AlarmClockSystem

應用目錄

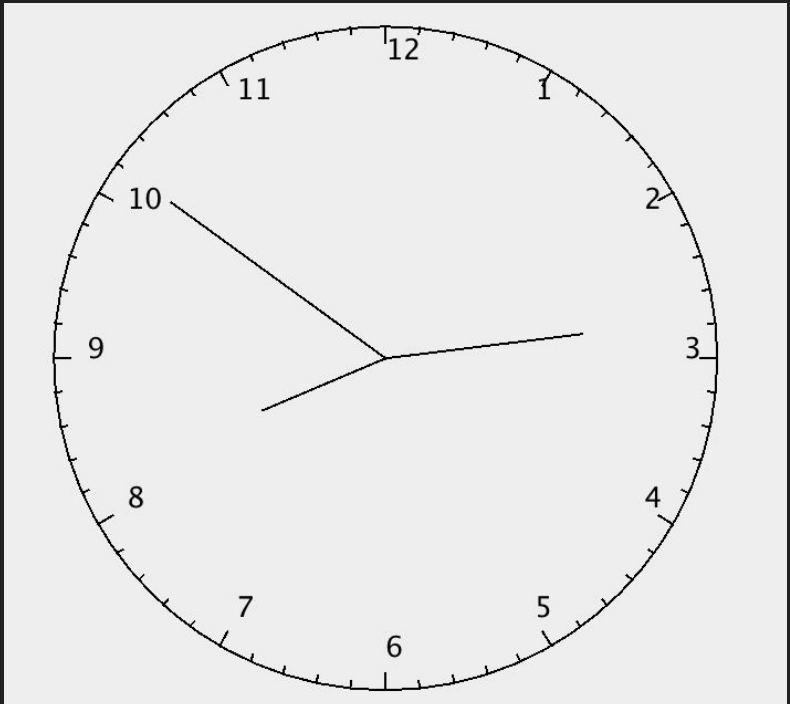
1. **swing**: BorderFactory, JButton, JCheckBox, JComboBox, JFrame, JInternalFrame, JLabel, JMenu, JMenuBar, JMenuItem, JOptionPane, JPanel, JTextField, Timer, PlainDocument, AttributeSet
2. **awt**: Component, Graphics, Graphics2D, GridLayout, ActionEvent, ActionListener, Ellipse2D, Line2D, Point2D
3. **sound**: AudioInputStream, AudioSystem, Clip
4. **sql**: Connection, DriverManager, PreparedStatement, ResultSet
5. **other**: File, ArrayList, Calendar

主要架構

- AlarmClock
  - ClockFrame
    - ClockPanel
    - AlarmClockItem
    - AlarmClockItemPanel
    - AlarmClockItemThread
    - AlarmClockItemPostgreSQL
    - resetAlarmClockItemPanel()
    - checkAlarmClockItem()
    - AddListener
    - ExitListener
    - AboutListener

- 整個專案主程式
  - 最主要框架
    - 顯示時鐘
    - 鬧鐘物件
    - 鬧鐘面板
    - 鬧鐘響執行緒
    - 鬧鐘物件連接資料庫
    - 重整鬧鐘物件()
    - 檢查鬧鐘響否()
    - 顯示新增鬧鐘視窗
    - 離開程式
    - 顯示關於視窗

ClockPanel




```
private final int radius = 150;
private final double centerX = 250;
private final double centerY = 200;
private final int clockBlock = 12 * 60 * 60;
private final Point2D.Double[] pointList = new Point2D.Double[this.clockBlock];
```

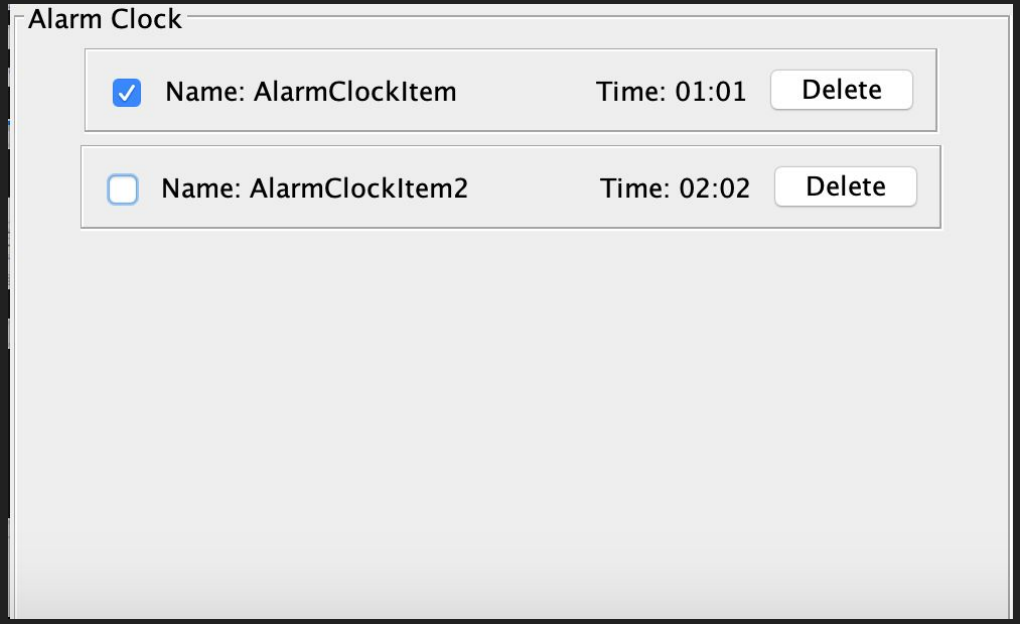
```
for (int i = 0; i < this.clockBlock; i++) {
    double x = Math.cos(Math.PI * 2 * i / this.clockBlock) * this.radius;
    double y = Math.sin(Math.PI * 2 * i / this.clockBlock) * this.radius;
    this.pointList[i] = new Point2D.Double(x, y);
}
```

```
private double reduce(double a, double b, double r) {
    return a + b * r;
}
```

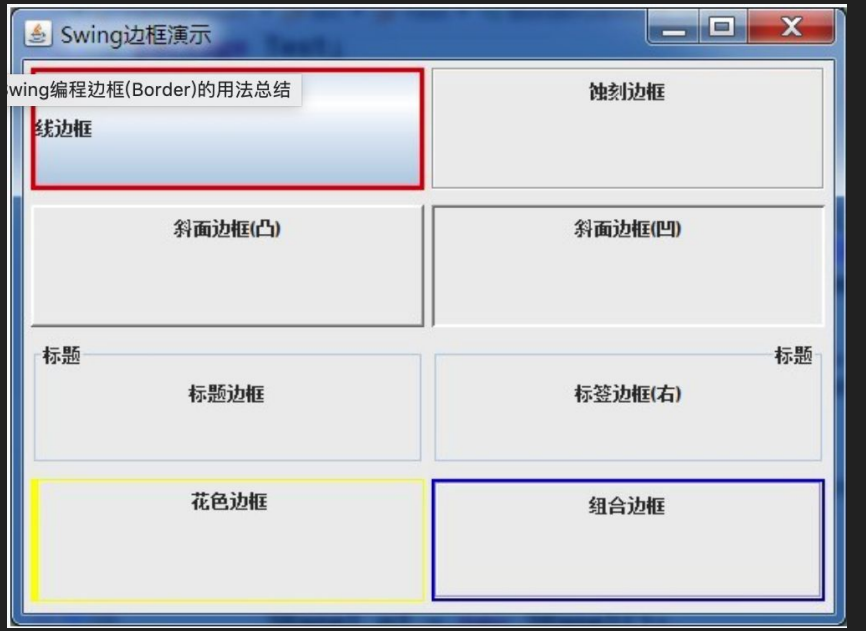
ClockPanel

```
@Override
public void paintComponent(Graphics graphics) {
    super.paintComponent(graphics);
    Graphics2D graphics2D = (Graphics2D) graphics;
    for (int minuteNumber = 1; minuteNumber <= 60; minuteNumber++) {
        if (minuteNumber % 5 == 0) {
            int hourNumber = minuteNumber / 5;
            int numberIndex = (hourNumber * 60 * 60 + 3 * this.clockBlock / 4) % this.clockBlock;
            Point2D.Double numberPoint = this.pointList[numberIndex];
            graphics2D.drawString(String.valueOf(hourNumber), (float) this.reduce(this.centerX, numberPoint.x, .9), (float) this.reduce(this.centerY, numberPoint.y, .9));
            graphics2D.draw(new Line2D.Double(this.centerX + numberPoint.x, this.centerY + numberPoint.y, this.reduce(this.centerX, numberPoint.x, .95), this.reduce(this.centerY, numberPoint.y, .95)
        } else {
            int numberIndex = (minuteNumber * 12 * 60 + 3 * this.clockBlock / 4) % this.clockBlock;
            Point2D.Double numberPoint = this.pointList[numberIndex];
            graphics2D.draw(new Line2D.Double(this.centerX + numberPoint.x, this.centerY + numberPoint.y, this.reduce(this.centerX, numberPoint.x, .98), this.reduce(this.centerY, numberPoint.y, .98)
        }
    }
    int hourIndex = (this.hour * 60 * 60 + this.minute * 60 + this.second + 3 * this.clockBlock / 4) % this.clockBlock;
    int minuteIndex = (12 * this.minute * 60 + 12 * this.second + 3 * this.clockBlock / 4) % this.clockBlock;
    int secondIndex = (12 * 60 * this.second + 3 * this.clockBlock / 4) % this.clockBlock;
    Point2D.Double hourPoint = this.pointList[hourIndex];
    Point2D.Double minutePoint = this.pointList[minuteIndex];
    Point2D.Double secondPoint = this.pointList[secondIndex];
    graphics2D.draw(new Ellipse2D.Double(this.centerX - this.radius, this.centerY - this.radius, 2 * this.radius, 2 * this.radius));
    graphics2D.draw(new Line2D.Double(this.centerX, this.centerY, this.reduce(this.centerX, hourPoint.x, .4), this.reduce(this.centerY, hourPoint.y, .4)));
    graphics2D.draw(new Line2D.Double(this.centerX, this.centerY, this.reduce(this.centerX, minutePoint.x, .6), this.reduce(this.centerY, minutePoint.y, .6)));
    graphics2D.draw(new Line2D.Double(this.centerX, this.centerY, this.reduce(this.centerX, secondPoint.x, .8), this.reduce(this.centerY, secondPoint.y, .8)));
}
```

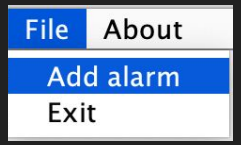
AlarmClockPanel / AlarmClockItemPanel




```
public AlarmClockItemPanel(int ID, AlarmClockItem alarmClockItem) {
    this.setSize(400, 100);
    this.setBorder(BorderFactory.createEtchedBorder());
```



AddAlarmClockItem




```
private class AddListener implements ActionListener {

    private AddClockInternalFrame addClockInternalFrame;

    private class AddClockInternalFrame extends JInternalFrame {...96 lines }

    @Override
    public void actionPerformed(ActionEvent event) {
        this.addClockInternalFrame = new AddClockInternalFrame();
        this.addClockInternalFrame.setVisible(true);
        this.addClockInternalFrame.pack();
        clockPanel.add(this.addClockInternalFrame);
    }
}
```

```
this.nameTextField = new JTextField();
this.nameTextField.setDocument(new JTextFieldLimit(15));
this.nameTextField.setText("Alarm Message");
this.panel.add(this.nameTextField);
```

```
private class JTextFieldLimit extends PlainDocument {

    private final int limit;

    public JTextFieldLimit(int limit) {
        super();
        this.limit = limit;
    }

    @Override
    public void insertString(int offset, String string, AttributeSet attr) throws BadLocationException {
        if (string != null) {
            if ((this.getLength() + string.length()) <= this.limit) {
                super.insertString(offset, string, attr);
            }
        }
    }
}
```

AlarmClockItemThread

```
for (AlarmClockItem alarmClockItem : alarmClockItemPostgreSQL.getAllAlarmClockItem()) {
    if (alarmClockItem.getFlag() && alarmClockItem.hour == hour && alarmClockItem.minute == minute) {
        new AlarmClockItemThread(alarmClockItem.name).start();
    }
}
```

```
private class AlarmClockItemThread extends Thread {

    private final String name;

    public AlarmClockItemThread(String name) {
        this.name = name;
    }

    @Override
    public void run() {
        try {
            AudioInputStream audioInputStream = AudioSystem.getAudioInputStream(new File("ringing.wav"));
            Clip clip = AudioSystem.getClip();
            clip.open(audioInputStream);
            clip.start();
            JOptionPane.showMessageDialog(
                    ClockFrame.this,
                    String.format("Alarm Clock %s ringing", this.name),
                    "Alarm Clock",
                    JOptionPane.PLAIN_MESSAGE
            );
        } catch (IOException | LineUnavailableException | UnsupportedAudioFileException ex) {
            System.out.println(ex.toString());
        }
    }
}
```

AlarmClockItemPostgreSQL

```java
private class AlarmClockItemPostgreSQL {

    private final Connection connection;
    private final String DRIVER = "org.postgresql.Driver";
    private final String URL = "jdbc:postgresql://localhost:5432/AlarmClock";
    private final String USERNAME = "Killua4564";
    private final String PASSWORD =             ;

    public AlarmClockItemPostgreSQL() throws ClassNotFoundException, SQLException {
        Class.forName(this.DRIVER);
        this.connection = DriverManager.getConnection(this.URL, this.USERNAME, this.PASSWORD);
    }

    public int insertAlarmClockItem(String name, int hour, int minute) throws SQLException {...11 lines }

    public ArrayList<AlarmClockItem> getAllAlarmClockItem() throws SQLException {...16 lines }

    public int updateAlarmClockItem(int ID, boolean flag) throws SQLException {...9 lines }

    public int deleteAlarmClockItem(int ID) throws SQLException {...7 lines }
}
```

```
AlarmClock=# select * from "AlarmClockItem";
 AlarmClockItemID | AlarmClockItemHour | AlarmClockItemMinute | AlarmClockItemName | AlarmClockItemFlag
------------------+--------------------+----------------------+--------------------+--------------------
                3 |                  1 |                    1 | AlarmClockItem     | t
                4 |                  2 |                    2 | AlarmClockItem2    | f
(2 rows)
```

Java Compile & Run / AlarmClockItemInit.sql

```
ip149-149:final_project Killua4564$ ll
total 3392
drwxr-xr-x   7 Killua4564  staff     224 11 29 16:40 ./
drwxr-xr-x  21 Killua4564  staff     672 11 29 14:56 ../
-rw-r--r--   1 Killua4564  staff   23146 11 29 16:14 AlarmClock.java
-rwxr-xr-x@  1 Killua4564  staff      71 11 25 19:54 AlarmClock.sh*
-rw-r--r--@  1 Killua4564  staff     647 11 25 19:53 AlarmClockItemInit.sql
-rw-r--r--@  1 Killua4564  staff  825943 11 25 16:35 postgresql.jar
-rw-r--r--@  1 Killua4564  staff  772674 11 18 14:49 ringing.wav
ip149-149:final_project Killua4564$ javac AlarmClock.java ; java -cp .:./postgresql.jar AlarmClock ; rm *.class
```

```
CREATE DATABASE "AlarmClock";

GRANT ALL ON DATABASE "AlarmClock" TO "Killua4564";

DROP TABLE "AlarmClockItem";

CREATE TABLE "AlarmClockItem"
(
    "AlarmClockItemID" SERIAL PRIMARY KEY,
    "AlarmClockItemHour" INT NOT NULL DEFAULT 0,
    "AlarmClockItemMinute" INT NOT NULL DEFAULT 0,
    "AlarmClockItemName" VARCHAR (15) DEFAULT NULL,
    "AlarmClockItemFlag" BOOLEAN DEFAULT true
);

REVOKE ALL ON TABLE "AlarmClockItem" FROM "Killua4564";
GRANT ALL ON TABLE "AlarmClockItem" TO "Killua4564";

INSERT INTO "AlarmClockItem" ("AlarmClockItemHour", "AlarmClockItemMinute", "AlarmClockItemName", "AlarmClockItemFlag")
VALUES (3, 30, 'insert sql test', true);
```