JUBITO: An interoperable platform for wellness

Yiannis Ambeliotis
Independent Software Developer
Athens, Greece
john.ambeliotis@gmail.com

Kostas Giokas (kgiokas), Dimitris Koutsouris (dkoutsou)
Biomedical Engineering Laboratory
ICCS, NTUA
Athens, Greece
@biomed.ntua.gr

***Abstract*—Jubito is a web server for connecting software and hardware components based on the open source jaNET framework [1]. Jubito provides you with an intuitive platform that enables you to interconnect hardware and software components, as well as publish your projects online. In this paper we will give the reader some examples of Jubito use with the the Arduino platform.**

Keywords-component; open source; interconnected software and hardware; wellness; rfid

I. INTRODUCTION *(HEADING 1)*

Assistive domotics is a field where home automation allows the elderly and disabled to stay comfortable at home but can also offer possibilities such as increased safety (access control, intrusion detection), robotics, etc. jaNET offers multiple functions and API commands that Jubito can use in order to interact with multiple vendor hardware (especially open source, e.g. Arduino, RaspberryPi, etc.). Using Jubito one can take control of jaNET via a mobile phone (our application is currently supported by Android KitKat or newer). The use of Jubito is to provide an intuitive user interface for the advance user to control sensors via jaNET. The advantage of our tools lies in their ease of use and the usability that is offered by the Android operating system so that any user with some programming skills can very quickly set up his controlled environment.

II. EXAMPLES OF USE AND IMPORTANT TOOLS

In this section we will describe some of the most useful examples of use for Jubito and jaNET that can very easily be integrated and used in a smart home environment. Also our most importa

A. Arduino temperature and humidity using DHT11/DHT22 modules

Below there is a tutorial on how to measure temperature and humidity of a particular room in the house. The hardware requirements for this example is an Arduino, a DHT11 or DHT22 module (temperature and humidity sensor module). The DHT library is also a requirement. We should use the following command to choose the type of sensor

#define DHTTYPE DHT11 or (1)

#define DHTTYPE DHT22 (2)

We should then launch the command to *get* the temperature (dhttemp) and we do that by adding a new launcher specifying an action.

Should we need to test the new launchers we can do so via the mobile app using the commands below:

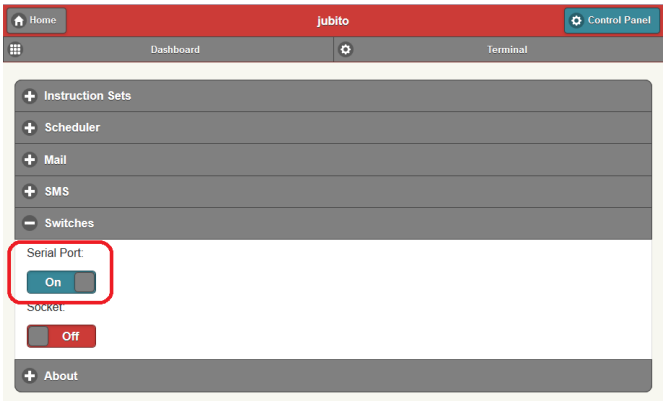

Figure 1. Connecting Arduino via seriar port UI

We can now add the launchers to our dashboard after filling in some mandatory and some optional fields. By editing html files we can enable notifications in the Dashboard's home screen.

The code we need to add contains the getData function and is as below:

```
$.get('_.html?cmd=judo serial send dhttemp', function (data) {
  $('div#tempc').html(data.replace('<br />', '') + '&deg;C');
}, html');

$.get('_.html?cmd=judo sleep 1500; judo serial send hum', function (data) {
  $('div#humid').html(data.replace('<br />', '') + '%');
}, 'html');
```

MOBIHEALTH 2014, November 03-05, Athens, Greece

DOI 10.4108/icst.mobihealth.2014.258062

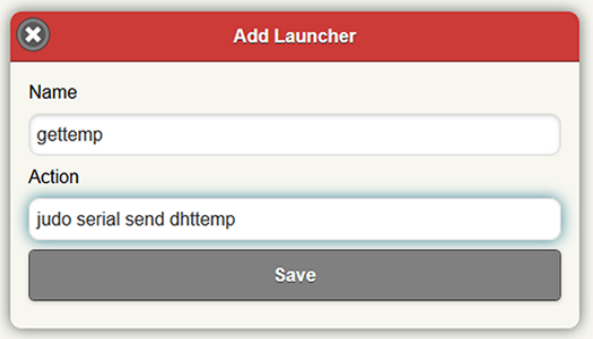

Figure 2. Add Launcher command

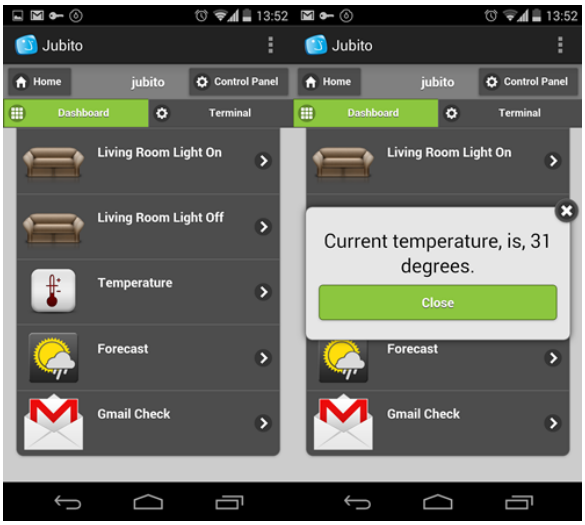

Figure 3. The Dashboard with notifications

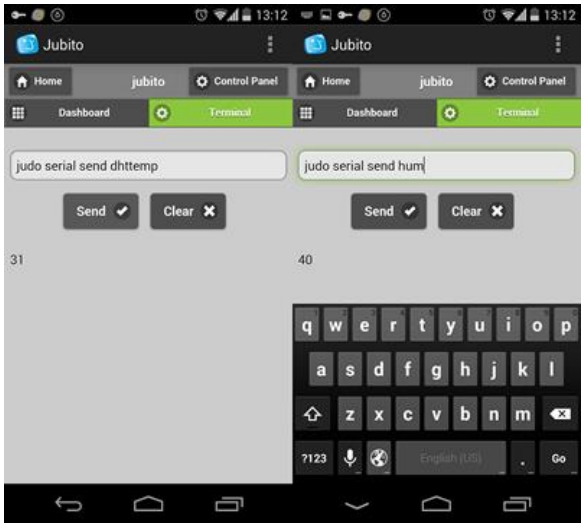

Figure 4. Testing the launchers via mobile app

B. *Posting actions via QRCode*

jaNET supports RESTful [2] commands. By using web services we can post a command to our webserver either over the web or use a common QR Code maker. By typing our web server‘s hostname together with the REST style command we can execute a corresponding action.

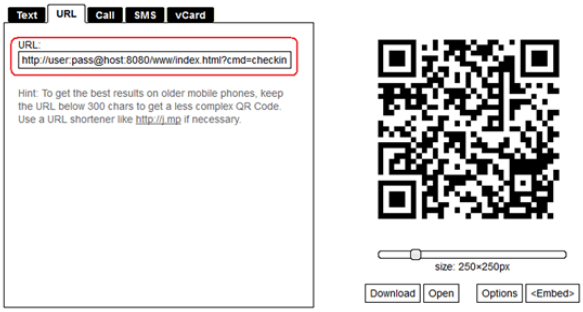

Figure 5. RESTful commands via QRCode

C. *The Scheduler*

One very important tool of Jubito is the scheduler. The scheduler can take the form of a simple notification, a hierarchy of tasks or even a daemon. The scheduler has several parameters such Name (a schedule handler), Date (when a recurring action should stop and be removed from the schedule when completed), Periodicity (actions that can take place daily, on weekends, on working days at a specific hour, etc.) and Action (can be a single sentence or an instruction set call)

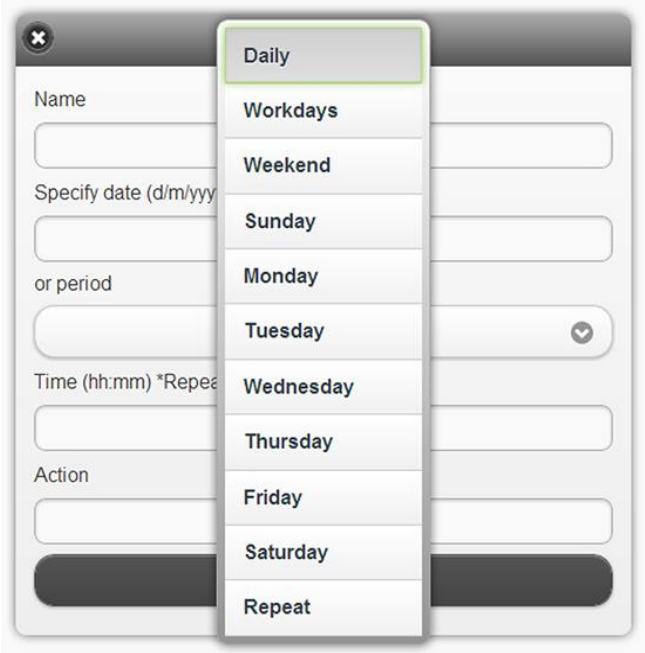

Figure 6. The scheduler

The scheduler can fulfil a number of scenarios like, wakeup call (ask me to wake up work days at 7am, tell me the weather conditions, turn on the room light), automation control (turn off garden lights daily at 11pm) or check-out after 10am when I'm living for work etc.

III. EVALUATION FUNCTION AND ITS USE

Almost all Jubito/jaNET functions can be extended by using the evaluation function. When we need to apply logic we use

Identify applicable sponsor/s here. *(sponsors)*

the evalBool statement. The evalBool function returns one or two states, depending on the evaluation of an expression. The parameters we can use are: Expression (required, Boolean, it is the expression we want to evaluate), True Part (Instruction Set, it is called when Expression evaluates to True), False Part (Instruction Set, it is called when Expression evaluates to False. Below is an example of the syntax:

{ evalBool (expression); <true part>; <false part>; } (3)

Below is a list of relational and conditional operators:

equal to: ==
i.e. (x == y)
i.e. ("x" == "y")
* String comparison require quotes
not equal to: !=
i.e. (x != y)
i.e. ("x" != "y")
* String comparison require quotes
greater than: >
i.e. (x > y)
less than: <
i.e. (x < y)
greater than or equal to: >=
i.e. (x >= y)
less than or equal to: <=
i.e. (x <= y)

conditional AND: &&
i.e. (x > 10 && x < 20)
conditional OR: ||
i.e. (x == 10 || x == 20)

contains (Supported on ver. 0.2.1 of libJanet. To check your version, go to terminal tab and type %about%)
i.e. (this is a test ~> is a)
i.e. (%todayconditions% ~> Sunny)

An example would be:

{ evalBool(*getTemp > 30); foo1; foo2; }

Which translates as:

"If temperature is greater than 30 degrees then turn on a fun (foo1) else turn it off (foo2)"

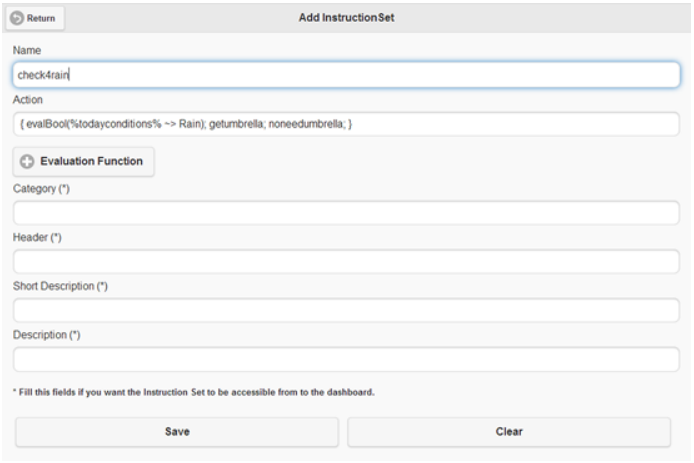

Figure 7. Evaluation function for the checkrain instruction

IV. CONCLUSIONS

Looking at the above examples we can understand the power and simplicity of the jaNET framework for manipulating and integrating software and hardware sources and components. Jubito is an intuitive user interface that can be used with or without a microcontroller (such as Arduino) and both can be used to link different projects and technologies and manage them from a central point (mobile device).
Moreover we offer these tools free of charge to the developer's community. We have set up a Google group as well as a code contribution channel [3] for developers

++++

REFERENCES

[1] Project jaNET – An open source framework for devices control, https://sites.google.com/site/projectjanet

[2] Richardson, L., & Ruby, S. (2008). *RESTful web services*. " O'Reilly Media, Inc.".

[3] Google Code group https://code.google.com/p/project-janet/