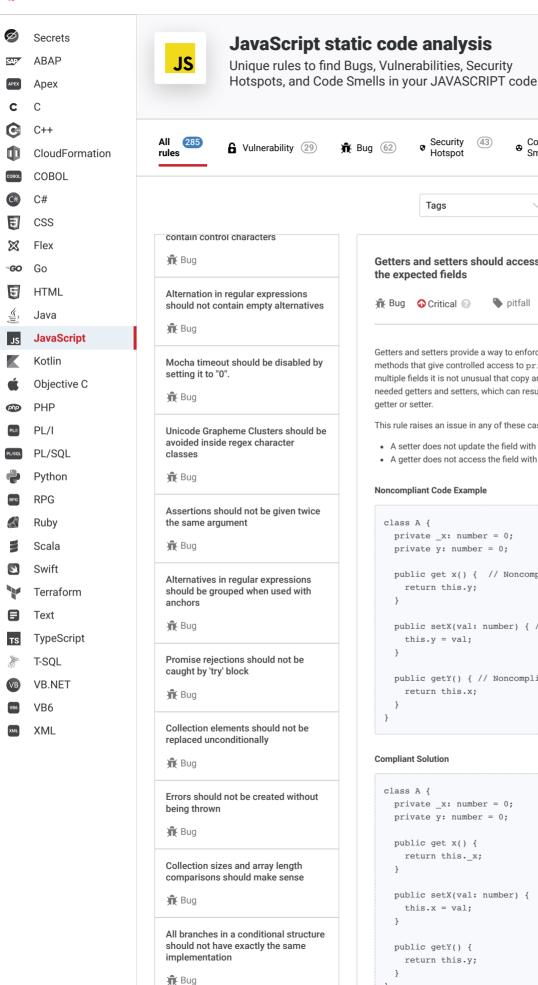


Products >





Security

Hotspot

Tags

Rug Oritical @

(43)

Analyze your code

Search by name.

O Quick 41 Fix

Q

Getters and setters provide a way to enforce encapsulation by providing public methods that give controlled access to private fields. However in classes with multiple fields it is not unusual that copy and paste is used to quickly create the needed getters and setters, which can result in the wrong field being accessed by a getter or setter.

This rule raises an issue in any of these cases:

• A setter does not update the field with the corresponding name.

pitfall

• A getter does not access the field with the corresponding name.

## **Noncompliant Code Example**

```
class A {
  private _x: number = 0;
  private y: number = 0;
  public get x() { // Noncompliant: field 'x' is not used i
    return this.y;
  public setX(val: number) { // Noncompliant: field 'x' is n
    this.y = val;
  public getY() { // Noncompliant: field 'y' is not used in
    return this.x:
}
```

## Compliant Solution

```
class A {
  private _x: number = 0;
  private y: number = 0;
  public get x() {
    return this._x;
  public setX(val: number) {
    this.x = val;
  public getY() {
   return this.y;
}
```

Destructuring patterns should not be



Available In:
sonarlint ⊖ | sonarcloud ↔ | sonarqube

© 2008-2022 SonarSource S.A., Switzerland. All content is copyright protected. SONAR, SONARSOURCE, SONARLINT, SONARQUBE and SONARCLOUD are trademarks of SonarSource S.A. All other trademarks and copyrights are the property of their respective owners. All rights are expressly reserved.

Privacy Policy