

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 ช่วยให้เราสามารถย้อนไฟล์บางไฟล์หรือแม้กระทั่งทั้งโปรเจกต์กลับไปเป็นเวอร์ชันเก่าได้ 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

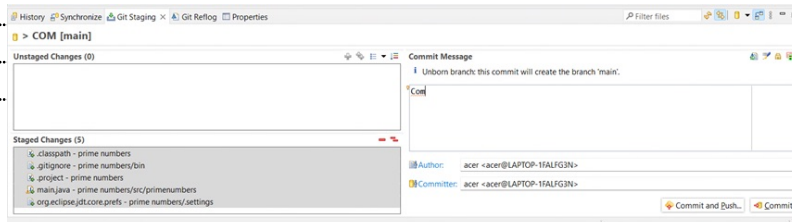
คือการเก็บสำเนาข้อมูลและการเปลี่ยนแปลงของ Source Code ทำให้สามารถย้อนกลับไปเวอร์ชันใดๆ ก่อนหน้า และดูรายละเอียดการเปลี่ยนแปลงของแต่ละเวอร์ชันได้ นอกจากนั้นยังสามารถดูได้ว่าใครเป็นคนแก้ไข

- 3.4. Clone คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

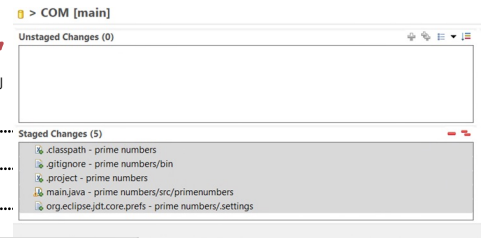
เวลาที่ผู้อ่านมี Repository อยู่บน Remote ชักหน่งอยู่แล้ว และต้องการ Sync มาลงเครื่องของเรา เราจะต้องทำสิ่งที่เรียกว่า Clone Repository หรือก็คือการก๊อป Repository จาก Remote คือ เวลาที่ผู้อ่านมี Repository อยู่บน Remote ชักหน่งอยู่แล้ว และต้องการ Sync มาลงเครื่องของเรา เราจะต้องทำสิ่งที่เรียกว่า Clone Repository หรือก็คือการก๊อป Repository จาก Remote

- 3.5. Commit คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ

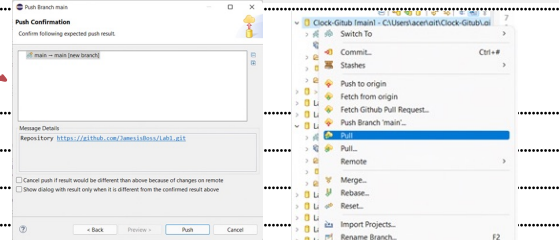
เวลาที่ข้อมูลที่เราแก้ไขเสร็จแล้ว (โค้ดที่เราเขียนคำสั่งบางอย่างเสร็จแล้ว) แล้วอยากจะทำ Backup เก็บไว้ใน VCS จะเรียกกันว่า Commit



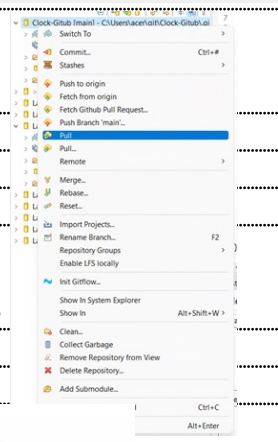
3.6. Staged และ Unstaged คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ
เวลาเราแก้ไขโค้ดหรือแก้ไขข้อมูล ไฟล์ที่ถูกแก้ไขจะอยู่ในสถานะ
Unstaged และเวลาที่เราทำอะไรเสร็จเรียบร้อย แล้วอยากจะทำ Commit เก็บไว้ จะ
ต้องเลือกไฟล์ที่ต้องการเพื่อย้ายเข้าสู่ในสถานะ Staged ก่อนถึงจะทำการ Commit
ได้



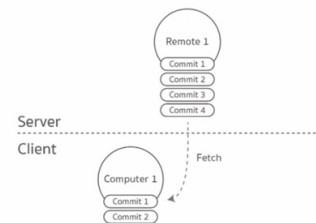
3.7. Push คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ
Push คือการนำโค้ดหรือไฟล์เข้าตัวระบบ Git Repository



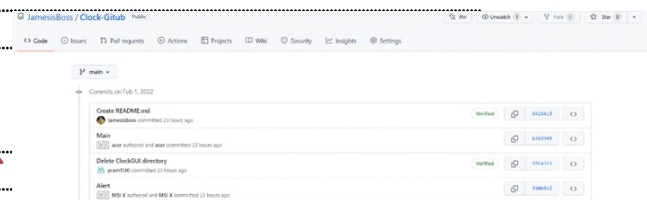
3.8. Pull คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ
คือ เวลา Sync จาก Remote เพื่อดึงข้อมูล Commit ใหม่ๆ ลงมาเก็บไว้ในเครื่องจะเรียกขั้นตอน
นี้ว่า Pull



3.9. Fetch คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ
อยากเช็คสถานะของ Remote เลยว่ามีใคร Push ข้อมูลใหม่
ขึ้นไปที่ Remote หรือป่าว เราเรียกวิธีนี้ว่า Fetch



3.10. Conflict ใน VSC คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ
คือ การเกิดปัญหาการชนของข้อมูลในไฟล์งานที่ทำร่วมกันกับเพื่อนเรา ซึ่งในช่องที่เราพัฒนาโปรแกรมหรือเขียนโค้ดกับเพื่อน
ร่วมงานอยู่นั้นเราไม่สามารถรู้ได้เลยว่าเพื่อนเราจะเขียนโค้ดไปในรูปแบบไหน



3.11. Merge Commit คืออะไร? อธิบายพร้อมยกตัวอย่างประกอบ
คือการที่มีการแตก branch ออกไป develop แยกกัน โดยที่
การแก้ไขไฟล์เดียวกันซึ่งโค้ดนั้นอาจมีการทับซ้อน หรืออยู่บรรทัด
เดียวกัน เมื่อใครคนใดคนหนึ่งนำโค้ดมา Merge รวมกันนั้นจะเกิด
สิ่งที่เรียกว่า 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 <https://download.eclipse.org/egit/updates> ลงในช่อง URL แล้วคลิกที่ Go
3. หลังจากนั้นกด Next แล้วรออาจใช้เวลาไม่นาน รอจนกว่าตัวโปรแกรมขึ้นให้ restart แล้วเปิดโปรแกรมใหม่หลังจากนั้นก็
สามารถใช้ส่วนของ Git ได้เลย

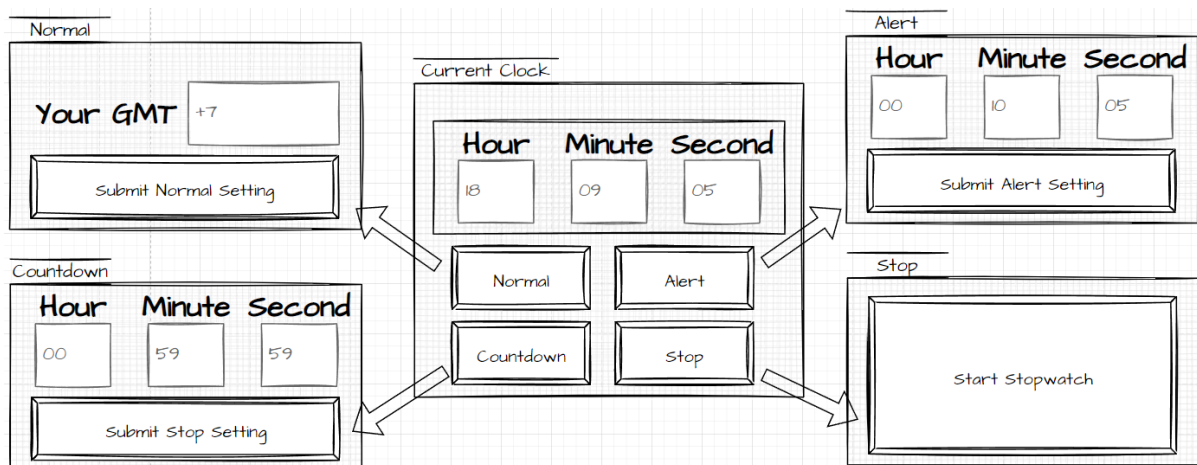
4. ลำดับขั้นการปฏิบัติการ

- 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 ->
ลิงค์การทดลองที่ 2 ->
ลิงค์การทดลองที่ 3 ->
ลิงค์การทดลองที่ 4 ->
ลิงค์การทดลองที่ 5 ->
ลิงค์การทดลองที่ 6 ->
ลิงค์การทดลองที่ 7 ->
ลิงค์การทดลองที่ 8 ->

- 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” ที่มีส่วนประกอบของฟิจเจอร์ต่างๆ 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 ก็จะทำให้การเริ่มต้นนับถอยหลังไปเรื่อยๆ 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 ก็จะทำให้เวลาตามปกติ แต่เมื่อถึงเวลาที่ตั้งปลุกเอาไว้ ระบบก็จะปรากฏหน้าต่างแจ้งเตือน
- 4.6.5. (หากมีสมาชิกในกลุ่มไม่ถึง 4 คน ไม่ต้องทำฟิจเจอร์นี้) หน้าต่าง Stop จะปรากฏหน้าต่างนี้เมื่อคลิกปุ่ม Stop ที่อยู่ในหน้า Current Clock ซึ่งจะ แสดงส่วนการตั้งค่าการจับเวลา หลังจากกดปุ่ม Start Stopwatch เรียบร้อย หน้าต่างการตั้งค่าจะหายไป และส่วนการแสดงนาฬิกาใน Current Clock ก็จะทำให้เริ่มจับเวลา โดยเริ่มตั้งแต่ 0 นาฬิกา 0 นาที 0 วินาที และ

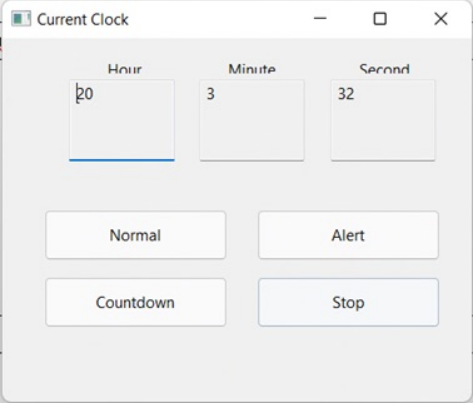
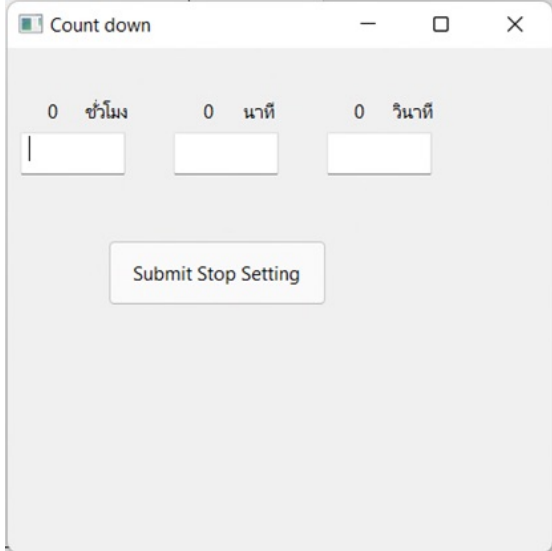
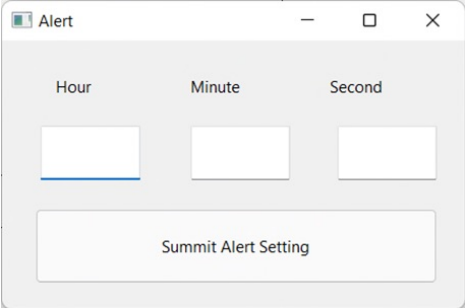
จำนวนวินาทีจะเริ่มต้นเพิ่มขึ้นไปเรื่อยๆ จนกว่าผู้ใช้งานจะกดปุ่ม Stop อีกครั้ง เพื่อเป็นการหยุดการทำงานของนาฬิกา
จับเวลา

- 4.7. จากฟีดแบจการทำงานของนาฬิกาข้างต้น 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 .บุษราภรณ์ พันธุ์ชาติ	รหัสนักศึกษา	62543502008-0
	หน้าที่รับผิดชอบ		
คนที่ 2	ชื่อ-นามสกุล	รหัสนักศึกษา
	หน้าที่รับผิดชอบ.....		
คนที่ 3	ชื่อ-นามสกุล	รหัสนักศึกษา
	หน้าที่รับผิดชอบ		
คนที่ 4	ชื่อ-นามสกุล	รหัสนักศึกษา
(ถ้ามี)	หน้าที่รับผิดชอบ		

ลิงค์งานกลุ่มของคุณที่อยู่ใน Github

ผลลัพธ์การทำงานของโปรแกรม

โค้ดโปรแกรมภายในหน้าต่าง Current Clock

```
1 import org.eclipse.swt.widgets.Display;
2 import org.eclipse.swt.widgets.Shell;
3 import org.eclipse.swt.widgets.Composite;
4 import org.eclipse.swt.SWT;
5 import org.eclipse.swt.widgets.Button;
6 import org.eclipse.swt.widgets.Label;
7 import org.eclipse.swt.widgets.Text;
8 import org.eclipse.swt.events.SelectionAdapter;
9 import org.eclipse.swt.events.SelectionEvent;
10 import org.eclipse.swt.widgets.DateTime;
11 import org.eclipse.swt.SWTResourceManager;
12
13 import javax.swing.*;
14 import java.awt.*;
15 import java.text.SimpleDateFormat;
16 import java.time.LocalDateTime;
17 import java.time.format.DateTimeFormatter;
18 import java.util.Calendar;
19 import java.util.GregorianCalendar;
20 import java.text.SimpleDateFormat;
21 import org.eclipse.ui.forms.widgets.FormToolkit;
22
23 public class Main1 {
24
25     protected Shell shell;
26
27
28     SimpleDateFormat timeFormat;
29     private String JH;
30     private String JM;
31     private String JS;
32     private Text Hour;
33     private int sec ;
34     private int minute ;
35     private int hour ;
36     public int Gmt = 0;
37     public int r = 0;
38     public int ah = 0;
39     public int am = 0;
40     public int as = 0;
41     Normal Nm = new Normal();
42     Alert A1 = new Alert();
43     Alert2 A12 = new Alert2();
44     Countdown Cd = new Countdown();
45     private Text Min;
46     private Text Sec;
47
48     /**
49      * Launch the application.
50      * @param args
51      */
52
53     public static void main(String[] args) {
```

```
56         try {
57             Main1 window = new Main1();
58             window.open();
59
60         } catch (Exception e) {
61             e.printStackTrace();
62         }
63     }
64
65
66
67
68     /**
69      * Open the window.
70      */
71
72     public void open() {
73         Display display = Display.getDefault();
74         setTime();
75         createContents();
76         shell.open();
77         shell.layout();
78
79         while (!shell.isDisposed()) {
80
81             if (!display.readAndDispatch()) {
82                 display.sleep();
83             }
84         }
85     }
86
87
88
89
90     /**
91      * Create contents of the window.
92      */
93
94     protected void createContents() {
95         shell = new Shell();
96         shell.setSize(473, 327);
97         shell.setText("Current Clock");
98
99         //setTime();
100         Composite composite = new Composite(shell, SWT.NONE);
101         composite.setBounds(30, 10, 415, 136);
102
103         Label lblH = new Label(composite, SWT.NONE);
104         lblH.setBounds(72, 10, 59, 14);
105         lblH.setText("Hour");
106
107         Label lblM = new Label(composite, SWT.NONE);
108         lblM.setBounds(189, 10, 59, 14);
109         lblM.setText("Minute");
110     }
```



```

112 Label lblS = new Label(composite, SWT.NONE);
113 lblS.setFont(SWTResourceManager.getFont(".AppleSystemUIFont", 11, SWT.NORMAL));
114 lblS.setBounds(316, 10, 59, 14);
115 lblS.setText("Second");
116
117 Hour = new Text(composite, SWT.BORDER);
118 Hour.setFont(SWTResourceManager.getFont(".AppleSystemUIFont", 30, SWT.NORMAL));
119 Hour.setEditable(false);
120 Hour.setBounds(35, 30, 102, 79);
121
122 Min = new Text(composite, SWT.BORDER);
123 Min.setFont(SWTResourceManager.getFont(".AppleSystemUIFont", 30, SWT.NORMAL));
124 Min.setEditable(false);
125 Min.setBounds(161, 30, 102, 79);
126
127 Sec = new Text(composite, SWT.BORDER);
128 Sec.setFont(SWTResourceManager.getFont(".AppleSystemUIFont", 30, SWT.NORMAL));
129 Sec.setEditable(false);
130 Sec.setBounds(288, 30, 102, 79);
131 //formToolkit.adapt(text, true, true);
132
133 //test Fetch
134
135 Button btnNewButton = new Button(shell, SWT.NONE);
136 btnNewButton.addSelectionListener(new SelectionAdapter() {
137     public void widgetSelected(SelectionEvent e) {
138         Nm.open();
139
140         Gmt = Nm.getGMT();
141         if(Gmt >=24) {
142             Gmt = Gmt - 24;
143         }
144     }
145 });
146
147 btnNewButton.setBounds(41, 166, 177, 49);
148 btnNewButton.setText("Normal");
149
150 Button btnNewButton_1 = new Button(shell, SWT.NONE);
151 btnNewButton_1.addSelectionListener(new SelectionAdapter() {
152     @Override
153     public void widgetSelected(SelectionEvent e) {
154         Al.open();
155         setAlert();
156     }
157 });
158 btnNewButton_1.setBounds(247, 166, 177, 49);
159 btnNewButton_1.setText("Alert");
160
161 Button btnNewButton_2 = new Button(shell, SWT.NONE);
162 btnNewButton_2.addSelectionListener(new SelectionAdapter() {
163     @Override
164     public void widgetSelected(SelectionEvent e) {
165         Cd.open();
166     }
167 });
168 btnNewButton_2.setBounds(41, 231, 177, 49);
169 btnNewButton_2.setText("Countdown");
170
171 Button btnNewButton_3 = new Button(shell, SWT.NONE);
172 btnNewButton_3.addSelectionListener(new SelectionAdapter() {
173     @Override
174     public void widgetSelected(SelectionEvent e) {
175     }
176 });
177 btnNewButton_3.setBounds(247, 231, 177, 49);
178 btnNewButton_3.setText("Stop");
179
180
181
182
183
184
185
186
187
188

```

```

190 public void setTime() {
191     new Thread(new Runnable() {
192         public void run() {
193             while (true) {
194                 try { Thread.sleep(1000); } catch (Exception e) {}
195                 Display.getDefault().asyncExec(new Runnable() {
196                     public void run() {
197                         Calendar cal = new GregorianCalendar();
198                         minute = cal.get(Calendar.MINUTE);
199                         hour = cal.get(Calendar.HOUR_OF_DAY);
200                         sec = cal.get(Calendar.SECOND);
201
202                         hour = hour+Gmt;
203
204                         if(hour >=24) {
205                             hour = hour- 24;
206                         }
207
208                         if(hour == ah && minute == am && sec == as) {
209                             Al2.open();
210                         }
211
212                         JH = ""+hour;
213                         JM = ""+minute;
214                         JS = ""+sec;
215                         Hour.setText(JH);
216                         Min.setText(JM);
217                         Sec.setText(JS);
218                     }
219                 });
220             }
221         }
222     }).start();
223 }
224
225 public void setAlert() {
226     ah = Al.h ;
227     am = Al.m ;
228     as = Al.s;
229 }
230
231
232
233
234
235
236
237
238

```

โค้ดโปรแกรมภายในหน้าต่าง Normal

```

1  import org.eclipse.swt.widgets.Display;
2  import org.eclipse.swt.widgets.Shell;
3  import org.eclipse.swt.widgets.Label;
4  import org.eclipse.swt.SWT;
5  import org.eclipse.wb.swt.SWTResourceManager;
6
7  import org.eclipse.swt.widgets.Text;
8  import org.eclipse.swt.widgets.Button;
9  import org.eclipse.swt.events.SelectionAdapter;
10 import org.eclipse.swt.events.SelectionEvent;
11
12 public class Normal {
13
14     protected Shell shell;
15     private Text text;
16     public int GMT = 0;
17
18     /**
19      * Launch the application.
20      * @param args
21      */
22     public static void main(String[] args) {
23         try {
24             Normal window = new Normal();
25             window.open();
26         } catch (Exception e) {
27             e.printStackTrace();
28         }
29     }
30
31     /**
32      * Open the window.
33      */
34     public void open() {
35         Display display = Display.getDefault();
36         createContents();
37         shell.open();
38         shell.layout();
39         while (!shell.isDisposed()) {
40             if (!display.readAndDispatch()) {
41                 display.sleep();
42             }
43         }
44     }
45
46     /**
47      * Create contents of the window.
48      */
49     protected void createContents() {
50         shell = new Shell();
51         shell.setSize(450, 300);
52         shell.setText("Normal");
53         Main1 window1 = new Main1();
54
55         Label lblYourGmt = new Label(shell, SWT.NONE);
56         lblYourGmt.setFont(SWTResourceManager.getFont(".AppleSystemUIFont", 44, SWT.NORMAL));
57         lblYourGmt.setBounds(10, 76, 200, 84);
58         lblYourGmt.setText("Your GMT");
59
60         text = new Text(shell, SWT.BORDER);
61         text.setBounds(206, 68, 220, 77);
62
63         Button btnNewButton = new Button(shell, SWT.NONE);
64         btnNewButton.addSelectionListener(new SelectionAdapter() {
65             @Override
66             public void widgetSelected(SelectionEvent e) {
67
68                 GMT = Integer.parseInt(text.getText());
69                 window1.r = GMT;
70                 shell.close();
71             }
72         });
73         btnNewButton.setBounds(20, 161, 406, 77);
74         btnNewButton.setText("Summit Normal Setting");
75     }
76
77     public int getGMT() {
78
79         return this.GMT ;
80     } //end method
81 }

```

```

1  import org.eclipse.swt.widgets.Display;
2  import org.eclipse.swt.widgets.Shell;
3
4  import org.eclipse.swt.SWT;
5  import org.eclipse.swt.widgets.Text;
6  import org.eclipse.swt.widgets.Button;
7  import org.eclipse.swt.events.SelectionAdapter;
8  import org.eclipse.swt.events.SelectionEvent;
9
10 import javax.swing.Timer;
11 import org.eclipse.swt.widgets.Label;
12 import org.eclipse.swt.SWTResourceManager;
13 public class Countdown {
14
15     protected Shell shell;
16     private Text text1;
17     private Text text2;
18     private Text text3;
19     Timer timer;
20     int sec,min,hour;
21     int usesec1,usesec2,usesec3;
22     String sec1= "";
23     String min1= "";
24     String hour1= "";
25     private Label la3;
26     private Label la2;
27     private Label la1;
28     private Label lblNewLabel;
29     private Label lblNewLabel_1;
30     private Label lblNewLabel_2;
31     /**
32      * Launch the application.
33      * @param args
34      */
35     public static void main(String[] args) {
36         try {
37             Countdown window = new Countdown();
38             window.open();
39         } catch (Exception e) {
40             e.printStackTrace();
41         }
42     }
43
44     /**
45      * Open the window.
46      */
47     public void open() {
48         Display display = Display.getDefault();
49         createContents();
50         shell.open();
51         shell.layout();
52         while (!shell.isDisposed()) {
53             if (!display.readAndDispatch()) {
54                 display.sleep();
55             }
56         }
57     }
58
59     /**
60      * Create contents of the window.
61      */
62     protected void createContents() {
63         shell = new Shell();
64         shell.setSize(364, 275);
65         shell.setText("Count down");
66
67         text1 = new Text(shell, SWT.BORDER);
68         text1.setBounds(10, 67, 84, 34);

```

```

71         text2 = new Text(shell, SWT.BORDER);
72         text2.setBounds(132, 67, 84, 34);
73
74         text3 = new Text(shell, SWT.BORDER);
75         text3.setBounds(254, 67, 84, 34);
76
77         Button button1 = new Button(shell, SWT.NONE);
78         button1.addSelectionListener(new SelectionAdapter() {
79             @Override
80             public void widgetSelected(SelectionEvent e) {
81                 hour1 = text1.getText();
82                 la1.setText(hour1);
83                 usesec1 = Integer.parseInt(hour1);
84
85                 min1 = text2.getText();
86                 la2.setText(min1);
87                 usesec2 = Integer.parseInt(min1);
88
89                 sec1 = text3.getText();
90                 la3.setText(sec1);
91                 usesec3 = Integer.parseInt(sec1);
92
93                 new Thread(new Runnable() {
94                     public void run() {
95                         while (true) {
96                             try { Thread.sleep(1000); } catch (Exception e) {}
97                             Display.getDefault().asyncExec(new Runnable() {
98                                 public void run() {
99                                     if(usesec3 < 62){
100                                         usesec3--;
101                                         String usetime3 = Integer.toString(usesec3);
102                                         //OptionPane.showMessageDialog(null,"show number get: "+usetime1);
103                                         la3.setText(""+usetime3);
104                                         if(usesec3 == 0) {
105                                             usesec3 = 60;
106                                             if(usesec2 == 0) {
107                                                 usesec2 = 60;
108                                                 usesec1--;
109                                                 String usetime1 = Integer.toString(usesec1);
110                                                 la1.setText(""+usetime1);
111                                             }
112                                             usesec2--;
113                                             String usetime2 = Integer.toString(usesec2);
114                                             la2.setText(""+usetime2);
115                                         }
116                                     }
117                                 }
118                             }); //if(usesec1 == 0) {
119                                 // break;
120                                 //}
121                             }
122                         }
123                     }).start();
124                 });
125
126                 button1.setBounds(80, 153, 174, 52);
127                 button1.setText("Submit Stop Setting");
128
129                 la3 = new Label(shell, SWT.NONE);
130                 la3.setFont(SWTResourceManager.getFont("Segoe UI", 12, SWT.NORMAL));
131                 la3.setBounds(276, 40, 24, 21);
132                 la3.setText("0");
133
134                 la2 = new Label(shell, SWT.NONE);
135                 la2.setFont(SWTResourceManager.getFont("Segoe UI", 12, SWT.NORMAL));
136                 la2.setBounds(156, 40, 24, 21);
137                 la2.setText("0");
138
139                 la1 = new Label(shell, SWT.NONE);
140                 la1.setFont(SWTResourceManager.getFont("Segoe UI", 12, SWT.NORMAL));
141                 la1.setText("0");
142                 la1.setBounds(32, 40, 24, 21);
143
144                 lblNewLabel = new Label(shell, SWT.NONE);
145                 lblNewLabel.setFont(SWTResourceManager.getFont("Segoe UI", 12, SWT.NORMAL));
146                 lblNewLabel.setBounds(307, 40, 44, 21);
147                 lblNewLabel.setText("\u0E27\u0E34\u0E19\u0E32\u0E17\u0E35");
148
149                 lblNewLabel_1 = new Label(shell, SWT.NONE);
150                 lblNewLabel_1.setFont(SWTResourceManager.getFont("Segoe UI", 12, SWT.NORMAL));
151                 lblNewLabel_1.setText("\u0E19\u0E32\u0E17\u0E35");
152                 lblNewLabel_1.setBounds(188, 40, 39, 21);
153
154                 lblNewLabel_2 = new Label(shell, SWT.NONE);
155                 lblNewLabel_2.setFont(SWTResourceManager.getFont("Segoe UI", 12, SWT.NORMAL));
156                 lblNewLabel_2.setText("\u0E0A\u0E31\u0E48\u0E27\u0E42\u0E21\u0E07");
157                 lblNewLabel_2.setBounds(62, 40, 44, 21);
158
159             }
160         }
161     }

```



```

1 import org.eclipse.swt.widgets.Display;
2 import org.eclipse.swt.widgets.Shell;
3 import org.eclipse.swt.widgets.Label;
4 import org.eclipse.swt.SWT;
5 import org.eclipse.swt.widgets.Text;
6 import org.eclipse.swt.widgets.Button;
7 import org.eclipse.swt.widgets.SWTResourceManager;
8 import org.eclipse.swt.events.SelectionAdapter;
9 import org.eclipse.swt.events.SelectionEvent;
10
11 public class Alert {
12
13     protected Shell shell;
14     private Text H;
15     private Text M;
16     private Text S;
17     public int h = 0;
18     public int m = 0;
19     public int s = 0;
20
21     /**
22      * Launch the application.
23      * @param args
24      */
25     public static void main(String[] args) {
26         try {
27             Alert window = new Alert();
28             window.open();
29         } catch (Exception e) {
30             e.printStackTrace();
31         }
32     }
33
34     /**
35      * Open the window.
36      */
37     public void open() {
38         Display display = Display.getDefault();
39         createContents();
40         shell.open();
41         shell.layout();
42         while (!shell.isDisposed()) {
43             if (!display.readAndDispatch()) {
44                 display.sleep();
45             }
46         }
47     }
48
49     /**
50      * Create contents of the window.
51      */
52     protected void createContents() {
53         shell = new Shell();
54         shell.setSize(450, 300);
55         shell.setText("Alert");
56         Main1 M1 = new Main1();

```

```

58         Label lblNewLabel = new Label(shell, SWT.NONE);
59         lblNewLabel.setFont(SWTResourceManager.getFont(".AppleSystemUIFont", 30, SWT.NORMAL));
60         lblNewLabel.setBounds(50, 33, 73, 30);
61         lblNewLabel.setText("Hour");
62
63         H = new Text(shell, SWT.BORDER);
64         H.setBounds(36, 80, 93, 51);
65
66         Button btnNewButton = new Button(shell, SWT.NONE);
67         btnNewButton.addSelectionListener(new SelectionAdapter() {
68             @Override
69             public void widgetSelected(SelectionEvent e) {
70
71                 h = Integer.parseInt(H.getText());
72                 m = Integer.parseInt(M.getText());
73                 s = Integer.parseInt(S.getText());
74
75                 M1.ah = h;
76                 M1.am = m;
77                 M1.as = s;
78
79                 shell.close();
80             }
81         });
82
83         btnNewButton.setFont(SWTResourceManager.getFont(".AppleSystemUIFont", 21, SWT.NORMAL));
84         btnNewButton.setBounds(31, 158, 375, 70);
85         btnNewButton.setText("Summit Alert Setting");
86
87         M = new Text(shell, SWT.BORDER);
88         M.setBounds(176, 80, 93, 51);
89
90         S = new Text(shell, SWT.BORDER);
91         S.setBounds(313, 80, 93, 51);
92
93         Label lblMinute = new Label(shell, SWT.NONE);
94         lblMinute.setText("Minute");
95         lblMinute.setFont(SWTResourceManager.getFont(".AppleSystemUIFont", 30, SWT.NORMAL));
96         lblMinute.setBounds(176, 33, 93, 30);
97
98         Label lblNewLabel_3_1 = new Label(shell, SWT.NONE);
99         lblNewLabel_3_1.setText("Second");
100         lblNewLabel_3_1.setFont(SWTResourceManager.getFont(".AppleSystemUIFont", 30, SWT.NORMAL));
101         lblNewLabel_3_1.setBounds(306, 33, 110, 30);
102
103     }
104 }
105

```

โค้ดโปรแกรมภายในหน้าต่าง Stop

```
1 public void widgetSelected(SelectionEvent e) {
2
3     Nm.open();
4
5     Gmt = Nm.getGMT();
6     if(Gmt >=24) {
7         Gmt = Gmt - 24;
8     }
9 }
10 });
11 btnNewButton.setBounds(41, 166, 177, 49);
12 btnNewButton.setText("Normal");
13
14 Button btnNewButton_1 = new Button(shell, SWT.NONE);
15 btnNewButton_1.addSelectionListener(new SelectionAdapter() {
16 @Override
17 public void widgetSelected(SelectionEvent e) {
18
19     Al.open();
20     setAlert();
21
22 }
23 });
24 btnNewButton_1.setBounds(247, 166, 177, 49);
25 btnNewButton_1.setText("Alert");
26 Button btnNewButton_2 = new Button(shell, SWT.NONE);
27 btnNewButton_2.addSelectionListener(new SelectionAdapter() {
28 @Override
29 public void widgetSelected(SelectionEvent e) {
30
31     Cd.open();
32
33 }
34 });
35 btnNewButton_2.setBounds(41, 231, 177, 49);
36 btnNewButton_2.setText("Countdown");
37
38 Button btnNewButton_3 = new Button(shell, SWT.NONE);
39 btnNewButton_3.addSelectionListener(new SelectionAdapter() {
40 @Override
41 public void widgetSelected(SelectionEvent e) {
42 }
43 });
44 btnNewButton_3.setBounds(247, 231, 177, 49);
45 btnNewButton_3.setText("Stop");
46
47 }
48
49 }
```

5. สรุปผลการปฏิบัติการ

สามารถใช้งาน Commit จาก eclipse ไปยัง Github แต่ติดปัญหาเรื่อง Merge งานเพื่อนชิ้นมาบนเครื่องเราแล้วติดปัญหา แต่ก็สามารถ Merge มาทำได้ ศึกษาจากคลิปอินเดีย สามารถทำตัวโปรแกรมของเรื่อง Time แต่ติดปัญหาเรื่องของ Start Stop GMT และ Time ติด Bug ของการเริ่มแล้วไม่หยุดไม่ได้

6. คำถามท้ายการทดลอง

6.1. ควร Commit อย่างไร เพื่อหลีกเลี่ยงการเกิด Conflict ให้เหมาะสมที่สุด

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

6.2. ควรมีหลักเกณฑ์ในการ Push ขึ้นไปบน Remote เมื่อใดจึงจะเหมาะสมที่สุด

6.3. เมื่อใดจึงควรใช้คำสั่ง Fetch

เลือกไฟล์ที่ต้องอัปเดต git แล้วหลังจากนั้นค่อย Share project เสร็จเช็คก่อนที่จะไป staged changes

ขั้นตอนต่อไปทำการ commit ก่อนแล้วค่อยไป push and commit

6.4. เราควรแยก Branch เมื่อใด? และควร Merge Branch เมื่อใด?

เมื่อเราจะอัปเดตไฟล์ไปใน Git Branch เพราะเราต่างคนต่างทำโค้ดอาจทำให้เวลาทำ Feature อาจไม่รู้ว่าเป็นของหรืออาจทำให้ Source Code รวมอยู่ในไฟล์เดียวกันได้ เราควร Merge Branch เมื่อมีการจะ push โค้ดเข้าในตัวของ git hub