Pyladies & Python 3.8

Laysa Uchoa

@laysauchoa

Pyladies International Community

- Started in 2011 in L.A.
- 100+ chapters around the world
- Promote, educate and advance diversity in the Python community

https://www.pyladies.com/about/

https://www.pyladies.com/locations/

@laysauchoa

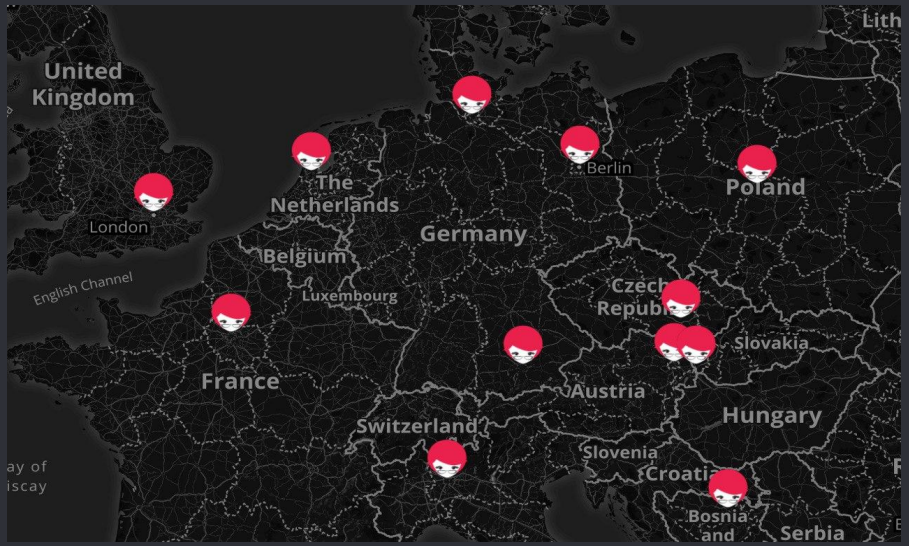
Pyladies Munich

@pyladiesmunich

@laysauchoa

Pyladies Munich 🥨

- Active since July, 2019
- Pyladies in DE: Munich, Berlin, Hamburg and Karlsruhe
- Meetups and Conferences



https://www.meetup.com/PyLadiesMunich/

https://www.facebook.com/PyLadiesMunich/

@laysauchoa

How to support 🥨

- Start your own chapter
- Be a speaker, offer to host the event or sponsor
- Encourage staff to coach at the events or participate

Pyladies Munich 🥨

- Welcome to beginners
- Attendee Policy
- Invite women to accompany you to tech events

About Our Meetups

Join our December Meetup !!! 🎄

December Meetup: https://www.meetup.com/PyLadiesMunich/events/266782688/

@laysauchoa

Updates - Python

- Python 3.8 released in Oct. 14th, 2019
- Python 2.7 will retire soon...
- RIP? ⚰️
- Not yet!

https://www.python.org/downloads/release/python-380/
https://docs.python.org/3/whatsnew/3.8.html

What's New In Python 3.8

- walrus operator (PEP 572)
- positional-only parameters (PEP 570)
- f-strings support for debugging

The Walrus Operator :=

a.k.a assignment expressions or a.k.a. named expressions

@laysauchoa

the walrus operator NAME := expr

- Allow the assignment and return of the value in the same expression
- Reuse of code
- Readability

Examples from the PEP 572

```
match1 = pattern1.match(data)
match2 = pattern2.match(data)
if match1:
    result = match1.group(1)
elif match2:
    result = match2.group(2)
else:
    result = None
```

```
# More efficient way
match1 = pattern1.match(data)
if match1:
    result = match1.group(1)
else:
    match2 = pattern2.match(data)
    if match2:
        result = match2.group(2)
   else:
       result = None
```

```
# With Walrus Operator
if (match1 := pattern1.match(data)):
    result = match1.group(1)
elif (match2 := pattern2.match(data)):
    result = match2.group(2)
else:
    result = None
```

Examples from the PEP 572

```
foo = [f(x), f(x)**2, f(x)**3]
```

```
y = f(x)
foo = [y, y**2, y**3]
```

```
foo = [y := f(x), y**2, y**3]
```

Examples from the PEP 572

```
chunk = file.read(8192)
while chunk:
    process(chunk)
    chunk = file.read(8192)
```

```
while chunk := file.read(8192):
    process(chunk)
```

=

Assignment operator

The walrus operator and assignment statements

```
# Multiple targets ❓✔️
x = y = z = 0
(z := (y := (x := 0)))
```

```
# Single assignment targets other than a single NAME ❌
a[i] = x
self.rest = []
```

```
# Priority around commas is different❓
x = 1, 2  # Sets x to (1, 2)
(x := 1, 2)  # Sets x to 1
```

@laysauchoa

The walrus operator and assignment statements

```
# Packing ✔
loc = x, y
(loc := (x, y))

info = name, phone, *rest # ✔
(info := (name, phone, *rest))
```

```
# No equivalent to Unpacking ❌
px, py, pz = position
name, phone, email, *other_info = contact
```

PEP 572 - Community Response

- PEP 572 was controversial
- PEP 572 was accepted by Guido van Rossum
- Guido van Rossum was Python's BDFL
- After that, Guido stepped down from his BDFL role

BDFL - Benevolent Dictator for Life

PEP 572 - Guido acceptance

“Thank you all. I will accept the PEP as is.

*I am happy to accept *clarification* updates to the PEP if people care to submit them as PRs.”*

Guido van Rossum

https://groups.google.com/forum/#!topic/dev-python/egS7A22oJgE%5B1-25%5D

@laysauchoa

/

Positional-Only Parameters

@laysauchoa

Keyword-Only Arguments - Python 3.0

```
In[8]: def bar(a, b, *, c):
  ...:     return a, b, c
  ...:
✔
In[9]: bar(1, 2, c=3) == bar(a=1, b=2, c=3) == bar(b=2, a=1, c=3)
Out[9]: True
```

```
❌
In[10]: bar(1, 2, 3)
---------------------------------------------------------------------------
TypeError                                 Traceback(most recent call last)
<ipython-input-10-2ed2fd028fab > in < module >
--- -> 1 bar(1, 2, 3)

TypeError: bar() takes 2 positional arguments but 3 were given
```

Positional-Only Parameters PEP 570

```
In [16]: def bar(a, /, b, *, c):
    ...:     return a, b, c
    ...:
    ...:
```

❌

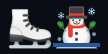
```
In[17]: bar(a=1, b=2, c=3)
---------------------------------------------------------------------------
TypeError                                 Traceback(most recent call last)
<ipython-input-17-d762e4b5480c > in < module >
--- -> 1 bar(a=1, b=2, c=3)

TypeError: bar() got some positional-only arguments passed as keyword arguments:
'a'
```

f"{name =}"

Debug support for f-strings

@laysauchoa

Debug support for f-strings



```
In[29]: ice_skating = "Schlittschuhlaufen"

In[30]: print(f'ice skating = {ice_skating}')
ice skating = Schlittschuhlaufen

In[31]: print(f'{ice_skating =}')
ice_skating = 'Schlittschuhlaufen'
```

Let us try out Python 3.8

And have some fun! 

Thank you, all!

@laysauchoa

ANY QUESTIONS?

You can find me here or @laysauchoa

@laysauchoa

END !

@laysauchoa