CIT 595 Spring 2017
Group Project #2

INTRODUCTION

In this project, you will work in a group to develop a system with "embedded" and "mobile" components. The goal of the project is to create a Pebble smartwatch application that is able to get data from and send control information to a remote sensor and display driven by an Arduino microcontroller.

REQUIREMENTS

System architecture

The system shall consist of three major components, described as follows:

- *Sensor and Display*: You will use a Gravitech 7-Segment Shield attached to an Arduino microcontroller. This component includes a temperature sensor, a seven-segment display (like a digital clock), and an RGB light. You may start with the Arduino program that is available in Canvas. This program reads the temperature and writes to the serial port once a second.
- *Middleware*: The Arduino shall be connected via USB to a Linux or Mac machine, which will run a C++ program that handles all communication between the sensor and the user interface.
- *User interface*: The user interface will be a Pebble smartwatch application, which communicates with the middleware over the Internet via an Android or iOS phone.

Note that because of firewall restrictions on the SEAS network, if you are running your middleware component on a Linux machine in the lab, you must use port 3001 for your server, and it can only be accessed by devices on one of the Penn networks. Please be sure to discuss this with a member of the teaching staff if you are unsure.

Functional requirements

Your system must fulfill *all* of the following requirements:

- The user should be able to see the most recent temperature sensor reading.
- The user should be able to see the average, low, and high temperature sensor reading for the past hour (if the sensor has been running for less than an hour, the user should be able to see the statistics for the time since the sensor started running).
- The user should be able to decide whether to see temperature readings (and statistics) either in Fahrenheit or Celsius. Additionally, the user should be able to change the 7-segment display on the sensor to show the temperature either in Fahrenheit or Celsius.
- The user should be able to put the sensor into a stand-by mode in which it is not reporting temperature readings (either to the user interface or to the display); however, during this time, the readings should be tracked for determining the high, low, and average. When the sensor is in stand-by mode, the user should also

be able to tell it to resume reporting the readings.

- If the user interface becomes disconnected from the middleware (e.g. because of a network error or if the middleware stops running), an appropriate message should be shown to the user the next time it tries to connect.
- If the middleware cannot get a reading from the sensor (e.g. because the sensor is disconnected), an appropriate message should be shown to the user the next time the user interface tries to get a reading.
- If the middleware becomes disconnected from the sensor but it is still running, it should be possible to reconnect the sensor and resume normal operations without needing to restart the middleware.

Note that you have complete freedom in how you implement these features in the user interface, i.e. what the user needs to do in order to see the data or send a command, and how it gets displayed. However, all user features need to be implemented in a single watchapp. Likewise, all Arduino code should be implemented in a single Arduino sketch (application).

In addition to the features described above, there should be a thread running on the middleware that waits/blocks until a user sitting at the terminal where the middleware was started enters the letter 'q'. Once it sees that, the middleware program should then terminate. It is okay if you require that one more request come in from the user interface before ending the program.

You may assume that all components are dedicated to each other, e.g. that your middleware component will never need to serve a different user interface, and that your user interface will never need to connect to a different server, so it is okay to hardcode addresses, etc. as needed.

Additional features
In addition to the functional requirements listed above, your group must come up with *two* other (non-trivial) features to include in your system. At least one feature must involve sending a control message from the user interface to the display; at least one other feature must involve the processing and/or display of the data that is sent from the sensor to the user interface. Please discuss your additional features with the instructor or a TA before proceeding.

MILESTONES

It is important that you make steady, continuous progress towards your final system. There are various milestones that your group should meet, as described below.

Arduino Tutorial (March 21)
During today's recitation session, your group should complete the Arduino tutorial and ensure that your middleware can receive temperature readings from the sensor.

This will not be graded, but you need to at least get the infrastructure in place so that you

can get data sent between the components.

Pebble Tutorial (March 28)
During today's recitation session, your group should complete the Pebble tutorial and ensure that your smartwatch can communicate with your middleware.

This will not be graded, but can be challenging so be sure to get as much of the tutorial done as possible during the recitation.

Prototype (April 11)
During today's recitation session, your group will have a 5-10 minute meeting with a member of the instruction staff. By this point, you should have much of the basic functionality completed: in particular, you will be required to demonstrate that it is possible to show the current temperature (from the Arduino sensor) on the watch and that you can send some control signal from the watch to the Arduino and have the Arduino take action based on that signal.

During this meeting, be prepared to discuss issues such as:

- where does data processing (e.g., calculating the average temperature) occur?
- what is the structure of your communication protocols between the different components?
- what sorts of error handling have you included in your code?

Also during the recitation, you will be asked about the additional features that you plan on implementing. These features need to be approved by a member of the instruction staff so please discuss them with your teammates before the start of the recitation.

This milestone is worth 25% of the project grade.

Final Presentation (April 25)
Your group will present your system to members of the instruction staff and discuss your implementation decisions. At this point, the entire system should be working and you should have all features implemented. If not, you may present the project at a later date, but with a late penalty to be decided by the instructor.

This milestone is worth 75% of the project grade.

ACADEMIC HONESTY

You should be working with the other members of your group project team, obviously. Other than that, this project is subject to the same policy as for all other assignments, as described in the Syllabus.

In particular, you may not discuss or share code with students in any other groups, or with students who have taken this course in the past. If you want to use a third-party library, or use any code that you did not write yourself, you **must** get permission from the instructor. Failure to do either of these will be considered academic dishonesty, and your team will receive a grade of 0 on this project. Which would be really, really bad.

If you run into problems, please ask a member of the teaching staff for help before trying to find solutions online!

SUBMISSION

All deliverables are due in Canvas by **Tuesday, April 25, 11:59pm.** Late submissions will be subject to a penalty determined by the instructor.

One member of your group should submit all Arduino, C++, and Pebble source code in three *separate* zip files. If you use third-party libraries (which have been approved by the instructor!), only submit those in compiled form. Please be sure that all of your code is well-commented so it is easy for the grader to find where different features are implemented. Remember: happy graders give higher grades!

Also, you must return the Arduino board, temperature sensor shield, USB cable, Pebble smartwatch (sorry!), and any other equipment before you can receive a grade for this assignment.

Updated: 23 March 2017, 4:31pm