Python static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your PYTHON code

All rules 216 | Vulnerability 29 | Bug 55 | Security Hotspot 31 | Code Smell 101

Tags | Search by name...

Functions should not have too many lines of code
Code Smell

Track uses of "NOSONAR" comments
Code Smell

Track comments matching a regular expression
Code Smell

Statements should be on separate lines
Code Smell

Functions should not contain too many return statements
Code Smell

Files should not have too many lines of code
Code Smell

Lines should not be too long
Code Smell

Methods and properties that don't access instance data should be static
Code Smell

New-style classes should be used
Code Smell

Parentheses should not be used after certain keywords
Code Smell

Track "TODO" and "FIXME" comments that do not contain a reference to a person
Code Smell

Module names should comply with a naming convention
Code Smell

Walrus operator should not make code confusing

Analyze your code

Code Smell | Minor

The walrus operator `:=` (also known as "assignment expression") should be used with caution as it can easily make code more difficult to understand and thus maintain. In such case it is advised to refactor the code and use an assignment statement (i.e. `=`) instead.

This rule raises an issue when the walrus operator is used in a way which makes the code confusing, as described in PEP 572.

Noncompliant Code Example

```
# using an assignment expression (:=) as an assignment state
(v := f(p))  # Noncompliant
v0 = (v1 := f(p))  # Noncompliant

# using an assignment expression in a function call when key
func(a=(b := f(p)))  # Noncompliant
func(a := f(p), b=2)  # Noncompliant
def func(param=(p := 21)):  # Noncompliant
    pass

# using an assignment expression in an annotation
def func(param: (p := 21) = 3):  # Noncompliant
    pass

# using assignment expression in an f-string. Character ":"
f'{(x:=10)}'  # Noncompliant
f'{x:=10}' # No issue raised but still not recommended. This
```

Compliant Solution

```
v = f(p)
v0 = v1 = f(p)

value = f(p)
func(a=value)
func(value, b=2)
def func(param=21):
    p = 21

p = 21
def func(param: p = 3):
    pass

x = 10
f'{x}'
```

See

- PEP 572 - Assignment Expressions

Comments should not be located at the end of lines of code

Code Smell

Lines should not end with trailing whitespaces

Code Smell

Files should contain an empty newline at the end

Code Smell

Long suffix "L" should be upper case

Code Smell

Available In:

sonarlint | sonarcloud | sonarqube