September 14, 2018

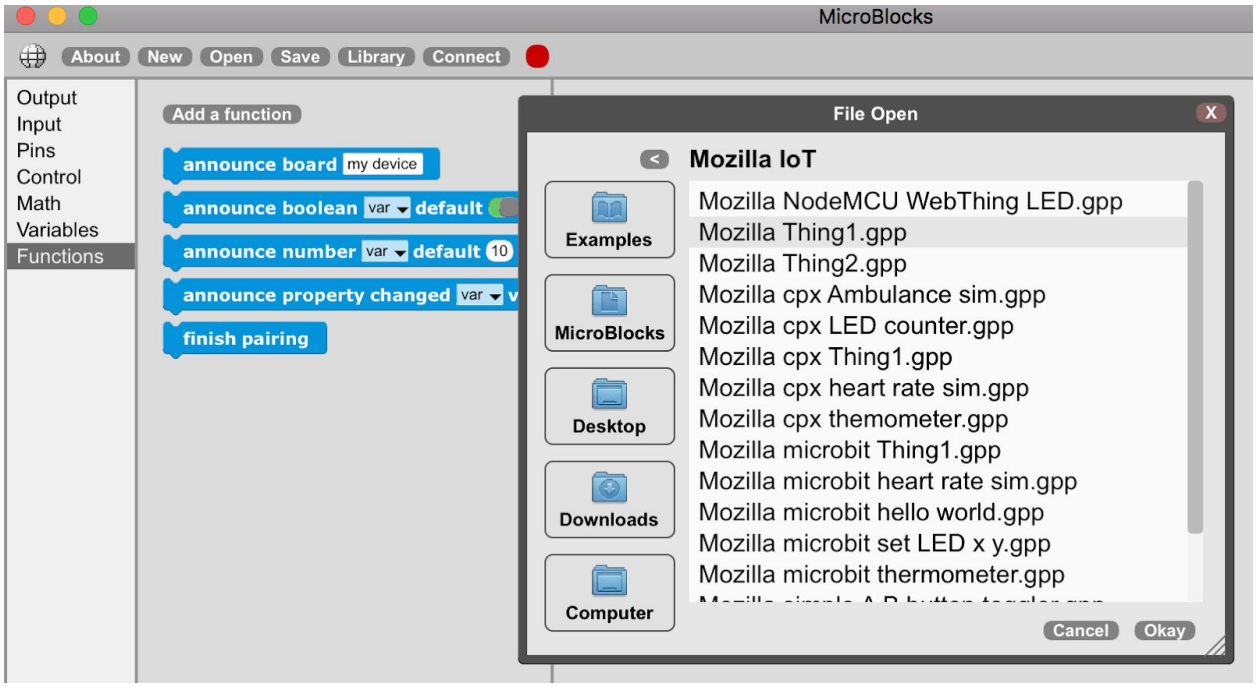
How to Create a MicroBlocks Web Thing Connected Over Wi-Fi Using NodeMCU

The first YouTube video link below shows control of a fan via the Mozilla Things Gateway user interface. A power-switching relay was attached to a programmable GPIO pin on the NodeMCU. A NodeMCU is a popular low-cost developer board running the Espressif ESP8266 processor, which includes built-in Wi-Fi. The simplicity of the webthing code in this example demonstrates the usefulness of the Mozilla Things framework.

Note: Before following this tutorial, note that the NodeMCU must first be imaged with a special MicroBlocks VM. See Appendix note for details.

Example video showing on/off control of a fan.
https://www.youtube.com/watch?v=XAy2cw7Skoc

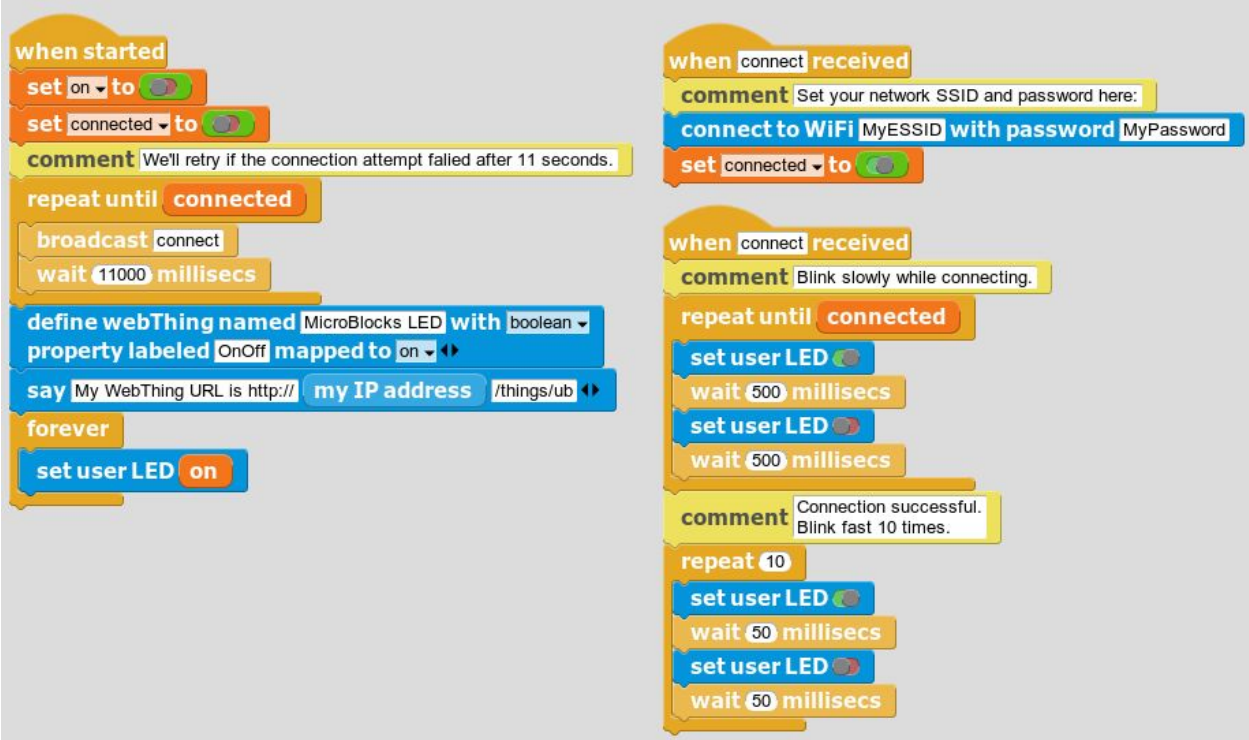
In MicroBlocks 0.1.22 or later you'll find an example under the **Examples** => **Mozilla IoT** folder called "Mozilla NodeMCU WebThing LED.gpp". (Click the **Open** button to load an example.)



September 14, 2018

You can also modify the Thing 1 and Thing 2 examples to add Wi-Fi connectivity and demonstrate them using the NodeMCU as well. For more complex examples using the NodeMCU, you will likely have to source and attach sensors and actuators to its GPIOs.

Here's what the **Mozilla NodeMCU WebThing LED.gpp** example code looks like:



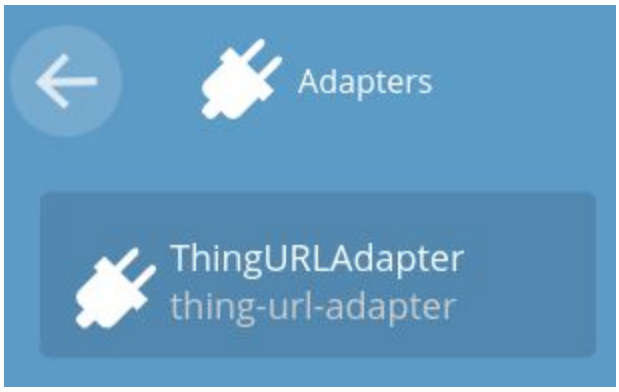
This example is relatively complex because it handles possible connection failure scenarios. The NodeMCU sometimes fails to connect to the network, so the example illustrates how to make sure the connection is successful before initiating commands to define the webthing. The blinking script at the bottom right is optional, but it's helpful because a change from slow to fast blink (then stops) tells you the connection has been successful.

The first step is update the top right script to enter a valid ESSID and PSK, corresponding to your Wi-Fi access point. Then click "Start" and wait for the connection to go through.
Note: the NodeMCU does not support connecting to 5GHz networks, only 2.4GHz.

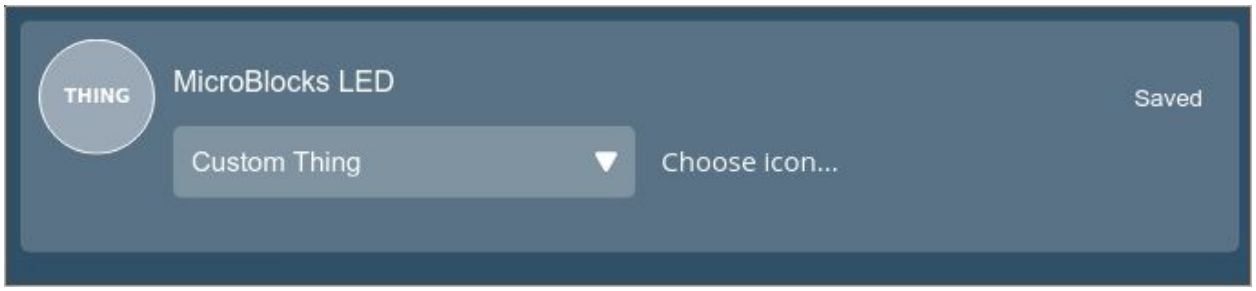
When the connection finishes, a speech bubble shows the URL for your new web thing (where in this example ,”ub” is the name given to the web thing):

September 14, 2018

On the Things Gateway, check the **Settings** => **Adapters** page to make sure you have the web thing (“ThingURLAdapter”) installed and enabled. If not, go to the **Settings** => **Add-ons** page to enable it.

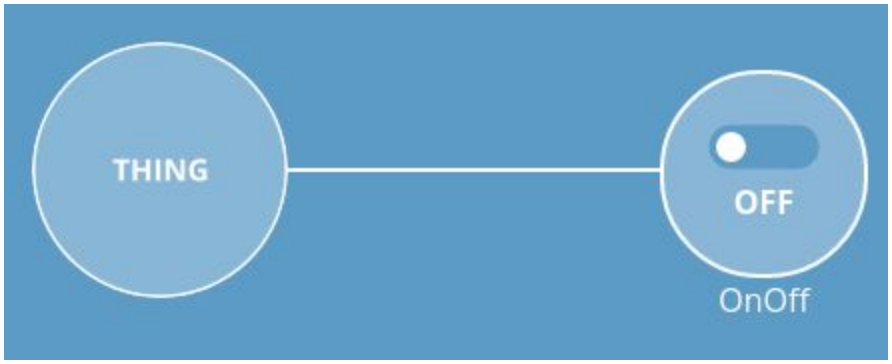


On the Things page, click on the (+) icon to scan for things. Ideally your NodeMCU web thing will be automatically detected if the router supports broadcast mDNS. If you don’t see it, click the "Add by URL..." link and enter the URL that the previous speech bubble gave you (http://192.168.1.137/things/ub). The gateway should detect there's a thing at that URL:



Now you can control the NodeMCU’s onboard LED from your browser!

September 14, 2018



Connect up more sensors and actuators, and map them using the standard capabilities whenever possible, or make up your own custom definitions.

Note that the NodeMCU response time is somewhat slow. After clicking on the switch you may need to wait a couple of seconds for the POST request to get through. The good thing is that the handling is now non-blocking to the other threads in the VM.

Happy hacking!

Appendix -- Flashing the NodeMCU the first time with the MicroBlocks Virtual Machine (VM)

Here's how to install MicroBlocks on a NodeMCU.

1. Install "esptool.py" on your Mac:
   ```
   $ sudo pip install esptool
   ```

2. Download the MicroBlocks virtual machine for the NodeMCU:

   ```
   $ cd <some_dir_where_you_want_the_file>
   $ curl
   https://bitbucket.org/john_maloney/smallvm/src/master/precompiled/vm.ino.nodemcu.bin > vm.ino.nodemcu.bin
   ```

   Or to download the latest version from your browser, go to: https://bitbucket.org/john_maloney/smallvm/src/master/precompiled/vm.ino.nodemcu.bin and click the "View raw" link. Or right-click and select “Save Link As…”. You want to download, not view, the file. In a terminal window, `cd` to the directory with the file.

3. Plug in your NodeMCU, make sure you are in the directory of that file, and run:

```
$ esptool.py write_flash 0 vm.ino.nodemcu.bin
```

If that does not work, try something like the following (with the appropriate port value). This shows the response you might see as well. (The entire process may take 20+ seconds.)

```
$ esptool.py --port /dev/tty.SLAB_USBtoUART write_flash 0
vm.ino.nodemcu.bin
esptool.py v2.2
Connecting........_
Detecting chip type... ESP8266
Chip is ESP8266EX
Uploading stub...
Running stub...
Stub running...
Configuring flash size...
Auto-detected Flash size: 4MB
Compressed 315184 bytes to 225996...
Wrote 315184 bytes (225996 compressed) at 0x00000000 in 20.0
seconds (effective 126.3 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting...
```

4. Start MicroBlocks, click the "Connect" button, and select the serial port. The indicator should turn green. Click the "set user LED" block to verify that everything is working.