypes) { (success, er
        guard success else {
            DispatchQueue.main.async {
                self.handleError(error)
            }
            return
        }
    }
}
```

Typically, apps that read or share HealthKit data automatically request authorization–either when the app first launches or just before the app needs to access the data. However, to provide readers with complete control over the authorization process, this sample code doesn't automatically request authorization. Instead, anyone using the app must manually request authorization, by scrolling to the bottom of the list and tapping the Authorize button.

Query for Health Records

To query for clinical records, the app uses an HKSampleQuery as shown below.

```
/// Use HKSampleQuery to query the HealthKit store for samples by type.
func queryForSamples() {
    let sortDescriptors = [NSSortDescriptor(key: HKSampleSortIdentifierStartDate, as
    let query = HKSampleQuery(sampleType: sampleType, predicate: nil, limit: 100, so
        DispatchQueue.main.async {
            guard let samples = samplesOrNil else {
                self.handleError(error)
                return
            }

            self.samples = samples
            self.tableView.reloadData()
        }
    }

    healthStore.execute(query)
}
```

Access Elements Within a FHIR Resource

After a person using the app has given the app access to their clinical records, the app needs to extract the relevant information from the FHIR JSON data so that it can do something useful with it. For example, to display the status of a medication in the form of a `MedicationStatement` resource, the app needs to access the `MedicationStatement.status` element.

The app uses the FHIRModels library to parse the JSON data into Swift classes. For more details on FHIRModels, see Handling FHIR Without Getting Burned.

It uses a `JSONDecoder` to convert the resource's JSON data:

```
/// Each clincal record retrieved from HealthKit is associated with a FHIR Resource.
func decode(resource: HKFHIRResource) throws -> DisplayItemSubtitleConvertible {
    if #available(iOS 14.0, *) {
        switch resource.fhirVersion.fhirRelease {
        case .dstu2:
            return try decodeDSTU2(resource: resource)
        case .r4:
            return try decodeR4(resource: resource)
        default:
            throw FHIRResourceDecodingError.versionNotSupported(resource.fhirVersion
        }
    } else {
        return try decodeDSTU2(resource: resource) // On iOS 12 and 13, HealthKit al
    }
}
```

This provides direct access to the `status` element, so that the app can display it. It appears as the subtitle of the Medication list view.

```
extension ModelsDSTU2.MedicationStatement: DisplayItemSubtitleConvertible {
    var displayItemSubtitle: String {
        return self.status.value?.rawValue ?? "unknown"
    }
}
```

See Also

Medical records

Accessing Health Records

Read clinical record data from the HealthKit store.

Accessing Sample Data in the Simulator

Set up sample accounts to build and test your app.

{} Accessing Data from a SMART Health Card

Query for and validate a verifiable clinical record.

`class HKClinicalRecord`

A sample that stores a clinical record.

`class HKFHIRResource`

An object containing Fast Healthcare Interoperability Resources (FHIR) data.

`class HKVerifiableClinicalRecord`

A sample that represents the contents of a SMART Health Card or EU Digital COVID Certificate.

`class HKVerifiableClinicalRecordSubject`

The subject associated with a signed clinical record.

`class HKCDADocumentSample`

A Clinical Document Architecture (CDA) sample that stores a single document.

`class HKDocumentSample`

An abstract class that represents a health document in the HealthKit store.

`static let CDA: HKDocumentTypeIdentifier`

The CDA Document type identifier, used when requesting permission to read or share CDA documents.

`class HKDocumentType`

A sample type used to create queries for documents.