WiFi

While the possibilities of interacting with hardware directly connected to the ESP32 are endless, some real fun can be had connecting to the Internet. By being able to send and receive data to servers across the Internet, the ESP32 can download current weather or time data, upload sensor data, and respond to events (e.g. on Twitter) in real time. This is just the beginning of exploring the Internet of Things (IoT).

To demonstrate accessing the Internet, we will write a quick MicroPython program that connects to a local WiFi and downloads the [example.com](http://example.com/) web page using an HTTP GET command.

**Note:** You will need a WiFi access point with connection to the Internet for this exercise.

Hardware Connections

We'll keep it simple this time--just the ESP32 Thing will be needed. We're going to use its built-in WiFi radio, get a web page, and then send the downloaded HTML contents over the serial connection.

Code: GET Some

In a new file, enter the following code (change <YOUR WIFI SSID> to the name of your WiFi network name and <YOUR WIFI PASSWORD> to the name of your WiFi's password--note that the name and password should be in quotation marks ""):

COPY CODE**import** machine

**import** sys

**import** network

**import** utime

**import** urequests

# Pin definitions

repl\_button **=** machine.Pin(0, machine.Pin.IN, machine.Pin.PULL\_UP)

repl\_led **=** machine.Pin(5, machine.Pin.OUT)

# Network settings

wifi\_ssid **=** "<YOUR WIFI SSID>"

wifi\_password **=** "<YOUR WIFI PASSWORD>"

# Web page (non-SSL) to get

url **=** "http://example.com"

# Create a station object to store our connection

station **=** network.WLAN(network.STA\_IF)

station.active(True)

# Continually try to connect to WiFi access point

**while** **not** station.isconnected():

# Try to connect to WiFi access point

print("Connecting...")

station.connect(wifi\_ssid, wifi\_password)

# Check to see if our REPL button is pressed over 10 seconds

**for** i **in** range(100):

**if** repl\_button.value() **==** 0:

print("Dropping to REPL")

repl\_led.value(1)

sys.**exit**()

utime.sleep\_ms(100)

# Continually print out HTML from web page as long as we have a connection

**while** station.isconnected():

# Display connection details

print("Connected!")

print("My IP Address:", station.ifconfig()[0])

# Perform HTTP GET request on a non-SSL web

response **=** urequests.get(url)

# Display the contents of the page

print(response.text)

# Check to see if our REPL button is pressed over 10 seconds

**for** i **in** range(100):

**if** repl\_button.value() **==** 0:

print("Dropping to REPL")

repl\_led.value(1)

sys.**exit**()

utime.sleep\_ms(100)

# If we lose connection, repeat this main.py and retry for a connection

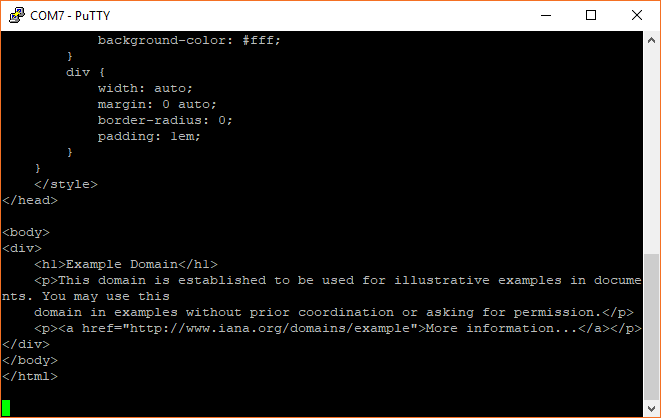
print("Connection lost. Trying again.")

Save the file with a name like *wifi.py*. In a command terminal, navigate to the directory where *wifi.py* is stored. Hold down button 0 on the ESP32 Thing to exit its currently running program. Back in the command terminal, enter the following commands (<PORT> should be changed to your serial port name/location):

COPY CODEcp wifi.py main.py

ampy --port **<**PORT**>** put main.py

Open a serial connection to the ESP32, and you should see your ESP32 connect to your WiFi network and then download the example.com HTML page. If you see text like the one below, you know that it worked:

[](https://cdn.sparkfun.com/assets/learn_tutorials/6/5/9/screen_10.png)

If you are unable to connect and download the page, double-check the SSID and password in the code. Additionally, some network configurations do not work well with the ESP32, which means you might need to try a different network.

Code to Note

In this example, we're relying on the network and urequests modules to handle much of the networking for us. We use the network module to perform the low-level communication over WiFi, including connecting to our Wifi access point. To create a connection to our WiFi access point, we first create a WLAN object and use network.STA\_IF to tell the ESP32 that we want to use it as a client that connects to a separate access point:

COPY CODEstation **=** network.WLAN(network.STA\_IF)

station.active(True)

From there, we attempt to establish a connection with the access point:

COPY CODE**while** **not** station.isconnected():

# Try to connect to WiFi access point

print("Connecting...")

station.connect(wifi\_ssid, wifi\_password)

The line station.connect(wifi\_ssid, wifi\_password) is what actually performs the connection. We wrap this up in a while loop that continually checks to see if we have a connection. This way, if no connection is established, the program will not move forward and try to perform a GET request on a website. Notice that we also include our "Drop to REPL" button check in this while loop, too, in case the ESP32 gets stuck here.

Once we have a connection, we use the urequests module to perform our GET request on example.com:

COPY CODEresponse **=** urequests.get(url)

This returns a Response object that contains all the information returned by the server. We can view the response in text form with the command:

COPY CODEprint(response.text)

We then wait for 10 seconds before requesting the page again. During those 10 seconds, we check for a button 0 push every 100 ms.

If you would like to learn more about networking in MicroPython, see the [network module documentation](http://docs.micropython.org/en/v1.9.3/esp8266/esp8266/tutorial/network_basics.html).

Bottom of Form