Some helping Notes

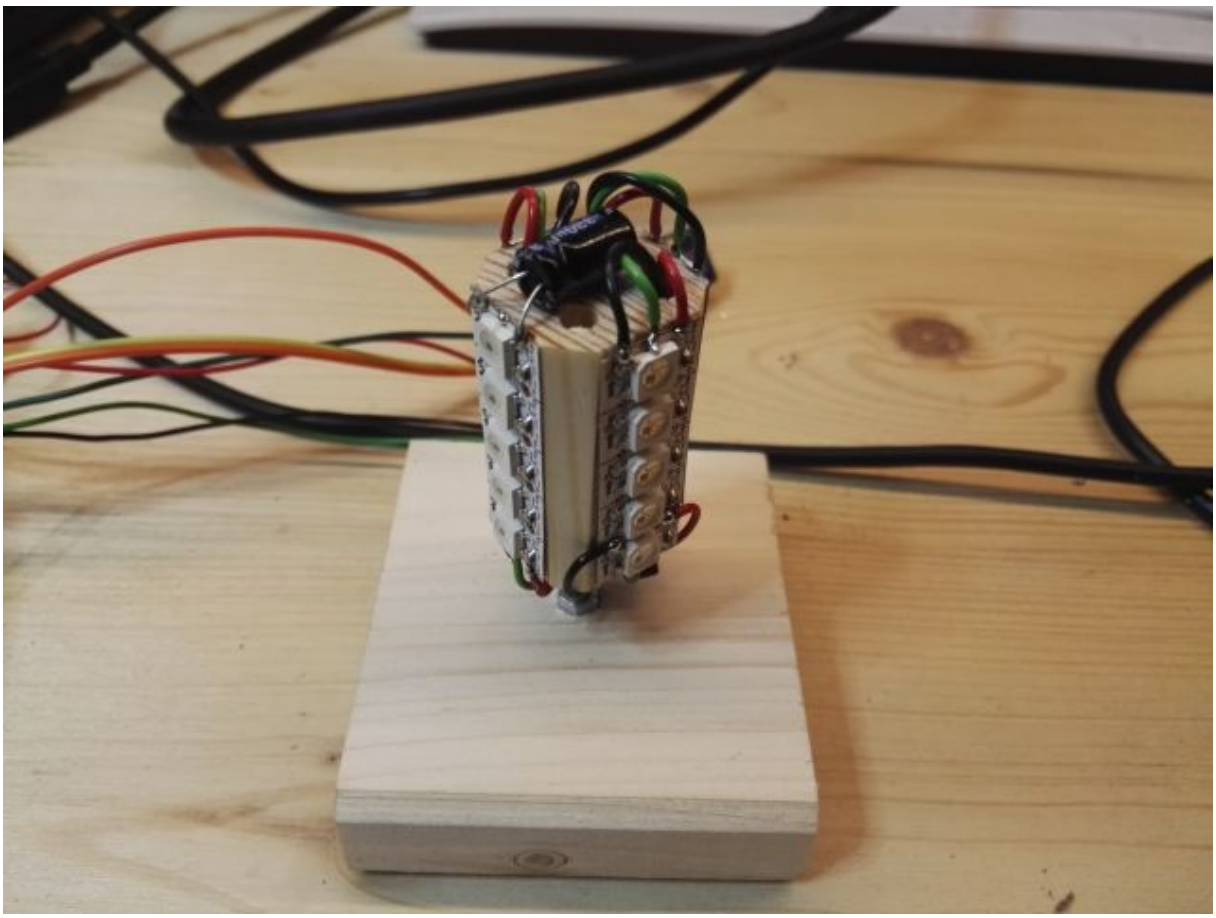
Contents

Wifi Candle

1

The Wifi Candle is a plugin for ESPEasy which uses some neopixels (WS2812) to simulate candle fire (beside some other simulations like a police siren). This guide will show you how to setup the Arduino IDE, assemble the source code and get the stuff running.

ESPEasy	http://www.letscontrolit.com/wiki/index.php/ESPEasy
ESPEasy Forum	http://www.letscontrolit.com/forum/index.php
Forum thread	http://www.letscontrolit.com/forum/viewtopic.php?t=2147
Video	http://www.logview.info/Temp/WifiCandle.mp4
EMail	dominik@logview.info



Wifi Candle

1

Setup the Arduino IDE

- Download the latest Arduino IDE (1.8.0 at the moment): https://www.arduino.cc/en/Main/Software

> *I always use the ZIP version without the installer because you can use it as portable version.*

- extract the ZIP file and within the arduion-1.8.0 folder create a new one which is called "`portable`"
- Start Arduino and open File -> Preferences
  - In the additional boards field add the following: `http://arduino.esp8266.com/stable/package_esp8266com_index.json`
  - It's a good idea to enable Line Numbers and Code folding as well :-)
- Open Tools -> Board -> Boards Manager
  - at the bottom install version 2.3.0 for esp8266
  - this will install the esp8266 toolchain in the created portable folder
- Close the Arduino IDE
- Use this page http://www.letscontrolit.com/wiki/index.php/ESPEasy#Loading_firmware to get the latest libraries for ESPEasy. I used the Release candidate 1.4.7_RC8.
  - download the ZIP and enter the `source\Libraries\` folder
  - copy the 5 folders in that directory to your Arduino folder. For me this is: `.\arduino-1.8.0\portable\packages\esp8266\hardware\esp8266\2.3.0\libraries`
- The preparation of the Arduino IDE is done ...

Setup the Sources

- Now grab a copy of the ESPEasy sources from github.
  - Go to https://github.com/ESP8266nu/ESPEasy
  - There is a green download button on the upper right side. Use "Download ZIP"
  - Extract the ZIP content to a folder on your computer
  - Since the folder is named "ESPEasy-master" we have to rename it to "ESPEasy". If you miss this step you will get into trouble within the Arduino IDE. The folder name must match the .ino name from your project.
- Now grab a copy of the ESPEasy Playground sources
  - Go to https://github.com/ESP8266nu/ESPEasyPluginPlayground
  - again download the ZIP
  - Open the ZIP and extract the file "_P121_Candle.ino" to your ESPEasy folder from the previous step.

If you like extract the "_P122_NeoPixel.ino", too. You can use it for a first test of your neopixels.

- The next steps are some code changes to get the candle working ...
  - Go to the file/tab "**ESPEasy**"
    - find the part with the line `#define PLUGIN_TEN_PER_SECOND 5"` (~line 200)
    - add a new line after that:
      `#define PLUGIN_FOURTY_PER_SECOND 99`
    - find `void loop()` (~line 660)
    - after the line "`runEach30Seconds();`" add a new code section:
      ```
      if (millis() > timer25ms)
            run40TimesPerSecond();
      ```
    - go to the end of the void loop() method (~line 700) and add the following code after that:
      ```
      void run40TimesPerSecond()
      {
        timer25ms = millis() + 25;
        PluginCall(PLUGIN_FOURTY_PER_SECOND, 0, dummyString);
      }
      ```
    - find "unsigned long timer100ms;" and after that line add the following:
      `unsigned long timer25ms;`
    - Enable SPIFFS (because we need to upload an additional JS file) (~line number 123)
      `#define FEATURE_SPIFFS true`
  - Go to the file/tab "**__Plugin**"
    - find `case PLUGIN_TEN_PER_SECOND:` (~line number 1097)
    - Insert a new line after that:
      `case PLUGIN_FOURTY_PER_SECOND:`
  - Go to the file/tab "**Webserver**"
    - find the line `str += F("//--></script>");` (~line number 57)
    - add the following line:
      `str += F("<script src=\"jscolor.min.js\"></script>\n");`
    - find the line `WebServer.sendHeader("Content-Disposition", "attachment;");` (~line number 2174 in method loadFromSPIFFS)
    - add the following lines:
      ```
      if (path.endsWith(".js"))
        WebServer.sendHeader("Cache-Control: max-age=3600", "must-revalidate");
      ```

Compile and Test

- Now make a first test...
  - Start the Arduino IDE

- Under Tools -> Board select your ESP8266 board. In my case I use a NodeMCU V1.0
- Open ESPEasy.ino from the folder above
- First check ... Look in the IDE if you find a tab for the _P121_Candle file.
- Now hit the compile button or simple press CTRL+R
- If all works well you get no errors and end up with an information like this:
  ```
  Sketch uses 463569 bytes (44%) of program storage space. Maximum is 1044464 bytes.
  Global variables use 50972 bytes (62%) of dynamic memory, leaving 30948 bytes for
  local variables. Maximum is 81920 bytes.
  ```

- Now attach your ESP8266 to the computer and upload the compile result to the device.
  - It's a good idea to start with the normal upload first. So don't attach the neopixels to the ESP8266.
  - Look in the serial monitor if the ESPEasy comes up.
  - Now we can clear the settings / SPIFFS to ensure starting from scratch.
    - Connect the TX / RX pins on the ESP8266 board
    - Restart the board (RESET)
    - Now all settings are gone
    - Remove the TX / RX connection
  - Connect to the ESP Easy access point and setup your local Wifi Network
  - Now check if you can insert a "**Wifi Candle**" Device. If not you have missed something from the above steps :-)
  - We need to upload a JavaScript file (jscolor) to the SPIFFS from our ESP8266 to ensure that the color picker will work.
    - Download the lib from here: http://jscolor.com/release/latest.zip
    - Extract jscolor.min.js
    - Now open the Web UI of your ESPEasy with this URL: http://<IP-ESPEasy>/upload
    - Select Browse ... and choose the extracted jscolor.**min**.js File (ensure the ...min... verion !!)
    - Press Upload and you are done.
    - You can check if the upload was successful with the following URL: http://<IP-ESPEasy>/filelist
      If you see the jscolor file in the list you are fine!
  - Next step is the hardware. I use some WS2812 pixels which are connected to GPIO-13 (D7).
    - **Keep in mind that the neopixel use a lot of current**. If you don´t connect a 5V power supply to your ESP8266 board it will not work correctly.
    - The neopixel I used are 4x5 pieces which are connected in a row.
  - Now add the Wifi Candle device and configure the 1st GPIO which is GPIO-13 for me.
    - Set the Delay to 30 and the IDX to 1.
    - Select a Flame Type

    - Press Submit
    - Now you must see the candle working.
- Steps you could think about ...
  - Remove all not used plugins. So every file whoch starts with "*_P...*". (don't remove __Plugin.ino !)