

Writing Python Scripts

Simple Examples of the Power of Python for Beginners

I'm on a mission!

To get more people to use python.

- Not just for data analysis!
- Not just for work!



Overview

This talk is geared towards beginner python programmers.
We'll start more basic and slowly build to larger concepts.

1. Introduction to Scripting
2. Scripts
 - a. **Simple Adding Script:** input, sys.argv
 - b. **Time Logging:** input loops, datetime module, writing to a file
 - c. **Bulk Image Organizer:** pathlib.Path, PIL, datetime.strptime
 - d. **Jisho API:** APIs, requests, JSON
 - e. **EBook Writer:** ebooklib, beautifulsoup, HTML

Questions after each section and at the end



What is a script?



script (n.)

late 14c., "something written, a written document," earlier *scrite* (c. 1300), from

Eymology

<https://www.etymonline.com/word/script>



script (n.)

late 14c., "something written, a written document," earlier *scrite* (c. 1300), from Anglo-French *scrit*, Old French *escriit* "piece of writing, written paper; credit note,

Etymology

<https://www.etymonline.com/word/script>



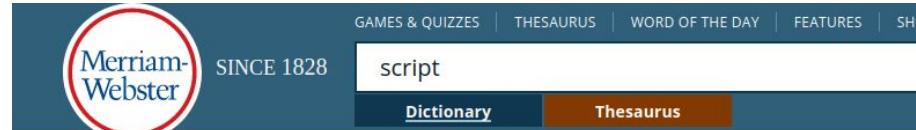
Etymology

<https://www.etymonline.com/word/script>

script (n.)

late 14c., "something written, a written document," earlier *scrite* (c. 1300), from Anglo-French *scrit*, Old French *escriit* "piece of writing, written paper; credit note, IOU; deed, bond" (Modern French *écrit*) and directly from Latin *scriptum* "a writing, book; law; line, mark," noun use of neuter past participle of *scribere* "to write"





Definition

<https://www.merriam-webster.com/dictionary/script>





SINCE 1828

script

Dictionary Thesaurus

GAMES & QUIZZES | THESAURUS | WORD OF THE DAY | FEATURES | SHOP

script noun (1)

 Save Word

\ 'skript \

plural **scripts**

Definition of *script* (Entry 1 of 3)

1 a : something written : TEXT



Definition

<https://www.merriam-webster.com/dictionary/script>



SINCE 1828

script

Dictionary Thesaurus

Definition

<https://www.merriam-webster.com/dictionary/script>



script noun (1)

 Save Word

\ 'skript \

plural **scripts**

Definition of *script* (Entry 1 of 3)

- 1 **a** : something written : **TEXT**
 - b** : an original or principal instrument or document
 - c** **(1)** : MANUSCRIPT sense 1
 - (2)** : the written text of a stage play, screenplay, or broadcast
specifically : the one used in production or performance
- 2 **a** : a style of printed letters that resembles handwriting
 - b** : written characters : HANDWRITING
 - c** : ALPHABET
- 3 : a plan of action
- 4 : *computing* : a sequence of instructions or commands for a computer to execute
especially : one that automates a small task (such as assembling or sorting a set of data)

// Two engineers, a biologist and a zoologist wrote a computer *script* that downloaded the lyrics, band names and genre categories from all songs on the website www.metalkingdom.net.
— Fedor Zarkhin



SINCE 1828

script

Dictionary Thesaurus

GAMES & QUIZZES | THESAURUS | WORD OF THE DAY | FEATURES | SHOP

script noun (1)

 Save Word

\ 'skript \

plural **scripts**

Definition of script (Entry 1 of 3)

4 computing: a sequence of instructions or commands for a computer to execute





SINCE 1828

script

Dictionary Thesaurus

GAMES & QUIZZES | THESAURUS | WORD OF THE DAY | FEATURES | SHOP

script noun (1)

 Save Word

\ 'skript \ plural **scripts**

Definition of *script* (Entry 1 of 3)

- 
- 4 *computing* : a sequence of instructions or commands for a computer to execute
especially : one that automates a small task (such as assembling or sorting a set of data)

Definition

<https://www.merriam-webster.com/dictionary/script>



SINCE 1828

script

Dictionary Thesaurus

script noun (1)

 Save Word

\ 'skript \ plural **scripts**

Definition of *script* (Entry 1 of 3)

- 4 *computing* : a sequence of instructions or commands for a computer to execute
especially : one that automates a small task (such as assembling or sorting a set of data)

// Two engineers, a biologist and a zoologist wrote a computer *script* that downloaded the lyrics, band names and genre categories from all songs on the website www.metalkingdom.net.

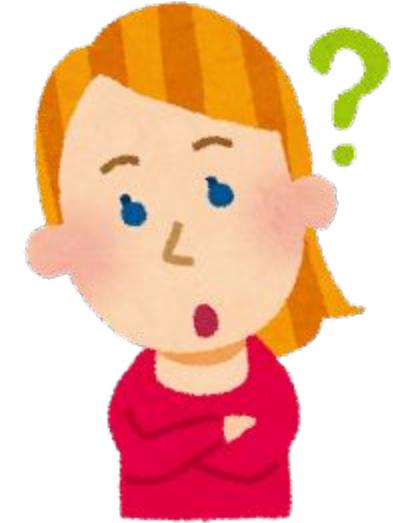
— Fedor Zarkhin



Definition

<https://www.merriam-webster.com/dictionary/script>

How to write a script?



Scripting Language

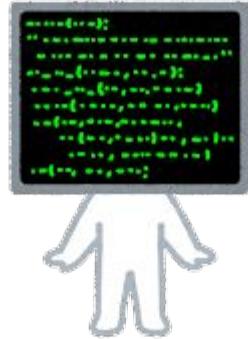
Bash / Shell



```
# example_bash_script.sh

#!/bin/bash
while true; do
    git status
    read -p "Do you wish to create $1? [y/n] " yn
    case $yn in
        [Yy]* ) break;;
        [Nn]* ) exit 1;;
        * ) echo "Please answer yes or no.";;
    esac
done
# echo "created $1"
git branch $1
git checkout $1
git push --set-upstream origin $1
```

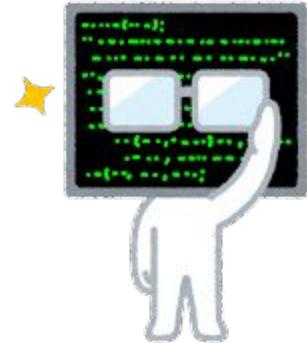
But we already know a scripting language!



But we already know a scripting language!

Python (programming language)

From Wikipedia, the free encyclopedia

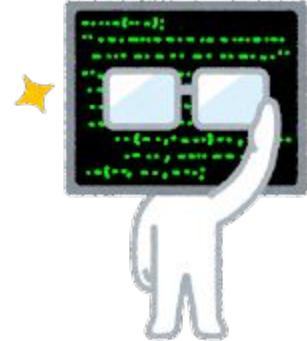


But we already know a scripting language!

Python (programming language)

From Wikipedia, the free encyclopedia

Python is a high-level, general-purpose programming language.

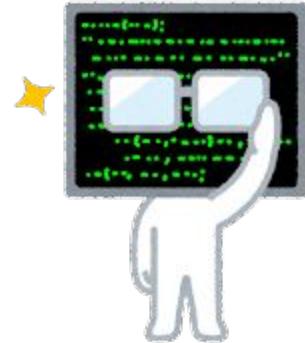


But we already know a scripting language!

Python (programming language)

From Wikipedia, the free encyclopedia

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation.

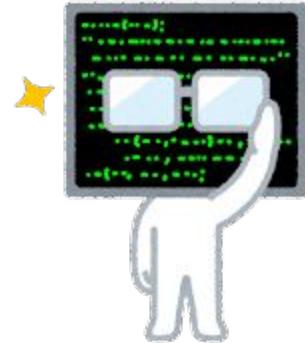


But we already know a scripting language!

Python (programming language)

From Wikipedia, the free encyclopedia

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Its language constructs and object-oriented approach aim to help programmers

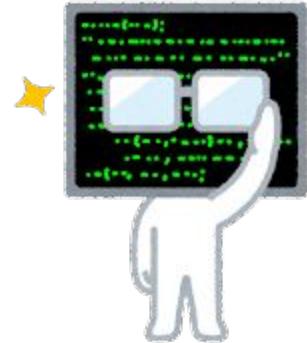


But we already know a scripting language!

Python (programming language)

From Wikipedia, the free encyclopedia

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small- and large-scale projects.^[30]



But we already know a scripting language!

Python (programming language)

From Wikipedia, the free encyclopedia

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small- and large-scale projects.^[30]



Python Example

<https://github.com/pandas-dev/pandas/blob/9a678df32112fdac7fa082954dd4b44f82d9301c/pandas/core/internals/managers.py#L62>

```
class BlockManager(DataManager):  
  
    def __setstate__(self, state):  
        def unpickle_block(values, mgr_locs, ndim: int):  
            # TODO(EA2D): ndim would be unnecessary with 2D EAs  
            return make_block(values, placement=mgr_locs, ndim=ndim)  
  
            if isinstance(state, tuple) and len(state) >= 4 and "0.14.1" in  
state[3]:  
                state = state[3]["0.14.1"]  
                self.axes = [ensure_index(ax) for ax in state["axes"]]  
                ndim = len(self.axes)  
                self.blocks = tuple(  
                    unpickle_block(b["values"], b["mgr_locs"], ndim=ndim)  
                    for b in state["blocks"]  
                )  
            else:  
                raise NotImplementedError("pre-0.14.1 pickles are no longer  
supported")  
  
            self._post_setstate()  
  
    def _post_setstate(self) -> None:  
        ...
```



Python Example

<https://github.com/pandas-dev/pandas/blob/9a678df32112fdac7fa082954dd4b44f82d9301c/pandas/core/internals/managers.py#L62>

```
class BlockManager(DataManager):  
  
    def __setstate__(self, state):  
        def unpickle_block(values, mgr_locs, ndim: int):  
            # TODO(EA2D): ndim would be unnecessary with 2D EAs  
            return make_block(values, placement=mgr_locs, ndim=ndim)  
  
            if isinstance(state, tuple) and len(state) >= 4 and "0.14.1" in  
state[3]:  
                state = state[3]["0.14.1"]  
                self.axes = [ensure_index(ax) for ax in state["axes"]]  
                ndim = len(self.axes)  
                self.blocks = tuple(  
                    unpickle_block(b["values"], b["mgr_locs"], ndim=ndim)  
                    for b in state["blocks"]  
                )  
            else:  
                raise NotImplementedError("pre-0.14.1 pickles are no longer  
supported")  
  
            self._post_setstate()  
  
    def _post_setstate(self) -> None:  
        ...
```

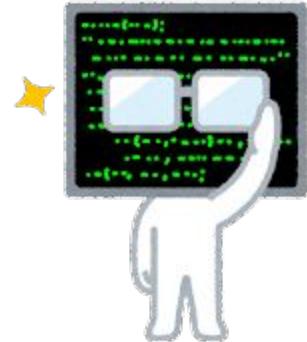


But we already know a scripting language!

Python (programming language)

From Wikipedia, the free encyclopedia

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small- and large-scale projects.^[30]



```
# simple_script.py  
print(5 + 10)
```

Small Scale



Small Scale

```
# simple_script.py  
print(5 + 10)  
  
# command line  
python simple_script.py  
15
```



Going Further: Simple Script

Simple Script

```
# simple_script.py  
print(5 + 10)  
  
# command line  
python simple_script.py  
15
```



```
# simple_script_1.py
user_number = input("What number? ")
print(user_number + 10)
```

Simple Script



Simple Script

```
# simple_script_1.py
user_number = input("What number? ")
print(user_number + 10)

# command line
$ python simple_script_1.py
What number? 10
Traceback (most recent call last):
  File "apps/simple_script/simple_script_1.py", line 2, in <module>
    print(user_number + 10)
TypeError: can only concatenate str (not "int") to str
```



Simple Script

```
# simple_script_2.py
user_number = input("What number? ")
print(int(user_number) + 10)
```

```
# command line
$ python simple_script_2.py
What number? 10
20
```



```
# simple_script_3.py
import sys
print(int(sys.argv[1]) + 10)
```

Simple Script



Simple Script

```
# simple_script_3.py
import sys
print(int(sys.argv[1]) + 10)
```

```
# command line
$ python simple_script_3.py 10
20
```



Simple Script

```
# simple_script_3.py
import sys

print(sys.argv)  # for inspection
print(int(sys.argv[1]) + 10)

# command line
$ python simple_script_3.py 10
['simple_script_3.py', '10']
20
```



Conclusion

1. scripts
2. input
3. sys.argv

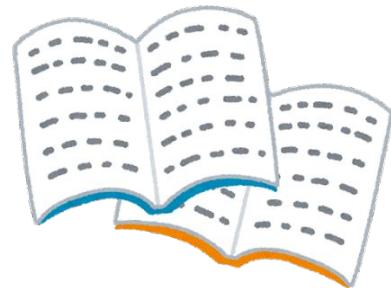
```
# simple_script_3.py
import sys

print(sys.argv)  # for inspection
print(int(sys.argv[1]) + 10)
```

```
# command line
$ python simple_script_3.py 10
['simple_script_3.py', '10']
20
```



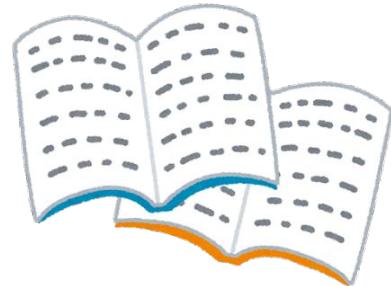
Time Logging (JLog): Demo



JLog (Demo)

```
# command line
$ python jlog.py
log: newday
note? yesterday I had coco curry for dinner

Yesterday, worked on tasks A B C
Today, will continue those tasks
Next meeting: tech talk 14:00
Next meeting:
[ ] will continue those tasks # 18:05
log: currently working on task A
18:08 - currently working on task A
log: q
$
```



JLog (Demo)

```
# jlog_output.txt
---04/20/2022 (Wednesday)--- yesterday I had coco curry for dinner
in 18:05
Yesterday, worked on tasks A B C
Today, will continue those tasks
> tech talk 14:00
--- 04/20/2022 (Wednesday) tech talk 14:00 ---
```

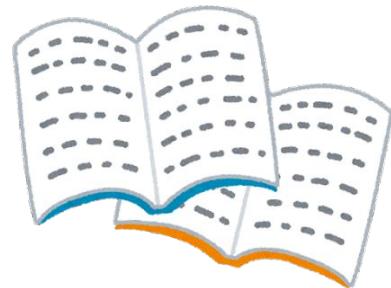
```
<<<<<<<<<<<<<<<<<<<<<<<<<
```

TODO:

```
[ ] will continue those tasks # 18:05
```

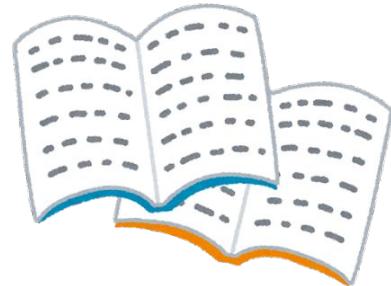
```
18:05 - finished checkin
```

```
18:08 - currently working on task A
```



```
# my_daily_log.txt  
---04/20/2022 (Wednesday) ---  
18:05 - finished checkin  
18:08 - currently working on task A
```

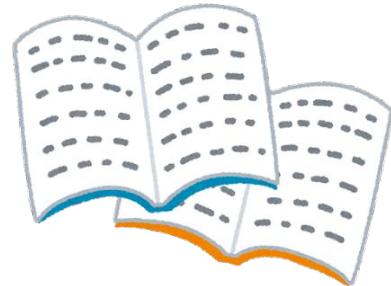
JLog Origins



```
# my_daily_log.txt
---04/20/2022 (Wednesday) ---
18:05 - finished checkin
18:08 - currently working on task A
```

JLog Goals

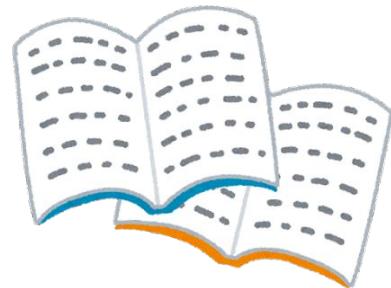
- Input loops
- Datetime module
- Writing to a file



JLog

```
# jlog_0.py
OUTPATH = "output/my_logs_0.txt"

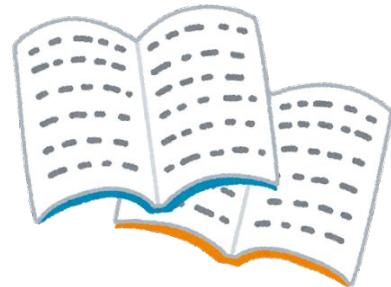
while True:
    user_input = input("log: ")
    with open(OUTPATH, mode="a") as f:
        print(user_input, file=f)
    print(user_input)
```



JLog

```
# jlog_0.py
OUTPATH = "output/my_logs_0.txt"

while True:
    user_input = input("log: ")
    with open(OUTPATH, mode="a") as f:
        print(user_input, file=f)
    print(user_input)
```



```
open(file, mode='r', buffering=-1, encoding=None, errors=None, newline=None,  
      closefd=True, opener=None)
```

Open *file* and return a corresponding [file object](#). If the file cannot be opened, an `OSError` is raised. See [Reading and Writing Files](#) for more examples of how to use this function.

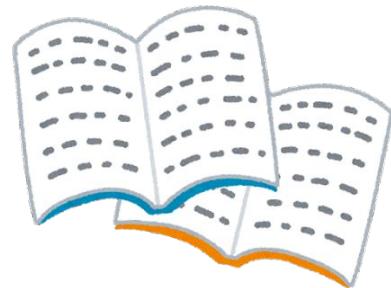
file is a [path-like object](#) giving the pathname (absolute or relative to the current working directory) of the file to be opened or an integer file descriptor of the file to be wrapped. (If a file descriptor is given, it is closed when the returned I/O object is closed unless *closefd* is set to `False`.)

mode is an optional string that specifies the mode in which the file is opened. It defaults to `'r'` which means open for reading in text mode. Other common values are `'w'` for writing (truncating the file if it already exists), `'x'` for exclusive creation, and `'a'` for appending (which on some Unix systems, means that *all* writes append to the end of the file regardless of the current seek position). In text mode, if *encoding* is not specified the encoding used is platform-dependent: `locale.getpreferredencoding(False)` is called to get the current locale encoding. (For reading and writing raw bytes use binary mode and leave *encoding* unspecified.) The available modes are:

Character	Meaning
<code>'r'</code>	open for reading (default)
<code>'w'</code>	open for writing, truncating the file first
<code>'x'</code>	open for exclusive creation, failing if the file already exists
<code>'a'</code>	open for writing, appending to the end of file if it exists
<code>'b'</code>	binary mode
<code>'t'</code>	text mode (default)
<code>'+'</code>	open for updating (reading and writing)

JLog

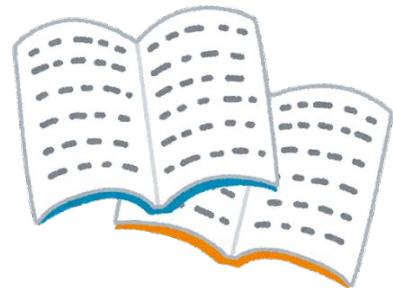
<https://docs.python.org/3/library/functions.html#open>



`open(file, mode='r',`

JLog

<https://docs.python.org/3/library/functions.html#open>



Character	Meaning
'r'	open for reading (default)
'a'	open for writing, appending to the end of file if it exists

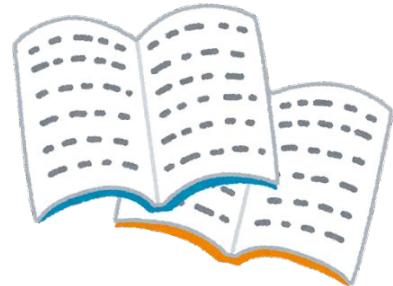
JLog

```
# jlog_0.py
OUTPATH = "output/my_logs_0.txt"

while True:
    user_input = input("log: ")
    with open(OUTPATH, mode="a") as f:
        print(user_input, file=f)
    print(user_input)

# command line
$ python apps/jlog/jlog_0.py
log: test 1
test 1
log: test 2
test 2

$ cat output/my_logs_0.txt
test 1
test 2
```



JLog

```
# jlog_1.py
from datetime import datetime

OUTPATH = "output/my_logs_1.txt"

def main():
    while True:
        user_input = input("log: ")
        log = f"{datetime.now()} - {user_input}"
        with open(OUTPATH, mode="a") as f:
            f.write(f"{log}\n")
        print(log)

if __name__ == "__main__":
    main()
```

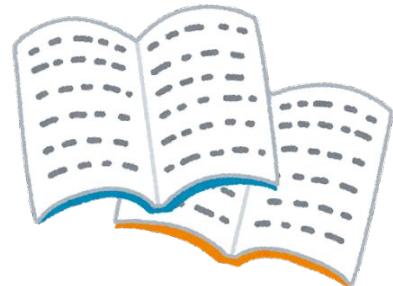


Table of Contents

- [datetime — Basic date and time types](#)
 - Aware and Naive Objects
 - Constants

datetime — Basic date and time types

[Source code: Lib/datetime.py](#)

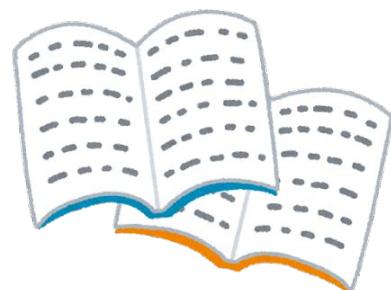
`classmethod datetime.now(tz=None)`

Return the current local date and time.

If optional argument `tz` is `None` or not specified, this is like `today()`, but, if possible, supplies more precision than can be gotten from going through a `time.time()` timestamp (for example, this may be possible on platforms supplying the C `gettimeofday()` function).

If `tz` is not `None`, it must be an instance of a `tzinfo` subclass, and the current date and time are converted to `tz`'s time zone.

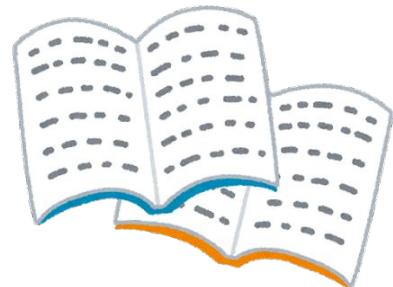
This function is preferred over `today()` and `utcnow()`.



JLog

```
# jlog_1.py
...
def main():
    while True:
        user_input = input("log: ")
        log = f"{datetime.now()} - {user_input}"
        with open(OUTPATH, mode="a") as f:
            f.write(f"{log}\n")
        print(log)
...
# command line
$ python jlog_1.py
log: finish presentation
2022-04-20 18:47:56.725245 - finish presentation
log: play with my cats
2022-04-20 18:48:00.552193 - play with my cats

$ cat output/my_logs_1.txt
2022-04-20 18:47:56.725245 - finish presentation
2022-04-20 18:48:00.552193 - play with my cats
```



Conclusion

- Input loops
- Datetime module
- Writing to a file

```
# jlog_1.py
...
def main():
    while True:
        user_input = input("log: ")
        log = f"{datetime.now()} - {user_input}"
        with open(OUTPATH, mode="a") as f:
            f.write(f"{log}\n")
        print(log)
...
# command line
$ python jlog_1.py
log: finish presentation
2022-04-20 18:47:56.725245 - finish presentation
log: play with my cats
2022-04-20 18:48:00.552193 - play with my cats

$ cat output/my_logs_1.txt
2022-04-20 18:47:56.725245 - finish presentation
2022-04-20 18:48:00.552193 - play with my cats
```

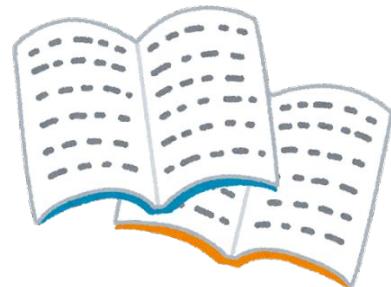


Image Sorting: Demo

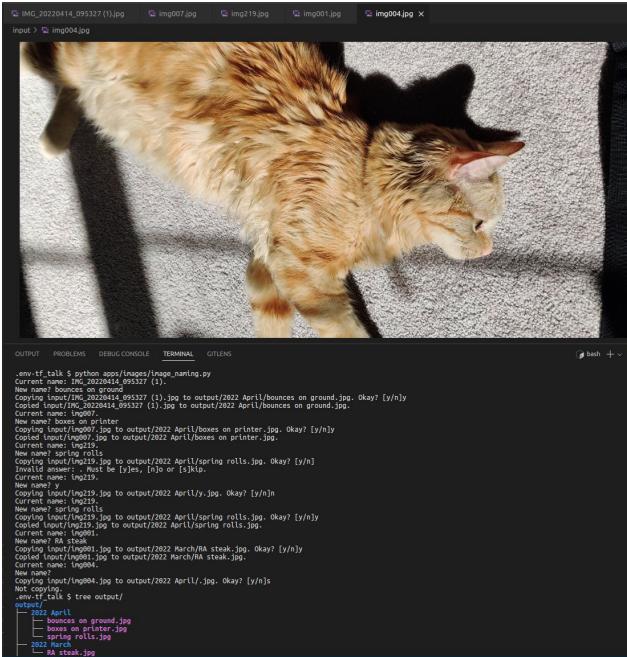


Images (Demo)



```
# jlog_0.py
$ tree input/
input/
├── Cryptocurrencies The Power of Memes | Research Affiliates.html
├── ESG Is a Preference, Not a Strategy | Research Affiliates.html
├── example_log.txt
├── img001.jpg
├── img004.jpg
├── img007.jpg
├── IMG_20220414_095327 (1).jpg
├── img219.jpg
└── presentation.md
    └── ra_cryptocurrencies.txt
```

Images (Demo)



```
.env-tf_talk $ python apps/images/image_naming.py
Current name: IMG_20220414_095327 (1).
New name? bounces on ground
Copying input/IMG_20220414_095327 (1).jpg to output/2022 April/bounces on ground.jpg. Okay? [y/n]y
Copied input/IMG_20220414_095327 (1).jpg to output/2022 April/bounces on ground.jpg.
Current name: img007.
New name? boxes on printer
Copying input/img007.jpg to output/2022 April/boxes on printer.jpg. Okay? [y/n]y
Copied input/img007.jpg to output/2022 April/boxes on printer.jpg.
Current name: img219.
New name? spring rolls
Copying input/img219.jpg to output/2022 April/spring rolls.jpg. Okay? [y/n]
Invalid answer: . Must be [y]es, [n]o or [s]kip.
Current name: img219.
New name? y
Copying input/img219.jpg to output/2022 April/y.jpg. Okay? [y/n]n
Current name: img219.
New name? spring rolls
Copying input/img219.jpg to output/2022 April/spring rolls.jpg. Okay? [y/n]y
Copied input/img219.jpg to output/2022 April/spring rolls.jpg.
Current name: img001.
New name? RA steak
Copying input/img001.jpg to output/2022 March/RA steak.jpg. Okay? [y/n]y
Copied input/img001.jpg to output/2022 March/RA steak.jpg.
Current name: img004.
New name?
Copying input/img004.jpg to output/2022 April/.jpg. Okay? [y/n]s
Not copying.
```

Images (Demo)

The terminal window shows the execution of a Python script for image renaming. The script asks for confirmation before overwriting existing files. It then creates a directory structure under 'output/' for the months April and March, containing files like 'bounces on ground.jpg' and 'RA steak.jpg'. The terminal also displays the current file tree.

```
.env-tf_talk $ python image_rename.py
Current dir: /Users/rafael/Downloads
Copied input/IMG_20220414_095327 (1).jpg to output/2022 April/bounces on ground.jpg. Okay? [y/n]y
Copied input/IMG_20220414_095327 (1).jpg to output/2022 April/bounces on ground.jpg.
Current name: img007.jpg
New name: boxes on printer.
Copyfile input/img007.jpg to output/2022 April/boxes on printer.jpg. Okay? [y/n]y
Copied input/img007.jpg to output/2022 April/boxes on printer.jpg.
Current name: img219.jpg
New name: spring rolls.
Copyfile input/img219.jpg to output/2022 April/spring rolls.jpg. Okay? [y/n]y
Copied input/img219.jpg to output/2022 April/spring rolls.jpg.
Current name: img001.jpg
New name: RA steak.
Copyfile input/img001.jpg to output/2022 March/RA steak.jpg. Okay? [y/n]y
Copied input/img001.jpg to output/2022 March/RA steak.jpg.
Current name: img004.jpg
New name: 
Copyfile input/img004.jpg to output/2022 April/.jpg. Okay? [y/n]y
.env-tf_talk $ tree output/
output/
├── 2022 April
│   ├── bounces on ground.jpg
│   ├── boxes on printer.jpg
│   └── spring rolls.jpg
└── 2022 March
    └── RA steak.jpg
```

```
.env-tf_talk $ tree output/
output/
├── 2022 April
│   ├── bounces on ground.jpg
│   ├── boxes on printer.jpg
│   └── spring rolls.jpg
└── 2022 March
    └── RA steak.jpg
```

Images Goals

- `pathlib.Path`
- `PIL.Image`
- `datetime.strptime`

```
.env-tf_talk $ tree output/
output/
├── 2022 April
│   ├── bounces on ground.jpg
│   ├── boxes on printer.jpg
│   └── spring rolls.jpg
└── 2022 March
    └── RA steak.jpg
```



PIL(low)

<https://pillow.readthedocs.io/en/stable/index.html>



[Pillow \(PIL Fork\)](#)



pillow

stable

Search docs

- Installation
- Handbook
- Reference
- Porting
- About
- Release Notes
- Deprecations and removals

[Edit on GitHub](#)

» Pillow

Pillow

Pillow is the friendly PIL fork by [Alex Clark and Contributors](#). PIL is the Python Imaging Library by [Fredrik Lundh and Contributors](#).

Pillow for enterprise is available via the Tidelift Subscription. [Learn more](#).

docs [passing](#) Lint [passing](#) Test Docker [passing](#) Test [passing](#) Test Windows [passing](#)

Test MinGW [passing](#) Windows build [passing](#) Wheels [passing](#) aarch64 wheels [passing](#) codecov [91%](#)

DOI [10.5281/zenodo.6406647](#) lifted! Tidelift Align [passing](#) pipi v9.1.0 downloads 40M/month

PIL(low)

<https://pillow.readthedocs.io/en/stable/index.html>



The screenshot shows the documentation for the Pillow (PIL Fork) project. At the top, there's a logo featuring a colorful, abstract icon next to the word "pillow". Below the logo, the word "stable" is displayed. A search bar labeled "Search docs" is present. On the left side, a sidebar contains links to "Installation", "Handbook", "Reference", "Porting", "About", "Release Notes", and "Deprecations and removals". The main content area has a dark background with white text, featuring the title "Pillow" in large letters. Below the title, a paragraph describes Pillow as a friendly PIL fork by Alex Clark and Contributors. It also mentions that Pillow for enterprise is available via the Tidelift Subscription and provides a link to learn more. At the bottom of the page, there are several green status indicators for various tests: "docs passing", "Lint passing", "Test Docker passing", "Test passing", "Test Windows passing", "Test MinGW passing", "Windows build passing", "Wheels passing", "aarch64 wheels passing", and "codecov 91%". There are also links for "DOI 10.5281/zenodo.6406647", "Tidelift Align passing", "pypi v9.1.0", and "downloads 40M/month".



Do not install arbitrary packages on your RA platform

The screenshot shows the GitHub repository page for the Pillow project. At the top right, there's a link to "Edit on GitHub". The main title is "Pillow". Below the title, a paragraph describes Pillow as the friendly PIL fork by Alex Clark and Contributors. It also mentions that Pillow for enterprise is available via the Tidelift Subscription and provides a link to learn more. At the bottom of the page, there are several green status indicators for various tests: "docs passing", "Lint passing", "Test Docker passing", "Test passing", "Test Windows passing", "Test MinGW passing", "Windows build passing", "Wheels passing", "aarch64 wheels passing", and "codecov 91%". There are also links for "DOI 10.5281/zenodo.6406647", "Tidelift Align passing", "pypi v9.1.0", and "downloads 40M/month".

PIL(low)

<https://pillow.readthedocs.io/en/stable/index.html>



Do not install arbitrary packages on your RA platform

```
(.env38) $ python -m venv ~/.env-simple_scripting_tf_talk
(.env38) $ . ~/.env-simple_scripting_tf_talk/bin/activate
(.env-simple_scripting_tf_talk) $ pip freeze
(.env-simple_scripting_tf_talk) $ pip install Pillow
Collecting Pillow
  Using cached
    Pillow-9.1.0-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (4.3 MB)
  Installing collected packages: Pillow
    Successfully installed Pillow-9.1.0
  WARNING: You are using pip version 21.1.1; however, version 22.0.4 is available.
  You should consider upgrading via the
    '/home/mccloskey/.env-simple_scripting_tf_talk/bin/python -m pip install
    --upgrade pip' command.
(.env-simple_scripting_tf_talk) $ pip freeze
Pillow==9.1.0
```

PIL(low)

<https://pillow.readthedocs.io/en/stable/handbook/tutorial.html>



Tutorial

Using the Image class

The most important class in the Python Imaging Library is the `Image` class, defined in the module with the same name. You can create instances of this class in several ways; either by loading images from files, processing other images, or creating images from scratch.

To load an image from a file, use the `open()` function in the `Image` module:

```
>>> from PIL import Image  
>>> im = Image.open("hopper.ppm")
```

If successful, this function returns an `Image` object. You can now use instance attributes to examine the file contents:

```
>>> print(im.format, im.size, im.mode)  
PPM (512, 512) RGB
```

The `format` attribute identifies the source of an image. If the image was not read from a file, it is set to None. The size attribute is a 2-tuple containing width and height (in pixels). The `mode` attribute defines the number and names of the bands in the image, and also the pixel type and depth. Common modes are "L" (luminance) for greyscale images, "RGB" for true color images, and "CMYK" for pre-press images.

If the file cannot be opened, an `osError` exception is raised.

Once you have an instance of the `Image` class, you can use the methods defined by this class to process and manipulate the image. For example, let's display the image we just loaded:

```
>>> im.show()
```



PIL(low)

<https://pillow.readthedocs.io/en/stable/handbook/tutorial.html>

Tutorial

Using the Image class

To load an image from a file, use the `open()` function in the `Image` module:

```
>>> from PIL import Image  
>>> im = Image.open("hopper.ppm")
```

Once you have an instance of the `Image` class, you can use the methods defined by this class to process and manipulate the image. For example, let's display the image we just loaded:

```
>>> im.show()
```

Images

```
# images.py
from pathlib import Path
import shutil
from PIL import Image

input_path = Path("input")
output_path = Path("output")
output_path.mkdir(exist_ok=True)

for path in input_path.iterdir():
    if path.suffix != ".jpg":
        continue

    Image.open(path).show()

    new_name = input(f"Current name: {path.stem}.\nNew name? ")
    new_path = output_path / f"{new_name}{path.suffix}"

    shutil.copyfile(path, new_path)
    print(f"Copied {path} to {new_path}.")
```



pathlib

<https://docs.python.org/3/library/pathlib.html>

Table of Contents

- [pathlib — Object-oriented filesystem paths](#)
 - Basic use
 - Pure paths
 - General properties
 - Operators

pathlib — Object-oriented filesystem paths

New in version 3.4.

Source code: [Lib/pathlib.py](#)

Path.**iterdir()**

When the path points to a directory, yield path objects of the directory contents:

```
>>> p = Path('docs')
>>> for child in p.iterdir(): child
...
PosixPath('docs/conf.py')
PosixPath('docs/_templates')
PosixPath('docs/make.bat')
PosixPath('docs/index.rst')
PosixPath('docs/_build')
PosixPath('docs/_static')
PosixPath('docs/Makefile')
```



Images

```
# images.py
from pathlib import Path
import shutil
from PIL import Image

input_path = Path("input")
output_path = Path("output")
output_path.mkdir(exist_ok=True)

for path in input_path.iterdir():
    if path.suffix != ".jpg":
        continue

    Image.open(path).show()

    new_name = input(f"Current name: {path.stem}.\nNew name? ")
    new_path = output_path / f"{new_name}{path.suffix}"

    shutil.copyfile(path, new_path)
    print(f"Copied {path} to {new_path}.")
```



shutil

<https://docs.python.org/3/library/pathlib.html>

Table of Contents

- [shutil — High-level file operations](#)
 - [Directory and files operations](#)
 - [Platform-dependent efficient copy operations](#)
 - [copytree example](#)
 - [rmtree example](#)

shutil — High-level file operations

Source code: [Lib/shutil.py](#)

The `shutil` module offers a number of high-level operations on files and collections of files. In particular, functions are provided which support file copying and removal. For operations on individual files, see also the `os` module.

Images

```
# images.py
from pathlib import Path
import shutil
from PIL import Image

input_path = Path("input")
output_path = Path("output")
output_path.mkdir(exist_ok=True)

for path in input_path.iterdir():
    if path.suffix != ".jpg":
        continue

    Image.open(path).show()

    new_name = input(f"Current name: {path.stem}.\nNew name? ")
    new_path = output_path / f"{new_name}{path.suffix}"

    shutil.copyfile(path, new_path)
    print(f"Copied {path} to {new_path}.")
```



Images



```
# images.py
...
    image.show()
    while True:
        new_name = input(f"Current name: {path.stem}\nNew name? ")
        new_path = output_path / f"{new_name}{path.suffix}"

        msg = f"Copying {path} to {new_path}. "
        if new_path.exists():
            msg += f"WARNING: {new_path} already exists. "
        answer = input(msg + "Okay? [y/n]").lower()
        if answer in {"y", "yes"}:
            shutil.copyfile(path, new_path)
            print(f"Copied {path} to {new_path}. ")
            break
        elif answer in {"n", "no"}:
            pass
        elif answer in {"s", "skip"}:
            print("Not copying.")
            break
        else:
            print(f"Invalid answer: {answer}. Must be [y]es, [n]o or [s]kip.")
```

Images



Pillow (PIL Fork)

stable

Search docs

Installation

<https://pillow.readthedocs.io/en/stable/reference/Image.html>

» Reference » Image Module

Edit on GitHub

Image Module

The `Image` module provides a class with the same name which is used to represent a PIL image. The module also provides a number of factory functions, including functions to load images from files, and to create new images.

`Image.getexif()` [source]





Images

```
>>> Image.open(path).getexif()  
# {296: 2, 282: 72.0, 256: 4000, 257: 1824, 34853: 788, 34665: 240,  
271: 'OnePlus', 272: 'GM1917', 305: 'Picasa', 274: 1, 306: '2022:04:14  
09:53:28', 530: (2, 2), 531: 1, 283: 72.0}
```

Images

```
>>> Image.open(path).getexif()
# {296: 2, 282: 72.0, 256: 4000, 257: 1824, 34853: 788, 34665: 240,
271: 'OnePlus', 272: 'GM1917', 305: 'Picasa', 274: 1, 306: '2022:04:14
09:53:28', 530: (2, 2), 531: 1, 283: 72.0}

# we want (year, month), like 2022 April
def extract_month(image: Image.Image) -> str | None:
    metadata = image.getexif()
    if 306 not in metadata:
        return None
    timestamp_text = metadata[306]
    try:
        timestamp = datetime.strptime(timestamp_text, "%Y:%m:%d
%H:%M:%S")
    except ValueError:
        return None

    return timestamp.strftime("%Y %B")
```



strftime / strptime

<https://docs.python.org/3/library/datetime.html#strftime-strptime-behavior>

strftime() and strptime() Behavior

`date`, `datetime`, and `time` objects all support a `strftime(format)` method, to create a string representing the time under the control of an explicit format string.

Conversely, the `datetime.strptime()` class method creates a `datetime` object from a string representing a date and time and a corresponding format string.

The table below provides a high-level comparison of `strftime()` versus `strptime()`:

	<code>strftime</code>	<code>strptime</code>
Usage	Convert object to a string according to a given format	Parse a string into a <code>datetime</code> object given a corresponding format
Type of method	Instance method	Class method
Method of	<code>date</code> ; <code>datetime</code> ; <code>time</code>	<code>datetime</code>
Signature	<code>strftime(format)</code>	<code>strptime(date_string, format)</code>



strftime() and strptime() Behavior

strftime / strptime

<https://docs.python.org/3/library/datetime.html#strftime-strptime-behavior>

The table below provides a high-level comparison of strftime() versus strptime():

	strftime	strptime
Usage	Convert object to a string according to a given format	Parse a string into a <code>datetime</code> object given a corresponding format
Signature	<code>strftime(format)</code>	<code>strptime(date_string, format)</code>



strftime / strptime

<https://docs.python.org/3/library/datetime.html#strftime-and-strptime-format-codes>



strftime() and strptime() Format Codes

The following is a list of all the format codes that the 1989 C standard requires, and these work on all platforms with a standard C implementation.

Directive	Meaning	Example	Notes
%a	Weekday as locale's abbreviated name.	Sun, Mon, ..., Sat (en_US); So, Mo, ..., Sa (de_DE)	(1)
%A	Weekday as locale's full name.	Sunday, Monday, ..., Saturday (en_US); Sonntag, Montag, ..., Samstag (de_DE)	(1)
%w	Weekday as a decimal number, where 0 is Sunday and 6 is Saturday.	0, 1, ..., 6	
%d	Day of the month as a zero-padded decimal number.	01, 02, ..., 31	(9)
%b	Month as locale's abbreviated name.	Jan, Feb, ..., Dec (en_US); Jan, Feb, ..., Dez (de_DE)	(1)
%B	Month as locale's full name.	January, February, ..., December (en_US); Januar, Februar, ..., Dezember (de_DE)	(1)
%m	Month as a zero-padded decimal number.	01, 02, ..., 12	(9)
%y	Year without century as a zero-padded decimal number.	00, 01, ..., 99	(9)
%Y	Year with century as a decimal number.	0001, 0002, ..., 2013, 2014, ..., 9998, 9999	(2)
%H	Hour (24-hour clock) as a zero-padded decimal number.	00, 01, ..., 23	(9)
%I	Hour (12-hour clock) as a zero-padded decimal number.	01, 02, ..., 12	(9)
%p	Locale's equivalent of either AM or PM.	AM, PM (en_US); am, pm (de_DE)	(1), (3)
%M	Minute as a zero-padded decimal number.	00, 01, ..., 59	(9)
%S	Second as a zero-padded decimal number.	00, 01, ..., 59	(4), (9)
%f	Microsecond as a decimal number, zero-padded to 6 digits.	000000, 000001, ..., 999999	(5)
%z	UTC offset in the form ±HHMM[SS[.ffff]] (empty string if the object is naive)	(empty), +0000, -0400, +1030, +063415, -030712.345216	(6)

strftime() and strptime() Format Codes

The following is a list of all the format codes that the 1989 C standard requires, and these work on all platforms with a standard C implementation.

Directive	Meaning	Example	Notes
%a	Weekday as locale's abbreviated name.	Sun, Mon, ..., Sat (en_US); So, Mo, ..., Sa (de_DE)	(1)
%A	Weekday as locale's full name.	Sunday, Monday, ..., Saturday (en_US); Sonntag, Montag, ..., Samstag (de_DE)	(1)
%w	Weekday as a decimal number, where 0 is Sunday and 6 is Saturday.	0, 1, ..., 6	
%d	Day of the month as a zero-padded decimal number.	01, 02, ..., 31	(9)
%b	Month as locale's abbreviated name.	Jan, Feb, ..., Dec (en_US); Jan, Feb, ..., Dez (de_DE)	(1)
%B	Month as locale's full name.	January, February, ..., December (en_US); Januar, Februar, ..., Dezember (de_DE)	(1)
%m	Month as a zero-padded decimal number.	01, 02, ..., 12	(9)
%y	Year without century as a zero-padded decimal number.	00, 01, ..., 99	(9)
%Y	Year with century as a decimal number.	0001, 0002, ..., 2013, 2014, ..., 9998, 9999	(2)
%H	Hour (24-hour clock) as a zero-padded decimal number.	00, 01, ..., 23	(9)
%I	Hour (12-hour clock) as a zero-padded decimal number.	01, 02, ..., 12	(9)
%p	Locale's equivalent of either AM or PM.	AM, PM (en_US); am, pm (de_DE)	(1), (3)
%M	Minute as a zero-padded decimal number.	00, 01, ..., 59	(9)
%S	Second as a zero-padded decimal number.	00, 01, ..., 59	(4), (9)
%f	Microsecond as a decimal number, zero-padded to 6 digits.	000000, 000001, ..., 999999	(5)
%z	UTC offset in the form ±HHMM[SS[.fffffff]] (empty string if the object is naive).	(empty), +0000, -0400, +1030, +063415, -030712.345216	(6)

```
>>> Image.open(path).getexif()
# {296: 2, 282: 72.0, 256: 4000, 257: 1824, 34853: 788,
34665: 240, 271: 'OnePlus', 272: 'GM1917', 305:
'Picasa', 274: 1, 306: '2022:04:14 09:53:28', 530: (2,
2), 531: 1, 283: 72.0}

# we want (year, month), like 2022 April
def extract_month(image: Image.Image) -> str | None:
    metadata = image.getexif()
    if 306 not in metadata:
        return None
    timestamp_text = metadata[306]
    try:
        timestamp = datetime.strptime(timestamp_text,
"%Y:%m:%d %H:%M:%S")
    except ValueError:
        return None
    return timestamp.strftime("%Y %B")
```

Images

```
# images.py
...
for path in input_path.iterdir():
    if path.suffix != ".jpg":
        continue
    image = Image.open(path)
    subprocess.run(["code", path]) # for WSL, open in vscode server
    # image.show() # for Windows/Linux/Mac

    month = extract_month(image)
    month_path = output_path if month is None else output_path / month
    month_path.mkdir(parents=True, exist_ok=True)
    while True:
        new_name = input(f"Current name: {path.stem}.\nNew name? ")
        new_path = month_path / f"{new_name}{path.suffix}"
    ...

```



Conclusion

- pathlib.Path
- PIL.Image
- datetime.strptime

```
# images.py
...
for path in input_path.iterdir():
    if path.suffix != ".jpg":
        continue
    image = Image.open(path)
    subprocess.run(["code", path]) # for WSL, open in vscode server
    # image.show() # for Windows/Linux/Mac

    month = extract_month(image)
    month_path = output_path if month is None else output_path / month
    month_path.mkdir(parents=True, exist_ok=True)
    while True:
        new_name = input(f"Current name: {path.stem}.\nNew name? ")
        new_path = month_path / f"{new_name}{path.suffix}"
    ...

```



APIs (Jisho)

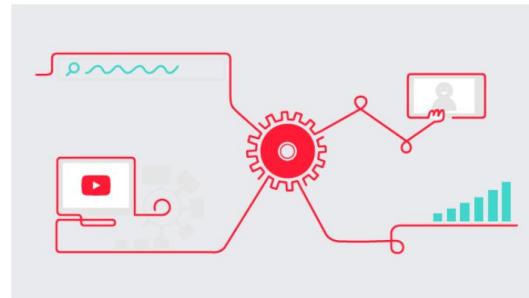
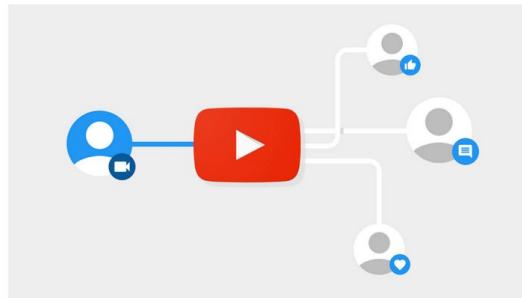


Add YouTube features to your application, including the ability to upload videos, create and manage playlists, and more.

Home Guides Reference Samples Support

Other APIs

Add YouTube functionality to your app



[Add YouTube functionality to your site](#)

With the YouTube Data API, you can add a variety of YouTube features to your application. Use the API to upload videos, manage playlists and subscriptions, update channel settings, and more.

[Get started](#) [Implementation guide](#)

[Search for content](#)

Use the API to search for videos matching specific search terms, topics, locations, publication dates, and much more. The APIs `search.list` method also supports searches for playlists and channels.

[Search for content](#)

Other Resources

Tools

The APIs Explorer lets you test unauthorized and authorized requests. The Quota Calculator shows how different requests impact your quota usage.

Code Samples

Use our code samples to jump-start your project. Samples are available for Apps Script, Go, Java, JavaScript, .NET, PHP, Python, and Ruby.



Walk Score® Professional

[Products](#) | [Why Walk Score?](#) | [Customers](#) | [Real Estate Professionals](#) | [Contact Us](#)

Walk Score APIs Overview



Next Steps

- [Start free API trial](#)
- [Features & pricing](#)
- [Get more information](#)

API Resources

- [APIs Overview](#)
- [Branding Requirements](#)
- [Walk Score API Docs](#)
- [Public Transit API Docs](#)
- [Travel Time API Docs](#)

Walk Score APIs

Software and website developers can use the Walk Score APIs to:

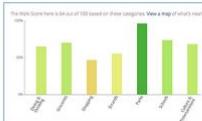
- Display the Walk Score or Transit Score of a location
- Enable search by Walk Score on your site
- Show public transit on a map
- Visualize travel time on a map
- Provide neighborhood insight
- Use for websites, mobile sites and mobile applications

Score API

Get the Walk Score, Transit Score or Bike Score for any location.
[View Documentation.](#)



Score details report:



[Sign up →](#)

Public Transit API

Get nearby public transit route and stop data.
[View Documentation.](#)



Bus lines:

17 Cesar Chavez	0.1 mi
171 Oak Hill Flyer	0.1 mi
127 Dove Springs Flyer	0.1 mi
111 South Mopac Flyer	0.1 mi
935 Tech Ridge Express	0.1 mi

[Sign up →](#)

Travel Time API and Widget

Calculate travel time between locations and show travel time on your map.
[View Documentation.](#)



[Sign up →](#)

Other APIs



POST /2/tweets



Run

Other APIs

Creation of a Tweet

[Go to docs](#)

i Looks like you don't have an access token. Don't worry, clicking **Run** will open the OAuth authorization window so you can get one.

Request body

The JSON payload for the request.

```
{"text": "🐦"}
```

[Run](#)

Jisho Origins: imiwa

https://is2-ssl.mzstatic.com/image/thumb/PurpleSource114/v4/a2/b0/31/a2b03127-f76a-fda6-8cbe-2dc4010b3ae3/ac3b9f29-8667-440e-b438-afa6fe2137cb_iPhone_8_Plus-00-Dictionary_sear ch.png/392x696bb.png



Carrier 4:58 PM

meaning

19 exact matches

意味 「いみ」
meaning, significance, sense / sens, significa...

義 「ぎ」
morality, righteousness, justice, honour (hon...

意義 「いぎ」
meaning, significance / sens, signification / B...

趣, 趣き 「おもむき」
meaning, tenor, gist, effect, influence, appea...

趣旨, 主旨 「しゅし」
meaning, point (e.g. of a statement), gist, eff...

訳 「わけ」
conclusion from reasoning, judgement or cal...

味わい 「あじわい」
flavour, flavor, taste, charm, appeal, interest,...

旨, 宗 「むね」
principle, aim, main purpose, central part, pill...

意味するもの 「いみするもの」

Jisho Origins: imiwa

https://is2-ssl.mzstatic.com/image/thumb/PurpleSource114/v4/a2/b0/31/a2b03127-f76a-fda6-8cbe-2dc4010b3ae3/ac3b9f29-8667-440e-b438-afa6fe2137cb_iPhone_8_Plus-00-Dictionary_sear ch.png/392x696bb.png



Carrier 4:58 PM

meaning

19 exact matches

意味 「いみ」
meaning, significance, sense / sens, significa...

義 「ぎ」
morality, righteousness, justice, honour (hon...

意義 「いぎ」
meaning, significance / sens, signification / B...

趣, 趣き 「おもむき」
meaning, tenor, gist, effect, influence, appea...

趣旨, 主旨 「しゅし」
meaning, point (e.g. of a statement), gist, eff...

訳 「わけ」
conclusion from reasoning, judgement or cal...

味わい 「あじわい」
flavour, flavor, taste, charm, appeal, interest,...

旨, 宗 「むね」
principle, aim, main purpose, central part, pill...

意味するもの 「いみするもの」

Jisho Origins: imiwa

<https://www.davidbcalhoun.com/wp-content/uploads/2013/02/imiwa-ios-list.jpg>

<https://www.davidbcalhoun.com/wp-content/uploads/2013/02/imiwa-ios-list-export.jpg>



.... NTT DOC... 10:31 PM 79%

Lists Edit

- History >
- Overheard 289 >
- Kanji 3 >
- Kanji 1-10 7 >
- L1-10 0 >
- L11-20 0 >
- L21-30 1 >
- L31-40 0 >

Dictionary Kanji Examples Lists Settings

.... NTT DOC... 10:31 PM 79%

Lists Overheard Edit

寮 [りょう]
hostel, dormitory, bureau (government)

勿論 [もちろん]
of course, certainly, naturally

匂い, 臭い, 匂 [におい]
odour, odor, scent, smell, stench, a

様々, 様々 [さまざま]
varied, various

具合, 具合, 工合 [ぐあい]
condition, state, manner, health

入院 [にゅういん]
hospitalization, hospitalisation

晴れる, 曙れる [はれる]
to clear up, to clear away, to be sunny, to sto...

公務員 [こうむいん]

Export
Copy
Merge
Rename

Dictionary Kanji Examples Lists Settings 82

Jisho

<https://jisho.org/>

Jisho is a powerful Japanese-English dictionary. It lets you find words, kanji, example sentences and more quickly and easily.

Enter any Japanese text or English word in the search box and Jisho will search a myriad of data for you.

Here's a few example searches to give you a taste of what Jisho can do.

- Great English search: [house](#)
- Text reading assistance: [昨日すき焼きを食べました](#)
- Inflection information: [走った](#)
- Multi word search: [日 sunlight](#)
- JLPT N3 adjectives: [#jlpt-n3 #adjective](#)
- Grade 1 jōyō kanji: [#grade:1 #kanji](#)
- Common words that end with 家: [#word #common ?*家](#)
- Convert Japanese years: [昭和 5 2](#)
- Convert Japanese numbers: [4 7 7 8 万](#)

There are more examples and explanations on the [search options page](#).



部
Draw
Radicals

All ▾ jisho

Jisho

<https://jisho.org/search/jisho>



Words — 45 found

じしょ

辞書

common word

jpt n5

wanikani level 16

[Play audio](#)[Show 1 collocation](#)[Links](#)

Noun

1. dictionary; lexicon

せいかく じしょ しら
正確なつづりは辞書で調べなさい。

I refer you to the dictionary for the correct spelling.

Noun

2. letter of resignation Archaism, See also [辞表](#)

[Details ▶](#)

じしょ

地所

[Links](#)

Noun

1. estate; plot of land

Other forms

地所 【ちしょ】

[Details ▶](#)

じしょ

字書

[Links](#)

Noun

1. dictionary of Chinese characters; kanji dictionary

Noun

Jisho: Demo



Jisho (Demo)



```
.env-tf_talk $ python sketch/jisho_og.py
Welcome to John's Jisho. Type .h for help.
Loaded Favorites from output/jisho/lists.json.
[Favorites]> .h
John's Jisho.
Current list: Favorites
Current json: output/jisho/lists.json
Type in a keyword or a special argument. Arguments
Q: quit
.H: helpstring
.CL <list>: change list to <list>
.NL <list>: create new list <list>
.SL: show current list
.EL: export list
M: more definitions
letter+number combo: save word-sense pair to list.
[Favorites]> jisho
...
```

Jisho (Demo)



```
[Favorites]> jisho

jisho
Showing entries 1-5/10
A. 辞書 (じしょ)
(1) dictionary, lexicon; (2) letter of resignation

B. 地所 (じしょ); 地所 (ちしょ)
(1) estate, plot of land

C. 字書 (じしょ)
(1) dictionary of Chinese characters, kanji dictionary; (2)
dictionary; (3) Chinese dictionary

D. 自署 (じしょ)
(1) autograph, signature

E. 自書 (じしょ)
(1) one's own writing
Press m to show more

[Favorites]> a1
Saved to Favorites. 辞書:dictionary, lexicon.
[Favorites]> .sl
Favorites:
1. 辞書 じしょ dictionary.
```

Jisho: Goals

- JSON
- requests



```
[Favorites]> jisho

jisho
Showing entries 1-5/10
A. 辞書 (じしょ)
(1) dictionary, lexicon; (2) letter of resignation

B. 地所 (じしょ); 地所 (ちしょ)
(1) estate, plot of land

C. 字書 (じしょ)
(1) dictionary of Chinese characters, kanji dictionary; (2)
dictionary; (3) Chinese dictionary

D. 自署 (じしょ)
(1) autograph, signature

E. 自書 (じしょ)
(1) one's own writing
Press m to show more

[Favorites]> a1
Saved to Favorites. 辞書:dictionary, lexicon.
[Favorites]> .sl
Favorites:
1. 辞書 じしょ dictionary.
```

Jisho: API

<https://jisho.org/api/v1/search/words?keyword=jisho>



Screenshot of a browser window showing the JSON response from the Jisho API search for the word 'jisho'. The URL in the address bar is <https://jisho.org/api/v1/search/words?keyword=jisho>.

```
{"meta": {"status": 200}, "data": [{"slug": "辞書", "is_common": true, "tags": ["wanikani16"], "jlpt": ["jlpt-n5"], "japanese": [{"word": "辞書", "reading": "じしょ"}]}, {"slug": "地所", "is_common": false, "tags": [], "jlpt": [], "japanese": [{"word": "地所", "reading": "ちしょ"}]}, {"slug": "字書", "is_common": false, "tags": [], "jlpt": [], "japanese": [{"word": "字書", "reading": "じしょ"}]}]}
```

Jisho: API

[https://jisho.org/api/v1/search/words?key
word=jisho](https://jisho.org/api/v1/search/words?keyword=jisho)



https://jisho.org/api/v1/search?keyword=jisho

JSON	Raw Data	Headers
<pre>{"meta": {"status": 200}, "data": [{"wanikani16": {"jlpt": [{"jlpt": "書", "reading": "じしょ"}]}, {"senses": [{"dictionary": "lexicon"}, {"parts_of_speech": []}], "restrictions": [], "see_also": [], "english_definitions": [{"text": "letter of resignation"}], "links": [], "tags": [{"tag": "辞表"}], "antonyms": [], "source": [{"source": "Wiktionary"}]}], "is_common": false, "tags": [{"tag": "書"}, {"tag": "辞書"}], "reading": "じしょ"}, {"word": {"english_definitions": [{"text": "letter of resignation"}]}, "parts_of_speech": [{"part": "Noun"}], "restrictions": [], "see_also": [], "antonyms": [], "source": [{"source": "Wiktionary"}]}]</pre>	<pre>Save Copy Collapse All Expand All Filter JSON</pre> <pre>meta: status: 200 data: 0: slug: "辞書" is_common: true tags: 0: 0: "wanikani16" jlpt: 0: 0: "jlpt-n5" japanese: 0: word: "辞書" reading: "じしょ" senses: 0: 0: english_definitions: 0: 0: "dictionary" 1: "lexicon" parts_of_speech: 0: part: "Noun" links: [] tags: [] restrictions: [] see_also: [] antonyms: [] source: [] info: [] 1: english_definitions: 0: 0: "letter of resignation" parts_of_speech: 0: part: "Noun" links: [] </pre>	

Jisho

```
# jisho.py
import requests
import typing as tp

BASE_URL = "https://jisho.org/api/v1/search"

def call_api(keyword: str) -> dict[str, tp.Any]:
    response = requests.get(f"{BASE_URL}/words?keyword={keyword}")
    response.raise_for_status()
    return response.json()

def main() -> None:
    keyword = input("Keyword? ")
    print(call_api(keyword))

if __name__ == "__main__":
    main()

# command line
$ python jisho_0.py
Keyword? jisho
{'meta': {'status': 200}, 'data': [{}{'slug': '辞書', 'is_common': True, 'tags': ['wanikani16'], 'jlpt': ['jlpt-n5'], 'japanese': [{}{'word': '辞書', 'reading': 'じしょ'}]}, 'senses': [{}{'english_definitions': ... 91
```



Jisho



```
# jisho.py
...
def main() -> None:
    keyword = input("Keyword? ")
    print(json.dumps(call_api(keyword), indent=2, ensure_ascii=False))
...

# command line
$ python jisho_1.py
Keyword? jisho
{
    "meta": {
        "status": 200
    },
    "data": [
        {
            "slug": "辞書",
            "is_common": true,
            "tags": [
                "wanikani16"
            ],
        },
    ...
}
```

Jisho



```
JSON Raw Data Headers
Save Copy Collapse All Expand All Filter JSON

▼ meta:
  status: 200
▼ data:
  ▶ 0: ...
  ▶ 1: ...
  ▶ 2: ...
  ▶ 3: ...
  slug: "自署"
  is_common: false
  tags: []
  jlpt: []
  ▼ japanese:
    ▼ 0:
      word: "自署"
      reading: "じしょ"
  ▼ senses:
    ▼ 0:
```

Jisho



```
[Favorites]> jisho

jisho
Showing entries 1-5/10
A. 辞書 (じしょ)
(1) dictionary, lexicon; (2) letter of resignation

B. 地所 (じしょ); 地所 (ちしょ)
(1) estate, plot of land

C. 字書 (じしょ)
(1) dictionary of Chinese characters, kanji dictionary; (2)
dictionary; (3) Chinese dictionary

D. 自署 (じしょ)
(1) autograph, signature

E. 自書 (じしょ)
(1) one's own writing
Press m to show more

[Favorites]> a1
Saved to Favorites. 辞書:dictionary, lexicon.
[Favorites]> .sl
Favorites:
1. 辞書 じしょ dictionary.
```

Jisho



```
# jisho.py

def parse_response(response_json: JsonT):
    data = response_json["data"]
    parsed_response = {}
    for i, entry in enumerate(data, 1):
        parsed_response[num_to_letter(i)] = parse_entry(entry)
    return parsed_response

def num_to_letter(num: int) -> str:
    return chr(num + 64)
```

Jisho



```
JSON Raw Data Headers
Save Copy Collapse All Expand All Filter JSON

▼ meta:
    status: 200
▼ data:
    ▶ 0: {...}
    ▶ 1: {...}
    ▶ 2: {...}
    ▶ 3:
        slug: "自署"
        is_common: false
        tags: []
        jlpt: []
        ▼ japanese:
            ▼ 0:
                word: "自署"
                reading: "じしょ"
        ▼ senses:
            ▼ 0:
                ▼ english_definitions:
                    0: "autograph"
                    1: "signature"
                ▼ parts_of_speech:
```

Jisho



```
[Favorites]> jisho

jisho
Showing entries 1-5/10
A. 辞書 (じしょ)
(1) dictionary, lexicon; (2) letter of resignation

B. 地所 (じしょ); 地所 (ちしょ)
(1) estate, plot of land

C. 字書 (じしょ)
(1) dictionary of Chinese characters, kanji dictionary; (2)
dictionary; (3) Chinese dictionary

D. 自署 (じしょ)
(1) autograph, signature

E. 自書 (じしょ)
(1) one's own writing
Press m to show more

[Favorites]> a1
Saved to Favorites. 辞書:dictionary, lexicon.
[Favorites]> .sl
Favorites:
1. 辞書 じしょ dictionary.
```

Jisho

```
# jisho.py

def parse_response(response_json: JsonT):
    data = response_json["data"]
    parsed_response = {}
    for i, entry in enumerate(data, 1):
        parsed_response[num_to_letter(i)] = parse_entry(entry)
    return parsed_response

def parse_entry(entry: JsonT):
    new_entry = {
        "definitions": get_english_definitions(entry["senses"]),
        "entry": entry["japanese"],
    }
    return new_entry

def get_english_definitions(senses: JsonT):
    eng_definitions = {}
    for i, sense in enumerate(senses, 1):
        eng_definitions[i] = ", ".join(sense["english_definitions"])
    return eng_definitions
```



Jisho



```
# command line
.env-tf_talk $ python jisho_2.py
Keyword? jisho
{
  "A": {
    "definitions": {
      "1": "dictionary, lexicon",
      "2": "letter of resignation"
    },
    "entry": [
      {
        "word": "辞書",
        "reading": "じしょ"
      }
    ]
  },
  "B": {
    "definitions": {
      "1": "estate, plot of land"
    },
    "entry": [
      {
        "word": "地所",
        "reading": "じしょ"
      },
      {
        "word": "地所",
        "reading": "ちしょ"
      }
    ]
  }
}
```

Conclusion

- JSON
- requests

```
# jisho.py
import constants

def get_english_definitions(senses: JsonT) -> str:
    eng_definitions = {}
    for i, sense in enumerate(senses, 1):
        eng_definitions[i] = ", ".join(sense["english_definitions"])
    return eng_definitions

def parse_entry(entry: JsonT) -> JsonT:
    new_entry = {
        "definitions": get_english_definitions(entry["senses"]),
        "entry": entry["japanese"],
    }
    return new_entry

def parse_response(response_json: JsonT) -> JsonT:
    data = response_json["data"]
    parsed_response = {}
    for i, entry in enumerate(data, 1):
        parsed_response[num_to_letter(i)] = parse_entry(entry)
    return parsed_response

def call_api(keyword: str) -> JsonT:
    response = requests.get(f"{BASE_URL}/words?keyword={keyword}")
    response.raise_for_status()
    return response.json()

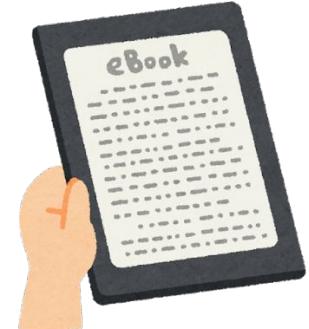
def main() -> None:
    keyword = input("Keyword? ")
    response_json = call_api(keyword)

    print(json.dumps(parse_response(response_json), indent=2, ensure_ascii=False))

if __name__ == "__main__":
    main()
```

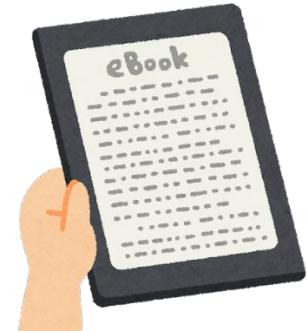


Ebook: Demo



```
# command line  
$ python ebook_4.py  
/home/mccloskey/src/john/techforum_talk/output/RA Publication  
Chapters.epub
```

Ebook (Demo)

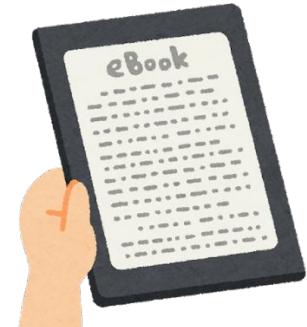


Ebook (Demo)



The screenshot shows a window titled "Unknown [EPUB] - E-book viewer". At the top, there is a navigation bar with a page number "1.0 / 97", a "Go to a reference num..." button, a "Search" button, and several icons for navigating through the document. On the left side, there is a vertical sidebar with icons for navigating between chapters, opening files, and other document-related functions. The main content area displays a table of contents with the following structure:

- 1. RA Publication Chapters
 - 1. [Cryptocurrencies: The Power of Memes](#)
 - 2. [ESG Is a Preference, Not a Strategy](#)



Ebook (Demo)

<https://www.researchaffiliates.com/publications/articles/913-cryptocurrencies-the-power-of-memes>



Unknown [EPUB] - E-book viewer

2.0 / 97 Go to a reference num... Search

Articles

Cryptocurrencies: The Power of Memes

By [Alex Pickard](#)
March 2022
Read Time: 20 min
Key Points

- The idea that cryptocurrencies will disrupt traditional finance is a meme, not a certainty.
- Cryptocurrencies are innovative, but to be disruptive they will need to overcome several difficult hurdles.
- Rather than overcome these hurdles, cryptocurrency has morphed into a permanently speculative investment vehicle, for which disruption lives forever in the future.

Ebook (Demo)

<https://www.researchaffiliates.com/publications/articles/853-esg-is-a-preference-not-a-strategy>



Unknown [EPUB] - E-book viewer

55.0 / 97 Go to a reference num... Search

Articles

ESG Is a Preference, Not a Strategy

By [Brent Leadbetter](#),
[Ari Polychronopoulos](#)
January 2022
Read Time: 20 min
Key Points

- A portfolio's return is driven by its investment strategy —a set of decisions that governs allocation and timing of capital among the portfolio's positions. Modest exclusions designed to align a broadly diversified portfolio with an investor's ESG principles do not change the underlying investment strategy and therefore are not responsible for driving the portfolio's return.
- ESG investing is a preference, not a strategy. We view this trait as a benefit to investors, who can align their portfolios' composition with their beliefs without

```
# command line  
$ python ebook_4.py  
/home/mccloskey/src/john/techforum_talk/output/RA Publication  
Chapters.epub
```

Ebook: Goals

- ebooklib
- beautifulsoup / html



Ebooklib

<https://pypi.org/project/EbookLib/>

```
from ebooklib import epub

book = epub.EpubBook()

# set metadata
book.set_identifier('id123456')
book.set_title('Sample book')
book.set_language('en')

book.add_author('Author Authorowski')
book.add_author('Danko Bananko', file_as='Gospodin Danko Bananko',
role='ill', uid='coauthor')

# create chapter
c1 = epub.EpubHtml(title='Intro', file_name='chap_01.xhtml',
lang='hr')
c1.content=u'

# Intro heading



Zaba je skocila u baru.

'

# add chapter
book.add_item(c1)
```



Ebooklib

<https://pypi.org/project/EbookLib/>

```
# define Table Of Contents
book.toc = (epub.Link('chap_01.xhtml', 'Introduction', 'intro'),
            (epub.Section('Simple book'),
             (c1, )))
)

# add default NCX and Nav file
book.add_item(epub.EpubNcx())
book.add_item(epub.EpubNav())

# define CSS style
style = 'BODY {color: white;}'
nav_css = epub.EpubItem(uid="style_nav", file_name="style/nav.css",
media_type="text/css", content=style)

# add CSS file
book.add_item(nav_css)

# basic spine
book.spine = ['nav', c1]

# write to the file
epub.write_epub('test.epub', book, {})
```



Ebook

```
# ebook.py
from ebooklib import epub
from pathlib import Path

OUTPUT_DIR = Path("output")
INPUT_DIR = Path("input")

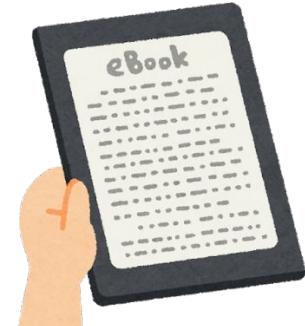
def main(title: str, input_path: Path) -> Path:
    # make chapter
    chapter = epub.EpubHtml(
        title=title, content=input_path.read_text(), file_name="file1.xhtml"
    )

    # make book
    book = epub.EpubBook()
    book.add_item(chapter)
    book.spine = [chapter]

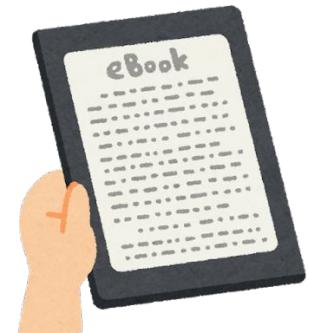
    # write book
    OUTPUT_DIR.mkdir(exist_ok=True)
    output_path = OUTPUT_DIR / f"{title}.epub"
    epub.write_epub(output_path, book)

    return output_path

if __name__ == "__main__":
    input_path = (
        INPUT_DIR / "Cryptocurrencies The Power of Memes | Research Affiliates.html"
    )
    print(main(title="RA Crypto Currency Article HTML", input_path=input_path))
```



Ebook



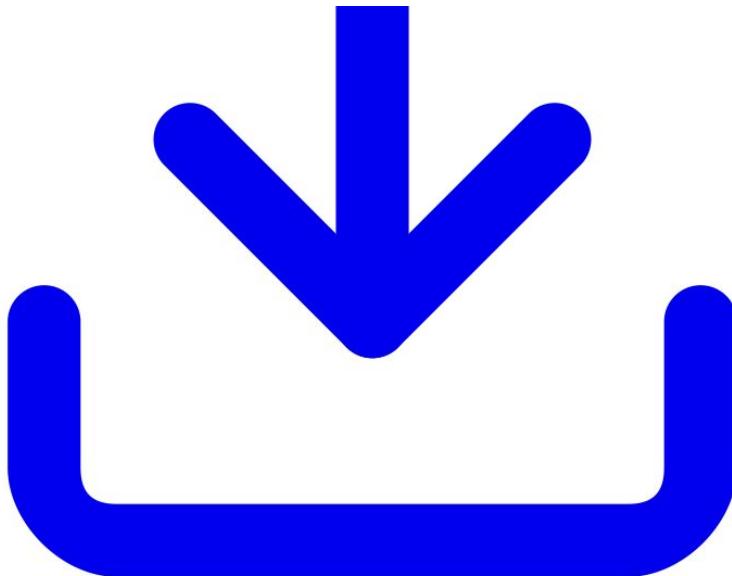
Unknown [EPUB] - E-book viewer

1.0 / 68 Go to a reference number... Search

• [Insights](#)

A screenshot of an e-book viewer application window. The title bar says "Unknown [EPUB] - E-book viewer". The top menu bar includes "1.0 / 68", "Go to a reference number...", "Search", and navigation icons. A sidebar on the left contains icons for file operations, zoom, orientation, and navigation. The main content area features a large, solid black rectangular redaction box covering most of the page. Above the redaction, there is a blue square icon with a question mark and the text "• [Insights](#)".

Ebook



Download

Key Points

- The idea that cryptocurrencies will disrupt traditional finance is a meme, not a certainty.
- Cryptocurrencies are innovative, but to be disruptive they will need to overcome several difficult hurdles.
- Rather than overcome these hurdles, cryptocurrency has morphed into a permanently speculative investment vehicle, for which disruption lives forever in the future.

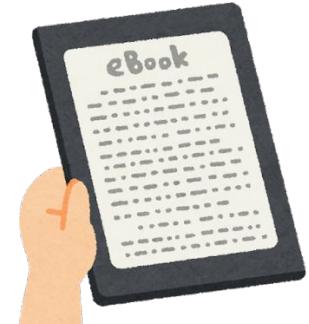
“It is a peculiarity of man that he can only live by looking to the

Ebook

The screenshot shows a website header for 'research affiliates' with navigation links for Insights, Tools, Strategies, Advisors, How to Invest, and About Us. It features a search bar and a greeting 'Hi, John'. The main content area has a dark background with a grid pattern and a banner image showing various financial charts and graphs. The word 'ARTICLES' is visible in the top right of the content area. A large, bold title 'Cryptocurrencies: The Power of Memes' is centered. Below it, the author is listed as 'By Alex Pickard' with the date 'March 2022' and a 'Read Time: 20 min'. At the bottom, there are three buttons: 'Share', 'Save', and 'Download'.

Key Points

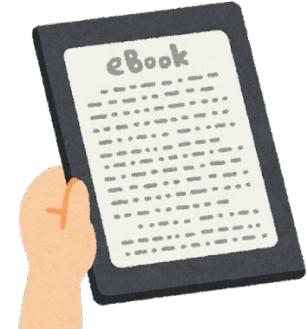
- The idea that cryptocurrencies will disrupt traditional finance is a meme, not a certainty



Ebook

```
# ebook.py

def main(title: str, input_path: Path) -> Path:
    # make chapter
    chapter = epub.EpubHtml(
        title=title, content=input_path.read_text(),
        file_name="file1.xhtml"
    )
```



Beautiful Soup

Quick Start

Here's an HTML document I'll be using as an example throughout this document. It's part of a story from *Alice in Wonderland*:

```
html_doc = """<html><head><title>The Dormouse's story</title></head>
<body>
<p class="title"><b>The Dormouse's story</b></p>

<p class="story">Once upon a time there were three little sisters; and their names were
<a href="http://example.com/elsie" class="sister" id="link1">Elsie</a>,
<a href="http://example.com/lacie" class="sister" id="link2">Lacie</a> and
<a href="http://example.com/tillie" class="sister" id="link3">Tillie</a>;
and they lived at the bottom of a well.</p>

<p class="story">...</p>
""""
```



```
<html><head><title>The Dormouse's story</title></head>
<body>
<p class="title"><b>The Dormouse's story</b></p>

<p class="story">Once upon a time there were three little sisters; and their names were
<a href="http://example.com/elsie" class="sister" id="link1">Elsie</a>,
<a href="http://example.com/lacie" class="sister" id="link2">Lacie</a> and
<a href="http://example.com/tillie" class="sister" id="link3">Tillie</a>;
and they lived at the bottom of a well.</p>

<p class="story">...</p>
```

```
<html><head><title>The Dormouse's  
story</title></head>  
<body>  
<p class="title"><b>The Dormouse's  
story</b></p>  
  
<p class="story">Once upon a time  
there were three little sisters; and  
their names were  
<a href="http://example.com/elsie"  
class="sister" id="link1">Elsie</a>,  
<a href="http://example.com/lacie"  
class="sister" id="link2">Lacie</a>  
and  
<a href="http://example.com/tillie"  
class="sister"  
id="link3">Tillie</a>;  
and they lived at the bottom of a  
well.</p>  
  
<p class="story">...</p>
```

Here are some simple ways to navigate that data structure:

```
soup.title  
# <title>The Dormouse's story</title>  
  
soup.title.name  
# u'title'  
  
soup.title.string  
# u'The Dormouse's story'  
  
soup.title.parent.name  
# u'head'  
  
soup.p  
# <p class="title"><b>The Dormouse's story</b></p>  
  
soup.p['class']  
# u'title'  
  
soup.a  
# <a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>  
  
soup.find_all('a')  
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>,  
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</a>,  
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie</a>]  
  
soup.find(id="link3")  
# <a class="sister" href="http://example.com/tillie" id="link3">Tillie</a>
```

One common task is extracting all the URLs found within a page's `<a>` tags:

```
for link in soup.find_all('a'):   
    print(link.get('href'))  
# http://example.com/elsie  
# http://example.com/lacie  
# http://example.com/tillie
```



Cryptocurrencies: The Power of Memes

By Alex Pickard

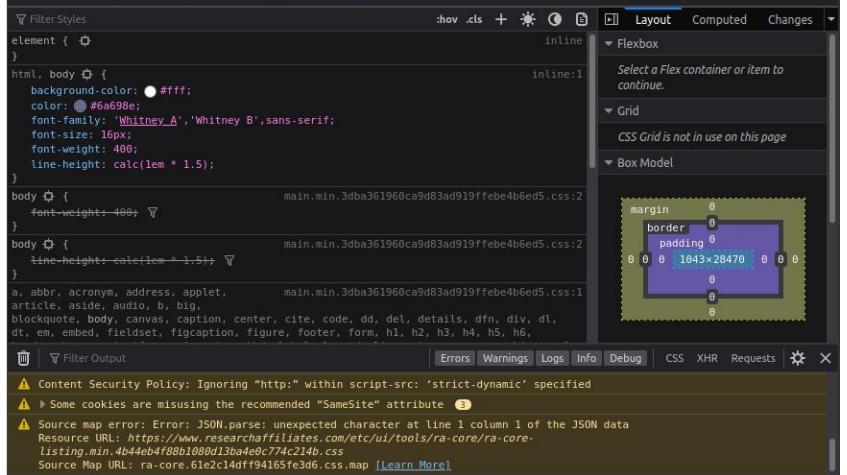
March 2022 Read Time: 20 min

 Share Save Download

Key Points

- The idea that cryptocurrencies will disrupt traditional finance is a meme, not a certainty.
 - Cryptocurrencies are innovative, but to be disruptive they will need to overcome several difficult hurdles.
 - Rather than overcome these hurdles, cryptocurrency has morphed into a permanently speculative investment vehicle, for which

```
</html>
<html class=" whatinput-types-initial whatinput-types-keyboard " xmlns:og="http://ogp.me/ns#" data-whatinput="keyboard" data-whatinten="mouse" lang="en-US">[event scroll]
  <head></head>
  <body class="viewport-desktop logged-in not-client-access" data-theme="ra"> event overflow
    <noscript></noscript>
    <div class="ra-theme-ra ra-grid-default ra-grid-width-l240 ra-grid-align-center ra-grid-cols-6 ra-grid-gutter-32 ra-grid-device-gutter-8">
      <div class="ra-page-head"></div>
      <div class="ra-page-banner"></div>
      <div class="ra-page-main">
        <div class="ra-component-publication">
          <div class="padding-bottom-08" data-t-name="Publication" data-t-id="2">
            <div class="publication-hero padding-top-14 device-padding-top-08 device-padding-left-02 device-padding-right-02" style="background-image: url('/content/dam/ra/publications/hero/913_ackground-repeat: no-repeat; background-position: center-top')>
              <div class="hero_image-print"></div>
            <div class="ra-grid-row ra-grid-inherit" flex>
              <div class="ra-grid-margin"></div>
              <div class="ra-grid-columns" flex>
                <div class="ra-grid-col-1 hero_box bg-color-white border-radius-top-left-04 border-radius-top-right-04">
                  <div class="ra-grid-col-1 hero_box_bg color-white border-radius-top-left-04 border-radius-top-right-04">
                    <div class="hero_heading font-chronicle-g1-italic font-color-prism-gray _device-padding-top-03 device-font-size-14 device-line-size-14"></div>
                    <div class="hero_title font-chronicle-display-semibold font-color-prism-er device-padding-03 device-font-size-32 device-line-size-40">
                      <h1>Cryptocurrencies: The Power of Memes</h1>
                    </div>
                    <div class="hero_byline font-color-prism-black line-size-33 font-whitney-book device-line-size-24 font-size-24 device-font-size-18"></div>
                    <div class="hero_credits" data-readthrough="font-color-nim-black font-size-19 font-w-ew-book device-font-size-16 device-line-size-24">
                      <small>by whatinputtypesinitial.whatinput...</small>
                    </div>
                  </div>
                </div>
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>
  </body>
```



Ebook

```
# ebook.py

def main(input_path: Path, output_suffix: str = "") -> Path:
    # make chapter
    chapter = html_to_chapter(input_path)
    ...

def html_to_chapter(html_path: Path) -> epub.EpubHtml:
    soup = bs4.BeautifulSoup(html_path.read_text(),
features="html.parser")
    [title_box] = soup.findAll(**{"class": "hero_box"})
    [title_element] = title_box.findAll(**{"class": "hero_title"})
    title = title_element.text
    [publication_keypoints] = soup.findAll(**{"class":
"publication_keypoints"})
    [publication_body] = soup.findAll(**{"class": "publication_body"})
    fname = f"{uuid.uuid4()}.xhtml"
    content = f"{title_box}{publication_keypoints}{publication_body}"
    return epub.EpubHtml(title=title, content=content, file_name=fname)
```



Key Points

- The idea that cryptocurrencies will disrupt traditional finance is a meme, not a certainty.
- Cryptocurrencies are innovative, but to be disruptive they will need to overcome several difficult hurdles.
- Rather than overcome these hurdles, cryptocurrency has morphed into a permanently speculative investment vehicle, for which disruption lives forever in the future.¹⁸

"It is a peculiarity of man that he can only live by looking to the future – sub specie aeternitatis."

— Viktor E. Frankl, *Man's Search for Meaning*

The screenshot shows the browser's developer tools with the element inspector open. The selected element is a `<div>` with the class `"publication__keypoints margin-top-05 device-margin-top-03 device-padding-left-03 device-padding-right-03"`. This element is part of a larger grid-based layout. The DOM tree shows nested `<div>` elements with classes like `ra-grid-row`, `ra-grid-inherit`, `ra-grid-cols-stack`, `ra-grid-margin`, `ra-grid-columns`, `ra-grid-col-1*`, and `ra-grid-col-4*`. A `` element contains four `` items. The `margin-top-05` style is highlighted with a red box in the element's bounding box. The right-hand panel of the developer tools displays the computed styles for the selected element, including `margin-top: 0.5em` and `calc(1rem * 2.5)`.

```
</div>
<div class="publication__keypoints margin-top-05 device-margin-top-03 device-padding-left-03 device-padding-right-03">
  <div class="ra-grid-row ra-grid-inherit ra-grid-cols-stack">flex
    <div class="ra-grid-margin"></div>
    <div class="ra-grid-columns">flex overflow
      <div class="ra-grid-col-1*"></div>
      <div class="ra-grid-col-4*">
        <div class="font-chronicle-display-semibold font-color-prim-navy font-size-16 device-line-size-32 padding-bottom-02">Key Points</div>
        <div class="font-color-prim-black font-size-23 font-whitney-book line-size-16 device-line-size-26">
          <ul>
            <li></li>
            <li></li>
            <li></li>
            <li></li>
          </ul>
        </div>
      </div>
      <div class="ra-grid-col-1"></div>
    </div>
    <div class="ra-grid-margin"></div>
  </div>
</div>
```

Filter Styles

element { }

.margin-top-05, .margin-top-hover-05:hover { margin-top: calc(1rem * 2.5); }

a, abbr, acronym, address, applet, article, aside, audio, b, big, blockquote, body, canvas, caption, center, cite, code, dd, del, details, dfn, div, dl, dt, em, embed, fieldset, figcaption, figure, footer, form, h1, h2, h3, h4, h5, h6, header, hgroup, i, iframe, img, ins, kbd, label, legend, li, mark, menu, nav, object, ol, output, p, pre, q, ruby, s, samp, section, small, span, strike, strong, sub, summary, sup, table, tbody, td, tfoot, th, thead, time, tr, tt, u, ul, var, video { margin: 0; margin-top: 0px; padding: 0; border: 0; font-size: 100%; }

Layout Computed Changes

Select a Flex container or item to continue.

Grid

CSS Grid is not in use on this page

Box Model

margin	40
border	0
padding	0
0 0 0	1043x411
0	0

Ebook

<https://www.researchaffiliates.com/publications/articles/913-cryptocurrencies-the-power-of-memes>



Unknown [EPUB] - E-book viewer

2.0 / 97 Go to a reference num... Search

Articles

Cryptocurrencies: The Power of Memes

By [Alex Pickard](#)
March 2022
Read Time: 20 min
Key Points

- The idea that cryptocurrencies will disrupt traditional finance is a meme, not a certainty.
- Cryptocurrencies are innovative, but to be disruptive they will need to overcome several difficult hurdles.
- Rather than overcome these hurdles, cryptocurrency has morphed into a permanently speculative investment vehicle, for which disruption lives forever in the future.

Ebook

```
# ebook.py

def main(input_paths: tp.Iterable[Path], title: str) -> Path:
    # make chapter
    chapters = [html_to_chapter(path) for path in input_paths]

    # make book
    book = epub.EpubBook()
    book.spine = ["nav"] + chapters
    for c in chapters:
        book.add_item(c)
    book.toc = ((epub.Section(title), chapters),)

    # add default Nav file
    book.add_item(epub.EpubNav())

    # write book
    OUTPUT_DIR.mkdir(exist_ok=True)
    output_path = OUTPUT_DIR / f"{title}.epub"
    epub.write_epub(output_path, book)

    return output_path
```



Conclusion

- ebooklib
- beautifulsoup / html



Python is an accessible scripting language!



Python is an accessible scripting language!

1. Simple Adding Script: input, sys.argv



Python is an accessible scripting language!

1. Simple Adding Script: input, sys.argv
2. Time Logging: loops, datetime, writing to a file



Python is an accessible scripting language!

1. Simple Adding Script: input, sys.argv
2. Time Logging: loops, datetime, writing to a file
3. Bulk Image Organizer: Path, PIL, strftime



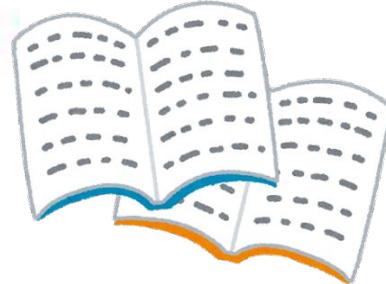
Python is an accessible scripting language!

1. Simple Adding Script: input, sys.argv
2. Time Logging: loops, datetime, writing to a file
3. Bulk Image Organizer: Path, PIL, strftime
4. Jisho API: APIs, requests, JSON



Python is an accessible scripting language!

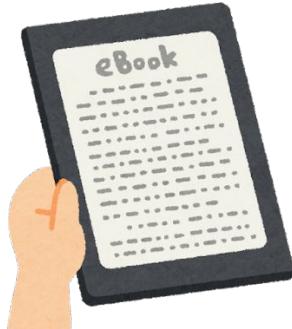
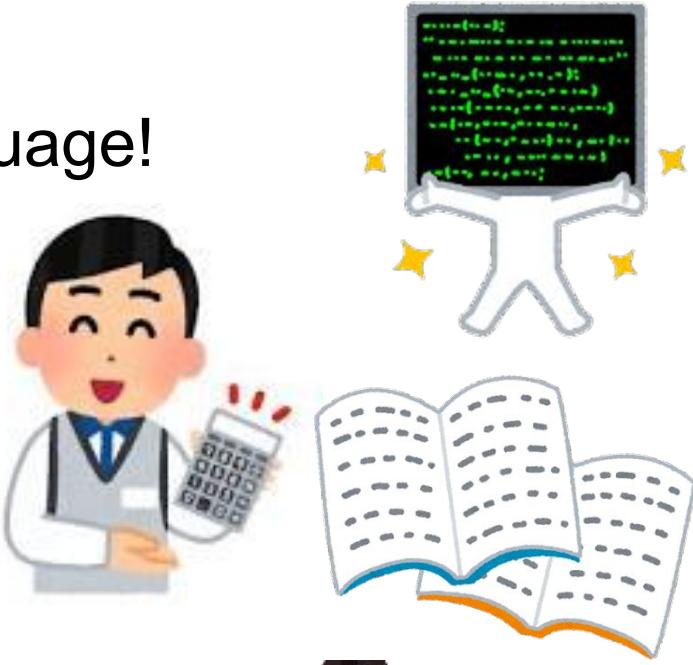
1. Simple Adding Script: input, sys.argv
2. Time Logging: loops, datetime, writing to a file
3. Bulk Image Organizer: Path, PIL, strftime
4. Jisho API: APIs, requests, JSON
5. EBook Writer: ebooklib, beautifulsoup, HTML



Python is an accessible scripting language!

1. Simple Adding Script: input, sys.argv
2. Time Logging: loops, datetime, writing to a file
3. Bulk Image Organizer: Path, PIL, strftime
4. Jisho API: APIs, requests, JSON
5. EBook Writer: ebooklib, beautifulsoup, HTML

And so much more!



Credits:

- Cute pictures from <https://www.irasutoya.com/>



Other topics

- When to not write a script?
- When to transition from small program to large program?