

PRÉSENTATION DE friendly

AFPy - 20 mai 2021

ANDRÉ ROBERGE

<https://github.com/aroberge/friendly>

BUTS:

- Présenter **friendly**.
- Vous motiver à faire des suggestions.

- Introduction et survol
- Démonstrations

QU'EST-CE QUE C'EST ?

Illustration sur HackInScience






À l'origine:

UnboundLocalError: ...

-> Qu'est-ce que ça veut dire?
... avec traduction possible.

Ceci est devenu **what()** qui sera démontré ...

Puis est venu s'ajouter **where()** ...

- ☐  **RecursionError** New case
#65 by JulienPalard was closed on 1 Aug 2020
- ☐  **[idea] Propose near matches on attribute error.**
#62 by JulienPalard was closed on 28 Jul 2020
- ☐  **Dead code spotted**
#59 by JulienPalard was closed on 27 Jul 2020
- ☐  **Internal error not being an internal error?**
#58 by JulienPalard was closed on 27 Jul 2020
- ☐  **Newcomers copy/pasting from repl**
#52 by JulienPalard was closed on 26 Jul 2020

Ceci a mené à `why()` ...

... `hint()` ...

... `www()` ...

DIFFÉRENTS ENVIRONNEMENTS

```
# terminal, VS Code, etc.  
from friendly          import ...
```

```
# cas particuliers  
from friendly.idle     import ...  
from friendly.mu       import ...  
from friendly.ipython  import ...  
from friendly.jupyter  import ...
```


OBSERVATION

Exécuter un programme avec
IDLE, Mu, Thonny, ...

```
python -i mon_programme.py
```

```
# mon_programme.py  
  
réponse = 42  
print("Bonjour AFPy !")
```

Exécution:

```
Bonjour AFPy !  
>>> réponse  
42  
>>>
```

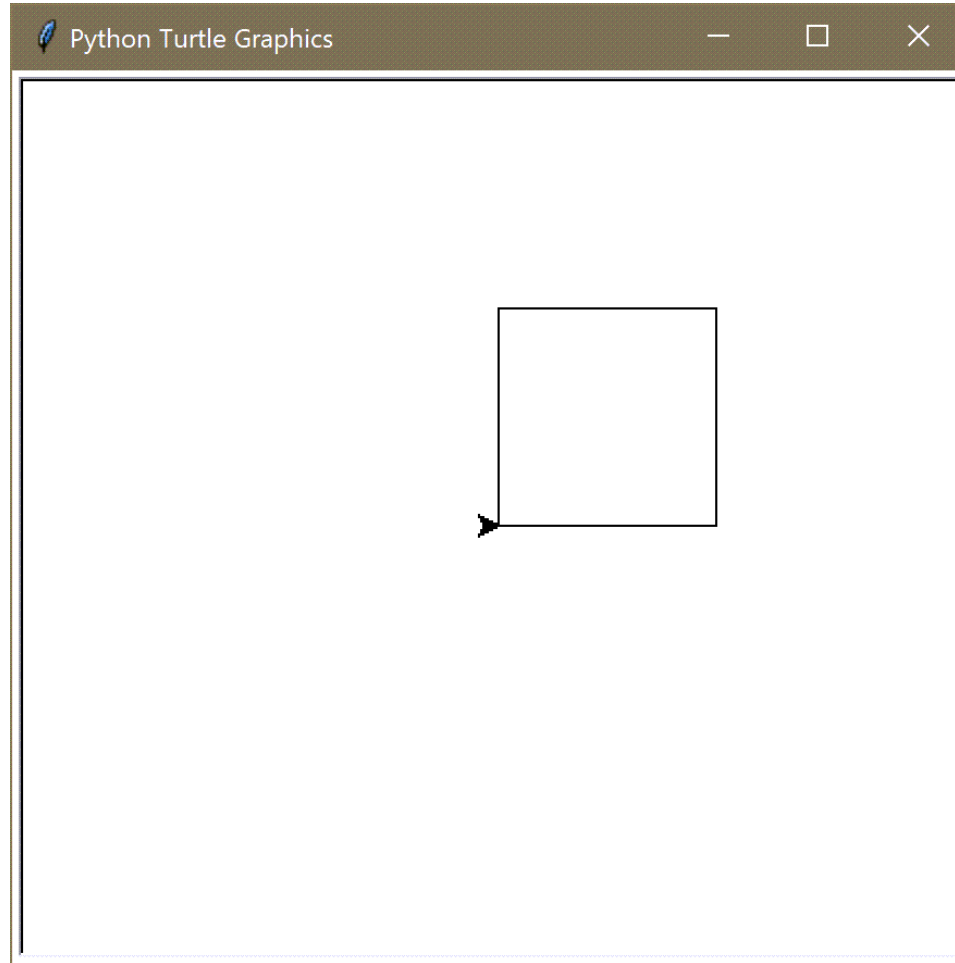
DÉMO 1

```
import turtle as t

for i in range(4):
    t.forward(100)
    t.left(90)
```



DÉMO 1: CE QU'ON VEUT...



DÉMO

```
>>> set_lang('fr')    # si requis  
>>> hint()  
>>> what()  
>>> where()  
>>> why()  
>>> www()
```

Si la démo ne fonctionne pas...



DEMO 1: RESULT

```
> python turtle.py
```

```
Traceback (most recent call last):
```

```
  File "turtle.py", line 3, in <module>
```

```
    import turtle as t
```

```
  File "...\\turtle.py", line 6, in <module>
```

```
    t.forward(100)
```

```
AttributeError: partially initialized module
```

```
  'turtle' has no attribute 'forward'
```

```
(most likely due to a circular import)
```

DEMO 1: RUNNING WITH friendly

```
python -m friendly turtle.py
```

```
> python -m friendly turtle.py

Traceback (most recent call last):
  File "turtle.py", line 3, in <module>
    import turtle as t
  File "CWD:\turtle.py", line 6, in <module>
    t.forward(100)
AttributeError: partially initialized module 'turtle' has no attribute 'forward'
(most likely due to a circular import)

Did you give your program the same name as a Python module?

An AttributeError occurs when the code contains something like object.x and x is
not a method or attribute (variable) belonging to object.

I suspect that you used the name turtle.py for your program and that you also
wanted to import a module with the same name from Python's standard library. If
so, you should use a different name for your program.

Execution stopped on line 3 of file 'turtle.py'.

1: # Draw a square
2:
-->3: import turtle as t

Exception raised on line 6 of file 'CWD:\turtle.py'.

4:
5: for i in range(4):
-->6:     t.forward(100)
      ^^^^^^^^^
7:     t.left(90)

t: <module turtle> from CWD:\turtle.py
```


DEMO 1: USING FRIENDLY

```
python -im friendly turtle.py
```

```
...
```

```
Friendly Console ...
```

```
>>>
```

DEMO 1: TRACEBACK AND `hint()`

DEMO 1: what()

```
>>> what()
```

AttributeError: partially initialized module 'turtle' has no attribute 'forward' (most likely due to a circular import)

An **AttributeError** occurs when the code contains something like `object.x` and `x` is not a method or attribute (variable) belonging to `object`.

DEMO 1: why()

```
>>> why()
```

I suspect that you used the name `turtle.py` for your program and that you also wanted to import a module with the same name from Python's standard library. If so, you should use a different name for your program.

DEMO 1: where()

```
>>> where()
```

Execution stopped on line 3 of file 'turtle.py'.

```
1: # Draw a square
2:
-->3: import turtle as t
```

Exception raised on line 6 of file 'CWD:\turtle.py'.

```
4:
5: for i in range(4):
-->6:     t.forward(100)
      ^^^^^^^^^^
7:     t.left(90)
```

```
t: <module turtle> from CWD:\turtle.py
```

Only one or two frames are shown.

DEMO 1: www()

```
>>> www()
```



AttributeError: partially initialized module turtle has no attribute forward



Tout Images Vidéos Actualités Carte

Préférences ▼



Canada (fr) ▼

Filtre parental : modérée ▼

À tout moment ▼

AttributeError: partially initialized module 'turtle' has ...



<https://stackoverflow.com/questions/60480328/attributeerror-partially-initialized-mod...>

AttributeError: partially initialized module 'turtle' has no attribute 'Turtle' (most likely due to a circular import)

attributeerror: partially initialized module 'turtle' has ...



<https://stackoverflow.com/questions/65962607/attributeerror-partially-initialized-mod...>

AttributeError: partially initialized module 'turtle' has no attribute 'Turtle' (most likely due

DEMO 1: what() IN FRENCH

```
>>> set_lang('fr')  
>>> what()
```

AttributeError: partially initialized module 'turtle' has no attribute 'forward' (most likely due to a circular import)

Une exception **AttributeError** se produit lorsque le code contient quelque chose comme **object.x** et **x** n'est pas une méthode ou un attribut (variable) appartenant à **objet**.

DEMO 1: what(...)

```
>>> set_lang('en')  
>>> what(UnboundLocalError)
```

In Python, variables that are used inside a function are known as local variables. Before they are used, they must be assigned a value. A variable that is used before it is assigned a value is assumed to be defined outside that function; it is known as a **global** (or sometimes **nonlocal**) variable. You cannot assign a value to such a global variable inside a function without first indicating to Python that this is a global variable, otherwise you will see an **UnboundLocalError**.

"DEMO 1": SUMMARY

We can use **friendly** to ask questions and obtain answers helping us understand what caused a given traceback.

FAUTES DE FRAPPE?

```
>>> calcul_long()  
Traceback (most recent call last):  
...  
>>> whyy()  
Traceback (most recent call last):  
...
```

Est-ce qu'un REPL peut être convivial?

Friendly Console version 0.3.45. [Python version: 3.10.0b1]

```
>>> if "word" := True:
```

Traceback (most recent call last):

File "<friendly-console:1>", line 1

```
    if "word" := True:
        ^
```

SyntaxError: cannot use assignment expressions with literal

You can only assign objects to identifiers (variable names).

```
>>> whyy()
```

```
Traceback (most recent call last):
```

```
File "<friendly-console:2>", line 1, in <module>
```

```
    whyy()
```

```
NameError: name 'whyy' is not defined
```

```
Did you mean why?
```

```
>>>
```



Did you mean **why**?

```
>>> why()
```

In your program, **whyy** is an unknown name. The similar name **why** was found in the local scope.

```
>>> |
```

```
>>> history()  
SyntaxError: cannot use assignment expressions with literal  
NameError: name 'whyy' is not defined  
>>> |
```

```
>>> back()
>>> history()
SyntaxError: cannot use assignment expressions with literal
>>> |
```

```
>>> why()
```

You cannot use the augmented assignment operator `:=`, sometimes called the walrus operator, with literals like `"word"`. You can only assign objects to identifiers (variable names).

Démo avec Jupyter Lab



POURQUOI *import* *?

```
>>> dir()
[
    'Friendly',
    '__builtins__',
    '_get_statement',
    'back',
    'debug',
    'debug_tb',
    'explain',
    'friendly_tb',
    'get_include',
    'get_lang',
    'hint',
    'history',
    'more',
    'python_tb',
    'set_formatter',
    'set_include',
    'set_lang',
    'show_info',
    'show_paths',
    'what',
    'where',
    'why',
    'www'
]
```

Friendly.why() == why(), etc

OÙ EST LE CODE?

<https://github.com/aroberge/friendly>

Vous y trouverez un lien menant à la documentation...

... mais il y a une façon plus *pythonique* ...

Documentation ? 😊

```
> python -m pip install friendly  
> python -m friendly
```

```
Friendly Console version ...
```

```
>>> www()
```



REMERCIEMENTS

- Nicholas Tollervey ([Mu](#))
- Aivar Annamaa ([Thonny](#))
- Julien Palard ([HackInScience](#))
- Alex Hall ([futurecoder](#) + plusieurs modules)
- Sylvain Desodt ([DidYouMean-Python](#) inspiré par Raymond Hettinger)
- Michael Kennedy, Brian Okken, et Hannah Stepanek dans [PythonBytes podcast #220](#)
- *et plusieurs autres ... et pourquoi pas vous?*

FIN

D'autres diapos en cas de problèmes ...



IDLE

DEMO



```
from friendly.idle import run  
run("test.py")
```

If the demo does not work.



Python 3.8.4 (tags/v3.8.4:dfa645a, Jul 13 2020, 16:30:28) [

```
>>> from friendly.idle import start_console
>>> start_console()
```

Friendly Console version 0.3.45. [Python version: 3.8.4]

```
>>> def pass():
```

Traceback (most recent call last):

File "<friendly-console:1>", line 1

```
def pass():
    ^
```

SyntaxError: invalid syntax

You cannot use a Python keyword as a function name.

Python 3.10.0b1 (tags/v3.10.0b1:ba42175, May 3 2021, 20:22:30) [M

```
>>> from friendly.idle import *  
>>> install()
```

WARNING

Friendly cannot handle SyntaxErrors for code entered in the shell.

```
>>> import Turtle
```

Traceback (most recent call last):

File "<pyshell#2>" line 1, in <module>

import Turtle

ModuleNotFoundError: No module named 'Turtle'

Did you mean turtle?

```
test.py - C:\Users\andre\test.py (3.8.4)
File Edit Format Run Options Window Help
from math import *

a = cost(pi)
```

Ln: 4 Col: 0

```
friendly_run.py - C:\Users\andre\friendly_run.py (3.8.4)
File Edit Format Run Options Window Help
from friendly.idle import run

run("test.py", lang="fr")
```

Ln: 4 Col: 0

===== RESTART: C:\Users\andre\friendly_run.py =====

Traceback (most recent call last):

File "CWD:\test.py", line 3, in <module>

a = cost(pi)

NameError: name 'cost' is not defined

Voulez-vous dire cos ?

>>> why()

Dans votre programme, cost est un nom inconnu.

Au lieu d'écrire cost, peut-être que vous vouliez écrire l'un des noms suivants :

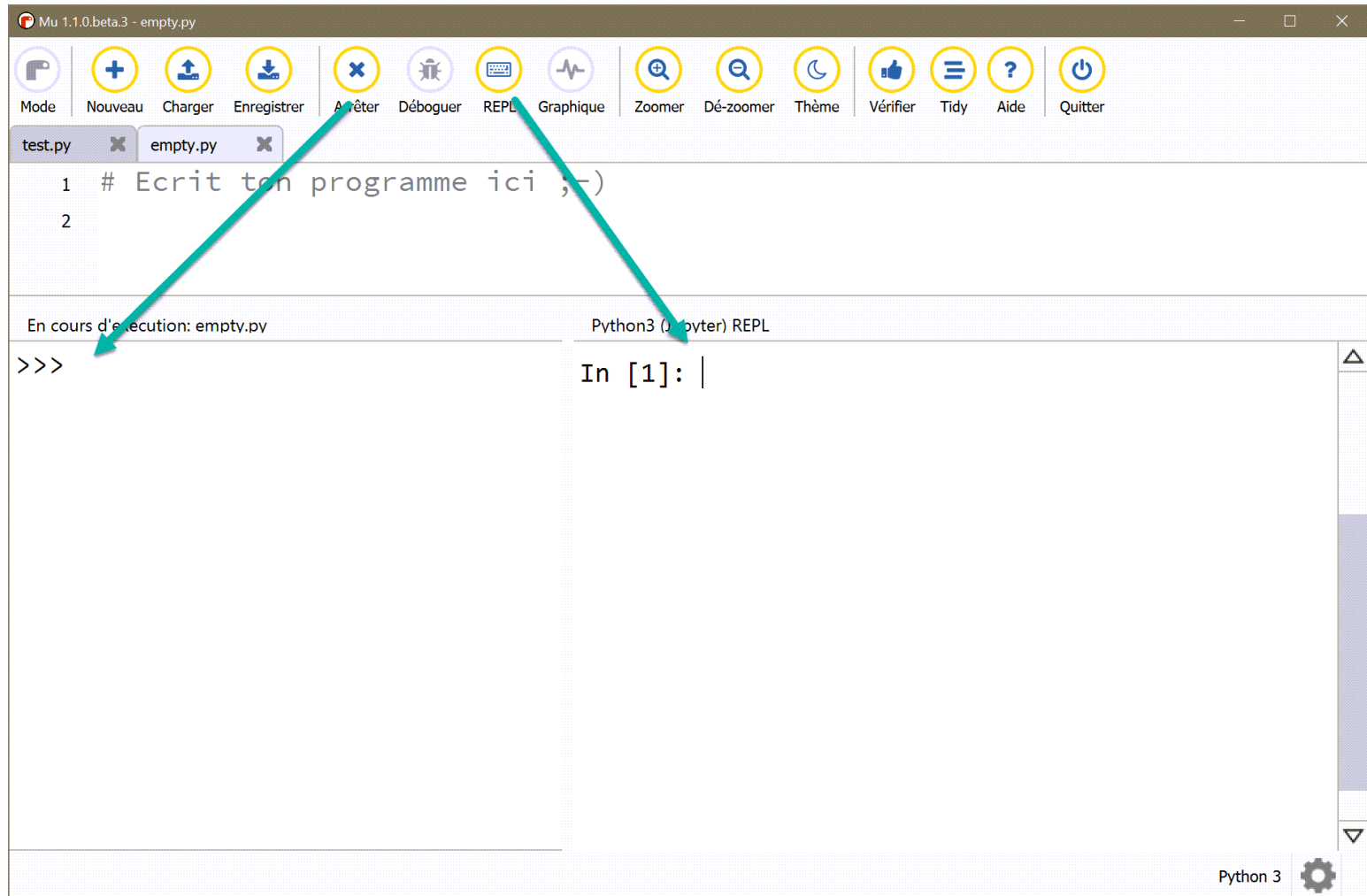
* Portée locale : cos, cosh, acos

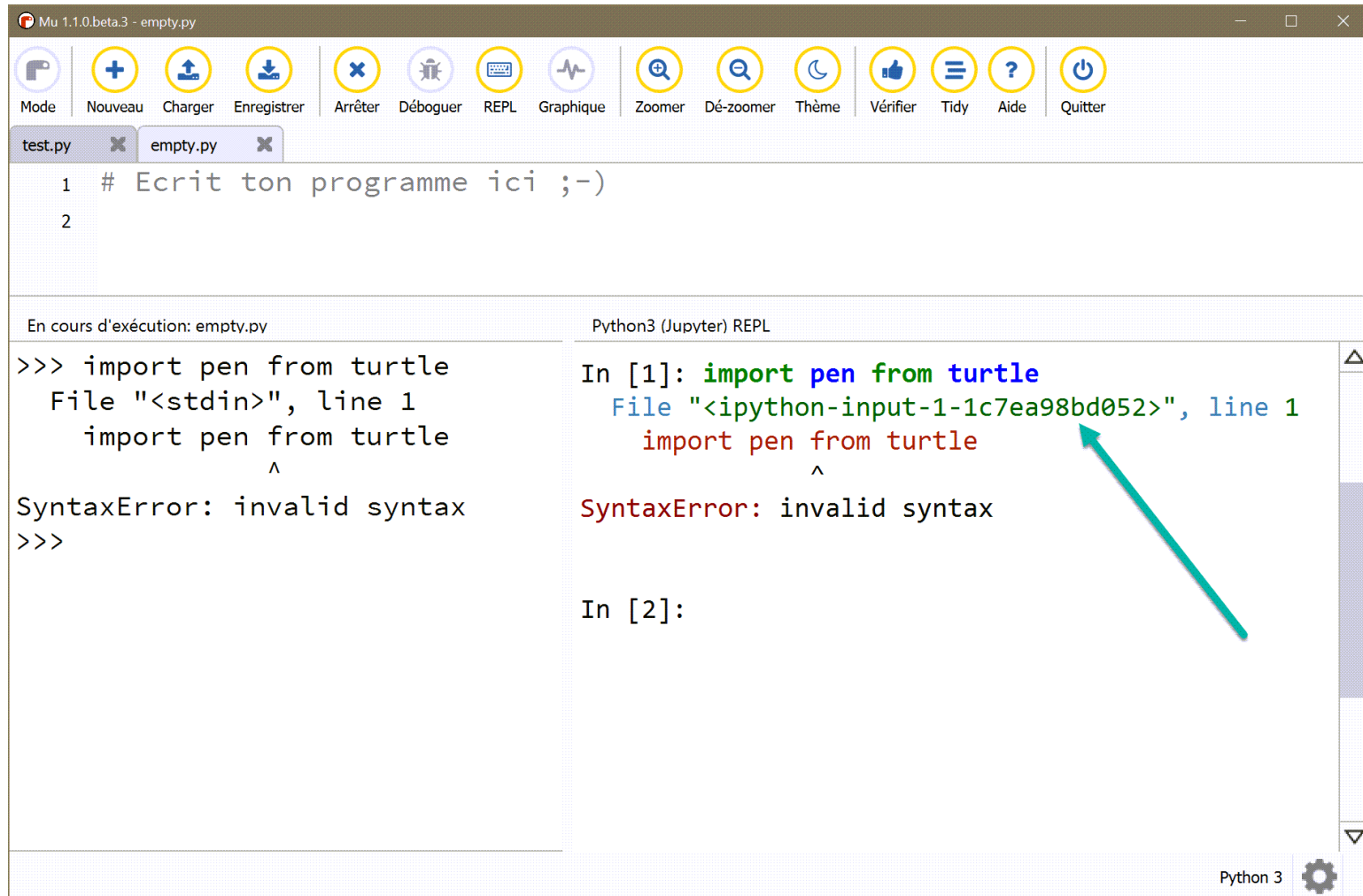


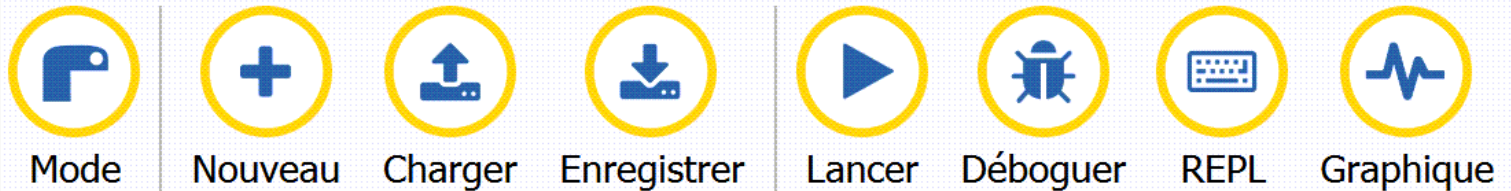
DEMO

If the demo does not work.









test.py

empty.py

```
1 a = [1, 2, 3]
2 print(a[1], a[2], a[3])
3
```

Python3 (Jupyter) REPL

In [1]: run test.py

IndexError

Traceback (most recent call last)

~\mu_code\test.py in <module>

```
1 a = [1, 2, 3]
```

```
----> 2 print(a[1], a[2], a[3])
```

IndexError: list index out of range

Mu 1.1.0.beta.3 - empty.py

Mode Nouveau Charger Enregistrer Arrêter Débugger REPL Graphique Zoomer Dé-zoomer Thème Vérifier Tidy

test.py empty.py

```
1 # Ecrit ton programme ici ;-)  
2
```

En cours d'exécution: empty.py

```
>>>  
>>>  
>>> import pen from turtle  
File "<stdin>", line 1  
    import pen from turtle  
      ^  
SyntaxError: invalid syntax  
>>>
```

Python3 (Jupyter) REPL

```
In [1]: from friendly.mu import *  
  
In [2]: import pen from turtle  
  
Traceback (most recent call last):  
File "In [2]", line 1  
    import pen from turtle  
      ^  
SyntaxError: invalid syntax  
  
Did you mean from turtle import pen?  
  
In [3]:
```

Python 3

In [4]: why()

You have tried to get the item with index 3 of `a`, a `list` of length 3. The largest valid index of `a` is 2.

In [5]: where()

Execution stopped on line 1 of file `'In [3]'`.

```
-->1: import test
```

Exception raised on line 2 of file `'CWD:\test.py'`.

```
1: a = [1, 2, 3]
-->2: print(a[1], a[2], a[3])
      ^^^^
```

```
a: [1, 2, 3]
```

Python3 (Jupyter) REPL

```
1: a = [1, 2, 3]
-->2: print(a[1], a[2], a[3])
      ^^^^
```

```
a: [1, 2, 3]
```

In [6]: night() 

In [7]: where()

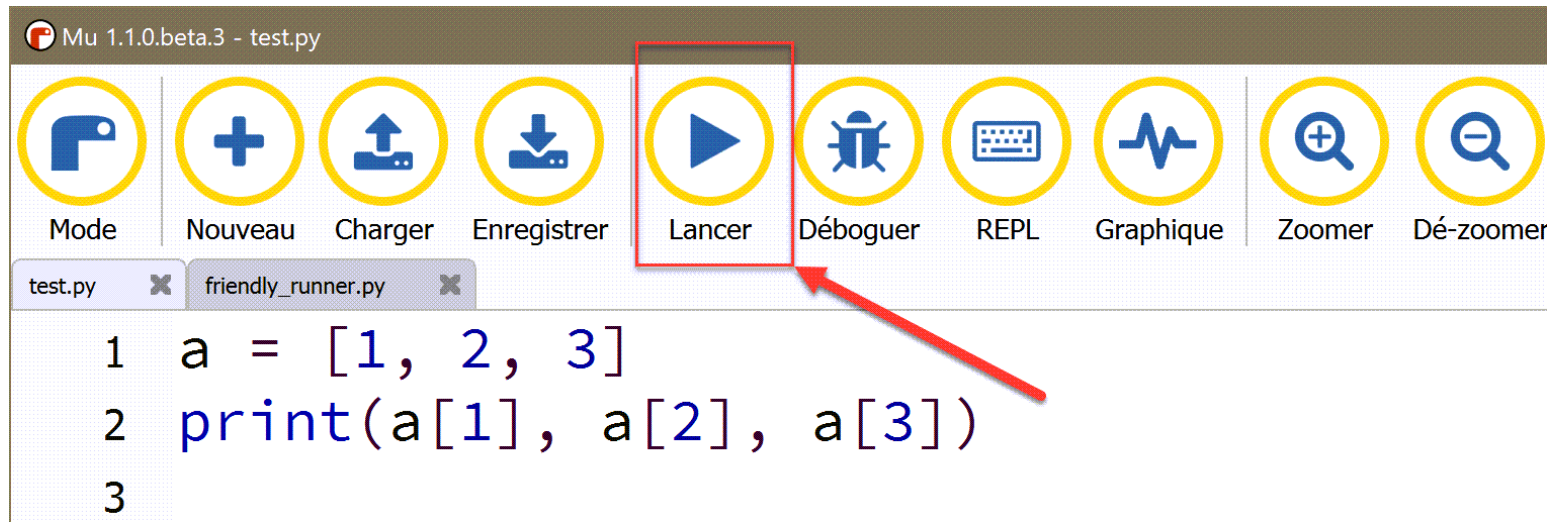
Execution stopped on line 1 of file 'In [3]'.

```
-->1: import test
```

Exception raised on line 2 of file 'CWD:\test.py'.

```
1: a = [1, 2, 3]
-->2: print(a[1], a[2], a[3])
      ^^^^
```

```
a: [1, 2, 3]
```



Mu 1.1.0.beta.3 - test.py

Mode Nouveau Charger Enregistrer Arrêter Déboguier REPL Graphique Zoomer Dé-zoomer Thème Vei

test.py x friendly_runner.py x

```
1 a = [1, 2, 3]
2 print(a[1], a[2], a[3])
3
```

En cours d'exécution: test.py

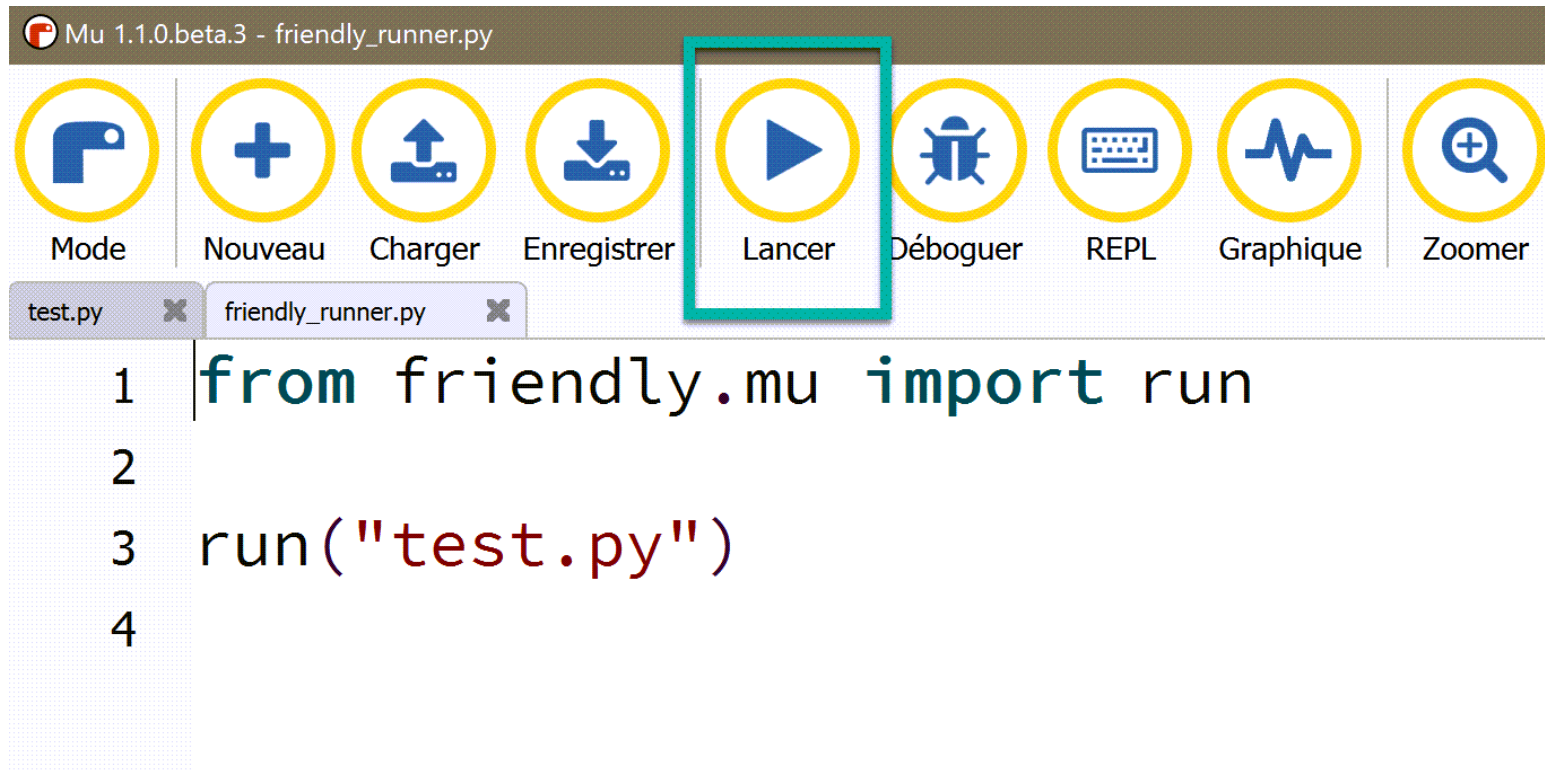
Traceback (most recent call last):

File "c:\users\andre\mu_code\test.py", line 2, in <module>

print(a[1], a[2], a[3])

IndexError: list index out of range

>>> |





Traceback (most recent call last):

```
File "CWD:\test.py", line 2, in <module>
    print(a[1], a[2], a[3])
IndexError: list index out of range
```

Remember: the first item of a `list` is at index 0.

```
>>> why()
```

You have tried to get the item with index `3` of `a`,
a `list` of length `3`.

The largest valid index of `a` is `2`.

```
>>>
```




jupyter

DEMO?



If the demo does not work ...

Untitled.ipynb

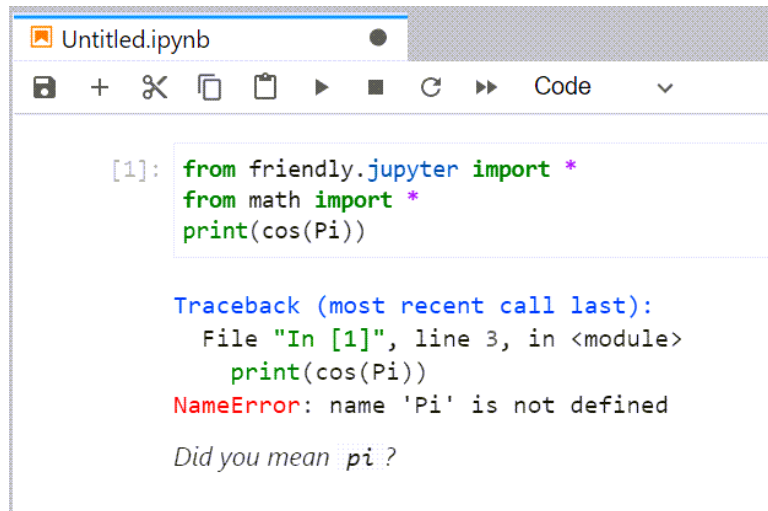
📁 + ✂ 📄 ▶ ■ ↺ ▶▶ Code ▼

```
[1]: from math import *  
     print(cos(Pi))
```

```
-----  
NameError                                Traceback (most recent call last)  
<ipython-input-1-1d0d2b73b38b> in <module>  
      1 from math import *  
----> 2 print(cos(Pi))
```

```
NameError: name 'Pi' is not defined
```

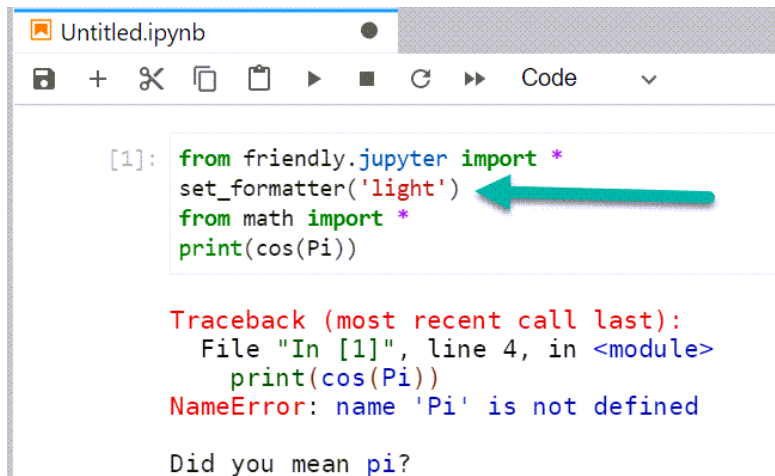
```
from friendly.jupyter import *
```



Untitled.ipynb

```
[1]: from friendly.jupyter import *  
from math import *  
print(cos(Pi))
```

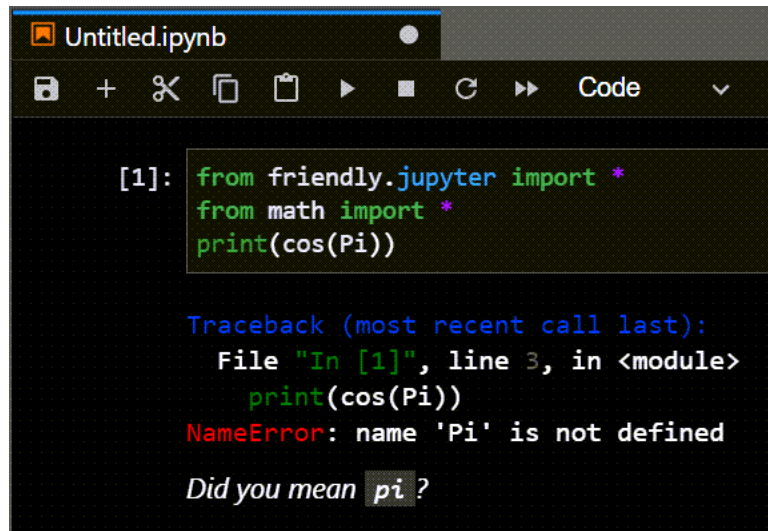
Traceback (most recent call last):
File "In [1]", line 3, in <module>
print(cos(Pi))
NameError: name 'Pi' is not defined
Did you mean pi?



Untitled.ipynb

```
[1]: from friendly.jupyter import *  
set_formatter('light')  
from math import *  
print(cos(Pi))
```

Traceback (most recent call last):
File "In [1]", line 4, in <module>
print(cos(Pi))
NameError: name 'Pi' is not defined
Did you mean pi?

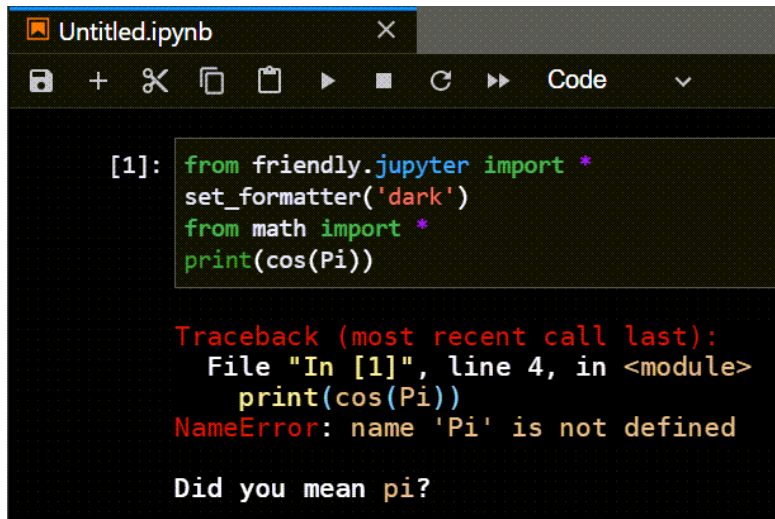


Untitled.ipynb

```
[1]: from friendly.jupyter import *  
    from math import *  
    print(cos(Pi))
```

Traceback (most recent call last):
 File "In [1]", line 3, in <module>
 print(cos(Pi))
NameError: name 'Pi' is not defined

Did you mean `pi`?

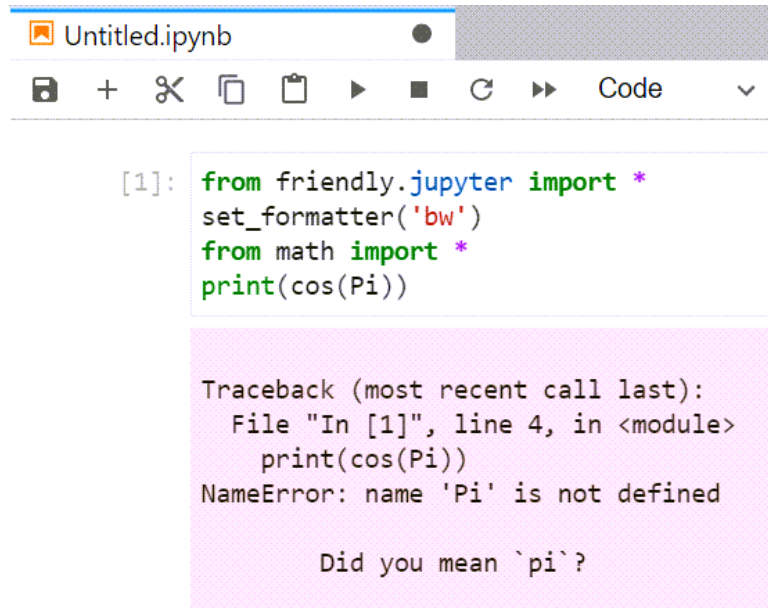


Untitled.ipynb

```
[1]: from friendly.jupyter import *  
    set_formatter('dark')  
    from math import *  
    print(cos(Pi))
```

Traceback (most recent call last):
 File "In [1]", line 4, in <module>
 print(cos(Pi))
NameError: name 'Pi' is not defined

Did you mean `pi`?



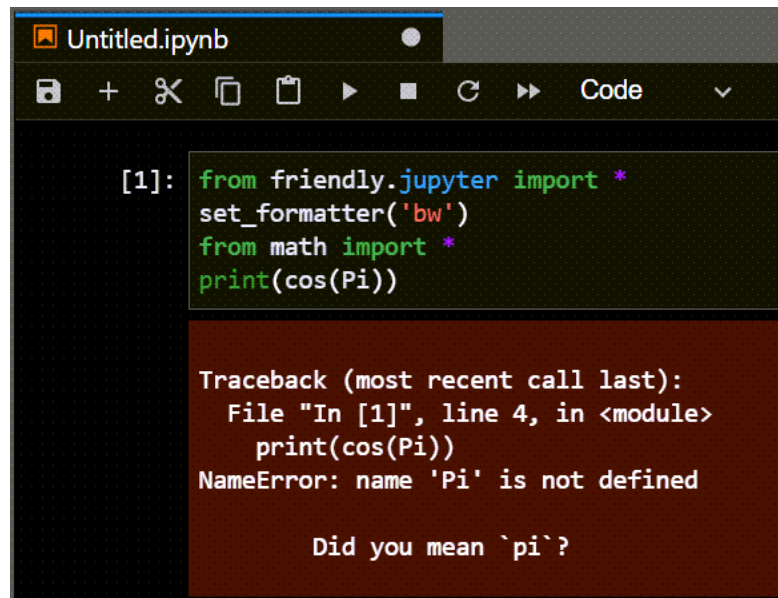
Untitled.ipynb

Code

```
[1]: from friendly.jupyter import *  
     set_formatter('bw')  
     from math import *  
     print(cos(Pi))
```

Traceback (most recent call last):
 File "In [1]", line 4, in <module>
 print(cos(Pi))
NameError: name 'Pi' is not defined

Did you mean `pi`?



Untitled.ipynb

Code

```
[1]: from friendly.jupyter import *  
     set_formatter('bw')  
     from math import *  
     print(cos(Pi))
```

Traceback (most recent call last):
 File "In [1]", line 4, in <module>
 print(cos(Pi))
NameError: name 'Pi' is not defined

Did you mean `pi`?

friendly used to be called **friendly-traceback**.

```
>>> import this
...
Explicit is better than implicit.
...
```

friendly-traceback is more explicit than **friendly**.

Why was the name changed?

```
from friendly          import ...
from friendly.idle     import ...
from friendly.mu       import ...
from friendly.ipython  import ...
from friendly.jupyter  import ...
```

are better than

```
from friendly_traceback.idle import ...
```

Besides, *traceback* is not exactly a beginner-friendly term and, in the future, **friendly** might do more than simply helping with tracebacks ...



Reuven M. Lerner

@reuvenmlerner

...

Today, a **#Python** student's code didn't print:

```
x = 5
if x == 5:
    print: ('yes!')
```

There was a typo, namely : after print. But: Huh?

Python sees this as a type annotation to print, but without assignment. So no output, **no warning**, much confusion!

WARNINGS

```
Friendly Console version 0.2.38. [Python version: 3.8.4]
```

```
>>> a : int
```

```
Warning: you used a type hint for a variable without assigning it a value.
```

```
Do you find these warnings useful?
```

```
Comment at https://github.com/aroberge/friendly-traceback/issues/112
```

```
>>> list = [1, 2, 3]
```

```
Warning: you have redefined the python builtin list.
```

```
Do you find these warnings useful?
```

```
Comment at https://github.com/aroberge/friendly-traceback/issues/112
```

```
>>> |
```

Only works in the friendly console, not with custom modules.

friendly is currently at version 0.3.55

Try it!

Let me know how it could be improved
for you and your students.