Demo: WaDa - An Android Smart Watch App for Sensor Data Collection

Md Abu Sayeed Mondol
Ifat Afrin Emi
Sirat Samyoun
M Arif Imtiazur Rahman
John A. Stankovic
Department of Computer Science
University of Virginia
Charlottesville, VA, USA
mondol@virginia.edu
iae4qb@virginia.edu
ss8hf@virginia.edu
mir6zw@virginia.edu
stankovic@virginia.edu

Abstract

Collecting sensor data is an essential step for most of the research works in the area of wearable computing. Researchers often spend significant effort and time in developing devices and/or apps for data collection. Most of the apps and devices are customized with limited options, and they are usually non-generic or publicly unavailable. In this demo, we present WaDa, an easy to use app for sensor data collection from commercial off-the-shelf Android smart watches. It provides handy features such as user-defined labelling, time synchronization, sensor and sampling-rate selection, and data navigation. WaDa, freely available for research and academic purposes, facilitates prompt data collection without requiring expertise and effort for custom device/app development.

ACM Classification Keywords

H.5.m [Information interfaces and presentation]: Miscellaneous

App Description

WaDa is a standalone smart watch app that does not require a smart phone. It can collect data from any sensor available in the watch. Due to small form factor, capturing user inputs is challenging for smart watches. WaDa provides easy to use interfaces (Figure 1 and 2) and mechanisms for inputs relevant to sensor data collection.

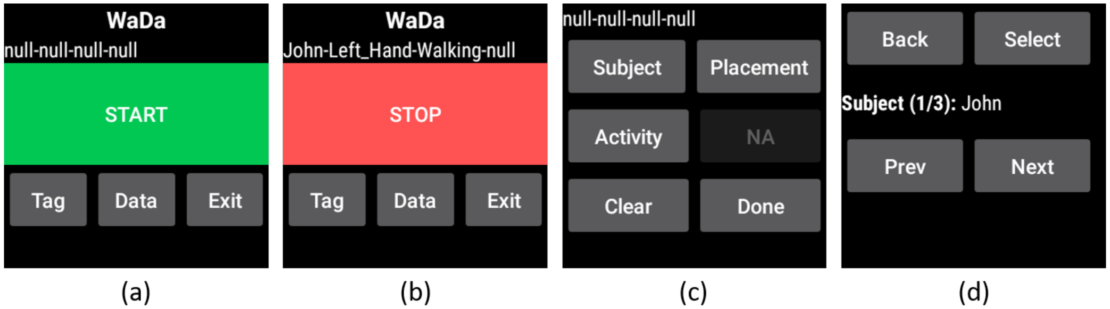
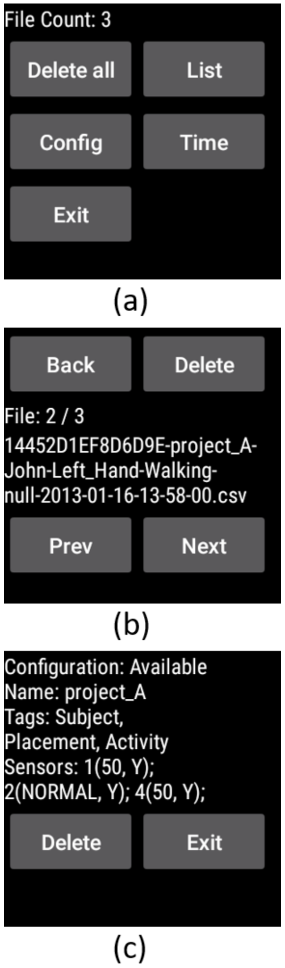


Figure 1: Interfaces from WaDa for (a) Starting and (b) Stopping data collection, (c) Buttons for different tags (d) Options for selecting a tag



File: 2 / 3
14452D1EF8D6D9E-project_A-John-Left_Hand-Walking-null-2013-01-16-13-58-00.csv

Configuration: Available
Name: project_A
Tags: Subject, Placement, Activity
Sensors: 1(50, Y); 2(NORMAL, Y); 4(50, Y);

Figure 2: Interfaces from WaDa (a) Different features, (b) File list, (c) Configuration summary

The app starts collecting sensor data when the ’START’ button is pressed, and the red ’STOP’ button shows up immediately (Figure 1(a) and 1(b)). Once started, the app runs in the background, and closing the app does not stop sensor data collection. Sensor data collection is stopped when the ’STOP’ button is pressed, and data collected between the start and the stop is saved in a single file in the watch. Before starting data collection, several tags can be selected that are included in the file name. Figure 1(c) shows some example tags (Subject, Placement, Activity), and clicking on a button opens a window to select a value for that tag. For example, the subject providing the data can be selected from the subject list (Figure 1(d)) that can be navigated using the ’Prev’ and ’Next’ buttons. The tags and lists can be configured using a configuration file that is pushed into the watch from a computer. A companion desktop app is also available to generate and insert the configuration file (omitted here due to space limitation). Maximum 4 different tags can be set. We used the Device Monitor tool from Android Studio to capture the screen-shots here while running the app on a Sony smart watch.

In addition to the tags, the file name also contain the serial number of the watch and start time. User can navigate through the list of available files in the watch (Figure 2(b)). Individual or all the files can be deleted from the watch if needed. Data files are usually deleted once they are downloaded to the computer. By default, the app collects data at 50 Hz from the Accelerometer. However, a list of sensors along with sampling rates for each of the sensors can be specified in the configuration file. Figure 2(c) shows a configuration in a concise format with the Sensor IDs (e.g., 1, 2, 4), sampling rates, and availability of the sensors in the watch (Y/N). The Sensor IDs are standard as defined by the Android platform (e.g., 1 for the Accelerometer).

Early version of this app has been used for hands-on experiments in two undergraduate courses at the University of Virginia. Students with no or little prior experience of sensor data collection or using smart watches found it simple and easy to use. This app has potential to be used widely for research and academic purposes. Feedback from the Ubicomp community received during the demonstration would be useful for improving the app.