Pebble Smartwatch Project - Technical Documentation

Group #7

What is the structure of the messages that are sent from the user interface to the middleware? Give examples.

Messages sent from the user interface to the middleware are in the form of sending a specific request from the phone through XMLHttpRequest object in Javacript. We defines several spicific URLs. The server gets the information from the the request and sends messages to Arduino accordingly. Our defined specific URLs:

Mode / Features	**Pebble Button**	**URL extension**	**Arduino Input**
Resume	up	/resume	1
Pause	up	/pause	2
Celsius	mid	/celsius	3
Fahrenheit	mid	/fahrenheit	4
Average/High/Low	mid long	/light	-
Light Color	down	/timer	6
Timer	down long	/statf /statc	5
Polling	up long	/polling	-

What is the structure of the messages that are sent from the middleware to the user interface? Give examples.

Messages sent from the middleware to the user interface consist of a JSON object with a single key “name” whose value is the string to be displayed. The “#” symbol is used so the phone side can replace it with “\n”. This JSON is sent from the server when gets the required data from the Arduino. For example, the JSON shows avg/high/low temperature looks like this:

{ name : #Average: xx.xxxx#Highest: xx.xxxx#Lowest:xx.xxxx}

What is the structure of the messages that are sent from the middleware to the sensor/display? Give examples.

Messages sent from the middleware to the display is the temperature read from the sensor. Arduino uses "Wire.read()" to read temperature per second. If the current mode is Fahrenheit, Celcius, Polling, Average/High/Low or Light, arduino will set the display on sensor to the temperature it just read in. "Dis_7SEG" method takes the temperature decimal number and send it to display. It also handles the situations where the temperature is negative or positive, in fahrenheit or in celsius, and the total digit of the temperature number. In Timer mode, the arduino keeps reading in temperature data while "Dis_7SEG_Timer" sends back incrementing integers to display per second, so that display shows the counting of total seconds since the timer starts. Both of the methods use "Send7SEG" method which takes the digit number and the ASCII code to set a specific digit on the display.

What is the structure of the messages that are sent from the sensor/display to the middleware? Give examples.

Messages sent from the sensor/display to the middle ware consists of the temperature data in celsius. According to "IsFahrenheit" flag, the arduino takes in this temperature directly or re-calculate it into temperature in fahrenheit. Arduino adds letter "F" or "C" after the temperature and then send this data to server.

How did you keep track of the average temperature? Describe your algorithm and indicate which part of your code implements this feature.

Maintain a queue to store all the temperatures in the past 3600 seconds. Initialize the average to be 0. When each temperature sent to server, check if the size of the queue is 3600:

If the size is less than 3600, multiply the previous average with the current size of the queue, and plus the newest temperature to it, then divide the result by (the current size of the queue + 1). Lastly, push the newest temperature into the queue.

If the size is 3600:

if previously the size is 3600, pop out one temperature from the queue, multiply the previous average with 3600 and plus the difference between the newest temperature and the one poped out, then divide the hte result by 3600. Lastly, push the newest temperature onto the queue.

if previously the size is less than 3600, so the same thing as that when the size is less than 3600

In the code for our server, there is a function named updateAverage(float, float), which is the code where we implement the algorithm.

What are the three additional features that you implemented? Indicate which parts of your code implement these features.

Light Color mode:

In light color mode, click the down button, and the phone sends a request with the "/light" url extension, the server then sends the character '5' to the Arduino. The method "Light_Changing()" (line 201) in temperature.ino changes the RGB bulb on the 7 segment display to flash off, red, green and blue twice (8 seconds). The call to change the bulb to specific color is "digitalWrite".

Timer Mode:

In timer mode, long press the down button, and the phone sends a request with the "/timer" url extension, the server then sends the character '6' to the Arduino. The method "Dis_7SEG_Timer" (line 353) in temperature.ino changes the 7 segment display to a timer. There is a global counter variable called "Timer_Counter" which is set to be zero if no in Timer Mode, and is increment by one every one second in Timer Mode.

Polling Mode:

In polling mode, long press the up button and the phne will sends a request with the "/polling" url extension every 10 second, the server gets the information from Arduino, and send reply with the current temperature in JSON. The polling mechnism is implemented in the Phone side in main.c, using an AppTimer, which creates a background thread that sleeps for 10000 miliseconds and then invokes a callback function in which the app_timer_register will be called again or quit the polling mode. Specifically, in main.c, in the up_long_click_handler function (line 205), we check the mode3 to see if it is in polling mode (mode3 = 1) or not. If it is polling mode, then it will call app_timer_register() funtion, so after 10 seconds, it will call timer_callback() (line 42). In the timer_callback(), it will check if it still in the polling mode. If it is, it will call the app_timer_register() again; if it is not, it will call resume() and it quit the polling mode and come back to the normal mode.