FB.UC\_USELESS\_VOID\_METHOD

Наш анализ показывает, что этот метод непустой пустоты на самом деле не выполняет никакой полезной работы. Следует его проверить: возможно, в его коде ошибка или его тело можно полностью удалить.

Мы стараемся максимально сократить количество ложных срабатываний, но в некоторых случаях это предупреждение может быть ошибочным. К частым ложноположительным случаям относятся:

1) Этот метод предназначен для запуска загрузки некоторого класса, что может иметь побочный эффект.

2) Этот метод предназначен для неявной генерации какого-то непонятного исключения.

Таблица №1 – Основные характеристики детектора FB.UC\_USELESS\_VOID\_METHOD

|  |  |
| --- | --- |
| Категория критичности | Minor |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Code Quality Issue |
| CWE | --- |
| CVE | --- |
| Возможные последствия | * Загромождение кода бесполезными методами * Введение в заблуждение разработчиков * Усложнение рефакторинга |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Потенциально небезопасное | Корректная конструкция |
| public static void incorrectTest() {  class DataProcessor {  // Некорректно: бесполезный void-метод  public void processData(String data) {  // Ничего не делает  }  }    DataProcessor processor = new DataProcessor();  processor.processData("test"); // Бесполезный вызов  } | public static void correctTest() {  class DataProcessor {  // Корректно: метод либо удален, либо делает полезную работу  public String processData(String data) {  return data != null ? data.toUpperCase() : "";  }  }    DataProcessor processor = new DataProcessor();  String result = processor.processData("test");  System.out.println("Processed: " + result);  } |