

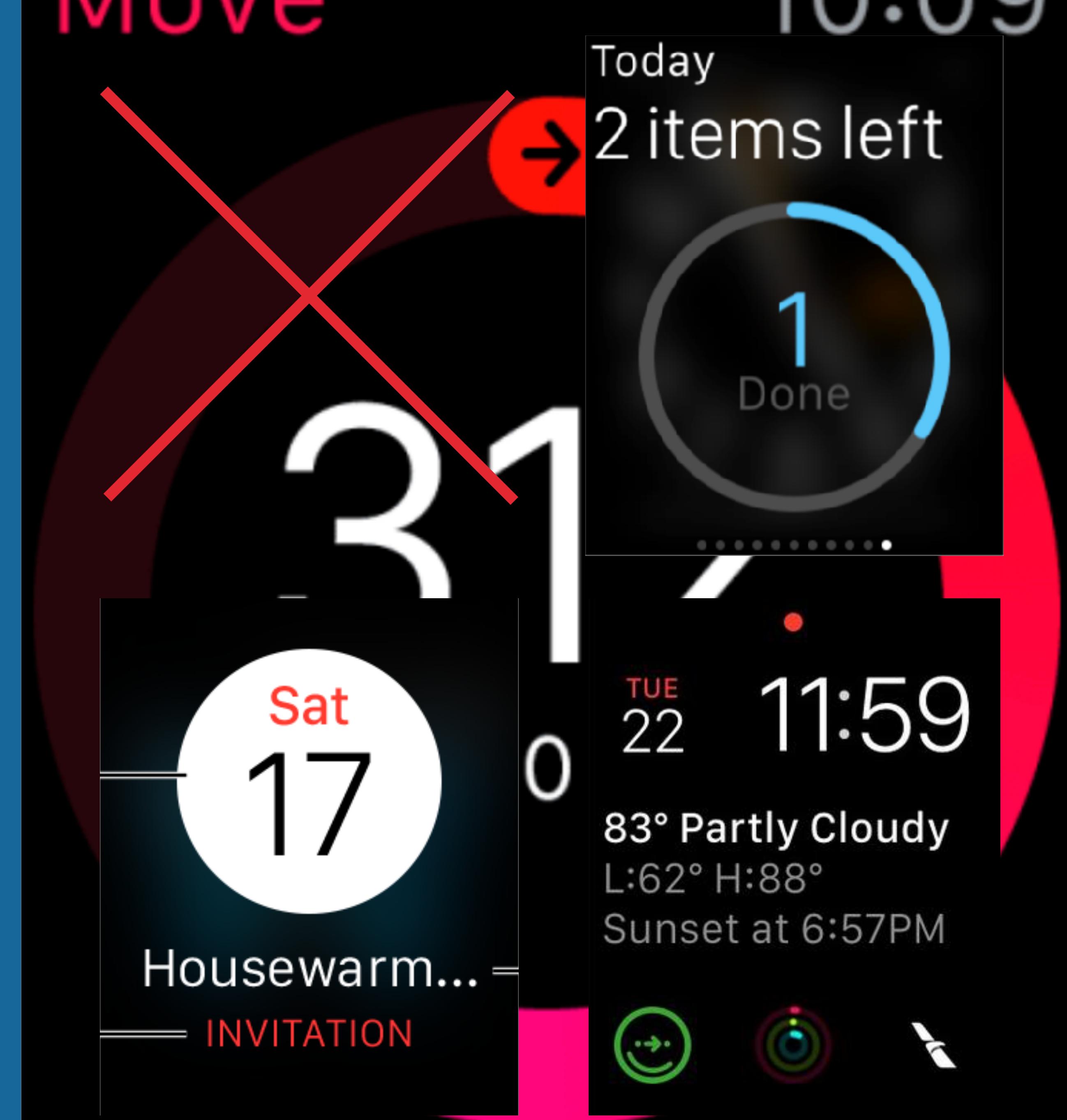
USER INTERACTIONS ON



USER INTERACTIONS

SUBTITLE

- User interactions available
 - WatchKit App
 - Glances
 - Dock
 - Custom notifications
 - Complication (new in 2.0)



USER INTERACTIONS

- What makes up a Watch App?
 - Interface
 - Notification Interface
 - Complication Interface
 - Code for managing all interfaces in extension

The screenshot shows the Xcode interface with a project titled "InterFaces" selected. The project structure is as follows:

- InterFaces (Target)
 - InterFaces (Group)
 - AppDelegate.swift
 - ViewController.swift
 - Main.storyboard
 - Assets.xcassets
 - LaunchScreen.storyboard
 - Info.plist
 - InterFaces WatchKit App (Group)
 - Interface.storyboard
 - Assets.xcassets
 - Info.plist
 - InterFaces WatchKit Extension (Group)
 - InterfaceController.swift
 - ExtensionDelegate.swift
 - NotificationController.swift
 - GlanceController.swift
 - ComplicationController.swift
 - Assets.xcassets
 - Info.plist
 - Supporting Files
 - Products

The code editor on the right displays the content of the `InterfaceController.swift` file:

```
// InterfaceController.swift
// InterFaces WatchKit Extension
// Created by T. Andrew Bi... 2016 The University of Texas at Austin
// Copyright © 2016 The University of Texas at Austin. All rights reserved.

import WatchKit
import Foundation
import WatchConnectivity

class InterfaceController: WKInterfaceController {

    // MARK: - IBOutlets
    @IBOutlet var nameLabel: WKInterfaceLabel

    // MARK: - Connectivity session
    let session = WCSession.default()

    override func awake(withContext context: Any?) {
        super.awake(withContext: context)
        // Set the name label
        nameLabel.setText("Hello, World!")
        // Check if the session is connected
        if session.isConnected {
            session.sendMessage(["text": "Hello, World!"], replyHandler: { [weak self] (reply) in
                self?.nameLabel.setText(reply["text"] as? String)
            }, errorHandler: { [weak self] (error) in
                print("Error: \(error.localizedDescription)")
            })
        }
    }

    @IBAction func tapToInteractive(_ sender: WKInterfaceButton) {
        print("Tap (interactive)")
        let applicationDict = [
            "text": "Hello, World!"
        ]
        session.sendMessage(applicationDict, replyHandler: { [weak self] (reply) in
            self?.nameLabel.setText(reply["text"] as? String)
        }, errorHandler: { [weak self] (error) in
            print("Error: \(error.localizedDescription)")
        })
    }

    @IBAction func tapToApplication(_ sender: WKInterfaceButton) {
        print("Tap (application)")
        do {
            let applicationDict = [
                "text": "Hello, World!"
            ]
            try session.default().sendMessage(applicationDict)
        } catch {
            // Handle errors here
            print(error)
        }
    }

    // MARK: - Lifecycle
    override func awake(withContext context: Any?) {
        super.awake(withContext: context)
        // Set the name label
        nameLabel.setText("Hello, World!")
        // Check if the session is connected
        if session.isConnected {
            session.sendMessage(["text": "Hello, World!"], replyHandler: { [weak self] (reply) in
                self?.nameLabel.setText(reply["text"] as? String)
            }, errorHandler: { [weak self] (error) in
                print("Error: \(error.localizedDescription)")
            })
        }
    }
}
```

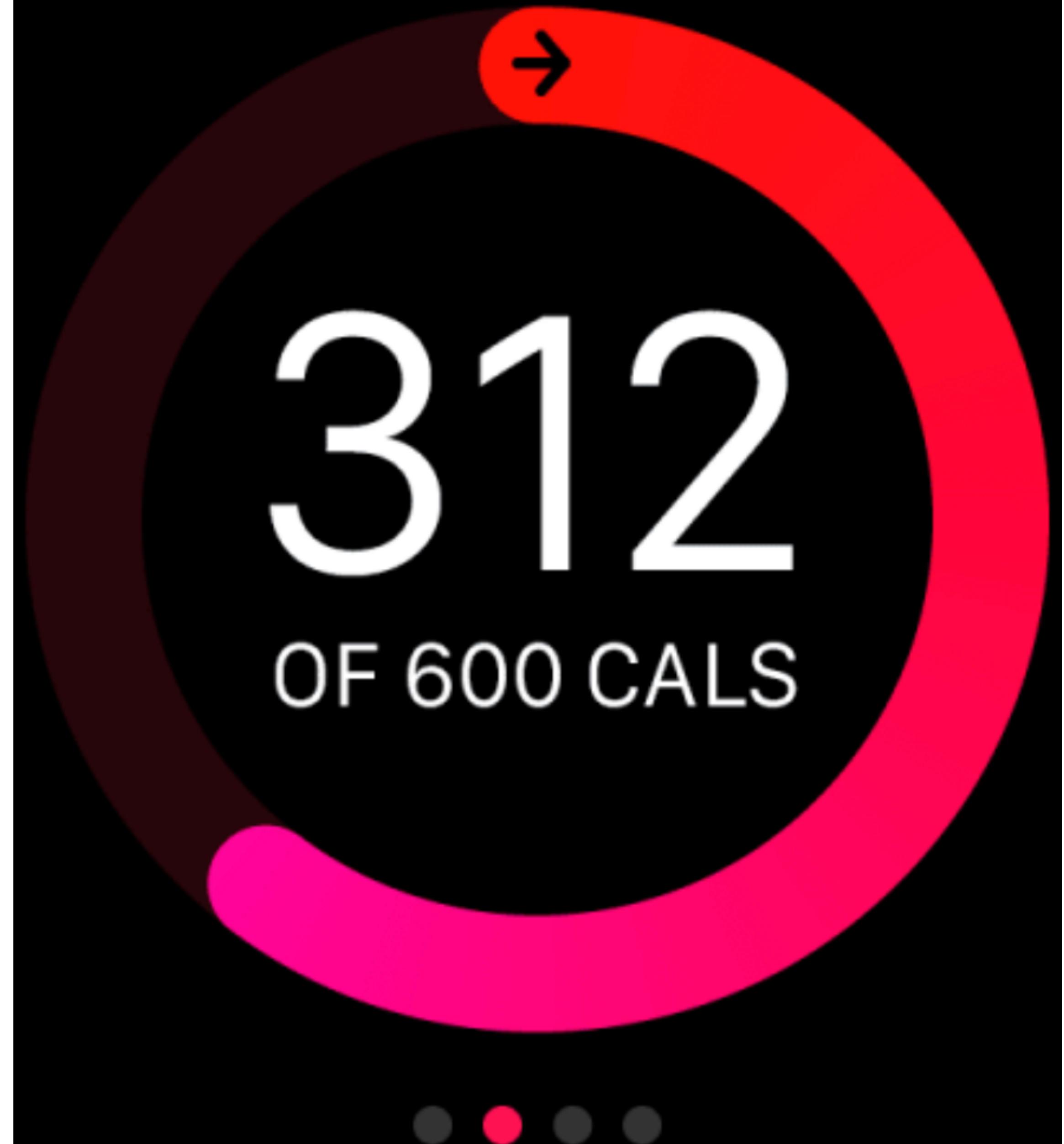
USER INTERACTIONS

SUBTITLE

- WatchKit App
 - Full-app experience which they interact with by opening your app from the Home screen
 - Main way to interact with data
 - Typically, users will only interact with a subset of data/features

Move

10:09



USER INTERFACES ON WATCHOS

USER INTERACTIONS



Glanceable



Actionable

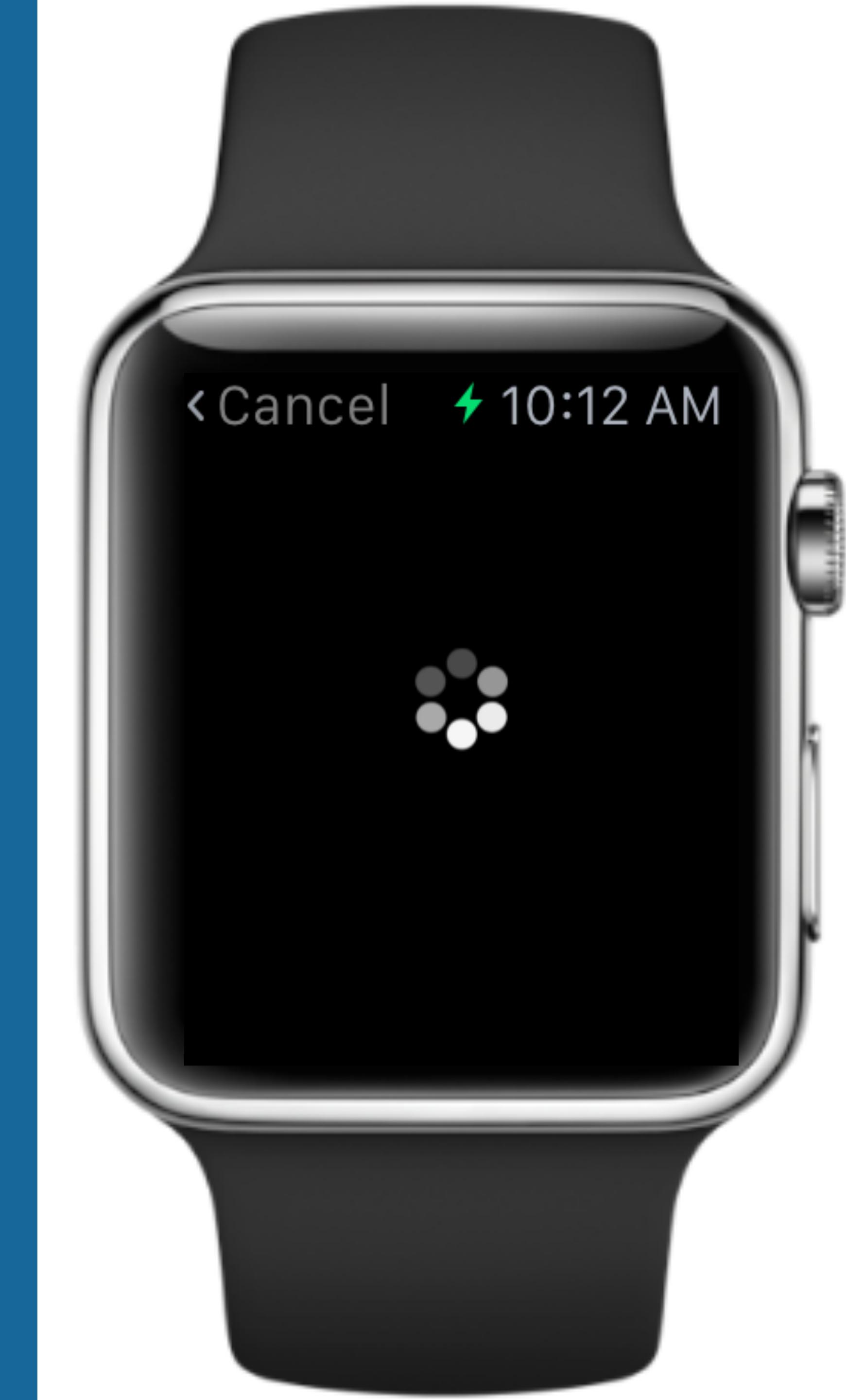


Responsive

- Design for quick interactions

USER INTERACTIONS

- Users will not be interested in seeing this interface in your application



USER INTERACTIONS

- How long is a quick interaction?

. 2 seconds

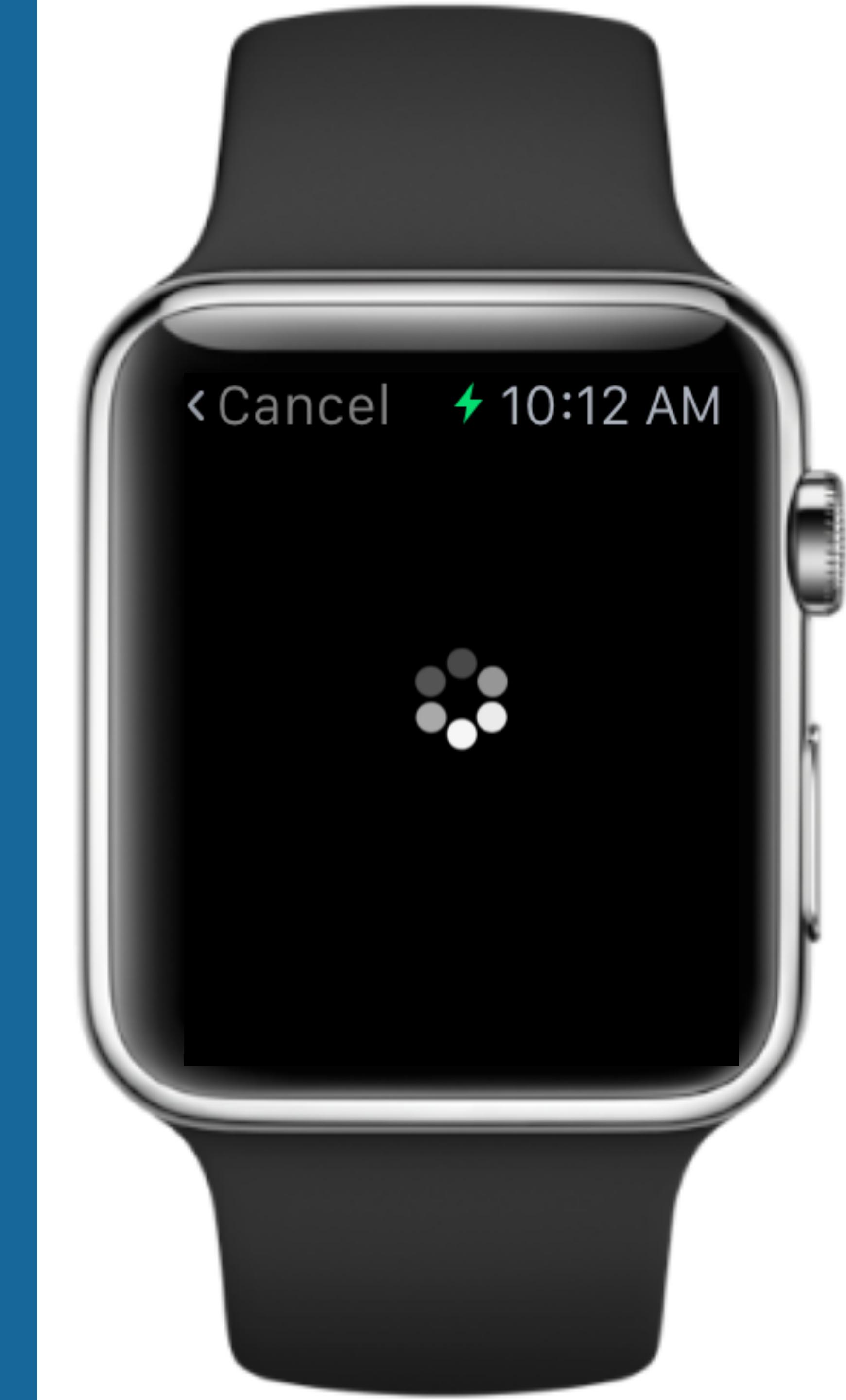
USER INTERACTIONS

- Design for quick and fast interactions
- Be very conscious of load times
- Compliment your iOS app
- Independent watch apps



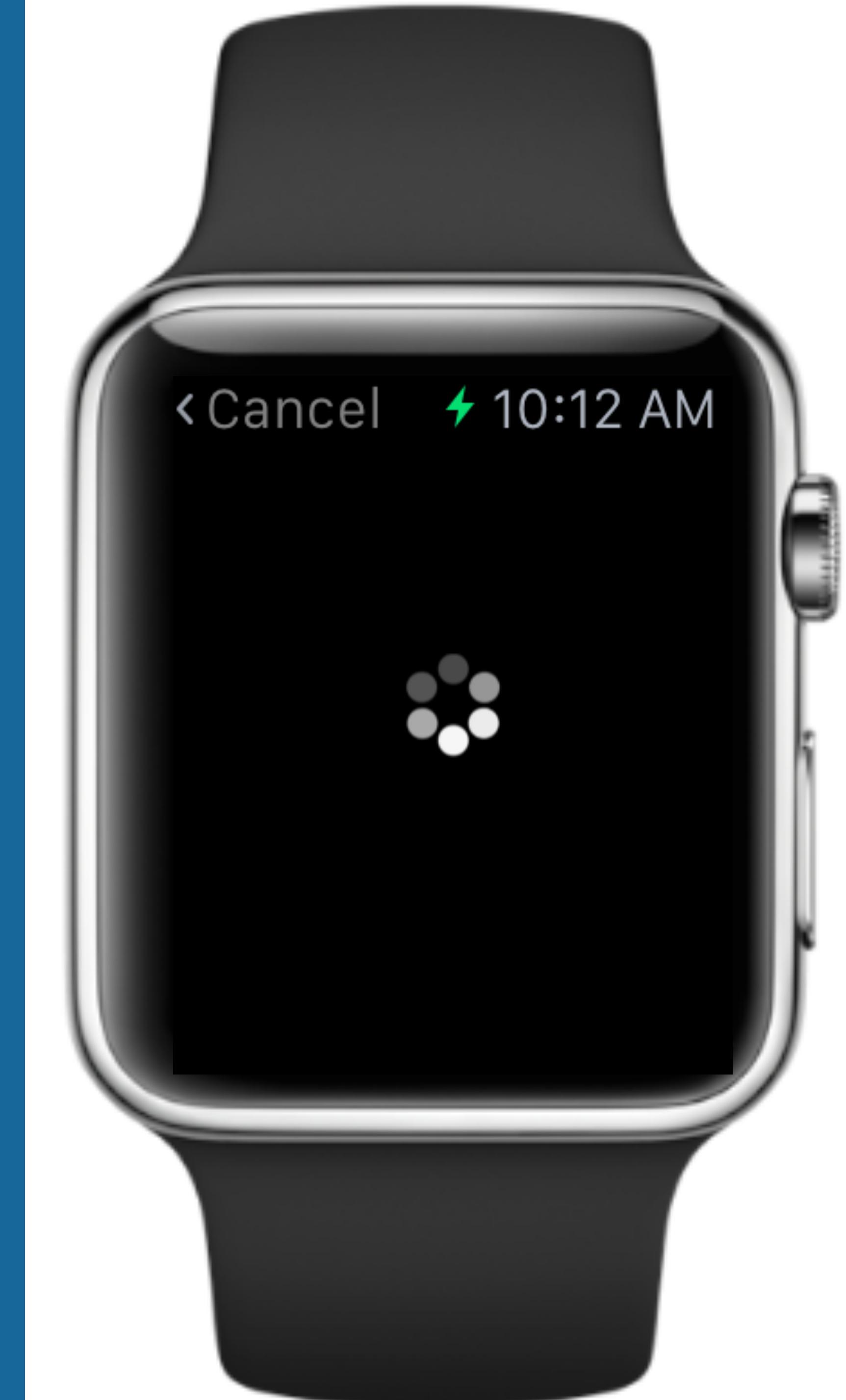
USER INTERACTIONS

- Interacting
 - Gesture recognizers
 - Digital crown rotation



USER INTERACTIONS

- watchOS additions to support displaying and updating information
 - Improved table navigation (vertical paging)
 - Support for new Notifications Framework
 - SceneKit and SpriteKit integration



SOME WATCH APPS

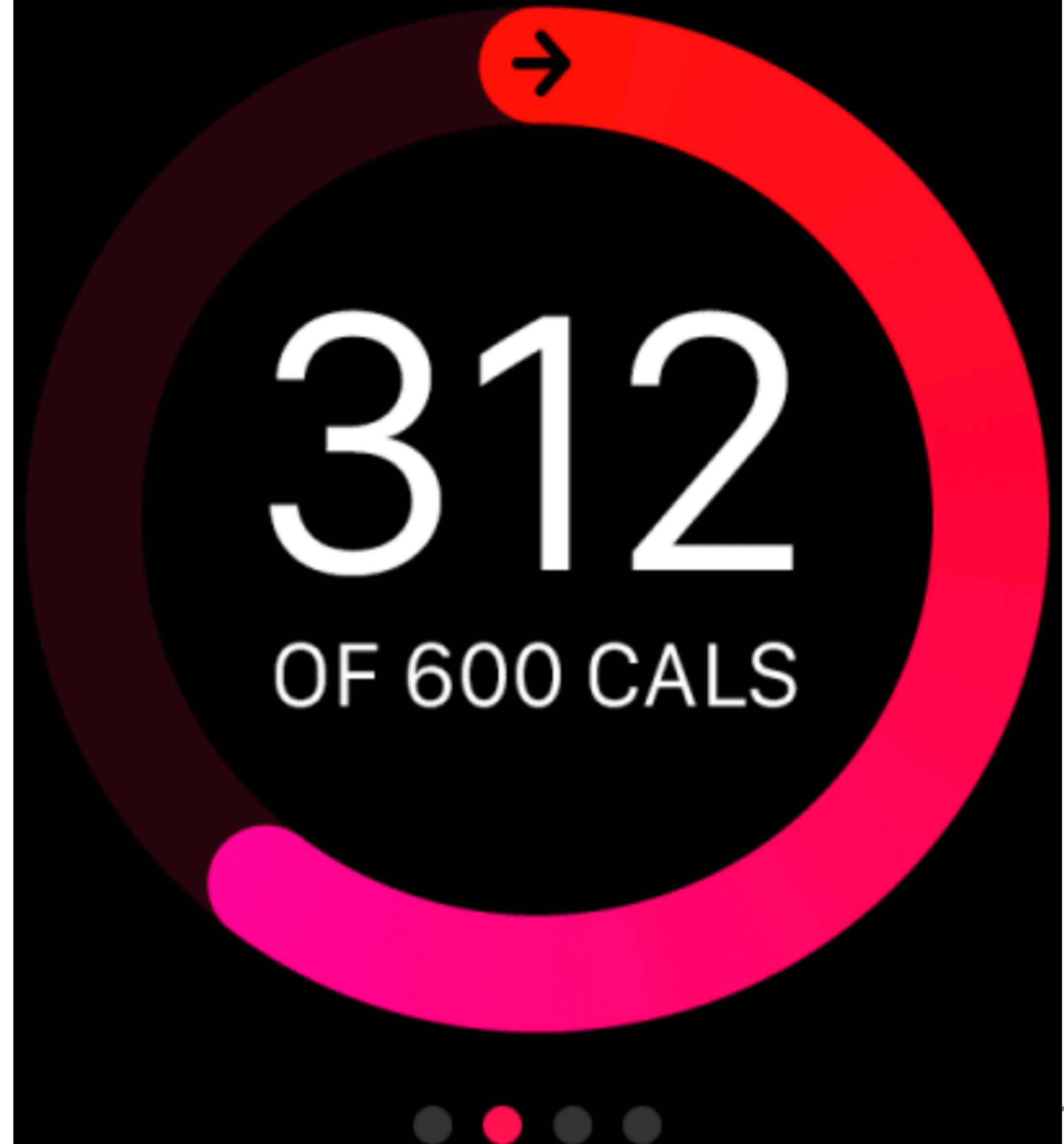
USER INTERACTIONS

SUBTITLE

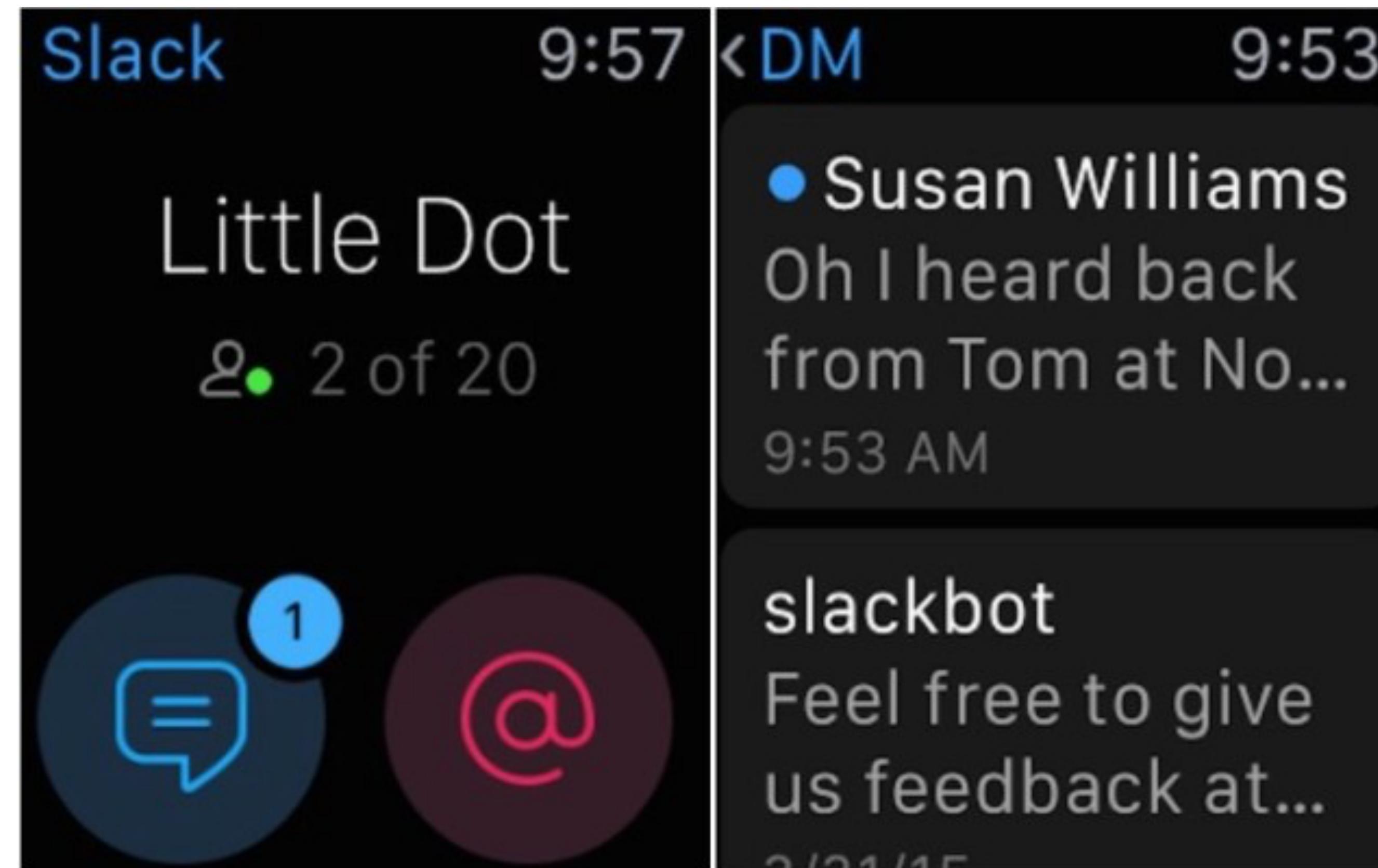
- Watch apps have different navigation and interaction models
 - Health app
 - Music app
 - Game app
 - App app

Move

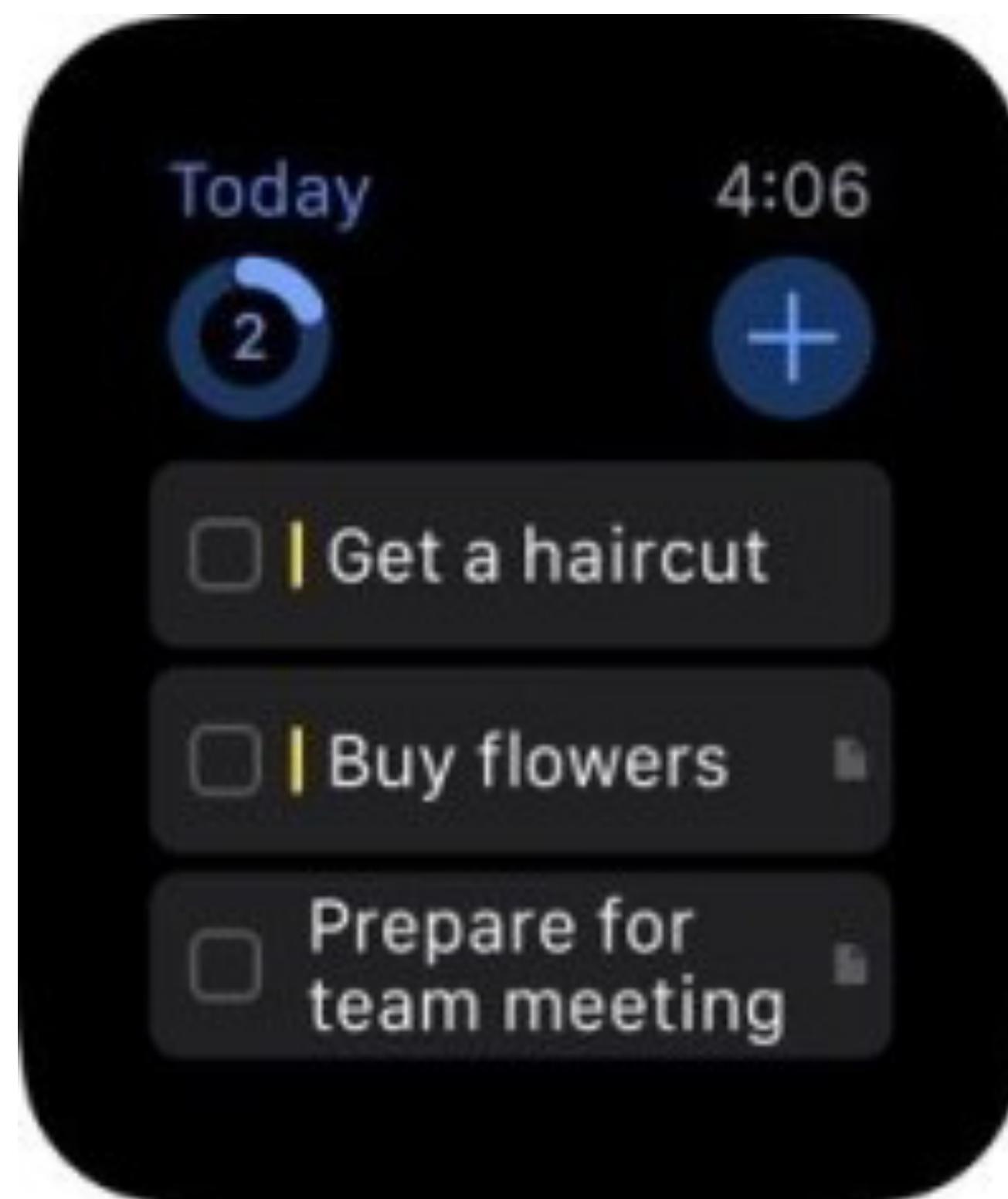
10:09



USER INTERACTIONS



USER INTERACTIONS



USER INTERACTIONS



USER INTERACTIONS



USER INTERACTIONS

Deliveries 10:09

Sailor Moon S... Delivered: Nort...

3 DAYS Seconds and... In transit: Toro...

8 DAYS 2 11-inch Ma... Ship date unk... Updated a moment ago

< Seconds an... 10:09

3 days

In transit: Toronto, ON

Delivered by Saturday

A map of the Great Lakes region showing the delivery route. A green dot marks the starting point in Buffalo, NY, and a grey dot marks the destination in Toronto, ON. The route follows the western shore of Lake Erie and the northern shore of Lake Ontario.

USER INTERACTIONS



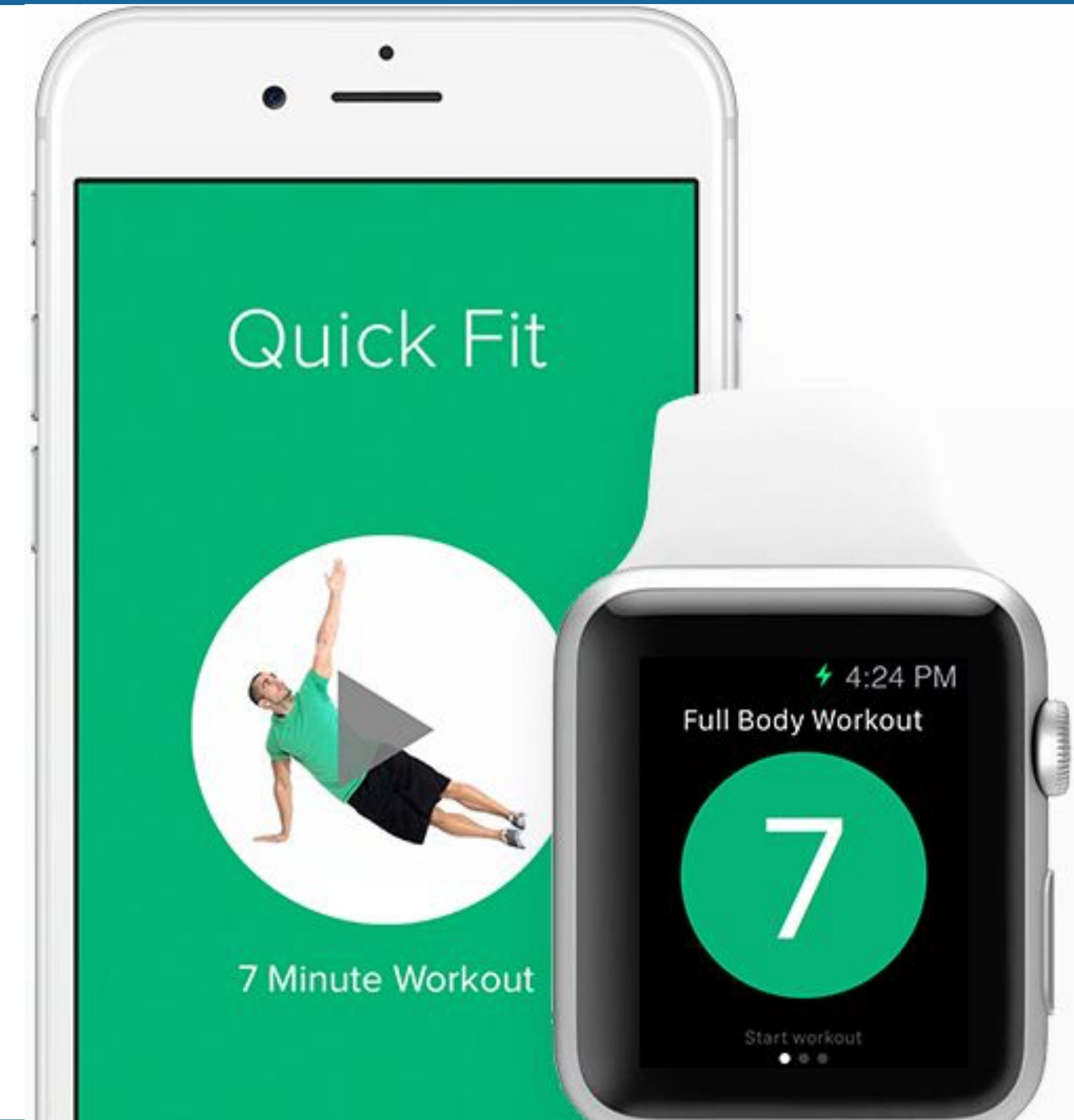
USER INTERACTIONS



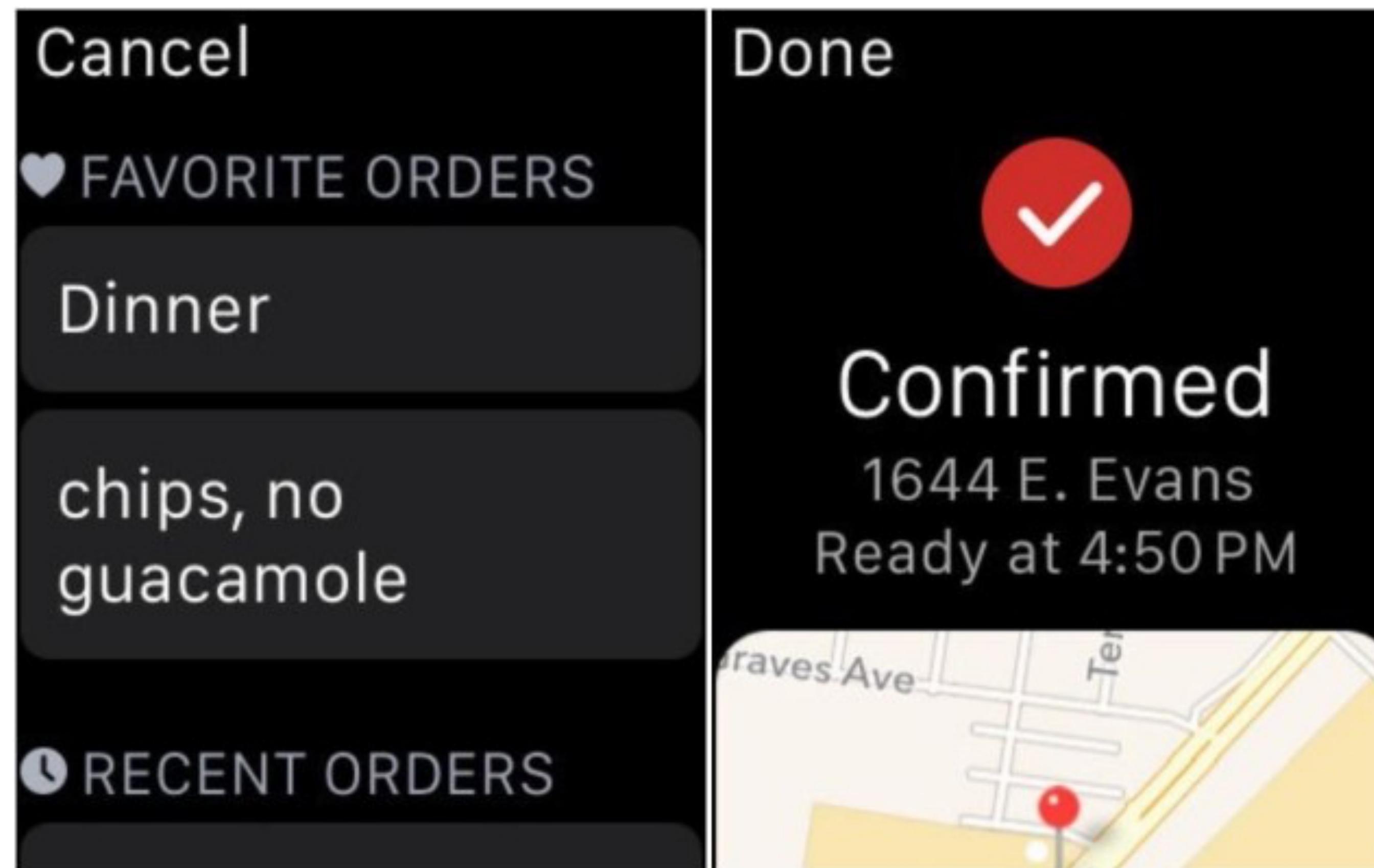
USER INTERACTIONS



USER INTERACTIONS



USER INTERACTIONS



GLANCES

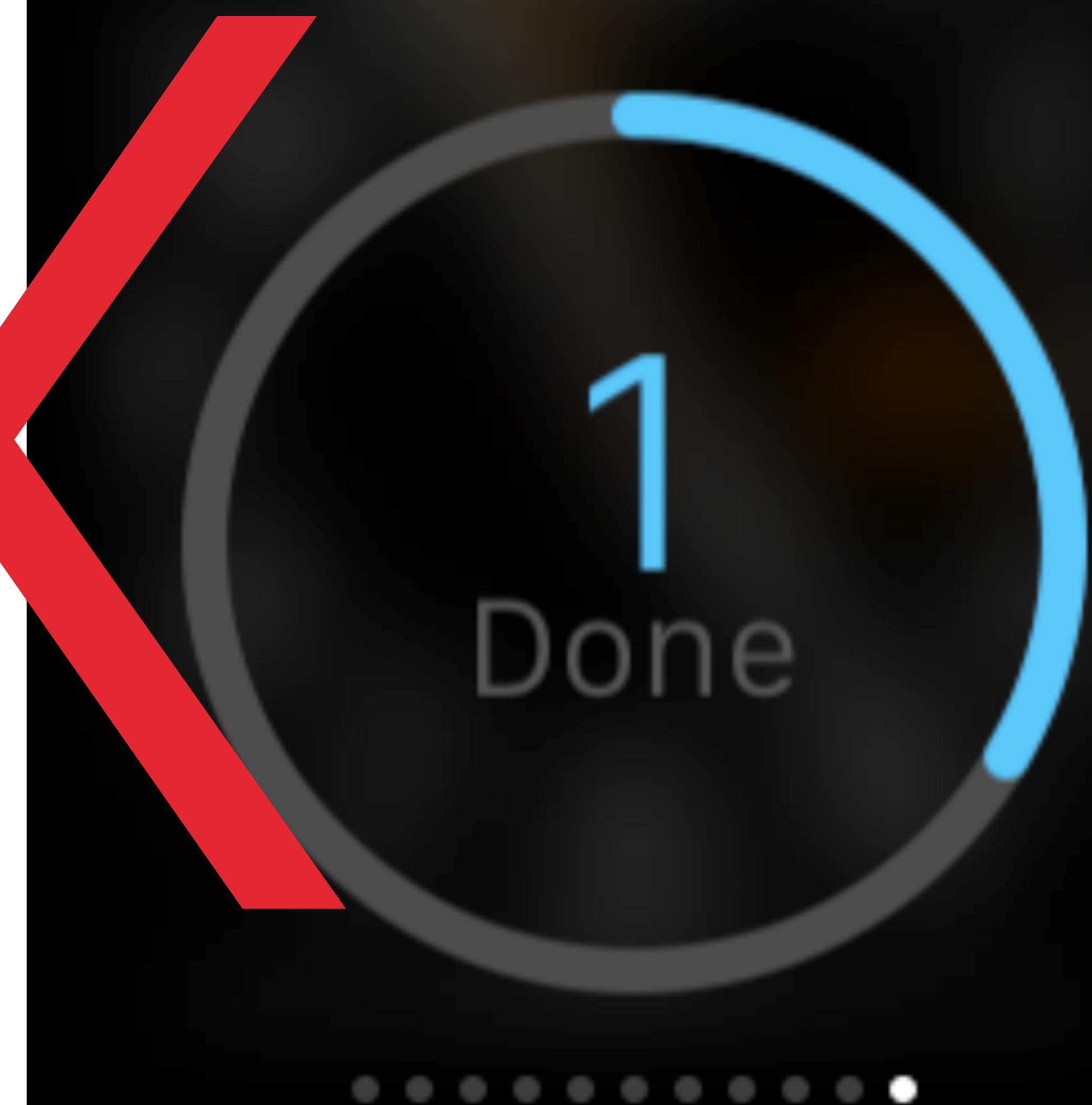


USER INTERACTIONS

SUBTITLE

- Glances
 - A focused interface that you use to display your app's most important information
 - Glances are nonscrolling and read-only
 - Cannot contain buttons, switches, or other interactive controls
 - Tapping a glance launches your WatchKit app
 - “Not to be used as an app launcher”

Today
2 items left



USER INTERACTIONS

GLANCES HAS BEEN REPLACED BY THE DOCK



NOTIFICATIONS

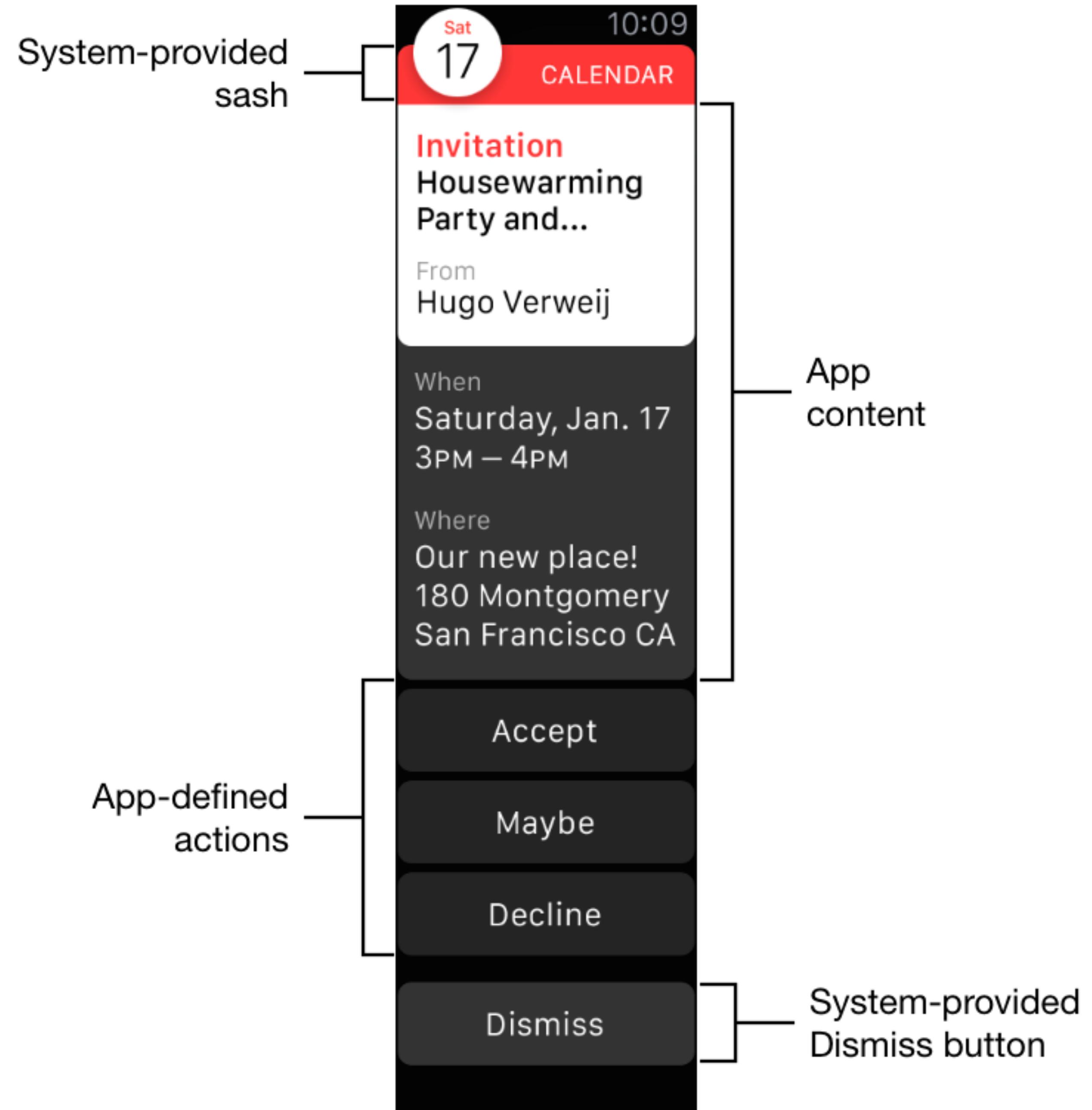
USER INTERACTIONS

- Custom Notifications
 - Works with iPhone to display local and remote notifications
 - Providing custom notification interfaces
 - Short
 - Long



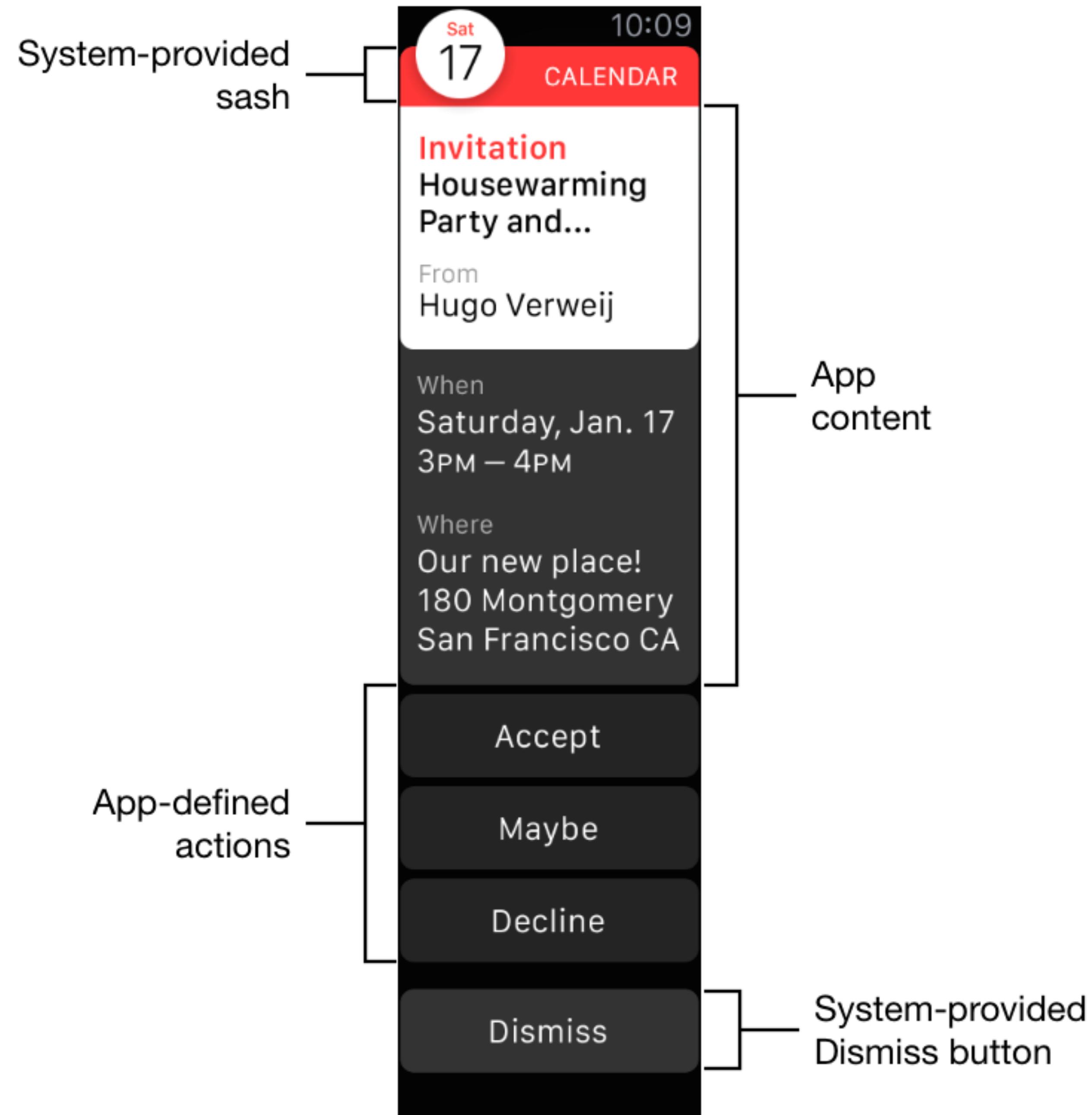
USER INTERACTIONS

- Long looks notifications
- Actionable notifications to increase functionality



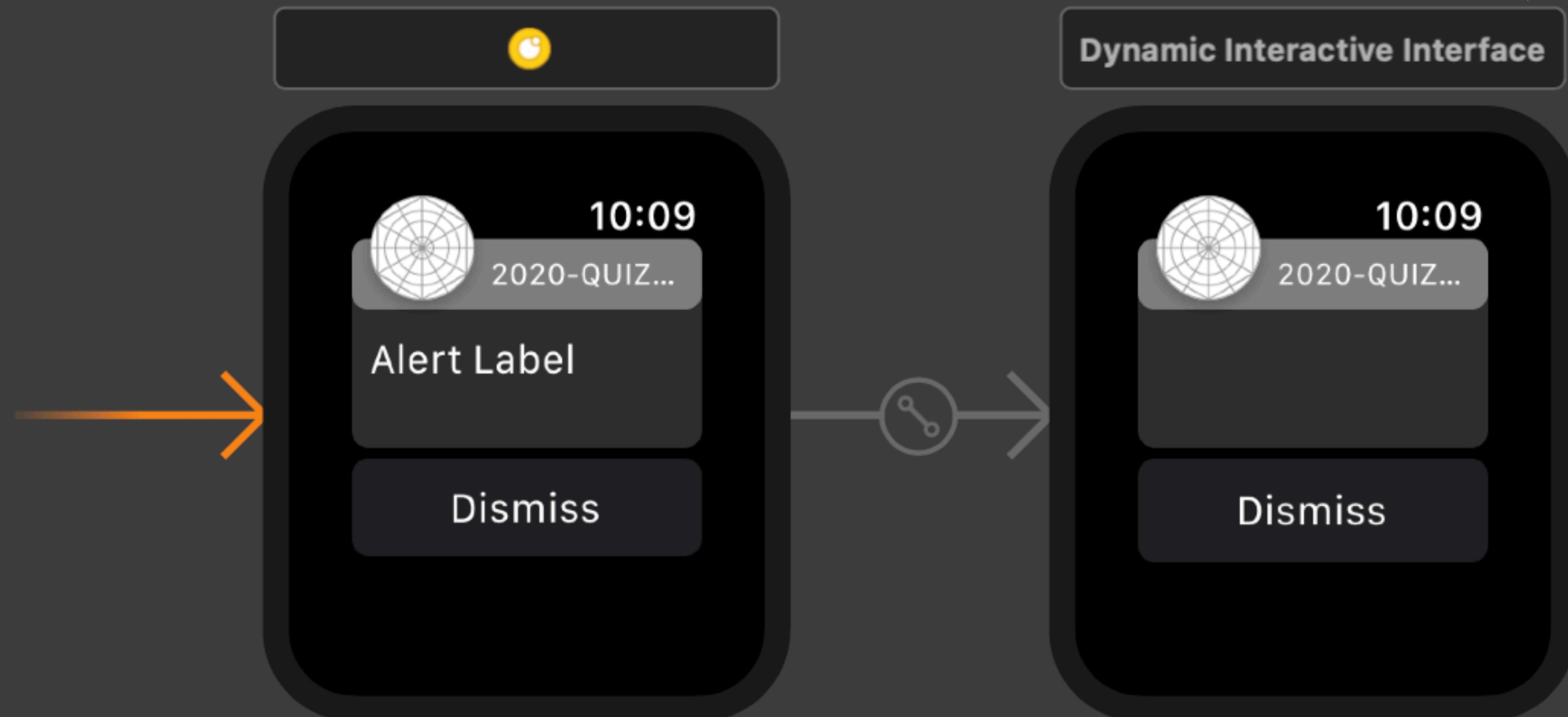
USER INTERACTIONS

- watchOS works with UNNotification framework
- Schedule and handle location notification on watch using extension
 - Time, location based
- Handle remote notifications delivered to watch



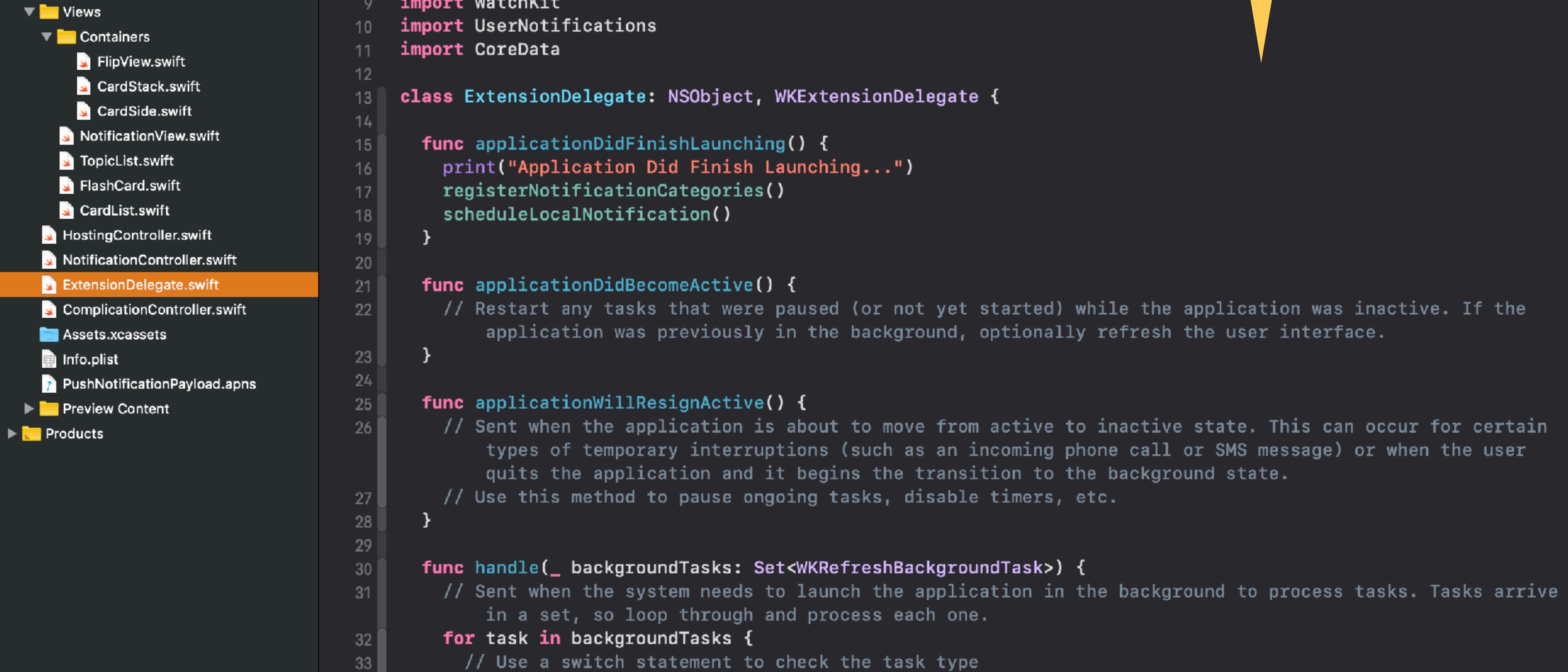
USER INTERACTIONS

Templates from
Xcode



USER INTERACTIONS

Templates from Xcode

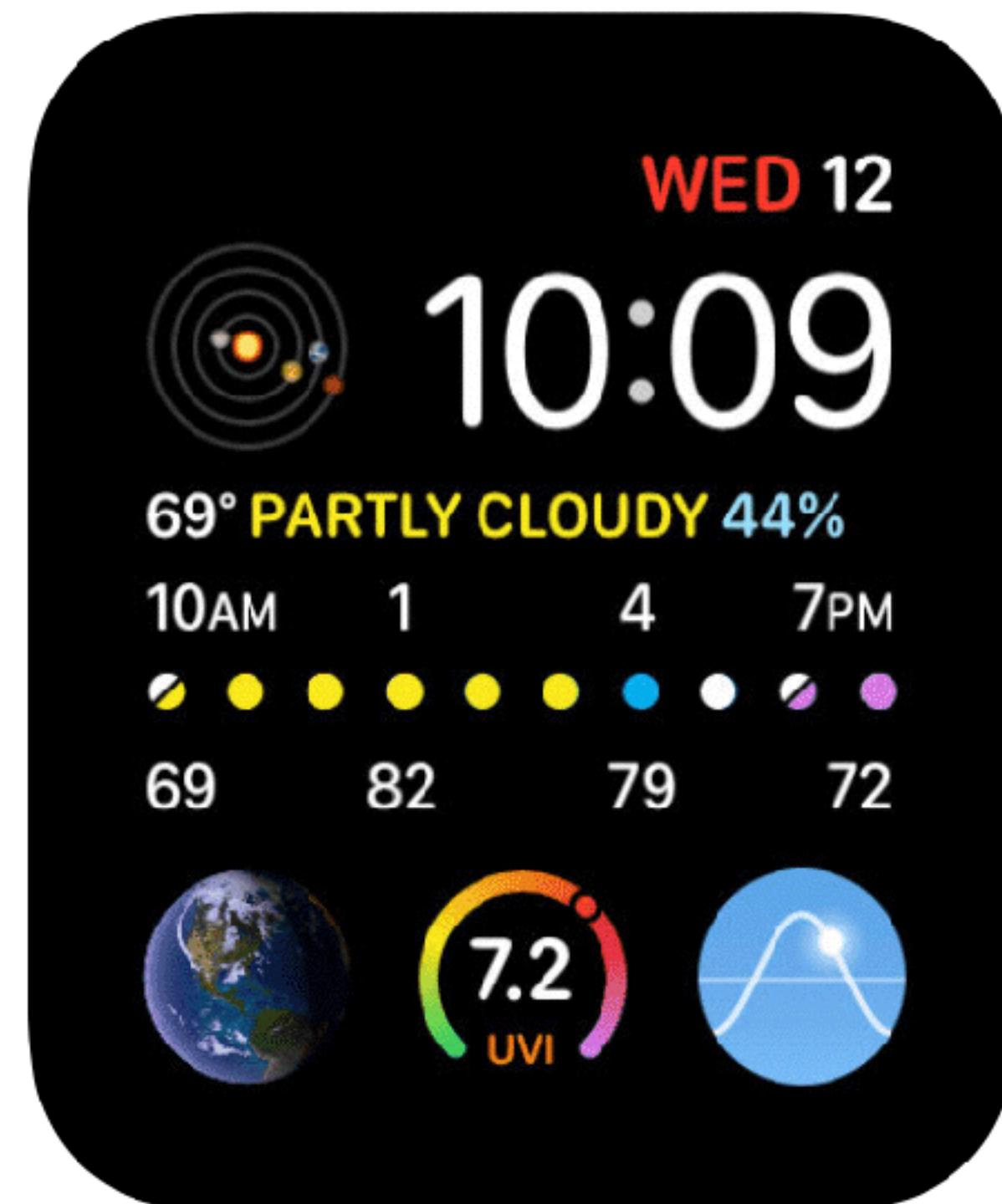


```
9 import WatchKit
10 import UserNotifications
11 import CoreData
12
13 class ExtensionDelegate: NSObject, WKExtensionDelegate {
14
15     func applicationDidFinishLaunching() {
16         print("Application Did Finish Launching...")
17         registerNotificationCategories()
18         scheduleLocalNotification()
19     }
20
21     func applicationDidBecomeActive() {
22         // Restart any tasks that were paused (or not yet started) while the application was inactive. If the
23         // application was previously in the background, optionally refresh the user interface.
24     }
25
26     func applicationWillResignActive() {
27         // Sent when the application is about to move from active to inactive state. This can occur for certain
28         // types of temporary interruptions (such as an incoming phone call or SMS message) or when the user
29         // quits the application and it begins the transition to the background state.
30         // Use this method to pause ongoing tasks, disable timers, etc.
31     }
32
33     func handle(_ backgroundTasks: Set<WKRefreshBackgroundTask>) {
34         // Sent when the system needs to launch the application in the background to process tasks. Tasks arrive
35         // in a set, so loop through and process each one.
36         for task in backgroundTasks {
37             // Use a switch statement to check the task type
38         }
39     }
40 }
```

COMPLICATIONS

USER INTERACTIONS

COMPLICATIONS

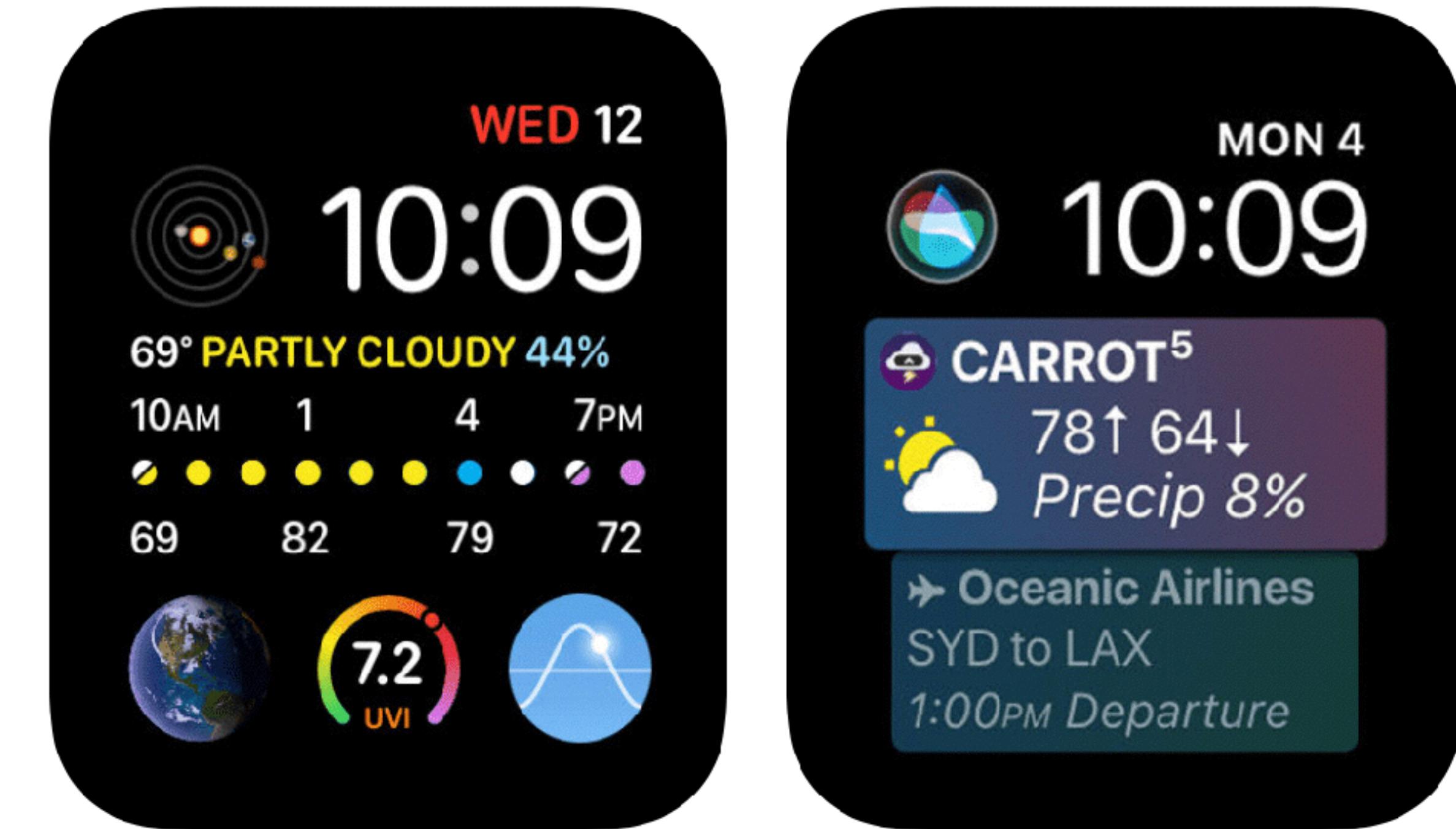


USER INTERACTIONS

COMPLICATIONS

SUBTITLE

- Small visual elements that appear on the watch face
 - Customizable
 - Always visible



USER INTERACTIONS



USER INTERACTIONS

COMPLICATIONS

SUBTITLE

- The term *complication* comes from watch making, where the addition of features added complexity to the watch construction



Customize

USER INTERACTIONS

COMPLICATIONS

Apple recommends that all Watch apps include a complication, even if that complication only acts as a button to launch the app. For information about complications and how to implement them, see [Complication Essentials](#).



USER INTERACTIONS

COMPLICATIONS

- Number of complications varies depending on the watch face

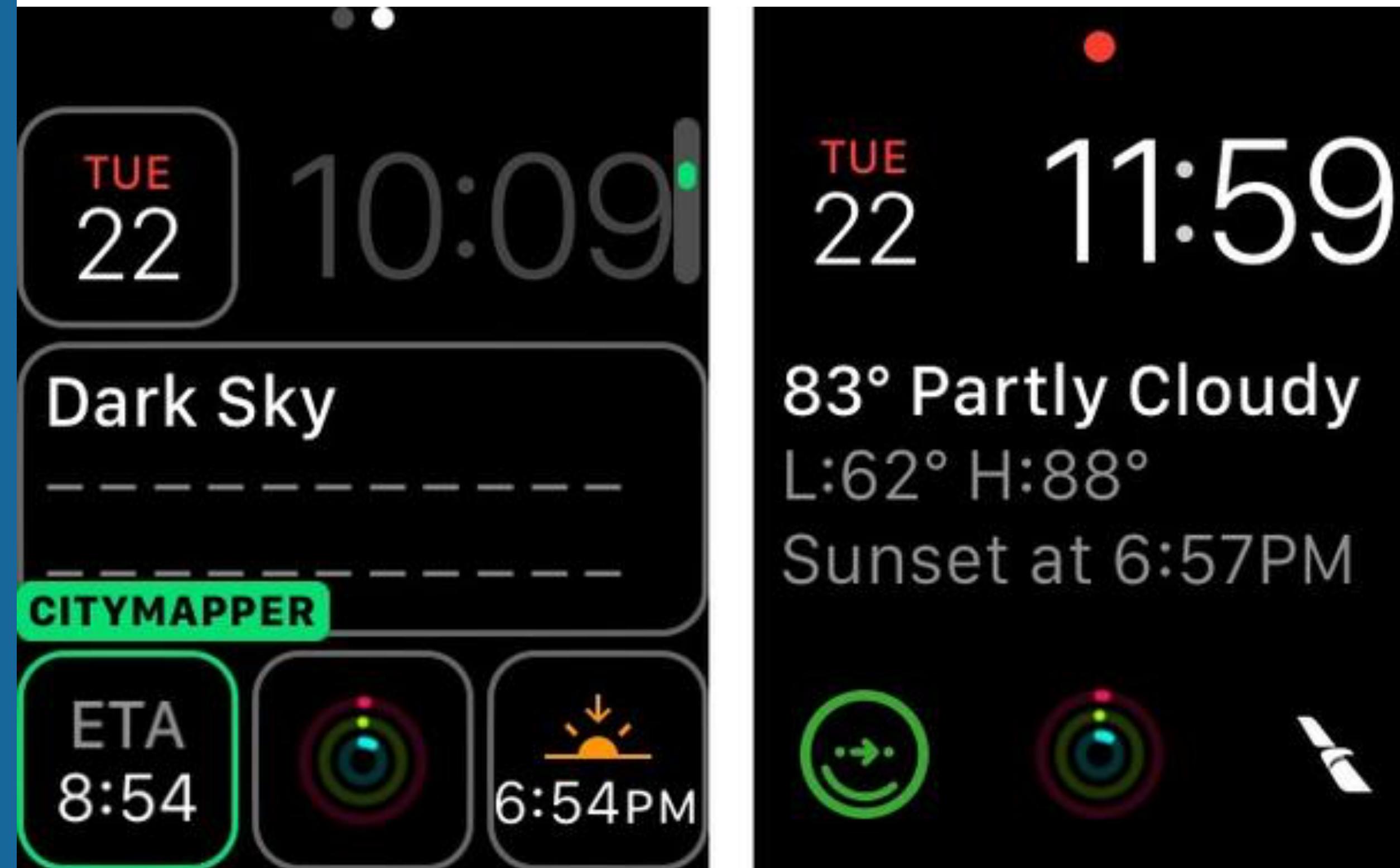


Nike Run Club

USER INTERACTIONS

COMPLICATIONS

- Apps whose complications are selected on the face get VIP treatment
 - Stays in memory; fast launch
 - Receives more time to execute background tasks
 - Receive background updates (at least 2x/hour)



USER INTERACTIONS

COMPLICATIONS

- Different templates for the different watch styles

Graphic Circular

These templates display text, gauges, and full-color images in small circular areas on Infograph and Infograph Modular watch faces. Some of the templates also support multicolor text. The Infograph and Infograph Modular faces are only available on Apple Watch Series 4.



Closed Gauge Image



Closed Gauge Text



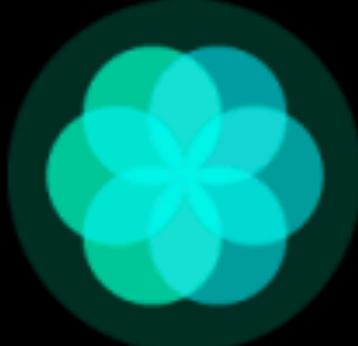
Open Gauge Image



Open Gauge Text



Open Gauge Range



Image

USER INTERACTIONS

COMPLICATIONS

The screenshot shows the Xcode project structure and target settings for a WatchKit Extension named "2020-quiz-starter".

Project Structure:

- Info.plist
- 2020-quiz-starter WatchKit Extension
 - Model
 - FlashCardModel.swift
 - Views
 - Containers
 - FlipView.swift
 - CardStack.swift
 - CardSide.swift
 - NotificationView.swift
 - TopicList.swift
 - FlashCard.swift
 - CardList.swift
 - HostingController.swift
 - NotificationController.swift
 - ExtensionDelegate.swift
 - ComplicationController.swift
 - Assets.xcassets
 - Info.plist
 - PushNotificationPayload.apns
- Preview Content
- Products

TARGETS

- 2020-quiz-starter
- 2020-quiz-starter...
- 2020-quiz-starter...

Display Name: 2020-quiz-starter WatchKit Extension
Bundle Identifier: mobi.uchicago.-020-quiz-starter.watchkitapp.watchkitapp
Version: 1.0
Build: 1

Deployment Info
Deployment Target: 6.2

Complications Configuration

Data Source Class: \$(PRODUCT_MODULE_NAME).ComplicationCor

Supported Families:

- Modular Small
- Modular Large
- Utilitarian Small
- Utilitarian Small Flat
- Utilitarian Large
- Circular Small
- Extra Large
- Graphic Corner
- Graphic Bezel
- Graphic Circular
- Graphic Rectangular

Complications Group: Complication

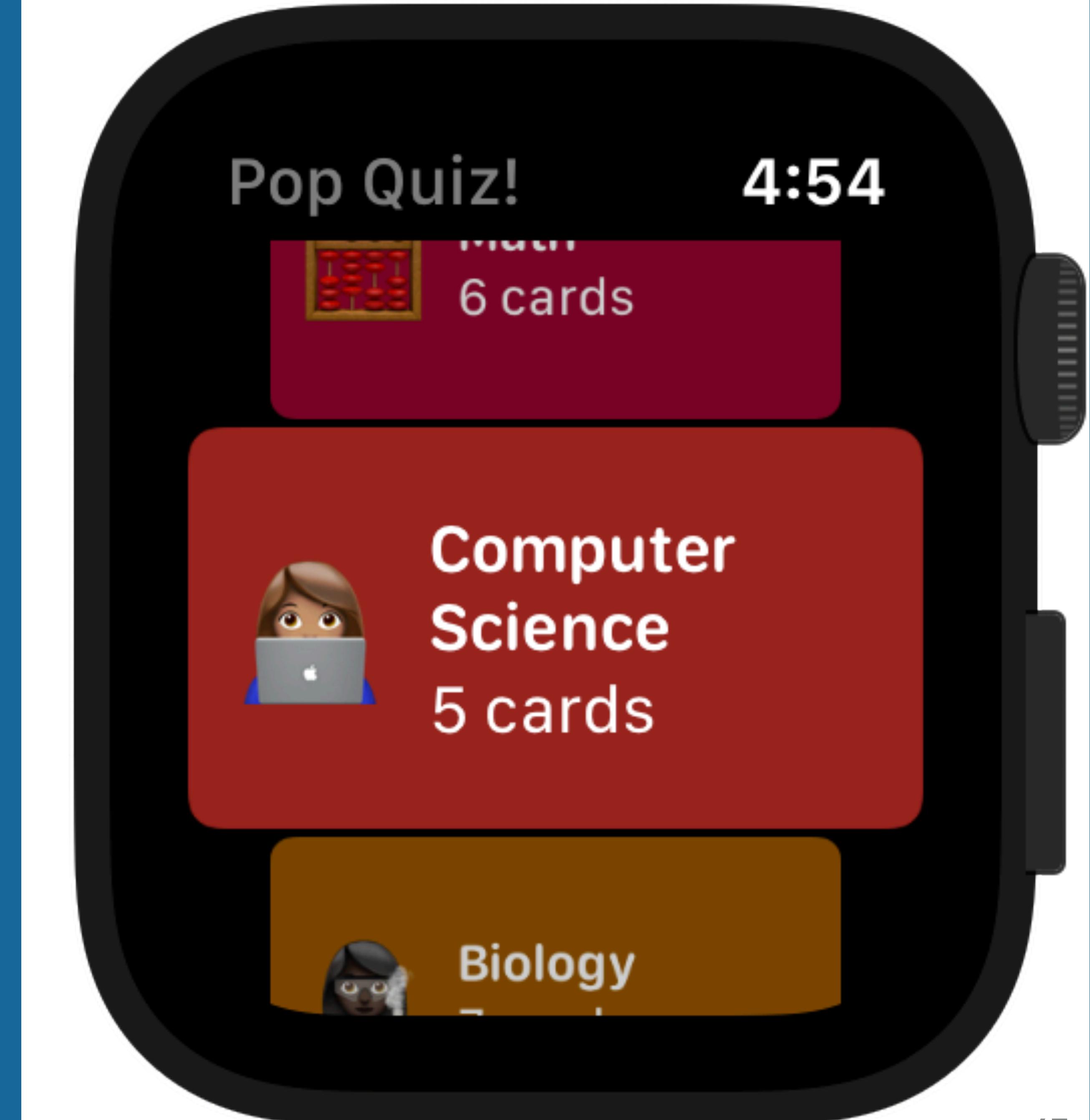
Supported complications

ASSIGNMENT 3

PART 2

ASSIGNMENT 3

- Pop Quiz!
- Independent watchOS application for studying flashcards



ASSIGNMENT 3

SUBTITLE

- Features
 - Core Data model for data persistence
 - Custom notifications
 - From APNS
 - Local notification
 - Complication
 - SwiftUI



ASSIGNMENT 3

SUBTITLE

- Template app that provides core functionality and design
 - Feel free to change anything



ASSIGNMENT 3

- Change data model to use CoreData
 - 3 subjects worth of data
 - 10 questions per subject
- Add buttons for "Correct" and "Incorrect"
- Keep statistics on correct/incorrect



ASSIGNMENT 3

SUBTITLE

- Update Complication to use a "Graphic Circle Closed Gauge Text"
 - Show percent answered correctly
 - Gauge completion is percent correct
- Add a background update to refresh the percentages



ASSIGNMENT 3

SUBTITLE

- Update Notification to handle CoreData model
- Local notification should randomly select an incorrect question from last session
- Handle statistics on the "Correct/Incorrect" actions



ASSIGNMENT 3

SUBTITLE

- Add all necessary app icons





apple WATCH APPLICATION DEVELOPMENT

MPCS 51032 • SPRING 2020 • SESSION 5