FB.UWF\_NULL\_FIELD

Детектор определяет ошибку, в ходе которой все записи в данное поле имеют постоянное значение null, поэтому все операции чтения поля возвращают значение null. Проверьте наличие ошибок или удалите его, если он бесполезен.

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | MAJOR |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Ошибка разработчика |
| CWE | — |
| CVE | — |
| Возможные последствия | Поле всегда содержит null, что может привести к NullPointerException при его использовании или указывать на избыточность поля, занимающего память без необходимости. |
| Комментарий | Следует проверить, должно ли поле инициализироваться значением, или оно является лишним и может быть удалено. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public static void incorrectTest() {  class Test {  private String value;  public Test() {  // value не инициализируется  }  }  } | public static void correctTest() {  class Test {  private String value;  public Test(String value) {  this.value = value;  }  }  } |

Методы устранения уязвимости при срабатывании детектора:

1. Проверьте, должно ли поле инициализироваться ненулевым значением. Если поле должно содержать какое-то состояние объекта, найдите место, где ему должно присваиваться это значение (например, в конструкторе или сеттере).
2. Если поле инициализируется в конструкторе, убедитесь, что присваивание действительно происходит. Возможно, есть опечатка в имени поля или параметра, или логика инициализации не выполняется.
3. Если поле должно инициализироваться лениво (при первом использовании), убедитесь, что логика ленивой инициализации реализована корректно. Проверьте, что присваивание значения происходит при первом обращении к полю.
4. Если поле используется только для хранения временного значения, рассмотрите возможность использования локальной переменной вместо поля класса. Это может упростить код и избежать потенциальных NullPointerException.
5. Если поле никогда не используется (не читается), удалите его. Наличие неиспользуемых полей засоряет код и может вводить в заблуждение.
6. Если поле предназначено для использования только в определенных сценариях (например, при определенных настройках), убедитесь, что его инициализация происходит в этих сценариях. В противном случае, рассмотрите возможность использования Optional для представления потенциально отсутствующего значения.
7. Проверьте, не является ли присваивание null ошибкой. Возможно, в коде есть место, где полю случайно присваивается null, хотя это не intended behavior.