What's New in HealthKit

Changes in iOS 9

Session 203

Shannon Tan iOS Software Engineer
Allan Shortlidge iOS Software Engineer

Roadmap

What’s new in HealthKit

Overview

Unit preferences

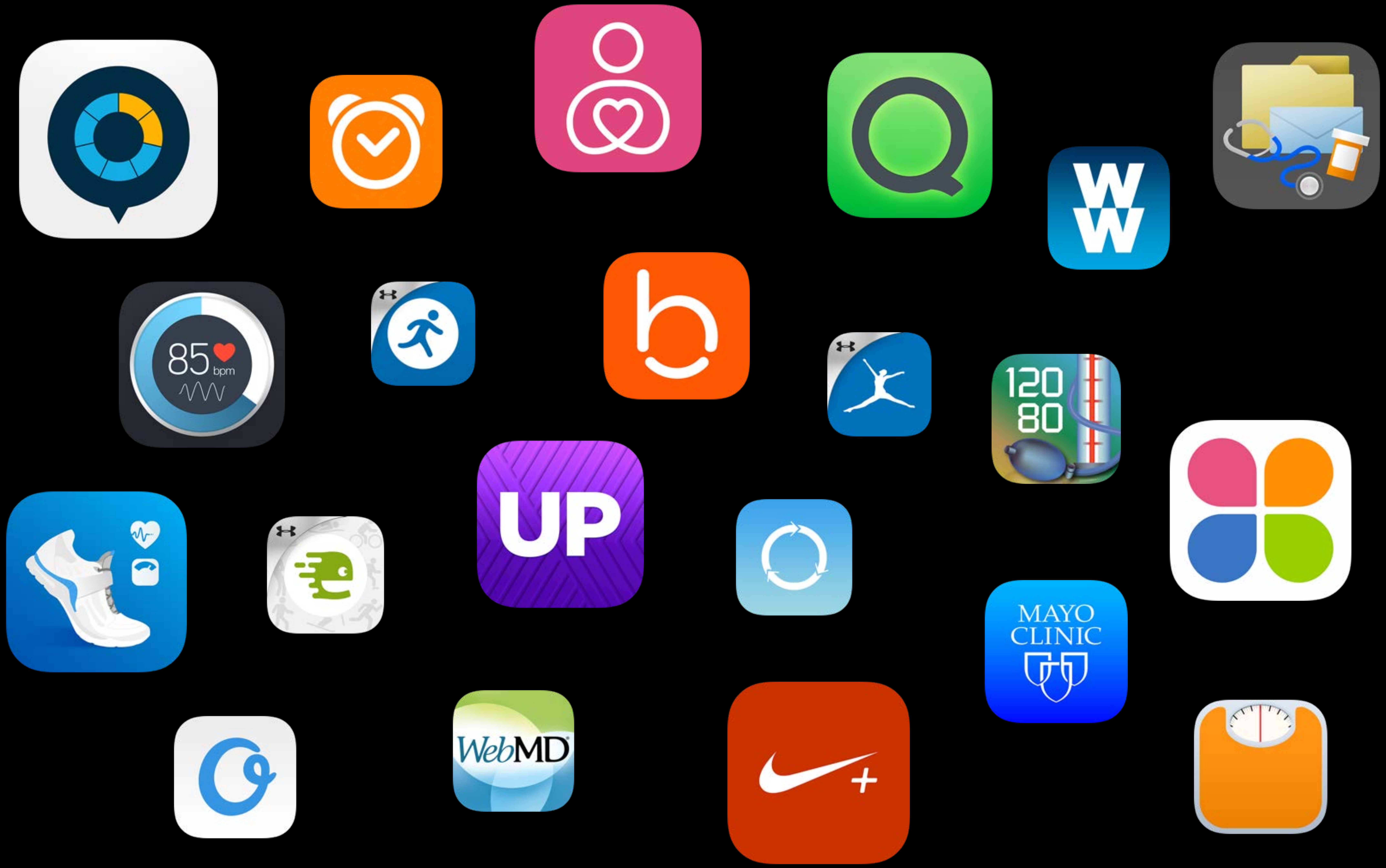
New data types

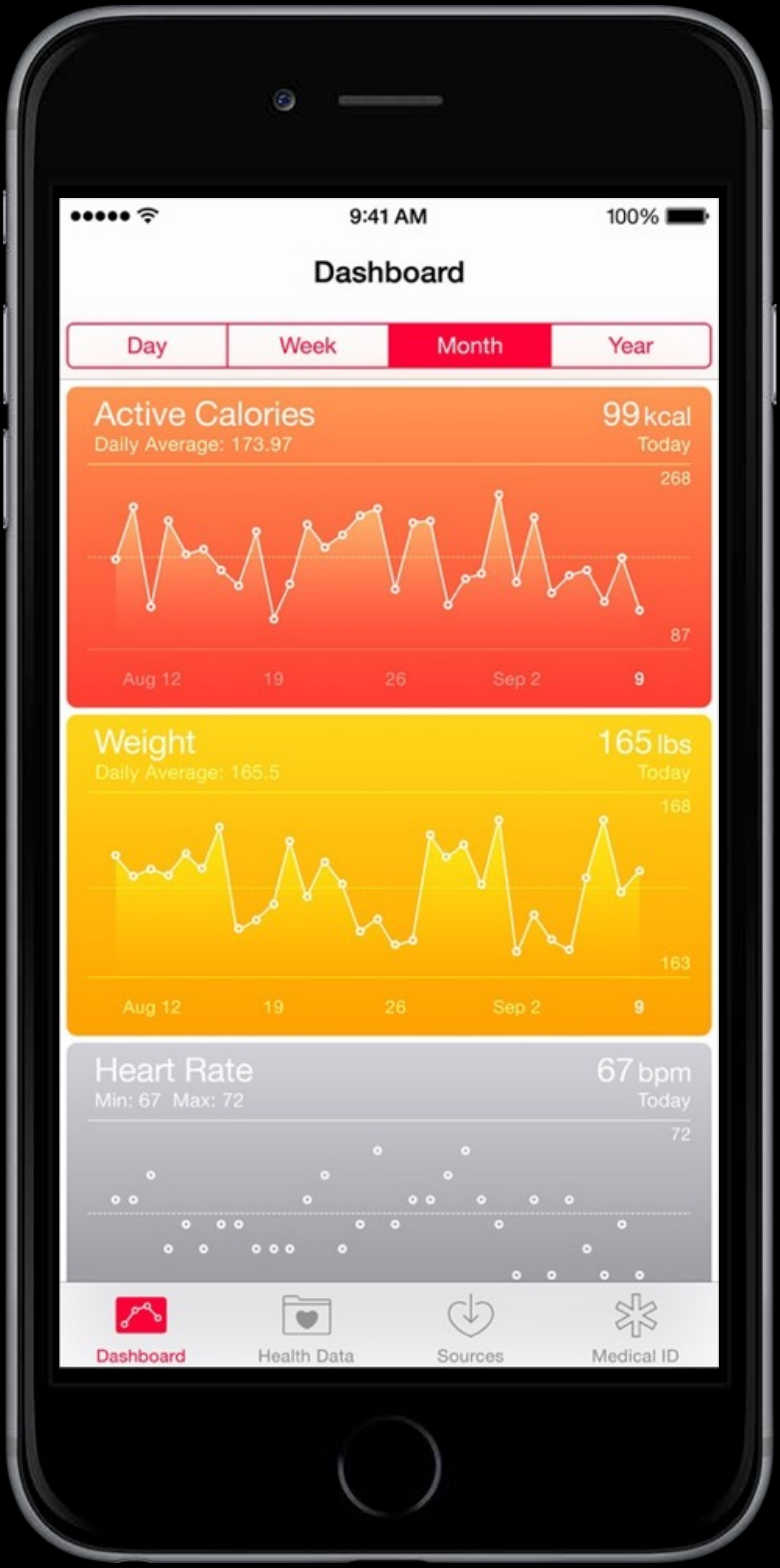
Source revisions and devices

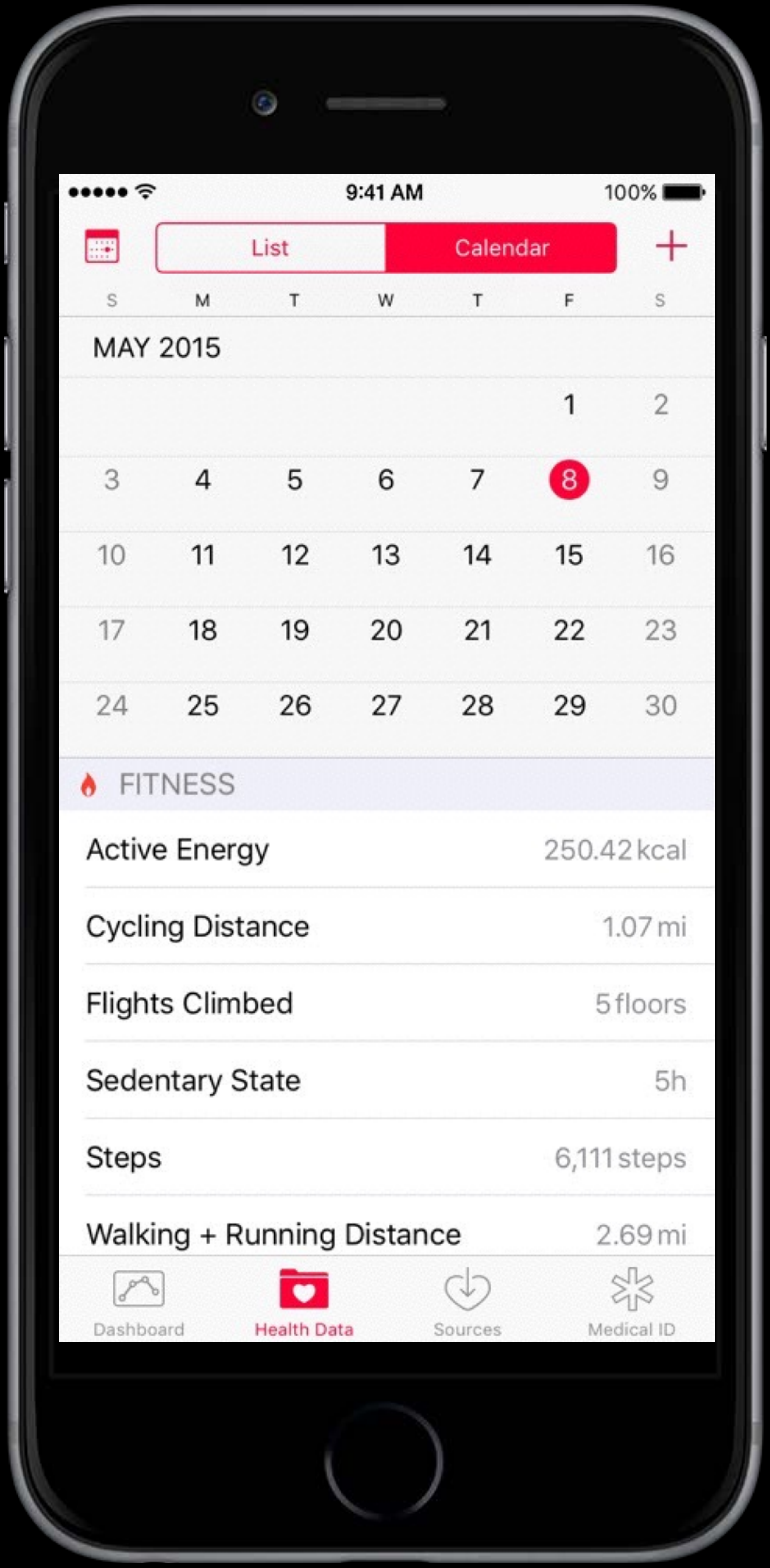
Deleted sample queries

Workout sessions

WatchKit demo


W W
85 bpm
UP
120 80
MAYO CLINIC
WebMD


9:41 AM
100%
Dashboard
Day
Week
Month
Year
Active Calories
99 kcal
Daily Average: 173.97
Today
268
87
Aug 12
19
26
Sep 2
9
Weight
165 lbs
Daily Average: 165.5
Today
168
163
Aug 12
19
26
Sep 2
9
Heart Rate
67 bpm
Min: 67 Max: 72
Today
72
Dashboard
Health Data
Sources
Medical ID



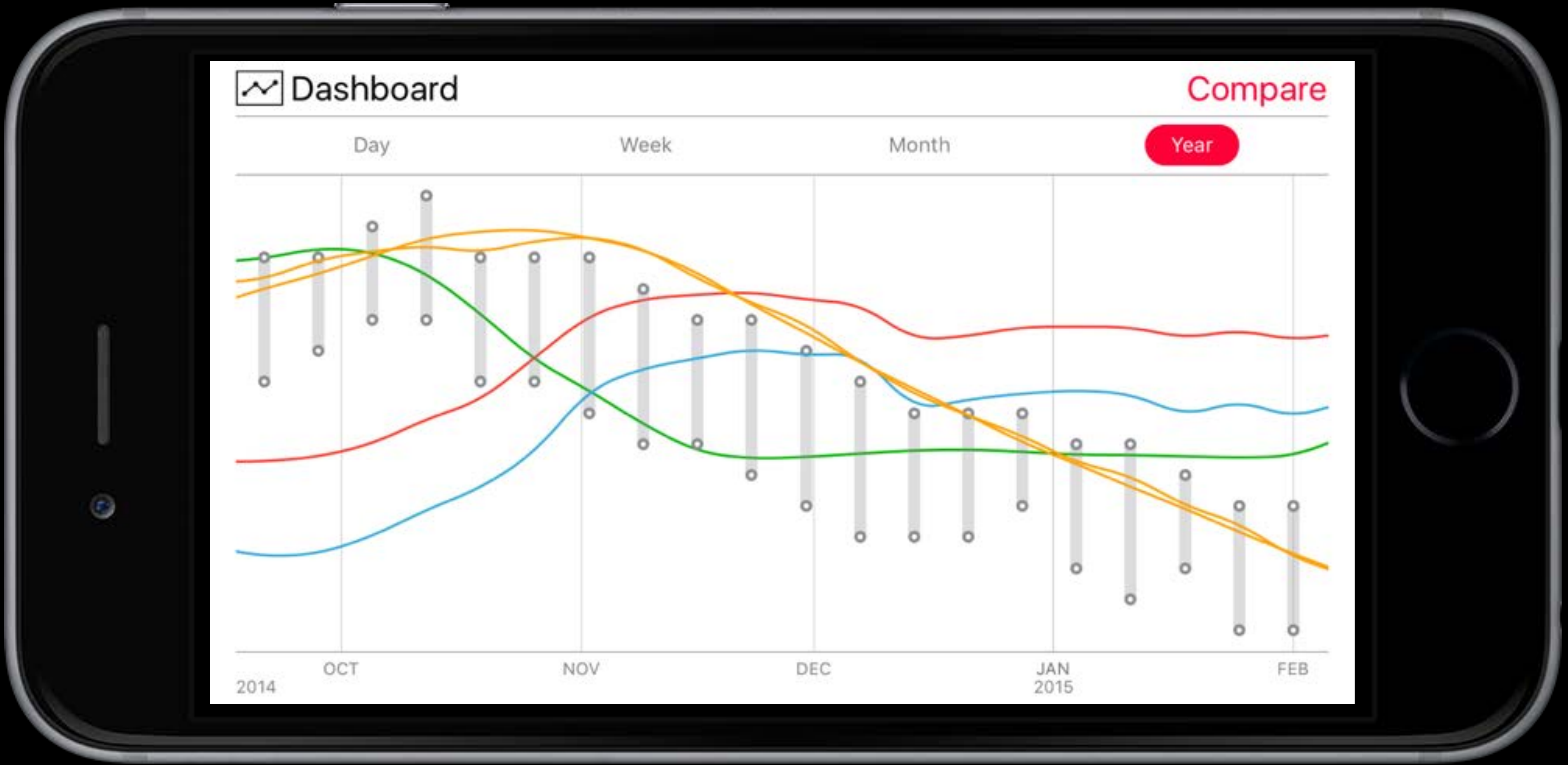
9:41 AM 100%

List | Calendar

S	M	T	W	T	F	S
MAY 2015						
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

FITNESS

Active Energy	250.42 kcal
Cycling Distance	1.07 mi
Flights Climbed	5 floors
Sedentary State	5h
Steps	6,111 steps
Walking + Running Distance	2.69 mi



Data Types and Units

HealthKit overview

Nike Fuel

Steps

Heart Rate

Vitamin C

Calories

Cycling Distance

Body Mass

Body Temperature

Oxygen Saturation

BMI

Blood Pressure

Vitamin B6

Potassium

Flights Climbed

Vitamin B12

BAC

Body Fat Percentage

Height

Vitamin A

Inhaler Usage

Respiratory Rate

Vitamin D

Blood Glucose

Adding Data Types

HealthKit overview

Adding Data Types

HealthKit overview

Existing hardware

Adding Data Types

HealthKit overview

Existing hardware

Existing software

Adding Data Types

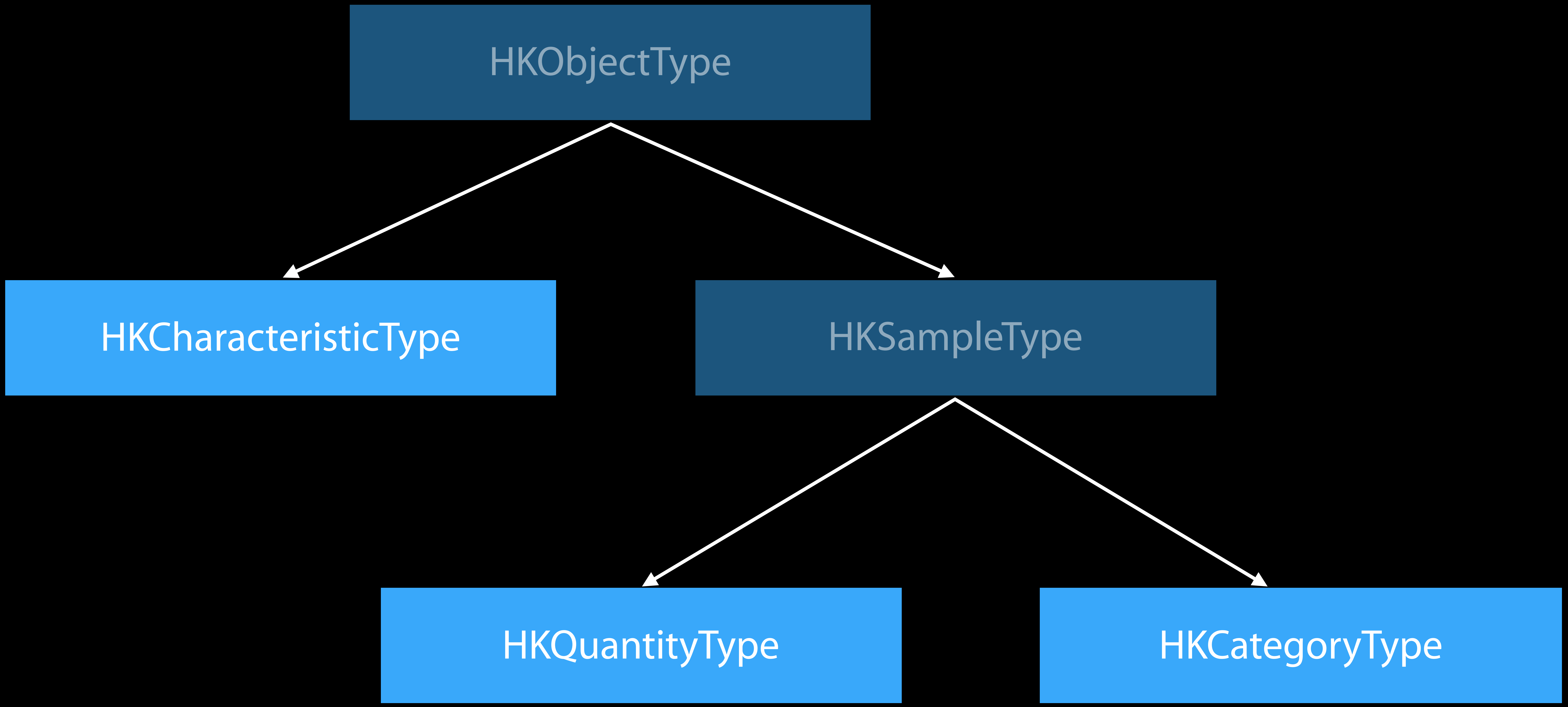
HealthKit overview

Existing hardware

Existing software

Customer and developer feedback

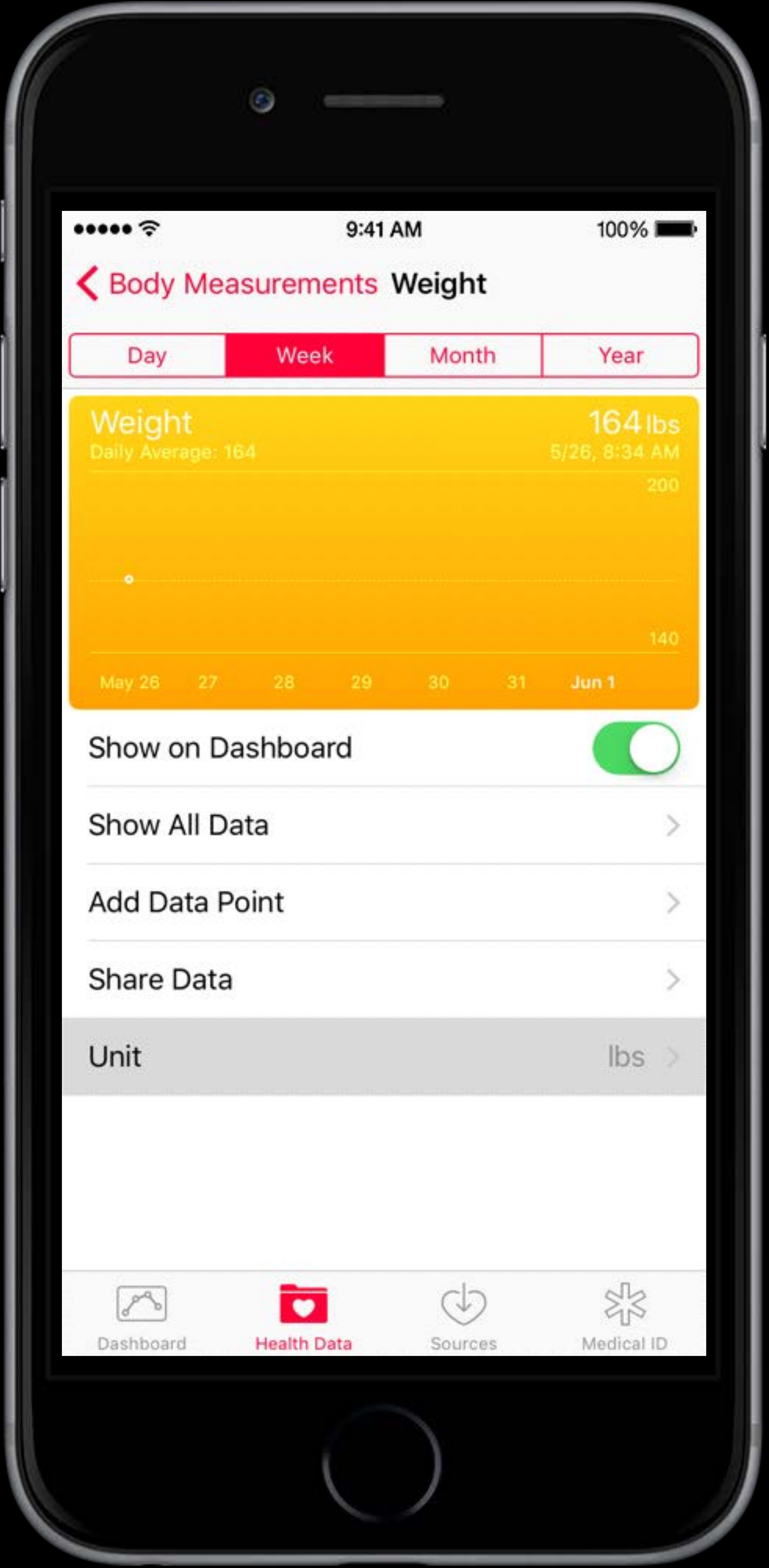
HKObjectType



For More Information

Unit Preferences

New in iOS 8.2



9:41 AM 100%

Body Measurements **Weight**

Day | Week | Month | Year

Weight 164 lbs
Daily Average: 164
5/26, 8:34 AM
200
140
May 26 27 28 29 30 31 Jun 1

Show on Dashboard

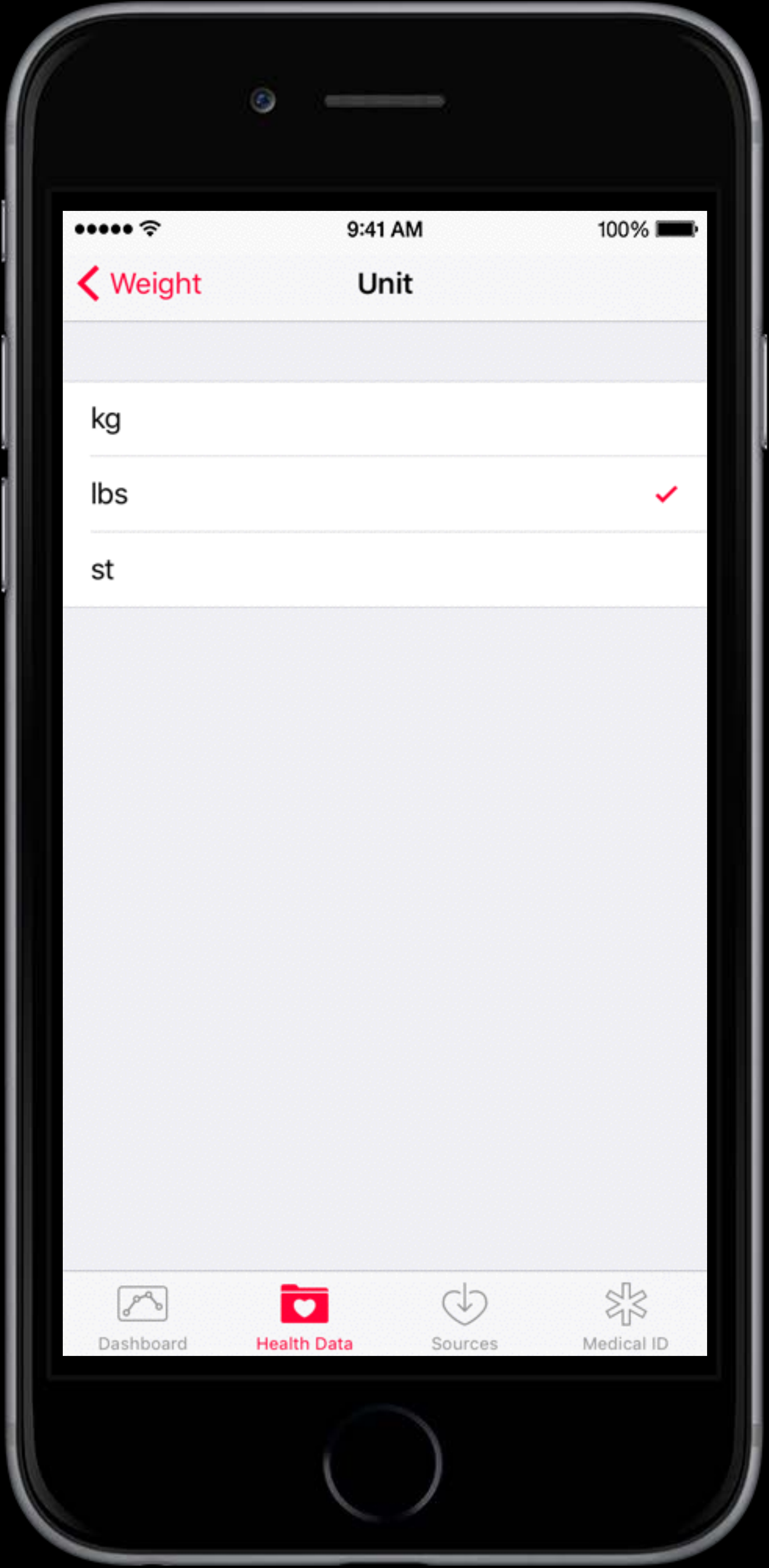
Show All Data

Add Data Point

Share Data

Unit lbs

Dashboard Health Data Sources Medical ID



9:41 AM 100%

Weight **Unit**

kg

lbs ✓

st

Dashboard Health Data Sources Medical ID

Unit Preferences

```
extension HKHealthStore {
    func preferredUnitsForQuantityTypes(quantityTypes: Set<HKQuantityType>,
       completion: ([HKQuantityType : HKUnit], NSError?) -> Void)
}

let HKUserPreferencesDidChangeNotification: String
```

New Data Types

Water Intake

New data types

Many apps track water intake

HKQuantityTypeIdentifierDietaryWater

Volume, cumulative

UV Exposure

New data types

HKQuantityTypeIdentifierUVExposure

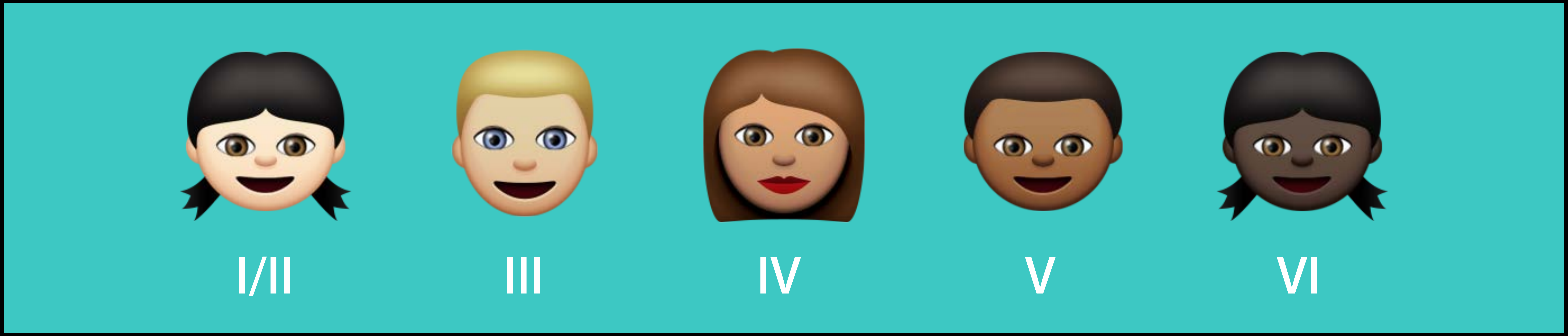
Unit: Scalar (UV index)

Fitzpatrick Skin Type

New data types

HKCharacteristicTypeIdentifierFitzpatrickSkinType

Types I-VI, or not set



Fitzpatrick Skin Type

New data types

```
class HKHealthStore : NSObject {
    …
    func fitzpatrickSkinTypeWithError() throws -> HKFitzpatrickSkinTypeObject
}

class HKFitzpatrickSkinTypeObject : NSObject {
   var skinType: HKFitzpatrickSkinType { get }
}
```

Reproductive Health

New data types

Reproductive Health

New data types

Basal body temperature

Cervical mucus quality

Ovulation

Menstrual flow

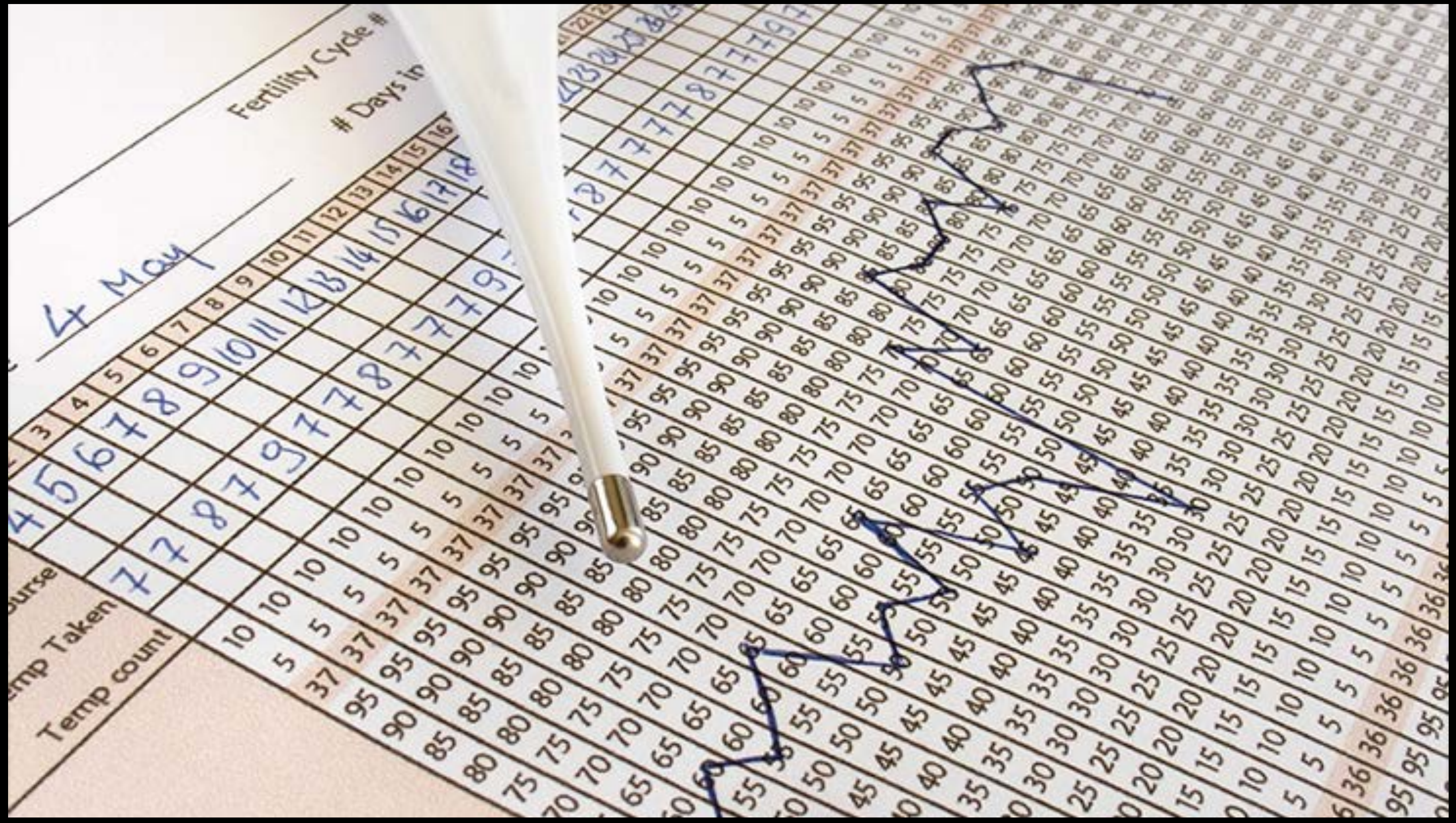
Vaginal spotting

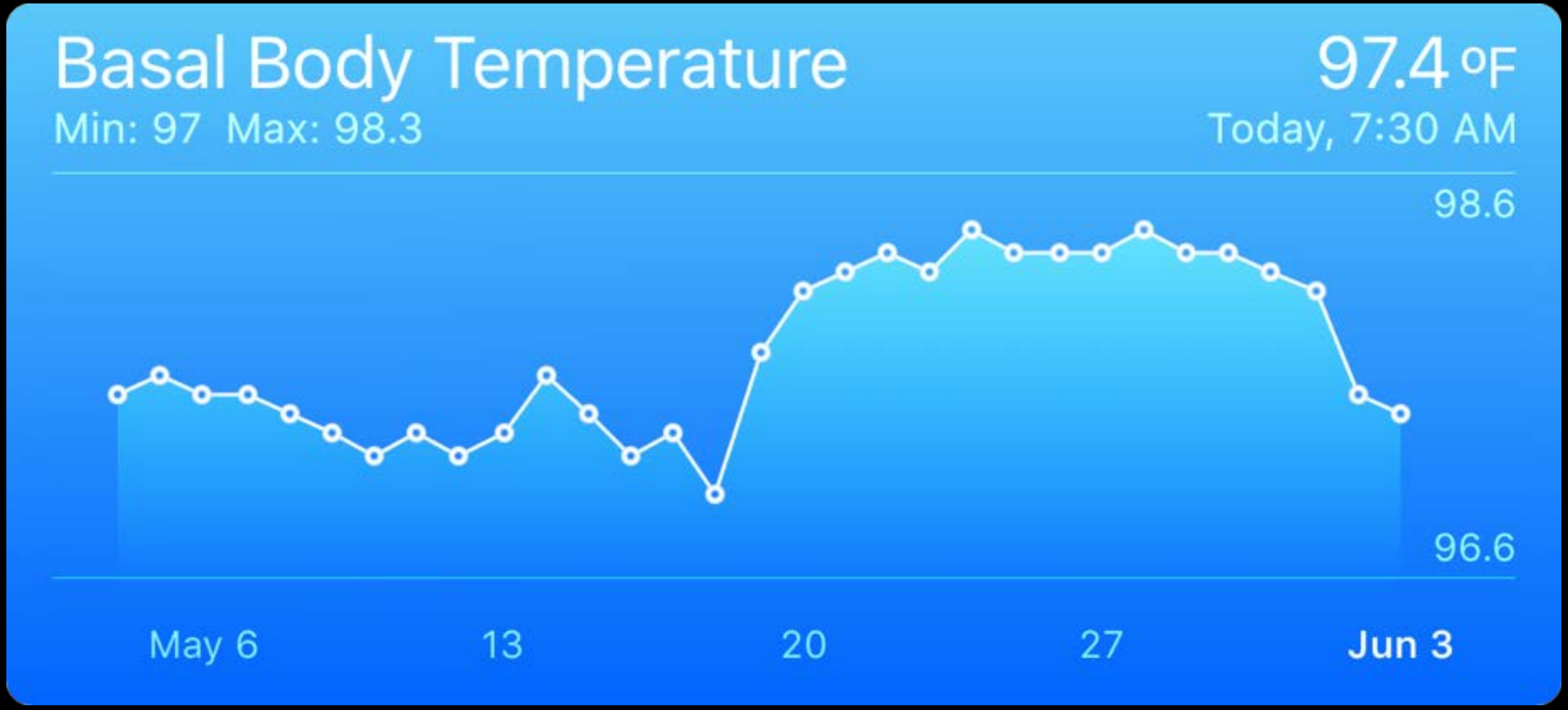
Sexual activity

Basal Body Temperature

New data types

```
// HKTypeIdentifiers
let HKQuantityTypeIdentifierBasalBodyTemperature: String
```



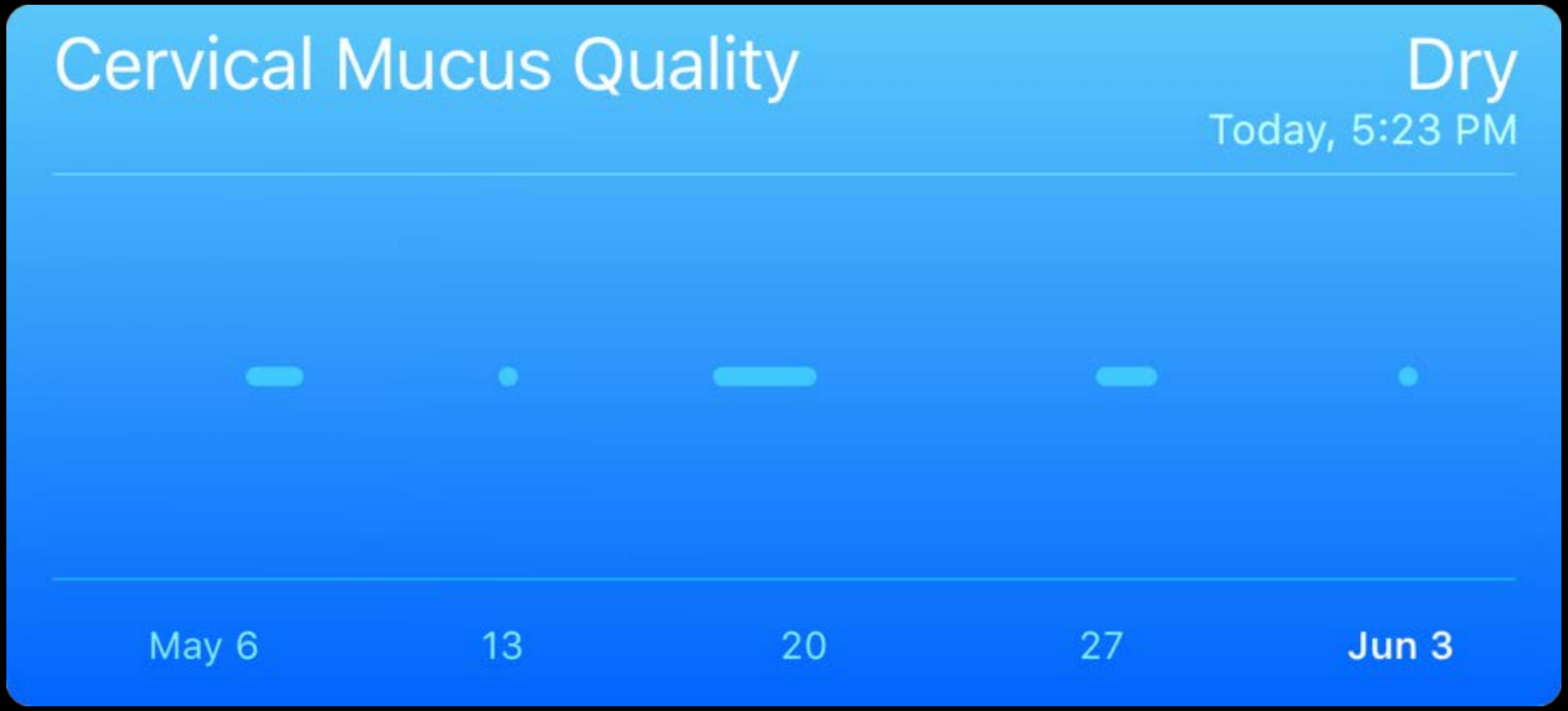


Cervical Mucus Quality

New data types

```
// HKTypeIdentifiers
let HKCategoryTypeIdentifierCervicalMucusQuality: String

// HKDefines
enum HKCategoryValueCervicalMucusQuality : Int {
    case Dry
    case Sticky
    case Creamy
    case Watery
    case EggWhite
}
```

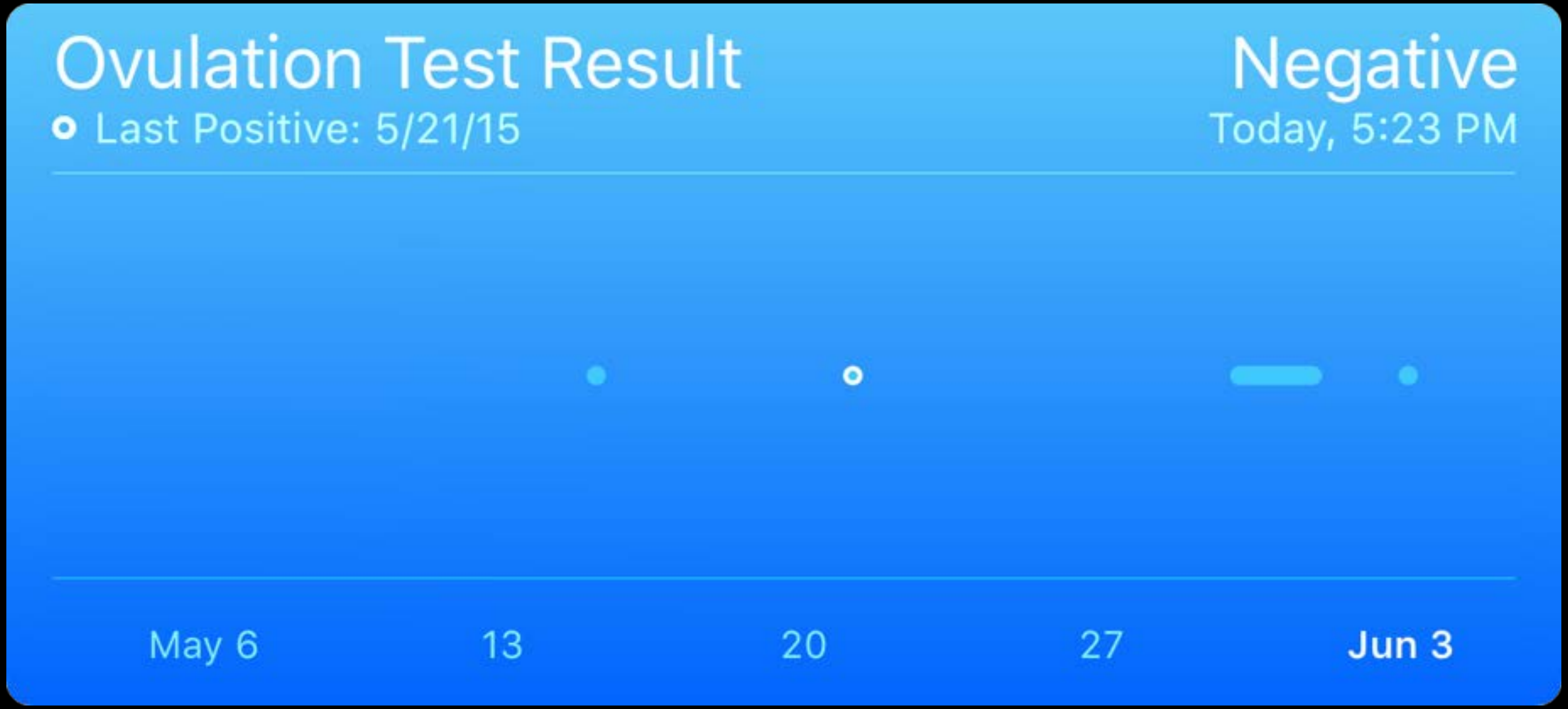


Ovulation Test Result

New data types

```
// HKTypeIdentifiers
let HKCategoryTypeIdentifierOvulationTestResult: String

// HKDefines
enum HKCategoryValueOvulationTestResult : Int {
    case Negative
    case Positive
}
```

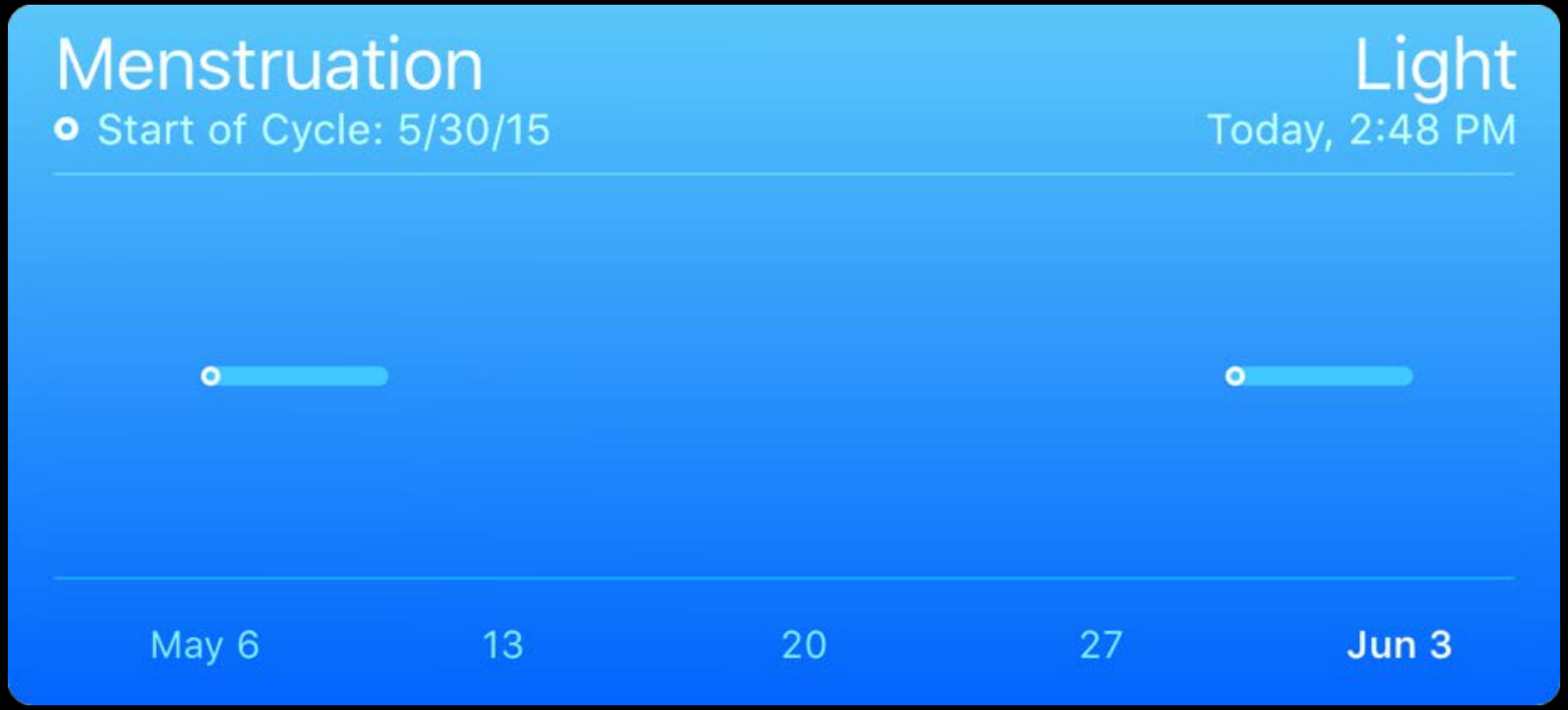


Menstruation

New data types

```
// HKTypeIdentifiers
let HKCategoryTypeIdentifierMenstrualFlow: String

// HKDefines
enum HKCategoryValueMenstrualFlow : Int {
    case Unspecified
    case Light
    case Medium
    case Heavy
}
```



Menstruation: Metadata

New data types

```
// HKMetadata
// The expected value is a Boolean
// Required
let HKMetadataKeyMenstrualCycleStart: String
```

Metadata Example

New data types

```
let healthStore: HKHealthStore = HKHealthStore()

let dict = [HKMetadataKeyMenstrualCycleStart: true]

let type = HKCategoryType.categoryTypeForIdentifier(
    HKCategoryTypeIdentifierMenstrualFlow)!
let value = HKCategoryValueMenstrualFlow.Unspecified.rawValue
let date = NSDate()

let sample = HKCategorySample(type: type, value: value, startDate: date,
endDate: date, metadata: dict)

healthStore.saveObject(sample) { … }
```

Metadata Example

New data types

```
let healthStore: HKHealthStore = HKHealthStore()

let dict = [HKMetadataKeyMenstrualCycleStart: true]

let type = HKCategoryType.categoryTypeForIdentifier(
    HKCategoryTypeIdentifierMenstrualFlow)!
let value = HKCategoryValueMenstrualFlow.Unspecified.rawValue
let date = NSDate()

let sample = HKCategorySample(type: type, value: value, startDate: date,
endDate: date, metadata: dict)

healthStore.saveObject(sample) { … }
```

Metadata Example

New data types

```
let healthStore: HKHealthStore = HKHealthStore()

let dict = [HKMetadataKeyMenstrualCycleStart: true]

let type = HKCategoryType.categoryTypeForIdentifier(
    HKCategoryTypeIdentifierMenstrualFlow)!
let value = HKCategoryValueMenstrualFlow.Unspecified.rawValue
let date = NSDate()

let sample = HKCategorySample(type: type, value: value, startDate: date,
endDate: date, metadata: dict)

healthStore.saveObject(sample) { … }
```

Single-Value Category Samples

New data types

Next two types will be category samples

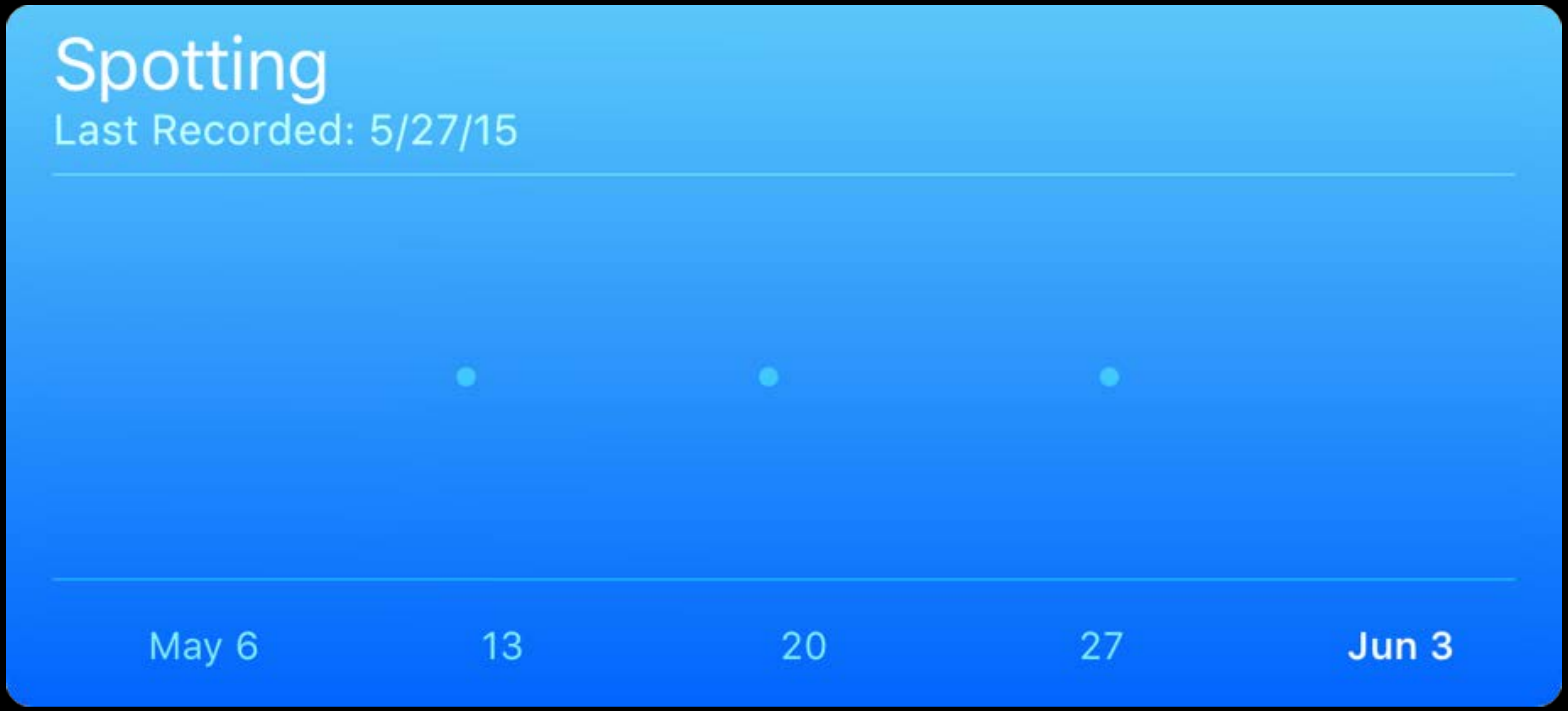
New category sample value: HKCategoryValueNotApplicable

Used for category samples that have dates, metadata, and other attributes, but do not have multiple values

Vaginal Spotting

New data types

```
// HKTypeIdentifiers
let HKCategoryTypeIdentifierVaginalSpotting: String
```

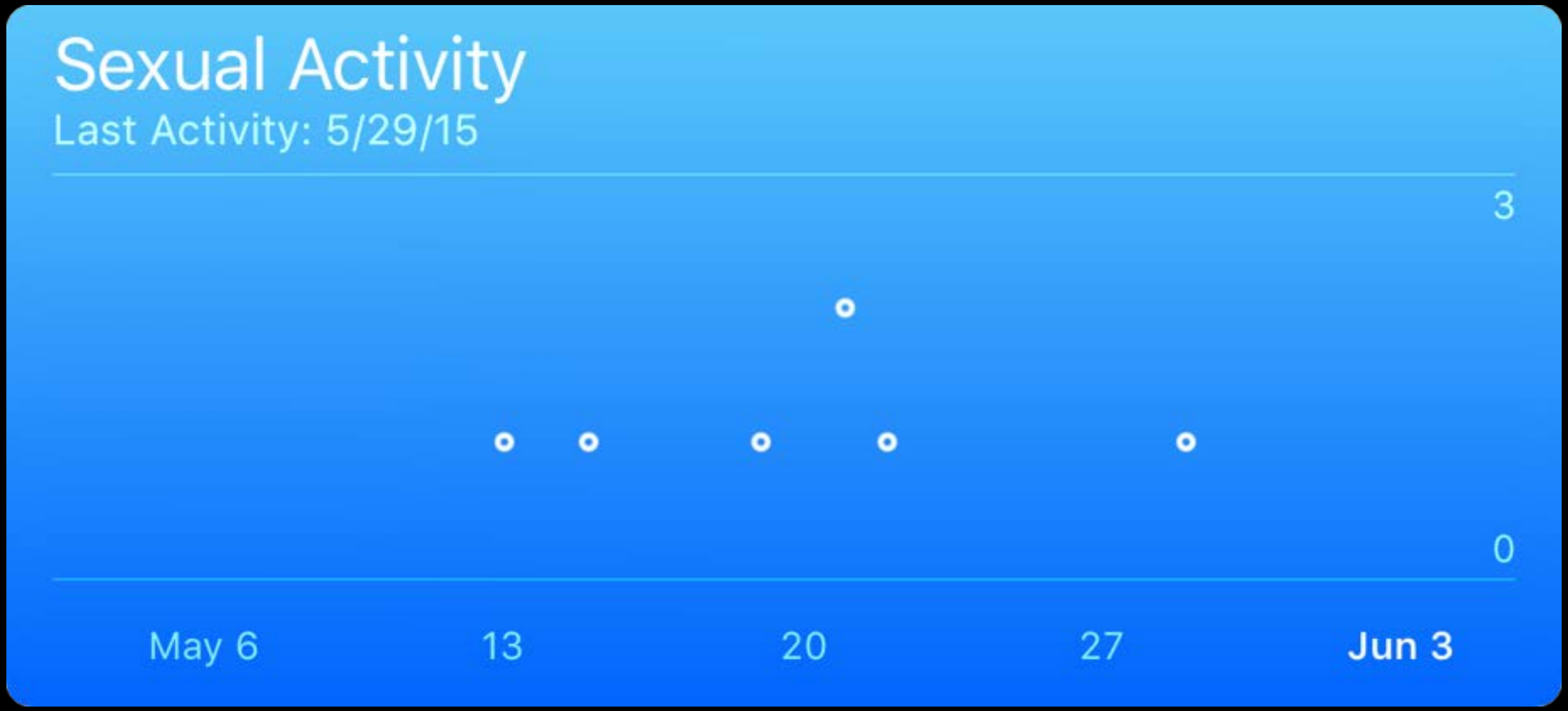


Sexual Activity

New data types

```
// HKTypeIdentifiers
let HKCategoryTypeIdentifierSexualActivity: String

// HKMetadata
// The expected value is a Boolean
let HKMetadataKeySexualActivityProtectionUsed: String
```



Summary

New data types

Water intake

UV exposure/Fitzpatrick Skin Type

Six reproductive health types

Data Sources

M8

M8

M8

M8

Sources and Device Metadata

Data sources

```
// HKObject
class HKObject {
    var source: HKSource { get }
    var metadata: [String : AnyObject]? { get }



}

// HKMetadata
let HKMetadataKeyDeviceName: String
let HKMetadataKeyDeviceManufacturerName: String
…
```

Sources and Device Metadata

Data sources

```
// HKObject
class HKObject {
    var source: HKSource { get }
    var metadata: [String : AnyObject]? { get }



}

// HKMetadata
let HKMetadataKeyDeviceName: String
let HKMetadataKeyDeviceManufacturerName: String
…
```

Sources and Device Metadata

Data sources

```
// HKObject
class HKObject {
    var source: HKSource { get }
    var metadata: [String : AnyObject]? { get }


}

// HKMetadata
let HKMetadataKeyDeviceName: String
let HKMetadataKeyDeviceManufacturerName: String
…
```

Object Source Revisions and Devices

Data sources

```
// HKObject
class HKObject {
    ~~var source: HKSource { get }~~
    var metadata: [String : AnyObject]? { get }
    var sourceRevision: HKSourceRevision { get }
    var device: HKDevice? { get }
}

// HKMetadata
let HKMetadataKeyDeviceName: String
let HKMetadataKeyDeviceManufacturerName: String
…
```

Source Revisions

Data sources

```
class HKSourceRevision {
    var source: HKSource { get }
    var version: String? { get }   // kCFBundleVersionKey
}
```

Devices

Data sources

```
class HKDevice {
    var name: String? { get }
    var manufacturer: String? { get }
    var model: String? { get }
    var hardwareVersion: String? { get }
    var firmwareVersion: String? { get }
    var softwareVersion: String? { get }
    var localIdentifier: String? { get }
    var UDIDeviceIdentifier: String? { get }


}
```

Devices

Data sources

```
class HKDevice {
    var name: String? { get }
    var manufacturer: String? { get }
    var model: String? { get }
    var hardwareVersion: String? { get }
    var firmwareVersion: String? { get }
    var softwareVersion: String? { get }
    var localIdentifier: String? { get }
    var UDIDeviceIdentifier: String? { get }

    class func localDevice() -> HKDevice
}
```

Saving Objects with Devices

Data sources

```
let device = HKDevice(name: "Scale", manufacturer: … )

let sample = HKQuantitySample(type: quantityType,
    quantity: quantity,
    startDate: startDate,
    endDate: endDate,
    device: device,
    metadata: nil)

healthStore.saveObject(sample) { … }
```

Saving Objects with Devices

Data sources

```
let device = HKDevice(name: "Scale", manufacturer: … )

let sample = HKQuantitySample(type: quantityType,
    quantity: quantity,
    startDate: startDate,
    endDate: endDate,
    device: device,
    metadata: nil)

healthStore.saveObject(sample) { … }
```

Saving Objects with Devices

Data sources

```
let device = HKDevice(name: "Scale", manufacturer: … )

let sample = HKQuantitySample(type: quantityType,
    quantity: quantity,
    startDate: startDate,
    endDate: endDate,
    device: device,
    metadata: nil)

healthStore.saveObject(sample) { … }
```

Saving Objects with Devices

Data sources

```
let device = HKDevice(name: "Scale", manufacturer: … )

let sample = HKQuantitySample(type: quantityType,
    quantity: quantity,
    startDate: startDate,
    endDate: endDate,
    device: device,
    metadata: nil)

healthStore.saveObject(sample) { … }
```

Querying for Devices and Source Revisions

Data sources

```
extension HKQuery {
    class func predicateForObjectsFromSourceRevisions(
        sourceRevisions: Set<HKSourceRevision>) -> NSPredicate

    class func predicateForObjectsFromDevices(
        devices: Set<HKDevice>) -> NSPredicate
}
```

Querying for Devices and Source Revisions

Data sources

```
let source = HKSource.defaultSource()
let revision = HKSourceRevision(source: source, version: "1.0")

let pred =
HKQuery.predicateForObjectsFromSourceRevisions([revision])

let query = HKSampleQuery(sampleType: sampleType,
    predicate: pred,
    limit: 0,
    sortDescriptors: nil) { … }

healthStore.executeQuery(query)
```

Querying for Devices and Source Revisions

Data sources

```
let source = HKSource.defaultSource()
let revision = HKSourceRevision(source: source, version: "1.0")

let pred =
HKQuery.predicateForObjectsFromSourceRevisions([revision])

let query = HKSampleQuery(sampleType: sampleType,
    predicate: pred,
    limit: 0,
    sortDescriptors: nil) { … }

healthStore.executeQuery(query)
```

Querying for Devices and Source Revisions

Data sources

```
let source = HKSource.defaultSource()
let revision = HKSourceRevision(source: source, version: "1.0")

let pred =
HKQuery.predicateForObjectsFromSourceRevisions([revision])

let query = HKSampleQuery(sampleType: sampleType,
    predicate: pred,
    limit: 0,
    sortDescriptors: nil) { … }

healthStore.executeQuery(query)
```

Querying for Devices and Source Revisions

Data sources

```
let source = HKSource.defaultSource()
let revision = HKSourceRevision(source: source, version: "1.0")

let pred =
HKQuery.predicateForObjectsFromSourceRevisions([revision])

let query = HKSampleQuery(sampleType: sampleType,
    predicate: pred,
    limit: 0,
    sortDescriptors: nil) { … }

healthStore.executeQuery(query)
```

Deletion

Deleting Objects

Deletion

```
class HKHealthStore {
    func deleteObject(object: HKObject,
        completion: (Bool, NSError?) -> Void)







}
```

Deleting Multiple Objects

Deletion

```
class HKHealthStore {
    func deleteObject(object: HKObject,
        completion: (Bool, NSError?) -> Void)

    func deleteObjects(objects: [HKObject],
        completion: (Bool, NSError?) -> Void)



}
```

Deleting Multiple Objects

Deletion

```
class HKHealthStore {
    func deleteObject(object: HKObject,
        completion: (Bool, NSError?) -> Void)

    func deleteObjects(objects: [HKObject],
        completion: (Bool, NSError?) -> Void)

    func deleteObjectsOfType(objectType: HKObjectType,
        predicate: NSPredicate,
        completion: (Bool, UInt, NSError?) -> Void)
}
```

Querying for Deleted Objects

Deletion

No easy way to determine which samples were deleted with iOS 8

Difficult to synchronize another database with HealthKit

Anchored Queries for Deleted Objects

Deletion

```
class HKAnchoredObjectQuery {
    init(type: HKSampleType,
        predicate: NSPredicate?,
        anchor: Int,
        limit: UInt,
        resultsHandler: (HKAnchoredObjectQuery, [HKSample]?,
            [HKDeletedObject]?, Int, NSError?) -> Void)




}
```

Deleted Objects

Deletion

```
class HKDeletedObject {
    var UUID: NSUUID { get }
}
```

Background Delivery of Deleted Objects

Deletion

```
class HKHealthStore {
    func enableBackgroundDeliveryForType(type: HKObjectType,
        frequency: HKUpdateFrequency,
        completion: (Bool, NSError?) -> Void)
}
```

Streaming Updates

Deletion

```
class HKAnchoredObjectQuery {
    init(type: HKSampleType,
        predicate: NSPredicate?,
        anchor: Int,
        limit: UInt,
        resultsHandler: (HKAnchoredObjectQuery, [HKSample]?,
            [HKDeletedObject]?, Int, NSError?) -> Void)

    var updateHandler: ((HKAnchoredObjectQuery, [HKSample]?,
        [HKDeletedObject]?, Int, NSError?) -> Void)?
}
```

Streaming Updates

Deletion

```
class HKAnchoredObjectQuery {
    init(type: HKSampleType,
        predicate: NSPredicate?,
        anchor: Int,
        limit: UInt,
        resultsHandler: (HKAnchoredObjectQuery, [HKSample]?,
            [HKDeletedObject]?, Int, NSError?) -> Void)

    var updateHandler: ((HKAnchoredObjectQuery, [HKSample]?,
        [HKDeletedObject]?, Int, NSError?) -> Void)?
}
```

HealthKit for watchOS

Using HealthKit in Apple Watch Apps

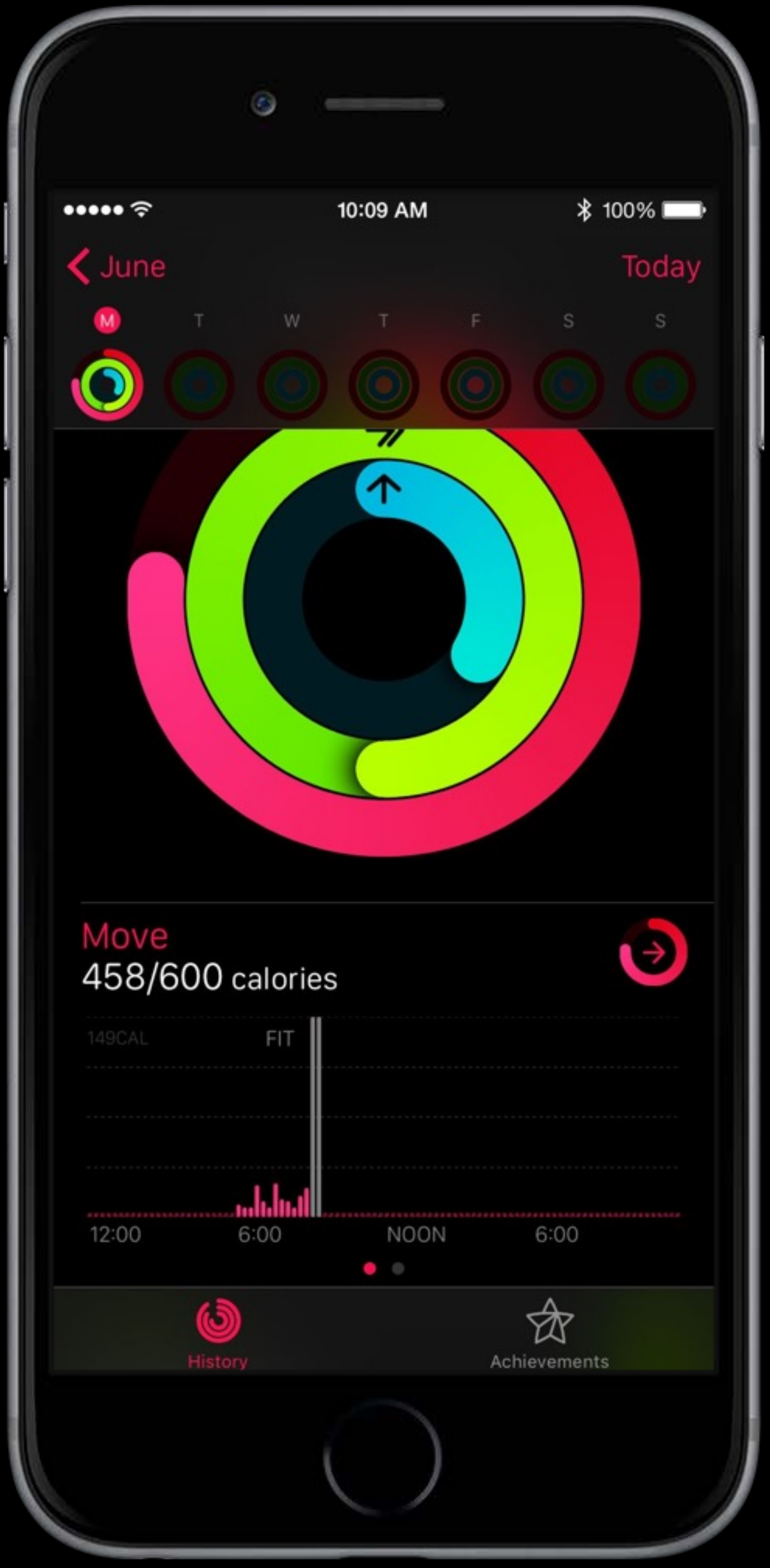
HealthKit on watchOS

Same APIs as iOS (limited historical data)

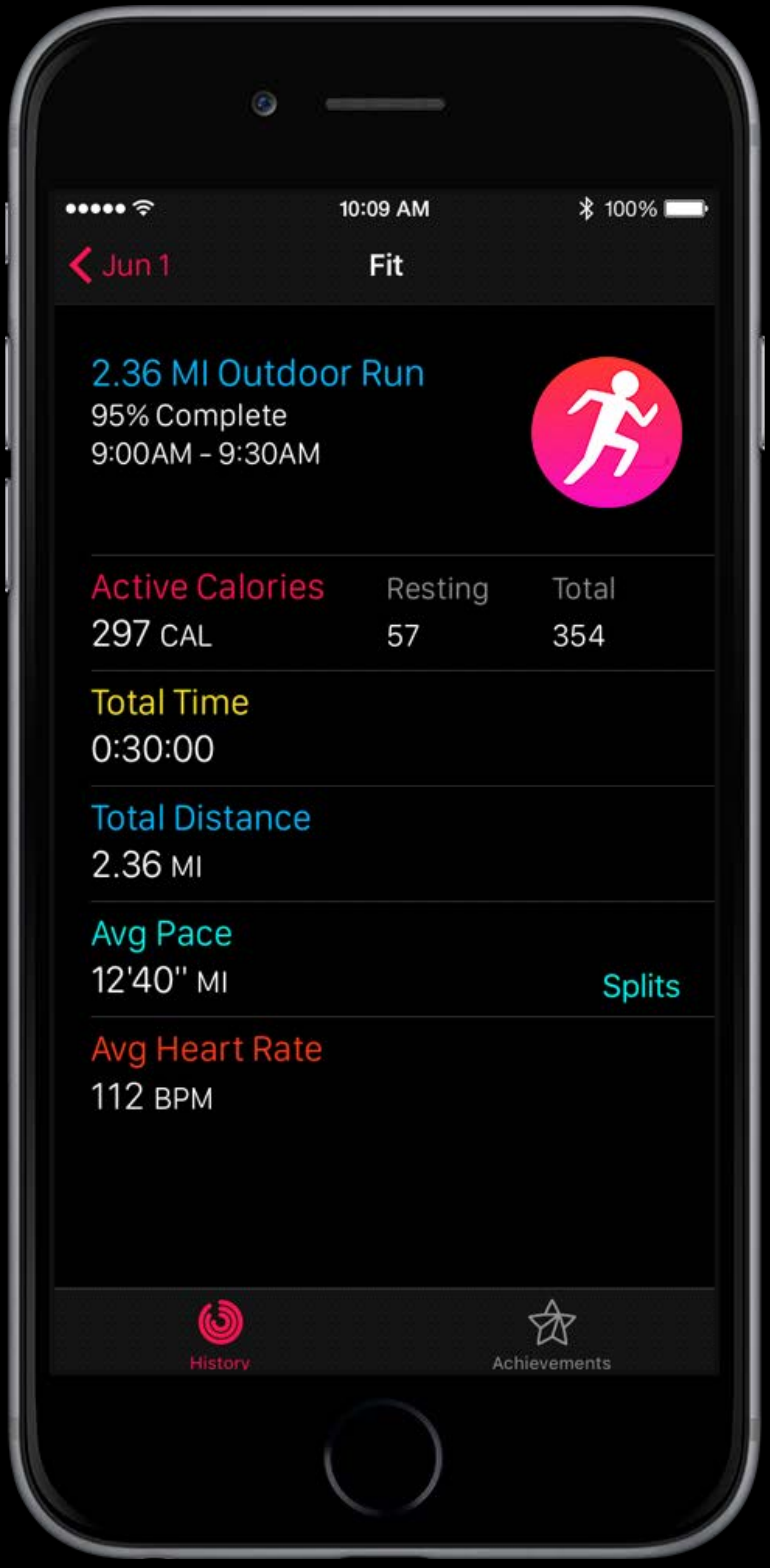
Access to activity and workout data

New workout APIs

Data syncs to companion device



Move
458/600 calories



2.36 MI Outdoor Run
95% Complete
9:00AM - 9:30AM

Active Calories	Resting	Total
297 CAL	57	354

Total Time
0:30:00

Total Distance
2.36 MI

Avg Pace
12'40'' MI

Splits

Avg Heart Rate
112 BPM

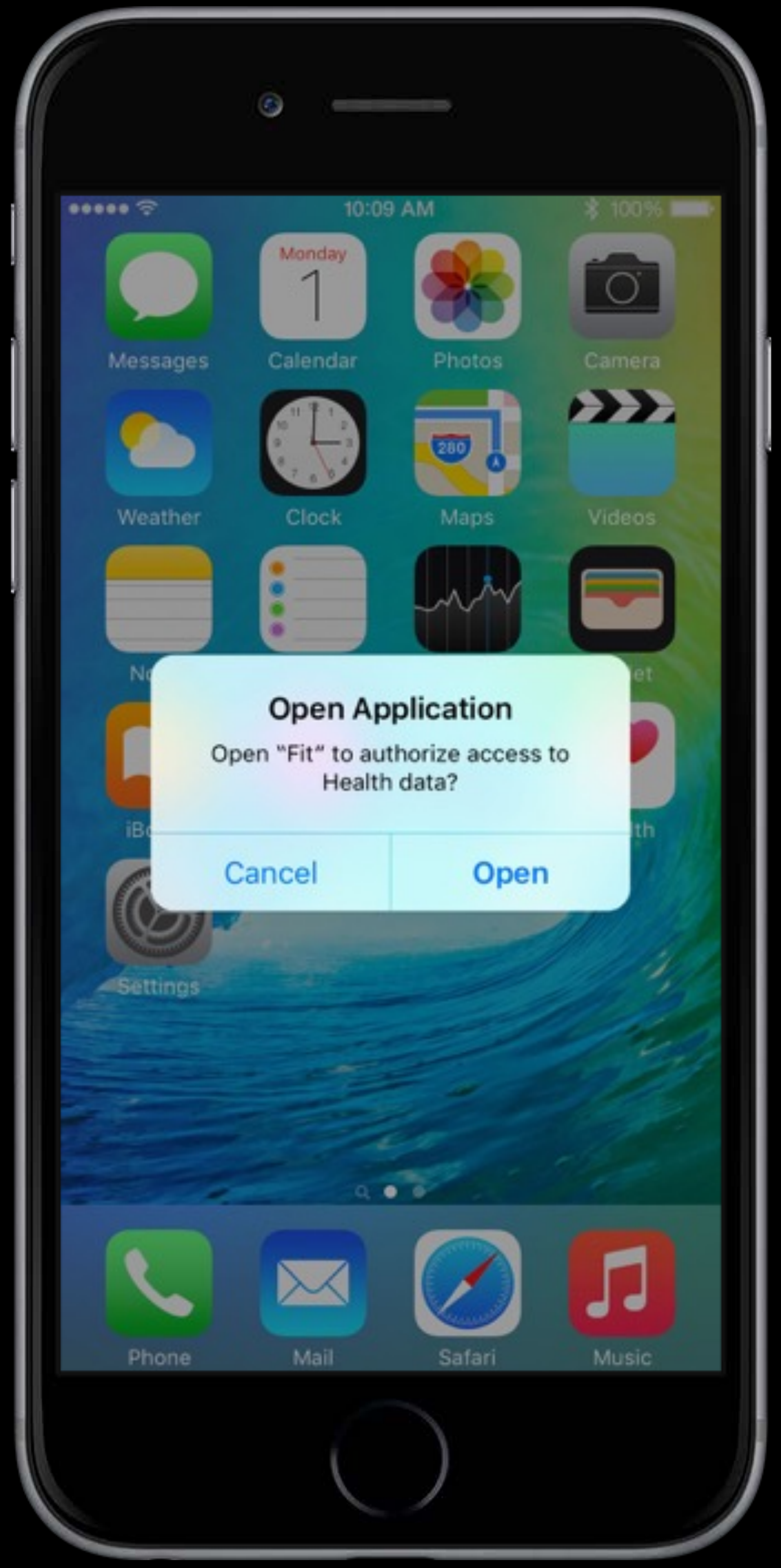
Privacy

HealthKit on watchOS

WatchKit apps must request authorization

User prompted on companion device

Authorizations shared between devices

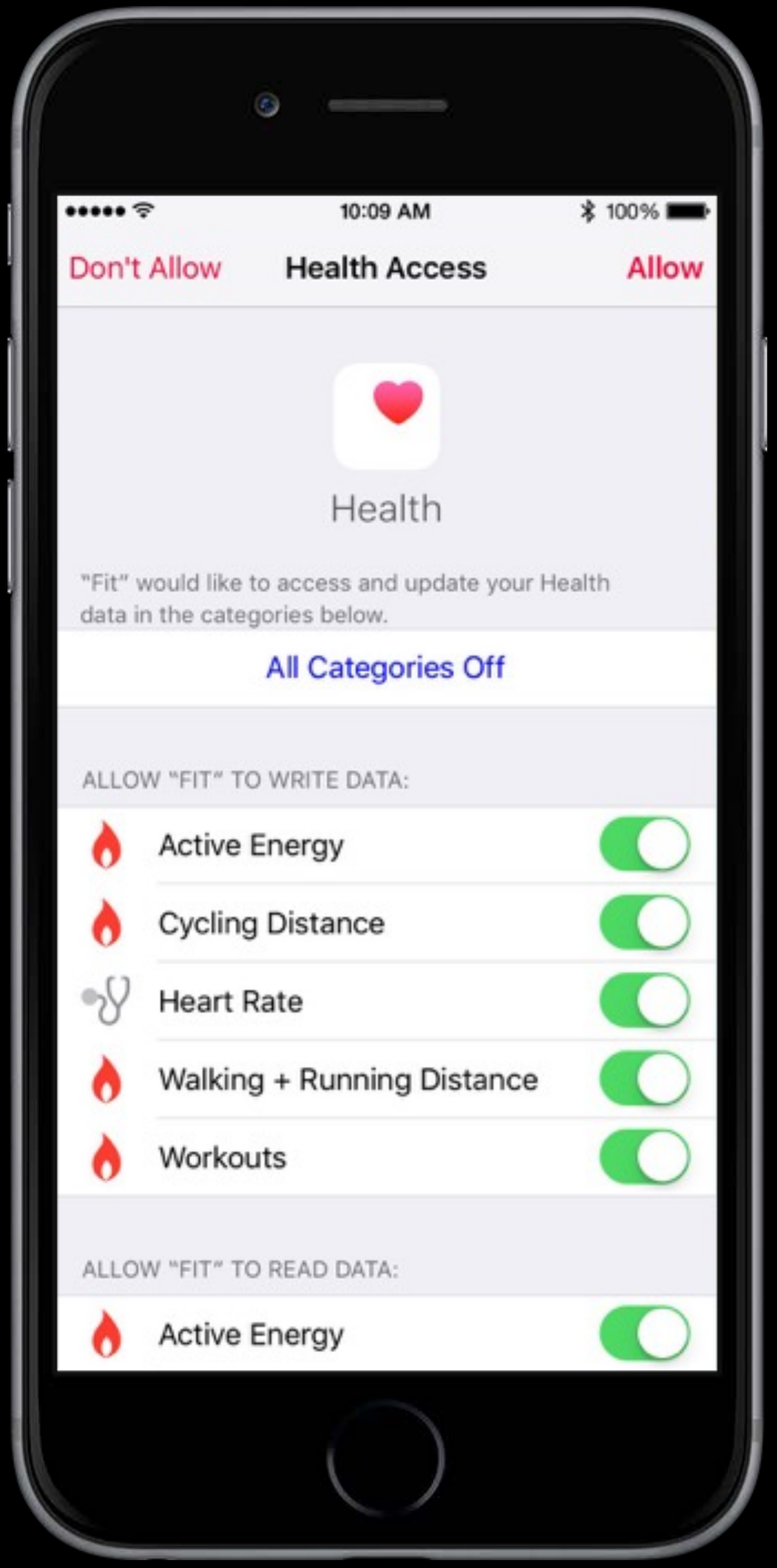


Authorization
HealthKit on watchOS

WatchKit apps must request authorization

User prompted on companion device

Authorizations shared between devices


10:09 AM 100%

Don't Allow | Health Access | Allow

Health

"Fit" would like to access and update your Health data in the categories below.

All Categories Off

ALLOW "FIT" TO WRITE DATA:

Active Energy	On
Cycling Distance	On
Heart Rate	On
Walking + Running Distance	On
Workouts	On

ALLOW "FIT" TO READ DATA:

Active Energy	On

Recording Workouts

HealthKit on watchOS

Use HKWorkoutSession while recording a workout

Specify activity type to improve accuracy

Apps stay foregrounded while in session

Only one workout session may run at a time

Workout Sessions

HealthKit on watchOS

```
class HKWorkoutSession : NSObject {
    var activityType: HKWorkoutActivityType { get }
    var locationType: HKWorkoutSessionLocationType { get }
    weak var delegate: HKWorkoutSessionDelegate?

    init(activityType: HKWorkoutActivityType,
        locationType: HKWorkoutSessionLocationType)
}
```

Workout Sessions

HealthKit on watchOS

```
class HKWorkoutSession : NSObject {
    var activityType: HKWorkoutActivityType { get }
    var locationType: HKWorkoutSessionLocationType { get }
    weak var delegate: HKWorkoutSessionDelegate?

    init(activityType: HKWorkoutActivityType,
        locationType: HKWorkoutSessionLocationType)
}
```

Workout Sessions

HealthKit on watchOS

```
class HKWorkoutSession : NSObject {
    var activityType: HKWorkoutActivityType { get }
    var locationType: HKWorkoutSessionLocationType { get }
    weak var delegate: HKWorkoutSessionDelegate?

    init(activityType: HKWorkoutActivityType,
        locationType: HKWorkoutSessionLocationType)
}
```

Workout Session Delegate

HealthKit on watchOS

```
protocol HKWorkoutSessionDelegate {
    func workoutSession(workoutSession: HKWorkoutSession,
        didChangeToState toState: HKWorkoutSessionState,
        fromState: HKWorkoutSessionState,
        date: NSDate)

    func workoutSession(workoutSession: HKWorkoutSession,
        didFailWithError error: NSError)
}
```

Workout Session Delegate

HealthKit on watchOS

```
protocol HKWorkoutSessionDelegate {
    func workoutSession(workoutSession: HKWorkoutSession,
        didChangeToState toState: HKWorkoutSessionState,
        fromState: HKWorkoutSessionState,
        date: NSDate)

    func workoutSession(workoutSession: HKWorkoutSession,
        didFailWithError error: NSError)
}
```

Workout Session Delegate

HealthKit on watchOS

```
protocol HKWorkoutSessionDelegate {
    func workoutSession(workoutSession: HKWorkoutSession,
        didChangeToState toState: HKWorkoutSessionState,
        fromState: HKWorkoutSessionState,
        date: NSDate)

    func workoutSession(workoutSession: HKWorkoutSession,
        didFailWithError error: NSError)
}
```

Starting and Stopping Workout Sessions

HealthKit on watchOS

```
healthStore.startWorkoutSession(workoutSession) { … }

…

healthStore.stopWorkoutSession(workoutSession) { … }
```

Demo

Demo Recap

HealthKit on watchOS

Requested authorization for workout data types

Started an HKWorkoutSession

Streamed samples with HKAnchoredObjectQuery

Saved HKWorkout and associated samples

Summary

Unit preferences

New data types

Source revisions and devices

Deleted sample queries

Workout sessions

WatchKit demo

More Information

Documentation

http://developer.apple.com/healthkit

Technical Support

Apple Developer Forums

http://developer.apple.com/forums

Developer Technical Support

http://developer.apple.com/support/technical

General Inquiries

healthkit@apple.com

Related Sessions and Labs

HealthKit and ResearchKit Lab	Frameworks Lab B	Wednesday 11:00AM
Building Apps with ResearchKit	Mission	Wednesday 4:30PM
Health, Fitness, and Research Get Together	Buena Vista Park	Wednesday 6:00PM
HealthKit and ResearchKit Lab	Frameworks Lab C	Thursday 11:00AM

WWDC15