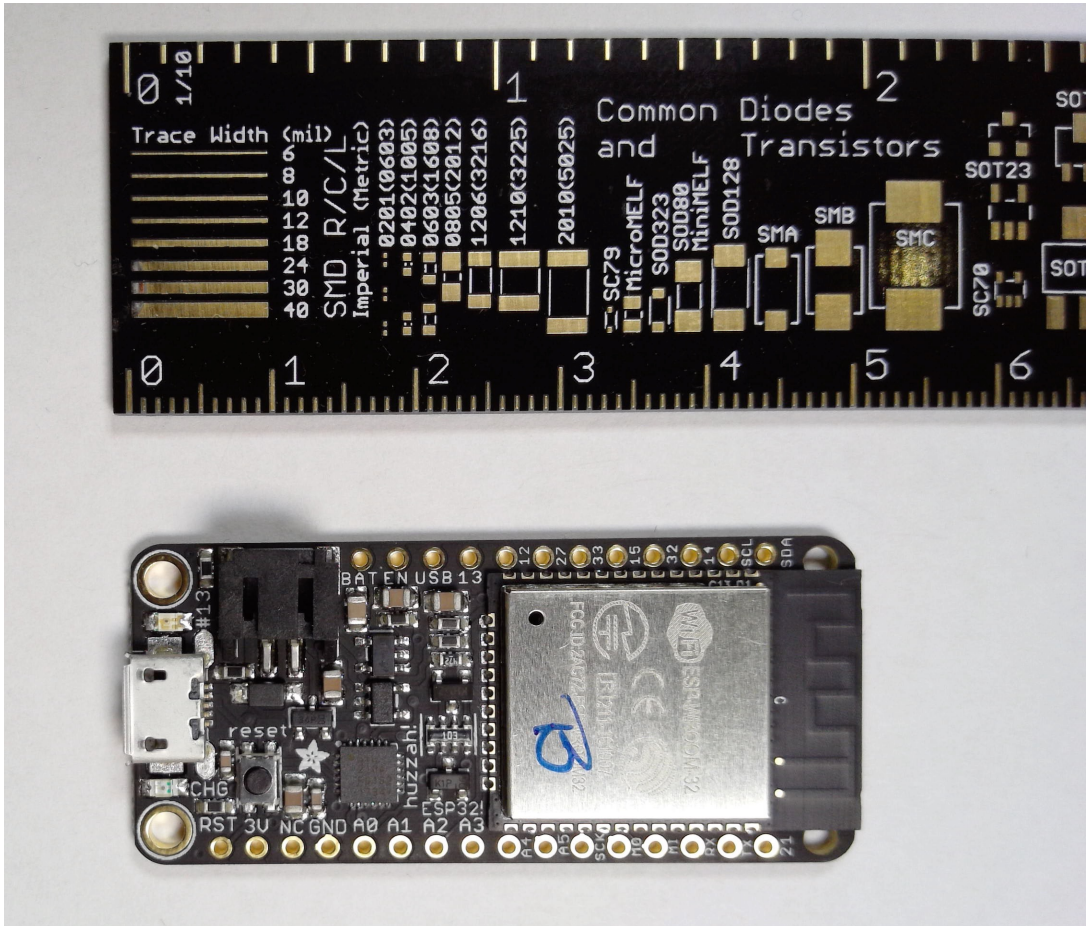
Electronics for IoT

Python Primer

Bernhard E. Boser
University of California, Berkeley
boser@eecs.berkeley.edu

Last Time


Common Diodes and Transistors
Trace Width (mil)
SMD R/C/L
Imperial (Metric)
0201(0603)
0402(1005)
0603(1608)
0805(2012)
1206(3216)
1210(3225)
2010(5025)
SC79
MicroMELF
SOD323
SOD80
MiniMELF
SOD128
SMA
SMB
SMC
SOT23
SC70
BAT EN USB 13
reset
CHG
RST 3V NC GND A0 A1 A2 A3
ESP32!

Programming

- Assembly
- C
- Python

Python

Language Rank	Types	Spectrum Ranking
1. Python		100.0
2. C		100.0
3. Java		99.4
4. C++		96.9
5. C#		88.6
6. R		88.1
7. JavaScript		85.3
8. PHP		81.1
9. Go		75.7
10. Swift		74.3
11. Arduino		72.4
12. Ruby		72.0
13. Assembly		71.7
14. Matlab		68.6
15. Scala		68.0
16. HTML		67.1

IEEE SPECTRUM

Python

Python

Python

```
print("Connecting to broker", BROKER, "...")
mqtt = MQTTClient(BROKER, user=USER, password=PWD)

print("Connected!")

def mqtt_callback(topic, msg):
    print("mqtt got topic={}, msg={}".format(topic, msg))

mqtt.set_callback(mqtt_callback)
mqtt.subscribe("iot49/a")
mqtt.subscribe("iot49/b")

try:
    for i in range(10000):
        mqtt.publish("iot49/send", "Hello {}".format(i))
        mqtt.check_msg()
        sleep(1)
finally:
    mqtt.disconnect()
```

Runs “everywhere”

- PC, Mac, Linux, ESP32, …
- Versions:
  - Python 2.7
  - Python 3.4+
    - MicroPython (missing some libraries, adding others)
    - Use Python 3 on host!
    - Most glaring difference:

```
print “Hello World!”
print(“Hello World!”)
```

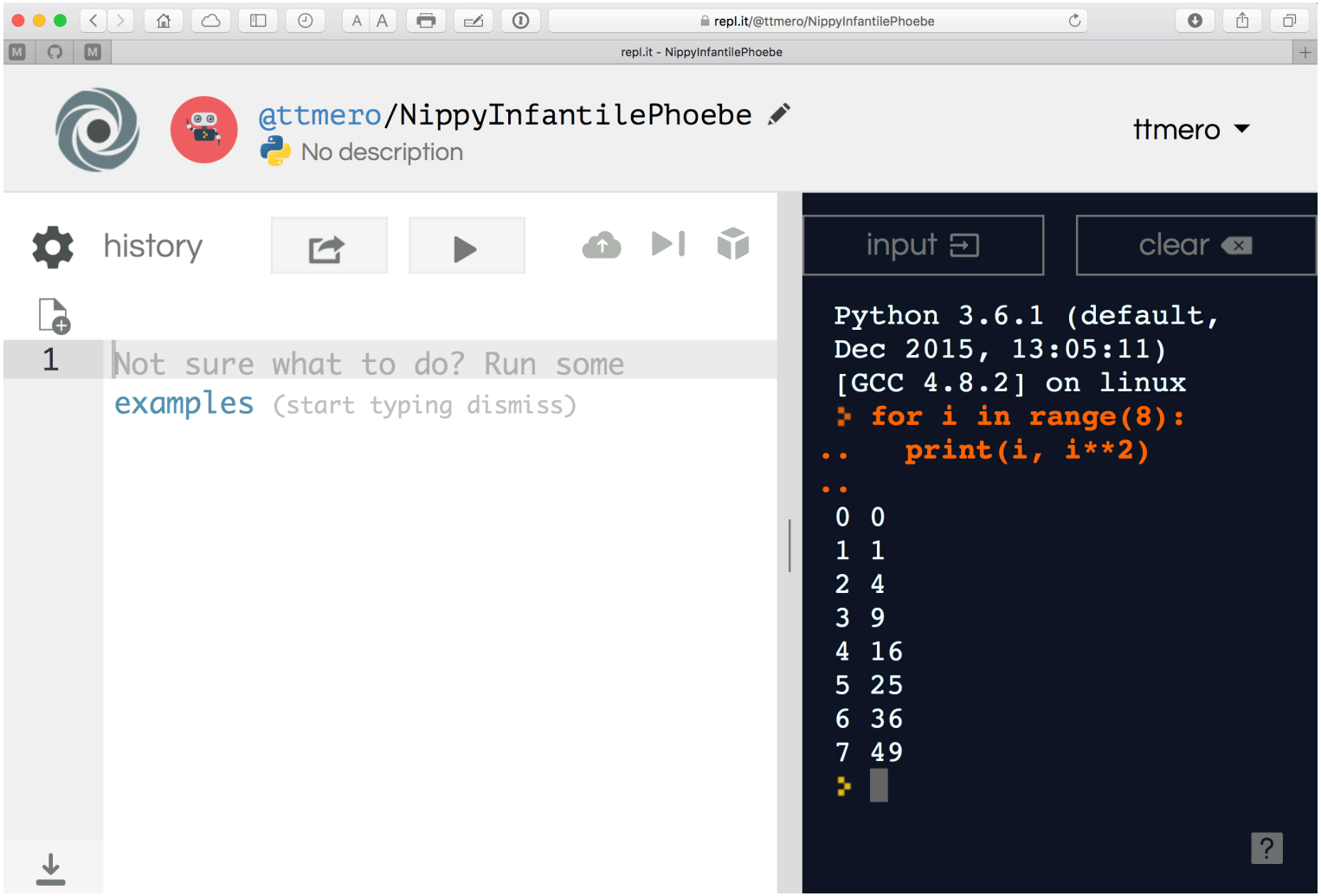
OS

- Windows, MacOS, Linux, …
- What's the OS on the ESP32?
  - None! (almost, FreeRTOS)
- REPL
  - Read evaluate print loop
  - Like a command window,
    but speaks "Python" rather than bash, …

Laptop …

```
MacPro-15:mcu boser$ python
Python 3.6.3 |Anaconda custom (x86_64)| (default, Oct 27 2017, 12:14:30)
[GCC 4.2.1 Compatible Clang 4.0.1 (tags/RELEASE_401/final)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 2**30
1073741824
>>>
```

Web … (http://repl.it)

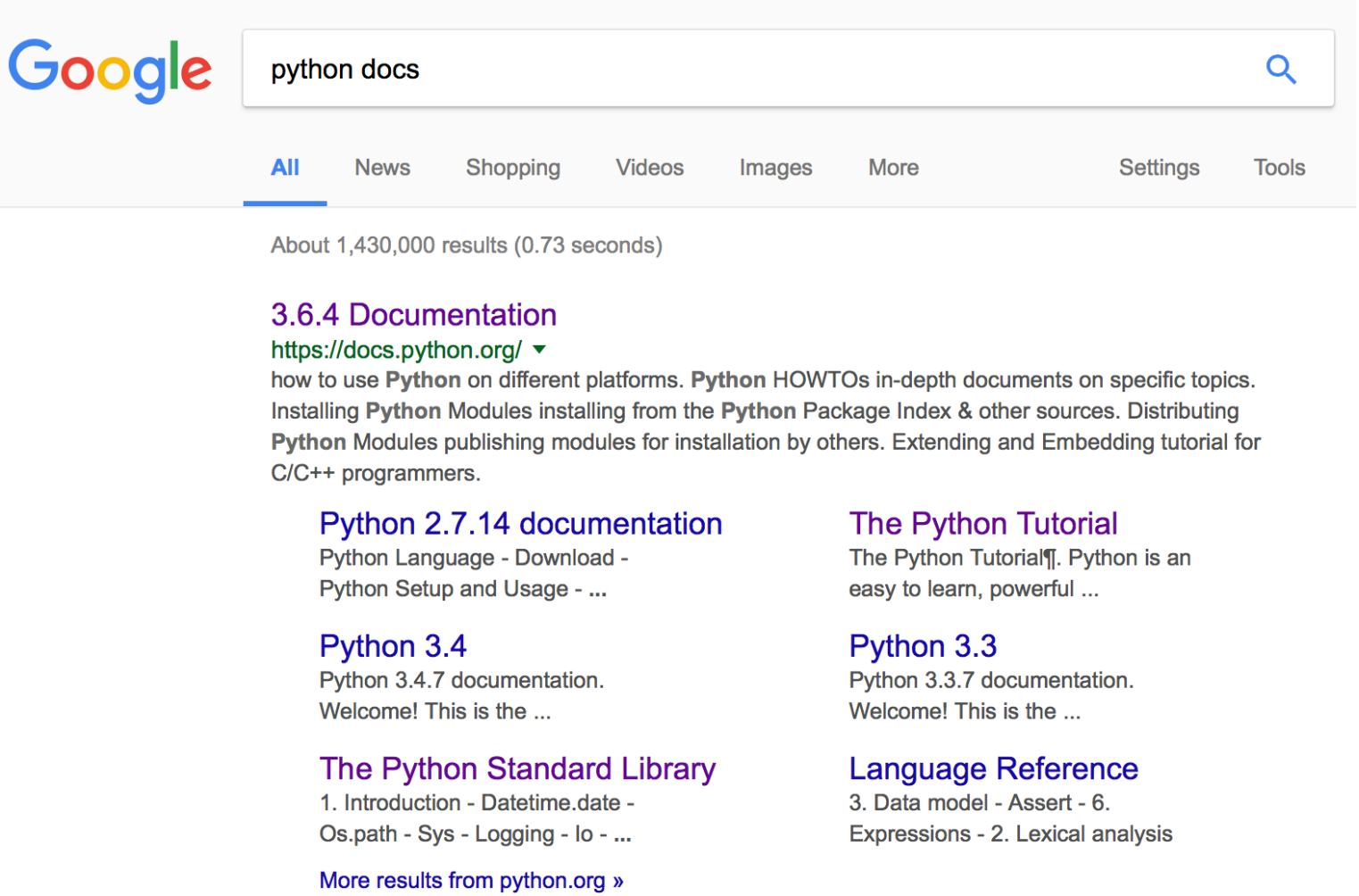


ESP32 Microcontroller

```
MacPro-15:mcu boser$ shell49
Loading configuration '/Users/boser/Dropbox/Files/Class/49/.shell49_rc.py'
Connecting via serial to /dev/cu.SLAB_USBtoUART @ 115200 baud ...
Connected to 'GPIO' (id=30aea43081a8), synchonizing time ...
Welcome to shell49. Type 'help' for information; Control-D to exit.

/Users/boser/Dropbox/Files/Class/49/esp32/mcu> repl
Entering REPL. Use Control-X to exit.
   Soft reset:  Control-D or sys.exit()
   Hard reset:  Reset button on board or machine.reset()
>
MicroPython ESP32_LoBo_v3.1.5 - 2017-01-16 on HUZZAH32 with ESP32
Type "help()" for more information.
>>>
>>> 2**1234
29581122460809862906004469571610359078633968713537299223955620705065735079623892426105383724837805018644364775907
09559931208208993303817609370272124828409449413621106654437751834957268119292038611820152183238920773559833931912
08928867652655993602487903113708549402668624521100611794270340232766099317098048887493809023127398253860618772619
03500988327294112954464011183718 4
>>>
```

Python Docs


Google
python docs
All
News
Shopping
Videos
Images
More
Settings
Tools
About 1,430,000 results (0.73 seconds)
3.6.4 Documentation
https://docs.python.org/
how to use Python on different platforms. Python HOWTOs in-depth documents on specific topics. Installing Python Modules installing from the Python Package Index & other sources. Distributing Python Modules publishing modules for installation by others. Extending and Embedding tutorial for C/C++ programmers.
Python 2.7.14 documentation
Python Language - Download - Python Setup and Usage - ...
The Python Tutorial
The Python Tutorial¶. Python is an easy to learn, powerful ...
Python 3.4
Python 3.4.7 documentation. Welcome! This is the ...
Python 3.3
Python 3.3.7 documentation. Welcome! This is the ...
The Python Standard Library
1. Introduction - Datetime.date - Os.path - Sys - Logging - Io - ...
Language Reference
3. Data model - Assert - 6. Expressions - 2. Lexical analysis
More results from python.org »

Python Docs

Python 3.6.4 documentation

Welcome! This is the documentation for Python 3.6.4.

Parts of the documentation:

What's new in Python 3.6?
or all "What's new" documents since 2.0

Tutorial
start here

Library Reference
keep this under your pillow

Language Reference
describes syntax and language elements

Python Setup and Usage
how to use Python on different platforms

Python HOWTOs
in-depth documents on specific topics

Installing Python Modules
installing from the Python Package Index & other sources

Distributing Python Modules
publishing modules for installation by others

Extending and Embedding
tutorial for C/C++ programmers

Python/C API
reference for C/C++ programmers

FAQs
frequently asked questions (with answers!)

Python Docs

- 13 MBytes (compressed)
- I have not read it either!

How to survive EE49 …

- … if I never programmed Python before
  - My situation 12 month ago!

Python for EE49

- Basic stuff
  - print, comments
- Data Types
  - Numbers, strings, list, dict
  - Indexing
- Expressions
- Variables
- Statements
  - If, for, while
- Library
  - Import
- Functions

Examples …

Examples

- Numbers
- Even/odd
- Division
- ints, floats
- Conversion, int, round
- Libraries: e.g. math.sin, math.pi

Editors – Plain Vanilla

mqtt.py — Edited

```
mqtt = MQTTClient(BROKER, user=USER, password=PWD)

print("Connected!")

def mqtt_callback(topic, msg):
    print("mqtt got topic={}, msg={}".format(topic, msg))

mqtt.set_callback(mqtt_callback)
mqtt.subscribe("iot49/a")
mqtt.subscribe("iot49/b")

try:
    for i in range(10000):
        mqtt.publish("iot49/send", "Hello {}".format(i))
        mqtt.check_msg()
        sleep(1)
finally:
    mqtt.disconnect()
```

Editors – Syntax Highlighting

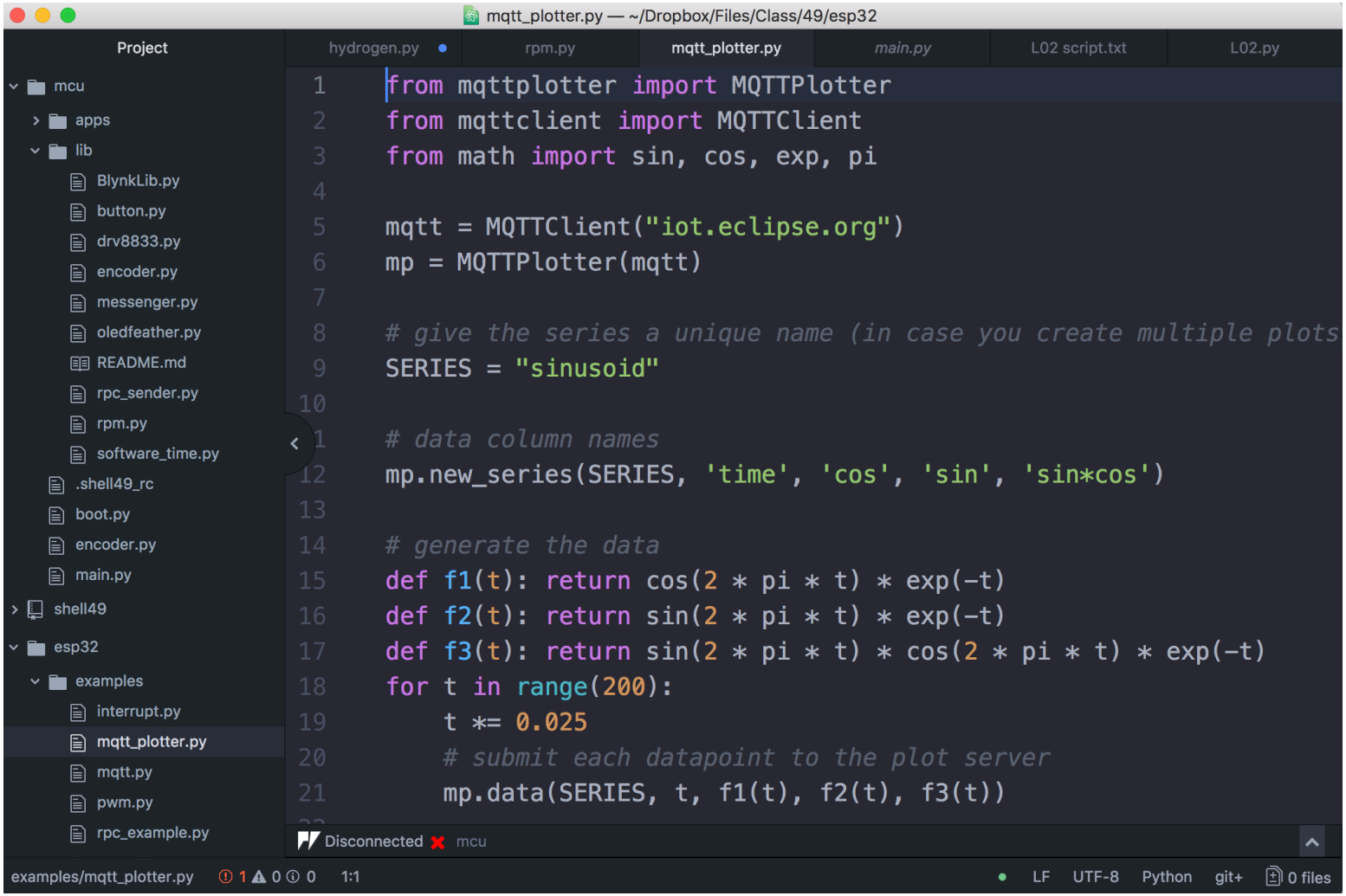
mqtt_plotter.py — examples

Add License

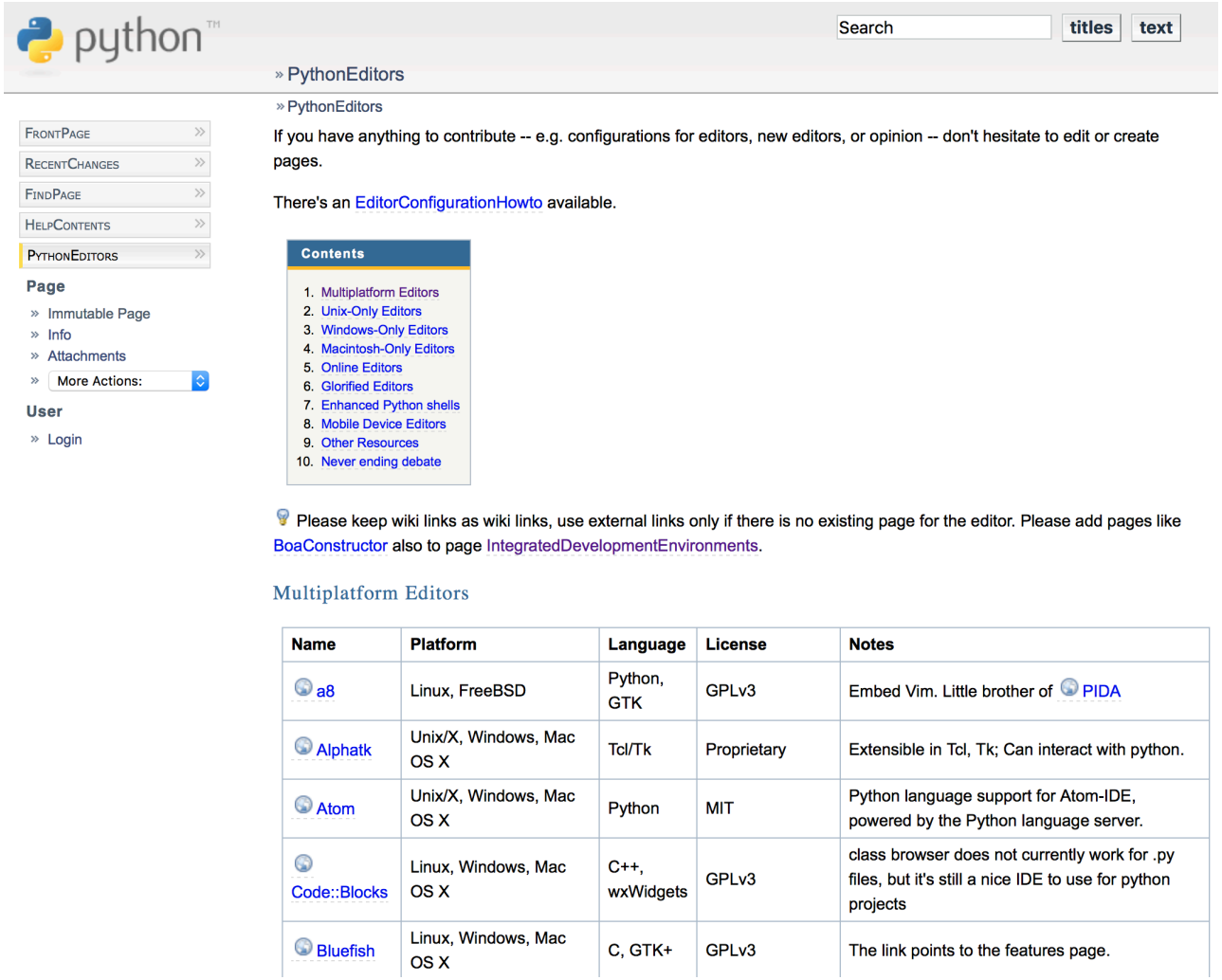
```
1  from mqttplotter import MQTTPlotter
2  from mqttclient import MQTTClient
3  from math import sin, cos, exp, pi
4
5  mqtt = MQTTClient("iot.eclipse.org")
6  mp = MQTTPlotter(mqtt)
7
8  # give the series a unique name (in case you create multiple plots)
9  SERIES = "sinusoid"
10
11 # data column names
12 mp.new_series(SERIES, 'time', 'cos', 'sin', 'sin*cos')
13
14 # generate the data
15 def f1(t): return cos(2 * pi * t) * exp(-t)
16 def f2(t): return sin(2 * pi * t) * exp(-t)
17 def f3(t): return sin(2 * pi * t) * cos(2 * pi * t) * exp(-t)
18 for t in range(200):
19     t *= 0.025
20     # submit each datapoint to the plot server
21     mp.data(SERIES, t, f1(t), f2(t), f3(t))
22
23 # save data as pkl document
24 # see plot_load_pkl.py for an example of loading it back into python
25 mp.save_series(SERIES)
26
27 # create a plot, default dir is $IoT49
28 mp.plot_series(SERIES,
29     filename="bin/examples/mqtt_plotter_example.pdf",
30     xlabel="Time [s]",
31     ylabel="Voltage [mV]",
```

Line: 1 | Python | Soft Tabs: 4

Editors - IDE


mqtt_plotter.py — ~/Dropbox/Files/Class/49/esp32
Project
mcu
apps
lib
BlynkLib.py
button.py
drv8833.py
encoder.py
messenger.py
oledfeather.py
README.md
rpc_sender.py
rpm.py
software_time.py
.shell49_rc
boot.py
encoder.py
main.py
shell49
esp32
examples
interrupt.py
mqtt_plotter.py
mqtt.py
pwm.py
rpc_example.py
hydrogen.py
rpm.py
mqtt_plotter.py
main.py
L02 script.txt
L02.py
1 from mqttplotter import MQTTPlotter
2 from mqttclient import MQTTClient
3 from math import sin, cos, exp, pi
4
5 mqtt = MQTTClient("iot.eclipse.org")
6 mp = MQTTPlotter(mqtt)
7
8 # give the series a unique name (in case you create multiple plots
9 SERIES = "sinusoid"
10
11 # data column names
12 mp.new_series(SERIES, 'time', 'cos', 'sin', 'sin*cos')
13
14 # generate the data
15 def f1(t): return cos(2 * pi * t) * exp(-t)
16 def f2(t): return sin(2 * pi * t) * exp(-t)
17 def f3(t): return sin(2 * pi * t) * cos(2 * pi * t) * exp(-t)
18 for t in range(200):
19 t *= 0.025
20 # submit each datapoint to the plot server
21 mp.data(SERIES, t, f1(t), f2(t), f3(t))
Disconnected mcu
examples/mqtt_plotter.py 1 0 0 1:1 LF UTF-8 Python git+ 0 files

Choices, choices …



python™ Search titles text

» PythonEditors

FrontPage
RecentChanges
FindPage
HelpContents
PythonEditors

Page
» Immutable Page
» Info
» Attachments
» More Actions:

User
» Login

» PythonEditors

If you have anything to contribute -- e.g. configurations for editors, new editors, or opinion -- don't hesitate to edit or create pages.

There's an EditorConfigurationHowto available.

Contents

1. Multiplatform Editors
2. Unix-Only Editors
3. Windows-Only Editors
4. Macintosh-Only Editors
5. Online Editors
6. Glorified Editors
7. Enhanced Python shells
8. Mobile Device Editors
9. Other Resources
10. Never ending debate

Please keep wiki links as wiki links, use external links only if there is no existing page for the editor. Please add pages like BoaConstructor also to page IntegratedDevelopmentEnvironments.

Multiplatform Editors

Name	Platform	Language	License	Notes
a8	Linux, FreeBSD	Python, GTK	GPLv3	Embed Vim. Little brother of PIDA
Alphatk	Unix/X, Windows, Mac OS X	Tcl/Tk	Proprietary	Extensible in Tcl, Tk; Can interact with python.
Atom	Unix/X, Windows, Mac OS X	Python	MIT	Python language support for Atom-IDE, powered by the Python language server.
Code::Blocks	Linux, Windows, Mac OS X	C++, wxWidgets	GPLv3	class browser does not currently work for .py files, but it's still a nice IDE to use for python projects
Bluefish	Linux, Windows, Mac OS X	C, GTK+	GPLv3	The link points to the features page.

More Python

https://docs.python.org/3/tutorial/

The Python Tutorial

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

The Python interpreter and the extensive standard library are freely available in source or binary form for all major platforms from the Python Web site, https://www.python.org/, and may be freely distributed. The same site also contains distributions of and pointers to many free third party Python modules, programs and tools, and additional documentation.

The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.

This tutorial introduces the reader informally to the basic concepts and features of the Python language and system. It helps to have a Python interpreter handy for hands-on experience, but all examples are self-contained, so the tutorial can be read off-line as well.

For a description of standard objects and modules, see The Python Standard Library. The Python Language Reference gives a more formal definition of the language. To write extensions in C or C++, read Extending and Embedding the Python Interpreter and Python/C API Reference Manual. There are also several books covering Python in depth.

This tutorial does not attempt to be comprehensive and cover every single feature, or even every commonly used feature. Instead, it introduces many of Python's most noteworthy features, and will give you a good idea of the language's flavor and style. After reading it, you will be able to read and write Python modules and programs, and you will be ready to learn more about the various Python library modules described in The Python Standard Library.

Exercises

http://www.practicepython.org

PRACTICE PYTHON

Beginner Python exercises

Follow

Follow @practice_python

Home Why Practice Python? Why Chilis? Resources for learners Exercises Blog About

Welcome to Practice Python! There are over 30 beginner Python exercises just waiting to be solved. Each exercise comes with a small discussion of a topic and a link to a solution. New exercise are posted monthly, so check back often, or follow on Feedly, Twitter, or your favorite RSS reader. To get started right away, read more about Practice Python or go straight to Exercise 1!

Latest exercise: Exercise 36 on 02 April 2017

Latest solution: Solution 35 on 19 March 2017

Latest blog post: *Installing Python to Get Started* on 24 March 2017

Primestudent FREE TWO-DAY SHIPPING FOR COLLEGE STUDENTS Learn more ›

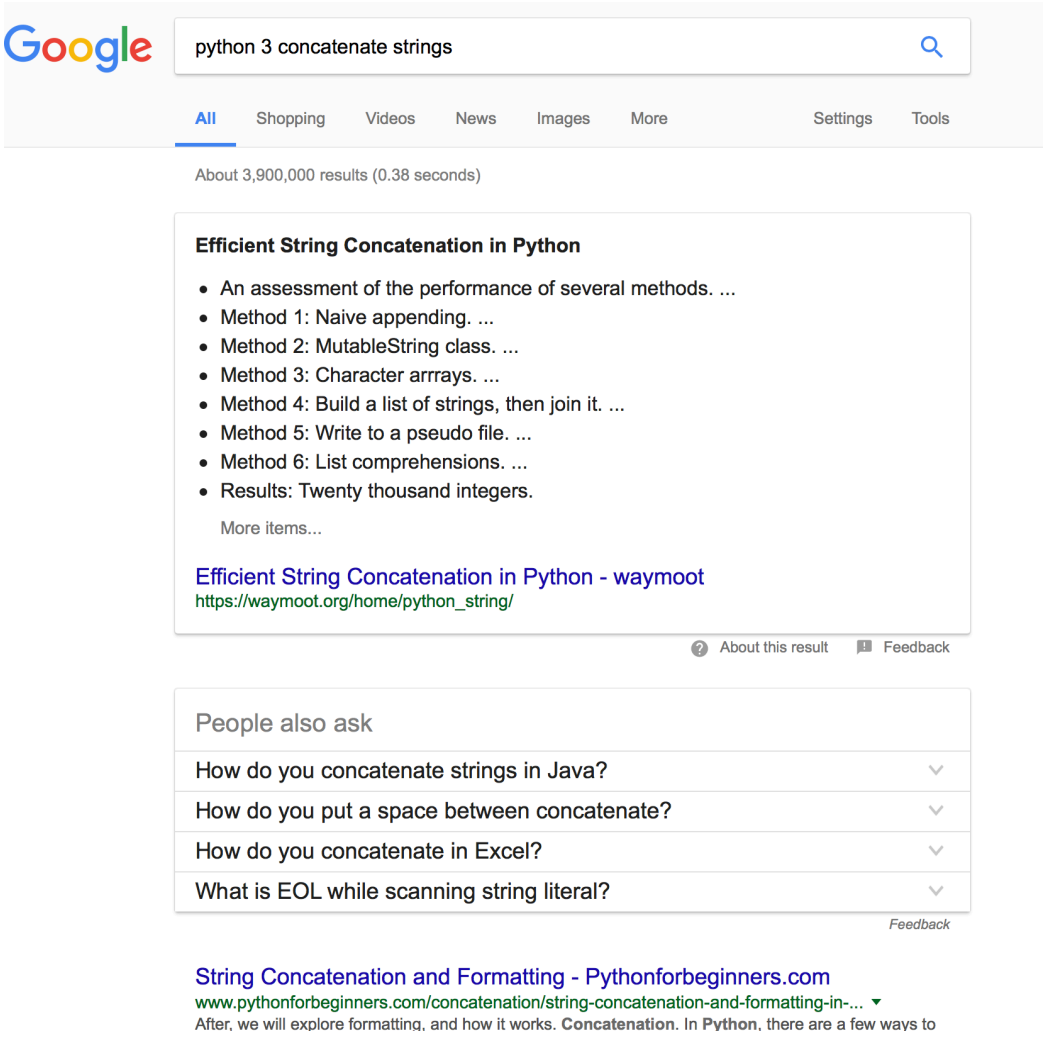
All Exercises

- 1: Character Input
- 2: Odd Or Even

All Solutions

- 1: Character Input Solutions
- 2: Odd Or Even Solutions

Help


Google
python 3 concatenate strings
All
Shopping
Videos
News
Images
More
Settings
Tools
About 3,900,000 results (0.38 seconds)
Efficient String Concatenation in Python
An assessment of the performance of several methods. ...
Method 1: Naive appending. ...
Method 2: MutableString class. ...
Method 3: Character arrrays. ...
Method 4: Build a list of strings, then join it. ...
Method 5: Write to a pseudo file. ...
Method 6: List comprehensions. ...
Results: Twenty thousand integers.
More items...
Efficient String Concatenation in Python - waymoot
https://waymoot.org/home/python_string/
About this result
Feedback
People also ask
How do you concatenate strings in Java?
How do you put a space between concatenate?
How do you concatenate in Excel?
What is EOL while scanning string literal?
Feedback
String Concatenation and Formatting - Pythonforbeginners.com
www.pythonforbeginners.com/concatenation/string-concatenation-and-formatting-in-...
After, we will explore formatting, and how it works. Concatenation. In Python, there are a few ways to

Blinking LED

Microcontroller I/O

GPIO	ALT	µPy	
	RESET		1
	3.3V		2
			3
	GND		4
26	DAC2	A0	5
25	DAC1	A1	6
34	ADC6	A2	7
39	ADC3	A3	8
36	ADC0	A4	9
4		A5	10
5	SCK	A16	11
18	MOSI	A17	12
19	MISO	A18	13
16		A19	14
17		A20	15
21		A21	16

µPy	ALT	GPIO	
	VBAT		28
	EN 3.3V		27
	VUSB		26
A12	LED	13	25
A11	BOOT	12	24
A10		27	23
A9	ADC5	33	22
A8		15	21
A7	ADC4	32	20
A6		14	19
A15	SCL	22	18
A14	SDA	23	17

Next Lecture

- Electricity!