← Moving Raspberry PI B SD Card to to B+ MicroSD Card ESP8266 and DS18B20: Transmitting Temperature Data →

ESP8266 / NodeMCU: Methods of Interrupting init.lua During Boot

Posted on April 24, 2015

(click here to see index of all ESP8266 posts)

As mentioned in my prior post, if you have a bug in yourinit.lua file such that it gets hung in a loop and/or reboots infinitely, then the only way to correct the situation is to reflash the ESP8266 with nodeMCU. Not something that I want to do on even an occasional basis.

The typical way I'm seeing to get around this problem is to delay the start of your code. During this delay you could delete or rename init.lua.

For my test, I have a file calledtest.lua that contains:

```
print('in test.lua')
```

This little snippet represents my 'production' program. This is the program that potentially could have an infinite loop or other bug that prevents regaining control of the ESP8266.

Pausing Before Executing 'Production' Program

To allow myself the ability to regain control of the ESP8266, I will wait 5 seconds after the start of the ESP8266 before starting the 'test.lua' program file.

This is done in init.lua this manner:

```
function startup()
    print('in startup')
    dofile('test.lua')
    end

tmr.alarm(0,5000,0,startup)
```

First is a function, called startup, that will run the actual 'production' program file from flash.

Next, I use tmr.alarm to execute the startup function 5 seconds after the ESP8266 boots.

Now when I reboot the ESP8266:

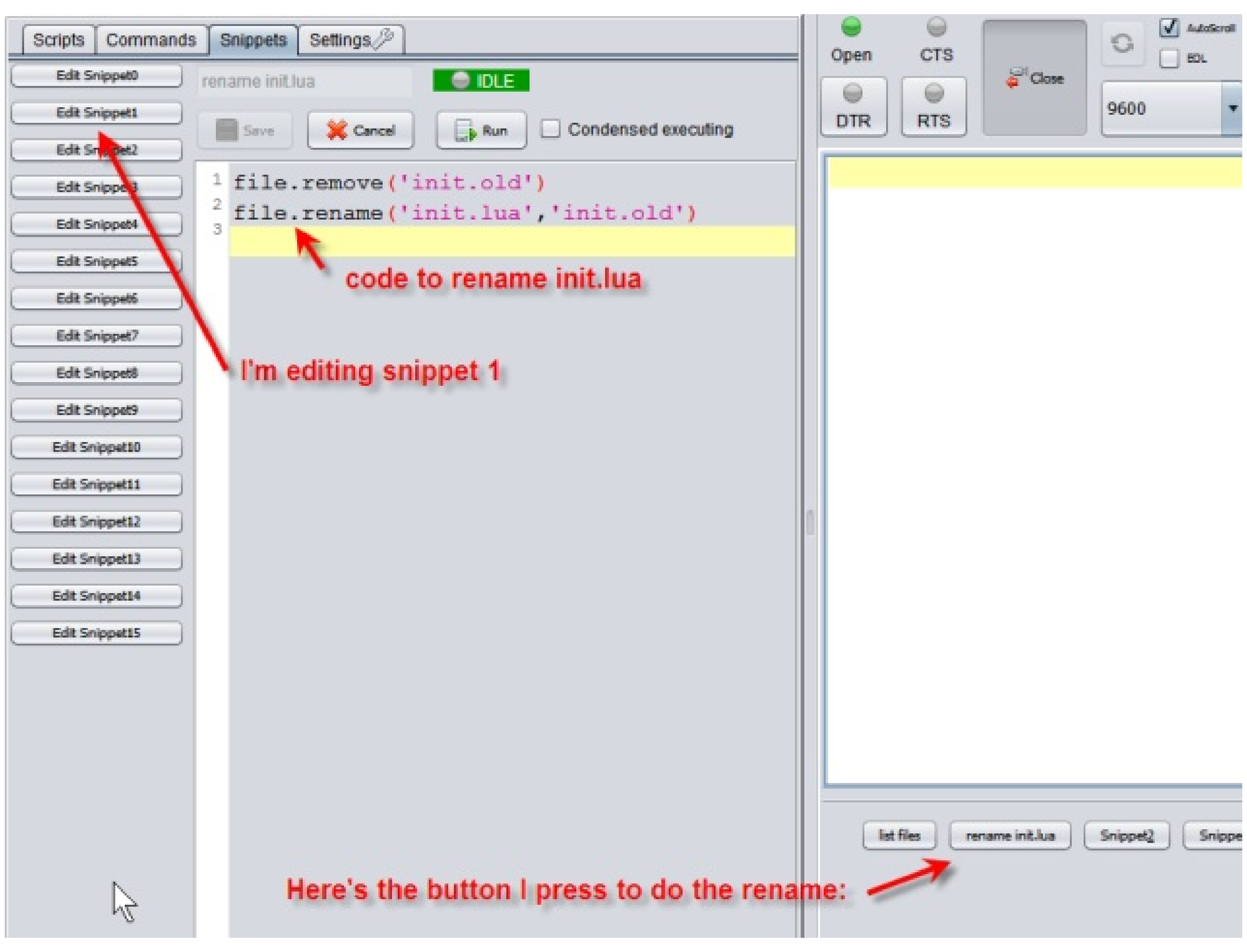
```
NodeMCU 0.9.5 build 20150318  powered by Lua 5.1.4
>
Communication with MCU...
Got answer! AutoDetect firmware...

NodeMCU firmware detected.
=node.heap()
20704
> ** 5 second pause here**
in startup
in test.lua
```

You can see that I have 5 seconds from the time the prompt is printed until the startup function is invoked. That gives me 5 seconds to interrupt the process.

The conventional way, that I've seen so far, to interrupt the process is to delete / rename init.lua. The problem with that, is I guarantee I'm not going to be able to type **file.remove('init.lua')** fast enough under pressure.

So to use this method of interruption, I create a code snippet button that will delete 'init.old' and then rename 'init.lua' to 'init.old':



```
file.remove('init.old')
file.rename('init.lua','init.old')
```

Now, I reboot the ESP8266, and as soon as I get a prompt, I press the 'renameinit.lua' button:

```
NodeMCU 0.9.5 build 20150318  powered by Lua 5.1.4
>
Communication with MCU...
Got answer! AutoDetect firmware...

NodeMCU firmware detected.
=node.heap()
20704
> file.remove('init.old')
> file.rename('init.lua','init.old
> in startup
in test.lua
```

NOTE that this DID NOT stop startup from running. However, the next time the ESP8266 is restarted, init.lua will be missing and you will be able to regain control of the ESP8266.

Overriding With a Boolean Variable

This procedure works fine, but I really don't want to be movinginit.lua around like this. It opens the possibility of accidentally erasing it. Murphy's Law applies to nearly everything I do.

So rather than moving the init.lua file during the 5 second wait, I'd rather just override a boolean variable.

To do this, I create the variable*abort*, initialized it to false, and test for it in the startup function:

```
function startup()
    if abort == true then
        print('startup aborted')
        return
        end
    print('in startup')
    dofile('test.lua')
    end

abort = false
tmr.alarm(0,5000,0,startup)
```

Now, when I want to abort the startup process, I have a code snippet I can execute at the first prompt which will abort the startup without messing with init.lua. PLUS, I don't have to reboot the ESP8266 to regain control:

Search

Translate(BETA)

TRANSLATION

Recent Posts

- Upgrading Teensy Code in the Field
- Using Modulus Arithmetic to Navigate a Circular List
- Writing Code to Handle Arduino's millis() Roll Over
- Obtaining a Local Phone Number for Google Voice
- Preventing Peg Board Hooks from Coming Out of the Peg Board

Recent Comments

Бесконопочные часы на… on ESP8266 / NodeMCU: Methods of…

niraj on Wiring the HobbyKing KK 2.1 Fl…

Dan TheMan on SIM800L GSM/GPRS Part II…

FTP on SIM800L GSM/GPRS Part II…

FTP on SIM800L GSM/GPRS Part II…

```
NodeMCU 0.9.5 build 20150318  powered by Lua 5.1.4
>
Communication with MCU...
Got answer! AutoDetect firmware...

NodeMCU firmware detected.
=node.heap()
21032
> abort=true
> startup aborted
```

Interrupting init.lua with the Return Key During Boot

I'm pretty happy with the above solution, but in my production Arduino Code I usually have some type of boot override that occurs by pressing the Return key during boot (typically to let me get in and change program constants). Can the same thing be done with the ESP8266 / nodeMCU?

Here is some code that does just that;

```
function abortInit()
    -- initailize abort boolean flag
    abort = false
    print('Press ENTER to abort startup')
    -- if <CR> is pressed, call abortTest
    uart.on('data\r', abortTest, 0)
    -- start timer to execute startup function in 5 seconds
    tmr.alarm(0,5000,0,startup)
    end

function abortTest(data)
    -- user requested abort
    abort = true
    -- turns off uart scanning
    uart.on('data   end

function startup()
    uart.on('data   -- if user requested abort, exit
    if abort == true then
        print('startup aborted')
        return
        end
    -- otherwise, start up
    print('in startup')
    dofile('test.lua')
    end

tmr.alarm(0,1000,0,abortInit)           -- call abortInit after 1s
```

I think that is pretty self evident. We start abortInit after 1 sec (giving the ESP8266 a bit of time to get going). abortInit sets up the uart to scan for a RETURN and then sets up the startup function to start in 5 seconds.

If the uart sees a RETURN it sets the abort flag to true.

Once the startup function is called, it clears the uart from further scanning, and tests the abort flag to determine if the user program should be called.

Here is what the output looks like WITHOUT aborting:

```
Communication with MCU...
Got answer! AutoDetect firmware...

Can't autodetect firmware, because proper answer not received.
powered by Lua 5.1.4
>
> Press ENTER to abort startup
in startup
in test.lua
```

and here it is when I press RETURN after the*Press ENTER to abort startup* message is displayed:

```
NodeMCU 0.9.5 build 20150318  powered by Lua 5.1.4
>
Communication with MCU...
Got answer! AutoDetect firmware...

NodeMCU firmware detected.
=node.heap()
19280
> Press ENTER to abort startup

> startup aborted
```

Share this:

Twitter Facebook Google

Loading...

Related

ESP8266 NodeMCU/LUA: Saving, Executing, and Compiling Script Files
In "c-esp8266"

ESP8266 and Using the Arduino IDE
In "c-esp8266"

ESP8266 Reading/Writing GPIO and Transmitting/Receiving UDP Packets
In "c-esp8266"

This entry was posted in c-esp8266 and tagged esp8266 init.lua boot. Bookmark the permalink.

← Moving Raspberry PI B SD Card to to B+ MicroSD Card   ESP8266 and DS18B20: Transmitting Temperature Data →

12 Responses to *ESP8266 / NodeMCU: Methods of Interrupting init.lua During Boot*

Pingback: *Can't upload new LUA files once ESP8266 init.lua starts | Austinlightguy's Blog*

Michal Kotnowski *says:*
September 15, 2015 at 1:43 pm

Great solutions.
The only thing, in the last example, I think, your code is missing some parts:
uart.on('data end
and
uart.on('data — if user requested abort, exit

Can you post the correct code?

Reply

Dan TheMan *says:*
September 15, 2015 at 2:39 pm

Hmmm, if you are showing verbatim examples from my code I have to wonder if something is happening in your browser as the code I'm looking at doesn't skip after . I successfully tested my example code which gave the output further below. I'll repost the example here, in case it helps.

function abortInit()
— initailize abort boolean flag
abort = false
print('Press ENTER to abort startup')
— if is pressed, call abortTest
uart.on('data', '\r', abortTest, 0)
— start timer to execute startup function in 5 seconds

```
tmr.alarm(0,5000,0,startup)
end

function abortTest(data)
— user requested abort
abort = true
— turns off uart scanning
uart.on('data')
end

function startup()
uart.on('data')
— if user requested abort, exit
if abort == true then
print('startup aborted')
return
end
— otherwise, start up
print('in startup')
dofile('test.lua')
end

tmr.alarm(0,1000,0,abortInit) — call abortInit after 1s
```

Reply

Michał Kotnowski *says:*
September 19, 2015 at 11:27 pm

Thank you for the response. And yes, the code I have quoted is like I can see it in Chrome – both in mobile and on a PC. In your last comment I can also see DSPlorer not ESPlorer and I do not know, if this is a typo or something elseelse.
Anyway, thank you for code in the comment, I prefer not to remove files, when experiencing issues.

Reply

Mike *says:*
October 20, 2015 at 3:38 am

Hi Dan – Thanks for the blog – to confirm, your code is also mangled on my browser (Safari on OS X 10.10). It's caused by the "skimresources" scripts on your page – if I blackhole r.skimresources.com and s.skimresources.com, the scripts are no longer mangled.

Dan Beks *says:*
September 19, 2015 at 4:56 am

(Not sure if my previous post posted?)

Or, a simpler way, if using DSPlorer, you can perform the following:
1. Connect your ESP8266 to your PC via UART, as you normally would for uploading scripts.
2. Power-up your ESP8266 – it will run the "init.lua" script.
3. Fire-up ESPlorer and open a connection to your ESP8266 – yes, whilst the ESP8266 is running.
Alternatively, keep the ESP8266 in reset mode whilst you open a connection in DSPlorer. Release the reset once you get the message "Communication with MCU" followed by "…"
4. Issue the command – 'file.remove("init.lua")' to your ESP8266.
5. Reset your ESP8266. ESPlorer should respond with "cannot open init.lua"

Job done 🙂

Reply

Andy *says:*
June 22, 2016 at 11:36 am

Very good! I used Overriding With a Boolean Variable with time 3000 ms and it works perfectly!
My story: I programmed init.lua with some "bad commands" and i thought that it's damaged, it didn't respond to any new commands. But i flashed blank512k.bin, then NodeMCU (both with NodeMCU flasher) and now I'm using this "stopping function" to prevent future bugs.
Thnak you! 🙂

Reply

Pingback: *ESP8266 / NodeMCU: Methods of Interrupting init.lua During Boot – dbsesp*

DBS ESP8266 Guy *says:*
July 9, 2016 at 12:32 pm

```
function abortInit()
print('Press ENTER to abort startup')
uart.on("data", "\r", function(data)
tmr.unregister(0)
uart.on("data")
print('Startup aborted')
end, 0)
tmr.alarm(0,5000,0, function()
uart.on("data")
print('Running startup')
dofile('main.lua')
end)
end

tmr.alarm(0,1000,0,abortInit)
```

Reply

DBS ESP8266 Guy *says:*
July 9, 2016 at 12:38 pm

Sorry, I wanted to thank you for this blog, it was a big help.
I made some improvements to the code, a bit shorter and does not need to wait the 5 sec to abort, the "Startup aborted" appears almost instantly.

```
function abortInit()
print('Press ENTER to abort startup')
uart.on("data", "\r", function(data)
tmr.unregister(0) — disable the start up timer
uart.on("data") — stop capturing the uart
print('Startup aborted')
end, 0)
tmr.alarm(0,5000,0, function()
uart.on("data") — stop capturing the uart
print('Running startup')
dofile('main.lua') — run the main program
end)
end

tmr.alarm(0,1000,0,abortInit)
```

Reply

Dan TheMan *says:*
July 9, 2016 at 6:13 pm

Thanks for the code!

Reply

Pingback: *Бесконопочные часы на esp8266 и их апгрейд - Loess.ru*

Leave a Reply

Enter your comment here...

Follow Blog via Email
Enter your email address to follow this blog and receive notifications of new posts by email.

Enter your email address

Big Dan the Blogging Man