Blink

Language: MicroPython

Prerequisites:

- Initial setup (WiFi setup recommended)

Objective:

- Manipulate Digital pins
- Explore single run , looped, threaded, and limited looped programs

Single run program

1. Connect to ESP32 and Create a new file (Green + button) called test.py
2. Paste the Code bellow and press the Run button

```
# We are getting some predefined functions
from machine import Pin
from time import sleep_ms
# Initializing the LED pin - the number will be the GPIO pin of the LED
# Some boards have onboard LEDs connected to Pin 2
led = Pin(2, Pin.OUT)
# Turn the LED on by pulling the pin high (3.3 V)
led.on()
# Wait 1000ms (1 second)
sleep_ms(1000)
# Turn the LED off by pulling the pin low (0 V)
led.off()
# Wait and turn back on
sleep_ms(1000)
led.on()
print("Done") # Print to the terminal
```

3. Your LED will turn on, then off, then remain on
4. The ESP32 will run the code and switch to the terminal

Endless-Loop program

1. Paste the Code bellow into **test.py** and press the Run button

```
from machine import Pin
from time import sleep_ms
led = Pin(2, Pin.OUT)
while True: # An endless loop
  # Note the indent (Python uses this to determine when inside of the loop ends
  print("Loop")
  led.on()
  sleep_ms(1000)
  led.off()
  sleep_ms(1000)
print("Done") # This will never be reached
```

2. The program will run and eventually timeout. Since the loop is endless.
3. Programs like these can only be shut-down via a Reset (Rescue button)
4. A single program like this can be put into **main.py** to be run at start-up. A better options is available

Threaded Endless-Loop program

1. Paste the Code bellow at the end of **main.py**, **Save**, and press the **Reset** button

```
import _thread
# We will define a function
def foo():
  # note the indent
  print( "Hello" )
  # import our libraries inside the function
  from machine import Pin
  from time import sleep_ms
  led = Pin(2, Pin.OUT)
  while True: # An endless loop in a _thread
    # note the second indent
    print( "Loop" )
    led.on()
    sleep_ms(500)
    led.off()
    sleep_ms(500)
# This creates a _thread that does not block other functions
_thread.start_new_thread(foo,())
print( "Done" ) # This will print now
# Create more functions here - replace 'foo' with any other name
```

2. The ESP32 will reboot and start blinking and the print functions will spam the terminal
3. Remove all lines with the print( ) function and reset again to make it run quietly
4. To turn it off, comment out the _thread.start[…] line by placing a **#** in front

NOTE: Threaded programs should be called from **boot.py** or **main.py**
MicroIDE eliminates threaded programs run from the browser to conserve RAM

Limited-Loop program

1. Paste the Code bellow at the end of **test.py** and press the **Run** button

```
print( "Program Start" )
print( "X,X*2,10/X,100-X" ) # Print labels
# creates a loop where x= 1,6,11...96
# range( start , end , increment=1 )
# if  increment is skipped it will be set to 1
for x in range( 1 , 100 , 5 ):
  # note the indent
  # {} indicates an placeholder in the string
  # .format( ) takes variables and inserts them into the {} in order
  # This will print a table with x and some functions
  print( "{},{},{},{}".format( x , x *2 , 10 / x , 100 - x ) )

print( "Program End" )
```

2. A bunch of numbers should appear between Program Start and Program End
3. Switch to Graph in the IDE to see the values plotted (first column is x-axis)
4. This technically is a single run program with a limited loop inside