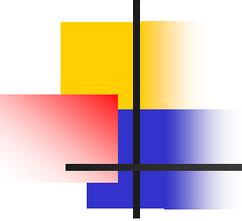


IoT For Fun!

Chen-Hanson Ting

SVFIG

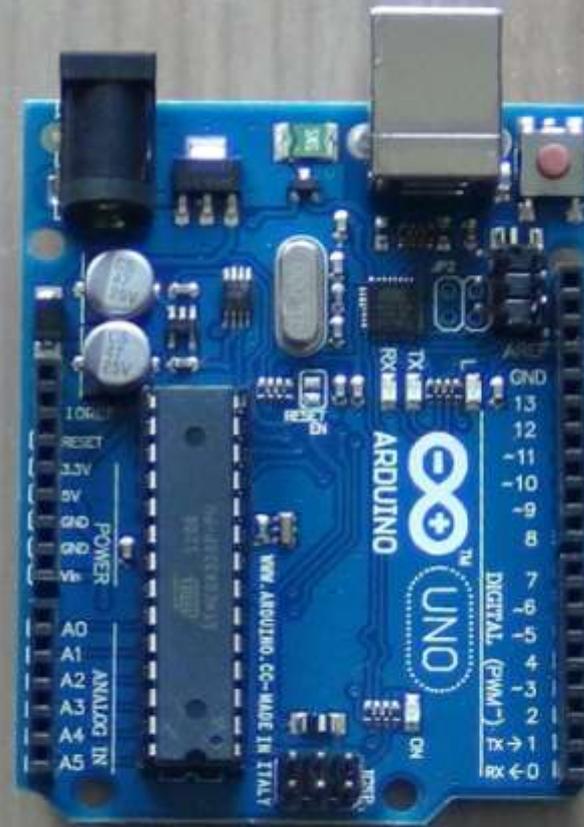
January 28, 2017



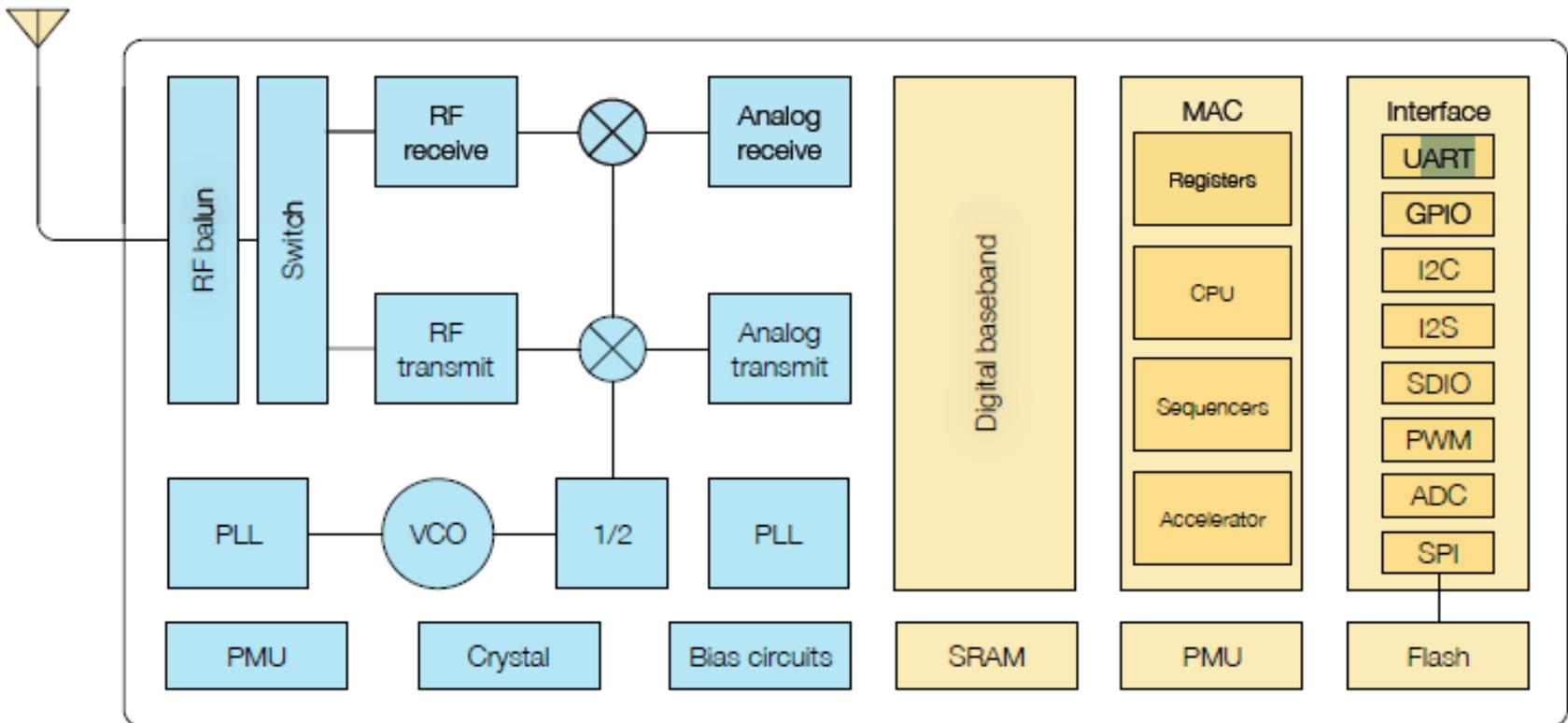
ESP8266

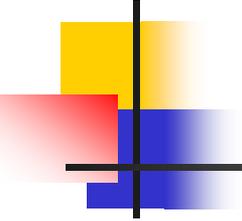
- It looks that ESP8266 12E will replace Arduino Uno, with its WiFi capability, 32-bit processor, and large memories.
- With ESP8266 kits, we can participate in the new IoT revolution, and have lots of fun.

ESP 12E and Arduino Uno



ESP8266





ESP8266 Hardware

- 3.3V, \sim 215 mA
- Xtensa LX3 CPU, 32-bit, 160 MHz
- RAM 32Kb, DRAM 80Kb, Flash 4 Mb
- Wi-Fi 802.11 b/g/n 2.4 GHz radio
- GPIO, PWM, ADC, UART, I2C, SPI

ESP 12E

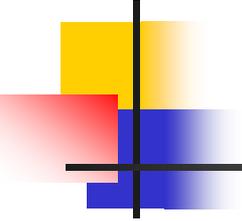


NodeMCU



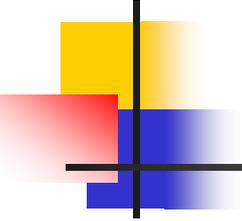
WiFiBoy





Local Area Network

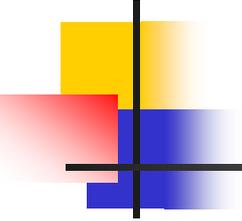
- A local area network (LAN) with a number of ESP8266 running Forth. They can communicate with a host computer.
- A host computer sends out Forth commands to each ESP8266 to accomplish certain task.



Soft Access Point



ESP8266 operating in the **Soft Access Point** mode



Demo

- 1 ESP8266 set up as an Access Point.
- 3 ESP8266 set up as Stations to receive Forth commands.
- PC serves as a UDP terminal, sends out Forth commands to Stations.

Local Area Network

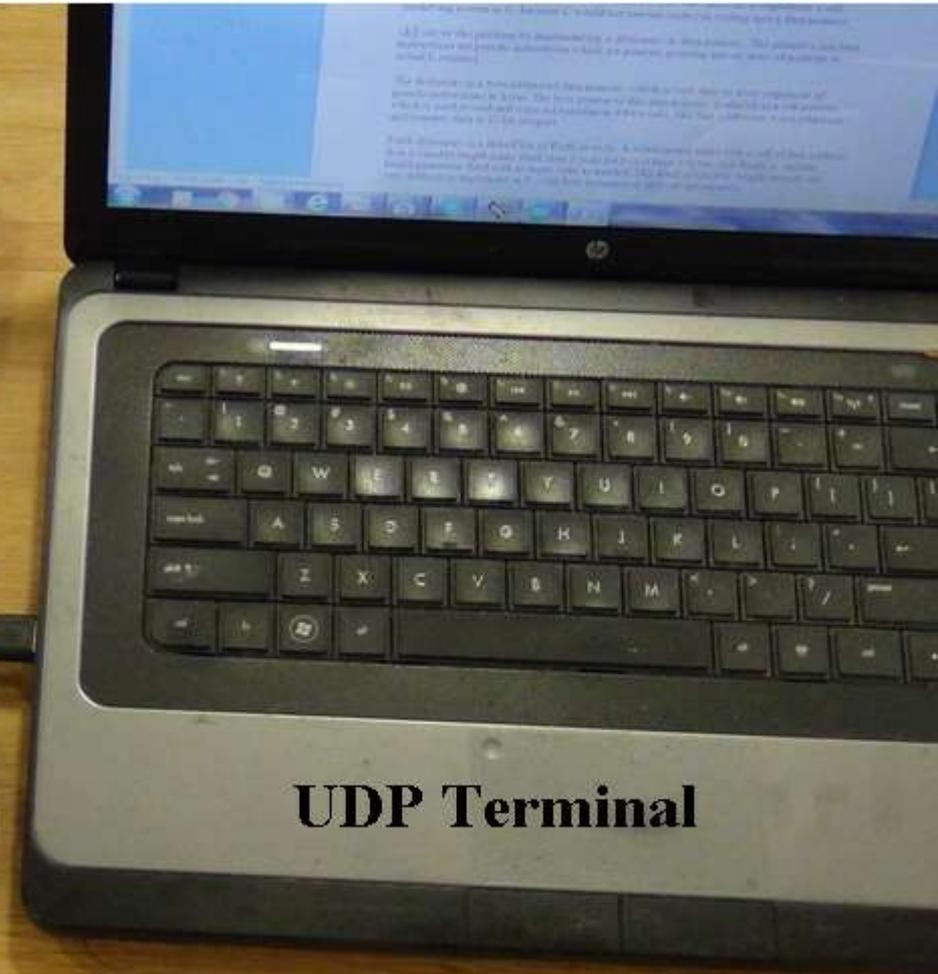
Station 1



Station 2



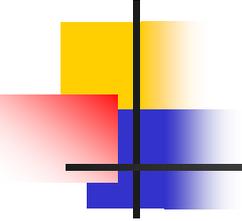
Access Point



UDP Terminal

Access Point





espForth on a Station

- On power-up, connect to LAN Access Point and display IP address and port number.
- Receive commands from Serial Terminal and UDP packets.
- Send text strings to Serial Terminal, UDP client, and optional LCD display.

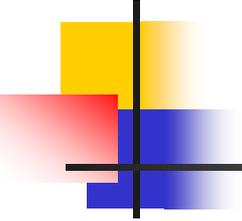
Station 1



Station 1, Close-up

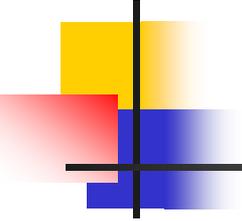
words

```
IMMEDIATE COMPILER
CONSTANT VARIABLE DOES
CREATE CODE " " C
ABORT WHILE WHEN
REPEAT AFT
GAIN UNTIL AHEAD IF
GIN FOR THEN COLOR
UDP FORGET WORDS
.ID >NAME P pp+
DUMP dm+
OVERT $COMPILER COMP
ILE [COMPILER] $
?UNIQUE "$" NAME
```



ESP8266 IDE

- Lua programming language with ESPlorer IDE
- Arduino IDE with C-like language and flashing tool
- Micropython with esptool and WebREPL interpreter



ESPlorer IDE

- You can send Lua commands directly to ESP8266.
- You can edit a file in the Editor panel.
- You can upload a file to ESP8266 flash memory.
- You can run a file in flash memory.

ESPlorer IDE

The screenshot displays the ESPlorer IDE interface. At the top, the title bar reads "ESPlorer v0.2.0-rc5 by 4refr0nt". The menu bar includes "File", "Edit", "ESP", and "View Links?". Below the menu is a toolbar with buttons for "Scripts", "Commands", "Snippets", and "Settings". A secondary toolbar contains icons for "Open", "Reload", "Save", "Save...", "Close", "Undo", "Redo", "Cut", "Copy", and "Paste".

The main editor area shows a file named "init.lua" with the following code:

```
1 spi.setup(1, spi.MASTER, spi.CPOL_LOW, spi.CPHA_LOW, 8, 8)
2 disp = ucg.wifiboys(8, 4)
3 disp.begin(ucg.FONT_MODE_TRANSPARENT)
4 disp.setFont(ucg.font_helvB08_hr)
5 disp.clearScreen()
6 disp.setPrintPos(8, 15)
7 disp.print("Hello, WiFiBoy!")
8
```

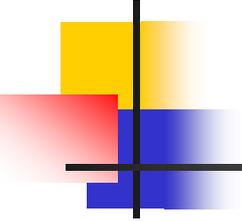
Below the editor, a status bar shows "IDLE" and the file path "C:\WiFiBoy\ESPlorer\ESPlorer\init.lua". A row of buttons includes "Save&Run", "Save&Compile", "Save&Compile&Run...", and "Save". A second row has "Save&Compile All", "View on ESP", "View on ESP", and "Save&". A third row features "Save to ESP", "Send to ESP", "Run", and "Upload".

On the right side, a "COM8" port selection dropdown is visible. Below it are buttons for "Open", "CTS", "Close", "DTR", and "RTS". A baud rate dropdown is set to "115200". Checkboxes for "AutoScroll", "CR", "LF", "BOL", "Hide Editor", and "Hide Terminal" are present. A "Donate" button is also shown.

The terminal window displays the output of the program, including the error "lua: cannot open init.lua" and the successful execution of the code, resulting in "Hello, WiFiBoy!".

Below the terminal, there are buttons for "Format", "FS Info", and "Reload". A row of buttons labeled "Snippet0" through "Snippet15" is also visible.

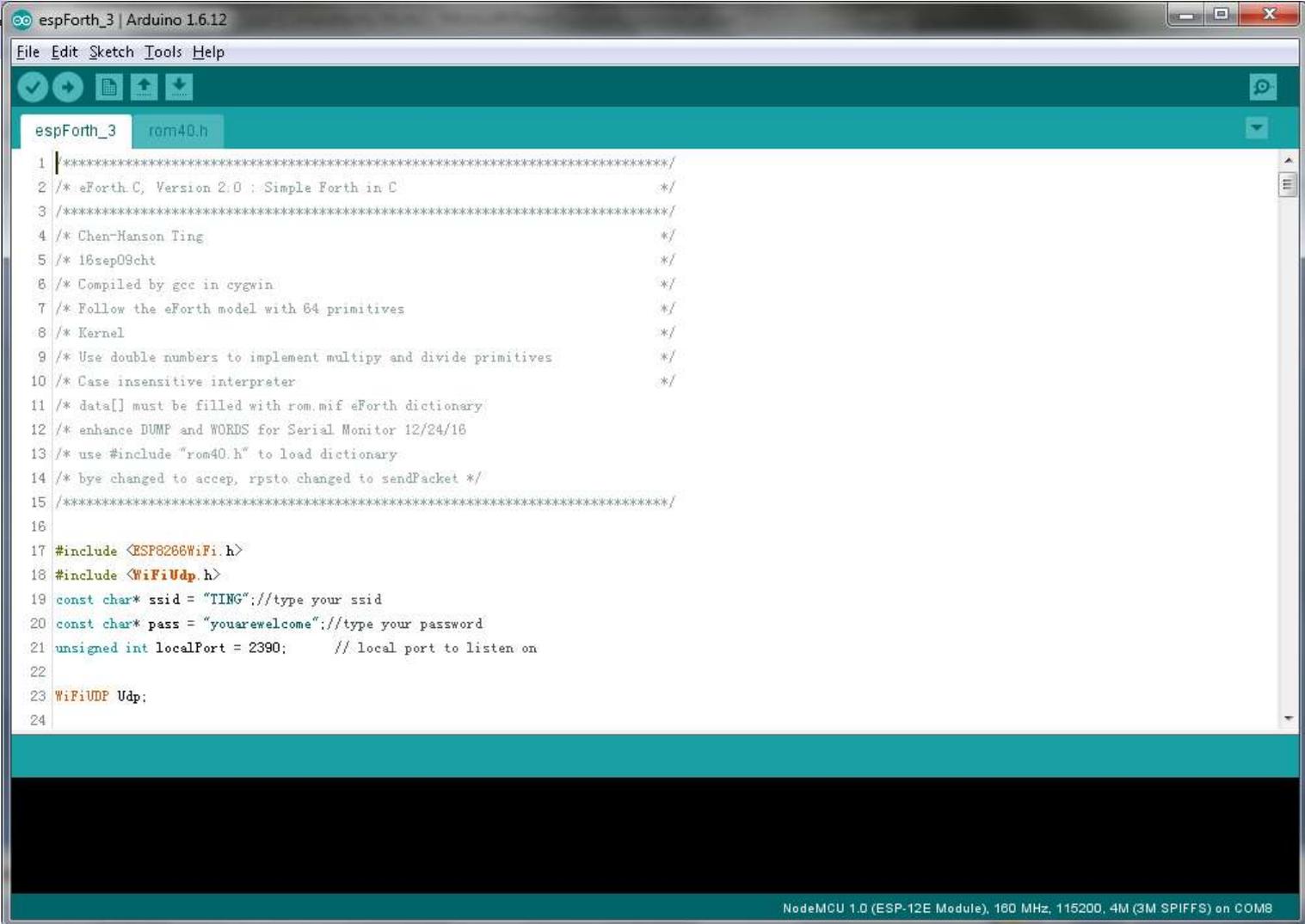
At the bottom right, buttons for "Heap", "Chip Info", "Chip ID", "Flash ID", and "Reset" are present. A "Send" button is located at the very bottom right.



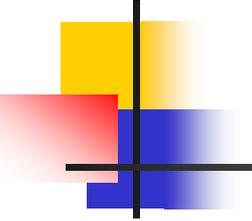
Arduino IDE

- You can load a sketch into editor window.
- A sketch is a C-like file with `setup()` and `loop()` routines.
- You can compile a sketch and upload it to ESP8266, and it gets executed.

Arduino IDE



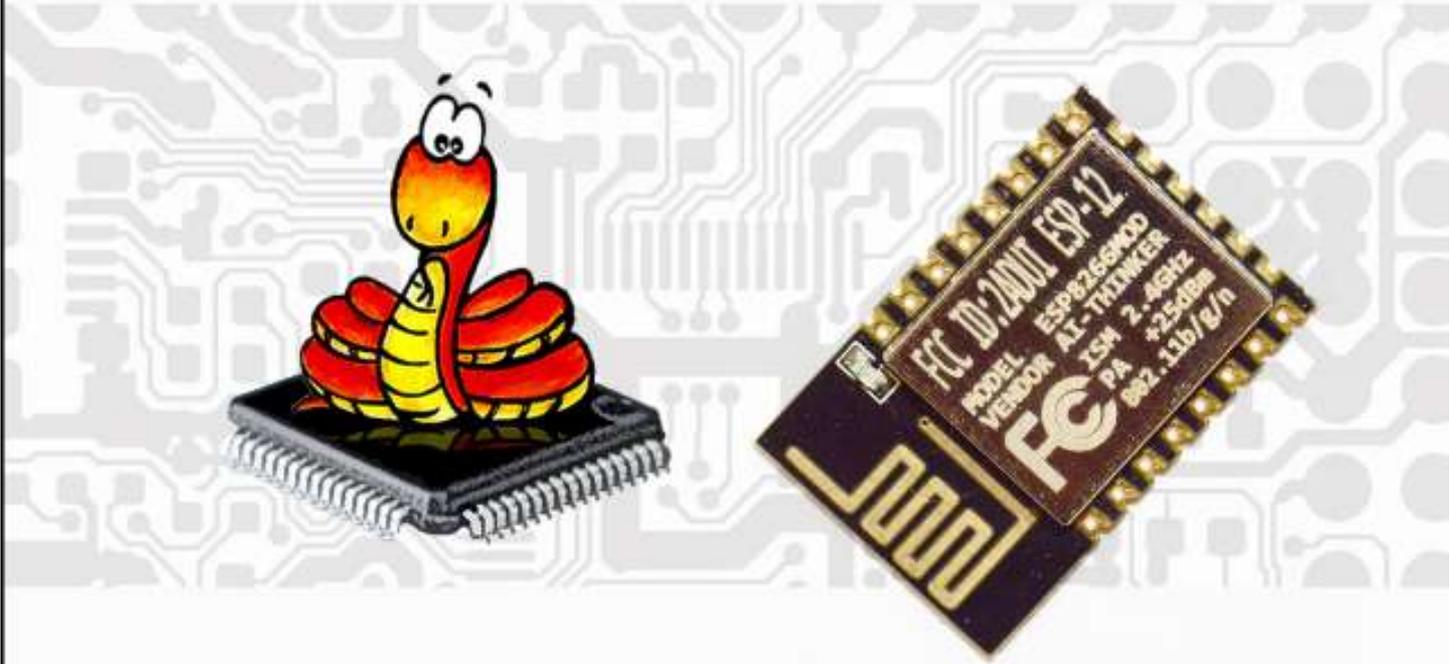
```
espForth_3 | Arduino 1.6.12
File Edit Sketch Tools Help
espForth_3 rom40.h
1 | *****/
2 /* eForth.C, Version 2.0 : Simple Forth in C */
3 /*****/
4 /* Chen-Hanson Ting */
5 /* 16sep09cht */
6 /* Compiled by gcc in cygwin */
7 /* Follow the eForth model with 64 primitives */
8 /* Kernel */
9 /* Use double numbers to implement multiply and divide primitives */
10 /* Case insensitive interpreter */
11 /* data[] must be filled with rom.mif eForth dictionary */
12 /* enhance DUMP and WORDS for Serial Monitor 12/24/16 */
13 /* use #include "rom40.h" to load dictionary */
14 /* bye changed to accep, rpsto changed to sendPacket */
15 /*****/
16
17 #include <ESP8266WiFi.h>
18 #include <WiFiUDP.h>
19 const char* ssid = "TING"; //type your ssid
20 const char* pass = "youarewelcome"; //type your password
21 unsigned int localPort = 2390; // local port to listen on
22
23 WiFiUDP Udp;
24
NodeMCU 1.0 (ESP-12E Module), 160 MHz, 115200, 4M (3M SPIFFS) on COM8
```



Micropython/WebREPL

- Micropython is a powerful interpreter for ESP8266.
- WebREPL (Read-Evaluate-Print Loop) is a graphic interface on PC for ESP8266.
- You can download/upload files to/from ESP8266, and interpret them in ESP8266.

Micropython



MicroPython on the ESP8266

Micropython/WebREPL

ws://192.168.4.1:8266/ Disconnect

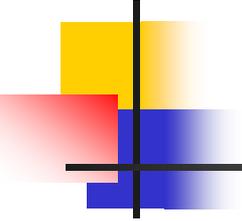
```
Welcome to MicroPython!  
Password:  
WebREPL connected  
>>> print('Hello, World!')  
Hello, World!  
>>> |
```

Send a file
Browse...
Send to device

Get a file
Get from device

(file operation status)

Terminal widget should be focused (text cursor visible) to accept input. Click on it if not.
To paste, press Ctrl+L then Ctrl+V



Closing Remarks

- ESP8266 is IoT ready.
- Are we ready?
- There are many IDE's already available for ESP8266.
- Will Forth play a role in this ESP8266 revolution?