

Python з нуля

Zen of Python

Богдан Кулинич

Web Academy

24 лютого 2015 р.

Повторення

- ▶ Змінна це просто назва об'єкту в пам'яті
- ▶ `item in collection`
 - перевіряє чи є об'єкт в колекції

```
some_list = [1, 2, 3]
3 in some_list # True
[3] in some_list # False
```

- ▶ `True` і `False` – булеві значення, `'True'` і `'False'` — рядки

Повторення

- ▶ print можна використовувати багато разів у функції, return одразу завершує виконання функції.

```
def is_prime(number):  
    ...for divisor in range(2, number):  
        ...if (number % divisor) != 0:  
            ...return True  
    ...return False
```

```
def is_prime(number):  
    ...for divisor in range(2, number):  
        ...if (number % divisor) == 0:  
            ...return False  
    ...return True
```

```
► def is_prime(number):  
    ...for divisor in range(2, number):  
        ...if (number % divisor) != 0:  
            ...return True  
    ...return False
```

Повторення

- ▶ `a is b` перевіряє чи `a` і `b` є одним і тим самим об'єктом, а `a == b` перевіряє, чи вони рівні

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

```
b = [1, 2]
```

```
c = a
```

```
a is b # False
```

```
a == b # True
```

```
a is c # True
```

- Ділення повертає float в Python 3

```
a = 1 / 2  
type(a) # float  
a = 1 // 2  
type(a) # int
```

- ▶ Debugging print from `ipdb import set_trace`
- ▶ Rubber duck debugging

Zen of Python

- ▶ import this

Zen of Python

- ▶ Beautiful is better than ugly

```
# if (is_valid(a) && b == 0 || s == 'yes') {}  
if is_valid(a) and b == 0 or s == 'yes': ...
```

- ▶ Explicit is better than implicit

```
# Bad  
from os import *  
print(getcwd())  
  
# Good  
import os  
print(os.getcwd())
```

Zen of Python

- ▶ Namespaces are one honking great idea – let's do more of those!
- ▶ Sparse is better than dense.

```
# Bad  
if i>0: return sqrt(i)  
elif i==0: return 0  
else: return 1j * sqrt(-i)
```

```
# Good  
if i > 0:  
    ...return sqrt(i)  
elif i == 0:  
    ...return 0  
else:  
    ...return 1j * sqrt(-i)
```

Zen of Python

- ▶ Readability counts
- ▶ Errors should never pass silently.

```
try:  
    ....import this  
except ImportError:  
    ....print 'this is not available'
```

- ▶ Unless explicitly silenced

```
try:  
    v = d[k]  
except KeyError:  
    v = d[k] = 'value'
```

Zen of Python

- ▶ In the face of ambiguity, refuse the temptation to guess.

```
# Bad
```

```
if not a and b: ...
```

```
# Good
```

```
if (not a) and b: ...
```

- ▶ There should be one – and preferably only one – obvious way to do it.

```
for item in items: ...
```

- ▶ Although that way may not be obvious at first unless you're Dutch.

Zen of Python

- ▶ Now is better than never.