App Frameworks

#WWDC16

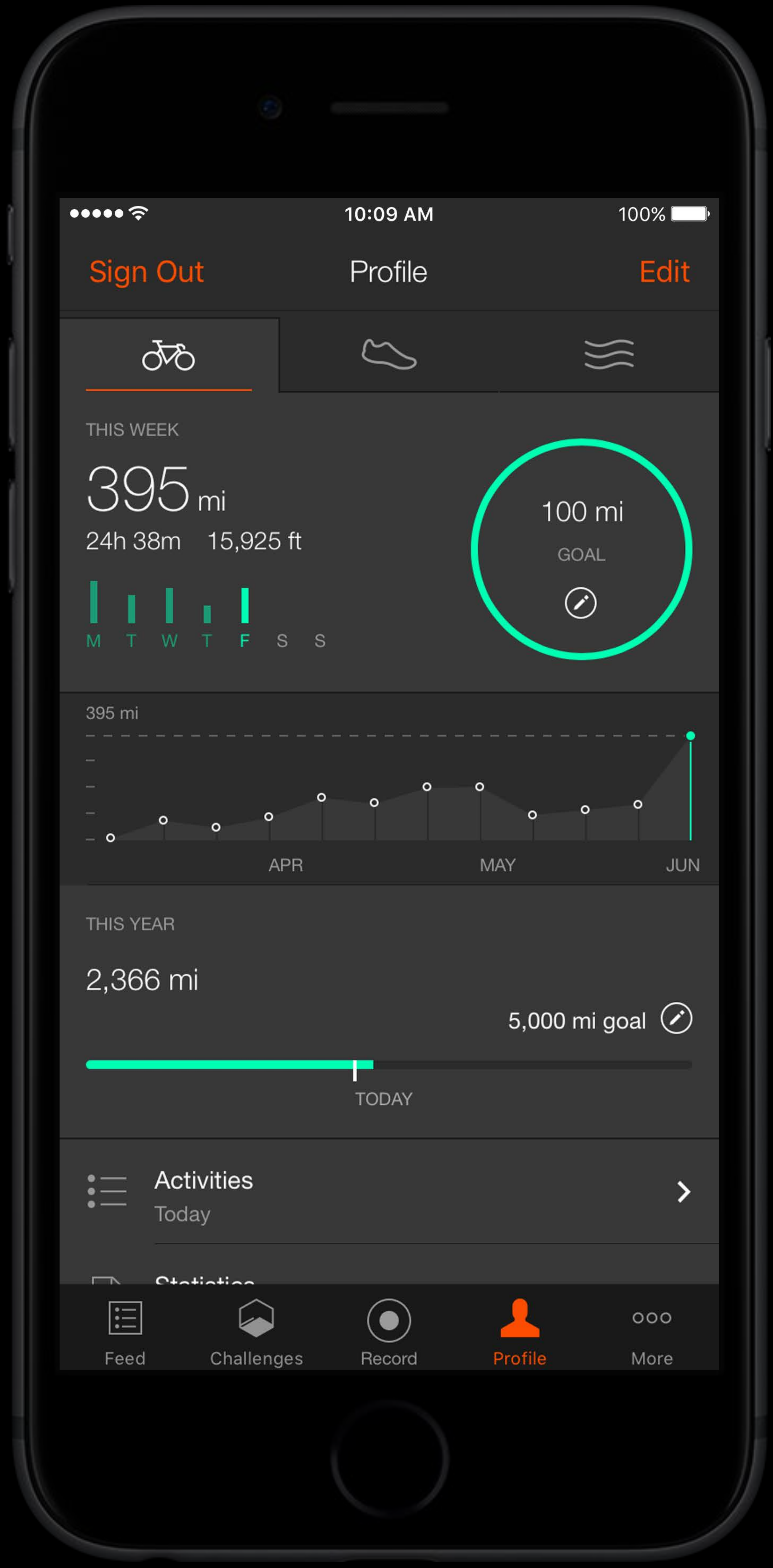
Building Great Workout Apps

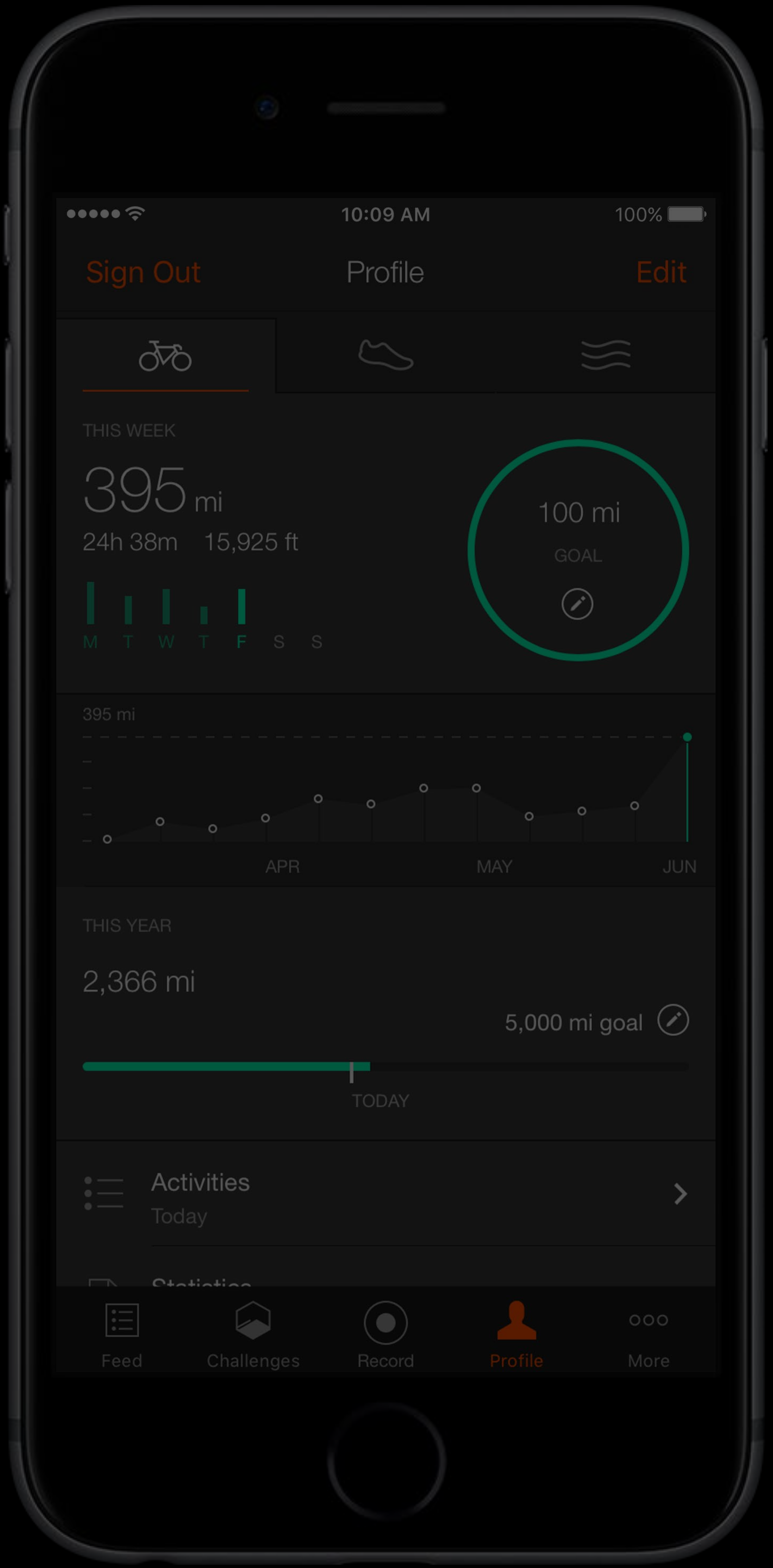
HealthKit APIs and best practices

Session 235

Dash Brittain iOS Software Engineer
Jorge Moriñigo iOS Software Engineer

178
1
17
132
16


●●●●●
10:09 AM
100%
Sign Out
Profile
Edit
THIS WEEK
395 mi
24h 38m 15,925 ft
M T W T F S S
100 mi
GOAL
395 mi
APR
MAY
JUN
THIS YEAR
2,366 mi
5,000 mi goal
TODAY
Activities
Today
Statistics
Feed
Challenges
Record
Profile
More



THIS WEEK

395 mi

24h 38m 15,925 ft

100 mi

GOAL

THIS YEAR

2,366 mi

5,000 mi goal

Activities

Today

Statistics

10:09 AM
100%
Sign Out
Profile
Edit
THIS WEEK
395 mi
24h 38m
15,925 ft
M T W T F S S
100 mi
GOAL
395 mi
APR
MAY
JUN
THIS YEAR
2,366 mi
5,000 mi goal
TODAY
Activities
Today
Statistics
Feed
Challenges
Record
Profile
More
10:09
START

Background Running

Background Running

Workout Lifecycle

Background Running

Workout Lifecycle

Activity Rings

Background Running

Workout Lifecycle

Activity Rings

Workouts in iOS 10

Background Running

Workout Lifecycle

Activity Rings

Workouts in iOS 10

Best Practices

Getting Started

HKWorkoutSession

HKWorkoutSession

Motion and calorimetry for activity type

HKWorkoutSession

Motion and calorimetry for activity type

Activity rings

HKWorkoutSession

Motion and calorimetry for activity type

Activity rings

Show app on wake

HKWorkoutSession

Motion and calorimetry for activity type

Activity rings

Show app on wake

Background running

Background Running

NEW

Background Running

NEW

Process sensor data

Background Running

NEW

Process sensor data

Live feedback

Background Running

NEW

Process sensor data

Live feedback

Quickly update UI

NEW

```
// Background Running
// WatchKit Extension Info.plist

<key>WKBackgroundModes</key>
<array>
   <string>workout-processing</string>
</array>
```

Background Running

Conserve power

NEW

Background Running

Conserve power

NEW

Limit background work

Background Running

Conserve power

NEW

Limit background work

App may be suspended

Background Running

Conserve power

NEW

Limit background work

App may be suspended

Measure background work with tools

Background Running

Conserve power

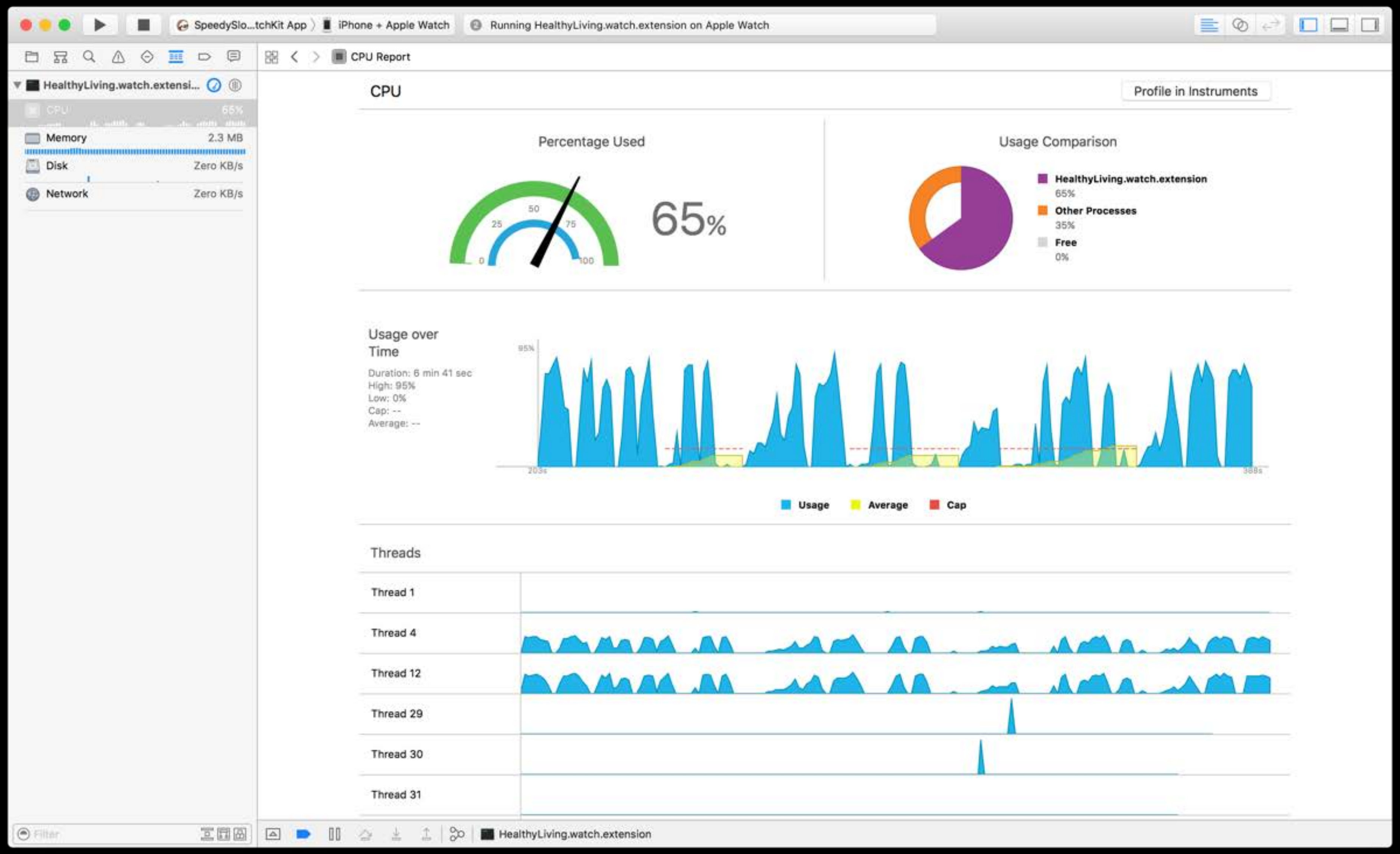
NEW

Limit background work

App may be suspended

Measure background work with tools

- CPU Report in Xcode



Duration: 6 min 41 sec
High: 95%
Low: 0%
Cap: --
Average: --

Background Running

Conserve power

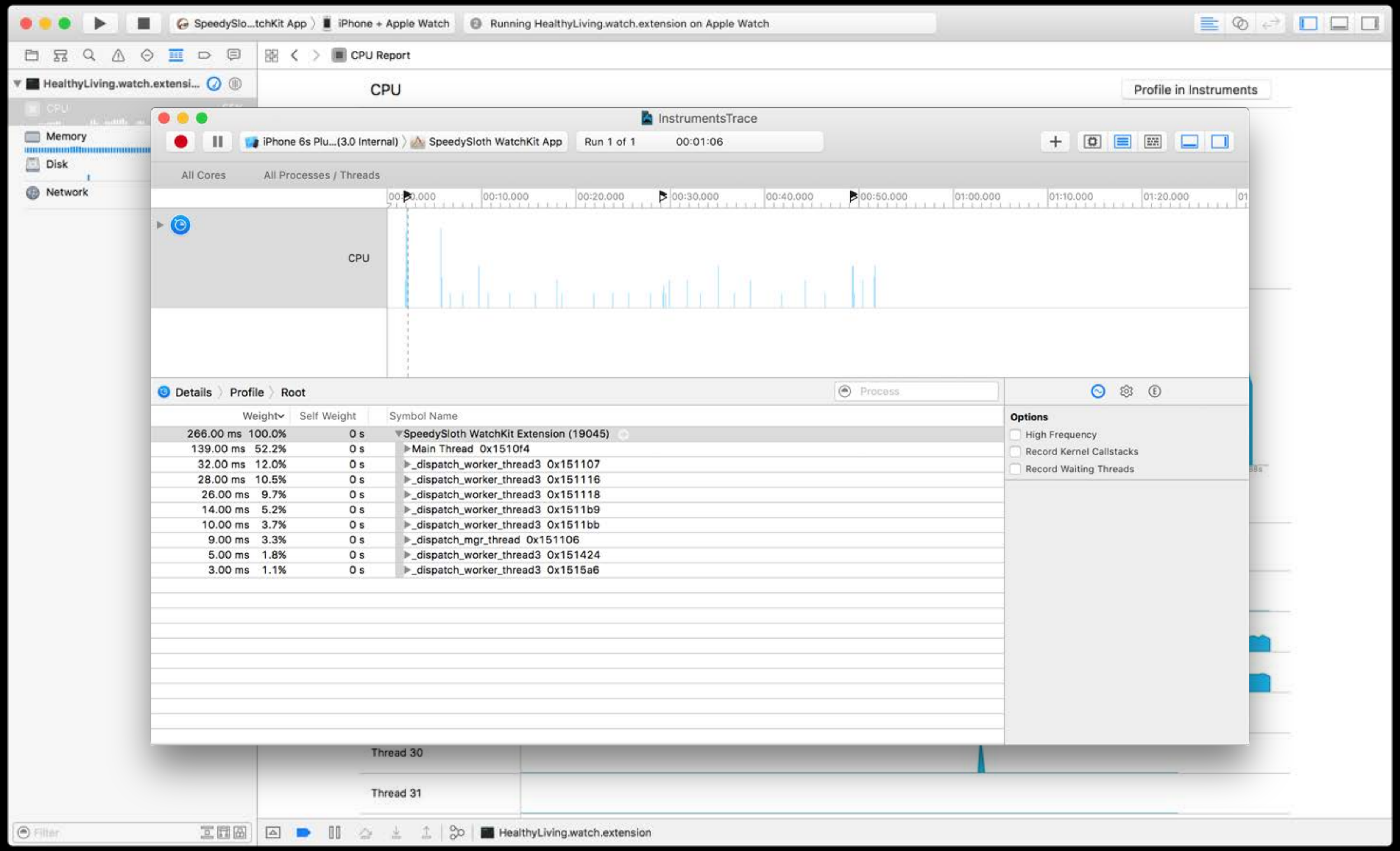
NEW

Limit background work

App may be suspended

Measure background work with tools

- CPU Report in Xcode
- Time Profiler in Instruments



Weight		Self Weight	Symbol Name
266.00 ms	100.0%	0 s	▼SpeedySloth WatchKit Extension (19045)
139.00 ms	52.2%	0 s	▶Main Thread 0x1510f4
32.00 ms	12.0%	0 s	▶_dispatch_worker_thread3 0x151107
28.00 ms	10.5%	0 s	▶_dispatch_worker_thread3 0x151116
26.00 ms	9.7%	0 s	▶_dispatch_worker_thread3 0x151118
14.00 ms	5.2%	0 s	▶_dispatch_worker_thread3 0x1511b9
10.00 ms	3.7%	0 s	▶_dispatch_worker_thread3 0x1511bb
9.00 ms	3.3%	0 s	▶_dispatch_mgr_thread 0x151106
5.00 ms	1.8%	0 s	▶_dispatch_worker_thread3 0x151424
3.00 ms	1.1%	0 s	▶_dispatch_worker_thread3 0x1515a6

Options

☐ High Frequency

☐ Record Kernel Callstacks

☐ Record Waiting Threads

Background Running

Conserve power

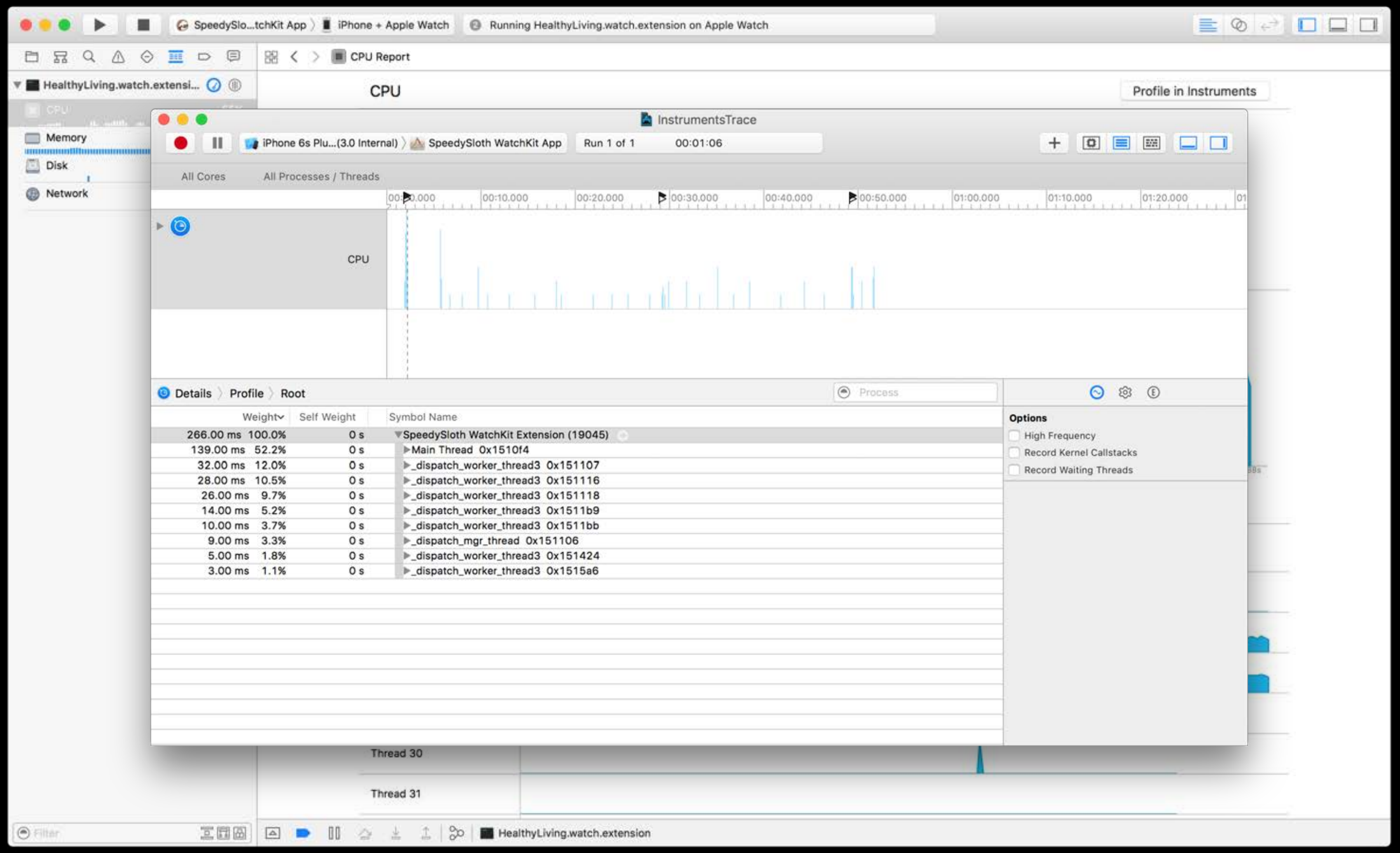
NEW

Limit background work

App may be suspended

Measure background work with tools

- CPU Report in Xcode
- Time Profiler in Instruments
- Backtrace log



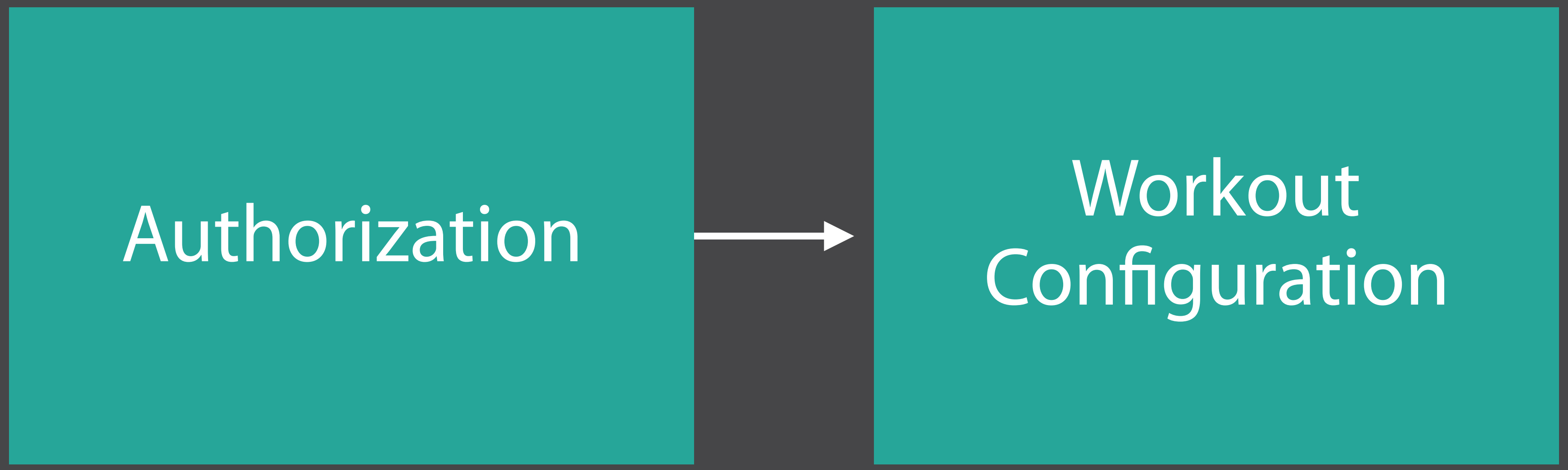
Weight		Self Weight	Symbol Name
266.00 ms	100.0%	0 s	▼SpeedySloth WatchKit Extension (19045)
139.00 ms	52.2%	0 s	▶Main Thread 0x1510f4
32.00 ms	12.0%	0 s	▶_dispatch_worker_thread3 0x151107
28.00 ms	10.5%	0 s	▶_dispatch_worker_thread3 0x151116
26.00 ms	9.7%	0 s	▶_dispatch_worker_thread3 0x151118
14.00 ms	5.2%	0 s	▶_dispatch_worker_thread3 0x1511b9
10.00 ms	3.7%	0 s	▶_dispatch_worker_thread3 0x1511bb
9.00 ms	3.3%	0 s	▶_dispatch_mgr_thread 0x151106
5.00 ms	1.8%	0 s	▶_dispatch_worker_thread3 0x151424
3.00 ms	1.1%	0 s	▶_dispatch_worker_thread3 0x1515a6

Starting a Workout

Starting a Workout

Authorization

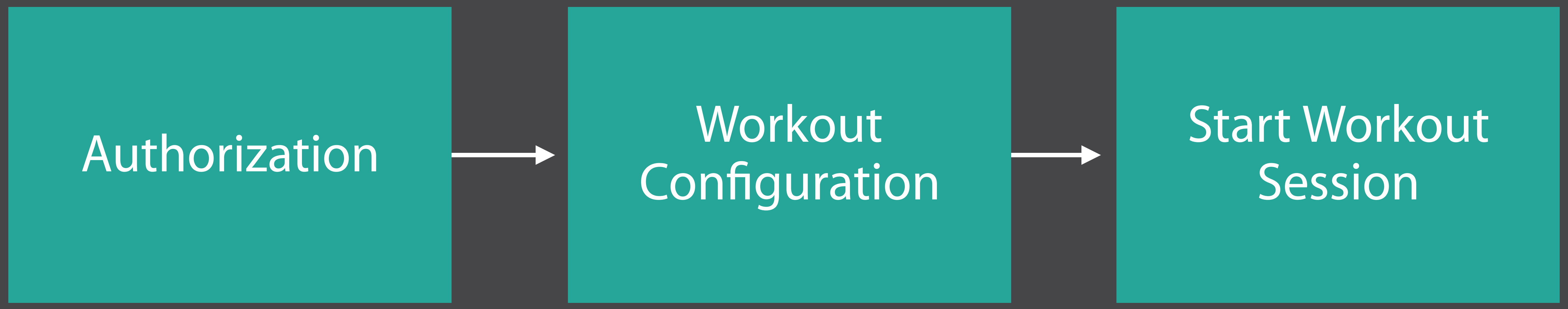
Starting a Workout



Authorization

Workout Configuration

Starting a Workout



Authorization

Authorization

Authorization to write

- Workouts

Authorization

Authorization to write

- Workouts

Authorization to read

- Energy burned
- Distance
- Heart rate

Authorization

Authorization to write

- Workouts

Authorization to read

- Energy burned
- Distance
- Heart rate

Getting the Most Out of HealthKit Wednesday

```swift
// Workout Configuration

var workoutConfiguration = HKWorkoutConfiguration()
workoutConfiguration.activityType = .running
workoutConfiguration.locationType = .outdoor
```

```
// Workout Configuration

var workoutConfiguration = HKWorkoutConfiguration()
workoutConfiguration.activityType = .running
workoutConfiguration.locationType = .outdoor
```

```swift
// Workout Configuration

var workoutConfiguration = HKWorkoutConfiguration()
workoutConfiguration.activityType = .running
workoutConfiguration.locationType = .outdoor
```

```swift
// Start Workout Session

let workoutSession = HKWorkoutSession(configuration: workoutConfiguration)
healthStore.start(workoutSession)
```

```swift
// Start Workout Session

let workoutSession = HKWorkoutSession(configuration: workoutConfiguration)
healthStore.start(workoutSession)
```

```swift
// Start Workout Session

let workoutSession = HKWorkoutSession(configuration: workoutConfiguration)
healthStore.start(workoutSession)
```

Demo

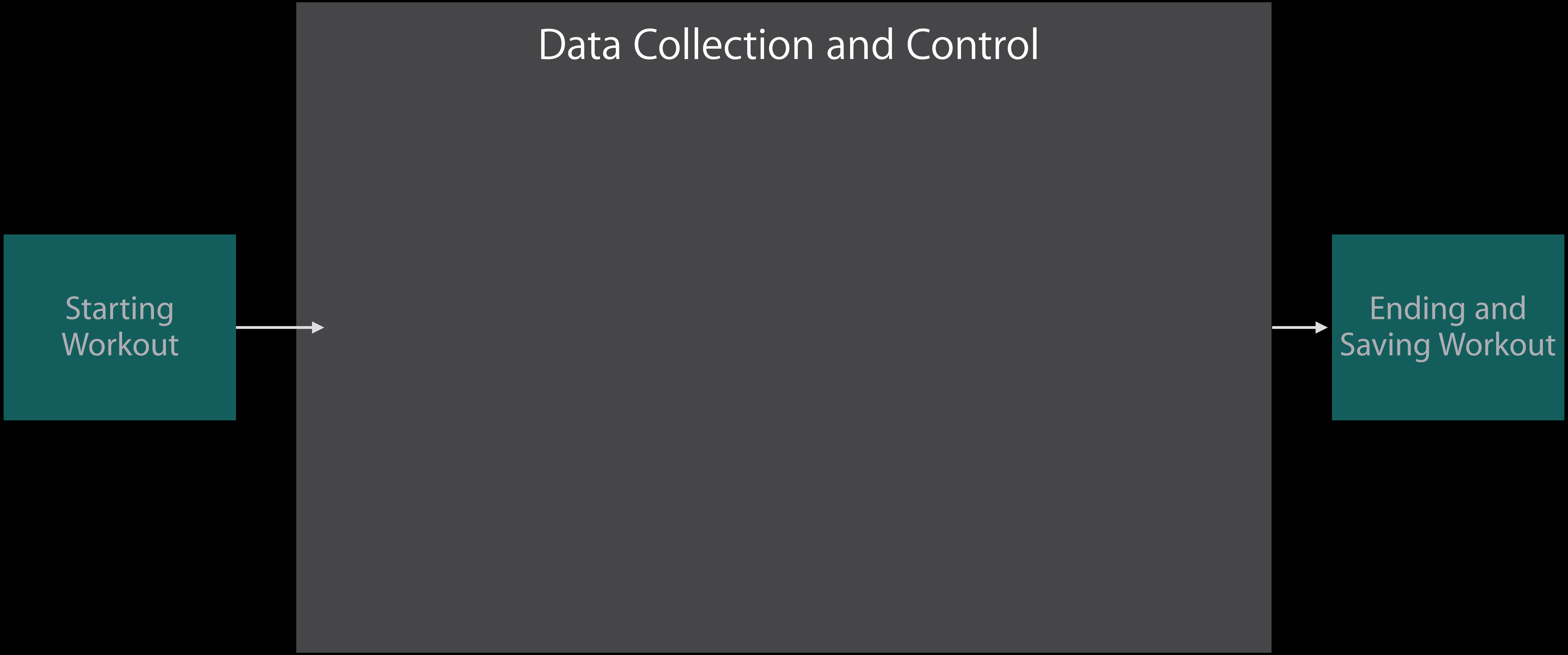
SpeedySloth: Starting a Workout

Dash Brittain

iOS Software Engineer

Data Collection and Control

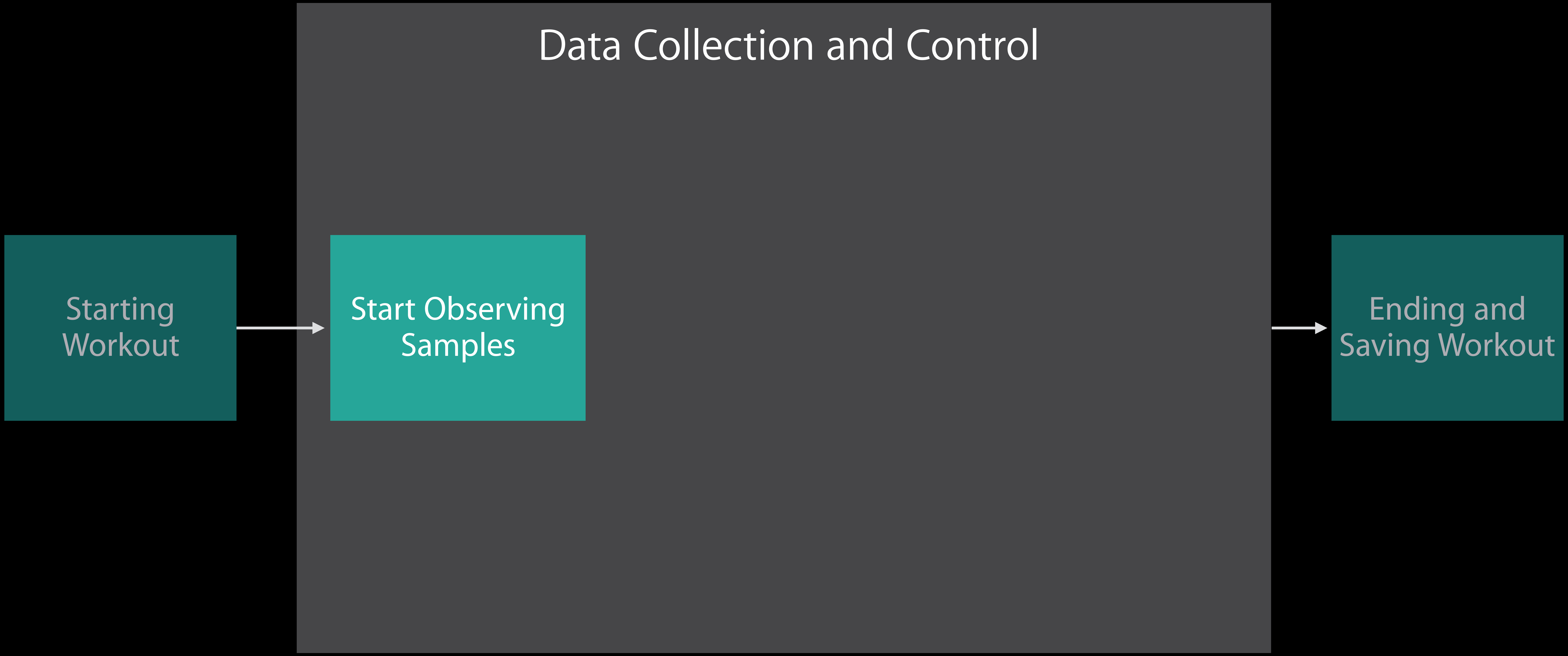
Jorge Moriñigo
iOS Software Engineer



Data Collection and Control

Starting Workout

Ending and Saving Workout

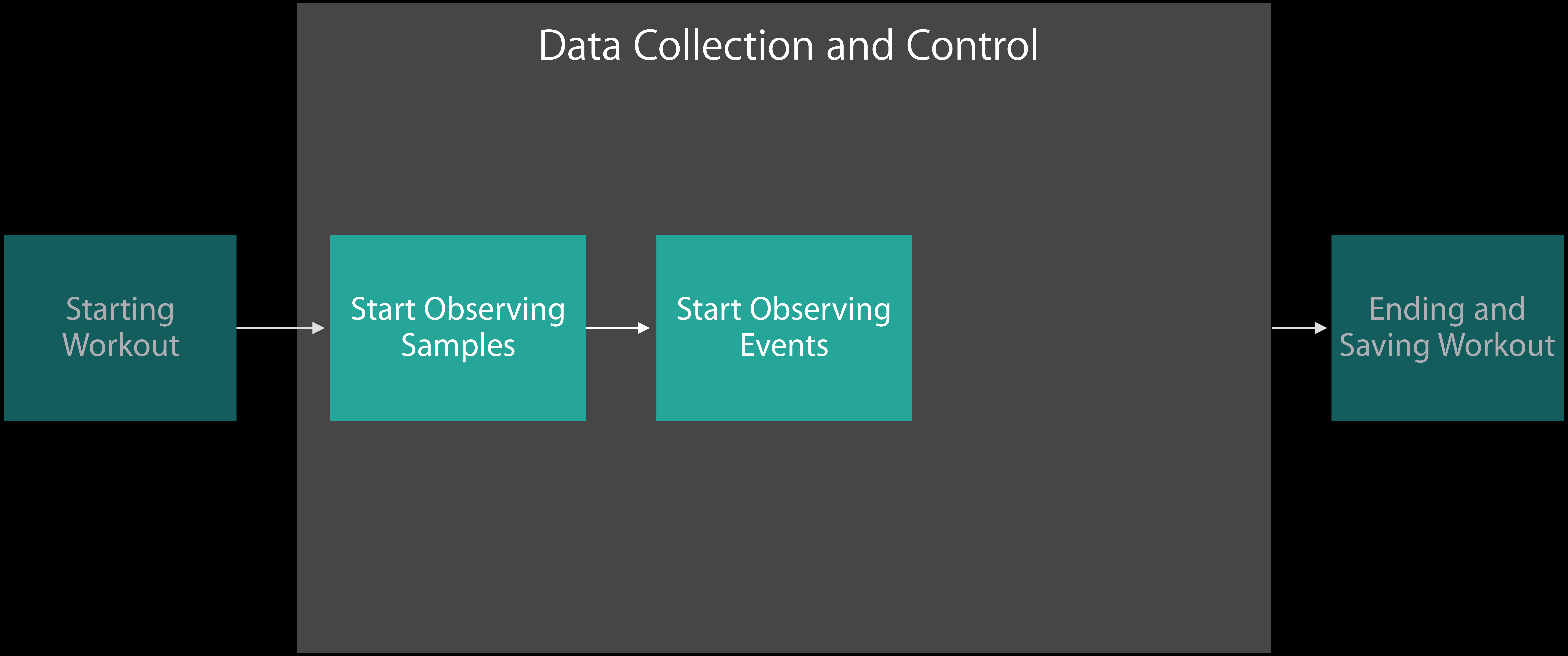


Data Collection and Control

Starting Workout

Start Observing Samples

Ending and Saving Workout



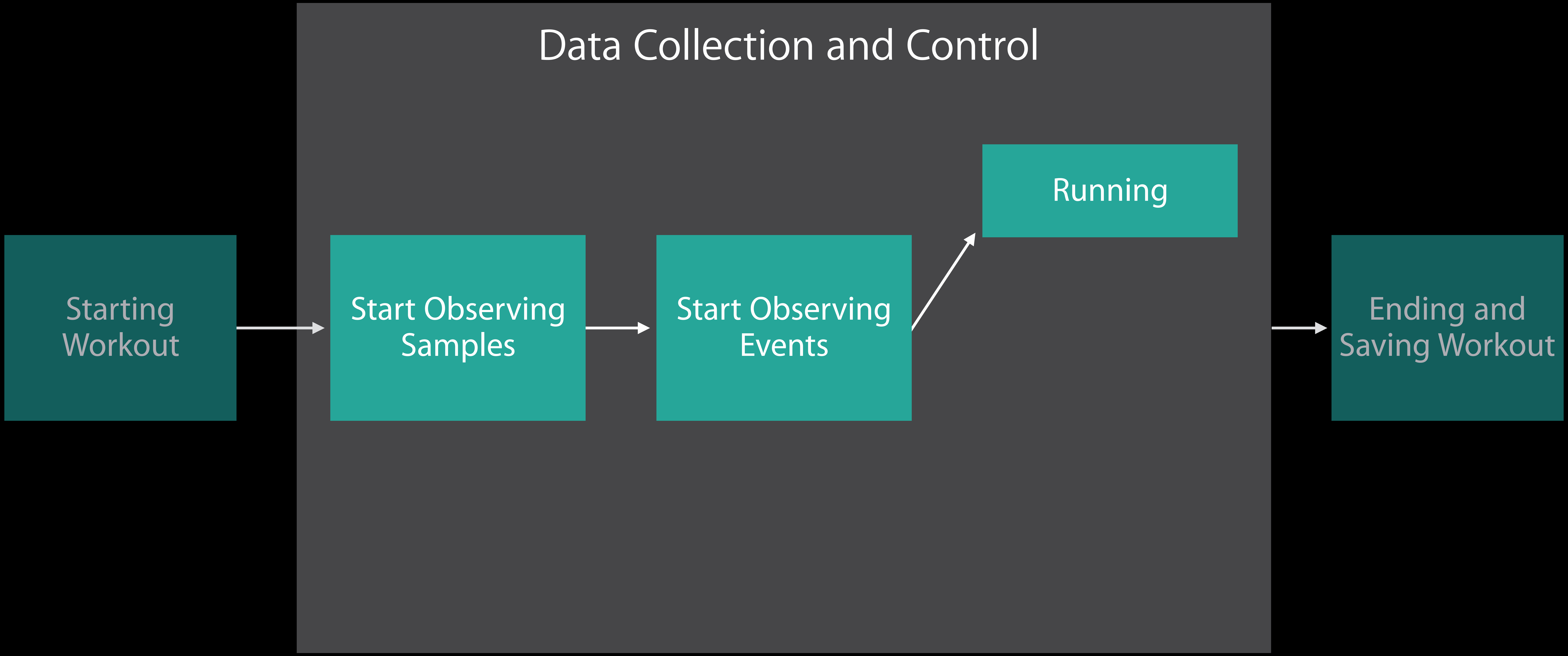
Data Collection and Control

Starting Workout

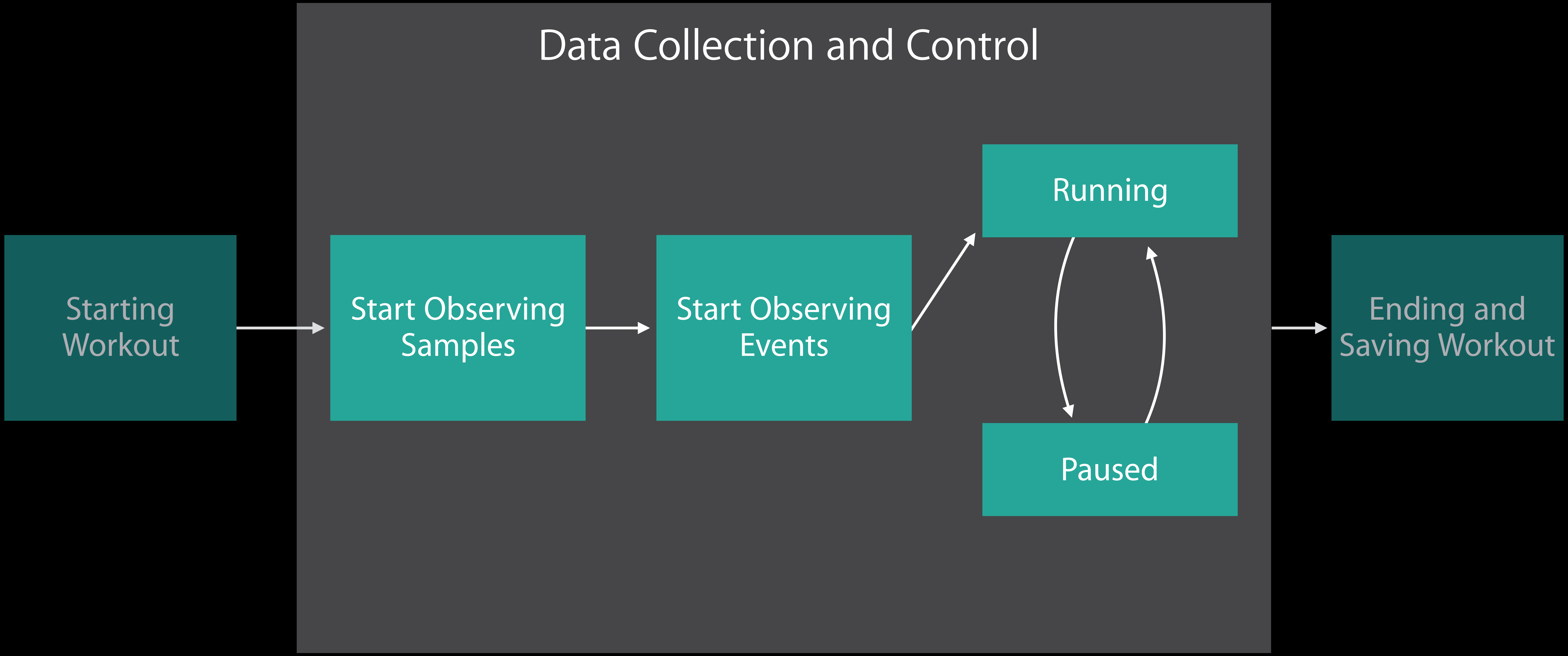
Start Observing Samples

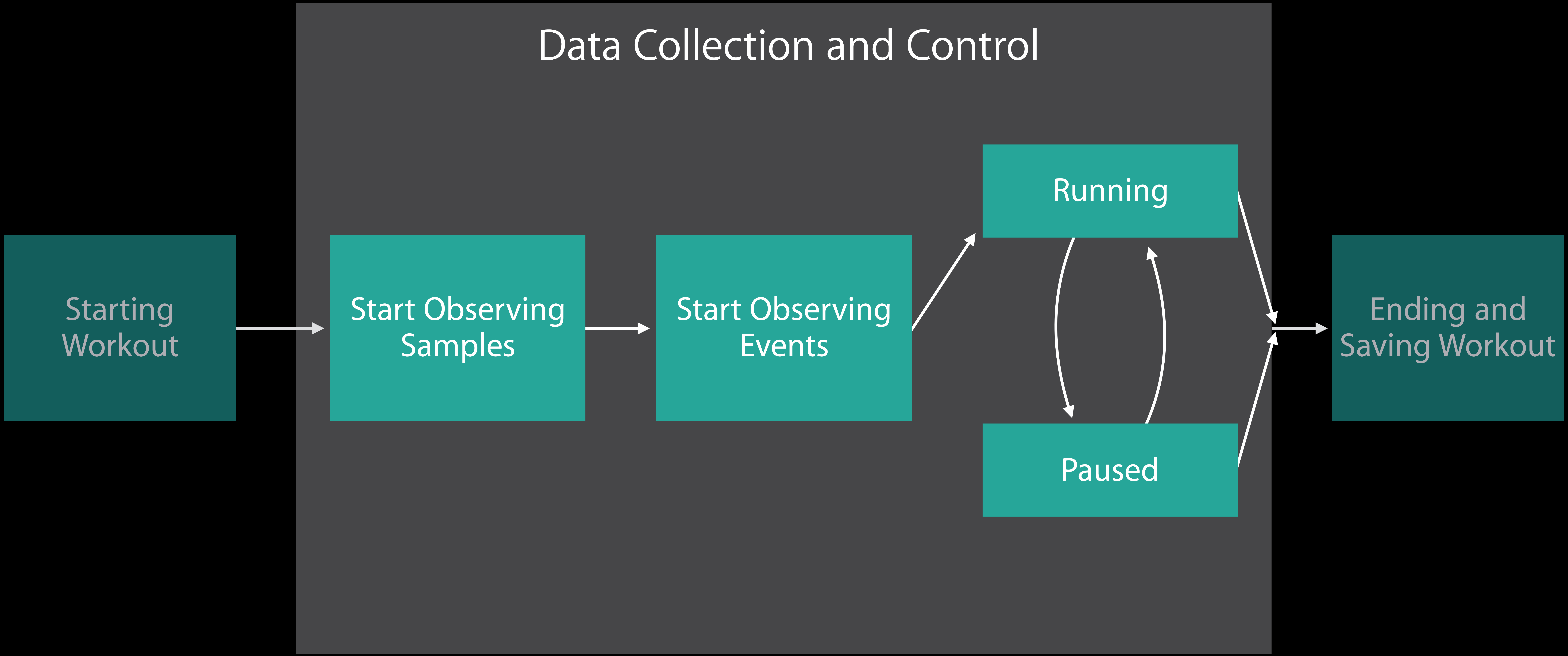
Start Observing Events

Ending and Saving Workout



Data Collection and Control
Starting Workout
Start Observing Samples
Start Observing Events
Running
Ending and Saving Workout


Data Collection and Control
Starting Workout
Start Observing Samples
Start Observing Events
Running
Paused
Ending and Saving Workout



Data Collection and Control

Starting Workout

Start Observing Samples

Start Observing Events

Running

Paused

Ending and Saving Workout

Observing Samples

Observing Samples

Open query for each data type

Observing Samples

Open query for each data type

Update your running totals

Observing Samples

Open query for each data type

Update your running totals

Display live data

Observing Samples

Open query for each data type

Update your running totals

Display live data

Notify user of reached goals

// Observing Samples

```swift
// Observing Samples
let datePredicate = HKQuery.predicateForSamples(withStart: workoutStartDate, end: nil,
                                                options: .strictStartDate)
```

```swift
// Observing Samples
let datePredicate = HKQuery.predicateForSamples(withStart: workoutStartDate, end: nil,
                                                options: .strictStartDate)
let devicePredicate = HKQuery.predicateForObjects(from: [HKDevice.local()])
```

```swift
// Observing Samples
let datePredicate = HKQuery.predicateForSamples(withStart: workoutStartDate, end: nil,
                                                options: .strictStartDate)
let devicePredicate = HKQuery.predicateForObjects(from: [HKDevice.local()])
let queryPredicate = CompoundPredicate(andPredicateWithSubpredicates:[datePredicate,
                                                                      devicePredicate])
```

```swift
// Observing Samples
let datePredicate = HKQuery.predicateForSamples(withStart: workoutStartDate, end: nil,
                                                options: .strictStartDate)
let devicePredicate = HKQuery.predicateForObjects(from: [HKDevice.local()])
let queryPredicate = CompoundPredicate(andPredicateWithSubpredicates:[datePredicate,
                                                                      devicePredicate])

let updateHandler: (HKAnchoredObjectQuery, [HKSample]?, [HKDeletedObject]?, HKQueryAnchor?,
   NSError?) -> Void = { query, samples, deletedObjects, queryAnchor, error in
   // Process samples
}
```

```swift
// Observing Samples
let datePredicate = HKQuery.predicateForSamples(withStart: workoutStartDate, end: nil,
                                                options: .strictStartDate)
let devicePredicate = HKQuery.predicateForObjects(from: [HKDevice.local()])
let queryPredicate = CompoundPredicate(andPredicateWithSubpredicates:[datePredicate,
                                                                      devicePredicate])

let updateHandler: (HKAnchoredObjectQuery, [HKSample]?, [HKDeletedObject]?, HKQueryAnchor?,
   NSError?) -> Void = { query, samples, deletedObjects, queryAnchor, error in
   // Process samples
}
let query = HKAnchoredObjectQuery(type: quantityType,
                                  predicate: queryPredicate,
                                  anchor: nil,
                                  limit: HKObjectQueryNoLimit,
                                  resultsHandler: updateHandler)
```

```
// Observing Samples
let datePredicate = HKQuery.predicateForSamples(withStart: workoutStartDate, end: nil,
                                                options: .strictStartDate)
let devicePredicate = HKQuery.predicateForObjects(from: [HKDevice.local()])
let queryPredicate = CompoundPredicate(andPredicateWithSubpredicates:[datePredicate,
                                                                      devicePredicate])

let updateHandler: (HKAnchoredObjectQuery, [HKSample]?, [HKDeletedObject]?, HKQueryAnchor?,
   NSError?) -> Void = { query, samples, deletedObjects, queryAnchor, error in
   // Process samples
}
let query = HKAnchoredObjectQuery(type: quantityType,
                                  predicate: queryPredicate,
                                  anchor: nil,
                                  limit: HKObjectQueryNoLimit,
                                  resultsHandler: updateHandler)
query.updateHandler = updateHandler
```

```
// Observing Samples
let datePredicate = HKQuery.predicateForSamples(withStart: workoutStartDate, end: nil,
                                                options: .strictStartDate)
let devicePredicate = HKQuery.predicateForObjects(from: [HKDevice.local()])
let queryPredicate = CompoundPredicate(andPredicateWithSubpredicates:[datePredicate,
                                                                      devicePredicate])

let updateHandler: (HKAnchoredObjectQuery, [HKSample]?, [HKDeletedObject]?, HKQueryAnchor?,
   NSError?) -> Void = { query, samples, deletedObjects, queryAnchor, error in
   // Process samples
}
let query = HKAnchoredObjectQuery(type: quantityType,
                                  predicate: queryPredicate,
                                  anchor: nil,
                                  limit: HKObjectQueryNoLimit,
                                  resultsHandler: updateHandler)
query.updateHandler = updateHandler

healthStore.execute(query)
```

Observing Samples

Notify user of reached goal

Observing Samples

Notify user of reached goal

Play haptic to alert user

Observing Samples

Notify user of reached goal

Play haptic to alert user

Update UI to reflect reached goal

Observing Samples

Notify user of reached goal

Play haptic to alert user

Update UI to reflect reached goal

```
WKInterfaceDevice.current().play(.notification)
```

Observing Events

Observing Events

Timestamps to highlight point in session

Observing Events

Timestamps to highlight point in session

Some events created by your app

Observing Events

Timestamps to highlight point in session

Some events created by your app

Some events created by HealthKit

Observing Events

Timestamps to highlight point in session

Some events created by your app

Some events created by HealthKit

```
protocol HKWorkoutSessionDelegate
   func workoutSession(_ workoutSession: HKWorkoutSession, didGenerate event: HKWorkoutEvent)
```

Laps and Markers

NEW

Laps and Markers

NEW

```
HKWorkoutEventType.lap
HKWorkoutEventType.marker
```

Laps and Markers

NEW

Create to store in a workout

```
HKWorkoutEventType.lap
HKWorkoutEventType.marker
```

Laps and Markers

NEW

Create to store in a workout

Timestamps for graphs and statistics

```
HKWorkoutEventType.lap
HKWorkoutEventType.marker
```

Laps and Markers

NEW

Create to store in a workout

Timestamps for graphs and statistics

Laps for equal distance partitions

```
HKWorkoutEventType.lap
HKWorkoutEventType.marker
```

Laps and Markers

NEW

Create to store in a workout

Timestamps for graphs and statistics

Laps for equal distance partitions

Markers for arbitrary points

```
HKWorkoutEventType.lap
HKWorkoutEventType.marker
```

Laps and Markers

NEW

Create to store in a workout

Timestamps for graphs and statistics

Laps for equal distance partitions

Markers for arbitrary points

```
HKWorkoutEventType.lap
HKWorkoutEventType.marker
```

Pausing and Resuming

Pausing and Resuming

Users stop their activity

Pausing and Resuming

Users stop their activity

Pause workouts to save power and space

Pausing and Resuming

Users stop their activity

Pause workouts to save power and space

Ignore data while paused

Pausing and Resuming

Users stop their activity

Pause workouts to save power and space

Ignore data while paused

HealthKit responds with pause/resume events

Pausing and Resuming

Users stop their activity

Pause workouts to save power and space

Ignore data while paused

HealthKit responds with pause/resume events

After pause event, you will stop receiving new events

Pausing and Resuming

NEW

```
class HKHealthStore {
    func pause(_ workoutSession: HKWorkoutSession)
    func resume(_ workoutSession: HKWorkoutSession)
    ...
}
```

Pausing and Resuming

NEW

```
class HKHealthStore {
    func pause(_ workoutSession: HKWorkoutSession)
    func resume(_ workoutSession: HKWorkoutSession)
    ...
}
```

```
HKWorkoutEventType.pause
HKWorkoutEventType.resume
```

Motion Events for Running Workouts

NEW

Motion Events for Running Workouts

NEW

```
HKWorkoutEventType.motionPaused
HKWorkoutEventType.motionResumed
```

Motion Events for Running Workouts

NEW

Watch detects when motion pauses or resumes

```
HKWorkoutEventType.motionPaused
HKWorkoutEventType.motionResumed
```

Motion Events for Running Workouts

NEW

Watch detects when motion pauses or resumes

Stop collecting data

```
HKWorkoutEventType.motionPaused
HKWorkoutEventType.motionResumed
```

Motion Events for Running Workouts

NEW

Watch detects when motion pauses or resumes

Stop collecting data

Don’t need to manual pause

```
HKWorkoutEventType.motionPaused
HKWorkoutEventType.motionResumed
```

Motion Events for Running Workouts

NEW

Watch detects when motion pauses or resumes

Stop collecting data

Don’t need to manual pause

Only for `HKWorkoutActivityTypeRunning`

```
HKWorkoutEventType.motionPaused
HKWorkoutEventType.motionResumed
```

Demo

SpeedySloth: Data Collection and Control

Jorge Moriñigo
iOS Software Engineer

Ending and Saving

Dash Brittain
iOS Software Engineer

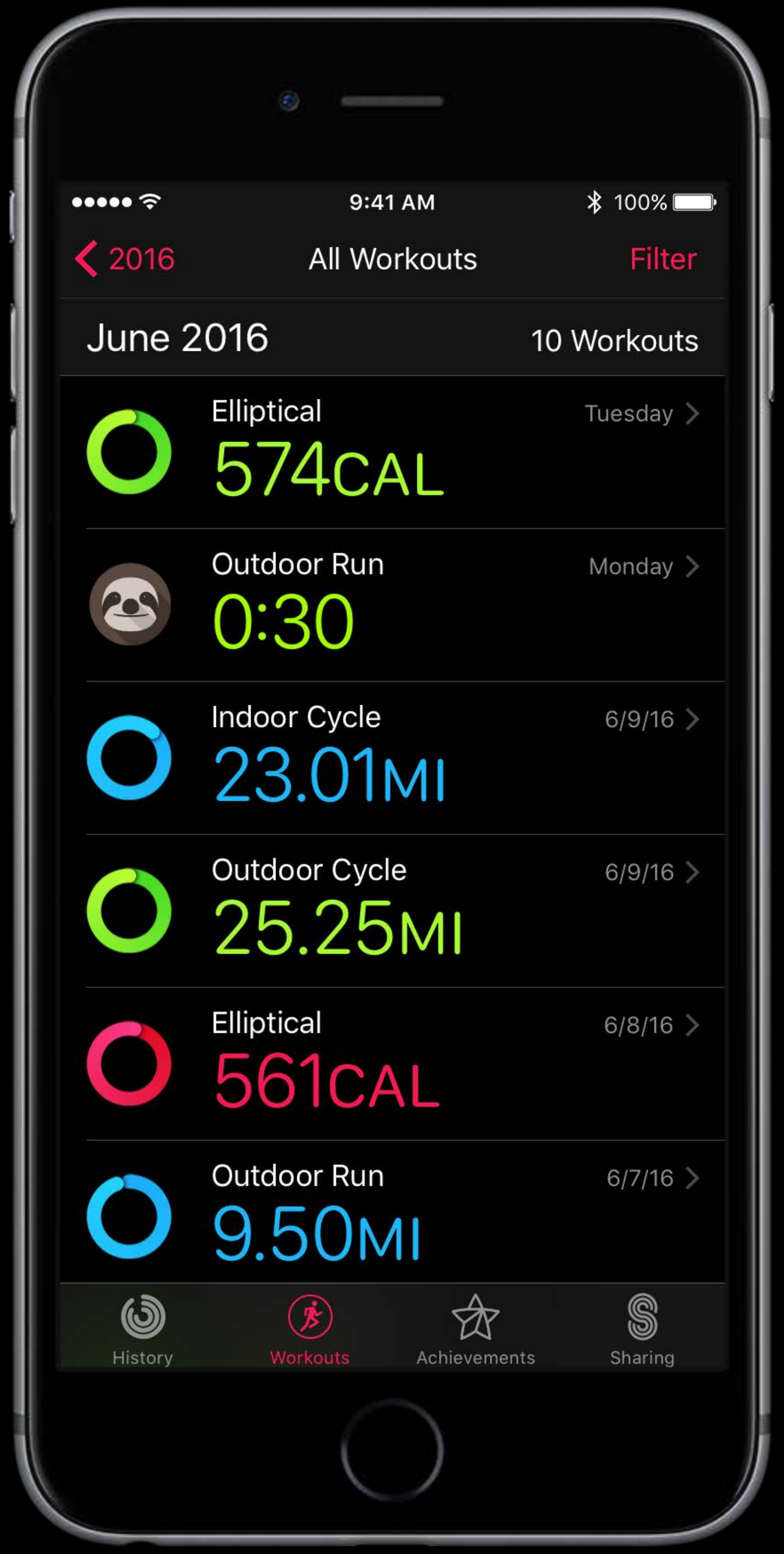
Workouts in Activity

NEW

Workouts in Activity

NEW

Activity workout list



2016 All Workouts Filter

June 2016 10 Workouts

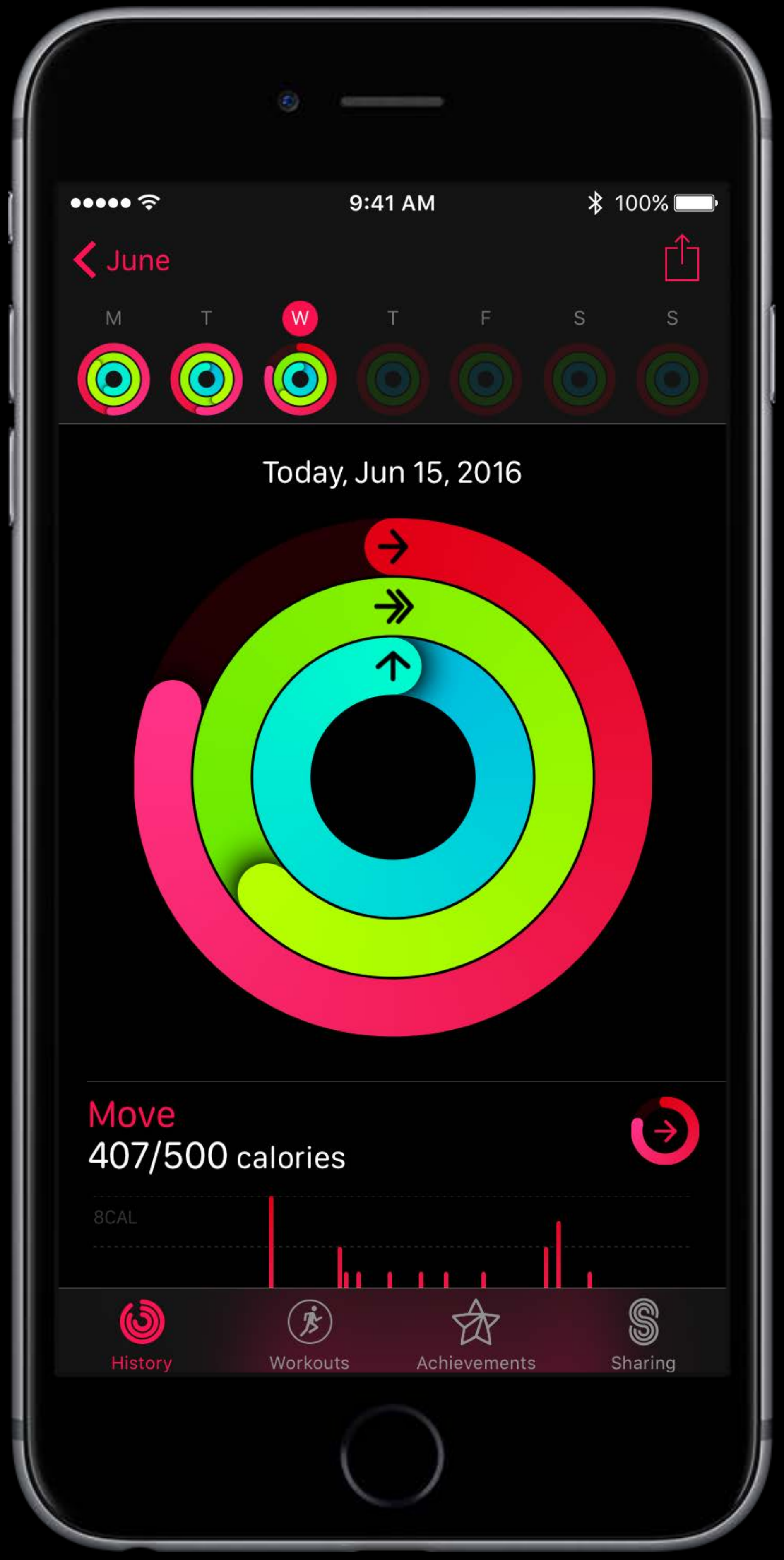
Elliptical 574CAL	Tuesday
Outdoor Run 0:30	Monday
Indoor Cycle 23.01MI	6/9/16
Outdoor Cycle 25.25MI	6/9/16
Elliptical 561CAL	6/8/16
Outdoor Run 9.50MI	6/7/16

Workouts in Activity

NEW

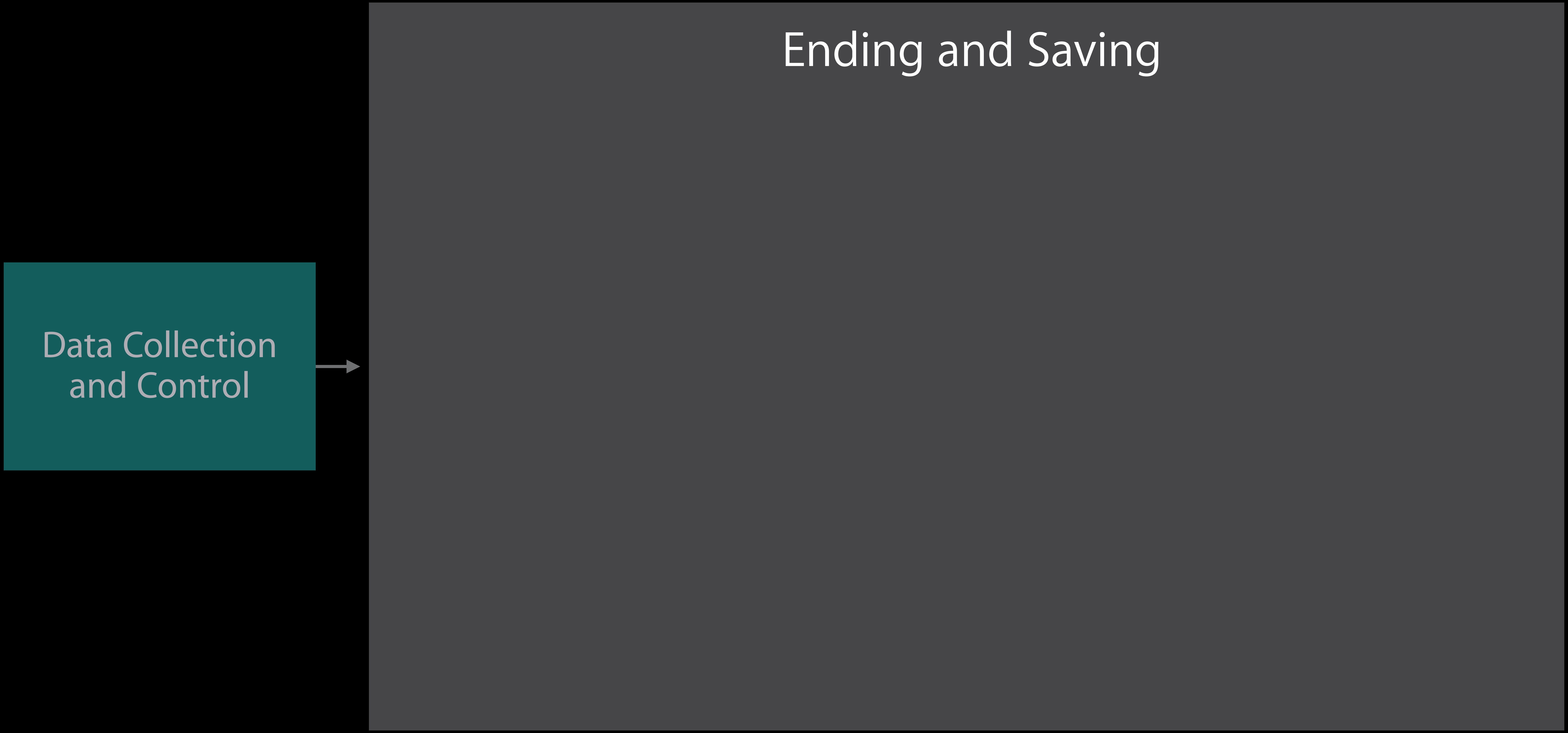
Activity workout list

Activity Move ring



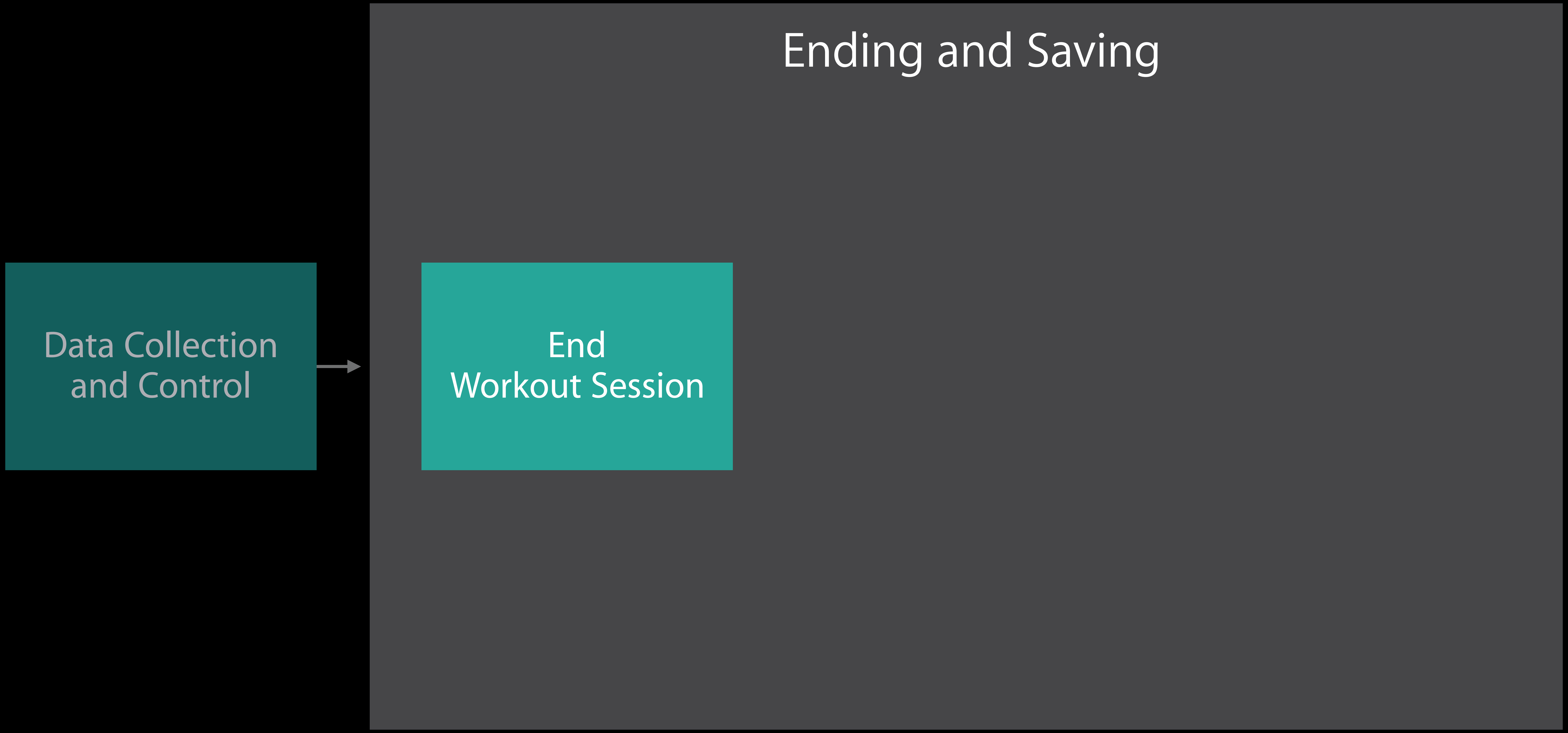
Today, Jun 15, 2016

Move
407/500 calories



Ending and Saving

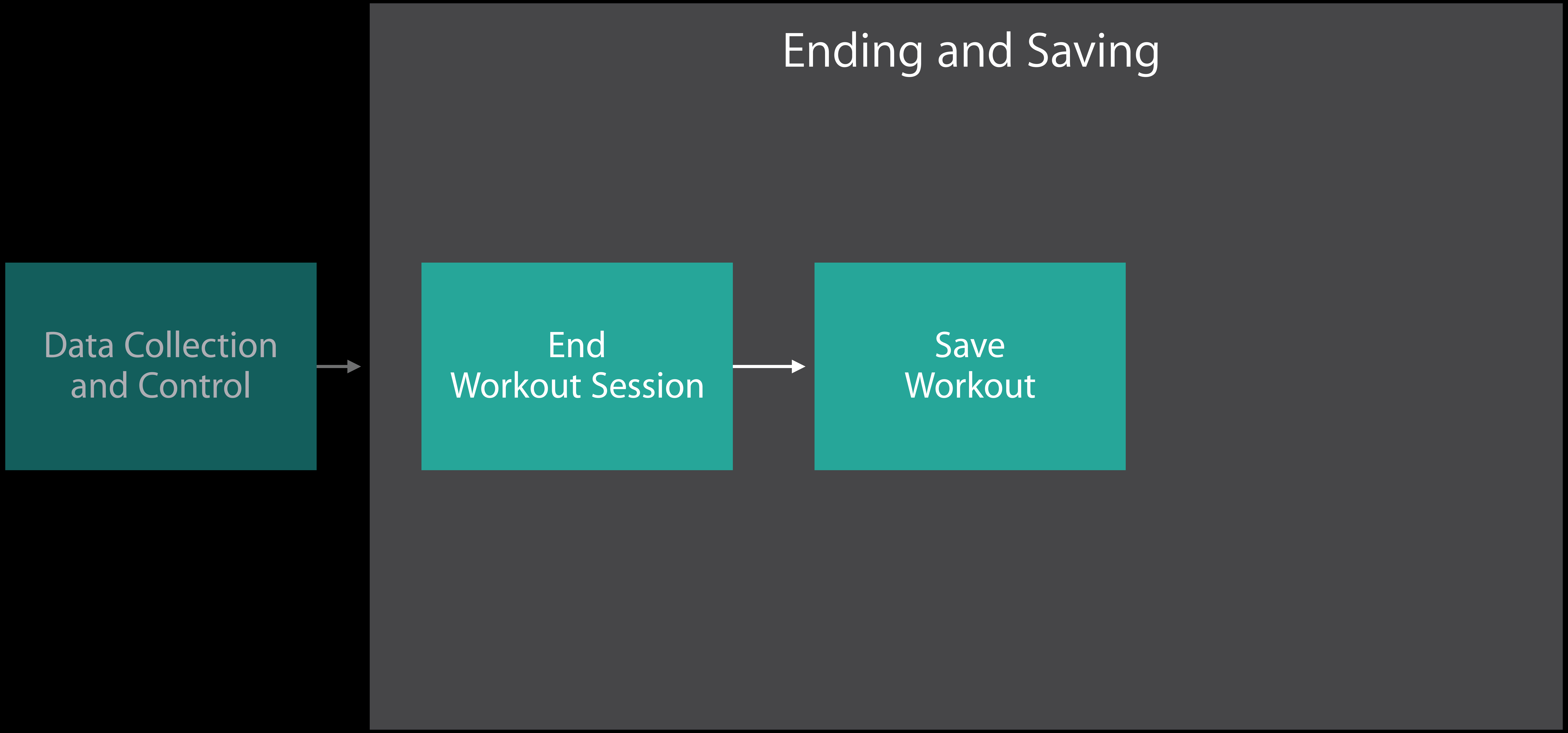
Data Collection and Control

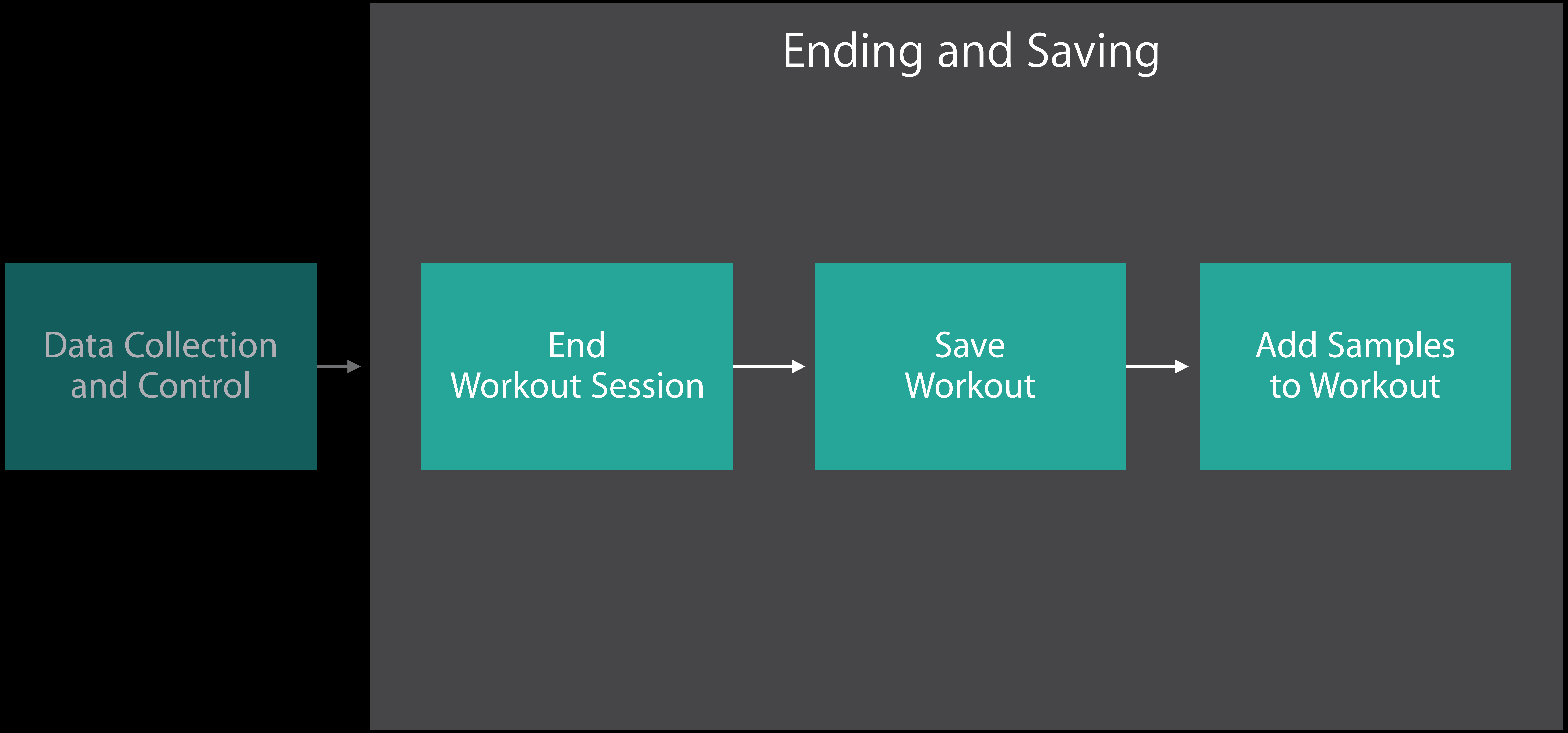


Ending and Saving

Data Collection and Control

End Workout Session


Ending and Saving
Data Collection and Control
End Workout Session
Save Workout



Ending and Saving

Data Collection and Control

End Workout Session

Save Workout

Add Samples to Workout

Ending an HKWorkoutSession

Ending an HKWorkoutSession

```
healthStore.end(workoutSession)
```

Ending an HKWorkoutSession

```
healthStore.end(workoutSession)
```

```
func workoutSession(_ workoutSession: HKWorkoutSession,
                    didChangeTo toState: HKWorkoutSessionState,
                    from fromState: HKWorkoutSessionState,
                    date: Date) {
        if toState == .ended {
           // Workout ended, time to save
       }
   }
```

```swift
// Save Workout

let workout = HKWorkout(
// Save Workout

let workout = HKWorkout(activityType: config.activityType,
```

```swift
// Save Workout

let workout = HKWorkout(activityType: config.activityType,
                        start: startDate,
// Save Workout

let workout = HKWorkout(activityType: config.activityType,
                        start: startDate,
                        end: endDate,
```

```swift
// Save Workout

let workout = HKWorkout(activityType: config.activityType,
                        start: startDate,
                        end: endDate,
                        workoutEvents: events,
// Save Workout

let workout = HKWorkout(activityType: config.activityType,
                        start: startDate,
                        end: endDate,
                        workoutEvents: events,
                        totalEnergyBurned: totalEnergyBurned,
// Save Workout

let workout = HKWorkout(activityType: config.activityType,
                        start: startDate,
                        end: endDate,
                        workoutEvents: events,
                        totalEnergyBurned: totalEnergyBurned,
                        totalDistance: totalDistance,
// Save Workout

let workout = HKWorkout(activityType: config.activityType,
                        start: startDate,
                        end: endDate,
                        workoutEvents: events,
                        totalEnergyBurned: totalEnergyBurned,
                        totalDistance: totalDistance,
                        metadata: [HKMetadataKeyIndoorWorkout: isIndoor])
```

```
// Save Workout

let workout = HKWorkout(activityType: config.activityType,
                        start: startDate,
                        end: endDate,
                        workoutEvents: events,
                        totalEnergyBurned: totalEnergyBurned,
                        totalDistance: totalDistance,
                        metadata: [HKMetadataKeyIndoorWorkout: isIndoor])

healthStore.save(workout) { (success, error) in
    // Workout saved
}
```

Add Samples to Workout

Add Samples to Workout

Creates an association

Add Samples to Workout

Creates an association

Query later for graphs

Add Samples to Workout

Creates an association

Query later for graphs

Activity Move ring credit

Add Samples to Workout

Creates an association

Query later for graphs

Activity Move ring credit

Match totals on HKWorkout

```swift
// Add Samples to Workout

healthStore.add(samples, to: workout) { (success, error) in
   // Samples added
}
```

Demo

SpeedySloth: Ending and Saving

Dash Brittain
iOS Software Engineer

Workouts in iOS 10

Jorge Moriñigo
iOS Software Engineer

Workouts in iOS 10

Workouts in iOS 10

Workout apps have parent iPhone application

Workouts in iOS 10

Workout apps have parent iPhone application

WatchConnectivity for messaging when Apple Watch app is running

Workouts in iOS 10

Workout apps have parent iPhone application

WatchConnectivity for messaging when Apple Watch app is running

Background running

Workouts in iOS 10

Workout apps have parent iPhone application

WatchConnectivity for messaging when Apple Watch app is running

Background running

Start a workout from iPhone

Workouts in iOS 10

Workout apps have parent iPhone application

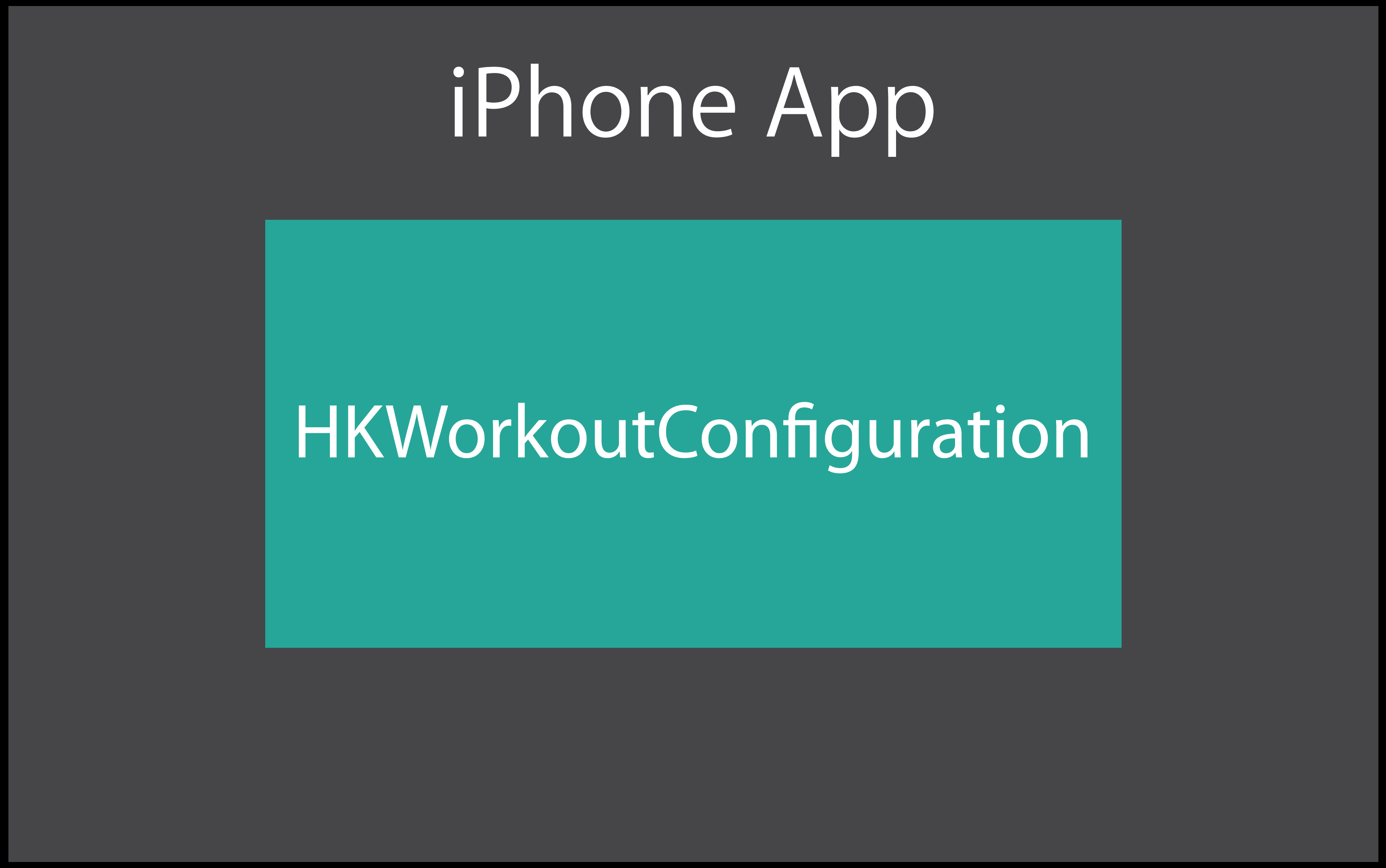
WatchConnectivity for messaging when Apple Watch app is running

Background running

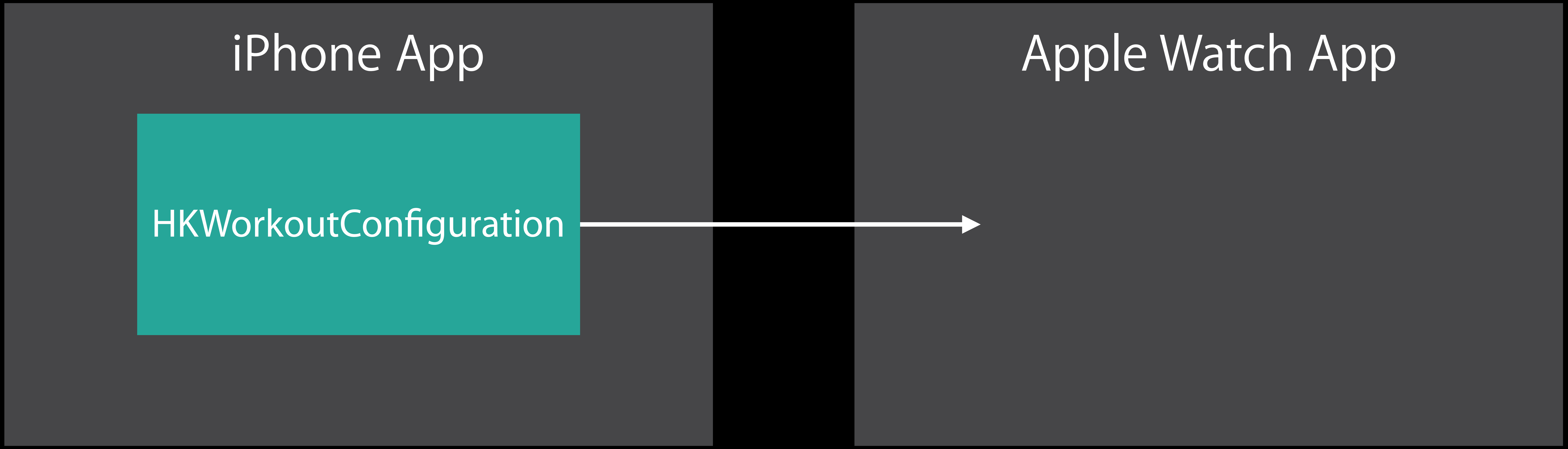
Start a workout from iPhone

No user intervention on Apple Watch

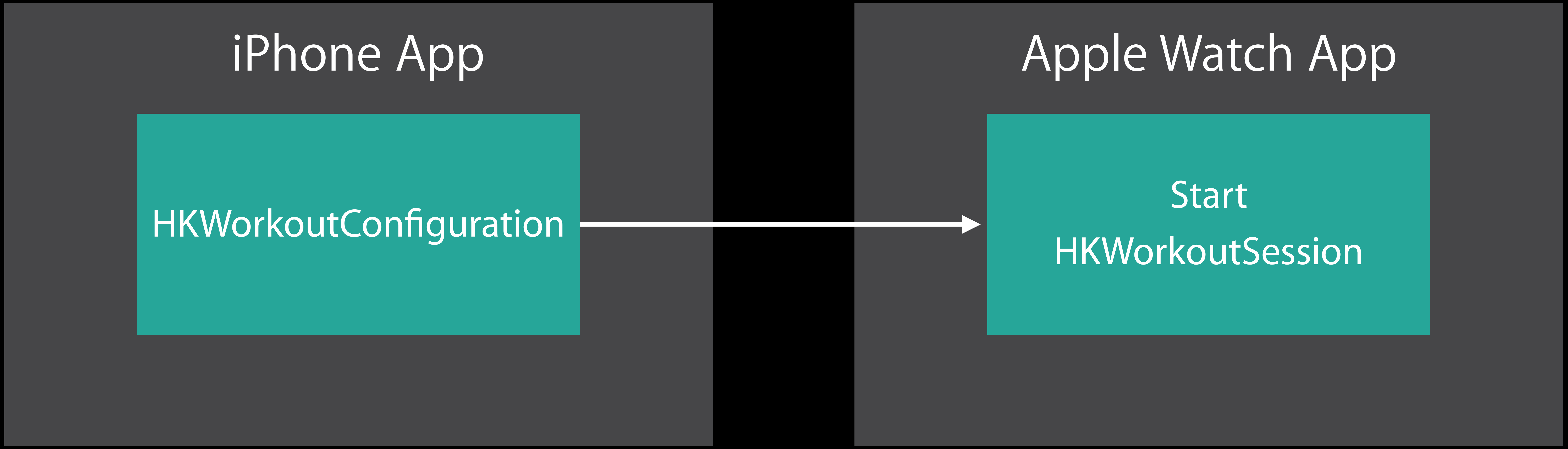
Workouts in iOS 10



Workouts in iOS 10



Workouts in iOS 10



iPhone App

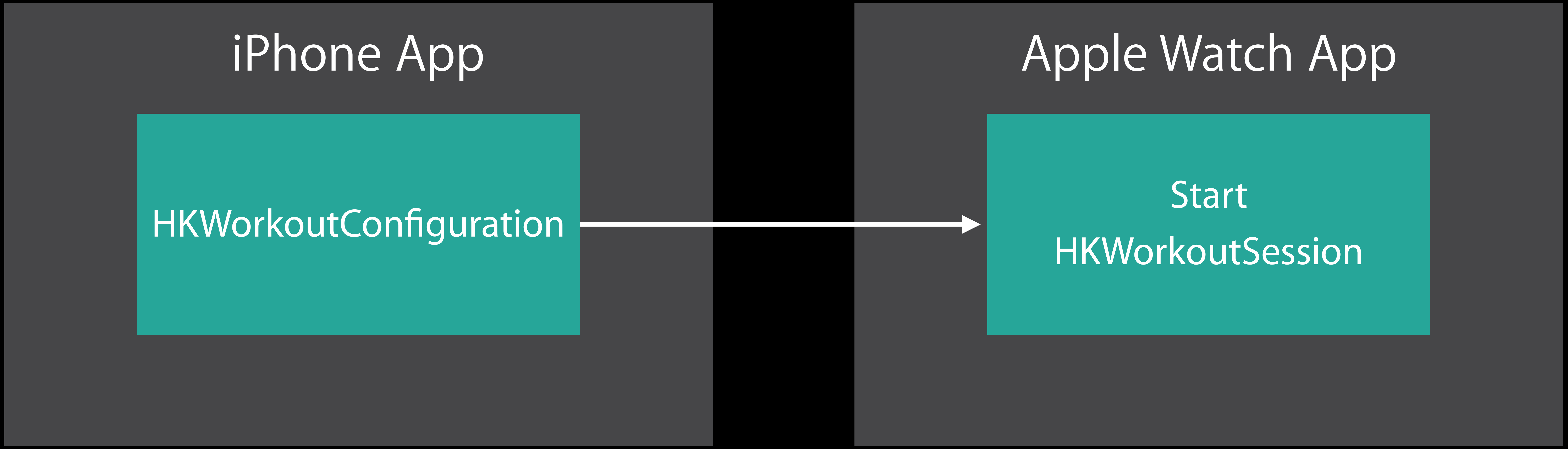
HKWorkoutConfiguration

Apple Watch App

Start

HKWorkoutSession

Workouts in iOS 10



iPhone App

HKWorkoutConfiguration

Apple Watch App

Start

HKWorkoutSession

```
<key>UIBackgroundModes</key>
<array>
   <string>workout-processing</string>
</array>
```

```swift
// Starting Workout from iPhone

var workoutConfiguration = HKWorkoutConfiguration()
workoutConfiguration.activityType = .running
workoutConfiguration.locationType = .outdoor

let workoutSession = HKWorkoutSession(configuration: workoutConfiguration)
workoutSession.delegate = self
healthStore.start(workoutSession)
```

```swift
// Starting Workout from iPhone

var workoutConfiguration = HKWorkoutConfiguration()
workoutConfiguration.activityType = .running
workoutConfiguration.locationType = .outdoor

let workoutSession = HKWorkoutSession(configuration: workoutConfiguration)
workoutSession.delegate = self
healthStore.start(workoutSession)
```

```swift
// Starting Workout from iPhone

var workoutConfiguration = HKWorkoutConfiguration()
workoutConfiguration.activityType = .running
workoutConfiguration.locationType = .outdoor

let workoutSession = HKWorkoutSession(configuration: workoutConfiguration)
workoutSession.delegate = self
healthStore.start(workoutSession)
```

```swift
// Starting Workout from iPhone

var workoutConfiguration = HKWorkoutConfiguration()
workoutConfiguration.activityType = .running
workoutConfiguration.locationType = .outdoor

let workoutSession = HKWorkoutSession(configuration: workoutConfiguration)
workoutSession.delegate = self
healthStore.start(workoutSession)
```

```swift
// Starting Workout from iPhone

// iPhone App
var workoutConfiguration = HKWorkoutConfiguration()
workoutConfiguration.activityType = .running
workoutConfiguration.locationType = .outdoor

// Apple Watch App
let workoutSession = HKWorkoutSession(configuration: workoutConfiguration)
workoutSession.delegate = self
healthStore.start(workoutSession)
```

```
// Starting Workout from iPhone

// iPhone App
var workoutConfiguration = HKWorkoutConfiguration()
workoutConfiguration.activityType = .running
workoutConfiguration.locationType = .outdoor
```

```
// Starting Workout from iPhone

// iPhone App
if wcSession.activationState == .activated && wcSession.isWatchAppInstalled {
    var workoutConfiguration = HKWorkoutConfiguration()
    workoutConfiguration.activityType = .running
    workoutConfiguration.locationType = .outdoor
}
```

```
// Starting Workout from iPhone

// iPhone App
if wcSession.activationState == .activated && wcSession.isWatchAppInstalled {
    var workoutConfiguration = HKWorkoutConfiguration()
    workoutConfiguration.activityType = .running
    workoutConfiguration.locationType = .outdoor
}
```

```swift
// Starting Workout from iPhone

// iPhone App
if wcSession.activationState == .activated && wcSession.isWatchAppInstalled {
   var workoutConfiguration = HKWorkoutConfiguration()
   workoutConfiguration.activityType = .running
   workoutConfiguration.locationType = .outdoor
}
```

```
// Starting Workout from iPhone

// iPhone App
if wcSession.activationState == .activated && wcSession.isWatchAppInstalled {
    var workoutConfiguration = HKWorkoutConfiguration()
    workoutConfiguration.activityType = .running
    workoutConfiguration.locationType = .outdoor
    healthStore.startWatchApp(with: workoutConfiguration) { (success, error) in
        ...
    }
}
```

```
// Starting Workout from iPhone

// iPhone App
if wcSession.activationState == .activated && wcSession.isWatchAppInstalled {
   var workoutConfiguration = HKWorkoutConfiguration()
   workoutConfiguration.activityType = .running
   workoutConfiguration.locationType = .outdoor
   healthStore.startWatchApp(with: workoutConfiguration) { (success, error) in
      ...
   }
}
```

```
// Starting Workout from iPhone
```

```swift
// Starting Workout from iPhone

// Apple Watch App
let workoutSession = HKWorkoutSession(configuration: workoutConfiguration)
workoutSession.delegate = self
healthStore.start(workoutSession)
```

```
// Starting Workout from iPhone

// Apple Watch App

   func handle(_ workoutConfiguration: HKWorkoutConfiguration) {
      let workoutSession = HKWorkoutSession(configuration: workoutConfiguration)
      workoutSession.delegate = self
      healthStore.start(workoutSession)
   }
```

```swift
// Starting Workout from iPhone

// Apple Watch App
class ExtensionDelegate: WKExtensionDelegate {
    func handle(_ workoutConfiguration: HKWorkoutConfiguration) {
        let workoutSession = HKWorkoutSession(configuration: workoutConfiguration)
        workoutSession.delegate = self
        healthStore.start(workoutSession)
    }
}
```

```
// Starting Workout from iPhone

// Apple Watch App
class ExtensionDelegate: WKExtensionDelegate {
   func handle(_ workoutConfiguration: HKWorkoutConfiguration) {
      let workoutSession = HKWorkoutSession(configuration: workoutConfiguration)
      workoutSession.delegate = self
      healthStore.start(workoutSession)
   }
}
```

Demo

SpeedySloth: iPhone App

Jorge Moriñigo
iOS Software Engineer

Best Practices

Best Practices

Ensure Apple Watch app is functional when iPhone is unreachable

Best Practices

Ensure Apple Watch app is functional when iPhone is unreachable

- Keep session running when losing connectivity

Best Practices

Ensure Apple Watch app is functional when iPhone is unreachable

- Keep session running when losing connectivity
- Use HealthKit distance when GPS is unavailable

Best Practices

Ensure Apple Watch app is functional when iPhone is unreachable

- Keep session running when losing connectivity
- Use HealthKit distance when GPS is unavailable

User should be able to begin workout on Apple Watch or iPhone

Best Practices

Ensure Apple Watch app is functional when iPhone is unreachable

- Keep session running when losing connectivity
- Use HealthKit distance when GPS is unavailable

User should be able to begin workout on Apple Watch or iPhone

Display workouts from other sources

Best Practices

Ensure Apple Watch app is functional when iPhone is unreachable

- Keep session running when losing connectivity
- Use HealthKit distance when GPS is unavailable

User should be able to begin workout on Apple Watch or iPhone

Display workouts from other sources

Don’t display deleted workouts

Summary

Summary

Background running

Summary

Background running

Contribute to Activity rings

Summary

Background running

Contribute to Activity rings

Start workout from Apple Watch or iPhone

More Information

https://developer.apple.com/wwdc16/235

Related Sessions

Getting the Most out of HealthKit	Wednesday
Health and Fitness with Core Motion	Thursday
Introducing HealthKit	WWDC 2014
Designing Accessories for iOS and OS X	WWDC 2014
What’s New in HealthKit	WWDC 2015

WWDC16