ใบงานการทดลองที่ 10
เรื่อง การควบคุมเวอร์ชันการทำงานผ่านโปรแกรม Eclipse

1. จุดประสงค์ทั่วไป

1.1. รู้และเข้าใจการติดต่อกับผู้ใช้งาน และการหลายงานพร้อมกัน

1.2. รู้และเข้าใจการติดต่อระหว่างงาน

2. เครื่องมือและอุปกรณ์

เครื่องคอมพิวเตอร์ 1 เครื่อง ที่ติดตั้งโปรแกรม Eclipse

3. ทฤษฎีการทดลอง

3.1. Version Control System (VCS) คืออะไร? มีประโยชน์อย่างไร?

คือ คือระบบซอร์ฟแวร์ที่จะคอยบันทึกเวอร์ชั่นการเปลี่ยนแปลงของโค๊ดหรือเอกสารต่างๆ
โดยจะทำการบันทึกไว้ด้วยว่าการเปลี่ยนแปลงแต่ละครั้งนั้นทำเพื่ออะไร และทำโดยใคร ประโยชน์
ช่วยให้สามารถย้อนไฟล์นางไฟล์หรือแม้กระทั่งทั้งโปรเจคกลับไปเป็นเวอร์ชั่นเก่าได้ นอกจากนั้นระบบ VCS
ยังจะช่วยให้เปรียบเทียบการแก้ไขที่เกิดขึ้นในอดีต ดูว่าใครเป็นคนแก้ไขคนสุดท้ายที่อาจทำให้เกิดปัญหา แก้ไขเมื่อไร

3.2. Git ต่างกับ Github อย่างไร?

Git เป็นระบบที่ช่วยจัดการการแก้ไขใน Repository ส่วน GitHub เป็นบริการจัดเก็บ Repository ออนไลน์พร้อมกับฟีเจอร์อำนวยความสะดวกต่าง ๆ
ที่ให้เราไปทำงานร่วมกันคนอื่นได้

3.3. Repository คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

Repositoryคือตำแหน่งที่เก็บข้อมูลส่วนกลางสำหรับระบบควบคุมเวอร์ชันที่ติดตามการเปลี่ยนแปลงทั้งหมดในซอร์สโค้ดของโครงการและไฟล์ที่เกี่ยวข้อง
เมื่อเวลาผ่านไป เปิดใช้งานการทำงานร่วมกันระหว่างนักพัฒนาหลายคน การกำหนดเวอร์ชันของโค้ด และการตรวจสอบโค้ด
ตัวอย่างระบบควบคุมเวอร์ชันที่ใช้ที่เก็บ ได้แก่ Git, SVN และ CVS ในโปรแกรม Eclipse สามารถสร้างที่เก็บได้โดยใช้เมนู "Team"
และเลือกระบบควบคุมเวอร์ชันที่เหมาะสม

3.4. Clone คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

การโคลนเป็นกระบวนการสร้างสำเนาในเครื่องของที่เก็บระยะไกลในระบบควบคุมเวอร์ชัน ช่วยให้นักพัฒนาสามารถทำงานบน codebase
ได้อย่างอิสระโดยไม่ส่งผลกระทบต่อพื้นที่เก็บข้อมูลดั้งเดิม การโคลนมีประโยชน์สำหรับการทำงานร่วมกัน งานออฟไลน์
และวัตถุประสงค์ในการกำหนดเวอร์ชัน ใน Eclipse การโคลนสามารถทำได้โดยใช้ระบบควบคุมเวอร์ชันต่างๆ เช่น Git, SVN หรือ CVS
และสร้างสำเนาที่สมบูรณ์ของโค้ดเบส รวมถึงประวัติเวอร์ชันทั้งหมด

3.5. Commit คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

การcommitการเปลี่ยนแปลงในระบบควบคุมเวอร์ชันหมายถึงการบันทึกการเปลี่ยนแปลงที่ทำกับสำเนาโลคัลของที่เก็บและส่งไปยังที่เก็บระยะไกลเพื่อรวม
ไว้ในโค้ดเบส การยืนยันการเปลี่ยนแปลงช่วยให้นักพัฒนาสามารถติดตามการเปลี่ยนแปลงของโค้ดเบสในช่วงเวลาหนึ่ง ทำงานร่วมกับผู้อื่น
และรับประกันคุณภาพของโค้ด ใน Eclipse การยอมรับการเปลี่ยนแปลงสามารถทำได้โดยใช้ระบบควบคุมเวอร์ชันต่างๆ เช่น Git, SVN หรือ CVS
ตัวอย่างของเหตุผลที่ยอมรับการเปลี่ยนแปลงอาจมีประโยชน์ ได้แก่ การกำหนดเวอร์ชัน การทำงานร่วมกัน และการตรวจสอบโค้ด

3.6. Staged และ Unstaged คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

Staged และ Unstaged ช่วยให้นักพัฒนาสามารถเลือกการเปลี่ยนแปลงที่จะรวมไว้ในการกระทำครั้งต่อไป เมื่อทำการเปลี่ยนแปลงกับไฟล์ ในตอนแรกจะถือว่า unstaged จนกว่าจะมีการเพิ่มลงในพื้นที่จัดเตรียม หลังจากนั้นจึงถือว่า Staging การเปลี่ยนแปลงการจัดเตรียมอนุญาตให้มีการเลือก commit การทบทวนโค้ด และการเลิกทำการเปลี่ยนแปลงขั้นตอนที่ผิดพลาด ในโปรแกรม Eclipse การเปลี่ยนแปลง staging และ unstaging สามารถทำได้โดยใช้ระบบควบคุมเวอร์ชัน เช่น Git, SVN หรือ CVS

3.7. Push คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

Push คือการนำโค้ดหรือไฟล์เข้าตัวระบบ Git Repository

3.8. Pull คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

คือ เวลา Sync จาก Remote เพื่อดึงข้อมูล Commit ใหม่ๆลงมาเก็บไว้ในเครื่องจะเรียกขั้นตอนนี้ว่า Pull

3.9. Fetch คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

อยากเช็คสถานะของ Remote เฉยๆว่ามีใคร Push ข้อมูลใหม่ขึ้นไปที่ Remote หรือป่ าว เราเรียกวิธีนี้ว่า Fetch

3.10. Conflict ใน VSC คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

คือ การเกิดปัญหาการชนของข้อมูลในไฟล์งานที่ทำ ร่วมกันกับเพื่อนเรา ซึ่งในช่วงที่เราพัฒนาโปรแกรมหรือขียนโค้ดกับเพื่อนร่วมงาน อยู่นั้นเราไม่สามารถรู้ได้เลยว่าเพื่อนเราจะเขียนโค้ดไปในรูปแบบไหน

3.11. Merge Commit คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

คือการที่มีการแตก branch ออกไป develop แยกกัน โดยที่มีการแก้ไขไฟล์เดียวกันซึ่งโค้ดนั้นอาจมีการทับซ้อน หรืออยู่บรรทัดเดียวกัน เมื่อใครคนใดคนหนึ่งนำ โค้ดมา Merge รวมกันนั้นจะเกิดสิ่งที่เรียกว่า Conflict คือโค้ดของทั้งสองคนมีความขัดแย้งกัน

3.12. ขั้นตอนที่อยู่ในระหว่าง Development Process ภายใน VSC มีอะไรบ้าง?

3.13. จงบอกและอธิบายขั้นตอนการติดตั้งส่วนขยายใน Eclipse เพื่อให้ใช้งาน Git

1.Install Plugin ทำการ Click ไปที่ Help และ Install new software

2.จากนั้นก็พิมพ์ http://download.eclipse.org/egit/updates ลงในช่อง URLแล้วติ๊กที่ Egit

4. ลำดับขั้นการปฏิบัติการ

4.1. ลงทะเบียน Github และตกแต่ง Profile ของตนเองให้เรียบร้อย

4.2. สร้าง Repository ใน Github

4.3. ทำการติดตั้งส่วนเสริมของ Git ลงใน Eclipse เพื่อเตรียมใช้งาน Version Control System ของ Github

4.4. การสร้างผลงานโค้ดโปรแกรมใน Github

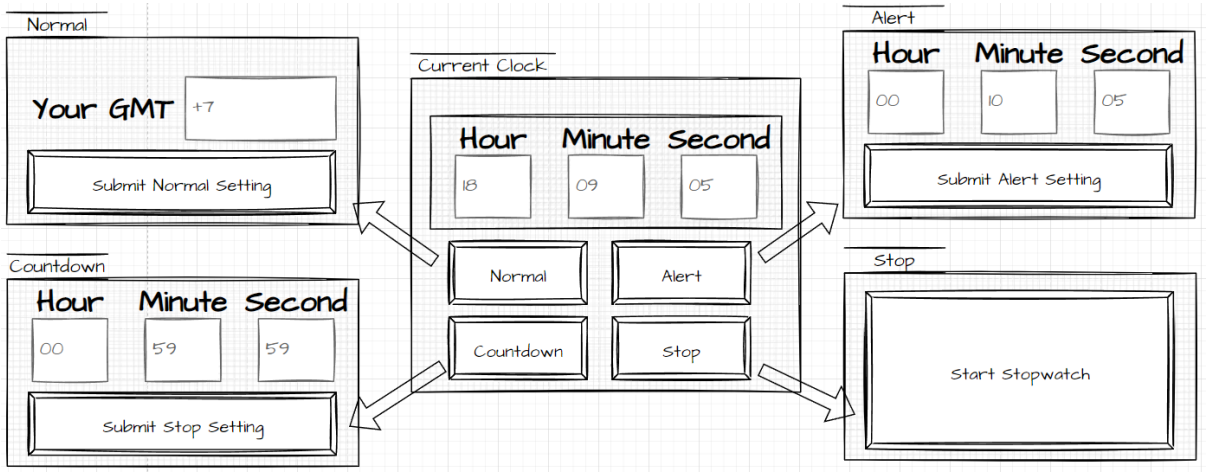
4.4.1. เชื่อมต่อ Eclipse ของคุณเข้ากับ Github

4.4.2. ทำการ Push โค้ดโปรแกรมตั้งแต่การทดลองที่ 1 ถึง 8 ขึ้นสู่ Remote ใน Github ผ่านโปรแกรม Eclipse

ลิงค์ Github ที่เก็บไฟล์ข้อมูลของการทดลองที่ 1 ถึง 8 ของคุณ
ลิงค์การทดลองที่ 1 -> https://github.com/xxdadixx/lab1-oop
ลิงค์การทดลองที่ 2 -> https://github.com/xxdadixx/lab2-oop
ลิงค์การทดลองที่ 3 -> https://github.com/xxdadixx/lab3-oop
ลิงค์การทดลองที่ 4 -> https://github.com/xxdadixx/lab4-oop
ลิงค์การทดลองที่ 5 -> https://github.com/xxdadixx/lab5-oop
ลิงค์การทดลองที่ 6 -> https://github.com/xxdadixx/lab6-oop
ลิงค์การทดลองที่ 7 -> https://github.com/xxdadixx/lab7-oop
ลิงค์การทดลองที่ 8 -> https://github.com/xxdadixx/lab8-oop

4.5. ทำการ Push โค้ดโปรแกรมตั้งแต่การทดลองที่ 1 ถึง 8 ขึ้นสู่ Remote โดยใช้โปรแกรม Eclipse

4.6. สร้างโปรเจคใหม่ใน Eclipse ที่เชื่อมต่อกับ Github ให้เรียบร้อย พร้อมทั้งหาสมาชิกร่วมกลุ่มจำนวน 3-4 คน เพื่อสร้างโปรแกรม “นาฬิกาสารพัดประโยชน์” ที่มีส่วนประกอบของฟีจเจอร์ต่างๆ ดังนี้



4.6.1. หน้าต่าง Current Clock เพื่อแสดงนาฬิกาที่จะทำงานตามโหมดต่างๆ ที่ผู้ใช้สั่งตามปุ่มต่างๆ

4.6.2. หน้าต่าง Normal จะปรากฏหน้าต่างนี้เมื่อคลิกปุ่ม Normal ที่อยู่ในหน้า Current Clock ซึ่งจะ แสดงส่วนการตั้งค่า GMT ให้กับนาฬิกาหลักหลังจากกดปุ่ม Submit Normal Setting เรียบร้อยแล้ว

4.6.3. หน้าต่าง Countdown จะปรากฏหน้าต่างนี้เมื่อคลิกปุ่ม Countdown ที่อยู่ในหน้า Current Clock ซึ่งจะ แสดงส่วนการตั้งค่าการนับเวลาถอยหลัง สามารถปรับค่าได้ในระดับชั่วโมง นาที และวินาที หลังจากกดปุ่ม Submit เรียบร้อย หน้าต่างการตั้งค่าจะหายไป และส่วนการแสดงนาฬิกาใน Current Clock ก็จะทำการเริ่มต้นนับถอยหลังไปเรื่อยๆ จนถึงเลข 0 นาฬิกา 0 นาที 0 วินาที

4.6.4. หน้าต่าง Alert จะปรากฏหน้าต่างนี้เมื่อคลิกปุ่ม Alert ที่อยู่ในหน้า Current Clock ซึ่งจะ แสดงส่วนการตั้งค่าเวลาปลุกเมื่อเวลาปัจจุบันเดินทางมาถึงเวลาที่กำหนดไว้ สามารถปรับค่าได้ในระดับชั่วโมง นาที และวินาที หลังจากกดปุ่ม Submit เรียบร้อย หน้าต่างการตั้งค่าจะหายไป และส่วนการแสดงนาฬิกาใน Current Clock ก็จะแสดงเวลาตามปกติ แต่เมื่อถึงเวลาที่ตั้งปลุกเอาไว้ ระบบก็จะปรากฏหน้าต่างแจ้งเตือน

4.6.5. (**หากมีสมาชิกในกลุ่มไม่ถึง 4 คน ไม่ต้องทำฟีจเจอร์นี้**) หน้าต่าง Stop จะปรากฏหน้าต่างนี้เมื่อคลิกปุ่ม Stop ที่อยู่ในหน้า Current Clock ซึ่งจะ แสดงส่วนการตั้งค่าการจับเวลา หลังจากกดปุ่ม Start Stopwatch เรียบร้อย หน้าต่างการตั้งค่าจะหายไป และส่วนการแสดงนาฬิกาใน Current Clock ก็จะเริ่มต้นจับเวลา โดยเริ่มตั้งแต่ 0 นาฬิกา 0 นาที 0 วินาที และ

จำนวนวินาทีจะเริ่มต้นเพิ่มขึ้นไปเรื่อยๆ จนกว่าผู้ใช้งานจะกดปุ่ม Stop อีกครั้ง เพื่อเป็นการหยุดการทำงานของนาฬิกาจับเวลา

4.7. จากฟีจเจอร์การทำงานของนาฬิกาข้างต้น ให้นักศึกษาแบ่งหน้าที่ในการกับเพื่อนร่วมงานในกลุ่มเพื่อสร้าง Repository และทำงานร่วมกันภายใน Remote นี้

4.7.1. ผู้รับผิดชอบทั้งหมด สร้างและพัฒนาส่วนของ Current Clock

4.7.2. ผู้รับผิดชอบคนที่ 1 สร้างและพัฒนาส่วนของ Normal

4.7.3. ผู้รับผิดชอบคนที่ 2 สร้างและพัฒนาส่วนของ Countdown

4.7.4. ผู้รับผิดชอบคนที่ 3 สร้างและพัฒนาส่วนของ Alert

4.7.5. ผู้รับผิดชอบคนที่ 4 (**ถ้ามี**) สร้างและพัฒนาส่วนของ Stop

4.8. นักศึกษาจะต้องทำงานร่วมกัน เพื่อให้เห็นภาพรวมการใช้งาน Eclipse ร่วมกับ Github ให้มองเห็นการทำงานเพื่อการแยก Branch, การ Merge Branch, การจัดการโค้ดโปรแกรมเมื่อเกิด Conflict

รายชื่อสมาชิกภายในกลุ่มของคุณ และหน้าที่รับผิดชอบภายในกลุ่ม

คนที่ 1 **ชื่อ-นามสกุล** อิทธิกร สิริเวชพันธุ์ **รหัสนักศึกษา** 62543502031-2
หน้าที่รับผิดชอบ Normal

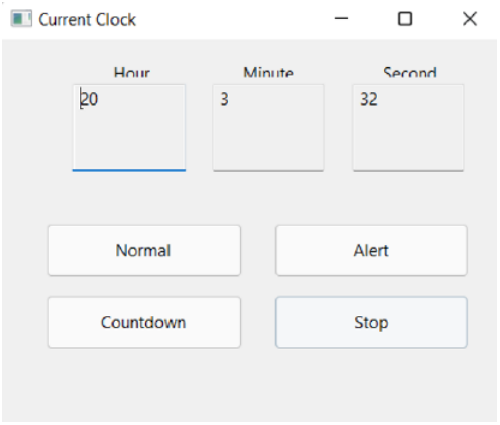
คนที่ 2 **ชื่อ-นามสกุล** อิทธิกร สิริเวชพันธุ์ **รหัสนักศึกษา** 62543502031-2
หน้าที่รับผิดชอบ Countdown

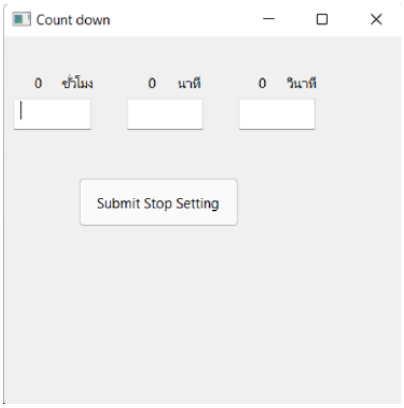
คนที่ 3 **ชื่อ-นามสกุล** อิทธิกร สิริเวชพันธุ์ **รหัสนักศึกษา** 62543502031-2
หน้าที่รับผิดชอบ Alert

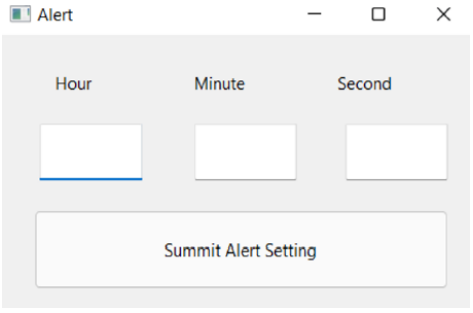
คนที่ 4 **ชื่อ-นามสกุล** อิทธิกร สิริเวชพันธุ์ **รหัสนักศึกษา** 62543502031-2
(ถ้ามี) **หน้าที่รับผิดชอบ** Stop

ลิงค์งานกลุ่มของคุณที่อยู่ใน Github

ผลลัพธ์การทำงานของโปรแกรม







โค้ดโปรแกรมภายในหน้าต่าง Current Clock

```java
import org.eclipse.swt.widgets.Display;
import org.eclipse.swt.widgets.Shell;
import org.eclipse.swt.widgets.Composite;
import org.eclipse.swt.SWT;
import org.eclipse.swt.widgets.Button;
import org.eclipse.swt.widgets.Label;
import org.eclipse.swt.widgets.Text;
import org.eclipse.swt.events.SelectionAdapter;
import org.eclipse.swt.events.SelectionEvent;
import org.eclipse.swt.widgets.DateTime;
import org.eclipse.wb.swt.SWTResourceManager;

import javax.swing.*;
import java.awt.*;
import java.text.SimpleDateFormat;
import java.time.LocalTime;
import java.time.format.DateTimeFormatter;
import java.util.Calendar;
import java.util.GregorianCalendar;
import java.text.SimpleDateFormat;
import org.eclipse.ui.forms.widgets.FormToolkit;

public class Main1 {

    protected Shell shell;


    SimpleDateFormat timeFormat;

    private String JH;
    private String JM;
    private String JS;
    private Text Hour;
    private int  sec ;
    private int  minute ;
    private int  hour ;
    public int Gmt  = 0;
    public int r  = 0;
    public int ah  = 0;
    public int am = 0;
    public int as  = 0;
    Normal Nm = new Normal();
    CLKAlertA Al = new CLKAlertA ();
    CLKAlertB Al2 = new CLKAlertB();
    Countdown Cd = new Countdown();
    private Text Min;
    private Text Sec;

    /**
     * Launch the application.
     * @param args
     */
    public static void main(String[] args) {


        try {

            Main1 window = new Main1();
            window.open();

        } catch (Exception e) {
            e.printStackTrace();
        }

    }


    /**
     * Open the window.
     */
    public void open() {
        Display display = Display.getDefault();
        setTime();
        createContents();
        shell.open();
        shell.layout();

        while (!shell.isDisposed()) {

            if (!display.readAndDispatch()) {

                display.sleep();

            }
        }

    }



    /**
     * Create contents of the window.
     */
    protected void createContents() {
        shell = new Shell();
        shell.setSize(473, 327);
        shell.setText("Current Clock");


        //setTime();
        Composite composite = new Composite(shell, SWT.NONE);
        composite.setBounds(30, 10, 415, 136);

        Label lblH = new Label(composite, SWT.NONE);
        lblH.setBounds(72, 10, 59, 14);
        lblH.setText("Hour");

        Label lblM = new Label(composite, SWT.NONE);
        lblM.setBounds(189, 10, 59, 14);
        lblM.setText("Minute");

        Label lblS = new Label(composite, SWT.NONE);
        lblS.setFont(SWTResourceManager.getFont(".AppleSystemUIFont", 11, SWT.NORMAL));
        lblS.setBounds(316, 10, 59, 14);
```

```
        lblS.setText("Second");

        Hour = new Text(composite, SWT.BORDER);
        Hour.setFont(SWTResourceManager.getFont(".AppleSystemUIFont", 30, SWT.NORMAL));
        Hour.setEditable(false);
        Hour.setBounds(35, 30, 102, 79);

        Min = new Text(composite, SWT.BORDER);
        Min.setFont(SWTResourceManager.getFont(".AppleSystemUIFont", 30, SWT.NORMAL));
        Min.setEditable(false);
        Min.setBounds(161, 30, 102, 79);

        Sec = new Text(composite, SWT.BORDER);
        Sec.setFont(SWTResourceManager.getFont(".AppleSystemUIFont", 30, SWT.NORMAL));
        Sec.setEditable(false);
        Sec.setBounds(288, 30, 102, 79);
        //formToolkit.adapt(text, true, true);

        //test Fetch



        Button btnNewButton = new Button(shell, SWT.NONE);
        btnNewButton.addSelectionListener(new SelectionAdapter() {

            public void widgetSelected(SelectionEvent e) {

                Nm.open();

                Gmt = Nm.getGMT();
                if(Gmt >=24) {
                    Gmt = Gmt - 24;

                }


            }
        });
        btnNewButton.setBounds(41, 166, 177, 49);
        btnNewButton.setText("Normal");

        Button btnNewButton_1 = new Button(shell, SWT.NONE);
        btnNewButton_1.addSelectionListener(new SelectionAdapter() {
            @Override
            public void widgetSelected(SelectionEvent e) {


                Al.open();
                setAlert();

            }
        });
        btnNewButton_1.setBounds(247, 166, 177, 49);
        btnNewButton_1.setText("Alert");

        Button btnNewButton_2 = new Button(shell, SWT.NONE);
        btnNewButton_2.addSelectionListener(new SelectionAdapter() {
            @Override
            public void widgetSelected(SelectionEvent e){

                Cd.open();

            }
        });
        btnNewButton_2.setBounds(41, 231, 177, 49);
        btnNewButton_2.setText("Countdown");

        Button btnNewButton_3 = new Button(shell, SWT.NONE);
        btnNewButton_3.addSelectionListener(new SelectionAdapter() {
            @Override
            public void widgetSelected(SelectionEvent e) {
            }
        });
        btnNewButton_3.setBounds(247, 231, 177, 49);
        btnNewButton_3.setText("Stop");


    }

    public void setTime() {

        new Thread(new Runnable() {
              public void run() {
                 while (true) {
                    try { Thread.sleep(1000); } catch (Exception e) { }
                    Display.getDefault().asyncExec(new Runnable() {
                       public void run() {

                                Calendar cal = new GregorianCalendar();
                                minute = cal.get(Calendar.MINUTE);
                                hour = cal.get(Calendar.HOUR_OF_DAY);
                                sec = cal.get(Calendar.SECOND);


                                hour = hour+Gmt;


                                if(hour >=24) {

                                    hour = hour- 24;

                                }

                                if(hour == ah && minute == am && sec == as) {

                                    Al2.open();

                                }


                                JH =""+hour;
                                JM =""+minute;
                                JS =""+sec;
                                Hour.setText(JH);
                                Min.setText(JM);
                                Sec.setText(JS);

                        }
                    });
                 }
              }
        }).start();


    }

    public void setAlert() {

        ah = Al.h ;
        am = Al.m ;
        as = Al.s;


    }
}
```

โค้ดโปรแกรมภายในหน้าต่าง Normal

```java
import org.eclipse.swt.widgets.Display;
import org.eclipse.swt.widgets.Shell;
import org.eclipse.swt.widgets.Label;
import org.eclipse.swt.SWT;
import org.eclipse.wb.swt.SWTResourceManager;

import org.eclipse.swt.widgets.Text;
import org.eclipse.swt.widgets.Button;
import org.eclipse.swt.events.SelectionAdapter;
import org.eclipse.swt.events.SelectionEvent;

public class Normal {

    protected Shell shell;
    private Text text;
    public int GMT = 0;
    /**
     * Launch the application.
     * @param args
     */
    public static void main(String[] args) {
        try {
            Normal window = new Normal();
            window.open();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    /**
     * Open the window.
     */
    public void open() {
        Display display = Display.getDefault();
        createContents();
        shell.open();
        shell.layout();
        while (!shell.isDisposed()) {
            if (!display.readAndDispatch()) {
                display.sleep();
            }
        }
    }

    /**
     * Create contents of the window.
     */
    protected void createContents() {
        shell = new Shell();
        shell.setSize(450, 300);
        shell.setText("Normal");
        Main1 window1 = new Main1();

        Label lblYourGmt = new Label(shell, SWT.NONE);
        lblYourGmt.setFont(SWTResourceManager.getFont(".AppleSystemUIFont", 44, SWT.NORMAL));
        lblYourGmt.setBounds(10, 76, 200, 84);
        lblYourGmt.setText("Your GMT");

        text = new Text(shell, SWT.BORDER);
        text.setBounds(206, 68, 220, 77);

        Button btnNewButton = new Button(shell, SWT.NONE);
        btnNewButton.addSelectionListener(new SelectionAdapter() {
            @Override
            public void widgetSelected(SelectionEvent e) {


                GMT = Integer.parseInt(text.getText());
                window1.r = GMT;
                shell.close();


            }
        });
        btnNewButton.setBounds(20, 161, 406, 77);
        btnNewButton.setText("Summit Normal Setting");

    }


    public int getGMT() {

        return this.GMT ;
    }//end method
}
```

โค้ดโปรแกรมภายในหน้าต่าง Countdown

```
import org.eclipse.swt.widgets.Display;
import org.eclipse.swt.widgets.Shell;

import org.eclipse.swt.SWT;
import org.eclipse.swt.widgets.Text;
import org.eclipse.swt.widgets.Button;
import org.eclipse.swt.events.SelectionAdapter;
import org.eclipse.swt.events.SelectionEvent;

import javax.swing.Timer;
import org.eclipse.swt.widgets.Label;
import org.eclipse.wb.swt.SWTResourceManager;
public class Countdown {

    protected Shell shell;
    private Text text1;
    private Text text2;
    private Text text3;

    Timer timer;
    int sec,min,hour;
    int usesec1,usesec2,usesec3;
    String sec1= "";
    String min1= "";
    String hour1= "";
    private Label la3;
    private Label la2;
    private Label la1;
    private Label lblNewLabel;
    private Label lblNewLabel_1;
    private Label lblNewLabel_2;
    /**
     * Launch the application.
     * @param args
     */
    public static void main(String[] args) {
        try {
            Countdown window = new Countdown();
            window.open();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    /**
     * Open the window.
     */
    public void open() {
        Display display = Display.getDefault();
        createContents();
        shell.open();
        shell.layout();
        while (!shell.isDisposed()) {
            if (!display.readAndDispatch()) {
                display.sleep();
            }
        }
    }
```

```

    /**
     * Create contents of the window.
     */
    protected void createContents() {
        shell = new Shell();
        shell.setSize(364, 275);
        shell.setText("Count down");

        text1 = new Text(shell, SWT.BORDER);
        text1.setBounds(10, 67, 84, 34);

        text2 = new Text(shell, SWT.BORDER);
        text2.setBounds(132, 67, 84, 34);

        text3 = new Text(shell, SWT.BORDER);
        text3.setBounds(254, 67, 84, 34);

        Button button1 = new Button(shell, SWT.NONE);
        button1.addSelectionListener(new SelectionAdapter() {
            @Override
            public void widgetSelected(SelectionEvent e) {
                hour1 = text1.getText();
                la1.setText(hour1);
                usesec1 = Integer.parseInt(hour1);

                min1 = text2.getText();
                la2.setText(min1);
                usesec2 = Integer.parseInt(min1);

                sec1 = text3.getText();
                la3.setText(sec1);
                usesec3 = Integer.parseInt(sec1);

                new Thread(new Runnable() {
                    public void run() {
                        while (true) {
                            try { Thread.sleep(1000); } catch (Exception e) { }
                            Display.getDefault().asyncExec(new Runnable() {
                                public void run() {
                                    if(usesec3 < 62){
                                        usesec3--;
                                        String usetime3 = Integer.toString(usesec3);
                                        //JOptionPane.showMessageDialog(null,"show number get: "+usetime1);
                                        la3.setText(""+usetime3);
                                        if(usesec3 == 0) {
                                            usesec3 = 60;
                                            if(usesec2 == 0) {
                                                usesec2 = 60;
                                                usesec1--;
                                                String usetime1 = Integer.toString(usesec1);
                                                la1.setText(""+usetime1);
                                            }
                                        usesec2--;
                                        String usetime2 = Integer.toString(usesec2);
                                        la2.setText(""+usetime2);
                                    }
```

```
                                }
                            });
                        }
                    }).start();
                }
            });
            button1.setBounds(80, 153, 174, 52);
            button1.setText("Submit Stop Setting");

            la3 = new Label(shell, SWT.NONE);
            la3.setFont(SWTResourceManager.getFont("Segoe UI", 12, SWT.NORMAL));
            la3.setBounds(276, 40, 24, 21);
            la3.setText("0");

            la2 = new Label(shell, SWT.NONE);
            la2.setFont(SWTResourceManager.getFont("Segoe UI", 12, SWT.NORMAL));
            la2.setBounds(156, 40, 24, 21);
            la2.setText("0");

            la1 = new Label(shell, SWT.NONE);
            la1.setFont(SWTResourceManager.getFont("Segoe UI", 12, SWT.NORMAL));
            la1.setText("0");
            la1.setBounds(32, 40, 24, 21);

            lblNewLabel = new Label(shell, SWT.NONE);
            lblNewLabel.setFont(SWTResourceManager.getFont("Segoe UI", 12, SWT.NORMAL));
            lblNewLabel.setBounds(307, 40, 44, 21);
            lblNewLabel.setText("\u0E27\u0E34\u0E19\u0E32\u0E17\u0E35");

            lblNewLabel_1 = new Label(shell, SWT.NONE);
            lblNewLabel_1.setFont(SWTResourceManager.getFont("Segoe UI", 12, SWT.NORMAL));
            lblNewLabel_1.setText("\u0E19\u0E32\u0E17\u0E35");
            lblNewLabel_1.setBounds(188, 40, 39, 21);

            lblNewLabel_2 = new Label(shell, SWT.NONE);
            lblNewLabel_2.setFont(SWTResourceManager.getFont("Segoe UI", 12, SWT.NORMAL));
            lblNewLabel_2.setText("\u0E0A\u0E31\u0E48\u0E27\u0E42\u0E21\u0E07");
            lblNewLabel_2.setBounds(62, 40, 44, 21);

        }
    }
```

โค้ดโปรแกรมภายในหน้าต่าง Alert

```java
import org.eclipse.swt.widgets.Display;
import org.eclipse.swt.widgets.Shell;
import org.eclipse.swt.widgets.Label;
import org.eclipse.swt.SWT;
import org.eclipse.swt.widgets.Text;
import org.eclipse.swt.widgets.Button;
import org.eclipse.wb.swt.SWTResourceManager;
import org.eclipse.swt.events.SelectionAdapter;
import org.eclipse.swt.events.SelectionEvent;

public class CLKAlertA {

    protected Shell shell;
    private Text Hour;
    private Text Minute;
    private Text Second;
    public int h = 0;
    public int m = 0;
    public int s = 0;

    /**
     * Launch the application.
     * @param args
     */
    public static void main(String[] args) {
        try {
            CLKAlertA window = new CLKAlertA();
            window.open();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    /**
     * Open the window.
     */
    public void open() {
        Display display = Display.getDefault();
        createContents();
        shell.open();
        shell.layout();
        while (!shell.isDisposed()) {
            if (!display.readAndDispatch()) {
                display.sleep();
            }
        }
    }

    /**
     * Create contents of the window.
     */
    protected void createContents() {
        shell = new Shell();
        shell.setSize(450, 300);
        shell.setText("Alert");
        Main1 M1 = new Main1();

        Label lblHour = new Label(shell, SWT.NONE);
        lblHour.setAlignment(SWT.CENTER);
        lblHour.setFont(SWTResourceManager.getFont("AngsanaUPC", 14, SWT.NORMAL));
        lblHour.setBounds(50, 33, 73, 30);
        lblHour.setText("Hour");

        Hour = new Text(shell, SWT.BORDER);
        Hour.setBounds(36, 80, 93, 51);

        Button btnSubmit = new Button(shell, SWT.NONE);
        btnSubmit.addSelectionListener(new SelectionAdapter() {
            @Override
            public void widgetSelected(SelectionEvent e) {

                h = Integer.parseInt(Hour.getText());
                m = Integer.parseInt(Minute.getText());
                s = Integer.parseInt(Second.getText());


                M1.ah = h;
                M1.am = m;
                M1.as = s;

                shell.close();

            }
        });
        btnSubmit.setFont(SWTResourceManager.getFont("AngsanaUPC", 22, SWT.NORMAL));
        btnSubmit.setBounds(31, 158, 375, 70);
        btnSubmit.setText("Submit Alert Setting");

        Minute = new Text(shell, SWT.BORDER);
        Minute.setBounds(176, 80, 93, 51);

        Second = new Text(shell, SWT.BORDER);
        Second.setBounds(313, 80, 93, 51);

        Label lblMinute = new Label(shell, SWT.NONE);
        lblMinute.setAlignment(SWT.CENTER);
        lblMinute.setText("Minute");
        lblMinute.setFont(SWTResourceManager.getFont("AngsanaUPC", 16, SWT.NORMAL));
        lblMinute.setBounds(176, 33, 93, 30);

        Label lblSecond = new Label(shell, SWT.NONE);
        lblSecond.setAlignment(SWT.CENTER);
        lblSecond.setText("Second");
        lblSecond.setFont(SWTResourceManager.getFont("AngsanaUPC", 16, SWT.NORMAL));
        lblSecond.setBounds(306, 33, 110, 30);

    }
}
```

```java
import org.eclipse.swt.widgets.Display;
import org.eclipse.swt.widgets.Shell;
import org.eclipse.swt.widgets.Label;
import org.eclipse.swt.SWT;
import org.eclipse.wb.swt.SWTResourceManager;
import org.eclipse.swt.widgets.Composite;
import org.eclipse.swt.widgets.Button;
import org.eclipse.swt.events.SelectionAdapter;
import org.eclipse.swt.events.SelectionEvent;

public class CLKAlertB {

    protected Shell shell;

    /**
     * Launch the application.
     * @param args
     */
    public static void main(String[] args) {
        try {
            CLKAlertB window = new CLKAlertB();
            window.open();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    /**
     * Open the window.
     */
    public void open() {
        Display display = Display.getDefault();
        createContents();
        shell.open();
        shell.layout();
        while (!shell.isDisposed()) {
            if (!display.readAndDispatch()) {
                display.sleep();
            }
        }
    }

    /**
     * Create contents of the window.
     */
    protected void createContents() {
        shell = new Shell();
        shell.setSize(450, 300);
        shell.setText("Alert!");

        Composite composite = new Composite(shell, SWT.NONE);
        composite.setBackground(SWTResourceManager.getColor(SWT.COLOR_WHITE));
        composite.setBounds(46, 28, 348, 106);

        Label lblAlert = new Label(composite, SWT.NONE);
        lblAlert.setBackground(SWTResourceManager.getColor(255, 0, 0));
        lblAlert.setAlignment(SWT.CENTER);
        lblAlert.setForeground(SWTResourceManager.getColor(SWT.COLOR_WHITE));
        lblAlert.setBounds(10, 10, 328, 86);
        lblAlert.setFont(SWTResourceManager.getFont("BrowalliaUPC", 32, SWT.BOLD));
        lblAlert.setText("It's Time to Alert!!!!!!");

        Button btnNewButton = new Button(shell, SWT.NONE);
        btnNewButton.addSelectionListener(new SelectionAdapter() {
            @Override
            public void widgetSelected(SelectionEvent e) {

                shell.close();

            }
        });
        btnNewButton.setFont(SWTResourceManager.getFont(".AppleSystemUIFont", 17, SWT.NORMAL));
        btnNewButton.setBounds(86, 152, 265, 76);
        btnNewButton.setText("Close");

    }
}
```

โค้ดโปรแกรมภายในหน้าต่าง Stop

```
142            public void widgetSelected(SelectionEvent e) {
143
144                Nm.open();
145
146                Gmt = Nm.getGMT();
147                if(Gmt >=24) {
148                    Gmt = Gmt - 24;
149
150                }
151
152
153
154            }
155        });
156        btnNewButton.setBounds(41, 166, 177, 49);
157        btnNewButton.setText("Normal");
158
159        Button btnNewButton_1 = new Button(shell, SWT.NONE);
160        btnNewButton_1.addSelectionListener(new SelectionAdapter() {
161            @Override
162            public void widgetSelected(SelectionEvent e) {
163
164
165                Al.open();
166                setAlert();
167
168            }
169        });
170        btnNewButton_1.setBounds(247, 166, 177, 49);
171        btnNewButton_1.setText("Alert");
172
173        Button btnNewButton_2 = new Button(shell, SWT.NONE);
174        btnNewButton_2.addSelectionListener(new SelectionAdapter() {
175            @Override
176            public void widgetSelected(SelectionEvent e){
177
178                Cd.open();
179
180            }
181        });
182        btnNewButton_2.setBounds(41, 231, 177, 49);
183        btnNewButton_2.setText("Countdown");
184
185        Button btnNewButton_3 = new Button(shell, SWT.NONE);
186        btnNewButton_3.addSelectionListener(new SelectionAdapter() {
187            @Override
188            public void widgetSelected(SelectionEvent e) {
189            }
190        });
191        btnNewButton_3.setBounds(247, 231, 177, 49);
192        btnNewButton_3.setText("Stop");
193
194
195
196    }
```

5. สรุปผลการปฏิบัติการ

สามารถใช้งาน Commit จาก eclipse ไปยัง Github สามารถทำตัวโปรแกรมของเรื่อง Time แต่ติดปัญหาเรื่องของ Start Stop GMT และ Time

6. คำถามท้ายการทดลอง

6.1. ควร Commit อย่างไร เพื่อหลีกเลี่ยงการเกิด Conflict ให้เหมาะสมที่สุด

ทำส่วนของ Project หรือ ตัวไฟล์งานไว้เลย อย่างรวมไฟล์แล้ว commit ทีเดียว เพราะอาจทำให้การ commit นั้นเกิด การ conflict และอาจทำให้ Pull code มีปัญหาได้

6.2. ควรมีหลักเกณฑ์ในการ Push ขึ้นไปบน Remote เมื่อใดจึงจะเหมาะสมที่สุด

เลือกไฟล์ที่ต้องอัพขึ้น git แล้วหลังจากนั้นค่อย Share project เสร็จเช็คก่อนที่จะไป staged changes ขั้นตอนต่อไปทำการ commit ก่อนแล้วค่อยไป push and commit

6.3. เมื่อใดจึงควรใช้คำสั่ง Fetch

เมื่อต้องการเช็คข้อมูลว่าใครที่ push เข้ามาทำแล้วบ้างเราไม่จำเป็นต้อง pull เข้าเครื่อง fetch ยังสามารถเช็ค history ทั้งหมดได้ด้วย

6.4. เราควรจะแยก Branch เมื่อใด? และควรจะ Merge Branch เมื่อใด?

เมื่อเราจะอัพไฟล์ไปใน Git Branch เพราะเราต่างคนต่างทำโค้ดอาจทำให้เวลาทำ Feature อาจไม่รู้ว่า เป็นของหรืออาจทำให้ Source Code รวมอยู่ในไฟล์เดียวกันได้ เราควร Merge Branch เมื่อมีการจะ push โค้ดเข้าในตัวของ git hub