เอกสารประกอบชั่วโมงปฏิบัติการ

วิชา CP353201 Software Quality Assurance **ปีการศึกษา 1/2568**

ผศ. ดร. ชิตสุธา สุ่มเล็ก **สาขาวิชาวิทยาการคอมพิวเตอร์ วิทยาลัยการคอมพิวเตอร์ มข.**

ชื่อ-นามสกุล นายศุภวิชญ์ ศักดิ์เทวินทร์ รหัสนศ.663380239-8 Section 1

Lab#12 – Analyze the quality of your software by using software metrics

วัตถุประสงค์การเรียนรู้

1. ผู้เรียนสามารถอธิบายความหมายของ Software metrics ที่วัดออกมาได้อย่างถูกต้อง
2. ผู้เรียนสามารถใช้ Static code analysis tools ในการวิเคราะห์ปัญหาคุณภาพของซอฟต์แวร์ได้
3. ผู้เรียนสามารถอธิบายความแตกต่างระหว่าง Static testing และ Dynamic testing ได้อย่างถูกต้อง

กิจกรรมที่ 12.1: การวิเคราะห์ปัญหาในโค้ดด้วย PMD

(1) ติดตั้ง PMD plugin ลงใน Eclipse ให้เรียบร้อย

(2) สร้าง local repository ของ git และตั้งค่าให้เรียบร้อย Workspace ให้เรียบร้อย

(3) Clone โค้ดตั้งต้นจาก
https://github.com/ChitsuthaCSKKU/SQA/tree/2025/Assignment/Lab12_SoftwareMetrics

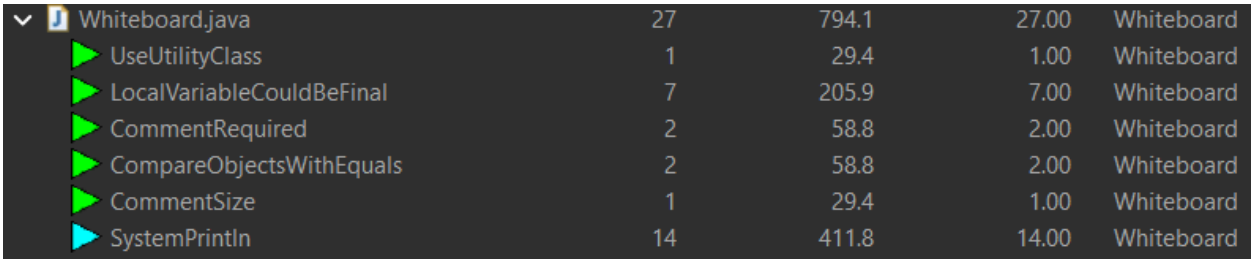
(4) ศึกษาการทำงานและโครงสร้างของ Source code ที่กำหนดให้

(5) วิเคราะห์ปัญหาใน Source code โดยใช้ PMD

(6) Capture หน้าจอ หรือ Export รายงานผลการทดสอบ แล้วตอบคำถามต่อไปนี้

Q1: PMD พบ Error และ Warning ใน Whitboard.java กี่ตัว (Capture หน้าจอประกอบคำตอบ)

ตอบ พบ 27 ตัว



Whiteboard.java	27	794.1	27.00	Whiteboard
UseUtilityClass	1	29.4	1.00	Whiteboard
LocalVariableCouldBeFinal	7	205.9	7.00	Whiteboard
CommentRequired	2	58.8	2.00	Whiteboard
CompareObjectsWithEquals	2	58.8	2.00	Whiteboard
CommentSize	1	29.4	1.00	Whiteboard
SystemPrintln	14	411.8	14.00	Whiteboard

Q2: จงอธิบายเกี่ยวกับปัญหา “Local variable 'managerB' could be declared final” ว่าเป็นปัญหาอะไร และมีแนวทางในการแก้ไขอย่างไร

ตอบ เป็นปัญหาที่เกิดจากตัวแปรภายในเมธอดที่ชื่อว่า manager ถูกกำหนดค่าเพียงครั้งเดียว และไม่เคยถูกเปลี่ยนแปลงอีกตลอดการทำงาน PMD จึงแนะนำว่าควรประกาศเป็น final

เอกสารประกอบชั่วโมงปฏิบัติการ

วิชา CP353201 Software Quality Assurance **ปีการศึกษา 1/2568**

ผศ. ดร. ชิตสุธา สุ่มเล็ก **สาขาวิชาวิทยาการคอมพิวเตอร์ วิทยาลัยการคอมพิวเตอร์ มข.**

ชื่อ-นามสกุล นายศุภวิชญ์ ศักดิ์เทวินทร์ รหัสนศ.663380239-8 Section 1

Q3: PMD ใช้ Metrics อะไรบ้างในการระบุ God class และแต่ละตัวมีความหมายว่าอะไรบ้าง

ตอบ

WMC (*Weighted Methods per Class*): จะนับว่าคลาสนี้มี เมธอดเยอะและซับซ้อนแค่ไหน ถ้ามีเมธอดจำนวนมากหรือแต่ละเมธอดซับซ้อน แปลว่าคลาสนั้นทำงานเยอะเกินไป

ATFD (Access To Foreign Data): จะนับว่าคลาสนี้ ไปดึงหรือใช้ข้อมูลของคลาสอื่นกี่ครั้ง ถ้าสูงแปลว่าคลาสนี้ไปยุ่งกับข้อมูลของคลาสอื่นมากเกิน ควรถูกแยกหน้าที่

TCC (Tight Class Cohesion): จะวัดว่า เมธอดในคลาสใช้ข้อมูลร่วมกันมากน้อยแค่ไหน ถ้าค่านี้ต่ำแปลว่าเมธอดแต่ละอันในคลาสแทบไม่เกี่ยวข้องกันเลย คลาสนี้ควรแยกออกเป็นหลายคลาส

กิจกรรมที่ 12.2: การวิเคราะห์ปัญหาในโค้ดด้วย Checkstyle

(1) ติดตั้ง checkstyle plugin ลงใน Eclipse ให้เรียบร้อย

(2) สร้าง local repository ของ git และตั้งค่าให้เรียบร้อย Workspace ให้เรียบร้อย

(3) Clone โค้ดตั้งต้นจาก https://github.com/ChitsuthaCSKKU/SQA/tree/2025/Assignment/Lab12_SoftwareMetrics

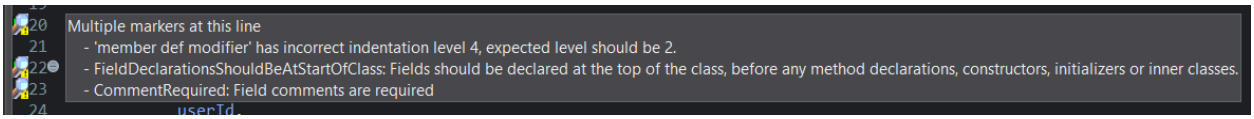
(4) ศึกษาการทำงานและโครงสร้างของ Source code ที่กำหนดให้

(5) วิเคราะห์ปัญหาใน Source code โดยใช้ checkstyle

(6) Capture หน้าจอ หรือ Export รายงานผลการทดสอบ แล้วตอบคำถามต่อไปนี้

Q4: จงระบุปัญหาการเขียน Code ที่ไม่ตรงกับ Coding standard ของ Java ที่ Checkstyle ระบุมา 3 รายการ พร้อมคำอธิบายว่า Code ส่วนนี้ไม่สอดคล้องกับมาตรฐานอย่างไร (Capture หน้าจอประกอบด้วย)

ตอบ



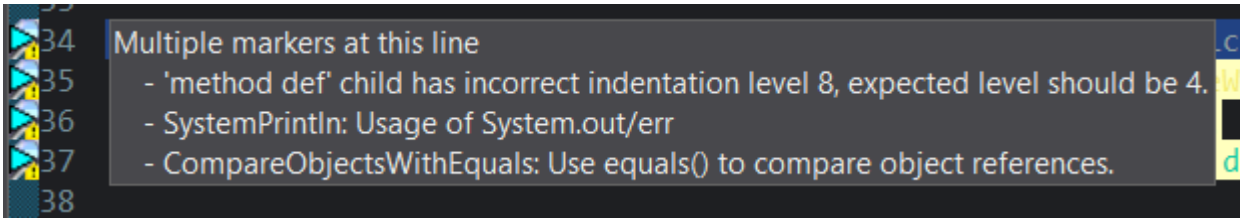
ปัญหา: FieldDeclarationsShouldBeAtStartOfClass ตามมาตรฐาน Java field ควรถูกประกาศไว้ ส่วนบนของคลาส ก่อนที่จะมี constructor, method หรือ inner class เพื่อให้โค้ดมีโครงสร้างที่ชัดเจนและค้นหา field ได้ง่าย และไม่สอดคล้องกับมาตรฐานด้านโครงสร้างคลาส (Class Structure / Field Declaration Order)

เอกสารประกอบชั่วโมงปฏิบัติการ

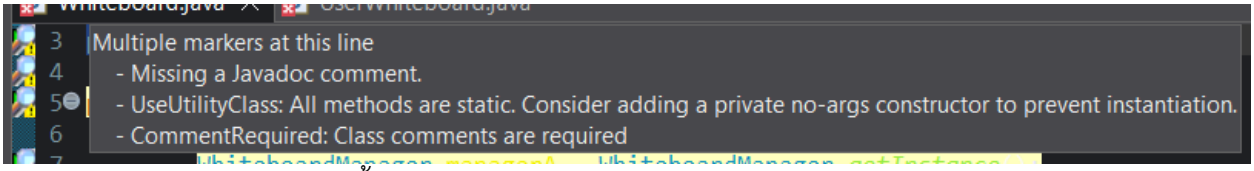
วิชา CP353201 Software Quality Assurance **ปีการศึกษา 1/2568**

ผศ. ดร. ชิตสุธา สุ่มเล็ก **สาขาวิชาวิทยาการคอมพิวเตอร์ วิทยาลัยการคอมพิวเตอร์ มข.**

ชื่อ-นามสกุล นายศุภวิชญ์ ศักดิ์เทวินทร์ รหัสนศ.663380239-8 Section 1



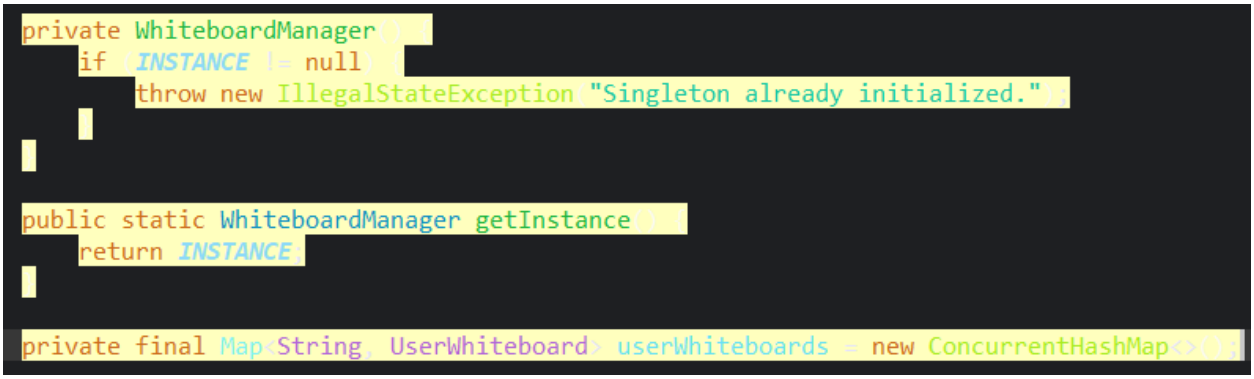
ปัญหา: CompareObjectsWithEquals ไม่ควรใช้ตัวดำเนินการ == เพื่อเปรียบเทียบค่าของ object ตามมาตรฐาน Java ให้ใช้ .equals() และไม่สอดคล้องกับมาตรฐานด้านความถูกต้องของโค้ด (Code Correctness / Object Comparison)



ปัญหา: UseUtilityClass คลาสนี้มีแต่วิธีการ (method) แบบ static จึงควรถูกใช้เป็น utility class คือ แนะนำให้เพิ่ม private no-args constructor และไม่สอดคล้องกับมาตรฐานด้านการออกแบบคลาส (Class Design / Utility Class)

Q5: จาก Q4 หากต้องการแก้ไข Source code เพื่อให้ผ่านมาตรฐานการเขียนโค้ดและการตรวจสอบโดย Checkstyle จะต้องแก้ไข Code ในบรรทัดนั้น ๆ อย่างไร

<u>ตอบ</u>



“private final Map<String, UserWhiteboard> userWhiteboards = new ConcurrentHashMap<>();” ควรถูกประกาศไว้ ส่วนบนของคลาส ก่อนที่จะมี constructor, method หรือ inner class

เอกสารประกอบชั่วโมงปฏิบัติการ

วิชา CP353201 Software Quality Assurance **ปีการศึกษา 1/2568**

ผศ. ดร. ชิตสุธา สุ่มเล็ก **สาขาวิชาวิทยาการคอมพิวเตอร์ วิทยาลัยการคอมพิวเตอร์ มข.**

ชื่อ-นามสกุล นายศุภวิชญ์ ศักดิ์เทวินทร์ รหัสนศ.663380239-8 Section 1

```
System.out.println("Alice Wb 1 == Alice Wb 2? " + (aliceWb1 == aliceWb2));
```

ควรแก้เป็น “System.out.println("Alice Wb 1 equals Alice Wb 2? " + aliceWb1.equals(aliceWb2));”

```
3  public class Whiteboard {
4
5  public static void main(String[] args) {
6
7          WhiteboardManager managerA = WhiteboardManager.getInstance();
8          WhiteboardManager managerB = WhiteboardManager.getInstance();
9
10         System.out.println("--- Manager Verification ---");
11         System.out.println("Manager A Hash: " + System.identityHashCode(managerA));
12         System.out.println("Manager B Hash: " + System.identityHashCode(managerB));
13         //System.out.println("Are both managers the same instance? " + (managerA == managerB));
14         System.out.println("----------------------------\n");
15
16
17         String userId1 = "alice@example.com";
18         String userId2 = "bob@example.com";
19
20         System.out.println("--- Whiteboard Generation ---");
21
22         UserWhiteboard aliceWb1 = managerA.getWhiteboardForUser(userId1);
23         aliceWb1.addContent("Alice starts drawing a square.");
24
25         UserWhiteboard bobWb = managerA.getWhiteboardForUser(userId2);
26         bobWb.addContent("Bob writes his name.");
27
28         UserWhiteboard aliceWb2 = managerB.getWhiteboardForUser(userId1);
```

แนะนำให้เพิ่ม private no-args constructor เพื่อป้องกันไม่ให้มีการสร้าง instance ของคลาสโดยไม่จำเป็น ก็คือ

“public final class Whiteboard {

```
    private Whiteboard() {}
    public static void main(String[] args) {
        // ...
    }
```

}