

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 คือระบบซอฟต์แวร์ที่จะคอยบันทึกเวอร์ชันการเปลี่ยนแปลงของโค้ดหรือเอกสารต่างๆ โดยจะทำการบันทึกไว้ด้วยว่าการเปลี่ยนแปลงแต่ละครั้งนั้นทำเพื่ออะไร และทำโดยใคร

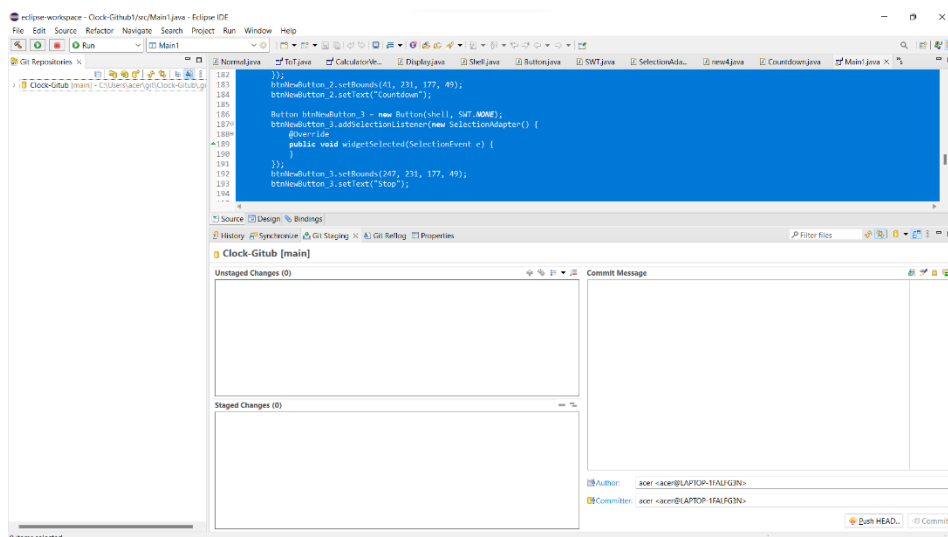
ประโยชน์ ช่วยให้สามารถย้อนไฟล์บางไฟล์หรือแม้กระทั่งทั้งโปรเจกต์กลับไปเป็นเวอร์ชันเก่าได้ นอกจากนั้นระบบ VCS ยังจะช่วยให้เปรียบเทียบการแก้ไขที่เกิดขึ้นในอดีต ดูว่าใครเป็นคนแก้ไขคนสุดท้ายที่อาจทำให้เกิดปัญหา แก้ไขเมื่อไร

- 3.2. Git ต่างกับ Github อย่างไร?

Git เป็นระบบที่ช่วยจัดการการแก้ไขใน Repository ส่วน GitHub เป็นบริการจัดเก็บ Repository ออนไลน์พร้อมกับฟีเจอร์อำนวยความสะดวกต่าง ๆ ที่ให้เราไปทำงานร่วมกันคนอื่นได้

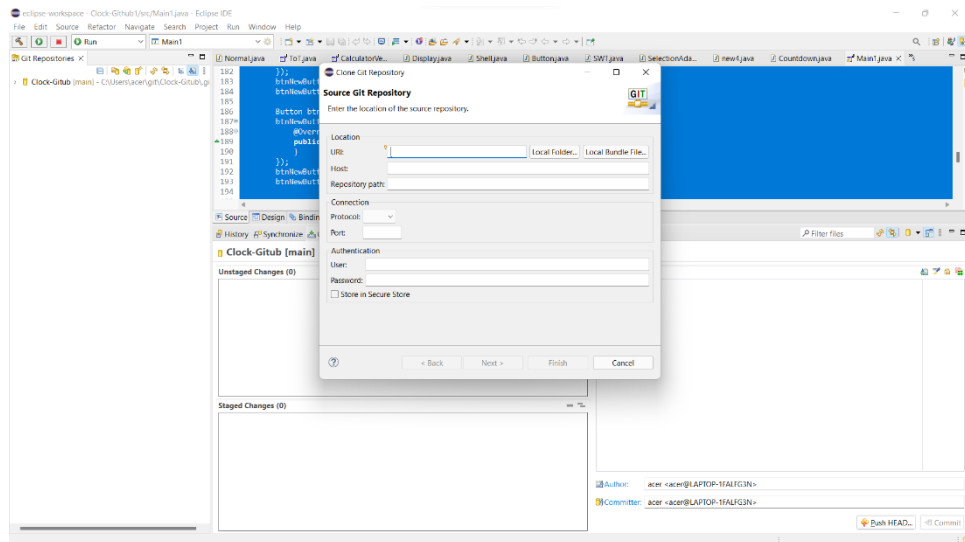
- 3.3. Repository คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

คือการเก็บสำรองข้อมูลและการเปลี่ยนแปลงของ Source Code ทำให้สามารถย้อนกลับไปที่เวอร์ชันใดๆ ก่อนหน้า และดูรายละเอียดการเปลี่ยนแปลงของแต่ละเวอร์ชันได้ นอกจากนั้นยังสามารถดูได้ว่าใครเป็นคนแก้ไข



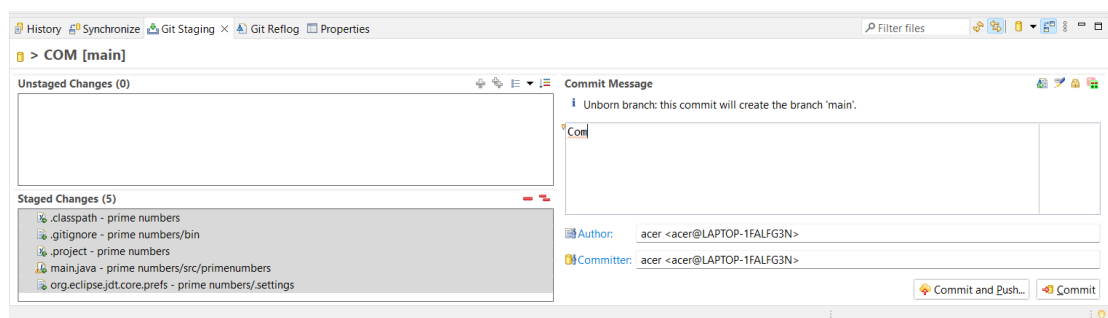
3.4. Clone คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

เวลาที่ผู้อ่านมี Repository อยู่บน Remote ซักแห่งอยู่แล้ว และต้องการ Sync มาลงเครื่องของเรา เราจะต้องทำสิ่งที่เรียกว่า Clone Repository หรือก็คือการก๊อปปี้ Repository จาก Remote คือ เวลาที่ผู้อ่านมี Repository อยู่บน Remote ซักแห่งอยู่แล้ว และต้องการ Sync มาลงเครื่องของเรา เราจะต้องทำสิ่งที่เรียกว่า Clone Repository หรือก็คือการก๊อปปี้ Repository จาก Remote



3.5. Commit คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

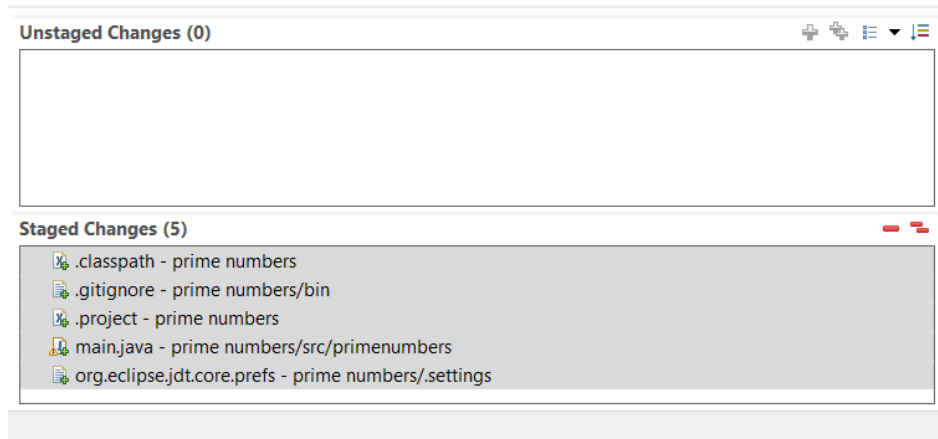
เวลาที่มีข้อมูลที่แก้ไขเสร็จแล้ว (โค้ดที่เขียนคำสั่งบางอย่างเสร็จแล้ว) แล้วอยากจะทำ Backup เก็บไว้ใน VCS จะเรียกกันว่า *Commit*



3.6. Staged และ Unstaged คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

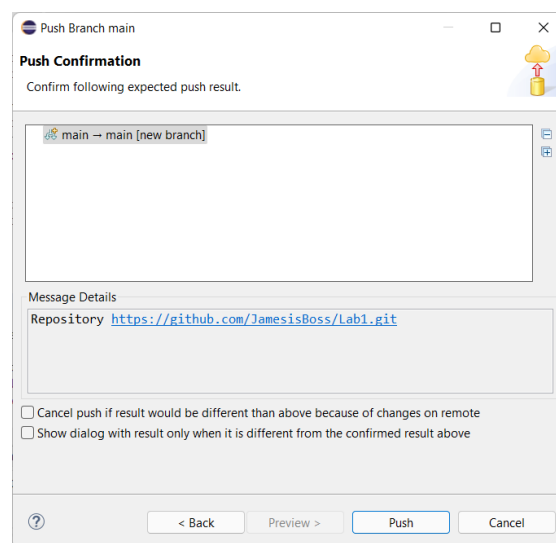
เวลาเราแก้ไขโค้ดหรือแก้ไขข้อมูล ไฟล์ที่ถูกแก้ไขจะอยู่ในสถานะ Unstaged และเวลาที่เราทำอะไรเสร็จเรียบร้อยแล้ว อยากจะ Commit เก็บไว้ จะต้องเลือกไฟล์ที่ต้องการเพื่อย้ายเข้าสู่สถานะ Staged ก่อนถึงจะทำการ Commit ได้

 > COM [main]



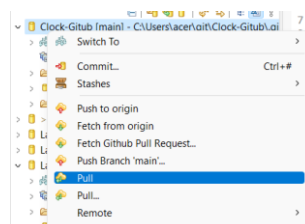
3.7. Push คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

Push คือการนำโค้ดหรือไฟล์เข้าตัวระบบ Git Repository



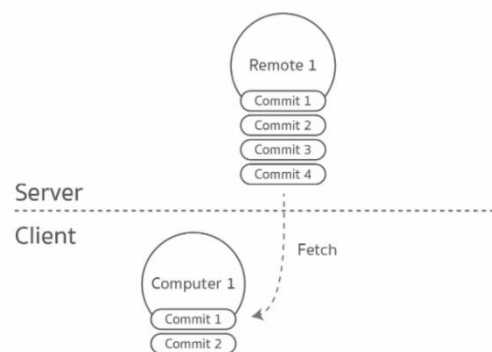
3.8. Pull คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

คือ เวลา Sync จาก Remote เพื่อดึงข้อมูล Commit ใหม่ ๆ ลงมาเก็บไว้ในเครื่องจะเรียกขั้นตอนนี้ว่า Pull



3.9. Fetch คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

อยากเช็คสถานะของ Remote ง่ายๆว่ามีใคร Push ข้อมูลใหม่ขึ้นไป Remote หรือป่าว เราเรียกวิธีนี้ว่า Fetch

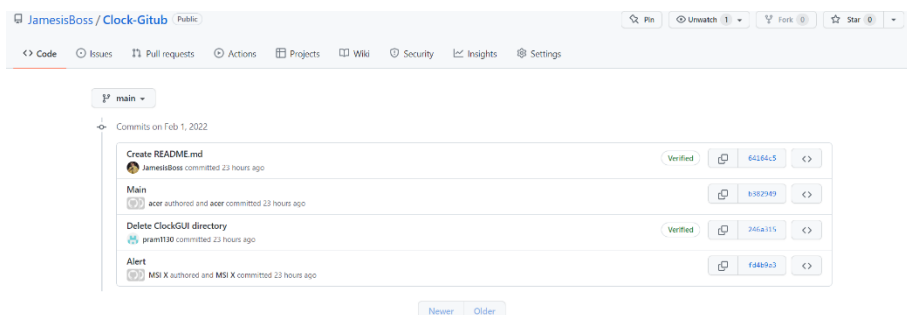


3.10. Conflict ใน VSC คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

คือ การเกิดปัญหาการชนของข้อมูลในไฟล์งานที่ทำร่วมกันกับเพื่อนเรา ซึ่งในช่วงที่เราพัฒนาโปรแกรม หรือเขียนโค้ดกับเพื่อนร่วมงานอยู่นั้นเราไม่สามารถรู้ได้เลยว่าเพื่อนเราจะเขียนโค้ดไปในรูปแบบไหน

3.11. Merge Commit คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

คือการที่มีการแตก branch ออกไป develop แยกกัน โดยที่มีการแก้ไขไฟล์เดียวกันซึ่งโค้ดนั้นอาจมีการทับซ้อน หรืออยู่บรรทัดเดียวกัน เมื่อใครคนใดคนหนึ่งนำโค้ดมา Merge รวมกันนั้นจะเกิดสิ่งที่เรียกว่า Conflict คือโค้ดของทั้งสองคนมีความขัดแย้งกัน

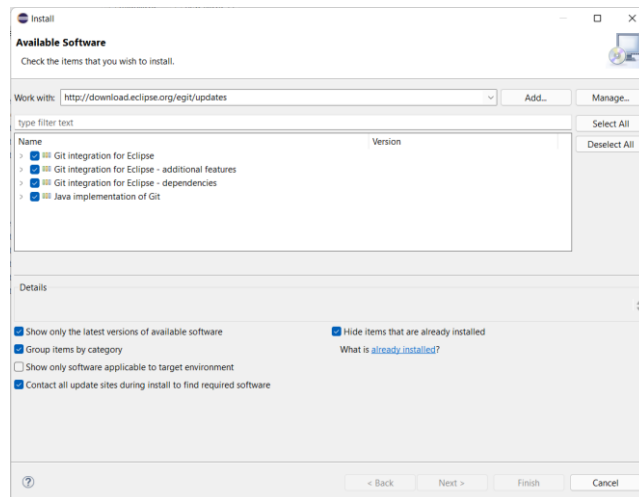


3.12. ขั้นตอนที่อยู่ในระหว่าง Development Process ภายใน VSC มีอะไรบ้าง?

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 URL แล้วคลิกที่ Egit



3. หลังจากนั้นกด Next แล้วรออาจใช้เวลาานาน รอจนกว่าตัวโปรแกรมจะขึ้นให้ restart แล้วเปิดโปรแกรมใหม่ หลังจากนั้นก็สามารถใช้ส่วนของ Git ได้เลย

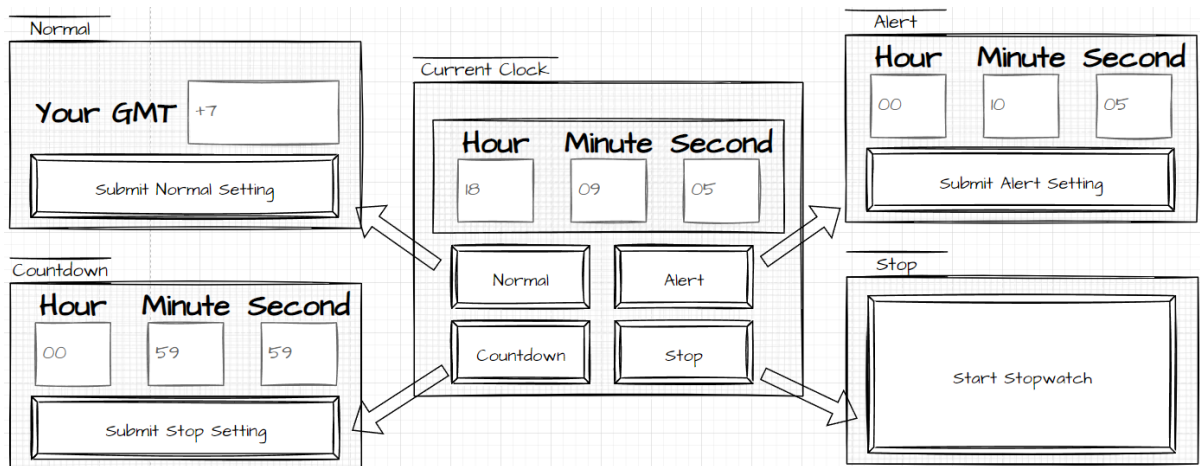
4. ลำดับขั้นตอนการปฏิบัติการ

- 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/JamesisBoss/Lab1>
- ลิงค์การทดลองที่ 2 -> <https://github.com/JamesisBoss/Lab2>
- ลิงค์การทดลองที่ 3 -> <https://github.com/JamesisBoss/Lab3>
- ลิงค์การทดลองที่ 4 -> <https://github.com/JamesisBoss/Lab4>
- ลิงค์การทดลองที่ 5 -> <https://github.com/JamesisBoss/Lab5>
- ลิงค์การทดลองที่ 6 -> <https://github.com/JamesisBoss/Lab6>
- ลิงค์การทดลองที่ 7 -> <https://github.com/JamesisBoss/Lab7>
- ลิงค์การทดลองที่ 8 -> <https://github.com/JamesisBoss/Lab8>

- 4.5. ทำการ Push โค้ดโปรแกรมตั้งแต่การทดลองที่ 1 ถึง 8 ขึ้นสู่ Remote โดยใช้โปรแกรม Eclipse
- 4.6. สร้างโปรเจกใหม่ใน Eclipse ที่เชื่อมต่อกับ Github ให้เรียบร้อย พร้อมทั้งหาสมาชิกในกลุ่มจำนวน 3-4 คน เพื่อสร้างโปรแกรม “นาฬิกาสารพัดประโยชน์” ที่มีส่วนประกอบของฟังก์เจอร์ต่างๆ ดังนี้



- 4.6.1. หน้าต่าง Current Clock เพื่อแสดงนาฬิกาที่จะทำงานตามโหมดต่างๆ ที่ผู้ใช้สั่งตามปุ่มต่างๆ
- 4.6.2. หน้าต่าง Normal จะปรากฏหน้าต่างนี้เมื่อคลิกปุ่ม Normal ที่อยู่ในหน้า Current Clock ซึ่งจะแสดงส่วนการตั้งค่า GMT ให้กับนาฬิกาหลักหลังจากกดปุ่ม Submit Normal Setting เรียบร้อยแล้ว
- 4.6.3. หน้าต่าง Countdown จะปรากฏหน้าต่างนี้เมื่อคลิกปุ่ม Countdown ที่อยู่ในหน้า Current Clock ซึ่งจะแสดงส่วนการตั้งค่าการนับเวลาถอยหลัง สามารถปรับค่าได้ในระดับชั่วโมง นาที

และวินาที หลังจากกดปุ่ม Submit เรียบร้อย หน้าต่างการตั้งค่าจะหายไป และส่วนการแสดงนาฬิกาใน Current Clock ก็จะทำให้การเริ่มต้นนับถอยหลังไปเรื่อยๆ จนถึงเลข 0 นาฬิกา 0 นาที 0 วินาที

4.6.4. หน้าต่าง Alert จะปรากฏหน้าต่างนี้เมื่อคลิกปุ่ม Alert ที่อยู่ในหน้า Current Clock ซึ่งจะ แสดงส่วนการตั้งค่าเวลาปลุกเมื่อเวลาปัจจุบันเดินทางมาถึงเวลาที่กำหนดไว้ สามารถปรับค่าได้ในระดับ ชั่วโมง นาที และวินาที หลังจากกดปุ่ม Submit เรียบร้อย หน้าต่างการตั้งค่าจะหายไป และส่วนการแสดงผลนาฬิกาใน Current Clock ก็จะแสดงเวลาตามปกติ แต่เมื่อถึงเวลาที่ตั้งปลุกเอาไว้ ระบบก็จะปรากฏหน้าต่างแจ้งเตือน

4.6.5. (หากมีสมาชิกในกลุ่มไม่ถึง 4 คน ไม่ต้องทำฟีเจอร์นี้) หน้าต่าง Stop จะปรากฏหน้าต่างนี้เมื่อคลิกปุ่ม Stop ที่อยู่ในหน้า Current Clock ซึ่งจะ แสดงส่วนการตั้งค่าการจับเวลา หลังจากกดปุ่ม Start Stopwatch เรียบร้อย หน้าต่างการตั้งค่าจะหายไป และส่วนการแสดงผลนาฬิกาใน Current Clock ก็จะเริ่มต้นจับเวลา โดยเริ่มตั้งแต่ 0 นาฬิกา 0 นาที 0 วินาที และจำนวนวินาทีจะเริ่มต้นเพิ่มขึ้นไปเรื่อยๆ จนกว่าผู้ใช้งานจะกดปุ่ม Stop อีกครั้ง เพื่อเป็นการหยุดการทำงานของนาฬิกาจับเวลา

4.7. จากฟีเจอร์การทำงานของนาฬิกาข้างต้น 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

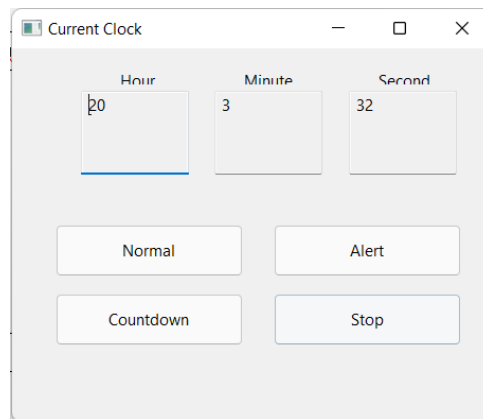
รายชื่อสมาชิกภายในกลุ่มของคุณ และหน้าที่รับผิดชอบภายในกลุ่ม

คนที่ 1 ชื่อ-นามสกุล นาย ศักรินทร์ เสนวิรัช รหัสนักศึกษา 62543502048-6
หน้าที่รับผิดชอบ Current Clock ,Countdown ,Alert

ลิงค์งานกลุ่มของคุณที่อยู่ใน Github

<https://github.com/saksornpoon/All-Lab/tree/main/Clock-Gitub/Clock-Gitub-main>

ผลลัพธ์การทำงานของโปรแกรม

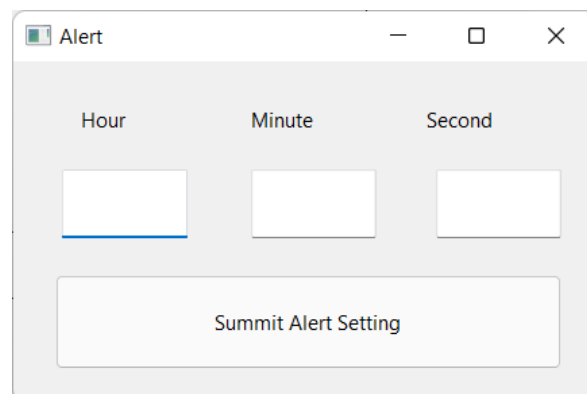


Current Clock

Hour: 20 Minute: 3 Second: 32

Normal Alert

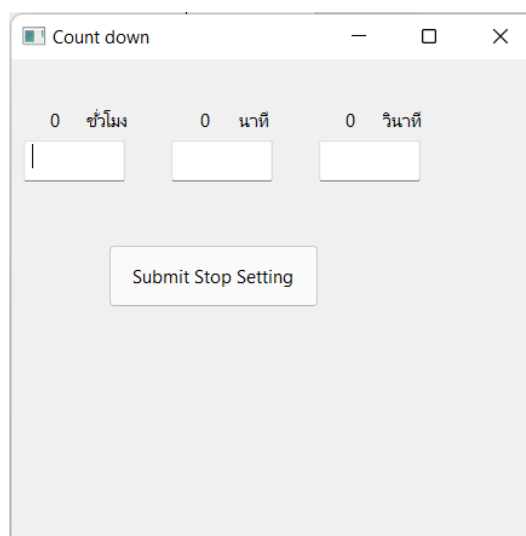
Countdown Stop



Alert

Hour Minute Second

Summit Alert Setting



Count down

0 ชั่วโมง 0 นาที 0 วินาที

Submit Stop Setting

โค้ดโปรแกรมภายในหน้าต่าง Current Clock

```
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.LocalDateTime;
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();
    Alert Al = new Alert();
    Alert2 Al2 = new Alert2();
    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

```
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.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--;
                                }
                            }
                        });
                    }
                }
            }).start();
        }
    });
}

```

```

Integer.toString(usesec3);

        //JOptionPane.showMessageDialog(null,"show number get: "+usetime1);

        la3.setText(""+usetime3);

60;

        if(usesec2 == 0) {

            usesec2 = 60;

            usesec1--;

            String usetime1 = Integer.toString(usesec1);

            la1.setText(""+usetime1);

        }

        Integer.toString(usesec2);

        la2.setText(""+usetime2);

        }

        }

        }); //if(usesec1 == 0) {

        //      break;

        //}

        }

        }

        }).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

```
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.swt.SWTResourceManager;
import org.eclipse.swt.events.SelectionAdapter;
import org.eclipse.swt.events.SelectionEvent;

public class Alert {

    protected Shell shell;
    private Text H;
    private Text M;
    private Text S;
    public int h = 0;
    public int m = 0;
    public int s = 0;

    /**
     * Launch the application.
     * @param args
     */
    public static void main(String[] args) {
        try {
            Alert window = new Alert();
            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 lblNewLabel = new Label(shell, SWT.NONE);
    lblNewLabel.setFont(SWTResourceManager.getFont(".AppleSystemUIFont", 30,
SWT.NORMAL));
    lblNewLabel.setBounds(50, 33, 73, 30);
    lblNewLabel.setText("Hour");

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

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

            h = Integer.parseInt(H.getText());
            m = Integer.parseInt(M.getText());
            s = Integer.parseInt(S.getText());

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

            shell.close();

        }
    });
    btnNewButton.setFont(SWTResourceManager.getFont(".AppleSystemUIFont", 21,
SWT.NORMAL));
    btnNewButton.setBounds(31, 158, 375, 70);
    btnNewButton.setText("Summit Alert Setting");

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

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

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

```

```

        Label lblNewLabel_3_1 = new Label(shell, SWT.NONE);
        lblNewLabel_3_1.setText("Second");
        lblNewLabel_3_1.setFont(SWTResourceManager.getFont(".AppleSystemUIFont",
30, SWT.NORMAL));
        lblNewLabel_3_1.setBounds(306, 33, 110, 30);

    }
}

```

โค้ดโปรแกรมภายในหน้าต่าง Stop

```

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");
}
```

5. สรุปผลการปฏิบัติการ

สามารถใช้งาน Commit จาก eclipse ไปยัง Github แต่ติดปัญหาเรื่อง Merge งานเพื่อนขึ้นมาบนเครื่องเราแล้ว ติดปัญหานิดหน่อย แต่ก็สามารถ Merge มาทำได้ ศึกษาจากคลิปอินเดียครับ สามารถทำตัวโปรแกรมของเรื่อง Time แต่ติดปัญหาเรื่องของ Start Stop GMT และ Time ติด Bug ของการเริ่มแล้วไม่หยุดไม่ได้ แต่ทั้งนี้ก็ขอบคุณ อาจารย์ด้วยนะครับที่ยืดเวลาให้ผม ขอขอบคุณครับ

6. คำถามท้ายการทดลอง

- 6.1. ควร Commit อย่างไร เพื่อหลีกเลี่ยงการเกิด Conflict ให้เหมาะสมที่สุด
ทำส่วนของ Project หรือ ตัวไฟล์งานไว้เลย อย่างรวมไฟล์แล้ว commit ที่เดียว เพราะอาจทำให้การ commit นั้นเกิด การ conflict และอาจทำให้ Pull code มีปัญหาได้
- 6.2. ควรมีหลักเกณฑ์ในการ Push ขึ้นไปบน Remote เมื่อใดจึงจะเหมาะสมที่สุด
เลือกไฟล์ที่ต้องอัปเดต git แล้วหลังจากนั้นค่อย Share project เสร็จเช็คก่อนที่จะไป staged changes ขึ้นต่อไปทำการ 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?
เมื่อเราจะอัปเดตไฟล์ไปใน Git Branch เพราะเราต่างคนต่างทำโค้ดอาจทำให้เวลาทำ Feature อาจไม่รู้ว่าเป็นของหรืออาจทำให้ Source Code รวมอยู่ในไฟล์เดียวกันได้ เราควร Merge Branch เมื่อมีการจะ push โค้ดเข้าในตัวของ git hub