Installation and use of the upload server by J.v.d.Ven, July 17th, 2023.

The objective of the upload server is to copy an ASCII source on a local network to
the file system of an ESP-12F or ESP32 on which extended version of cforth runs.
1It can be cloned using: $ **git clone https://github.com/Jos-Ven/cforth/tree/WIP--branch WIP**

The ASCII file to upload should not contain any byte with 0x00.
To transfer a file a kind of sliding window over an UPD connection is used.
Speeds of 110 kbits/sec are reachable. Most of my sources are transferred in about 1/2 second.
The server part runs under Gforth version gforth-0.7.9_20220713 under Linux (Bullseye/Bookworm)
or under Win32Forth on a windows10/11 PC.
In both cases a HTML-gui on the server can be used to send or compile a file on a ESP-system
Before you start you should be able to use gforth on a linux PC.
Then **install** and run the websterverlight. Downloadable from:
https://github.com/Jos-Ven/A-smart-home-in-Forth
Connect the ESP-system to the a linux PC using a CP2102.

Actions on a Linux PC to install the **receiver** on the ESP-system in **ROM**.

On an **ESP8266** use:	On an **ESP32** use:
$ **su** Add in ~/cforth/src/app/esp8266-rtos/app.fth the line: **fl ../esp/rcvfile.fth** before the line: : interrupt? (-- flag) $ **cd** ~/cforth/build/esp8266-rtos $ **ls** # Be sure to see: Makefile $ **rm *.*** # Delete a number of files for a fresh start $ **make flash**	$ **su** Add in ~/cforth/src/app/esp32/app.fth the line: **fl ../esp/rcvfile.fth** before the line: : interrupt? (-- flag) $ **cd** ~/cforth/build/esp32 $ **ls** # Be sure to see: Makefile $ **rm *.*** # Delete a number of files for a fresh start $ **COMPORT=/dev/ttyUSB0 make flash**

Then start a communication program to use cforth on the ESP-system.
Eg: $ **picocom /dev/ttyUSB0 -b 115200 --omap crlf --imap delbs**
ttyUSB0 is the port to the serial connection on my PC.
Push the reset button of the ESP-system to see the OK prompt of cforth.
Exit the communication program. In picocomm **<Ctrl+c+a+x>**

Installation for the **server** side on the Linux PC.: xxx
copy _UploadServer.f, UdpSender.f and gf.sh GpioSwitch_server.fth to the directory
where the webserverlight is installed.
Under Linux enter: **./gf.sh** to start the upload server in the background. (could be a CRON-job)
After compilation, the upload server should show its **html-address.**

To test:
Start a communication program on the Linux PC to get into cforth on the ESP-system **.**
Then hit **r <enter>** to start the receiver.
Enter the needed **SSID** and **password** to logon to your wifi router.
The receiver should show the IP address of the ESP-system

Start a browser and go to html page of the upload server. (EG: http://192.168.0.219:8080/home)
Then in the form on the html page:
Enter/change the IP address of the ESP-system.
Hit **Save** and **Report** to test the connection.

Each time you start the upload server you must test the connection first.
If the connection is successful you can send files by hitting **Send** and **Report**.
Try to send and compile a source.
The link **Receiver:** can be used to enter a webserver on the ESP-system

Notes:
1) The button **Save** also interrupts an upload session.
2) When another directory is entered the button **Set** should be pushed.
3) When another extension is used, you should also hit **Save** and **<UpdateList**
4) After selecting another file you should hit **Save**.
5) The receiver can be cancelled by pushing the space bar in cforth and then hit **Save** on the server.
6) Load only works after the involved file has been sent to the ESP-12F.
7) Under Win32Forth 6.15 enter: **needs _UploadServer.f** to start the server.
8) If your application is started by default after a reboot (Using the Start file)
and able to activate the receiver then you might not need the serial connection.
9) The link at "Send to" links to: http://192.168.0.216/home unless the IP address has been changed and saved.
10) Various test revealed that _UploadServer.f ran without problems on my network on
a Rpi Zero, with Gforth 0.7.9_20200910, under Linux Debian and under Win10 with Win32Forth.
11) TCP port 8080 (html-server) and UDP port 8899 must be allowed for usage in your firewall and router.

Known issues:
1) Under window you may have to disable the **Hyper-V Virtual Ethernet Adapter** in order to get the server working.
2) Use only 1 upload server at the time.
3) A serial connection to a device that is OFF will prevent a proper boot at the ESP-12F.
4) Found on: https://electrosome.com/esp8266-arduino-programming-led-blink/
"ESP-12 and ESP-01 has blue color on board LED.
Which is connected in reverse i.e. Anode(+ve) of the LED
is connected to VCC and cathode (-ve) is connected to ESP-12 GPIO2.
It means that LED becomes on when we output LOW and off when we output HIGH.
This pin is also Tx, you cannot use serial when using this pin as LED."
That is not a problem on my ESP-12F while flashing.
Non ESP-12F systems might need a change for the sysled at the start of
rcvfile.fth in ~/cforth/src/app/esp8266-rtos
5) If you get a bad connection indication after you hit Report **2 times** then check:
1) Has the receiver been activated?
2) Re-connect the ESP-system to the power source.
Hit Save as soon as you started the receiver on the ESP-system.
3) Check if the IP address is right in the upload server.
4) Check if the ESP-system has been connected to your router (ping it).
6) A long line like: ok ïïïïïïïïïïïïïïïïïïïï is shown or a message like:
FATAL: cannot open /dev/ttyUSB0: Input/output error appears.
Reboot the PC. The best way is to start the ESP-system first and then the PC.
7) assertion "Invalid mbox" failed: file "~/ESP8266_RTOS_SDK/components/lwip/lwip/src/api/tcpip.c",
line 455, function: tcpip_send_msg_wait_sem
abort() was called at PC 0x40223340 on core 0
Happens when not connected to the wifi router.
Enter: **logon**
8) One Rpi A with a WiFi plug a did not work.
9) On an ESP8266 the receiver in the background and a webserver can not run at the same time (hangs)
It is possible to exit the webserver and than start the receiver.
That's all.