

PEP 498: THE MONOLOGUE

MARIATTA WIJAYA



imageworks



PYCON CANADA 2017

@mariatta

#PEPtalk

 You cool people!

Who are you?

Are you ...

... new to Python?

Are you ...

... only aware of one PEP?

Are you ...

... using Python < 3.6?

Who am I?



Who am I?



Who am I?

@mariatta





Who am I?

@mariatta





Who am I?

@mariatta



Software Engineer



imageworks



Who am I?

@mariatta



Software Engineer



imageworks

Python Core Developer



What's a PEP?

Python **E**nhancement **P**roposal

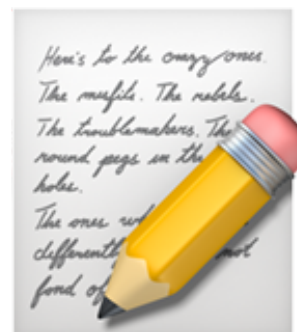
PEP 1

PEP Purpose and Guideline

<https://www.python.org/dev/peps/pep-0001>



pitch to python-ideas



draft a PEP

official template: PEP 12



BDFL Pronouncement



BDFL Pronouncement





BDFL Pronouncement



450 PEPs today
88 rejected

PEP 498

2015

From: Mike Miller
Subject: [python-ideas] Briefer string format

Jul 20 01:12:31 CEST

<https://mail.python.org/pipermail/python-ideas/2015-July/034657.html>

PYCON CANADA 2017

@mariatta

#PEPtalk

2015

Jul 20 01:12:31 CEST

From: Mike Miller

Subject: [python-ideas] Briefer string format

Have long wished python could format strings easily like
bash or perl do ... and then it hit me

<https://mail.python.org/pipermail/python-ideas/2015-July/034657.html>

PYCON CANADA 2017

@mariatta

#PEPtalk

2015

Jul 20 01:12:31 CEST

From: Mike Miller

Subject: [python-ideas] Briefer string format

Have long wished python could format strings easily like
bash or perl do ... and then it hit me

```
csstext += f'{nl}{selector}{space}{{{nl}}}'
```

<https://mail.python.org/pipermail/python-ideas/2015-July/034657.html>

2015

Jul 20 01:12:31 CEST

From: Mike Miller

Subject: [python-ideas] Briefer string format

Have long wished python could format strings easily like bash or perl do ... and then it hit me

```
csstext += f'{nl}{selector}{space}{{{nl}}}'
```

I've seen others make similar suggestions, but to my knowledge they didn't include this **pleasing brevity** aspect.

<https://mail.python.org/pipermail/python-ideas/2015-July/034657.html>

2015

Jul 20 01:12:31 CEST

Jul 20 01:27:42 CEST

From: Eric V. Smith
Subject: Re: [python-ideas] Briefer string format

<https://mail.python.org/pipermail/python-ideas/2015-July/034658.html>

PYCON CANADA 2017

@mariatta

#PEPtalk

2015

Jul 20 01:12:31 CEST

Jul 20 01:27:42 CEST

From: Eric V. Smith

Subject: Re: [python-ideas] Briefer string format

What would this do?

<https://mail.python.org/pipermail/python-ideas/2015-July/034658.html>

PYCON CANADA 2017

@mariatta

#PEPtalk

2015

From: C. A.
Subject: Re: [python-ideas] Briefer string format

Jul 20 01:12:31 CEST

Jul 20 01:27:42 CEST

Jul 20 01:44:09 CEST

<https://mail.python.org/pipermail/python-ideas/2015-July/034660.html>

PYCON CANADA 2017

@mariatta

#PEPtalk

2015

Jul 20 01:12:31 CEST

Jul 20 01:27:42 CEST

Jul 20 01:44:09 CEST

From: C. A.

Subject: Re: [python-ideas] Briefer string format

I'm **-1** on the specific idea, though definitely sympathetic to the broader concept of simplified formatting of strings.

<https://mail.python.org/pipermail/python-ideas/2015-July/034660.html>

PYCON CANADA 2017

@mariatta

#PEPtalk

2015

From: S. D.
Subject: Re: [python-ideas] Briefer string format

Jul 20 01:12:31 CEST

Jul 20 01:27:42 CEST

Jul 20 01:44:09 CEST

Jul 20 02:43:29 CEST

<https://mail.python.org/pipermail/python-ideas/2015-July/034669.html>

PYCON CANADA 2017

@mariatta

#PEPtalk

2015

Jul 20 01:12:31 CEST

Jul 20 01:27:42 CEST

Jul 20 01:44:09 CEST

Jul 20 02:43:29 CEST

From: S. D.

Subject: Re: [python-ideas] Briefer string format

It's **syntactic sugar** for a simple function call with perfectly well defined semantics - don't even have to modify the string literal.

I'm **+1**.

<https://mail.python.org/pipermail/python-ideas/2015-July/034669.html>

that escalated quickly

2015

From: Guido van Rossum
Subject: Re: [python-ideas] Briefer string format

Jul 20 01:12:31 CEST

Jul 20 01:27:42 CEST

Jul 20 01:44:09 CEST

Jul 20 02:43:29 CEST

Jul 21 08:05:15 CEST

<https://mail.python.org/pipermail/python-ideas/2015-July/034729.html>

PYCON CANADA 2017

@mariatta

#PEPtalk

2015

Jul 20 01:12:31 CEST

Jul 20 01:27:42 CEST

Jul 20 01:44:09 CEST

Jul 20 02:43:29 CEST

Jul 21 08:05:15 CEST

From: Guido van Rossum

Subject: Re: [python-ideas] Briefer string format

Thanks, Eric! You're addressing all my concerns and you're going exactly where I wanted this to go. **I hope that you will find the time to write up a PEP;**

<https://mail.python.org/pipermail/python-ideas/2015-July/034729.html>

August 7th, 2015

PEP 498: Literal String Formatting

By: Eric V. Smith

August 7th, 2015

August 30th, 2015

PEP 498: Literal String Interpolation

By: Eric V. Smith

August 7th, 2015

August 30th, 2015

September 5th, 2015

PEP 498: Literal String Interpolation

ready for pronouncement

August 7th, 2015

August 30th, 2015

September 5th, 2015

September 7th, 2015

PEP 498: Literal String Interpolation

ACCEPTED

RATIONALE

“The existing ways of formatting are either **error prone**, **inflexible**, or **cumbersome**.”

```
>>> name = "Bart"  
>>> print("Hello, %s." % name)  
Hello, Bart.
```



```
>>> name = "Bart"
>>> age = 10
>>> print("Hello, %s. You're %s." % name, age)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: not enough arguments for format string
```

```
>>> name = "Bart"
>>> age = 10
>>> print("Hello, %s. You're %s." % (name, age))
Hello, Bart. You're 10.
```

PEP 3101: str.format

<https://www.python.org/dev/peps/pep-3101>


```
>>> name = "Bart"
>>> age = 10
>>> print("Hello, {name}. You're {age}.".format(name=name, age=age))
Hello, Bart. You're 10.
```

PEP 498: f-string

<https://www.python.org/dev/peps/pep-0498>

```
>>> name = "Bart"
>>> age = 10
>>> print("Hello, {name}. You're {age}.".format(name=name, age=age))
```



```
>>> name = "Bart"  
>>> age = 10  
>>> print("Hello, {name}. You're {age}.")
```

```
>>> name = "Bart"  
>>> age = 10  
>>> print(f"Hello, {name}. You're {age}.")
```



```
>>> name = "Bart"
>>> age = 10
>>> print(f"Hello, {name}. You're {age}.")
Hello, Bart. You're 10.
```

recap

```
"Hello, %s. You're %s." % (name, age)
```



```
"Hello, %s. You're %s." % (name, age)
```

```
"Hello, {name}. You're {age}.".format(name=name, age=age)
```

```
"Hello, %s. You're %s." % (name, age)
```

```
"Hello, {name}. You're {age}.".format(name=name, age=age)
```

```
f"Hello, {name}. You're {age}."
```

Expression



```
f"Hello, {name}. You're {age}."
```


Literal



f"Hello, {name}. You're {age}."

f-string

f"Hello, world."



`f"..."`



`F'...'`



`f"|||||..."`

Raw f-strings



`r + f = fr"..."`

```
>>> print("The smiley face emoji is \U0001f600")
```



```
>>> print("The smiley face emoji is \U0001f600")
```

The smiley face emoji is 😊

```
>>> print("The smiley face emoji is \U0001f600")
```

The smiley face emoji is 😊

```
>>> print(r"The smiley face unicode is \U0001f600")
```

```
>>> print("The smiley face emoji is \U0001f600")
```

The smiley face emoji is 😊

```
>>> print(r"The smiley face unicode is \U0001f600")
```

The smiley face unicode is \U0001f600


```
>>> code = "emoji"  
>>> print(f"The smiley face {code} is \U0001f600")
```

```
>>> code = "emoji"  
>>> print(f"The smiley face {code} is \U0001f600")
```

The smiley face emoji is 😊


```
>>> code = "emoji"
>>> print(f"The smiley face {code} is \U0001f600")
```

The smiley face emoji is 😊

```
>>> code = "unicode"
>>> print(fr"The smiley face {code} is \U0001f600")
```

```
>>> code = "emoji"
>>> print(f"The smiley face {code} is \U0001f600")
```

The smiley face emoji is 😊

```
>>> code = "unicode"
>>> print(fr"The smiley face {code} is \U0001f600")
```

The smiley face unicode is \U0001f600



`fr"..."`



`RF"..."`



`rF""""...""""`



`fb"..."`



`uf"..."`

```
>>> name = "bart simpson"  
>>> print(f"Hi {name.upper()}!")  
Hi BART SIMPSON!
```

```
>>> def how_many_tacos(people):  
...     return people * 3  
...  
>>> people = 5  
>>> print(f"Order {how_many_tacos(people)} tacos for meetup!")  
Order 15 tacos for meetup!
```



```
>>> pi = 3.14159265
```

```
>>> print("pi 3 decimal places %.3f" % pi)  
3.142
```

```
>>> print(f"pi 3 decimal places {pi:.3f}")  
3.142
```

```
>>> number = 1024
```

```
>>> print(f"hex: {number:#0x}")
```

```
hex: 0x400
```

```
>>> print(f"binary: {number:#0b}")
```

```
binary: 0b100000000000
```

```
>>> print(f"octal: {number:#0o}")
```

```
octal: 0o2000
```



```
>>> pycon_ca = datetime(year=2017, month=11, day=19)
```

```
>>> print(f"{pycon_ca:%b %d, %Y}")
```

```
Nov 19, 2017
```

```
>>> print(f"{name:>20}")
```

```
Bart
```

```
>>> print(f"{age:=+5d}")
```

```
+ 10
```



Not in Docstrings

[bpo-28739](#)

```
>>> def spam():  
...     f"doing stuff"  
...  
>>> spam.__doc__ is None  
True
```



Multi-line Strings?

bpo-29668

```
>>> name = "Bart"
>>> prize = 50
>>> tomorrow = today() + timedelta(days=1)
>>> message = (f"Dear {name},"
...             "You can win {prize:.2f}$"
...             "Make a purchase before {tomorrow:%Y-%b-%d}")
```




Multi-line Strings

bpo-29668

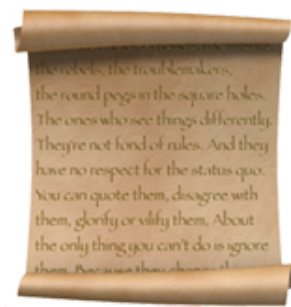
```
>>> message = (f"Dear {name},"  
...           f"You can win {prize:.2f}$"  
...           f"Make a purchase before {tomorrow}:%Y-%b-%d")
```



IDLE Needs Syntax Highlighting

[bpo-29287](#)

- Needs separate colorization to make the expression distinct from the rest of the string.
- Needs close-brace matching.
- Would be desirable to have autocompletion as well.



Documentation ... ?

Search Results

Python 3.6.2

Your search did not match any documents. Please make sure that all words are spelled correctly and that you've selected enough categories.



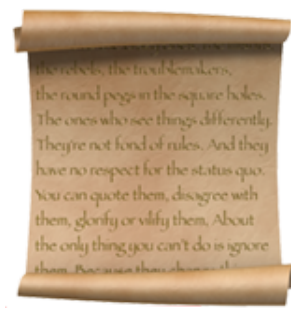
Python »

3.6.2rc1



Documentation »

[modules](#) | [index](#)



Documentation ... ?



f-string

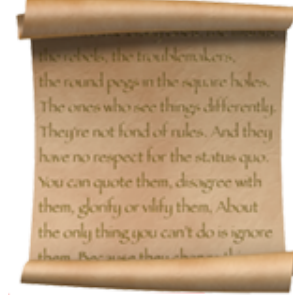
search

Python 3.6.3

Search Results

Search finished, found 170 page(s) matching the search query.

-  **array.array.fromstring** (Python method, in 8.7. **array** — Efficient arrays of numeric values)
-  **array.array.fromstring** (Python method, in 8.7. **array** — Efficient arrays of numeric values)
-  **array.array.tostring** (Python method, in 8.7. **array** — Efficient arrays of numeric values)
-  **ast.get_docstring** (Python function, in 32.2. **ast** — Abstract Syntax Trees)
-  **base64.decodestring** (Python function, in 19.6. **base64** — Base16, Base32, Base64, Base85 Data Encodings)



Documentation



Formatted String Literal



[docs.python.org](https://docs.python.org/3/reference/lexical_analysis.html#formatted-string-literals) > glossary > f-string



https://docs.python.org/3/reference/lexical_analysis.html#formatted-string-literals



timeit

```
$ python3 -mtimeit -s 'a=2' "'%s' % a"  
10000000 loops, best of 3: 0.197 usec per loop
```

```
$ python3 -mtimeit -s 'a=2' "'%s' % a"  
10000000 loops, best of 3: 0.197 usec per loop
```

```
$ python3 -mtimeit -s 'a=2' "{}".format(a)  
10000000 loops, best of 3: 0.341 usec per loop
```



```
$ python3 -mtimeit -s 'a=2' "'%s' % a"
10000000 loops, best of 3: 0.197 usec per loop
```

```
$ python3 -mtimeit -s 'a=2' '{}'.format(a)
10000000 loops, best of 3: 0.341 usec per loop
```

```
$ python3 -mtimeit -s 'a=2' 'f"{a}"'
10000000 loops, best of 3: 0.105 usec per loop
```

```
$ python3 -mtimeit -s 'a=2' "'%s' % a"
10000000 loops, best of 3: 0.197 usec per loop
```

```
$ python3 -mtimeit -s 'a=2' '{}'.format(a)
10000000 loops, best of 3: 0.341 usec per loop
```

```
$ python3 -mtimeit -s 'a=2' 'f"{a}"'
10000000 loops, best of 3: 0.105 usec per loop
```

PEP 498

f-strings



Python 3.6

download at www.python.org



Python 3.6



download at www.python.org



bonus!

16 PEPs included

16 PEPs included

PEP 468

PEP 506

PEP 520

PEP 526

PEP 487

PEP 509

PEP 523

PEP 528

PEP 495

PEP 515

PEP 524

PEP 529

PEP 498

PEP 519

PEP 525

PEP 530

~~16~~ PEPs included

PEP 468

PEP 506

PEP 520

PEP 526

PEP 487

PEP 509

PEP 523

PEP 528

PEP 495

PEP 515

PEP 524

PEP 529

PEP 498

PEP 519

PEP 525

PEP 530

17 ~~16~~ PEPs included

PEP 468

PEP 506

PEP 520

PEP 526

PEP 487

PEP 509

PEP 523

PEP 528

PEP 495

PEP 515

PEP 524

PEP 529

PEP 498

PEP 519

PEP 525

PEP 530

17 ~~16~~ PEPs included

PEP 628: add math.tau

PEP 468

PEP 506

PEP 520

PEP 526

PEP 487

PEP 509

PEP 523

PEP 528

PEP 495

PEP 515

PEP 524

PEP 529

PEP 498

PEP 519

PEP 525

PEP 530

Thank you!

Mariatta Wijaya | mariatta@python.org

file issues at bugs.python.org



imageworks



PYCON CANADA 2017

@mariatta

#PEPtalk