System development life-cycle
Style

Petr Svarny, 2020

From just programming to good programming.

Analysis,
Design,
Tests,
Style.

Style!

… because most of the time you will read code.

What is Python coding style

- PEP8
  - With variations as Google
  - Main point is consistency (in the context/group) to improve readability see RealPython, Python-guide
- Automation with:
  - Linters as Pylint
    - Often some linter part of the IDE (don't ignore it)
    - Pylint is configurable
  - Formatters as ~~is~~ Black, AutoPEP8
    - Its style is "opinionated" and not configurable
- We saw already some ideas in PEP20 - Zen of Python

"Explicit is better than implicit."

Examples

1

```
def make_complex(*args):
    x, y = args
    return dict(**locals())
```

2

```
def make_complex(x, y):
    return {'x': x, 'y': y}
```

Examples

1

```
def make_complex(*args):
    x, y = args
    return dict(**locals())
```

2

```
def make_complex(x, y):
    return {'x': x, 'y': y}
```

Examples

1

```
print 'one'
print 'two'

if x == 1:
   print 'one'

cond1 = <complex comparison>
cond2 = <other complex comparison>
if cond1 and cond2:
   # do something
```

2

```
print 'one'; print 'two'

if x == 1: print 'one'

if <complex comparison> and <other
complex comparison>:
   # do something
```

Examples

1

```
print 'one'
print 'two'

if x == 1:
    print 'one'

cond1 = <complex comparison>
cond2 = <other complex comparison>
if cond1 and cond2:
    # do something
```

2

```
print 'one'; print 'two'

if x == 1: print 'one'

if <complex comparison> and <other
complex comparison>:
    # do something
```

Examples

1

```
spam(1)
```

2

```
spam (1)
```

Examples

1

```
spam(1)
```

2

```
spam (1)
```

Examples

1

```
x             = 1
y             = 2
long_variable = 3
```

2

```
x = 1
y = 2
long_variable = 3
```

Examples

1

```
x             = 1
y             = 2
long_variable = 3
```

2

```
x = 1
y = 2
long_variable = 3
```

Examples

1

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

2

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

Examples

1

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

2

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

Examples

1

```
if foo == 'blah': do_blah_thing()
for x in lst: total += x
while t < 10: t = delay()
```

2

```
if foo == 'blah': do_blah_thing()
else: do_non_blah_thing()

try: something()
finally: cleanup()

do_one(); do_two(); do_three(long, argument,
                             list, like, this)

if foo == 'blah': one(); two(); three()
```

Examples

1

```
if foo == 'blah': do_blah_thing()
for x in lst: total += x
while t < 10: t = delay()
```

2

```
if foo == 'blah': do_blah_thing()
else: do_non_blah_thing()

try: something()
finally: cleanup()

do_one(); do_two(); do_three(long, argument,
                             list, like, this)

if foo == 'blah': one(); two(); three()
```

Examples

1

```
FILES = ['setup.cfg', 'tox.ini',]
initialize(FILES, error=True,)
```

2

```
FILES = [
    'setup.cfg',
    'tox.ini',
    ]
initialize(FILES,
           error=True,
           )
```

Examples

1

```
FILES = ['setup.cfg', 'tox.ini',]
initialize(FILES, error=True,)
```

2

```
FILES = [
    'setup.cfg',
    'tox.ini',
    ]
initialize(FILES,
           error=True,
           )
```

Examples

1

```
x = x + 1        # Compensate for border
```

2

```
x = x + 1        # Increment x
```

Examples

1

```
x = x + 1        # Compensate for border
```

2

```
x = x + 1        # Increment x
```

Some other Python habits

- `setup.py` for setting up the program environment
- `requirements.txt` for a `pip install -r requirements` launch