

- 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...



Character classes should be preferred over reluctant quantifiers in regular expressions

Code Smell

Character classes should be preferred over reluctant quantifiers in regular expressions

Code Smell

A subclass should not be in the same "except" statement as a parent class

Code Smell

Walrus operator should not make code confusing

Code Smell

Jump statements should not be redundant

Code Smell

"pass" should not be used needlessly

Code Smell

"except" clauses should do more than raise the same issue

Code Smell

Boolean checks should not be inverted

Code Smell

Unused local variables should be removed

Code Smell

Local variable and function parameter names should comply with a naming convention

Code Smell

Field names should comply with a naming convention

Code Smell

### Alternatives in regular expressions should be grouped when used with anchors

Analyze your code

Bug Major ? regex

In regular expressions, anchors (`^`, `$`, `\A`, `\Z` and `\z`) have higher precedence than the `|` operator. So in a regular expression like `^alt1|alt2|alt3$`, `alt1` would be anchored to the beginning, `alt3` to the end and `alt2` wouldn't be anchored at all. Usually the intended behavior is that all alternatives are anchored at both ends. To achieve this, a non-capturing group should be used around the alternatives.

In cases where it is intended that the anchors only apply to one alternative each, adding (non-capturing) groups around the anchors and the parts that they apply to will make it explicit which parts are anchored and avoid readers misunderstanding the precedence or changing it because they mistakenly assume the precedence was not intended.

#### Noncompliant Code Example

```
r"^a|b|c$"
```

#### Compliant Solution

```
r"^(?:a|b|c)$"
```

or

```
r"^a$|^b$|^c$"
```


or, if you do want the anchors to only apply to a and c respectively:

```
r"^(?:^a)|b|(?^c$)"
```


Available In:

sonarlint | sonarcloud | sonarqube


Class names should comply with a naming convention

 Code Smell


Method names should comply with a naming convention

 Code Smell

Track uses of "TODO" tags

 Code Smell

HTML autoescape mechanism should not be globally disabled

 Vulnerability

Variables, classes and functions should be either defined or imported

