COS 333 Product Guide: Jolt

Manbir Gulati, Neamah Hussein, Varun Narayan, Aneesh Rai, and Sharon You

May 10, 2016

1 User Guide

1.1 Introduction

Jolt is an Apple Watch application that functions as a smart alarm system: it ensures the user stays awake during a user-specified period of time. After the user sets a period of time during which they want to stay awake, the application uses an algorithm to track whether the user is awake, and can detect if the user has fallen asleep on the basis of their heartrate and acceleration data. This guide will walk through how to set up the app, and how to operate it. The user interface is fairly simple, with the majority of the work going on the background in order to predict sleep.

1.2 Installing the App

Jolt is currently not in the App Store. Therefore, in order to install it, you will need to clone the git repository into the folder of your choce on your system. In case you're unfamiliar with git, this article explains how to clone a repository to your local device.

At this stage, our source code is contained in a private repository (i.e. we cannot provide a URL); therefore, to access it, please send an email to mgulati@princeton.edu and you will be given access to the code. We plan to have Jolt in the App Store by the end of this year.

Before installing, please make sure the following things are set up:

- Your system is an Apple Mac with at least version 7.3 of XCode installed (this means you need to be running OS X 10.11 or later).
- Your iPhone is charged. Also, your iPhone needs to be running iOS 9.0 or later.
- Your Apple Watch is fully charged. Also, your Apple Watch needs to be running WatchOS 2.2.1 or later.

Connect your iPhone to the computer using your USB cable. Before hitting Run in XCode, make sure your app is set to the main Jolt app bundle, and choose your phone from the device menu. So, instead of "iPhone 6s", as is depicted below, it should read something

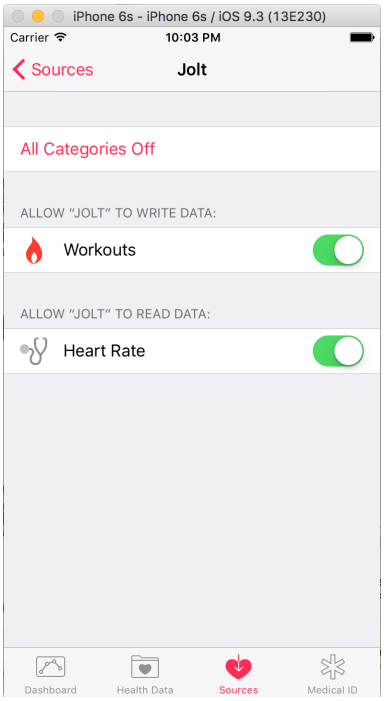
like "Jane's iPhone".

Click the play button to run. The Jolt app will install on your phone. The Jolt iPhone app will now load. It should simply display a black screen with the Jolt logo. The Jolt app will now begin installing on your Apple Watch.

1.3 HealthKit Setup

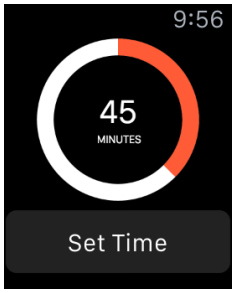
Jolt should automatically request access to relevant HealthKit data via the companion app on your iPhone. In that case, simply authorize it to access heartrate and workout data, as indicated in the screenshot below.

In case you are not automatically requested for access to HealthKit data, go to the Health app on your iPhone. Go to Sources and, under Apps, select Jolt. Select "All Categories On," which authorizes Jolt to access information pertaining to user vitals (see the screenshot below).

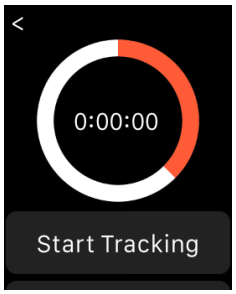


1.4 Set Tracking Time

Once you start the Jolt application on the Apple Watch, the first screen is the Set-Time screen. Using the Apple Watch picker, you can set the amount of time, in one minute increments, that you need Jolt to keep you awake for. The maximum amount of time you can set it is 2 hours (this is based on current battery life estimates of the Apple Watch).



Once you've set your desired time, hit the "Set Time" button. This will now take you to the Tracking screen. Note, the watch has not started tracking you yet. Your screen should the same as in the image below.



1.5 Start Tracking

On the Tracking Screen, hit the "Start Tracking" button. The button text will now change to "Stop Tracking". The watch is now tracking your heart-rate and acceleration, which the algorithm will use to determine whether or not you are falling asleep. The screen will now show you a countdown to the end of your tracking period.

1.6 Now What?

You simply go about your business and let Jolt do its work! The intended functionality of the app is that if you fall asleep while it's still tracking, you'll get a haptic and sound notification to wake you up.

Please note that the current version of WatchKit release has a limitation on third party applications that prevents the app from sending a haptic notification when the watch screen is asleep. So, you will only get haptic and sound feedback to wake you up when the watch screen is active and showing you the Tracking Screen. We anticipate that future releases of WatchKit will resolve this.

1.7 Current Feature Set

Jolt currently has the following capabilities:

- Allowing the user to set a period of time where their vitals are tracked and their sleep status is continuously predicted

- Tracking the user's heartrate in real-time
- Tracking the user's acceleration in real-time
- Utilizing the user's heartrate and acceleration data to predict whether or not they have fallen asleep, in real-time. This involves making predictions using a logistic regression (see the Developer Guide for more details)
- In case the user has fallen asleep, triggering haptic and sound notifications to wake them up. Please note that this functionality is only available if the Watch is awake and the app screen is visible, due to the aforementioned limitaiton with the current version of WatchKit

As previously mentioned, we anticipate that future releases of WatchKit will resolve the issue with third party applications and notifications: once this is the case, we will be able to send notifications even if the Watch is asleep. This will allow us to include additional features, and incorporate user feedback to fine-tune the coefficients on our logistic regression.

2 Developer Guide

2.1 Introduction

Jolt is an intelligent Alarm System built for the Apple Watch. This document will serve as a brief guide to help maintain and familiarize yourself with the platform as you build upon it. We will give a brief introduction to the technologies used to make Jolt possible, a short explanation of how Jolt works, and finally an explanation of the structure of our implementation.

2.2 Dependencies

All non-proprietary technologies were standard development libraries provided by Apple. Jolt depends on the following technologies for development and deployment:

- Apple Watch - Jolt currently has an implementation for all Apple Watch models currently available. All graphics provided have been sized appropriately so as to seamlessly scale between the 38mm and 42mm variants of the Apple Watch. Thorough testing has been conducted on the 42mm variant in hardware, and the 38mm variant in software.
- Xcode $\geq$ 7D175 - This is vital as many of the newer features available on the Apple Watch are only able on the newest version of Xcode currently available.
- WatchOS $\geq$ 2.2.1 - The newest version of WatchOS is vital. WatchOS 2.0 is specifically needed due to introduced support for native apps (Jolt is a native app) and access to raw sensor data. Jolt has known issues involving not being able to operate in the foreground on versions lower than 2.2.0. If you are experiencing this issue, this is likely the cause.
- iOS $\geq$ 9.0 - Jolt depends very little on it's companion iPhone app but 9.0 is needed to maintain health access.
- HealthKit - HealthKit is needed as an entitlement and is an import in both the iOS companion app and the WatchKit extension.
- CoreMotion - CoreMotion is imported into the WatchKit extension.

2.3 Setup

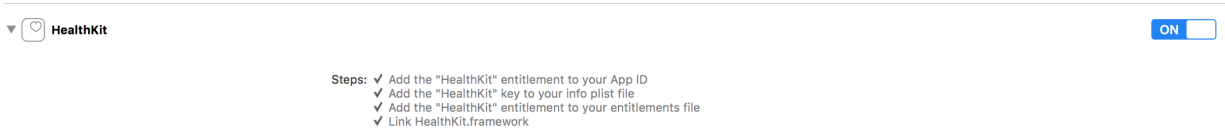
Jolt is currently not in the App Store. Therefore, in order to install it, you will need to clone the git repository into the folder of your choce on your system. At this stage, our source code is contained in a private repository (i.e. we cannot provide a URL); therefore, to access it, please send an email to mgulati@princeton.edu and you will be given access to the code. We plan to have Jolt in the App Store by the end of this year.

Once the repository has been cloned for development, there are a few pieces of setup needed in order to get Jolt to build.

1. Provisioning Profiles need to be configured to support automatic team provisioning. The provisioning profile must be specifically configured to support HealthKit. A detailed guide on how to setup this up may be found below:

   http://bit.do/Team-Provisioning

2. The next step is to add the HealthKit entitlement to both the Companion app target and the WatchKit Extension target. The Companion target entitlements can be found by going to the Project Navigator > Jolt > Capabilities and then enabling HealthKit. The Extension can be enabled by navigating to Project Navigator > Jolt WatchKit Extension > Capabilities. A properly configured HealthKit capability should look as follows:



2.4 App Structure

The majority of the code can be found in the WatchKit Extension folder in the Project Navigator of Xcode. The only proprietary files in the companion app are the image assets for the icons and HealthManager.swift which controls prompting the user for access to HealthKit data. The extension is structured as follows:

- timerRunningInterface.swift - This file is by far the most important in the app. It contains all logic for sleep detection and user interaction while tracking.
- InterfaceController.swift - This file contains the code to manage the first user interaction screen.
- NotificationController.swift - The files contains logic for receiving notifications and handling them.
- Interface.storyboard - There are two files by this name, one in the companion app and one in the WatchKit app. The important file is the one WatchKit app. This contains the entire GUI. The rest of the files, excluding image assets, in the project are boilerplate files which exist in any Apple Watch App.

2.5 The Implementation

The bulk of the codebase is the logistic classifier used to predict sleep, and is roughly outlined below:

Algorithm 1 Classify sleep using Accelerometer and Heart Rate samples

Require: $n_a \geq 9000 \wedge n_{hr} \geq 120$

$A \leftarrow a_0 \dots a_{n_a}$
$HR \leftarrow hr_0 \dots hr_{n_{hr}}$
$\overline{HR} \leftarrow$mean(HR)
$\overline{A} \leftarrow$mean(A)
$\sigma_a^2 \leftarrow$var(A)
$\sigma_{hr}^2 \leftarrow$ var(HR)
while DATA **do**
  $\overline{HR} \leftarrow \overline{HR} \cdot n_{hr} - hr_0 + hr_{n_{hr}+1}$
  $\overline{A} \leftarrow \overline{A} \cdot n_a - a_0 + a_{n_a+1}$
  $\sigma_a^2 \leftarrow$var(A)
  $\sigma_{hr}^2 \leftarrow$ var(HR)
  $P_a \leftarrow \frac{1}{1+e^{-g(\overline{A},\sigma_a^2)}}$
  $P_{hr} \leftarrow \frac{1}{1+e^{-g(\overline{HR},\sigma_{hr}^2)}}$
end while

Essentially, the algorithm searches for known anomalies in the heartrate data and accelerometer data. A heartrate anomaly is defined as a monotonic decrease in beats per minute (BPM) over a period of 10 minutes in which there is an average drop of at least 0.4 $\frac{BPM}{M}$. An accelerometer anomaly is defined as a change of at most 5° between two successive accelerometer samples, sustained over a period of 3 minutes. Sleep is predicted to occur when an accelerometer occurs at the midpoint of a heart rate anomaly ±3min. In simple terms, this means the person's heart rate is decreasing and their wrists are barely moving. Our logistic regression is trained to look for both of these anomalies in the means and variances of the heart rate and accelerometer data.

You can find our implementations of these two algorithms (one for heartrate and one for acceleration) in timerRunningInterface.swift. We have highlighted the relevant code.

The application itself is implemented in three tiers. The first being our GUI implemented using Xcode and auto layout. The next being our sleep detection algorithm outlined above and our control flow. The final being our storage for health data and image assets (logo, animation files, etc...).

2.6 Known Issues

- Notifications do not work on the watch while the screen is off. This is a hardware limitation, and we hope it will be corrected soon.
- Prediction accuracy is somewhat low. The current implementation of the algorithm is in place for proof of concept. Future revisions to Jolt will improve upon this.

2.7 Tips and Tricks

- We suggest keeping the watch plugged into the power cable while developing, battery is drained quickly and testing features can be cumbersome. Jolt will not deploy properly if the watch is not plugged in and battery is low.
- Make sure the watch is snug on your wrist while testing sensors, measurements can be delayed if the watch is not on your wrist or is too loose.
- When testing, be sure to open up the companion app on your iPhone and allow heart rate tracking. This is only needed the first time Jolt is launched.

2.8 Sources of Borrowed/Inspired Code

- We modeled the code to use the sensors for heartrate and accelerometer data off of sample code we found on shu223's GitHub repository that contains code examples for WatchOS2 features.
- We modeled the code to set up a streaming query for heartrate, as well as the appropriate WorkoutSession code off of sample code presented at a 2015 Apple WWDC talk.
- The accelerometer algorithm was primarily derived from here: Accelerometer Algorithm
- The heart rate algorithm was primarily derived from here: Heart Rate Algorithm
- This website was used to create the images for the dials that are an integral part of our UI. We made some of .our own modifications as well.
- This tutorial was used to create the animations for time elapsing whilst tracking, as well as the wake-up screen.
- We used this link for help on setting up the digital crown.