Nilcons Labs, Sept. 14, 2019

Before we even begin...

LET'S TRY IT!

1) Turn it on with the tiny slider switch on the top.
2) Read the text and be amazed.
3) Plug it in with the micro USB cable to your computer.
4) Wait for it to show up as a drive called CIRCUITPY. (If you are on a Linux where it doesn't mount automatically, I'm sure you know how to mount it.)
5) On the drive, look for the file called 'Europython 2019.py' and rename it to something else.
6) See the device automatically reload and show something else...
7) Use the buttons and figure out how to play.

Can you answer this?

First

- The file names from the drive are listed in the menu. How are they ordered?
- Can you find in which file and where this order is specified?

Can you answer this?

Second

- Where did the initially scrolling text EuroPython 2019 came from? Can you find it in the code?
- Search, grep and visual recognition system all allowed on the mounted drive...
- ...or on my GitHub repo. These are the files on your PewPew, I promise: https://github.com/napszel/PewPew/tree/master/original_files

Can you answer this?

Third

- So it came from the file name? Then why didn’t the new file name started scrolling after rename?

Introduction

PewPew for everyone

- EuroPython 2019, every participant gets one
- Special made, unique design and hardware for the event
- PewPew 10.2 - 2017, Radomir Dopieralski
- Main goals of the design:
  - cheap
  - simple
  - for kids
  - for game programming
- standalone, hand-held
- pre-installed with software AND games
- needs no additinal software or hardware (just AAA batteries)

Hardware

[]

- USB port - can NOT be powered from it
- Two AAA batteries - fries if you put them in the wrong way
- 64 led screen - 8*8 blue leds; made of plastic
- Only one color but 3 brightnesses for each
- 6 buttons - differentiated if you press some together
- Does not have sound
- RAM? Yes. (See later)
- 12-pin connector interface - GPIO pins - connect electronics such as neopixels, a speaker, sensors, etc.
- Atmel SAMD21 Controller

Software

()

- Includes the CircuitPython interpreter (Python for microcontrollers)
- Absolutely no additional software needed to be installed on the device or the computer
- Same as on Adafruit devices -> the drivers for connected electronics should work
- Comes with menu system main.py. Scrolls the file names and lets you select and run them.
- Actually, it searches for code.txt, code.py, main.txt, main.py in this order and it runs the first that it finds.
- If the file is edited and saved, it automatically re-runs the code.

Development

Intended development process

1) You don’t touch the main.py
2) Instead, create a code.py
3) Develop your game in there
4) Enjoy the automatic re-start process at each save
5) After done, give it a proper name
6) The main.py’s menu system will pick it up
7) Run your game from the menu from now on

Let’s write some code

Hello Pixel!

1) On the CIRCUITPY drive, create a file called code.py and open it with your favorite text editor. Write some code to light a pixel:

```
import pew
pew.init()
screen = pew.Pix()
screen.pixel(3, 3, 2) # x=3, y=3, brightness=2
pew.show(screen)
```

2) Add a `while True:` before `pew.show(screen)` so the pixel stays lit. Quick questions: a) Are the display coordinates zero indexed? b) What are the valid brightness numbers? c) What happens if you give a too high brightness? d) What happens if you leave a syntax error in your program?

Let’s get seri(al)ous

Connect to the serial console

- Output, like print messages and errors are sent to the serial console over USB
- Connect to the serial console:https://learn.adafruit.com/welcome-to-circuitpython/advanced-serial-console-on-mac-and-linux
- Something like: Linux `screen /dev/ttyACM0 115200`. Mac: `screen /dev/cu.usbmodem142101 115200`.
- After connected: Add a `print(''Hello World!'')` to your code and see it print.
- Then try a syntax error and see what happens.

REPL

Read Evaluate Print Loop

- If the running program ends in error or you interrupt it with Ctrl+C -> you end up in a REPL console
- Interactive Python console to try commands instantly.
- Try these:

`help()`
`help(''modules'')` -> list modules
`import board`
`dir(board)` -> list available pins
`import gc`
`gc.mem_free()` -> available memory in BYTES

- Ctrl+D to continue execution
- Ctrl+A k to exit

Memory

I don’t know how much but not much

- What to do if not enough?
- You will get a ’MemoryError’
- Reset the board -> it reallocates memory
- Use the .mpy version of libs instead of the .py ones
- Shorten your code: no comments, move functions to .mpy libs
- The order of lib imports might matter too!
- Interestingly: re-coping all the original files to the device failed.

MU

The recommended editor for beginners

- Simple, user and kid friendly UI with minimal features
- Connects to the board: Adafruit's dev board and BCC micro:bit, Pygame Zero. Or just runs python 3 code.
- Shows the serial console output built in.
- Works on Mac, Linux, Win and RaspberryPi.
- Has syntax check, warnings and helps messages.

Implementation ideas

- No colors and only 3 brightnesses? How about the main character blinking? -> sokoban.py
- Higher color numbers give option to smart pixel type calculations. -> sokoban.py
- To prevent double move on one click use pressing=False logic. -> draw.py
- Break from top level while True to stop the exection and go back to the menu.

Links

- Buy: https://makerfabs.com/pewpew-standalone.html
- Official website, Library Reference: https://pewpew.readthedocs.io
- Pewmulator: https://github.com/pewpew-game/pew-pygame
- Connect to the serial console: https://learn.adafruit.com/welcome-to-circuitpython/advanced-serial-console-on-mac-and-linux
- REPL: https://learn.adafruit.com/welcome-to-circuitpython/the-repl
- CircuitPython bundle: https://github.com/adafruit/Adafruit_CircuitPython_Bundle/releases
- Mu editor: https://codewith.mu