

- Secrets
- ABAP
- Apex
- C
- C++
- CloudFormation
- COBOL
- C#
- 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...



Code Smell

Unused function parameters should be removed

Code Smell

Unused class-private methods should be removed

Code Smell

Track uses of "FIXME" tags

Code Smell

"Exception" and "BaseException" should not be raised

Code Smell

Redundant pairs of parentheses should be removed

Code Smell

Nested blocks of code should not be left empty

Code Smell

Functions, methods and lambdas should not have too many parameters

Code Smell

Collapsible "if" statements should be merged

Code Smell

Logging should not be vulnerable to injection attacks

Vulnerability

Repeated patterns in regular expressions should not match the empty string

Bug

Function parameters initial values should not be ignored

The first argument to class methods should follow the naming convention

Analyze your code

Code Smell Critical convention confusing pitfall

By convention, the first argument to class methods, i.e. methods decorated with `@classmethod`, is named `cls` as a representation and a reminder that the argument is the class itself. Name the argument something else, and you stand a good chance of confusing both users and maintainers of the code. It might also indicate that the `cls` parameter was forgotten, in which case calling the method will most probably fail. This rule also applies to methods `__init_subclass__`, `__class_getitem__` and `__new__` as their first argument is always the class instead of "self".

By default this rule accepts `cls` and `mcs`, which is sometime used in metaclasses, as valid names for class parameters. You can set your own list of accepted names via the parameter `classParameterNames`.

This rule raises an issue when the first parameter of a class method is not an accepted name.

Noncompliant Code Example

```
class Rectangle(object):  
  
    @classmethod  
    def area(bob, height, width): #Noncompliant  
        return height * width
```

Compliant Solution

```
class Rectangle(object):  
  
    @classmethod  
    def area(cls, height, width):  
        return height * width
```

See

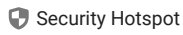
- PEP8 - [Function and Method Arguments](#)

Available In:

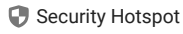
sonarlint | sonarcloud | sonarqube



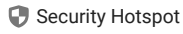
Disabling versioning of S3 buckets is security-sensitive



Disabling server-side encryption of S3 buckets is security-sensitive



Having a permissive Cross-Origin Resource Sharing policy is security-sensitive



Delivering code in production with debug features activated is security-sensitive