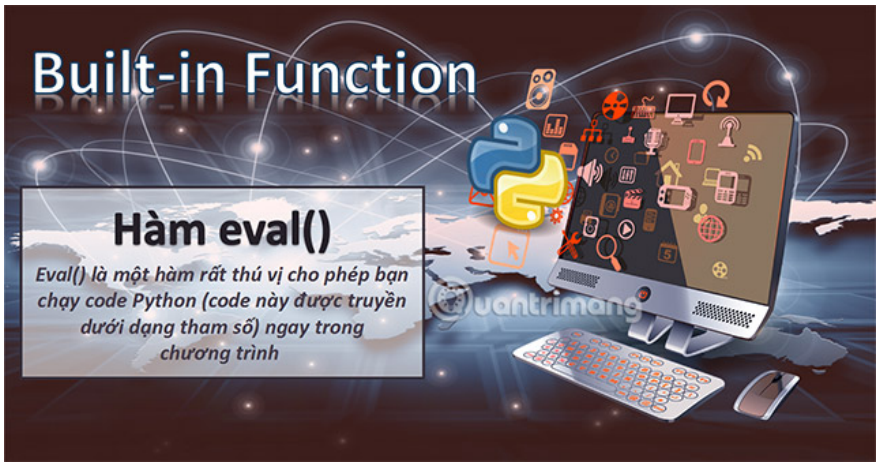
The eval () function in Python.

In this article, Quantrimang will continue to introduce you to a built-in function in Python, eval (). Eval () is an interesting utility that allows you to run Python code (this code is passed as parameters) right in the program.

In this article, Quantrimang will continue to introduce you to a built-in function in Python, *eval ()*. *Eval ()* is an interesting utility that allows you to run Python code (this code is passed as parameters) right in the program.



The syntax of eval () function in Python

```
eval(bieuthuc, global=None, local=None)
```

The parameters of the eval () function:

The eval () function has three parameters:

1. *bieuthuc:* can be any valid Python expression
2. *global:* a dictionary that specifies methods and global variables available.
3. *local:* a dictionary specifies local methods and variables available.

Global and *local use* will be analyzed by Quantrimang later in this article.

Value returned from eval ()

Eval () returns results made from *bieuthuc*

Example 1: How does the eval () function work?

```
>>> eval("5 == 5")
True
>>>
>>> eval("4  10")
True
>>>
>>> eval("8 + 4 - 2 * 3")
6
>>>
>>> eval("'py ' * 5")
'py py py py py '
>>>
>>> eval("10 ** 2")
100
>>>
>>> eval("'hello' + 'py'")
'hellopy'
```

Eval () not only performs simple expressions but can also execute functions, call methods, reference variables .

```
>>> eval("abs(-11)")
11
>>>
>>>
>>> eval('"hello".upper()')
'HELLO'
>>>
>>>
>>> import os
>>>
>>>
>>> eval('os.getcwd()') # hi?n th? th? m?c ?ang làm vi?c hi?n t?i
'/home/thepythonguru'
>>>
>>>
>>> x = 2
>>>
>>> eval("x+4")
6
```

Example 2: Using eval ()

```
# Vi?t b?i TipsMake.com
# Chu vi hinh vuong
def tinhChuvi(l):
 return 4*l

# Dien tich hinh vuong
def tinhDientich(l):
 return l*1
```

```
thucthi = input("Nh?p hàm s? d?ng: ")

for l in range(1, 5):
 if (thucthi == 'tinhChuvi(l)'):
 print("N?u ?? dài b?ng ", l , ", Chu vi = ", eval(thucthi))
 elif (thucthi == 'tinhDientich(l)'):
 print("N?u ?? dài b?ng ", l , ", Di?n tích = ", eval(thucthi))
 else:
 print('Hàm không chính xác!')
 break
```

Run the program, the result is:

```
Nh?p hàm s? d?ng: tinhDientich(l)
N?u ?? dài b?ng 1 , Di?n tích = 1
N?u ?? dài b?ng 2 , Di?n tích = 2
N?u ?? dài b?ng 3 , Di?n tích = 3
N?u ?? dài b?ng 4 , Di?n tích = 4

Nh?p hàm s? d?ng: tinhChuvi(l)
N?u ?? dài b?ng 1 , Chu vi = 4
N?u ?? dài b?ng 2 , Chu vi = 8
N?u ?? dài b?ng 3 , Chu vi = 12
N?u ?? dài b?ng 4 , Chu vi = 16

Nh?p hàm s? d?ng: tinhdientich
Hàm không chính xác!
```

You should be careful when using eval!

You need to pay a little attention if you are using a Unix system like macOS, Linux . and perform the functions of the os module. The os module is built to provide methods to help you create, delete, and change folders.

Then, if you enter the value as *eval (input ())*, the user may have problems changing the file even if it deletes the file with the *os.system ('rm -rf *')* command .

So, if you use *eval (input ())* in your code, it is reasonable to check which variables and methods you can use. You should do this with the function *dir ()*.

```
from math import *
print(eval('dir()'))
```

Running the code and the resulting result will look like this:

```
['__annotations__', '__builtins__', '__doc__', '__file__', '__loader__', '__name
 '__spec__', 'acos', 'acosh', 'asin', 'asinh', 'atan', 'atan2', 'atanh', 'ceil',
 'cosh', 'degrees', 'e', 'erf', 'erfc', 'exp', 'expm1', 'fabs', 'factorial', 'fl
 , 'fsum', 'gamma', 'gcd', 'hypot', 'inf', 'isclose', 'isfinite', 'isinf', 'isna
 , 'log', 'log10', 'log1p', 'log2', 'modf', 'nan', 'pi', 'pow', 'radians', 'rema
 , 'sqrt', 'tan', 'tanh', 'tau', 'trunc']
```

The case does not pass global and local parameters

When using *eval ()* without passing these two parameters as the examples above, the expression will be executed in the current scope. You can check the variables and methods available with dir () as mentioned above.

```
print(eval('dir()'))
```

Where to pass global parameters and ignore local

Global and *local* (dictionary) parameters are used for global and local variables respectively. If *dictionary local* is ignored, this default program is in the *global* form . That is, *global* will be used for both global and local variables.

You can check local and global dictionary with *global ()* and *local ()* built-in functions *()*

Example: Transfer an empty dictionary to the global parameter

```
from math import *
print(eval('dir()', {}))

# Code d??i ?ây s? x?y ra exception
# print(eval('sqrt(25)', {}))
```

If you pass a blank dictionary into the *global* , only the *__builtins__* function is valid for *bieuthuc.* You will not be able to access any function of the math module even though the math module has been imported into the program above.

Run the program, the result is:

```
['__builtins__']
```

In this example, the expression can use the *sqrt ()* and *pow () functions* together with *__builtins__*.

```
from math import *
print(eval('dir()', {'sqrt': sqrt, 'pow': pow}))
```

Alternatively, you can change the name of the available method for your desired expression.

```
from math import *
print(eval('dir()', {'canbachai': sqrt, 'luythua': pow}))

# dùng canbachai trong bi?u th?c
print(eval('canbachai(9)', {'canbachai': sqrt, 'luythua': pow}))
```

The case transmits both global and local parameters

```
from math import *

a = 5
print(eval('sqrt(a)', {'__builtins__': None}, {'a': a, 'sqrt': sqrt}))
```

Run the program, the result is returned

```
2.23606797749979
```

Previous article: enumerate () function in Python

Next lesson: float () function in Python

You finished reading the article "**The eval () function in Python.**" edited by the **TipsMake** team. We hope this article has provided you with many useful tech tips and tricks. You can search for similar articles on tips and guides. Thank you for reading and for following us regularly.

© 2019 TipsMake.com