Python Style Rules

2020.04.30

학부생 연구원 이지헌

배경

- Python은 Google에서 사용되는 주요 동적 언어
- Google Python Style Guide는 파이썬 프로그램에 대한 **할 일**과 **하지 말아야 할 일**의 목록
- http://google.github.io/styleguide/pyguide.html

Semicolons

- 세미콜론을 이용해서 문장을 끝내거나, 한 줄에 2개의 구문 X

Python Shell

```
>>> print('Python is very good !');   ( X )

>>> a = [1, 2, 3]; print(a);   ( X )


>>> print('Python is very good !')   ( O )

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

>>> print(a)   ( O )
```

Line length

- 최대 줄 길이는 80자
- Python의 소/중/대 괄호 내부의 묵시적 라인결합을 사용
- 필요한 경우 구문 양쪽에 추가로 괄호를 더 할 수 있음

Python Shell

```
>>> a = ('This is very long long long '
         'long long long long long string')   /// 괄호를 이용하여 묵시적 라인결합 사용
```

Line length

- 3개 이상의 컨텍스트 매니저를 요구하는 with 구문을 제외하고

  백슬래쉬(\ )를 이용한 문장연장 사용 X

- 3줄 이상이 필요한 with 구문을 정의할 때는 백슬래쉬(\ ) 사용 허용

  2줄인 경우 네스티드 with을 사용

[올바른 사용 예]

```
with very_long_first_expression_function() as spam, \
    very_long_second_expression_function() as beans, \
    third_thing() as eggs:
    place_order(eggs, beans, spam, beans)
```

```
with very_long_first_expression_function() as spam:
    with very_long_second_expression_function() as beans:
        place_order(beans, spam)
```

Line length 예외

- 긴 import 구문
- URL, 경로이름 또는 주석의 긴 플래그
- 공백을 포함하지 않는 긴 모듈수준 문자상수를 여러 줄에 나누어 기록하기 불편할 경우

Parentheses

- 괄호를 적게 사용
- 필요하지는 않지만 튜플의 양쪽에 괄호를 사용하여도 무방
- 하지만 묵시적 문장연장이나 튜플을 나타내기 위한 상황을 제외하고 return문이나 조건문에는 사용 X

```
if (x):
    bar()

if not(x):
    bar()
return (foo)
```

```
if foo:
    bar()

if not x:
    bar()
return foo
```

Indentation

- 코드를 작성할 때 4칸 들여쓰기 사용
- Tab을 사용해도 되지만 Tab과 Space를 섞어서는 사용 X
- 묵시적 문장연장의 경우 동일한 문장에 포함된 요소들을 수직정렬 또는 첫 열린괄호 이후로는 4칸 hanging indent를 적용해야 함

```
# Aligned with opening delimiter
foo = long_function_name(var_one, var_two,
                         var_three, var_four)
```

```
# 4-space hanging indent; nothing on first line
foo = long_function_name(
    var_one, var_two, var_three,
    var_four)
```

Blank Lines

- 함수 또는 객체 선언은 최상위 선언문과는 2개의 빈 줄을 사이에 두어야 함
- 각 메소드 선언 또는 Class Line과 첫 번째 메소드 선언 시 사이에 한 개의 빈 줄이 있어야 함
- def Line 이후에는 빈 줄이 없어야 함
- 함수와 메소드 사이에 판단하에 적절하게 한 개의 빈 줄을 사용

Whitespace

- 표준 조판 규칙을 따라 구두점 주변에 스페이스를 사용
- 괄호, 중괄호, 대괄호 내부에는 화이트스페이스가 없어야 함

Python Shell

```
>>> spam( ham[ 1 ], { eggs: 2 }, [  ] )   ( X )
```

```
>>> spam(ham[1], {eggs: 2}, [ ])   ( O )
```

Whitespace

- 컴마, 세미콜론, 콜론 앞에는 화이트스페이스가 없어야 함
- 문장 끝을 제외하고 컴마, 세미콜론, 콜론 뒤에 화이트스페이스를 사용 X

```
if x == 4 :
    print(x , y)
x , y = y , x
```

```
if x == 4:
    print(x, y)
x, y = y, x
```

Whitespace

- 매개변수 목록, 인덱싱, 슬라이싱의 시작에 사용된 열린 소/중괄호 앞에는 화이트스페이스를 사용 X
- 대입, 비교, 불린과 같은 바이너리 연산자는 앞, 뒤로 한 칸 띄워야 함

Python Shell

>>> spam (1)	>>> spam(1)
>>> dict ['key'] = list [index]	>>> dict['key'] = list[index]
>>> x<1 	>>> x < 1 

Whitespace

- 키워드 매개변수나 기본값을 지정하는 경우 (=) 기호 앞뒤로 공백 사용 X

  (Type 지정이 존재할 때는 사용)

- 공백을 이용하여 연속된 여러 줄을 수직정렬 사용 X ex) : , #, =

```
def complex(real, imag = 0.0): return Magic(r = real, i = imag)


foo           = 1000  # comment
long_name = 2     # comment that should not be aligned

dictionary = {
    'foo'           : 1,
    'long_name': 2,
}
```

```
def complex(real, imag=0.0): return Magic(r=real, i=imag)


foo = 1000  # comment
long_name = 2  # comment that should not be aligned

dictionary = {
    'foo': 1,
    'long_name': 2,
}
```

Shebang Line

- Shebang은 sharp(#) + bang(!) 합성어
- Unix 계열 OS(리눅스, Mac)에서 스크립트(bash, python 등) 코드 최상단에서 해당 파일을 해석해줄 인터프리터의 절대경로를 지정
- 대부분의 .py 파일은 #! 로 시작하지 않아도 됨
- 이 줄은 파이썬 파일을 import 할 때는 무시되지만 실행될 때는 커널이 어떤 파이썬 인터프리터를 사용해야 하는지 알려줌 (기록하는 것이 적합)

감사합니다