FB.UUF\_UNUSED\_PUBLIC\_OR\_PROTECTED\_FIELD

Детектор определяет ошибку, в ходе которой поле никогда не используется. Поле является общедоступным или защищенным, поэтому, возможно, оно предназначено для использования с классами, не рассматриваемыми как часть анализа. Если нет, рассмотрите возможность удаления его из класса.

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | MAJOR |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Ошибка разработчика |
| CWE | — |
| CVE | — |
| Возможные последствия | Если поле объявлено как public или protected, но оно никогда не используется, это может привести к увеличению сложности кода, создавая ложное представление о его важности. |
| Комментарий | Если поле не используется, его следует удалить, чтобы снизить сложность кода. Если предполагается его использование в других частях программы, которые не были проанализированы, подумайте о том, чтобы сделать его private и предоставить методы доступа. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public int unusedPublicField;  protected String unusedProtectedField;  public static void incorrectTest() {  var example = new UUF\_UNUSED\_PUBLIC\_OR\_PROTECTED\_FIELD();  example.unusedPublicField = 100; // Поле инициализируется, но не используется  example.unusedProtectedField = "Test";  } | public int usedPublicField;  protected String usedProtectedField;  public static void correctTest() {  var example = new UUF\_UNUSED\_PUBLIC\_OR\_PROTECTED\_FIELD();  example.usedPublicField = 100;  example.usedProtectedField = "Test";  System.out.println(example.usedPublicField);  System.out.println(example.usedProtectedField);  } |

Методы устранения уязвимости при срабатывании детектора:

1. Удалите поле, если вы уверены, что оно не используется нигде за пределами текущего анализа кода. Это самый простой способ упростить код.
2. Сделайте поле private и предоставьте методы доступа (геттеры и сеттеры), если предполагается его использование в других частях программы. Это инкапсулирует состояние и делает API класса более контролируемым.
3. Проверьте, не было ли поле предназначено для использования в будущем. Если поле планируется использовать, оставьте его, но добавьте комментарий с объяснением его назначения.
4. Убедитесь, что нет опечаток в имени поля при его предполагаемом использовании в других частях кодовой базы. Поиск по всему проекту может помочь выявить такие случаи.
5. Рассмотрите возможность удаления кода, который присваивает значение полю, если поле нигде не читается. Это также указывает на избыточность.
6. Если поле используется только для константных значений, рассмотрите возможность объявления его как public static final (или protected static final) и использования напрямую.
7. В случае сериализации, даже если поле не читается явно в коде, оно может использоваться при восстановлении объекта. Убедитесь, что поле действительно не нужно для корректной десериализации.