

- Secrets
- ABAP
- Apex
- C
- C++
- CloudFormation
- COBOL
- Code Smell
- CSS
- Flex
- Go
- HTML
- Java
- JavaScript
- Kotlin
- Objective C
- PHP
- PL/I
- PL/SQL
- Python
- RPG
- Ruby
- Scala
- Swift
- Terraform
- Text
- TypeScript
- T-SQL
- VB.NET
- VB6
- XML



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

Unused private nested classes should be removed Analyze your code

Code Smell Major ? unused

"Private" nested classes that are never used inside the enclosing class are usually dead code: unnecessary, inoperative code that should be removed. Cleaning out dead code decreases the size of the maintained codebase, making it easier to understand the program and preventing bugs from being introduced.

Python has no real private classes. Every class is accessible. There are however two conventions indicating that a class is not meant to be "public":

- classes with a name starting with a single underscore (ex: `_MyClass`) should be seen as non-public and might change without prior notice. They should not be used by third-party libraries or software. It is ok to use those classes inside the library defining them but it should be done with caution.
- "class-private" classes are defined inside another class, and have a name starting with at least two underscores and ending with at most one underscore. These classes' names will be automatically mangled to avoid collision with subclasses' nested classes. For example `_MyClass` will be renamed as `_classname_MyClass`, where `classname` is the enclosing class's name without its leading underscore(s). Class-Private classes shouldn't be used outside of their enclosing class.

This rule raises an issue when a private nested class (either with one or two leading underscores) is never used inside its parent class.

Noncompliant Code Example

```
class Noncompliant:
    class __MyClass1(): # Noncompliant
        pass






    class _MyClass2(): # Noncompliant
        pass
```

Compliant Solution

```
class Compliant:
    class __MyClass1():
        pass

    class _MyClass2():
        pass

    def process(self):
        return Compliant.__MyClass1()
```



 Code Smell
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

```
def process(self):  
    return Compliant._MyClass2()
```

See

- [Python documentation – Private Variables](#)
- [PEP 8 – Style Guide for Python Code](#)

Available In:

 |  | 