FB.CD\_CIRCULAR\_DEPENDENCY

Этот класс имеет циклическую зависимость от других классов. Это затрудняет создание этих классов, поскольку правильность построения каждого зависит от другого. Стоит рассмотреть возможность использования интерфейсов, чтобы разорвать жесткую зависимость.

Таблица №1 – Основные характеристики детектора FB.CD\_CIRCULAR\_DEPENDENCY

|  |  |
| --- | --- |
| Категория критичности | Minor |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Design Flaw |
| CWE | --- |
| CVE | --- |
| Возможные последствия | * Трудности в тестировании * Проблемы с компиляцией * Усложнение рефакторинга |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Потенциально небезопасное | Корректная конструкция |
| // Некорректно: циклическая зависимость  static class ClassA {  private ClassB b;    public ClassA() {  this.b = new ClassB(this);  }  }  static class ClassB {  private ClassA a;    public ClassB(ClassA a) {  this.a = a;  }  }  public static void incorrectTest() {  ClassA a = new ClassA();  System.out.println("Created circular dependency (A->B->A)");  } | interface ICommonFunctionality {  void doSomething();  }  static class ProperClassA implements ICommonFunctionality {  private ProperClassB b;    public ProperClassA() {  this.b = new ProperClassB();  }    @Override  public void doSomething() {  System.out.println("ClassA functionality");  }  }  static class ProperClassB {  private ICommonFunctionality a;    public ProperClassB() {  // Зависимость через интерфейс  }    public void setDependency(ICommonFunctionality a) {  this.a = a;  }  }  public static void correctTest() {  ProperClassA a = new ProperClassA();  ProperClassB b = new ProperClassB();  b.setDependency(a);    System.out.println("Created proper dependency structure");  } |