FB.UC\_USELESS\_OBJECT

Анализ показывает, что этот объект бесполезен. Он создается и модифицируется, но его значение никогда не выходит за пределы метода и не приводит к какому-либо побочному эффекту. Либо произошла ошибка и объект предназначен для использования, либо его можно удалить.

Данный анализ редко дает ложноположительные результаты. К частым ложноположительным случаям относятся: объект используется для неявной генерации какого-то непонятного исключения, объект используется как заглушка для обобщения кода или объект использовался для хранения сильных ссылок на объекты со слабыми/мягкими ссылками.

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

|  |  |
| --- | --- |
| Категория критичности | Minor |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Code Quality Issue |
| CWE | --- |
| CVE | --- |
| Возможные последствия | * Создание ненужных объектов * Нагрузка на сборщик мусора * Снижение производительности |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Потенциально небезопасное | Корректная конструкция |
| public static void incorrectTest() {  // Некорректно: создание ненужного объекта  String result = new String("constant"); // Избыточный конструктор  System.out.println(result);    // Другой пример  Integer sum = new Integer(0); // Ненужная обертка  for (int i = 0; i < 10; i++) {  sum += i;  }  System.out.println("Sum: " + sum);  } | public static void correctTest() {  // Корректно: только необходимые проверки  String text = "example";  System.out.println("Direct use: " + text.length());    // Проверка нужна только когда тип неочевиден  Object obj = "test";  if (obj instanceof String) { // Оправданная проверка  String str = (String) obj;  System.out.println("Valid length: " + str.length());  }    // Для nullable объектов  Integer number = null;  if (number != null) { // Достаточно одной проверки  System.out.println("Number: " + number);  }  } |