# Red Pitaya Visual Programming Setup

I set up my Red Pitaya for Visual Programming before attempting to connect to it with GNU Radio and Vivado. Visual programming is a web-based approach to programming that simplifies coding down to drag-and-drop blocks. It's intended to make Red Pitaya more accessible in educational situations.

In order to use Red Pitaya visual programming, there is a license that you have to buy. At the time of this writing, the cost was 5 euros per month. You subscribe through the Red Pitaya store. I did this and it was fast and easy, just like all of the other Red Pitaya store experiences have been so far.

Visual Programming on the Red Pitaya requires internet access. Internet access is also required when upgrading the Red Pitaya operating system, installing applications from the marketplace, and unlocking licenses for applications from the marketplace.

Traditional coding on the Red Pitaya, like using itwith GNU Radio and/or Vivado, does not require internet access.

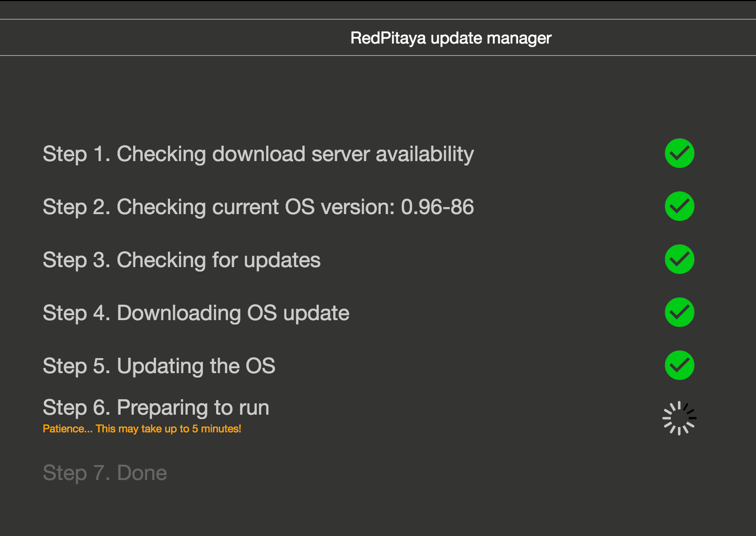
Basic setup for the Red Pitaya needs to be accomplished before setting up Visual Programming.

The major tasks are unboxing, obtaining a proper power supply, updating the operating system, and being able to connect to it by using a web browser. Documentation on how to accomplish the above setup can be found at:  
<http://redpitaya.readthedocs.io/en/latest/doc/quickStart/needs.html> You will need to log in to your Red Pitaya account. You will then add your board to your account. When successful, the board should show up in the My Equipment page. For me, the link was   
<https://store.redpitaya.com/myequipment/list/>

And the page (after setting things up) looked like this:



Here’s what the task of updating the operating system (OS) looks like.



This is updating the OS after you’ve prepared the SD card and successfully set up the Red Pitaya. Depending on when you bought your Red Pitaya, you may have to set up the SD card yourself, or it might have the operating system already installed. Either way, regular updates to the OS are made and should be installed. The update OS function can be found by clicking the IP address under LAN IP and selecting the OS update menu option.

Completely setting up the Red Pitaya for Visual Programming is not hard, but there are some fiddly bits. All of the steps were accurate at the time of this writing, but things may change in the future.

Red Pitaya has a video tutorial for blinking LEDs from November 2015. You can view it here:

<https://www.youtube.com/watch?v=V4ZSB8oetDQ>

We’re going to duplicate the steps, expand on a few of those steps, and show updated screenshots.

Go to the My Equipment page.

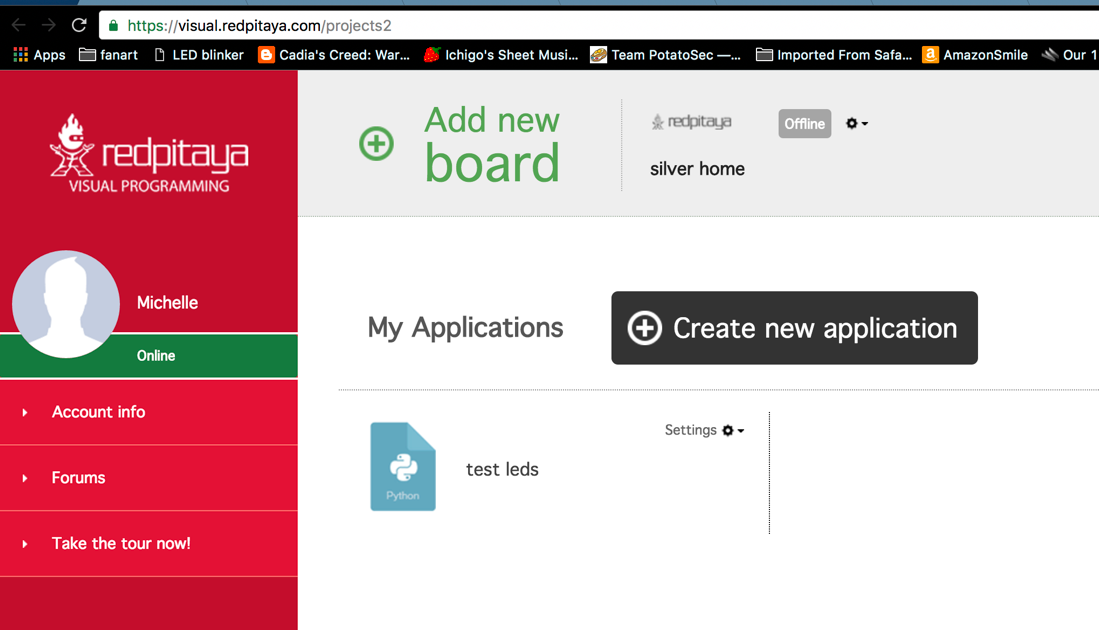


Click the link Visit Visual programming [HERE](https://visual.redpitaya.com/).

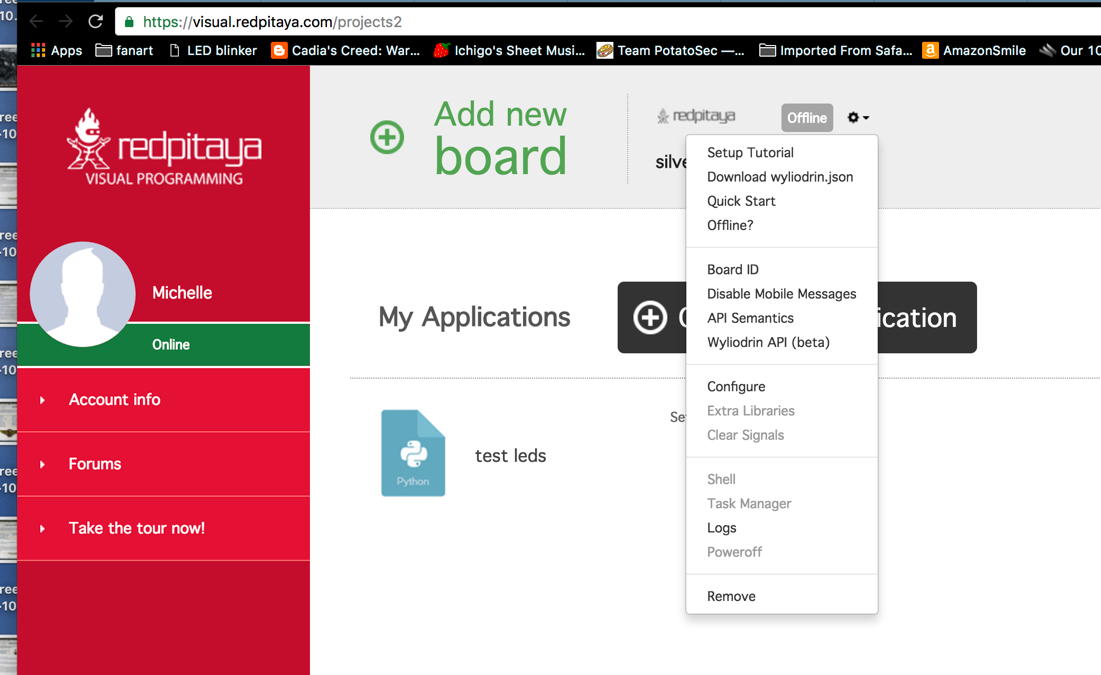
This launched the page

<https://visual.redpitaya.com/>

You should see a page that looks something like this. I needed to add my board on this page to get it to show up, even though it was successfully listed on the My Equipment page.



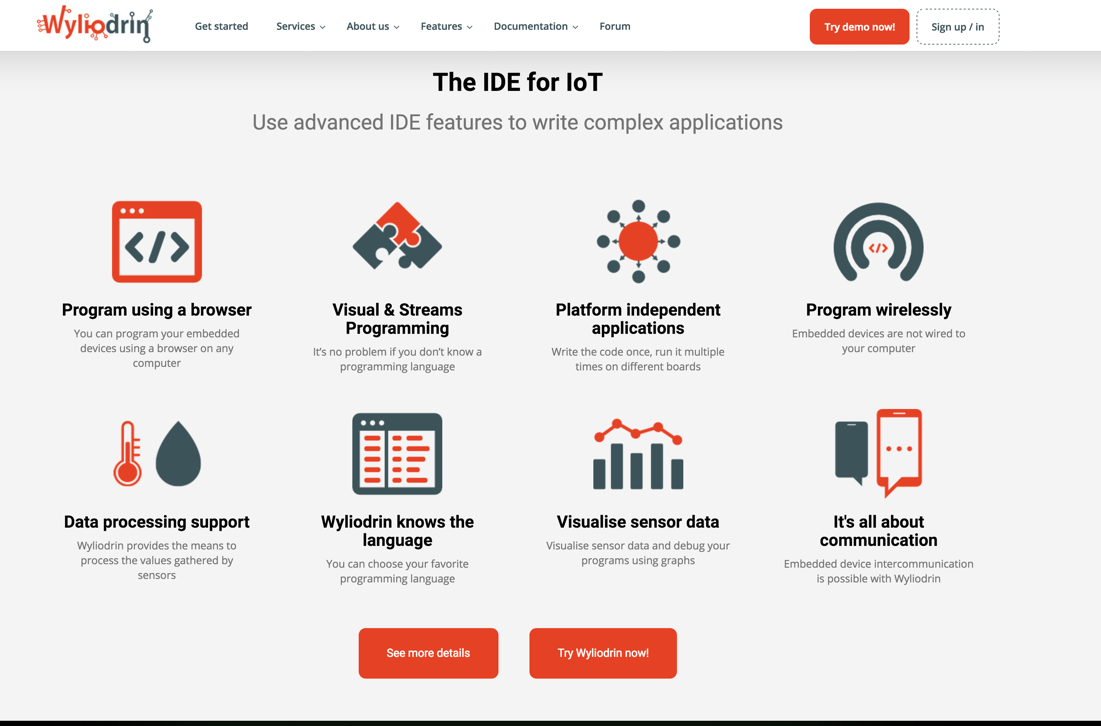
There is a gear menu at the top that we need to use for several important functions.



After you add your board, you are going to have to configure, get extra libraries, and download the wyliodrin.json file. This JSON file must be put on your SD card.

What is wyliodrin? It's an integrated development environment intended to serve the Internet of Things marketplace.

<https://www.wyliodrin.com/>



Red Pitaya uses the wyliodrin IDE to provide the Visual Programming platform.

Interestingly, Red Pitaya is not listed on the wyliodrin site as one of the boards the downloaded version of the IDE supports. However, wyliodrin does have a video tutorial that shows wyliodrin visual programming in action.

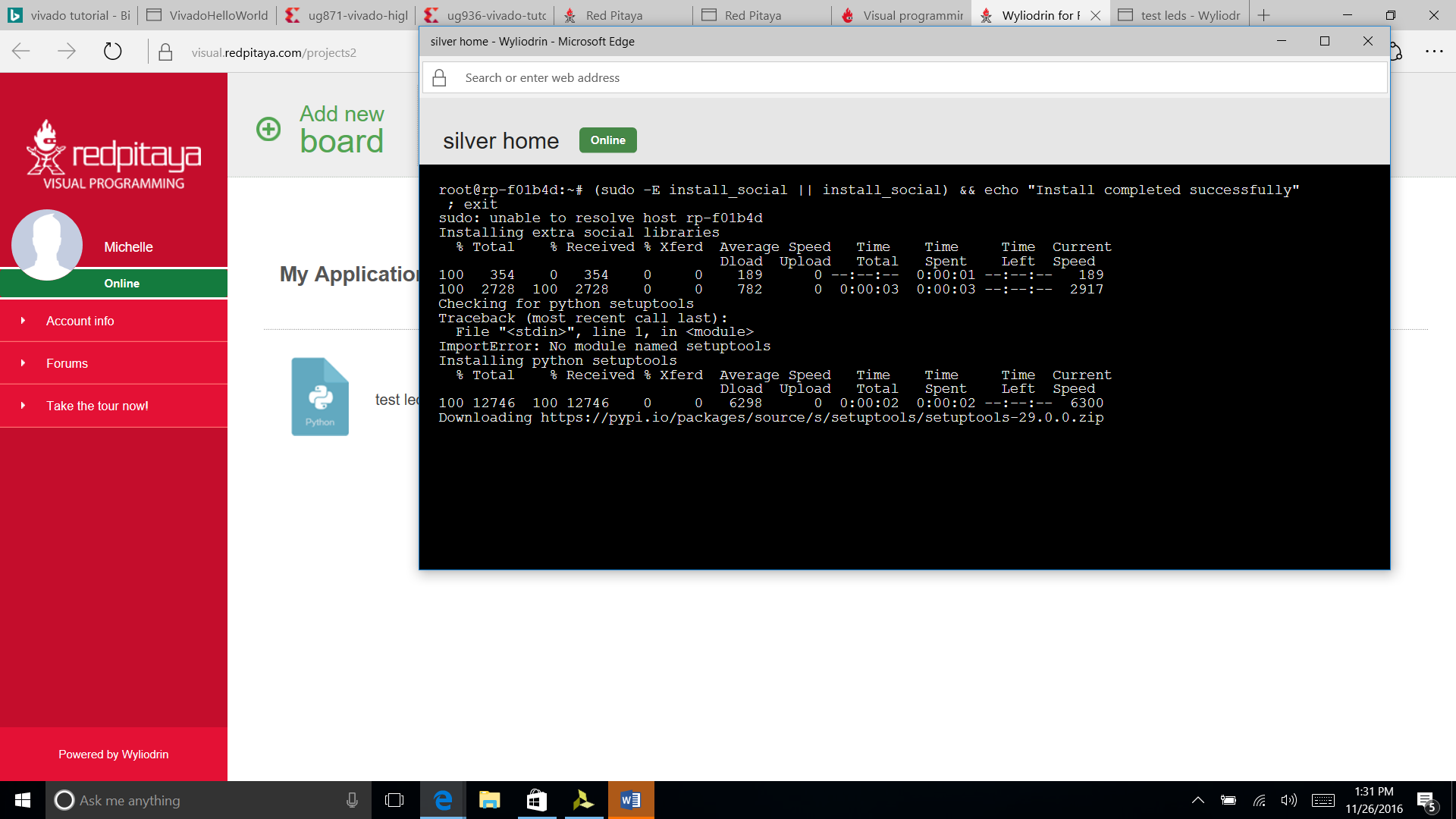
<https://projects.wyliodrin.com/wiki/video_tutorials/visual>

The example video tutorial is a radio project on a Raspberry Pi. It's a radio with a VU meter. Not a bad "hello world" project. In wyliodrin, the blocks (think of them as Legos) that form the visual programming are compiled into Python code. There's a pane for viewing this code within the IDE.

JSON stands for JavaScript Object Notation. JSON files are lists of attributes and value pairs. The lists can be gathered up into JSON objects and/or expressed as JSON arrays. Arrays and objects can be nested.

JSON files are fetched and key-value data extracted. JSON objects are used to create very powerful connected experiences. It's an open standard. Almost all of the applications for the Red Pitaya are enabled by JSON. Your web browser is the front-end, and your Red Pitaya is the back-end. Data between the Red Pitaya and the web browser is passed back and forth in JSON format. This is essentially the same strategy as Phase 4 Ground radios will use.

Here's what it looks like when you install extra libraries.

  
  
The Setup Tutorial link in the gear menu has all the steps we've discussed so far and a direct link to the JSON file.

Without this JSON file, you can click on the visual programming link all day long and nothing productive will happen, but it won't give you a useful error message. Ask me how I know.

In the process of handling the JSON file on a Windows machine, I found out about brackets, which is a "modern open source text editor that understands web design". I found it useful. It can be found here:

<http://brackets.io/>

The JSON file is human readable and as of mid-December 2016 was (for me) looked something like this.

{

"jid": "[abraxas3d\_silver\_home@visual.redpitaya.com](mailto:abraxas3d_silver_home@visual.redpitaya.com)",

"password": "<REMOVED>",

"socketpassword": "<ALSO\_REMOVED>",

"owner": "[abraxas3d@visual.redpitaya.com](mailto:abraxas3d@visual.redpitaya.com)",

"timeout": 2000,

"maxBuffer": 200,

"firewall": false,

"ping": 50,

"ssid": "",

"scan\_ssid": 1,

"psk": ""

}

This file controls relationships between your Red Pitaya board and several other entities.

The board will send status message to Red Pitaya servers for "improvement purposes". This can be turned off by adding a line to the wliodrin.json file.

Add

"privacy":true

Somewhere within the curly brackets.

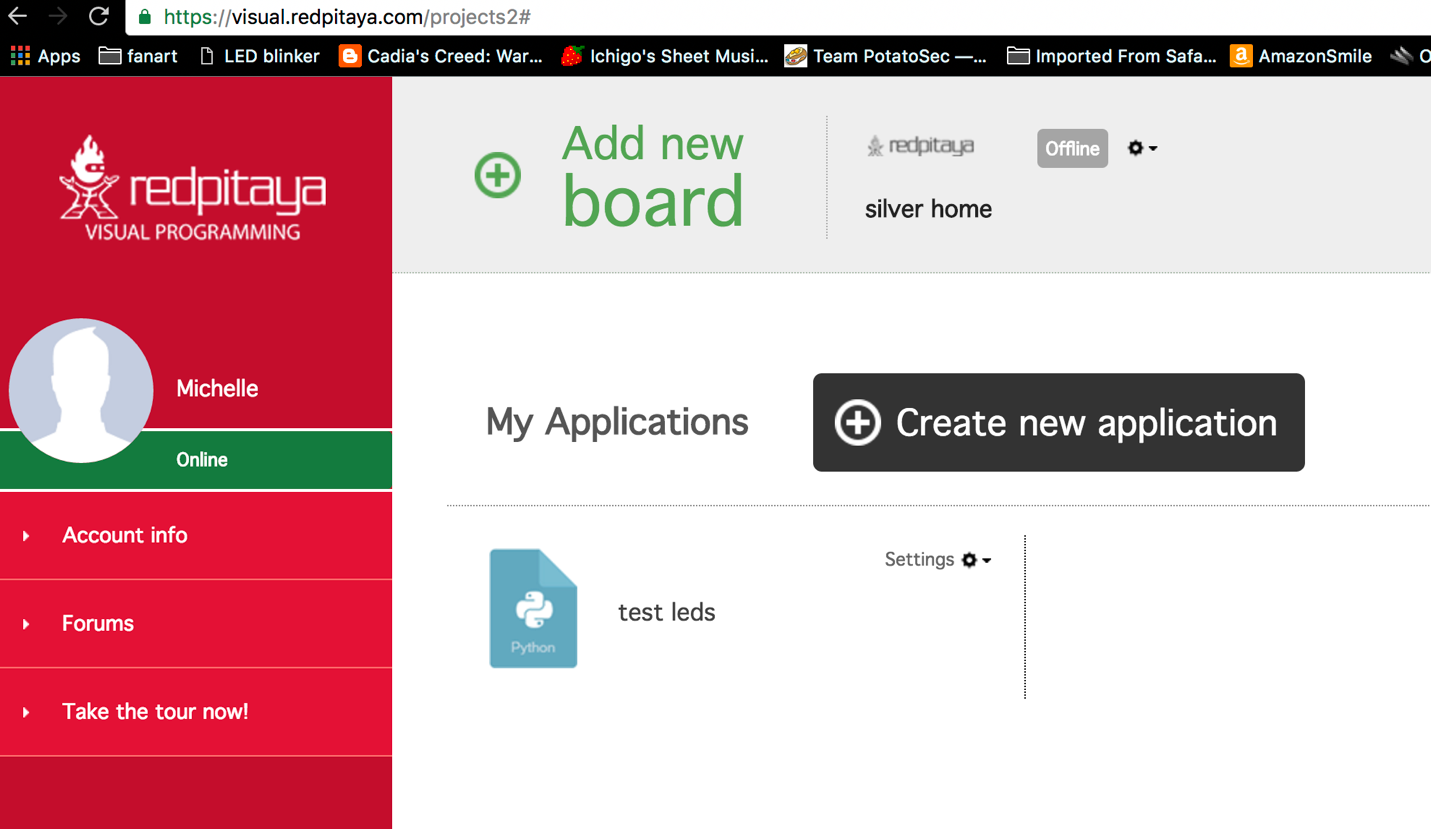
If only online privacy were always this easy everwhere.

What do the other lines of this JSON file do?

jid and password are XMPP connection credentials. Wyliodrin servers use the listrophe library. The jid is used to send messages to the Red Pitaya. The board ID can be found in the gear menu option.

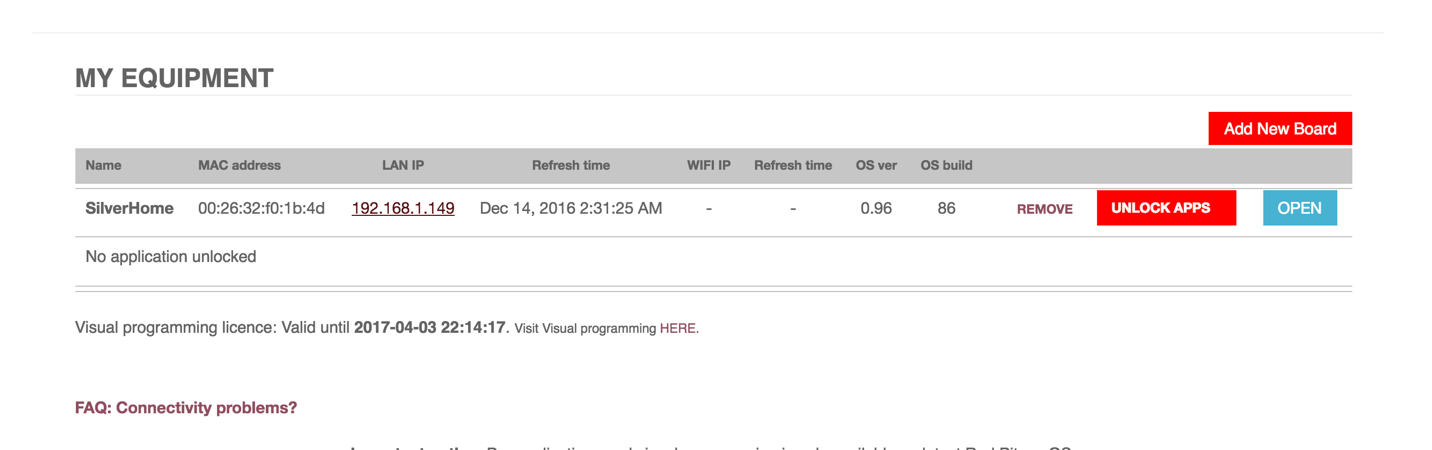
The other pairs of values seem to support common server-client communications functions. If I find out any additional functions that can be enabled through this file, I'll include them in a future revision. If you know of any, then please feel free to update this file or send me edits.

After the wyliodrin.json file is on the SD card, and after everything is configured, and after selecting “visual programming”, you might see something like the following screen



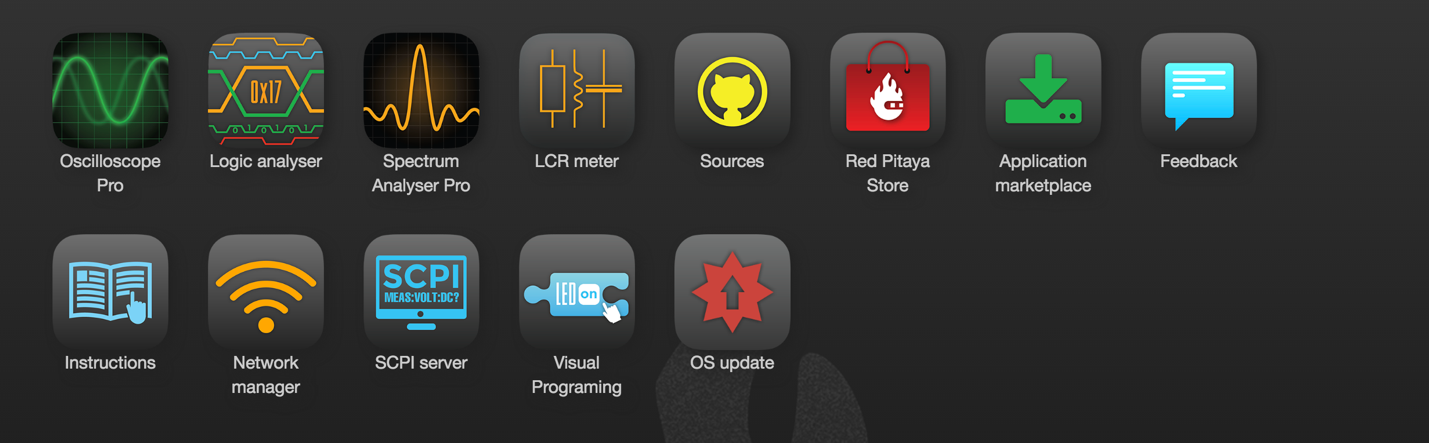
Notice in the upper middle-right part of the screen is a greyed-out box that says Offline, even though the green status indicator on the right-hand red-background menu says Online?

We need to start the visual programming server. Go back to the My Equipment screen.

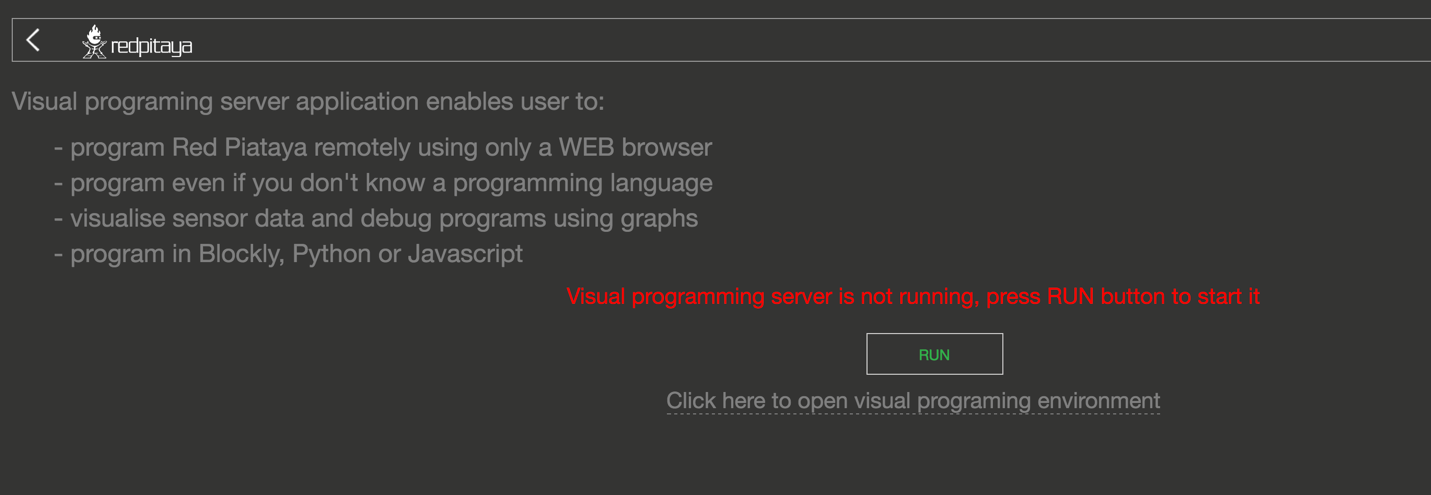


Where it says LAN IP, there is a link to the IP address of the Red Pitaya. If everything else has gone well, then the Red Pitaya is indeed sitting on your LAN, waiting to be told what amazing things it’s going to be doing.

Click on that IP address. You should see the following screen.

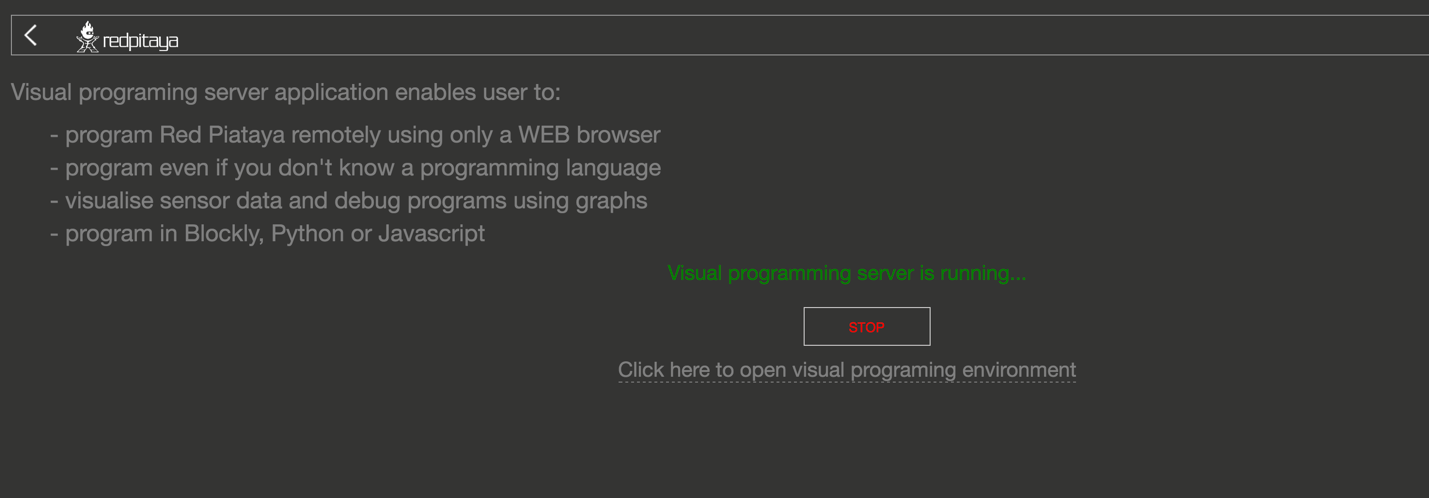


Click on Visual Programming. You should see the following screen.



See where it says “Visual programming server is not running, press RUN button to start it”?

Press the button.



OK that’s promising!

Back to the visual programming tab.

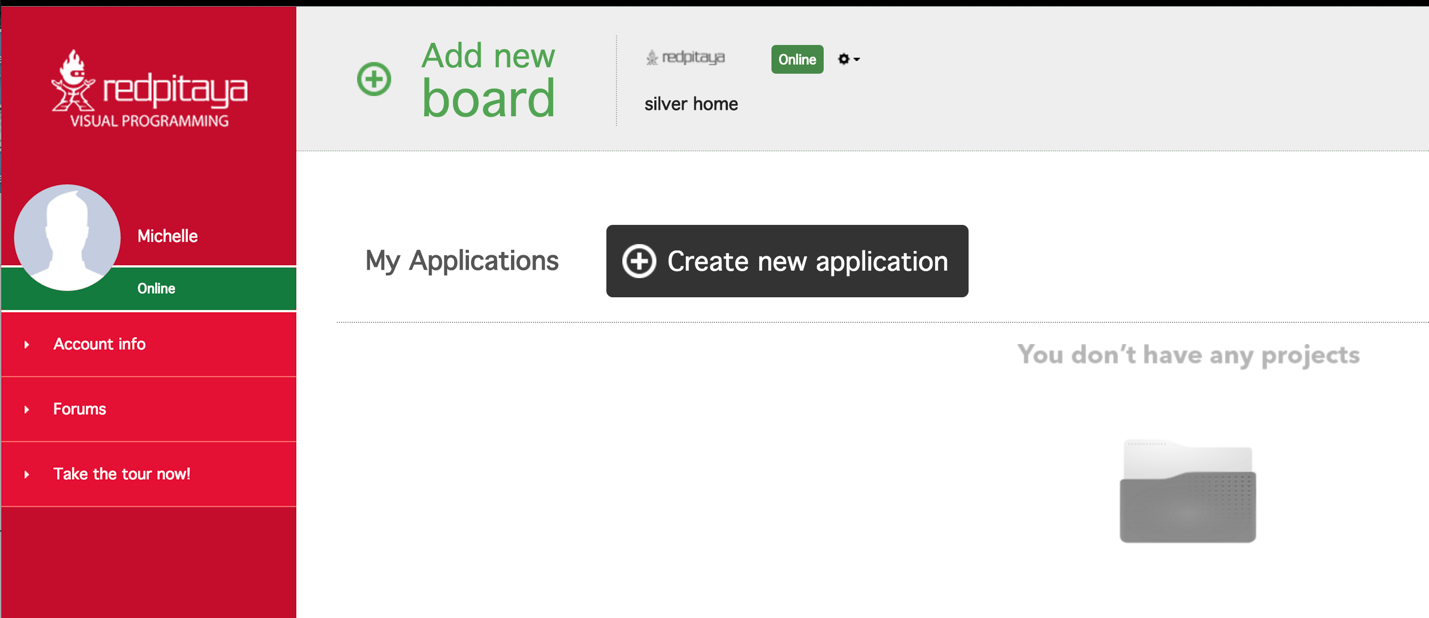


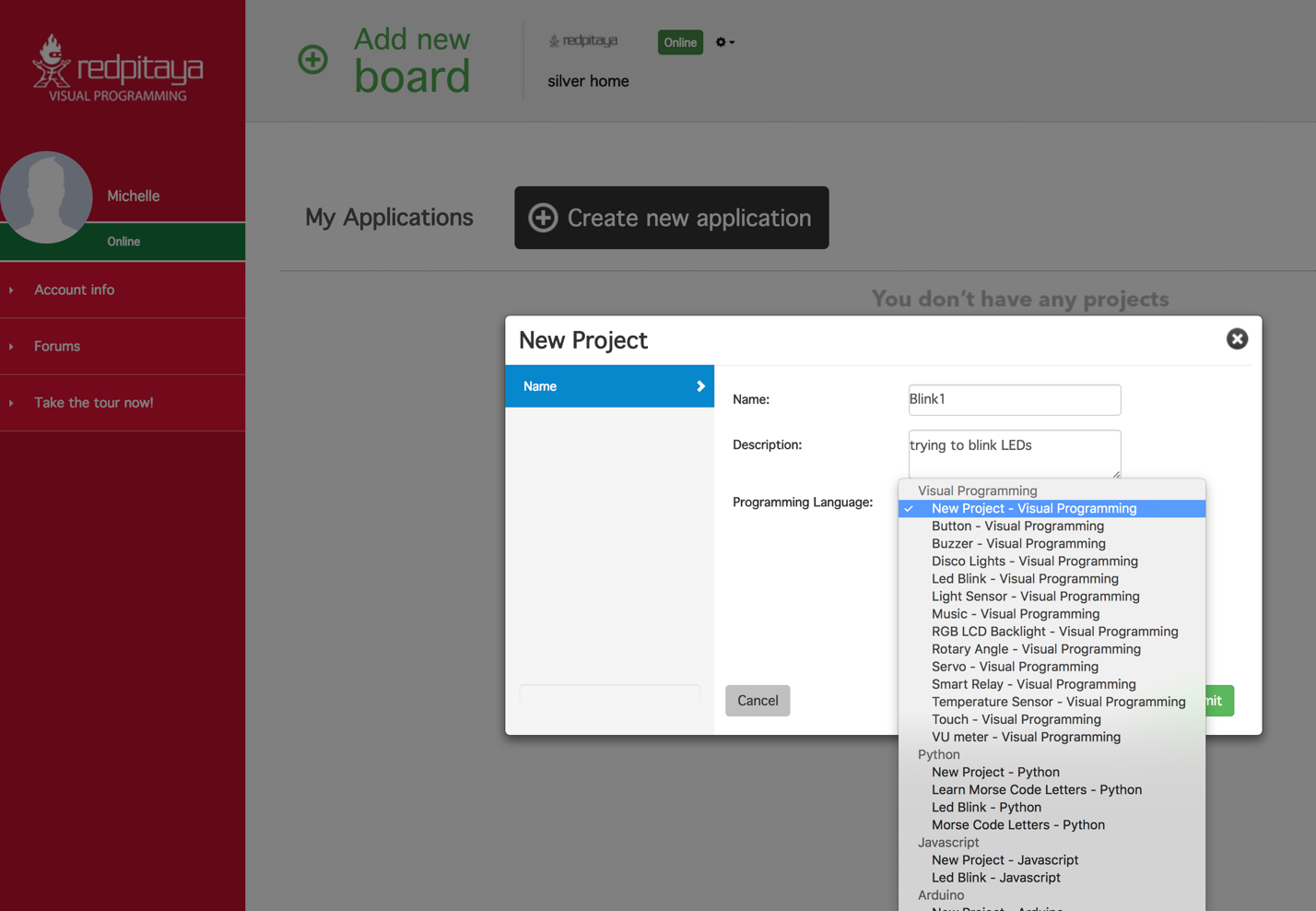
Improvement! The gray Offline box has turned to green Online. This is the point where we can get some traction from the published tutorials from wyliodrin and Red Pitaya.

# Using Visual Programming

Here's a simple LED blinking experiment within Visual Programming for the Red Pitaya.

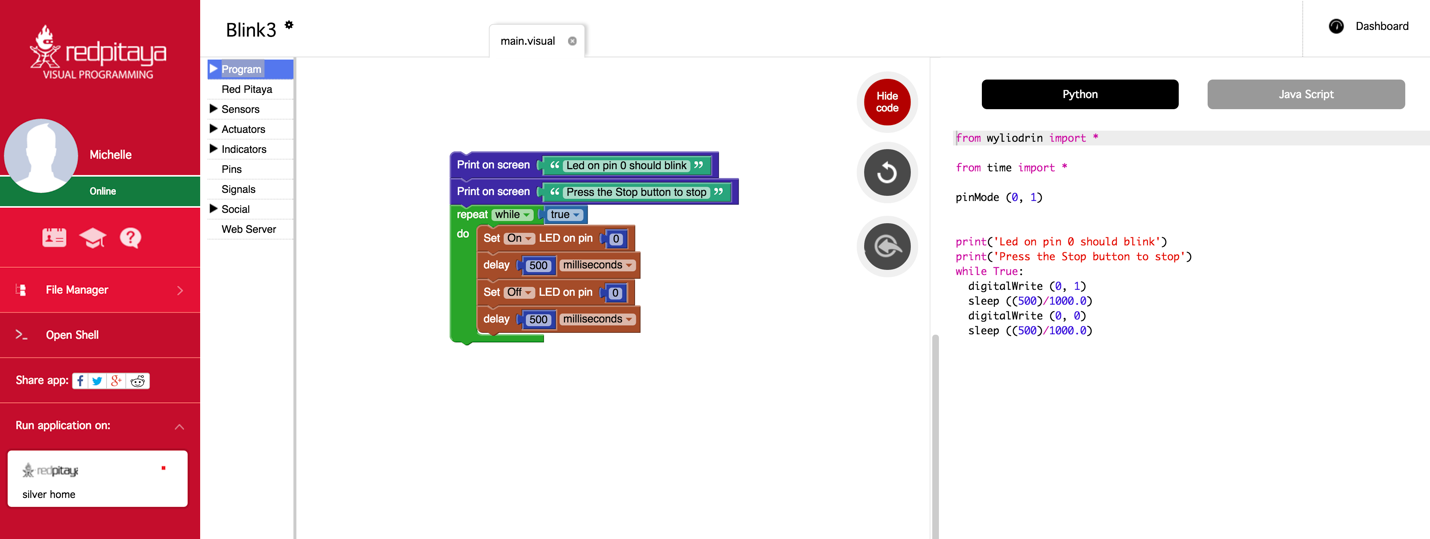
First, I created a new application. Select Create New Application and fill out the form.





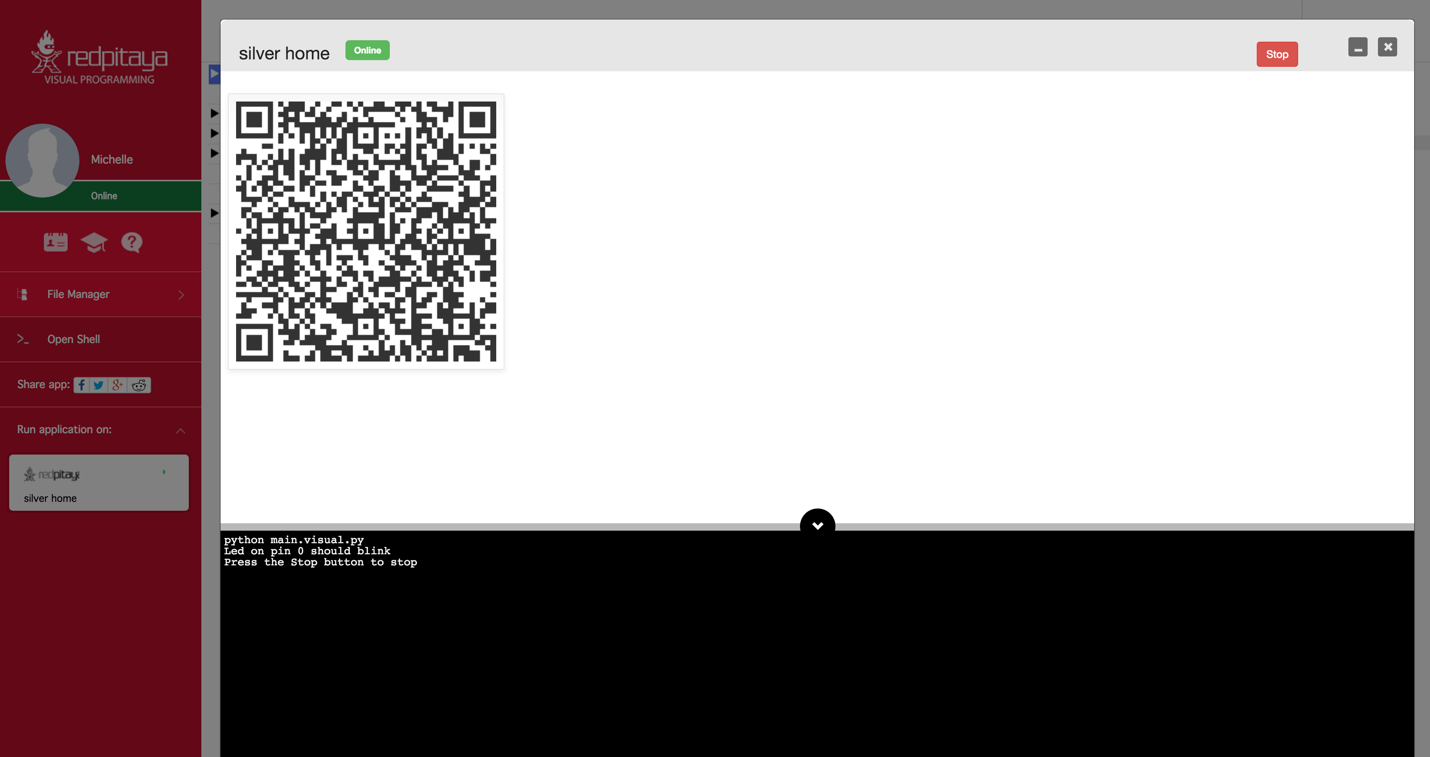
Well, there are a lot of choices in Programming Language. I went with New Project - Visual Programming. Look at all the template projects in the Visual Programming section! If you pick one, then you can see that they are the stock template projects from wyliodrin.

I created another project and opened up Led Blink. Here’s what it looks like when you click Open Code button to show the Python pane.



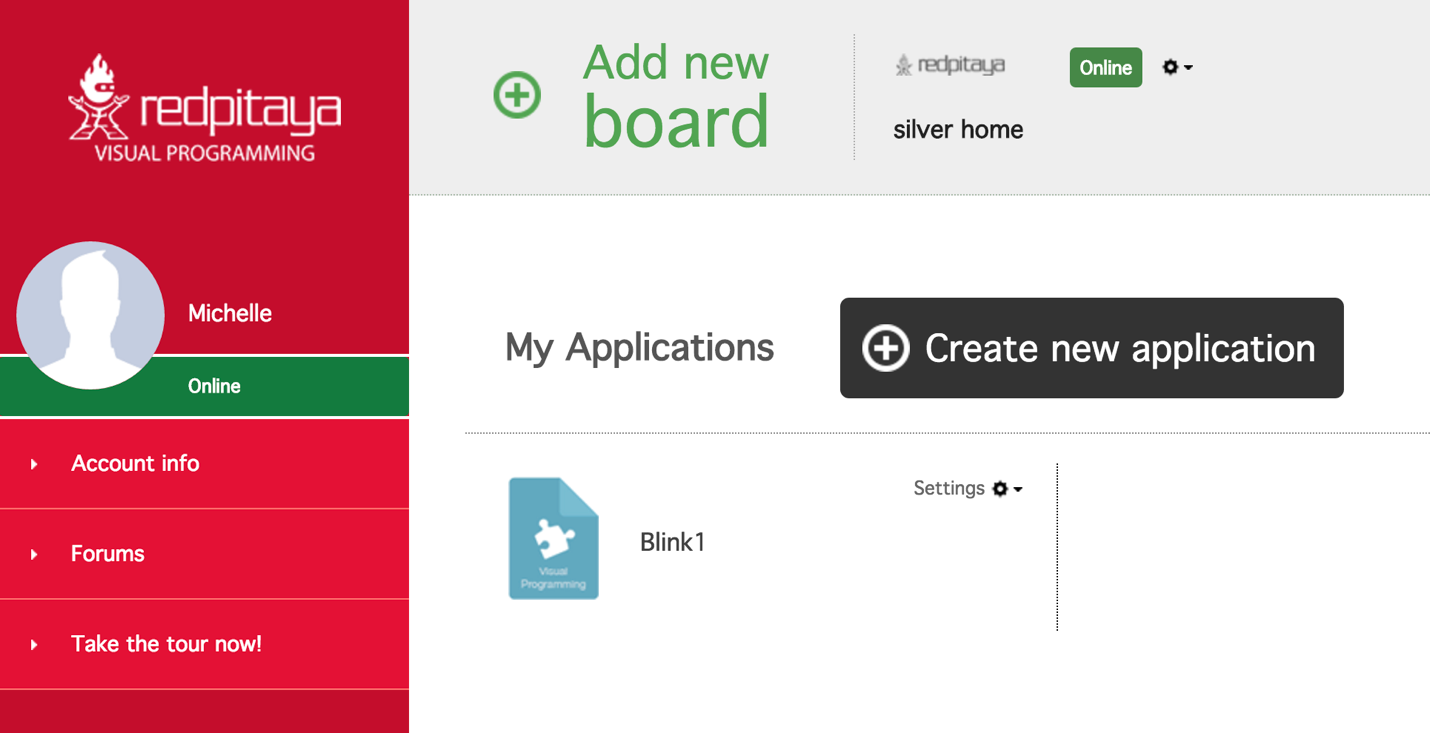
In the lower left corner is Run application on: and my Red Pitaya (silver home) shows up. I clicked this button.

After a short verification window appears and declares success, we get the following screen.

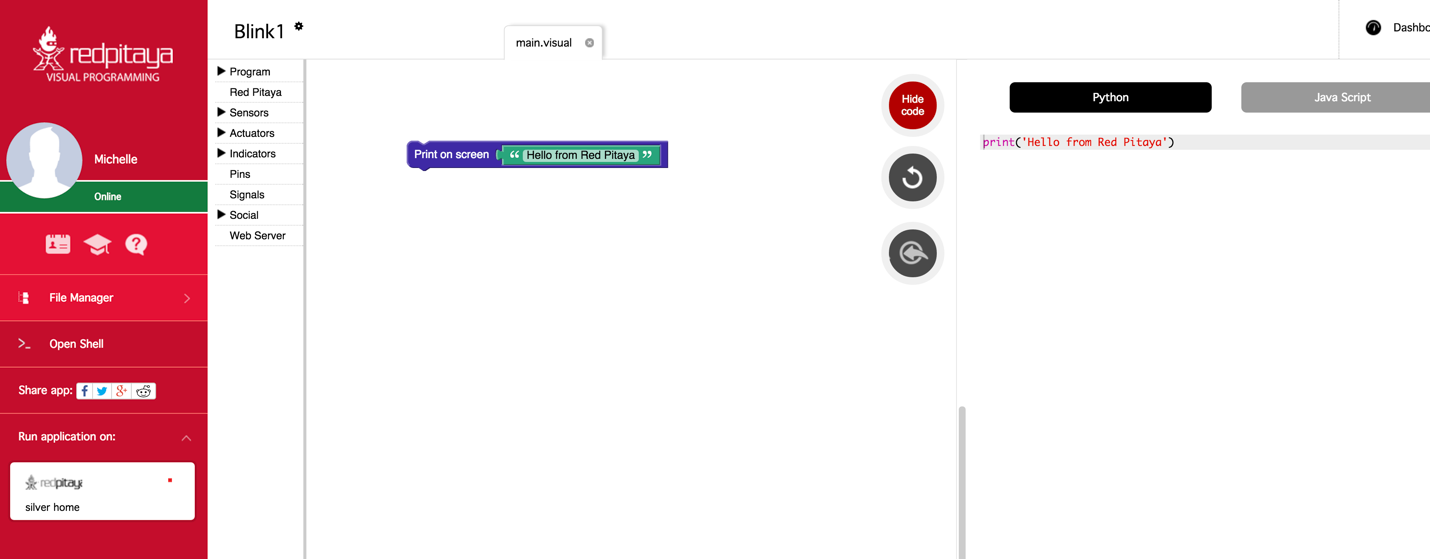


The 2D code is the board ID. You can see this same board ID in the gear menu. The stop button is in the upper right. The console declares that pin 0 should blink. I don’t see anything blinking, but I haven’t added an LED to the board or anything like that. Clicking stop made it stop claim that it had stopped. I was able to back out to the Visual Programming screen and the Red Pitaya was still online. Good progress!

When you Create new application, and select New Project – Visual Programming, you get a new project with a single line of code in it to start you off.



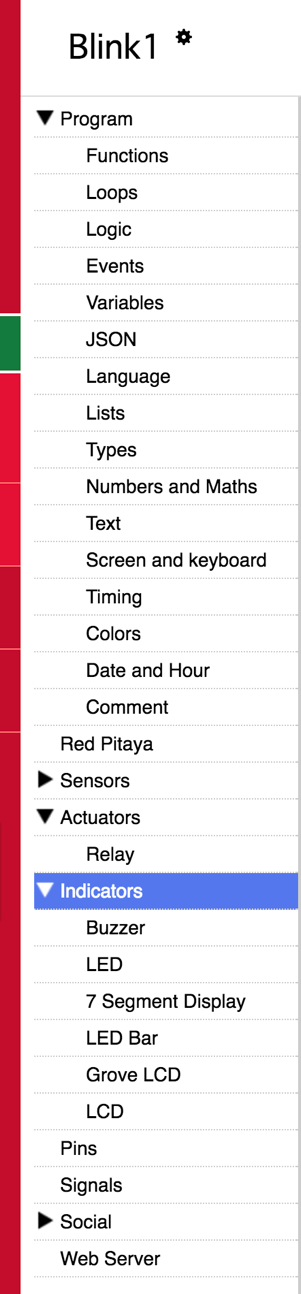
Here I created Blink1 and opened it, then clicked Show code to reveal the Python. You can also work in Java Script, but I prefer Python.



Running this code displays the output of print(‘Hello from Red Pitaya’) in the console, as expected.

Now, where do you get those Lego-like blocks that you use to build programs? Immediately to the right of the red menu on the left-hand side is a column of menus, with Program, Red Pitaya, Sensors, Actuators, Pins, Signals, Social, and Web Server.

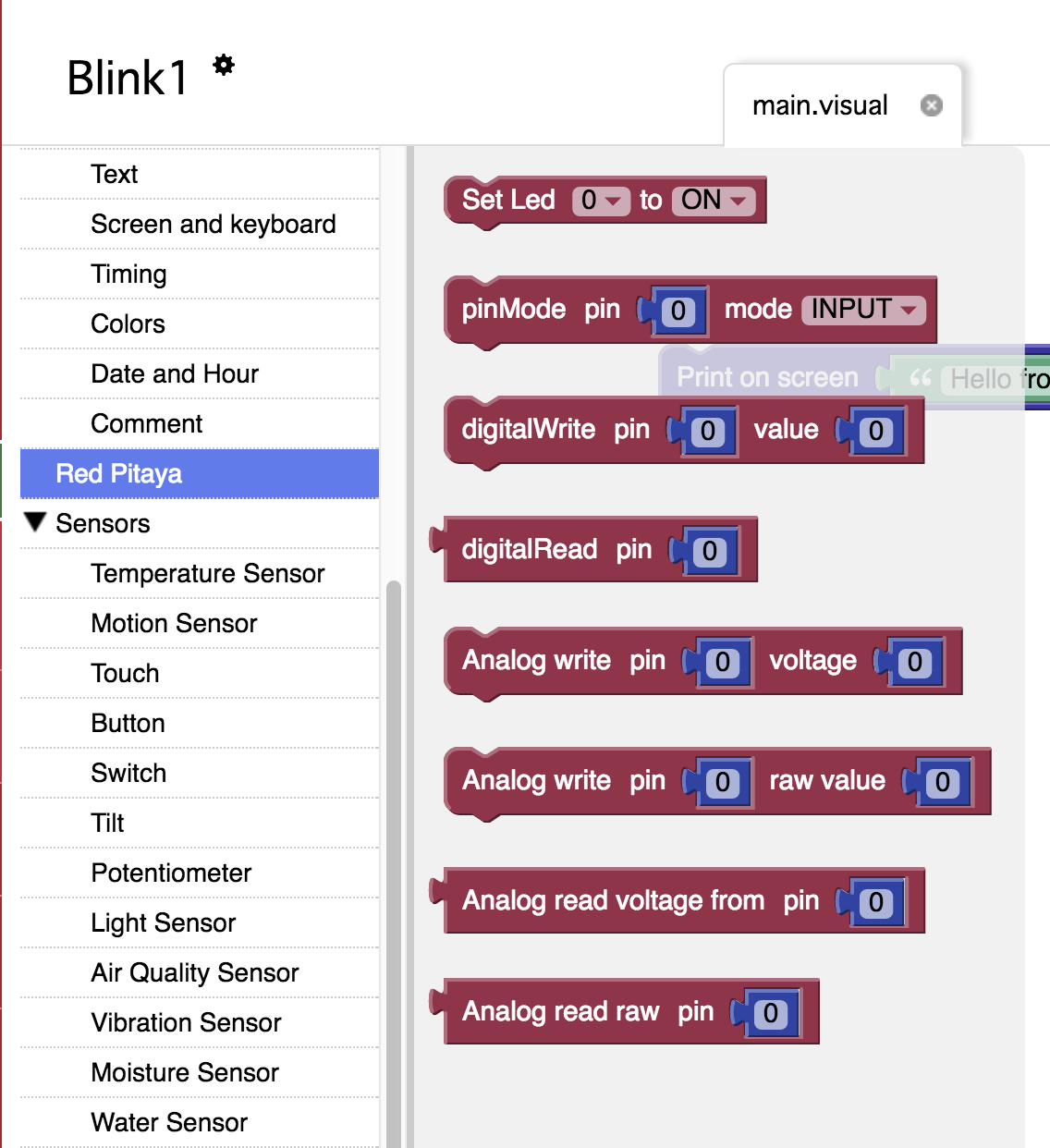
Those drop downs are where the programming blocks are located.



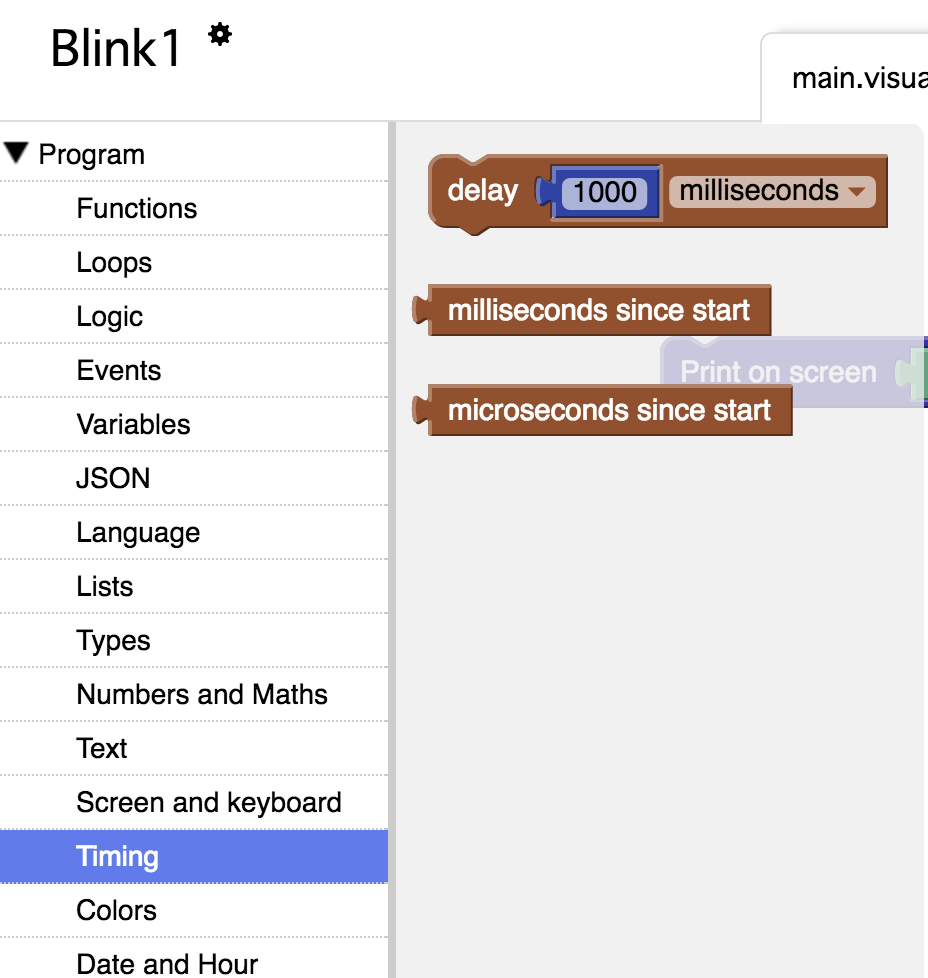
So, for our first program, we’re going to need a few blocks. In thinking about the instructions to flash LEDs, we know we’re doing a simple task over and over. An obvious way to accomplish this is with a Repeat block. This will cause continuous executions of everything which is inside the block. Think of it as if it was a while loop. Inside the Repeat block we are going to have two Set LED blocks. These blocks switch ON and OFF the LEDs. Between switching the LEDs on and off, we need time delays so that we can see the LEDs with our boring old human eyes.

What are the advantages of using Visual Programming?

Ability to create own dashboards with real time graphs, dials, meters, sliders, and buttons • Ability to control the program flow from a PC, smartphone or tablet • Ability to share measurements or send notifications to email or even social networks like Facebook and Twitter • Measures temperature, moisture, alcohol, water level, vibrations, UV light, sound, pressure, air quality detect motion, and other • Controls actuators and indicators like LEDs, displays, motors or relays in order to control high load devices\*The last two features require the use of the Red Pitaya Sensor extension module & sensors • Programming with blocks is a very fun experience, but is also highly instructive and encourages the user to begin thinking subconsciously like a real programmer. All of this is just the beginning of the learning process. This format also enables users to watch and learn what the real programming language code behind the graphical blocks looks like – and how to program using it.



The set LED function is in the Red Pitaya menu (of course).



The “delay some amount of unit time” block is the Program menu under Timing.

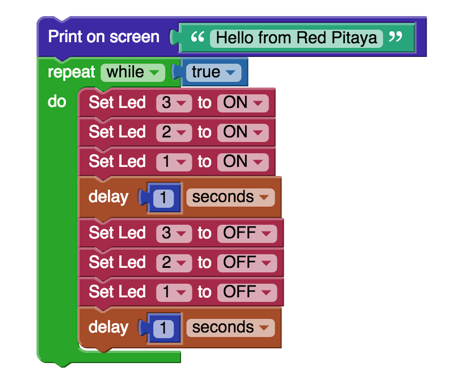


The loop control blocks are in Program under Loops. For this project I picked repeat X times block. At the time of this writing, it was the second block from the top in this section. Here’s what it looks like when you click all the blocks together and configure each of them to make blinking happen.



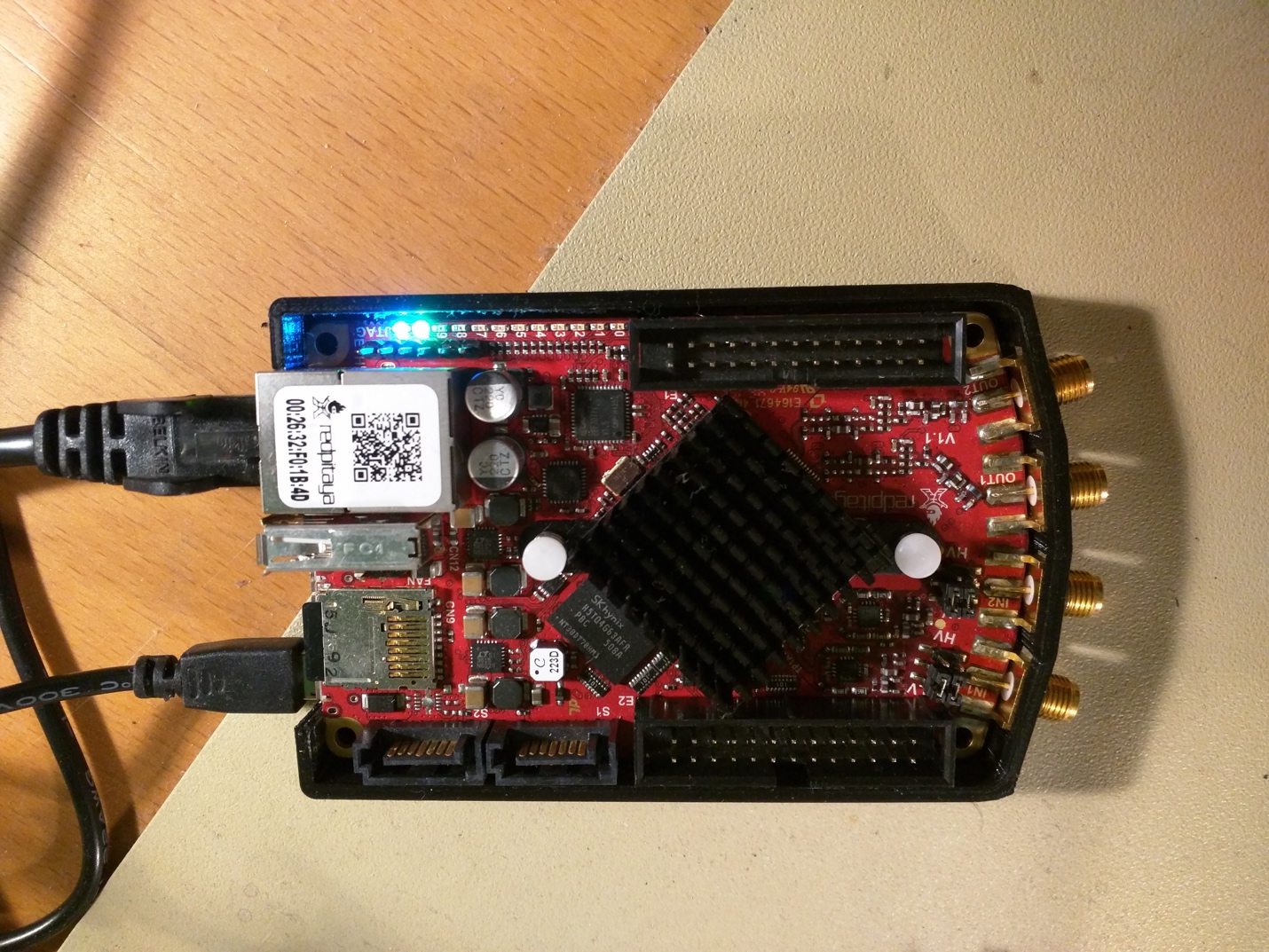
Not being satisfied with one, I decided to turn on and off three!

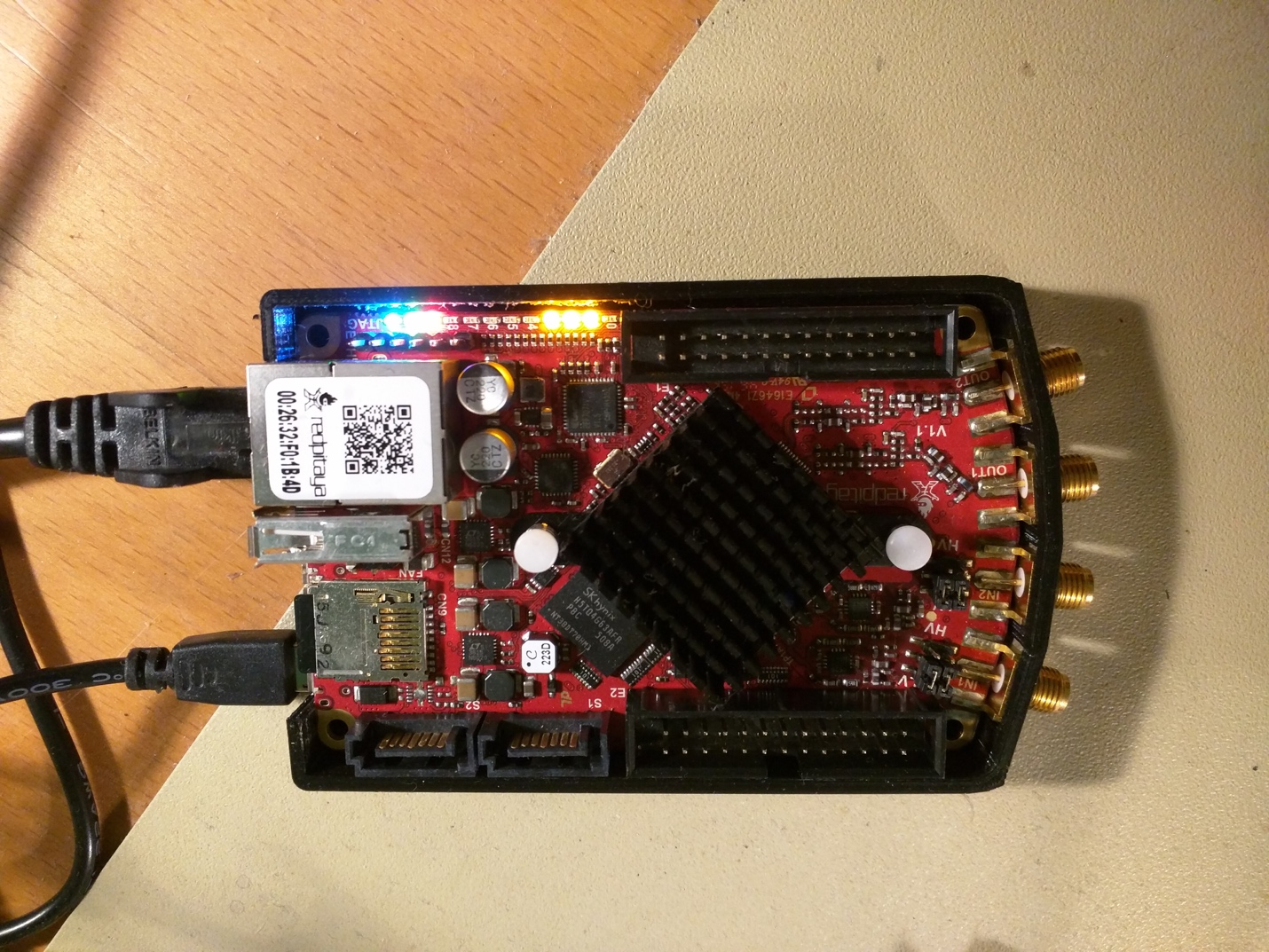
Also, I changed it to run forever.



The drop-down menu in the Set LED block for LED number has the choices 0 through 7. I selected 3, 2, and 1 for the second blinking LED exercise. What do these numbers correspond to?

If you look on your Red Pitaya board, you can see 8 LEDs along the edge next to the Ethernet connector. Each LED corresponds to one of the 0 through 7 LEDs references in the Set LED block. They are built into the Red Pitaya board. The LEDs on the board have their number printed right by them on the silk screen. Can’t get much easier than that.





Here’s the three LEDs on.

There’s more to do with Visual Programming that just turn LEDs on. Grove support provides the horsepower behind the Sensors and Actuators menus, for example. But, Visual Programming is limited. Visual Programming interface provides pre-existing blocks. It is primarily high-level access to a fixed set of registers. There isn’t a way to reprogram the FPGA from Visual Programming. Visual Programming does not have blocks that let you access the Fast Analog outputs and inputs.

Despite the limitations, Visual Programming can produce a wide variety of Internet of Things functionality. Interactive functions under the Social menu include blocks such as mail, Facebook, Twitter, Twilio, and Board Communication. You can set up Web Server functions as well. Leveraging the power of the wyliodrin IDE was a good choice and provides useful functionality. What it doesn’t seem to provide is access to the FPGA.