โครงงานการเขียนโปรแกรมเชิงวัตถุ

จัดทำโดย

นางสาววรรณวิภา รุคะ รหัส 6306021620123

นางสาวณัฐชนก วิสุทธิพันธุ์ รหัส 6306021620131

นางสาวปณิชา ปัทมวราวงศ์ รหัส 6306021620166

นายสหรัถ ทองอินทร์ รหัส 6306021620191

เสนอ

ผู้ช่วยศาสตราจารย์สมชัย เชียงพงพันธุ์

รายงานฉบับนี้เป็นส่วนหนึ่งของวิชา 060243104 Object-oriented Programming

คณะเทคโนโลยีและการจัดการอุตสาหกรรม มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

ภาคการเรียนที่ 2 ปีการศึกษา 2563

คำนำ

โครงงานการเขียนโปรแกรมเชิงวัตถุนี้เป็นส่วนหนึ่งของวิชา 060243104 Object-oriented Programming โดยมีจุดประสงค์เพื่อให้ผู้อ่านได้ศึกษา เรียนรู้ เกี่ยวกับการเขียนโปรแกรมเชิงวัตถุ พร้อมทั้งศึกษาเกี่ยวกับ Class Diagram ที่ใช้แสดง Class และความสัมพันธ์ในแง่ต่าง ๆ ระหว่าง Class ของโปรแกรมในรายงานฉบับนี้

คณะผู้จัดทำต้องขอขอบคุณผู้ช่วยศาสตราจารย์สมชัย เชียงพงพันธุ์ ผู้ให้ความรู้ และแนวทางการศึกษา ทางคณะผู้จัดทำหวังเป็นอย่างยิ่งว่า ผู้อ่านจะได้รับประโยชน์จากรายงานฉบับนี้ หากเนื้อหารายงานเล่มนี้มีข้อผิดพลาดประการใด ทางผู้จัดทำขอน้อมรับไว้ทุกประการ

คณะผู้จัดทำ

สารบัญ

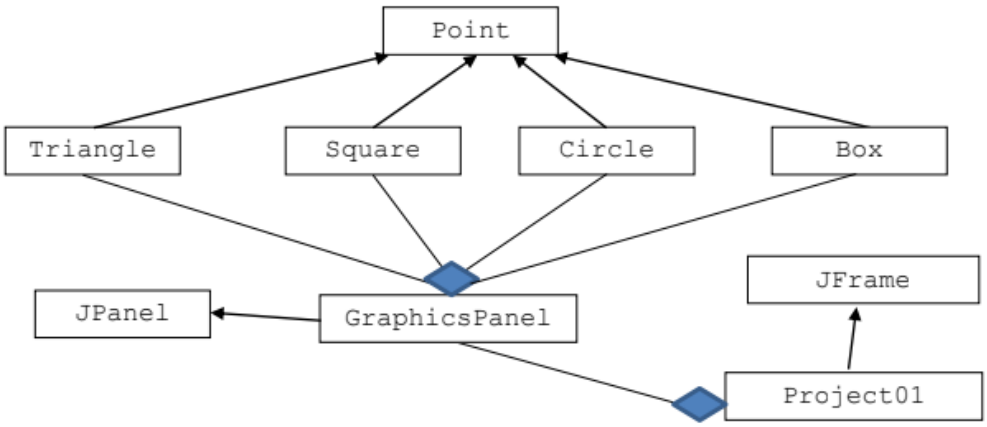
เนื้อหา

สารบัญรูป

โครงงานที่ 1

โปรแกรม Java Application สำหรับการวาดรูป

โครงงานที่ 1 เป็นโปรแกรม Java Application สำหรับการวาดรูป ขนาดหน้าจอ 1024x768 พิกเซล มีหน้าจอโปรแกรมดังรูปด้านล่างที่ใช้งาน JPanel มาออกแบบ โดยต้องมีการออกแบบคลาสมาจัดเก็บข้อมูลก่อน ซึ่งเป็นการออกแบบในลักษณะ Inheritance โดยมีคลาส Point เป็น Super Class ดังภาพด้านล่าง สำหรับการวาดรูปจะต้องสร้างเป็นเมธอดอยู่ในคลาสของแต่ละคลาส โปรแกรมสามารถใช้เมาส์กำหนดตำแหน่งแรกของรูปในการวาดได้

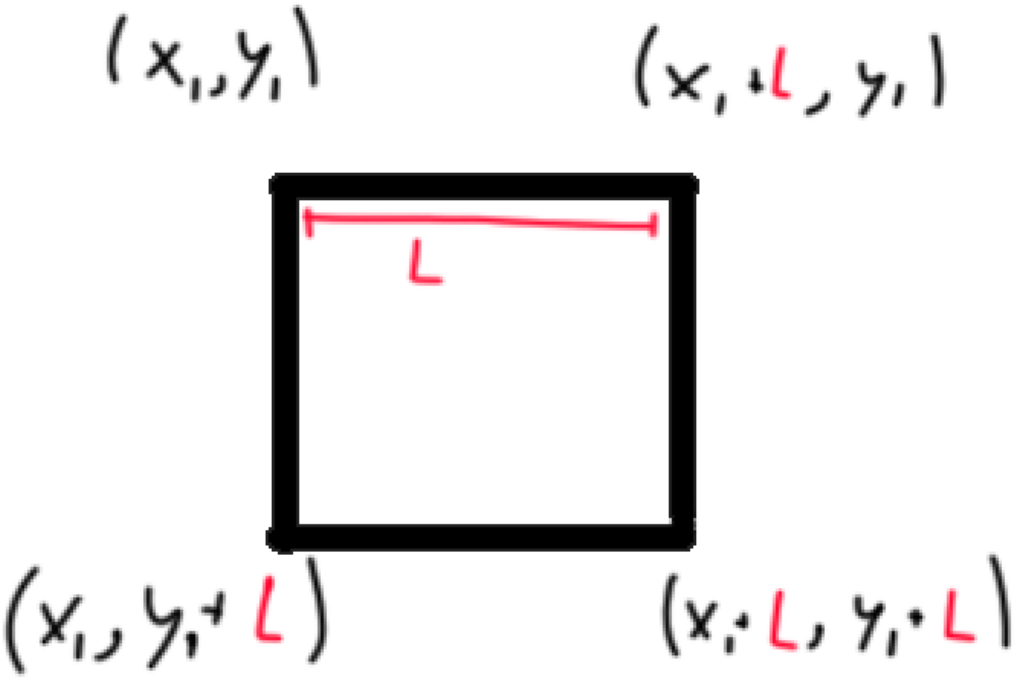


ภาพที่ 1 ตัวอย่างโปรแกรม Java Application สำหรับการวาดรูป

หลักการออกแบบ Class โปรแกรม Java Application สำหรับการวาดรูป

หลักการออกแบบ Class Square

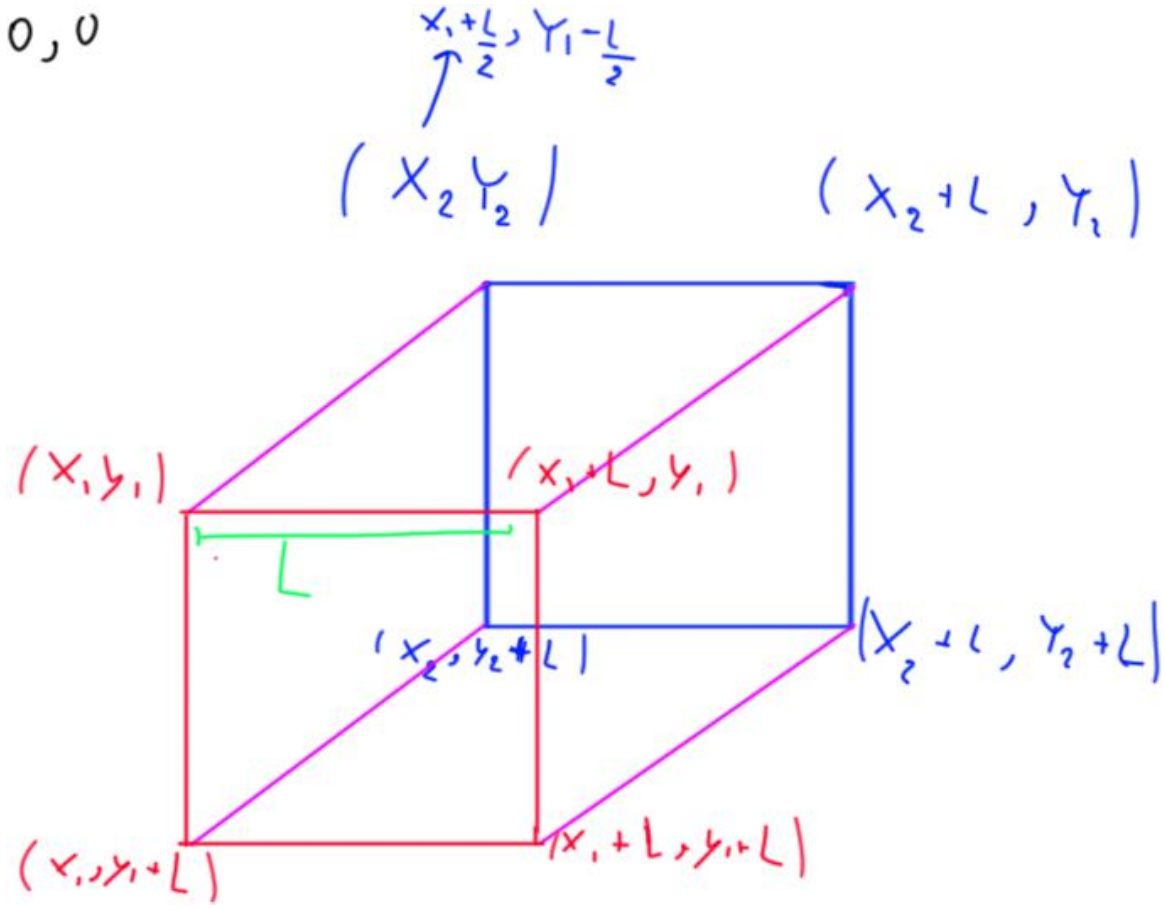
หลักการออกแบบการวาดรูปสีเหลี่ยมนั้นจะเป็นการกำหนดให้จุดที่เมาส์ผู้ใช้คลิกเป็นจุดที่ 1 (x1,y1) ส่วนระหว่างที่ผู้ใช้กดเมาส์ค้างไปจนถึงการปล่อยปุ่มจะเป็นจุดที่ 2 (x2,y2) จากนั้นจึงหาว่าความต่างของพิกัดของ x1 กับ x2 และ y1 กับ y2 นั้นเป็นเท่าไหร่ ให้เราเลือกความต่างที่มากที่สุด เพื่อนำมาใช้เป็นความยาวในการวาดรูปสี่เหลี่ยม แล้วก็กำหนดให้จุดสี่เหลี่ยมนั้นเริ่มวาดจากพิกัด x1,y1 โดยให้จุดนี้เป็นจุด ๆ หลัก ให้วาดออกมาสี่เส้นตามตัวอย่างรูปภาพ



ภาพที่ 2 รูปภาพประกอบการออกแบบ Class Square

หลักการออกแบบ Class Box

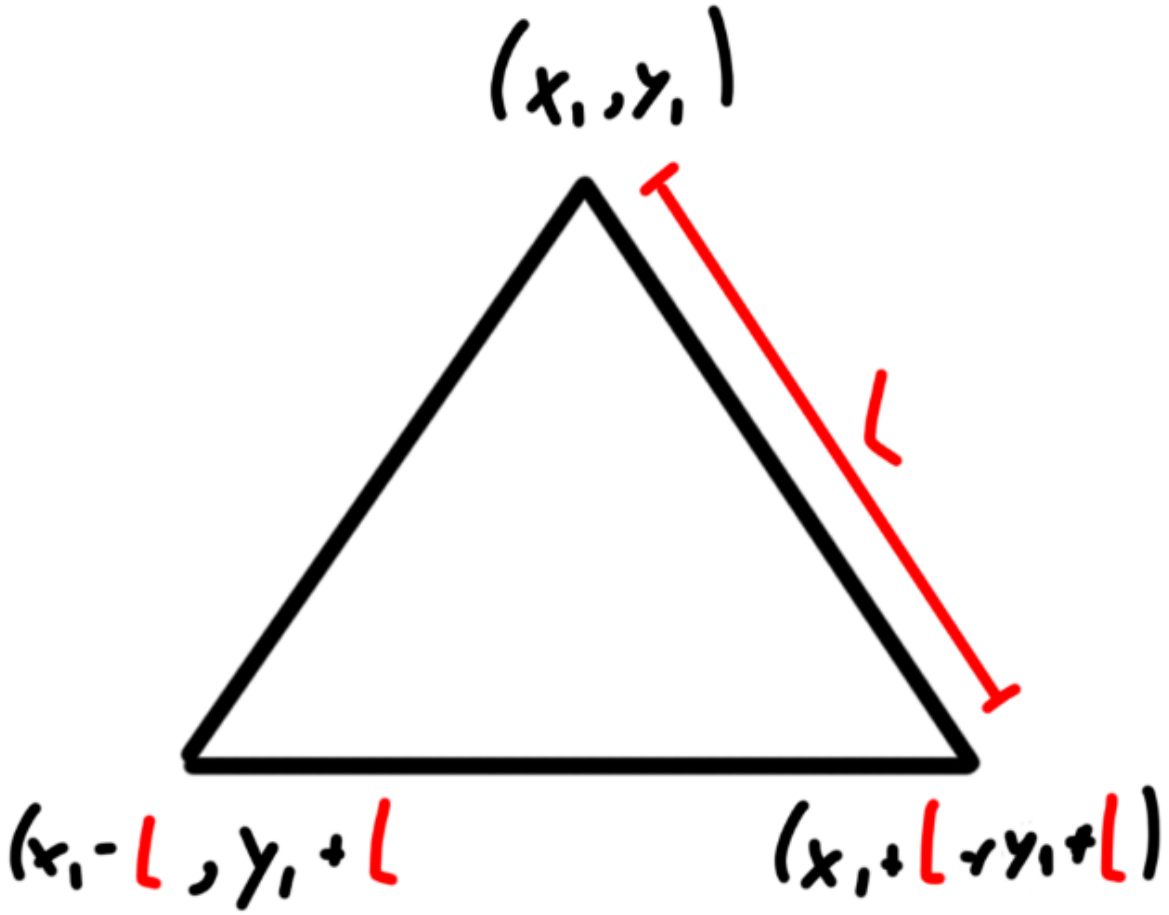
หลักการออกแบบการวาดรูปของกล่องนั้นมีหลักการเหมือนกับการออกแบบรูปสี่เหลี่ยม แต่จะแตกต่างกันโดยที่จะมีการวาดสี่เหลี่ยมเพิ่มอีกหนึ่งรูป และมีเส้นแนวเฉียงเพิ่มอีกสี่เส้น หลักการออกแบบเป็นตัวอย่างตามดังรูปภาพ



ภาพที่ 3 รูปภาพประกอบการออกแบบ Class Box

หลักการออกแบบ Class Triangle

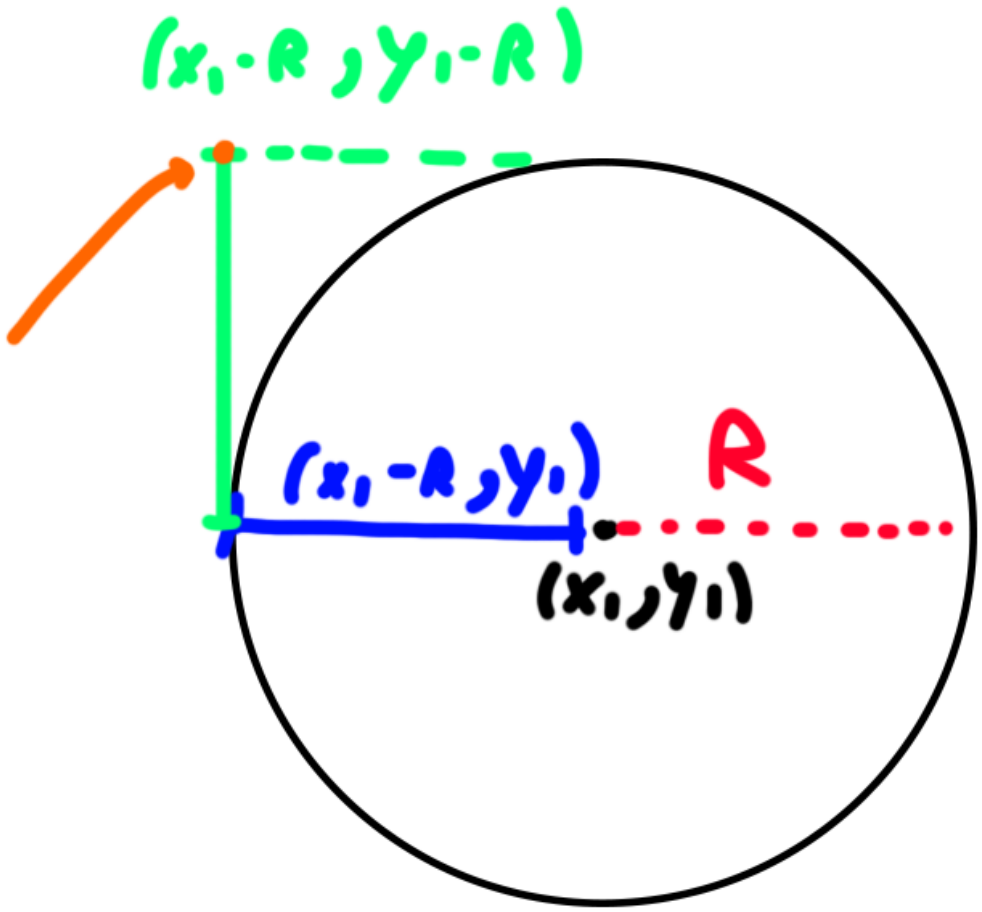
หลักการออกแบบการวาดรูปสามเหลี่ยมนั้นมีหลักการเดียวกับการวาดรูปสี่เหลี่ยมเช่นกัน โดยมีการวาดเส้นที่ละเส้นทั้งหมดสามเส้น แล้วพิกัดของแต่ละเส้นอ้างอิงจากจุดเริ่มต้นในการวาด (x1,y1) เสมอ ดังตัวอย่างตามรูป



ภาพที่ 4 รูปภาพประกอบการออกแบบ Class Triangle

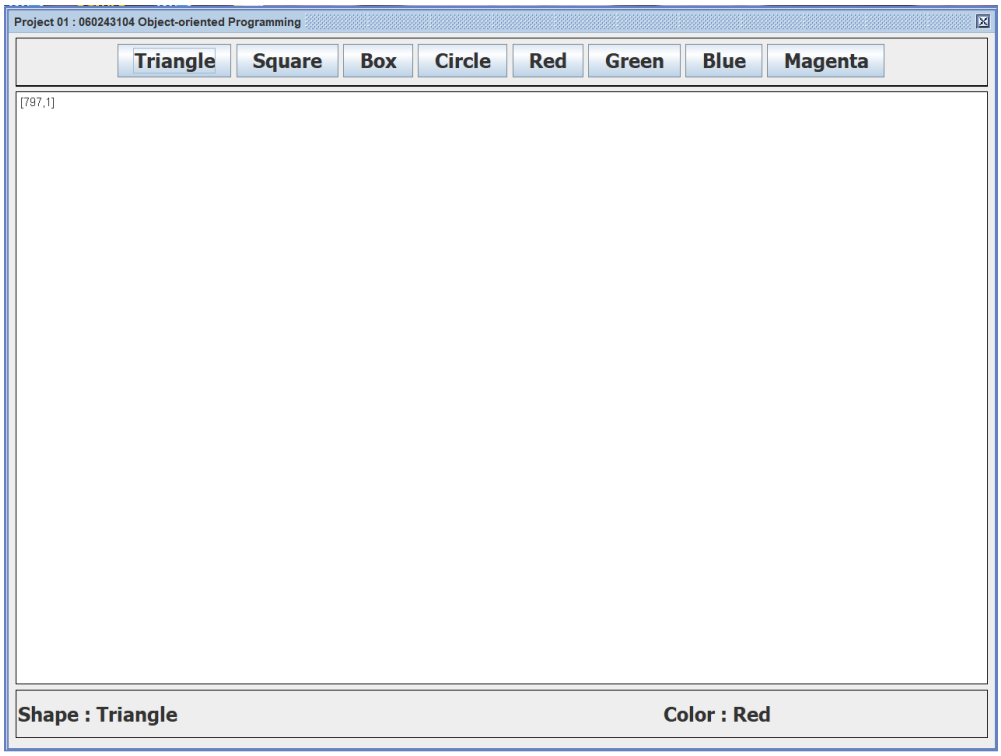
หลักการออกแบบ Class Circle

หลักการออกแบบการวาดรูปวงกลมนั้นจะมีคำสั่งที่วาดลงมาจากด้านซ้ายบนเสมอ ซึ่งเราจะต้องหาวิธีในการหาจุดเริ่มต้นในการวาดวงกลม ความกว้างและความสูงของวงกลม โดยอ้างอิงจากจุดศูนย์กลาง (x1, y1) ของวงกลมเสมอ ๆ โดยแปรผันตามรัศมีของวงกลมดังตัวอย่างดังรูป และความกว้างความยาวของวงกลม คือ พิกัดของจุดเริ่มต้นการวาดของวงกลม + รัศมี x 2

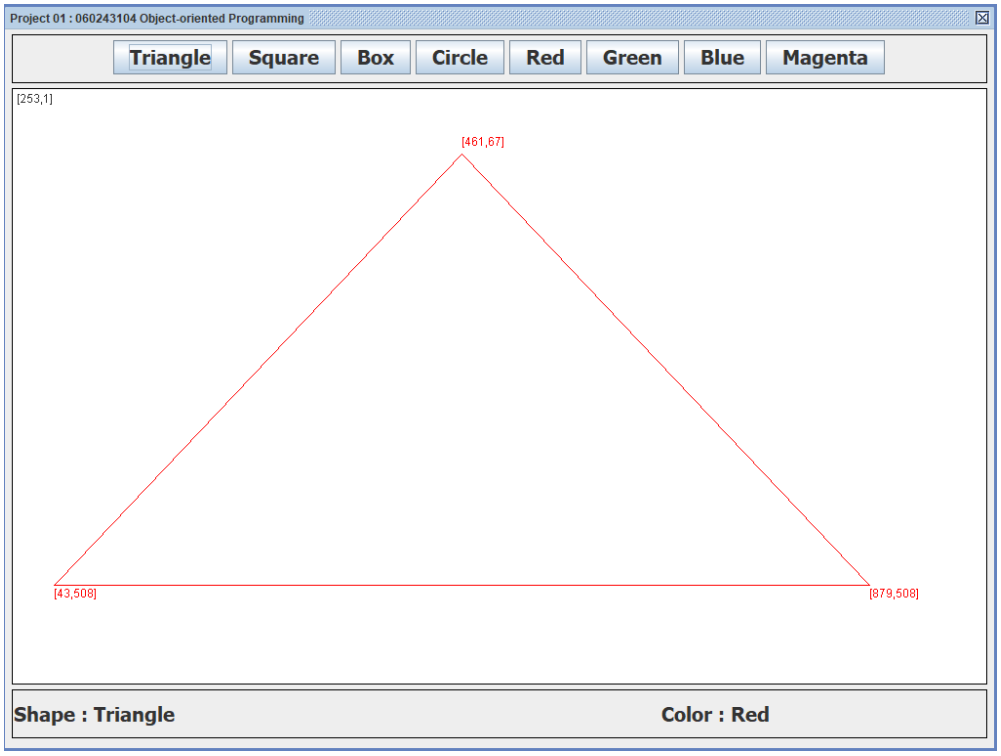


ภาพที่ 5 รูปภาพประกอบการออกแบบ Class Circle

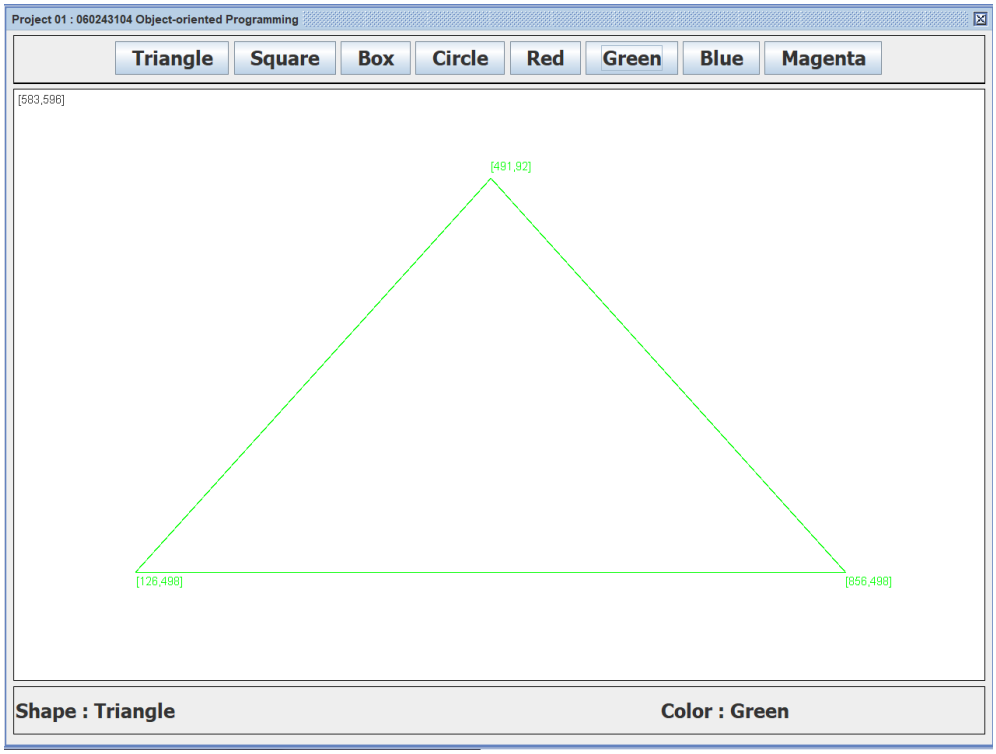
ภาพหน้าจอการทำงานโปรแกรม Java Application สำหรับการวาดรูป



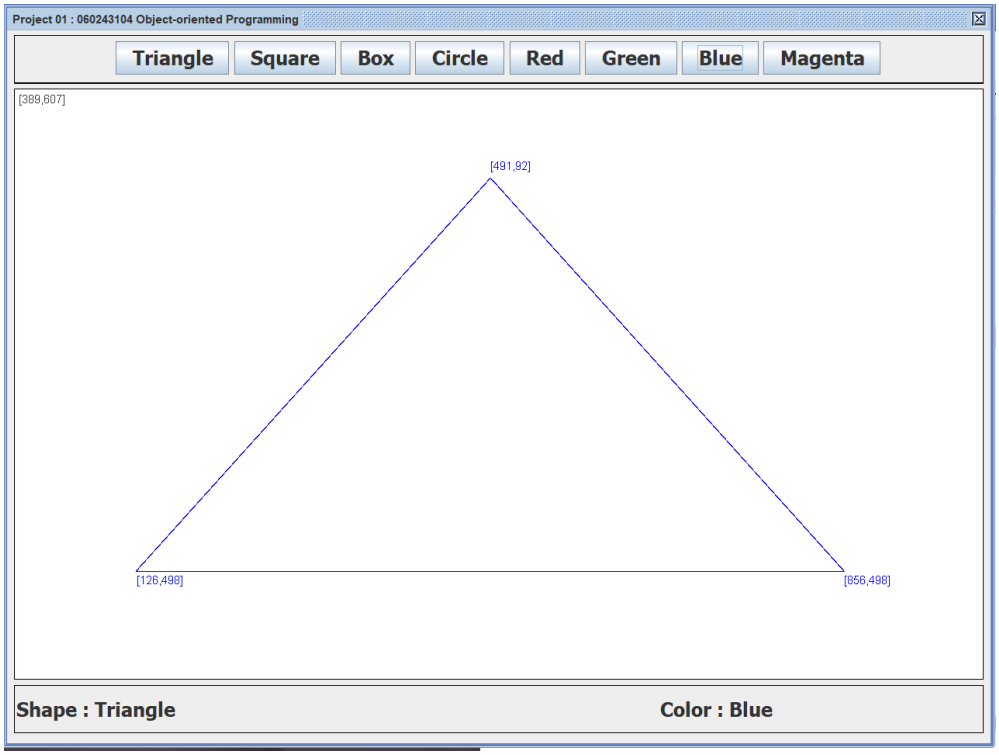
ภาพที่ 6 หน้าจอ Window โปรแกรม Java Application สำหรับการวาดรูป



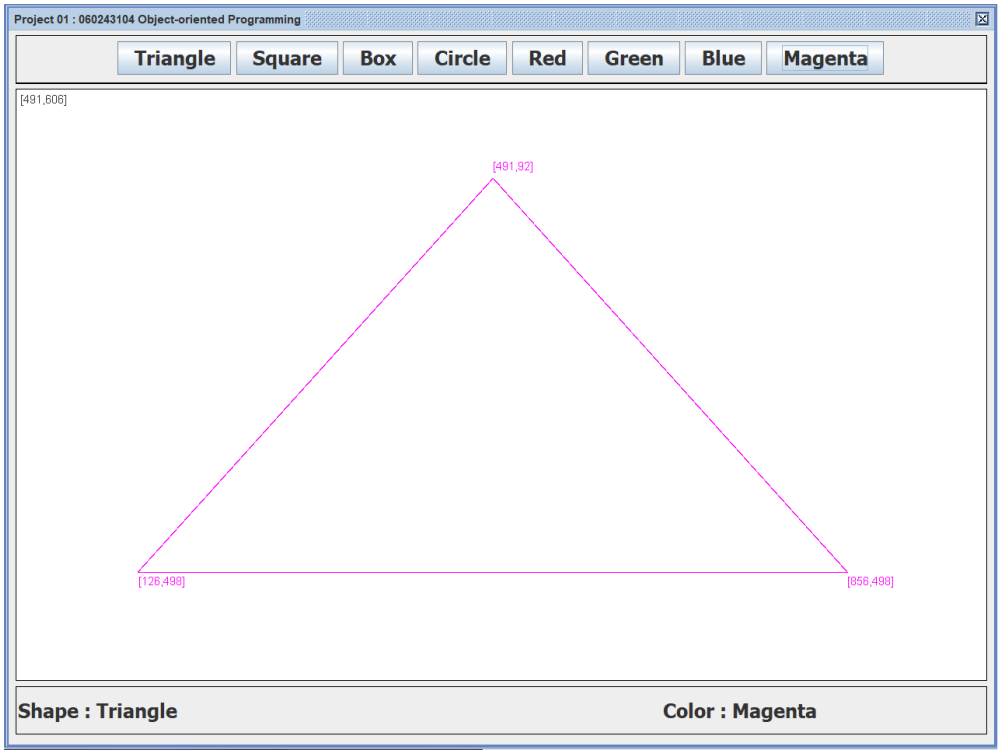
ภาพที่ 7 ใช้เมาส์วาดรูปสามเหลี่ยม โดยกำหนดเส้นเป็นสีแดง



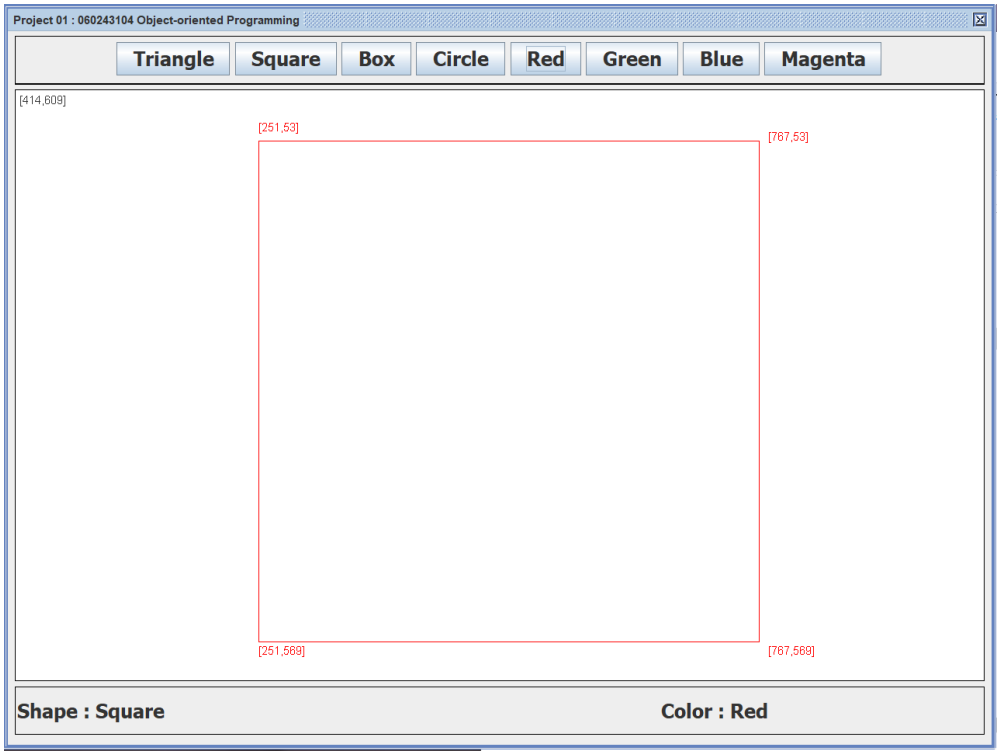
ภาพที่ 8 ใช้เมาส์วาดรูปสามเหลี่ยม โดยกำหนดเส้นเป็นสีเขียว



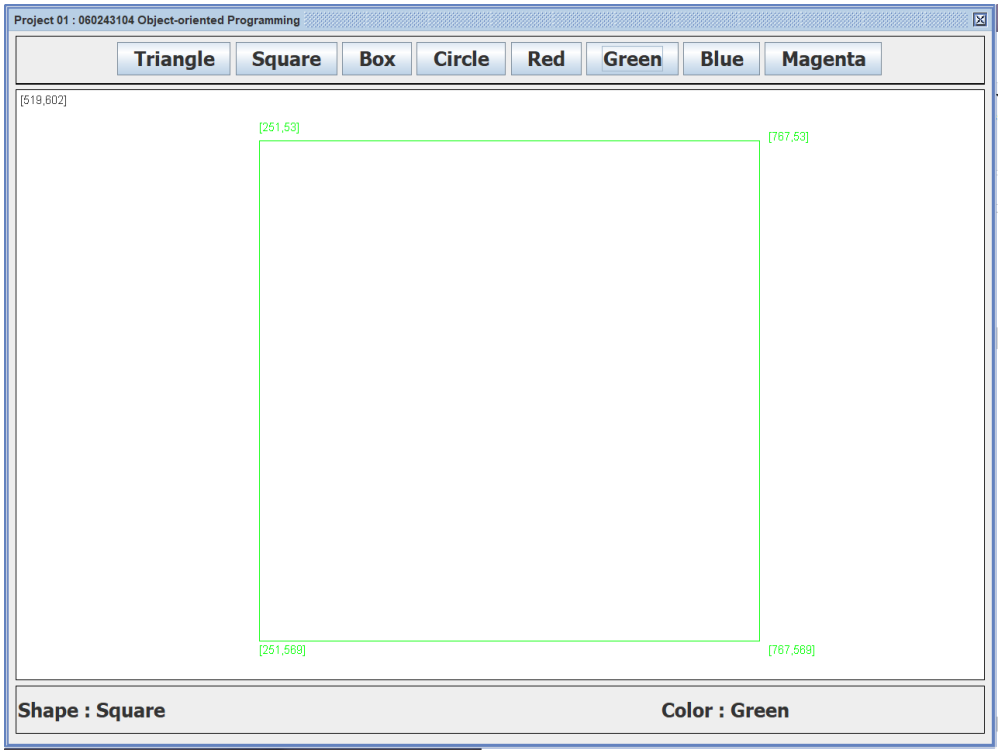
ภาพที่ 9 ใช้เมาส์วาดรูปสามเหลี่ยม โดยกำหนดเส้นเป็นสีฟ้า



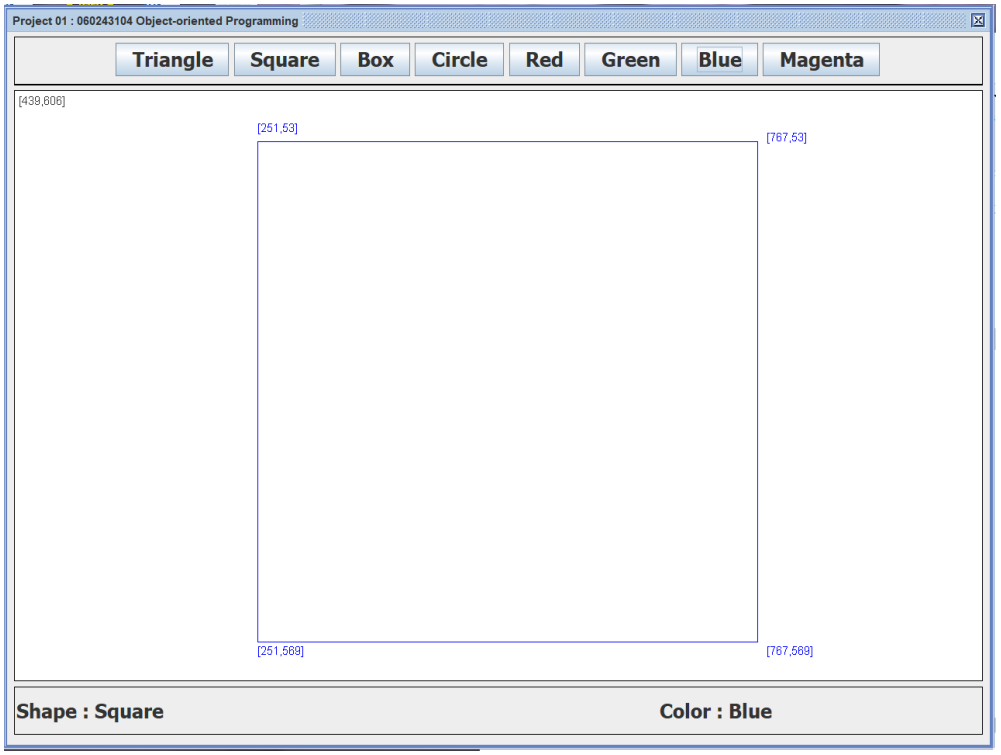
ภาพที่ 10 ใช้เมาส์วาดรูปสามเหลี่ยม โดยกำหนดเส้นเป็นสีมาเจนต้า



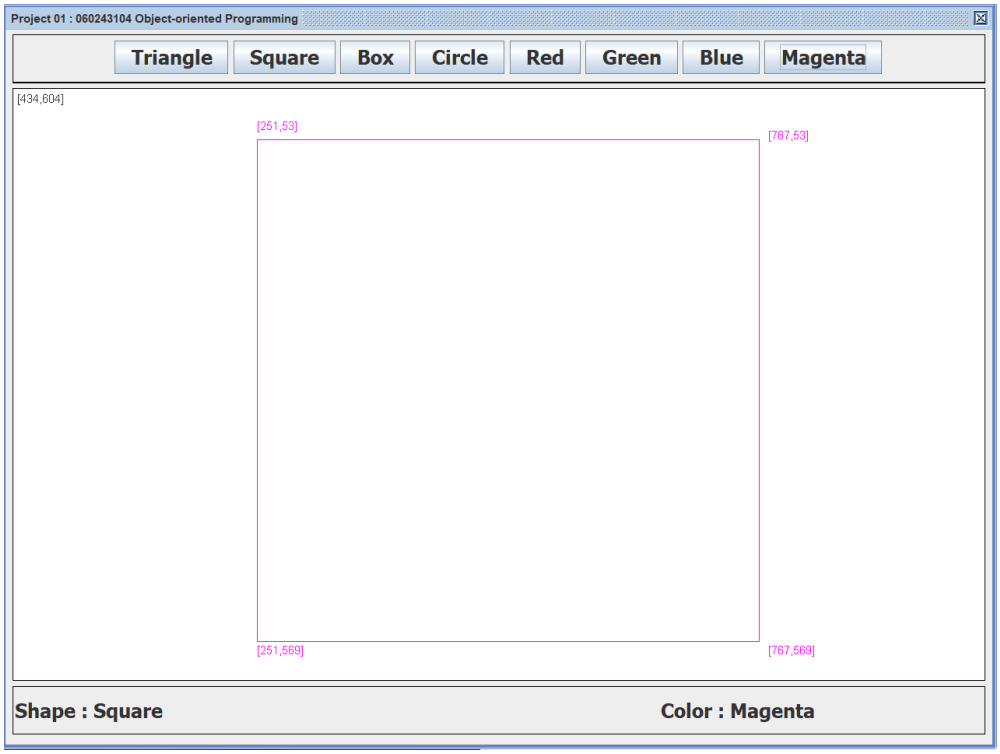
ภาพที่ 11 ใช้เมาส์วาดรูปสี่เหลี่ยม โดยกำหนดเส้นเป็นสีแดง



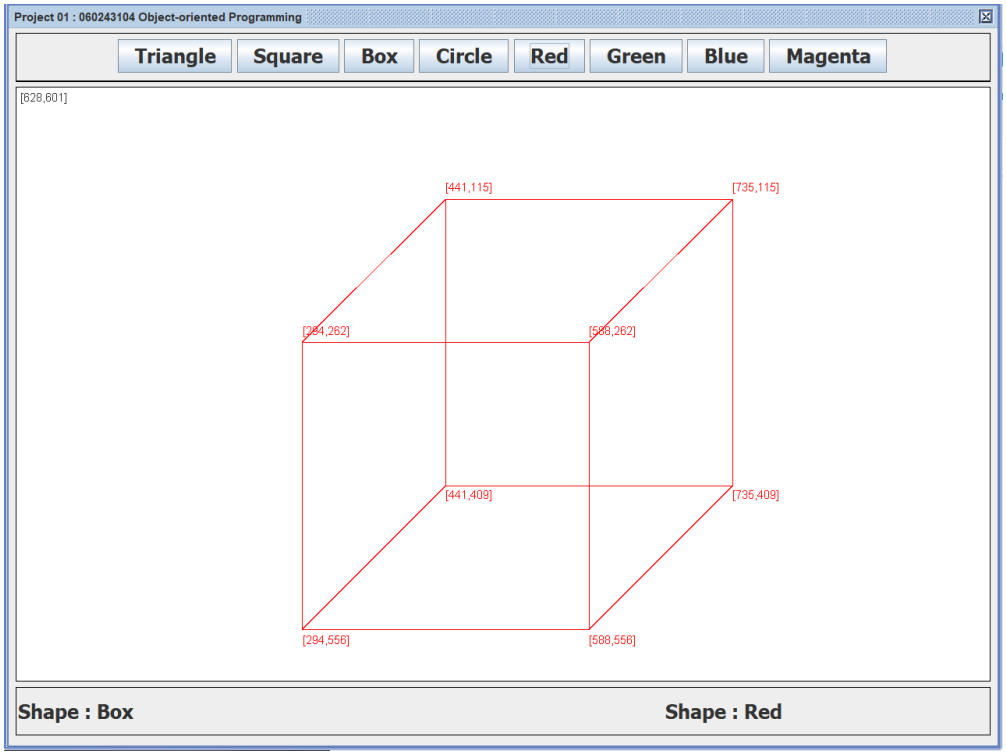
ภาพที่ 12 ใช้เมาส์วาดรูปสี่เหลี่ยม โดยกำหนดเส้นเป็นสีเขียว



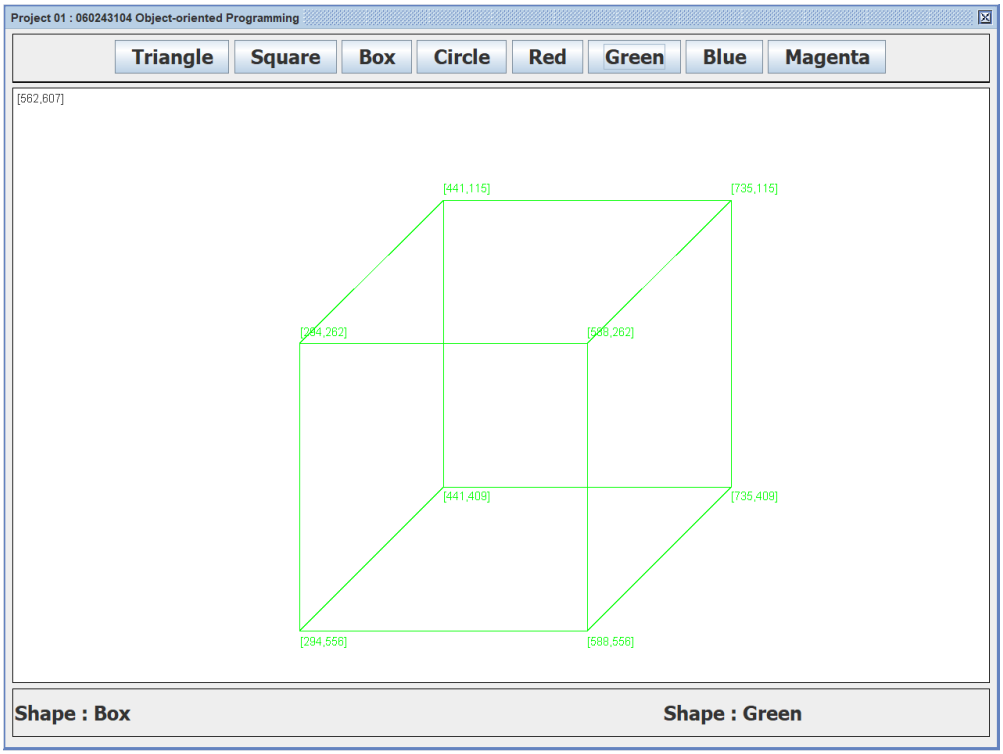
ภาพที่ 13 ใช้เมาส์วาดรูปสี่เหลี่ยม โดยกำหนดเส้นเป็นสีฟ้า



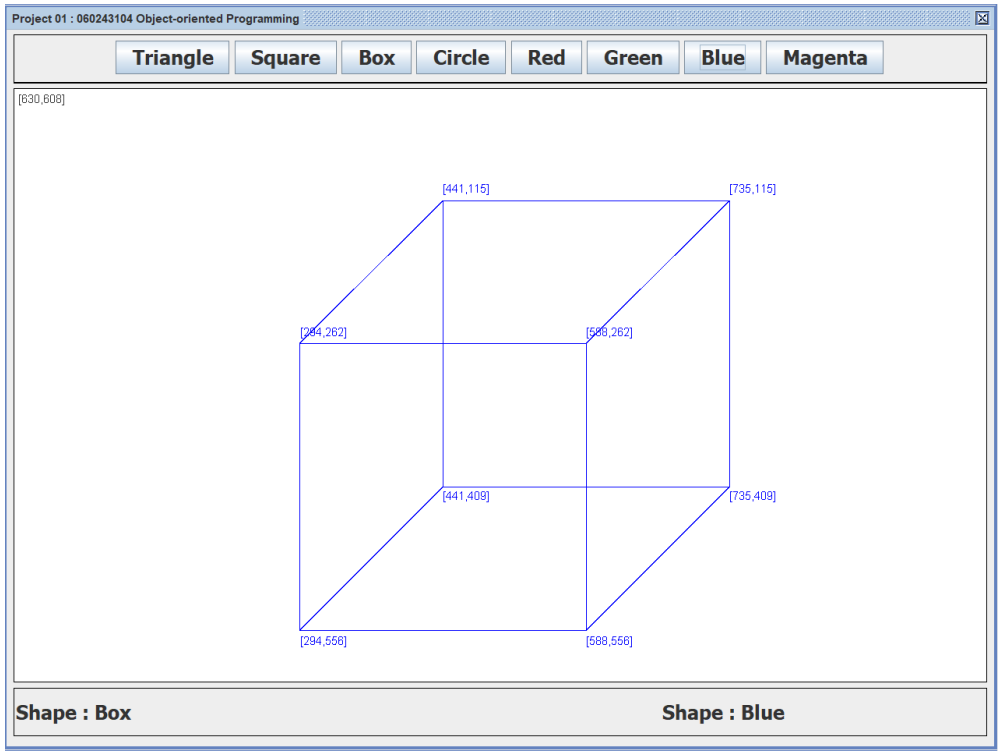
ภาพที่ 14 ใช้เมาส์วาดรูปสี่เหลี่ยม โดยกำหนดเส้นเป็นสีมาเจนต้า



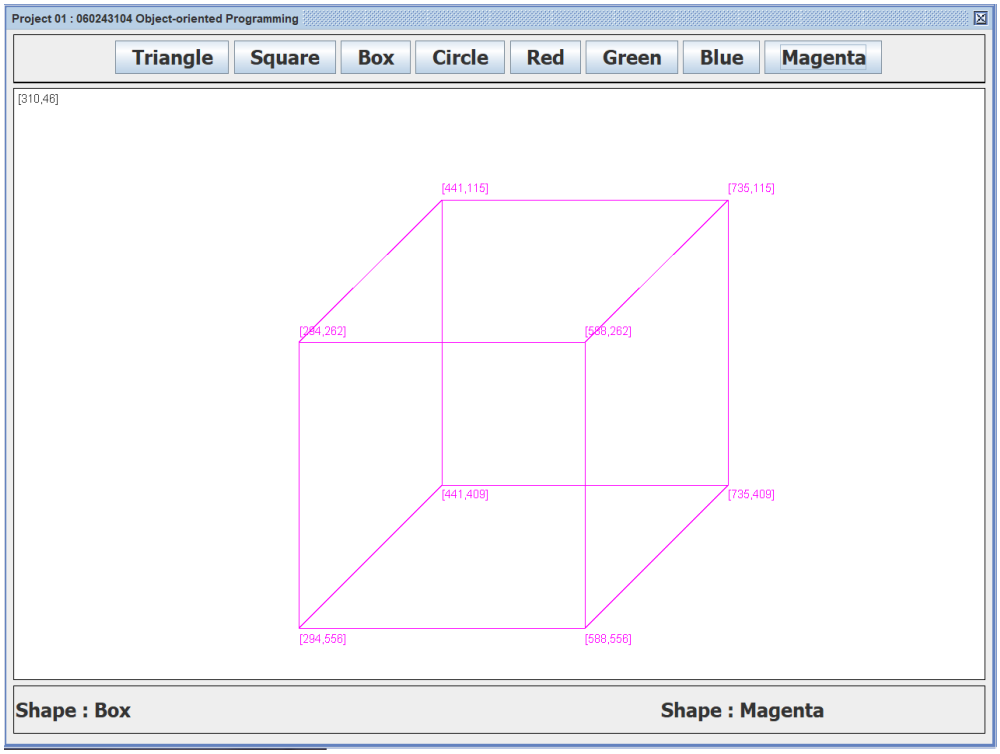
ภาพที่ 15 ใช้เมาส์วาดรูปกล่อง โดยกำหนดเส้นเป็นสีแดง



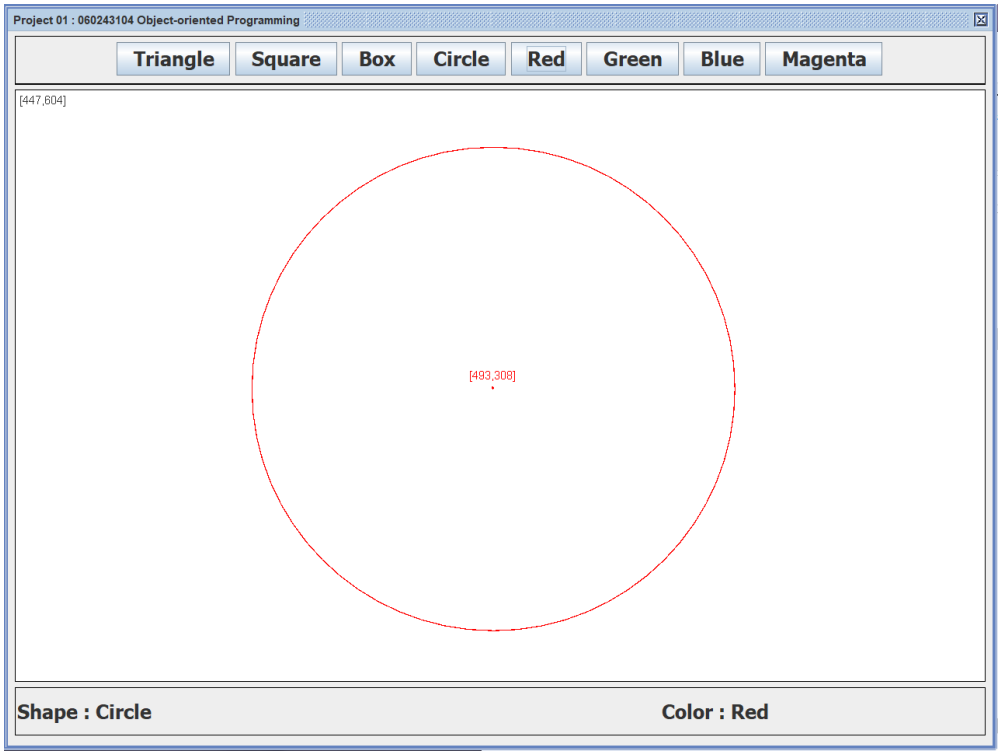
ภาพที่ 16 ใช้เมาส์วาดรูปกล่อง โดยกำหนดเส้นเป็นสีเขียว



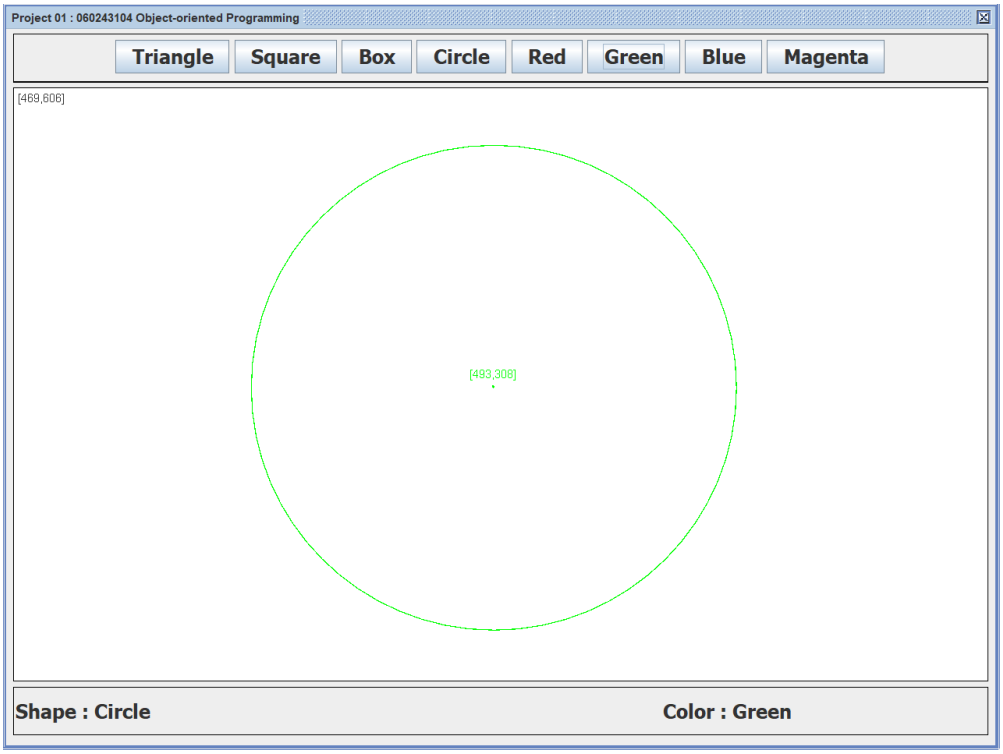
ภาพที่ 17 ใช้เมาส์วาดรูปกล่อง โดยกำหนดเส้นเป็นสีฟ้า



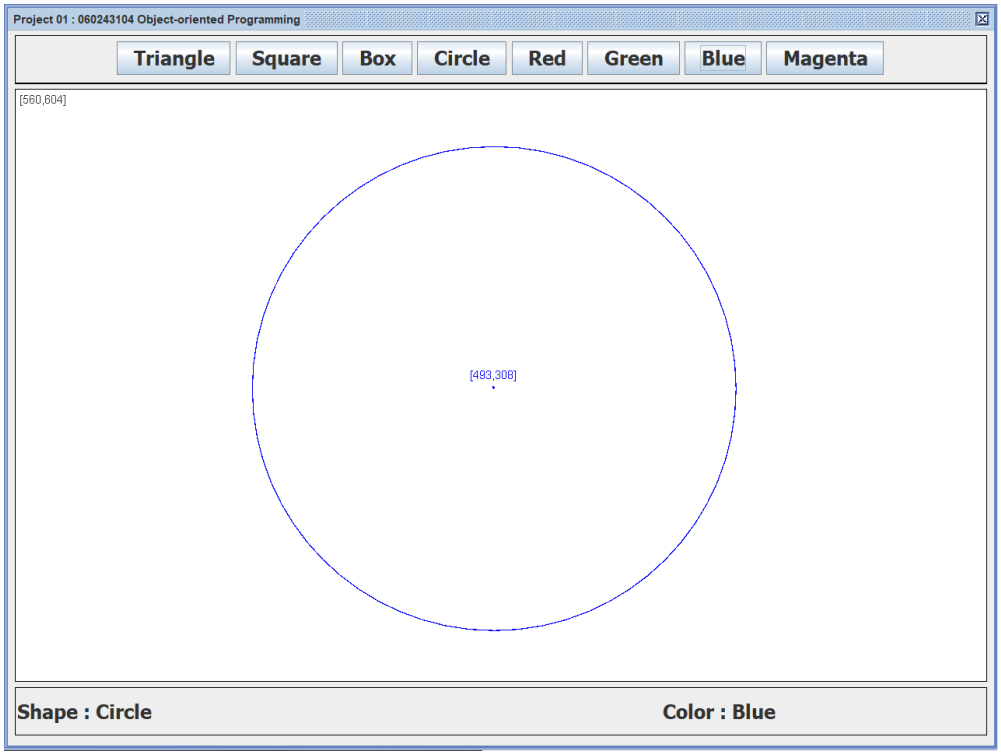
ภาพที่ 18 ใช้เมาส์วาดรูปกล่อง โดยกำหนดเส้นเป็นสีมาเจนต้า



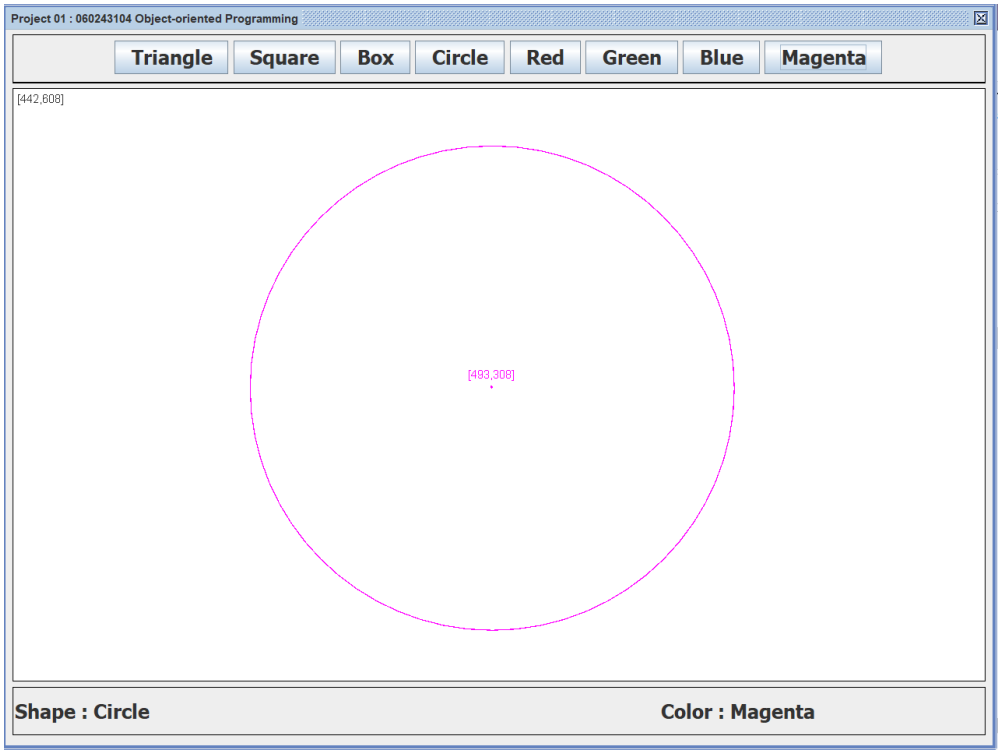
ภาพที่ 19 ใช้เมาส์วาดรูปวงกลม โดยกำหนดเส้นเป็นสีแดง



ภาพที่ 20 ใช้เมาส์วาดรูปวงกลม โดยกำหนดเส้นเป็นสีเขียว

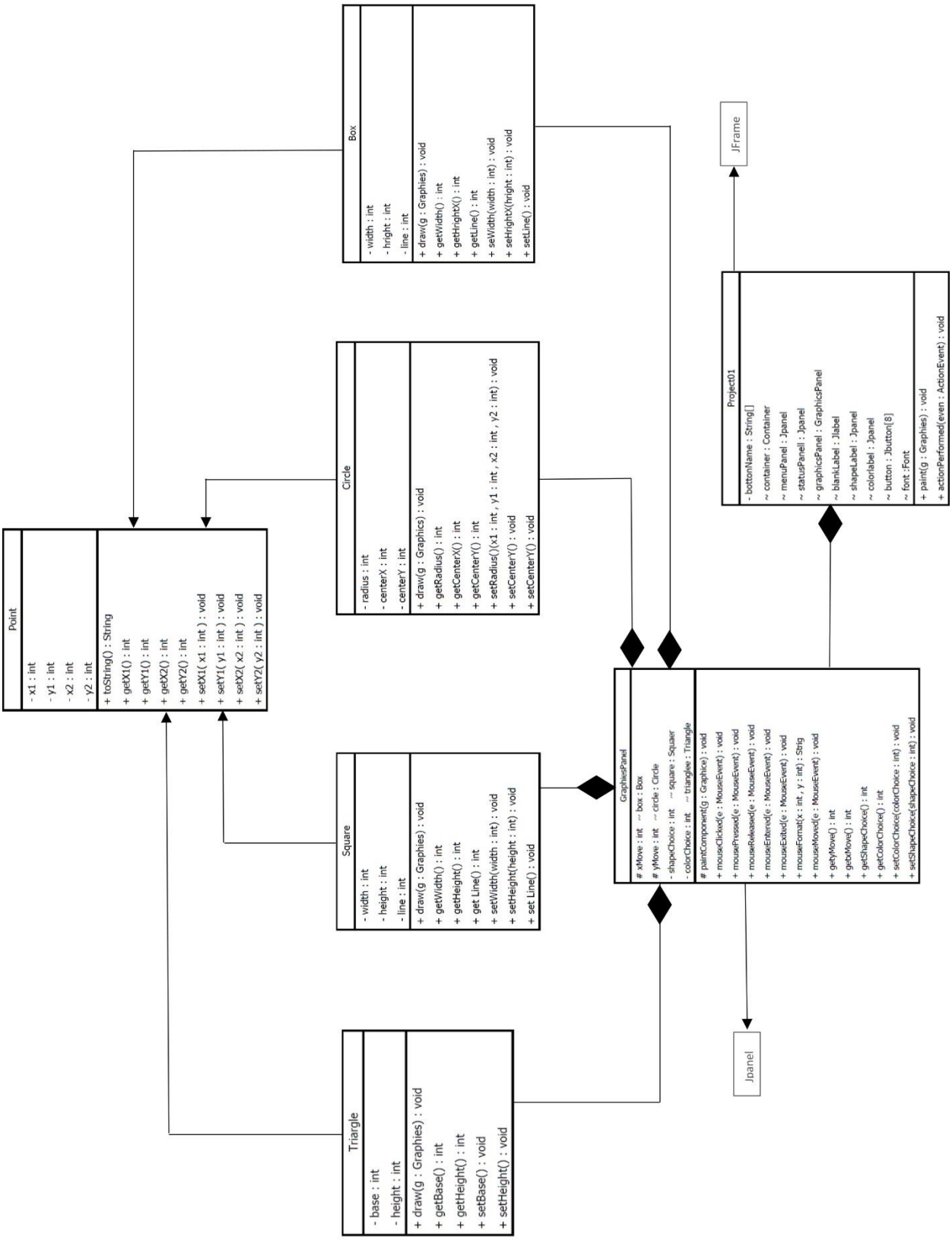


ภาพที่ 21 ใช้เมาส์วาดรูปวงกลม โดยกำหนดเส้นเป็นสีฟ้า



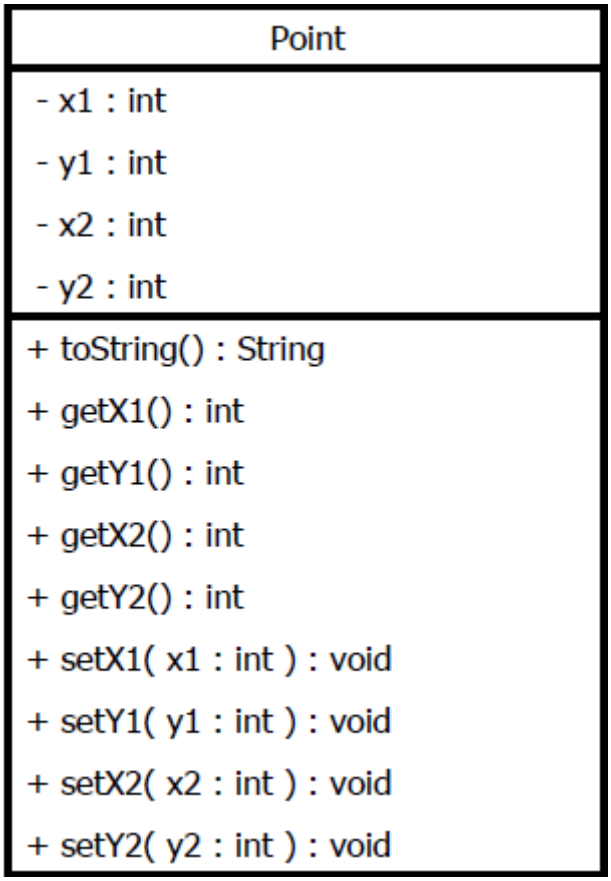
ภาพที่ 22 ใช้เมาส์วาดรูปวงกลม โดยกำหนดเส้นเป็นสีมาเจนต้า

Class Diagram โปรแกรม Java Application สำหรับการวาดรูป



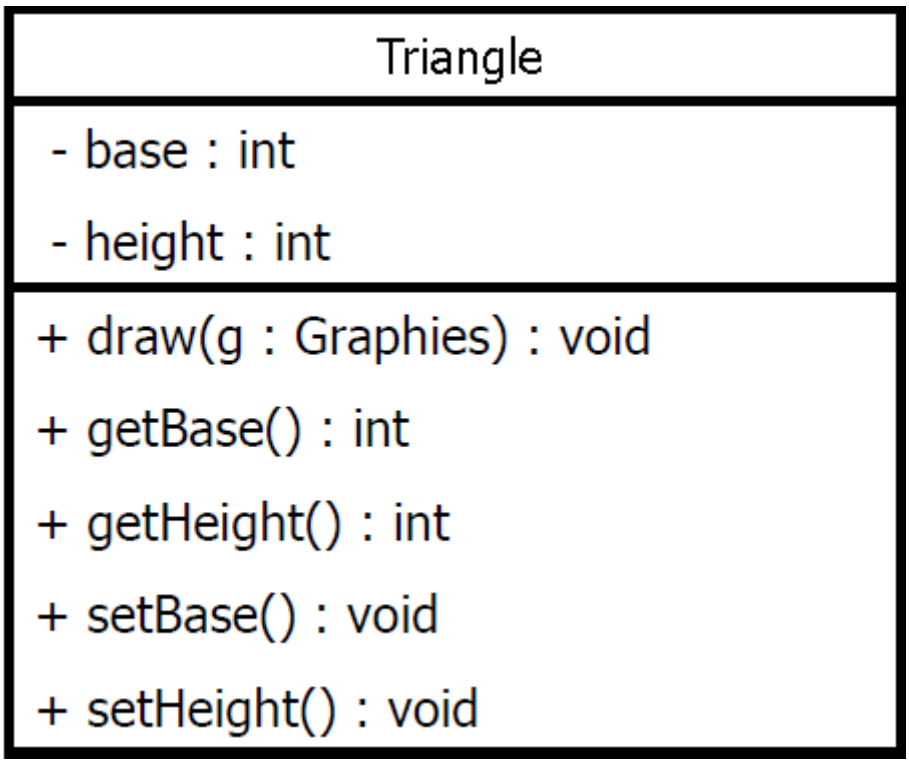
ภาพที่ 23 Class Diagram โปรแกรม Java Application สำหรับการวาดรูป

1. Class Point



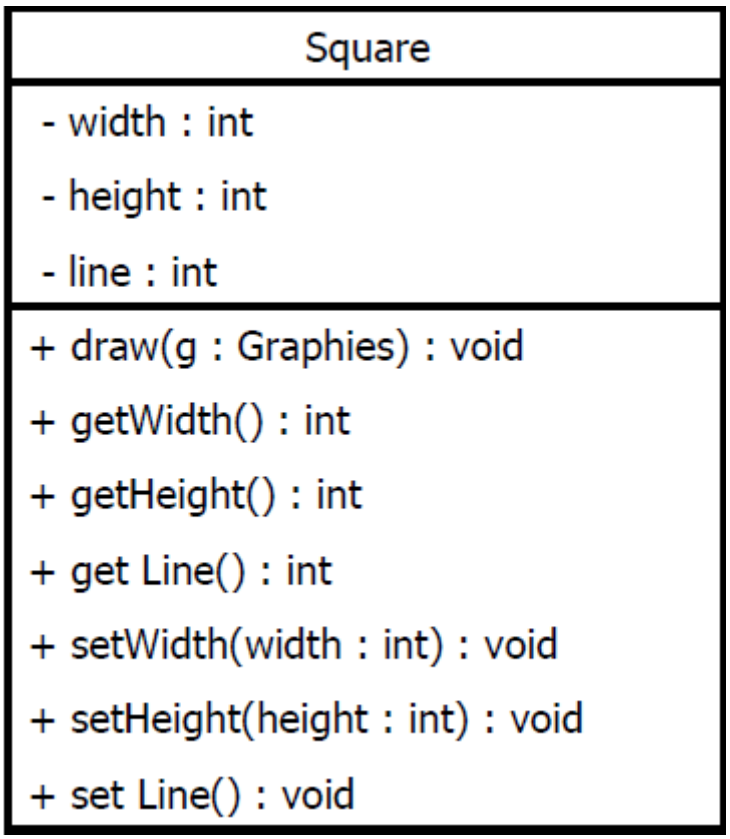
ภาพที่ 24 Class Diagram ของ Class Point

2. Class Triangle



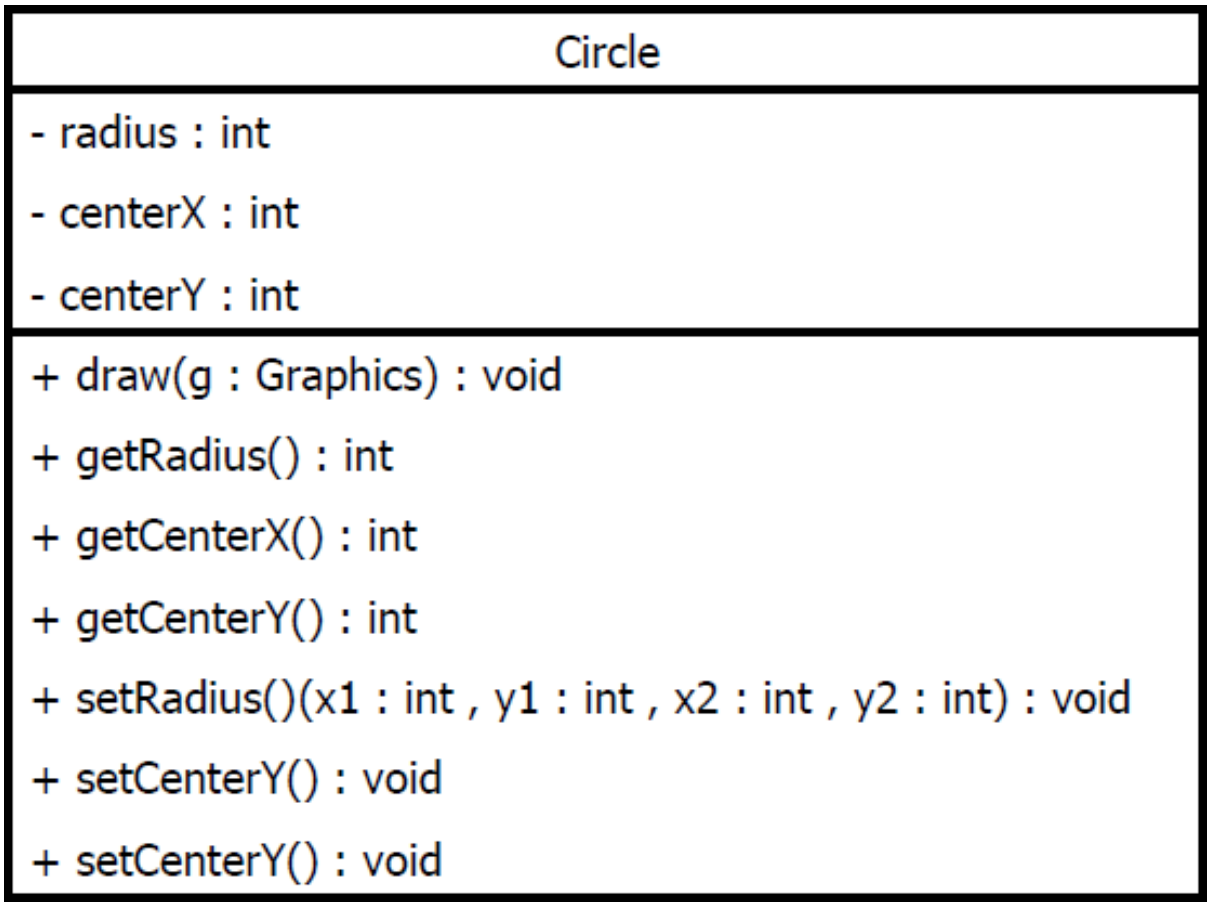
ภาพที่ 25 Class Diagram ของ Class Triangle

3. Class Square



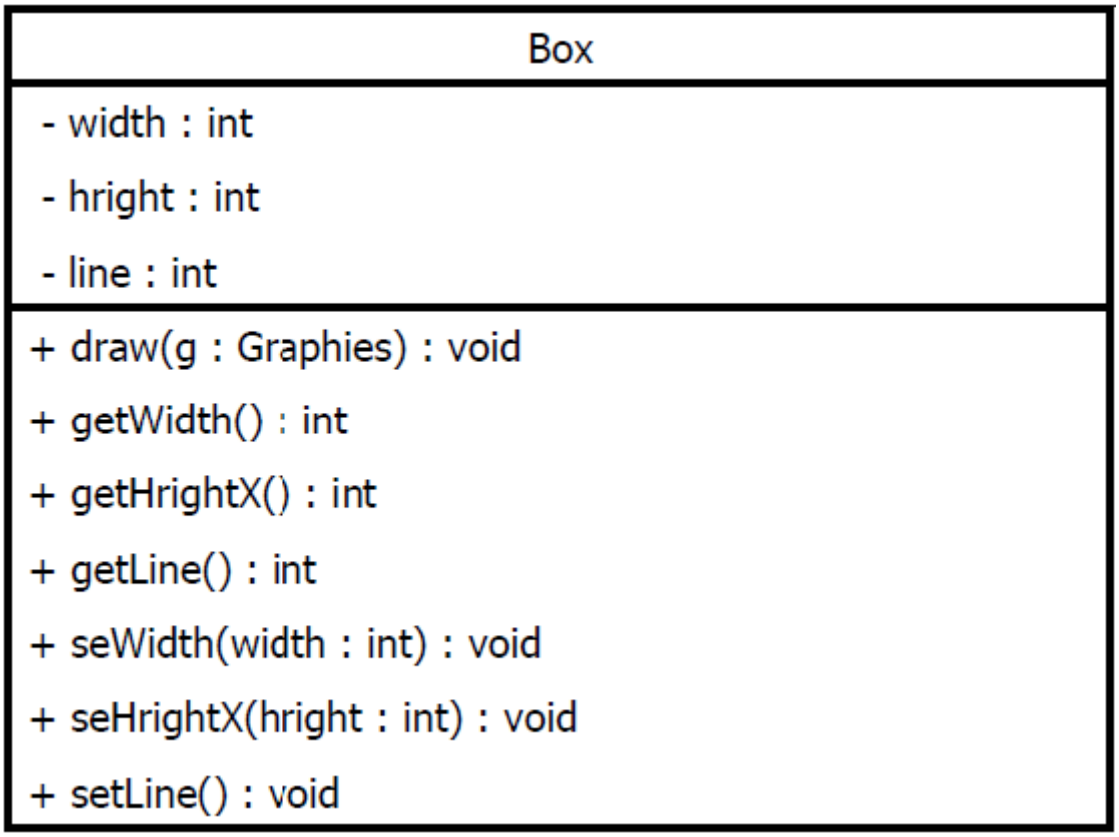
ภาพที่ 26 Class Diagram ของ Class Square

4. Class Circle



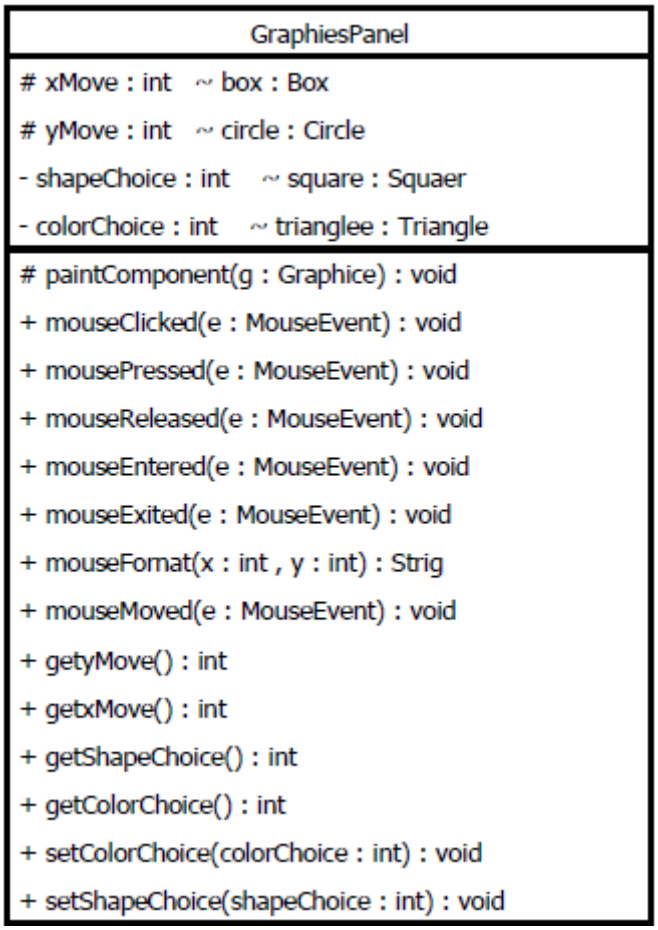
ภาพที่ 27 Class Diagram ของ Class Circle

5. Class Box



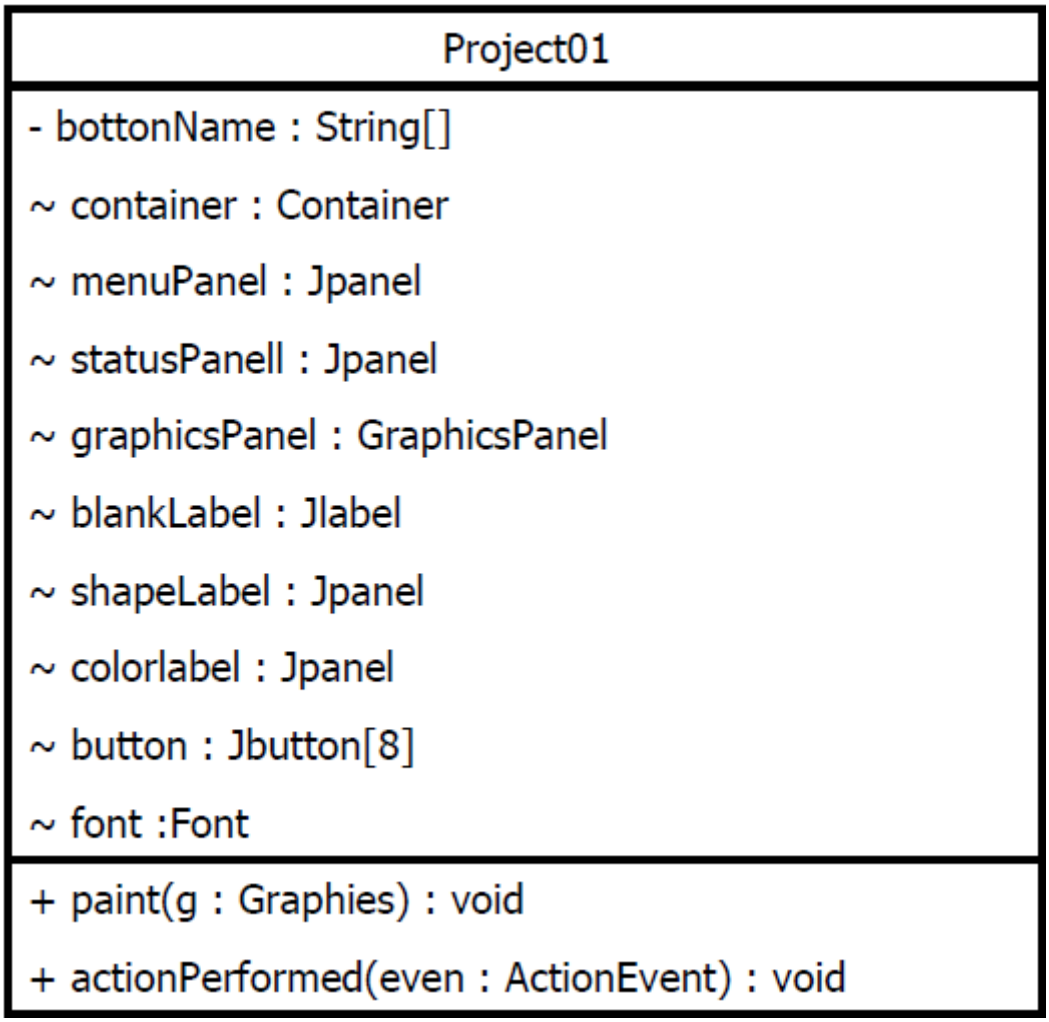
ภาพที่ 28 Class Diagram ของ Class Box

6. Class GraphicsPanel



ภาพที่ 29 Class Diagram ของ Class GraphicsPanel

7. Class Project01



ภาพที่ 30 Class Diagram ของ Class Project01

Source Program Java Application สำหรับการวาดรูป

Source Program Class Point

```
1.   public class Point {
2.      private int x1, y1;
3.      private int x2, y2;
4.
5.      Point(int x1, int y1, int x2, int y2){
6.          this.x1 = x1;
7.          this.y1 = y1;
8.          this.x2 = x2;
9.          this.y2 = y2;
10.     }
11.
12.     @Override
13.     public String toString(){
14.         return "(" + getX1() + ", " + getY1() + ")"
15.                 + ", (" + getX2() + ", " + getY2() + ")";
16.     }
17.
18.     public int getX1() {
19.         return x1;
20.     }
21.
22.     public int getY1() {
23.         return y1;
24.     }
25.
26.     public int getX2() {
27.         return x2;
28.     }
29.
30.     public int getY2() {
31.         return y2;
32.     }
33.
34.     public void setX1(int x1) {
35.         this.x1 = x1;
36.     }
37.
38.     public void setY1(int y1) {
39.         this.y1 = y1;
40.     }
41.
42.     public void setX2(int x2) {
43.         this.x2 = x2;
44.     }
45.
46.     public void setY2(int y2) {
47.         this.y2 = y2;
48.     }
49. }
50.
```

Source Program Class Triangle

```
1.  import java.awt.Graphics;
2.  public class Triangle extends Point{
3.      private int base;
4.      private int height;
5.      Triangle(int x1, int y1, int x2, int y2){
6.          super(x1, y1, x2, y2);
7.      }
8.
9.      public void draw(Graphics g){
10.         setBase(); setHeight();
11.
12.         //Line1 (Top to bottom right)
13.         g.drawLine(getX1(),getY1(),getX1()+base,getY1()+height);
14.         g.drawString(GraphicsPanel.positionFormat(getX1(),getY1()), getX1(),getY1()-8 );
15.
16.         //Line2 ( bottom right to bottom left )
17.         g.drawLine(getX1()+base,getY1()+height,getX1()-base,getY1()+height);
18.         g.drawString(GraphicsPanel.positionFormat(getX1()+base,getY1()+height), getX1()+
    base,getY1()+height +13);
19.
20.         //Line3
21.         g.drawLine(getX1()-base,getY1()+height, getX1(),getY1() );
22.         g.drawString(GraphicsPanel.positionFormat(getX1()-base,getY1()+height), getX1()-
    base,getY1()+height +13);
23.     }
24.
25.     public int getBase() {
26.         return base;
27.     }
28.
29.     public int getHeight() {
30.         return height;
31.     }
32.
33.     public void setBase() {
34.         this.base = getX2()-getX1();
35.     }
36.
37.     public void setHeight() {
38.         this.height = getY2()-getY1();
39.     }
40. }
41.
```

Source Program Class Square

```
1.  import java.awt.Graphics;
2.  public class Square extends Point{
3.      private int width, height, line;
4.      Square(int x1, int y1, int x2, int y2){
5.          super(x1,y1,x2,y2);
6.      }
7.
8.      public void draw(Graphics g){
9.          setWidth(getX2() - getX1() );
10.         setHeight(getY2() - getY1() );
11.         setLine();
12.
13.         //Line1 (Start from Upper Left to Upper Right)
14.         g.drawLine(getX1(),getY1(), getX1()+getLine(), getY1() );
15.         g.drawString(GraphicsPanel.positionFormat( getX1(),getY1() ), getX1(),getY1()-
    10 );
16.
17.         //Line2
18.         g.drawLine(getX1()+getLine(),getY1(), getX1()+getLine(), getY1()+getLine());
19.         g.drawString(GraphicsPanel.positionFormat( getX1()+getLine(),getY1() ), getX1()+
    getLine()+10,getY1() );
20.
21.         //Line3
22.         g.drawLine(getX1()+getLine(),getY1()+getLine(), getX1(),getY1()+getLine());
23.         g.drawString(GraphicsPanel.positionFormat(
24.                 getX1()+getLine(),getY1()+getLine() ), getX1()+getLine()+10,getY1()+getL
    ine()+13 );
25.
26.         //Line4
27.         g.drawLine(getX1(),getY1()+getLine(), getX1(),getY1());
28.         g.drawString(GraphicsPanel.positionFormat(
29.                 getX1(),getY1()+getLine() ), getX1(),getY1()+getLine()+13 );
30.
31.     }
32.
33.     public int getWidth() {
34.         return width;
35.     }
36.
37.     public int getHeight() {
38.         return height;
39.     }
40.
41.     public int getLine() {
42.         return line;
43.     }
44.
45.     public void setWidth(int width) {
46.         this.width = width;
47.     }
48.
49.     public void setHeight(int height) {
50.         this.height = height ;
51.     }
52.
53.     public void setLine() {
54.         this.line = Math.max(getWidth(), getHeight());
55.     }
56. }
57.
```

Source Program Class Circle

```
1.  import java.awt.Graphics;
2.
3.  public class Circle extends Point{
4.      private int radius;
5.      private int centerX, centerY;
6.
7.      Circle(int x1, int y1, int x2, int y2){
8.          super(x1, y1, x2, y2);
9.          setRadius(x1, y1, x2, y2);
10.     }
11.
12.     public void draw(Graphics g){
13.
14.         //(x1,y1, x2,y2)
15.         g.drawOval(getCenterX() - getRadius(),
16.                 getCenterY() - getRadius(),
17.                 getRadius()*2,
18.                 getRadius()*2);
19.
20.         g.drawOval(getCenterX()-2,
21.                 getCenterY() -2,
22.                 2,
23.                 2);
24.
25.         g.drawString(GraphicsPanel.positionFormat(getX1(),getY1()),getCenterX()-
    25,getCenterY()-10);
26.
27.     }
28.
29.     public int getRadius() {
30.         return radius;
31.     }
32.
33.     public int getCenterX() {
34.         return centerX;
35.     }
36.
37.     public int getCenterY() {
38.         return centerY;
39.     }
40.
41.     public void setRadius(int x1, int y1, int x2, int y2) {
42.         // this is difference of x1 x2 and y1 y2
43.         int x = Math.abs(x2-x1);
44.         int y = Math.abs(y2-y1);
45.
46.         this.radius = Math.max(x,y);
47.     }
48.
49.     public void setCenterX() {
50.         this.centerX = getX1();
51.     }
52.
53.     public void setCenterY() {
54.         this.centerY = getY1();
55.     }
56. }
57.
```

Source Program Class Box

```
1.  import java.awt.Graphics;
2.  public class Box extends Point{
3.      private int width, height, line;
4.
5.      Box(int x1, int y1, int x2, int y2){
6.          super(x1, y1, x2, y2);
7.
8.      }
9.
10.     public void draw(Graphics g){
11.         setLine();
12.
13. //*********************************************SQUARE1*****************************
    ************************//
14.         // Square1 Line1 is the main point, so every line of this Box have relation with
     x1, y1
15.         //Line1 (Start from Upper Left to Upper Right)
16.         g.drawLine(getX1(),getY1(), getX1()+getLine(), getY1() );
17.         g.drawString(GraphicsPanel.positionFormat( getX1(),getY1() ), getX1(),getY1()-
    8 );
18.
19.         //Line2
20.         g.drawLine(getX1()+getLine(),getY1(), getX1()+getLine(), getY1()+getLine());
21.         g.drawString(GraphicsPanel.positionFormat( getX1()+getLine(),getY1() ), getX1()+
    getLine(),getY1()-8 );
22.
23.         //Line3
24.         g.drawLine(getX1()+getLine(),getY1()+getLine(), getX1(),getY1()+getLine());
25.         g.drawString(GraphicsPanel.positionFormat(
26.                 getX1()+getLine(),getY1()+getLine() ), getX1()+getLine(),getY1()+getLine
    ()+15 );
27.
28.         //Line4
29.         g.drawLine(getX1(),getY1()+getLine(), getX1(),getY1());
30.         g.drawString(GraphicsPanel.positionFormat(
31.                 getX1(),getY1()+getLine() ), getX1(),getY1()+getLine()+15 );
32. //*********************************************SQUARE1*****************************
    ************************//
33.
34. //*********************************************SQUARE2*****************************
    ************************//
35.         int s2x1,s2y1; // square2 x1 and square2 y1
36.         s2x1 = getX1()+getLine()/2;
37.         s2y1 = getY1()-getLine()/2;
38.
39.         //Line1 (Start from Upper Left to Upper Right)
40.         g.drawLine(s2x1,s2y1, s2x1+getLine(), s2y1 );
41.         g.drawString(GraphicsPanel.positionFormat( s2x1,s2y1 ), s2x1,s2y1-8 );
42.
43.         //Line2
44.         g.drawLine(s2x1+getLine(), s2y1, s2x1+getLine(), s2y1+getLine() );
45.         g.drawString(GraphicsPanel.positionFormat( s2x1+getLine(), s2y1 ), s2x1+getLine(
    ), s2y1-8 );
46.
47.         //Line3
48.         g.drawLine(s2x1+getLine(), s2y1+getLine(), s2x1, s2y1+getLine());
49.         g.drawString(GraphicsPanel.positionFormat(
50.                 s2x1+getLine(), s2y1+getLine() ), s2x1+getLine(), s2y1+getLine() +13 );
51.
52.         //Line4
53.         g.drawLine(s2x1, s2y1+getLine(), s2x1,s2y1 );
54.         g.drawString(GraphicsPanel.positionFormat( s2x1, s2y1+getLine() ), s2x1, s2y1+ge
    tLine()+13 );
55.
56. //*********************************************SQUARE2*****************************
    ************************//
```

```
57.
58. //*********************************************SQUARE3*****************************
    ************************//
59.         //Line1 start from upper left to upper right
60.         g.drawLine(getX1(),getY1(),s2x1,s2y1);
61.
62.         //Line2
63.         g.drawLine(getX1()+getLine(),getY1(), s2x1+getLine(), s2y1);
64.
65.         //Line3 start from bottom right to bottom left
66.         g.drawLine(getX1()+getLine(), getY1()+getLine(), s2x1+getLine(),s2y1+getLine());
67.
68.         //Line4
69.         g.drawLine(getX1(), getY1()+getLine(), s2x1,s2y1+getLine());
70. //*********************************************SQUARE3*****************************
    ************************//
71.
72.     }
73.
74.     public int getWidth() {
75.         return width;
76.     }
77.
78.     public int getHeight() {
79.         return height;
80.     }
81.
82.     public int getLine() {
83.         return line;
84.     }
85.
86.     public void setWidth(int width) {
87.         this.width = width;
88.     }
89.
90.     public void setHeight(int height) {
91.         this.height = height ;
92.     }
93.
94.     public void setLine() {
95.         setWidth( getX2() -getX1() );
96.         setHeight(getY2() - getY1());
97.         this.line = Math.max(getWidth(), getHeight());
98.     }
99. }
100.
```

Source Program Class GraphicsPanel

```
1.  import java.awt.*;
2.  import java.awt.event.*;
3.  import javax.swing.*;
4.  import javax.swing.border.LineBorder;
5.
6.  public class GraphicsPanel extends JPanel implements MouseListener, MouseMotionListener{
7.      protected int xMove, yMove;
8.      private int shapeChoice;
9.      private int colorChoice;
10.
11.     Box box;
12.     Circle circle;
13.     Square square;
14.     Triangle triangle;
15.
16.     GraphicsPanel(){
17.         super(true);
18.         super.setLayout(new FlowLayout(FlowLayout.CENTER,5,5));
19.         super.setPreferredSize(new Dimension(1000,610));
20.         super.setBorder(new LineBorder(Color.BLACK, 1));
21.         super.setBackground(Color.WHITE);
22.
23.         super.addMouseListener(this);
24.         super.addMouseMotionListener(this);
25.
26.         // Set Shape Default Location
27.         box = new Box(-100,0,-100,-100);
28.         circle = new Circle(-100,-100,-100,-100);
29.         square = new Square(-100,-100,-100,-100);
30.         triangle = new Triangle(-100,-100,-100,-100);
31.
32.     }
33.
34.     protected void paintComponent(Graphics g) {
35.         super.paintComponent(g);
36.         g.drawString(positionFormat(xMove,yMove), 5,15);
37.
38.         //set Color
39.         switch (colorChoice){
40.             case 4:
41.                 g.setColor(Color.RED); break;
42.             case 5:
43.                 g.setColor(Color.GREEN); break;
44.             case 6:
45.                 g.setColor(Color.BLUE); break;
46.             case 7:
47.                 g.setColor(Color.MAGENTA); break;
48.         }
49.
50.         //Draw Shape
51.         switch(shapeChoice) {
52.             case 0:
53.                 triangle.draw(g);
54.                 break;
55.             case 1:
56.                 square.draw(g);
57.                 break;
58.             case 2:
59.                 box.draw(g);
60.                 break;
61.             case 3:
62.                 circle.draw(g);
63.                 break;
64.         }
65.         super.repaint();
66.     }
```

```
67.
68.     public void mouseClicked(MouseEvent e) {
69.         // For debugging
70.         System.out.println("mouseClicked " + "(" + getxMove() + ", " + getyMove() + ")")
    ;
71.     }
72.
73.     public void mousePressed(MouseEvent e) {
74.         switch(shapeChoice){
75.             case 0 :
76.                 triangle.setX1(e.getX());
77.                 triangle.setY1(e.getY());
78.                 triangle.setX2(e.getX());
79.                 triangle.setY2(e.getY());
80.                 break;
81.             case 1 :
82.                 square.setX1(e.getX());
83.                 square.setY1(e.getY());
84.                 square.setX2(e.getX());
85.                 square.setY2(e.getY());
86.                 break;
87.             case 2:
88.                 box.setX1(e.getX());
89.                 box.setY1(e.getY());
90.                 box.setX2(e.getX());
91.                 box.setY2(e.getY());
92.                 break;
93.             case 3:
94.                 circle.setX1(e.getX());
95.                 circle.setY1(e.getY());
96.                 circle.setX2(e.getX());
97.                 circle.setY2(e.getY());
98.                 circle.setDiameter(0,0,0,0);
99.                 circle.setCenterX(); circle.setCenterY();
100.                  break;
101.         }
102.         repaint();
103.
104.     }
105.
106.     public void mouseReleased(MouseEvent e) {
107.
108.     }
109.
110.     public void mouseEntered(MouseEvent e) {
111.
112.     }
113.
114.     public void mouseExited(MouseEvent e) {
115.
116.     }
117.
118.     public void mouseDragged(MouseEvent e) {
119.         //For debugging
120.         //System.out.println("mouseDragged " + "(" + e.getX() + ", " + e.getY() + ")")
    ;
121.         switch(shapeChoice){
122.             case 0 :
123.                 triangle.setX2(e.getX());
124.                 triangle.setY2(e.getY());
125.                 break;
126.             case 1 :
127.                 square.setX2(e.getX());
128.                 square.setY2(e.getY());
129.                 break;
130.             case 2:
131.                 box.setX2(e.getX());
132.                 box.setY2(e.getY());
133.                 break;
```

```
134.                 case 3:
135.                     circle.setX2(e.getX());
136.                     circle.setY2(e.getY());
137.                     circle.setDiameter(circle.getX1(),circle.getY1(),circle.getX2(),circle
    .getY2());
138.                     circle.setCenterX(); circle.setCenterY();
139.                     break;
140.             }
141.
142.             repaint();
143.
144.         }
145.         public static String positionFormat(int x, int y){
146.             return "["+ x + "," + y + "]";
147.         }
148.
149.         public void mouseMoved(MouseEvent e) {
150.             this.xMove = e.getX();
151.             this.yMove = e.getY();
152.             //System.out.println("mouseMoved " + "(" + getxMove() + ", " + getyMove() + ")
    ");
153.         }
154.
155.         public int getxMove() {
156.             return xMove;
157.         }
158.
159.         public int getyMove() {
160.             return yMove;
161.         }
162.
163.         public int getShapeChoice() {
164.             return shapeChoice;
165.         }
166.
167.         public int getColorChoice() {
168.             return colorChoice;
169.         }
170.
171.         public void setShapeChoice(int shapeChoice) {
172.             this.shapeChoice = shapeChoice;
173.         }
174.
175.         public void setColorChoice(int colorChoice) {
176.             this.colorChoice = colorChoice;
177.         }
178.
179.  }
180.
```

Source Program Class Project01

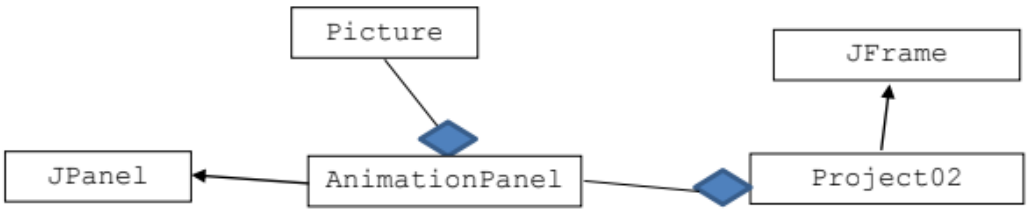
```
1.  import java.awt.*;
2.  import java.awt.event.*;
3.  import javax.swing.*;
4.  import javax.swing.border.LineBorder;
5.
6.  public class Project01 extends JFrame implements ActionListener{
7.      final private String[] buttonName = {"Triangle","Square","Box","Circle","Red","Green
    ","Blue","Magenta"};
8.
9.      // This is for get user screen resolution x,y
10.     Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
11.
12.     Container container;
13.     JPanel menuPanel, statusPanel;
14.     GraphicsPanel graphicsPanel;
15.     JLabel blankLabel = new JLabel();
16.     JLabel shapeLabel, colorLabel;
17.     JButton[] button = new JButton[8];
18.     Font font = new Font("Tahoma",Font.BOLD,20);
19.
20.     Project01(){
21.         super("Project 01 : 060243104 Object-oriented Programming");
22.         ////////////////////////////////////////////////////
23.         //This is for decoration program title. (By use old style JFrame);
24.         super.setUndecorated(true);
25.         super.getRootPane().setWindowDecorationStyle( 2);
26.         ////////////////////////////////////////////////////
27.         super.getRootPane().setFont(font);
28.
29.         setDefaultLookAndFeelDecorated(true);
30.         container = getContentPane();
31.         container.setLayout(new FlowLayout(FlowLayout.CENTER));
32.
33. ////////////////////////////////////////////////////////////////////////////////
34. ////////////////////////////////// Create Panel ////////////////////////////////
35.         //*********** Create Selection Menu Panel ***********//
36.         menuPanel = new JPanel();
37.         menuPanel.setLayout(new FlowLayout(FlowLayout.CENTER,5,5));
38.         menuPanel.setPreferredSize(new Dimension(1000,50));
39.         menuPanel.setBorder(new LineBorder(Color.BLACK, 1));
40.
41.         //create and add Button
42.         for (int i = 0; i < button.length; i++){
43.             button[i] = new JButton(buttonName[i]);
44.             button[i].addActionListener(this);
45.             button[i].setFont(font);
46.
47.             menuPanel.add(button[i]);
48.         }
49.
50.         //*********** Create Painting Area Panel ***********//
51.         graphicsPanel = new GraphicsPanel();
52.
53.         //set Default Shape Choice ( 0 - 3 )
54.         graphicsPanel.setShapeChoice(0);
55.         //set Default Color Choice( 4 - 7 )
56.         graphicsPanel.setColorChoice(4);
57.
58.         //*********** Create Status Panel ***********//
59.         statusPanel = new JPanel();
60.         statusPanel.setPreferredSize(new Dimension(1000,50));
61.         statusPanel.setBorder(new LineBorder(Color.BLACK, 1));
62.         statusPanel.setLayout(new GridLayout(1,3));
63.
64.         shapeLabel = new JLabel("Shape : " + buttonName[graphicsPanel.getShapeChoice()]
    , SwingConstants.LEFT);
```

```
65.         shapeLabel.setFont(font);
66.         statusPanel.add(shapeLabel);
67.
68.         statusPanel.add(blankLabel);
69.
70.         colorLabel = new JLabel("Color : " + buttonName[graphicsPanel.getColorChoice()],
    SwingConstants.LEFT);
71.         colorLabel.setFont(font);
72.
73.         statusPanel.add(colorLabel);
74.
75.         // Add Panel to Container
76.         container.add(menuPanel);
77.         container.add(graphicsPanel);
78.         container.add(statusPanel);
79.
80. ////////////////////////////////// Finished Panel ////////////////////////////////////
81. ////////////////////////////////////////////////////////////////////////////////////////
82.         setSize(1024,768);
83.         setLocation(
84.                 //set window position to center
85.                 //screenSize/2 - windowWidth/2
86.                 (int)screenSize.getWidth()/2-(super.getWidth()/2),
87.                 (int)screenSize.getHeight()/2-(super.getHeight()/2) );
88.         setVisible(true);
89.         setResizable(false);
90.         setDefaultCloseOperation(EXIT_ON_CLOSE);
91.     }
92.
93.     public void actionPerformed(ActionEvent event){
94.         for (int i =0; i < button.length; i++){
95.             if(event.getSource() == button[i]){
96.                 if (i < 4){
97.                     graphicsPanel.setShapeChoice(i);
98.                 }else graphicsPanel.setColorChoice(i);
99.                 shapeLabel.setText("Shape : " + buttonName[graphicsPanel.getShapeChoice(
    )]);
100.                   colorLabel.setText("Color : " + buttonName[graphicsPanel.getColorChoic
    e()]);
101.              } // end of if(event.getSource() == button[i])
102.          } // end for statement.
103.
104.     }
105.
106.     public static void main(String[] args) {
107.         new Project01();
108.     }
109.
110.  }
111.
```

โครงงานที่ 2

โปรแกรม Java Application สำหรับการแสดงการเคลื่อนที่ของรูปภาพ

โครงงานที่ 2 เป็นโปรแกรม Java Application สำหรับการแสดงการเคลื่อนที่ของรูปภาพ UFO ขนาดหน้าจอ 1024x768 พิกเซล มีรายละเอียดหน้าจอโปรแกรมดังรูปด้านล่าง โดยต้องมีการออกแบบคลาสมาจัดเก็บข้อมูลก่อน ซึ่งมียาน UFO จำนวนสูงสุด 30 ลำ โปรแกรมใช้คีย์บอร์ดควบคุมการทำงาน โดยการกดปุ่ม P เพื่อเล่นแอนิเมชัน ปุ่ม S เพื่อหยุดแอนิเมชัน กด 1 กำหนดจำนวน UFO 5 ลำ กด 2 กำหนดจำนวน UFO 10 ลำ กด 3 กำหนดจำนวน UFO 15 ลำ กด 4 กำหนดจำนวน UFO 20 ลำ กด 5 กำหนดจำนวน UFO 25 ลำ กด 6 เพื่อกำหนดจำนวน UFO 30 ลำ



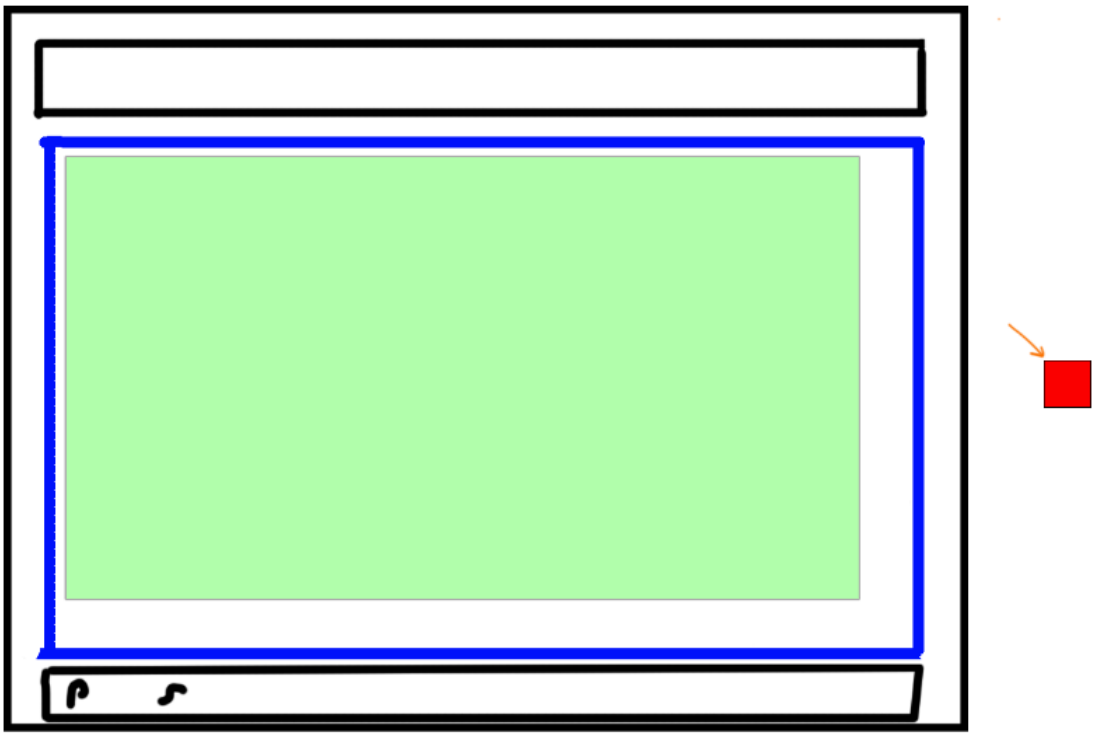
ภาพที่ 31 ตัวอย่างโปรแกรม Java Application สำหรับการแสดงการเคลื่อนที่ของรูปภาพ

หลักการออกแบบ Class โปรแกรม Java Application สำหรับการแสดงการเคลื่อนที่ของรูปภาพ

หลักการออกแบบ Class Picture

โดยหลักการออกแบบการเกิดของ UFO นั้นจะสุ่มได้แค่ภายในพื้นที่สีเขียว โดนเว้นออกจากขอบของจอเล็กน้อยเพื่อป้องกันข้อผิดพลาดเวลา UFO เคลื่อนที่ ตรงบริเวณขวาและล่างนั้นต้องเว้นมากกว่าปกติ เพราะจุดของ UFO นั้นจะอยู่ตรงมุมซ้ายบนของรูป UFO เสมอดังลูกศรสีส้มที่ชี้ตรงกล่องสี่เหลี่ยมสีแดง ดังนั้นจึงต้องเว้นขอบเพิ่มเติม

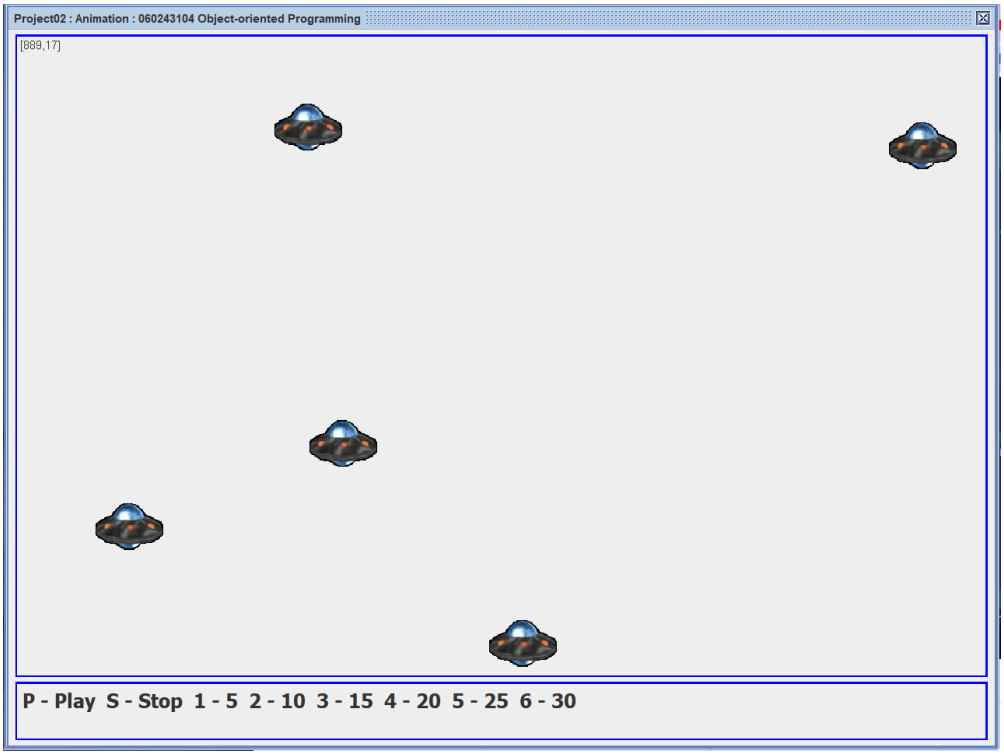
หลักการออกแบบของการเคลื่อนที่ UFO นั้นจะออกแบบโดยไม่ให้ UFO เคลื่อนที่เกินกล่องสีน้ำเงิน ถ้าหากเกินให้ UFO เคลื่อนที่กลับเข้ามา



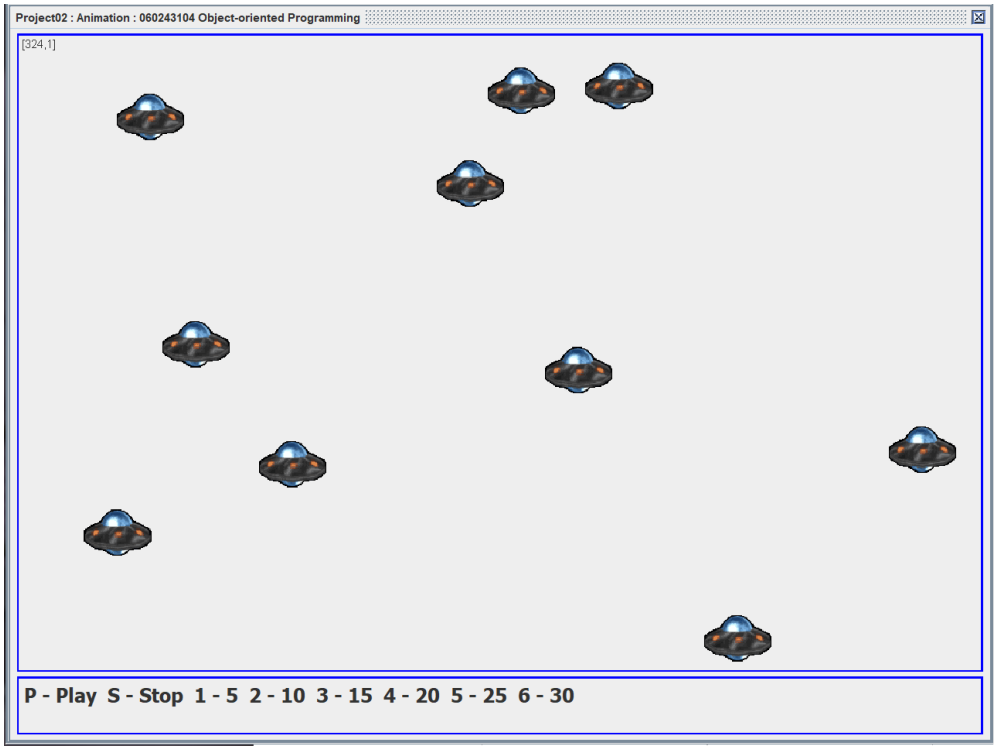
ภาพที่ 32 รูปภาพประกอบการออกแบบ Class Picture

ภาพหน้าจอการทำงานโปรแกรม Java Application สำหรับการแสดงการเคลื่อนที่ของรูปภาพ

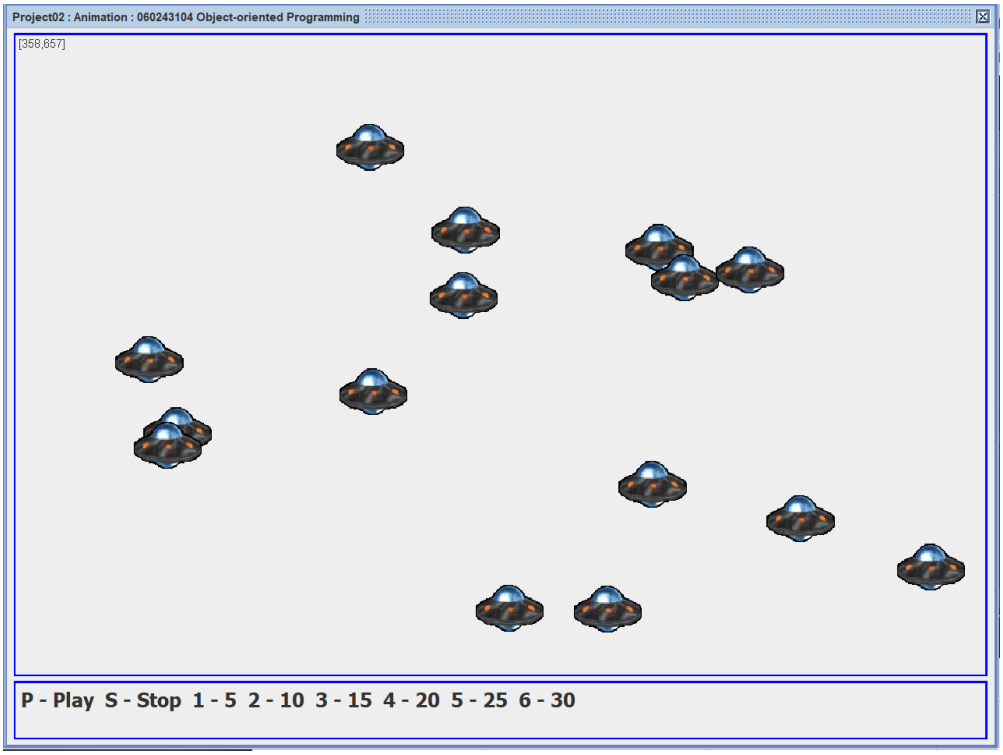
ภาพที่ 33 หน้าจอ Window โปรแกรม Java Application สำหรับการแสดงการเคลื่อนที่ของรูปภาพ



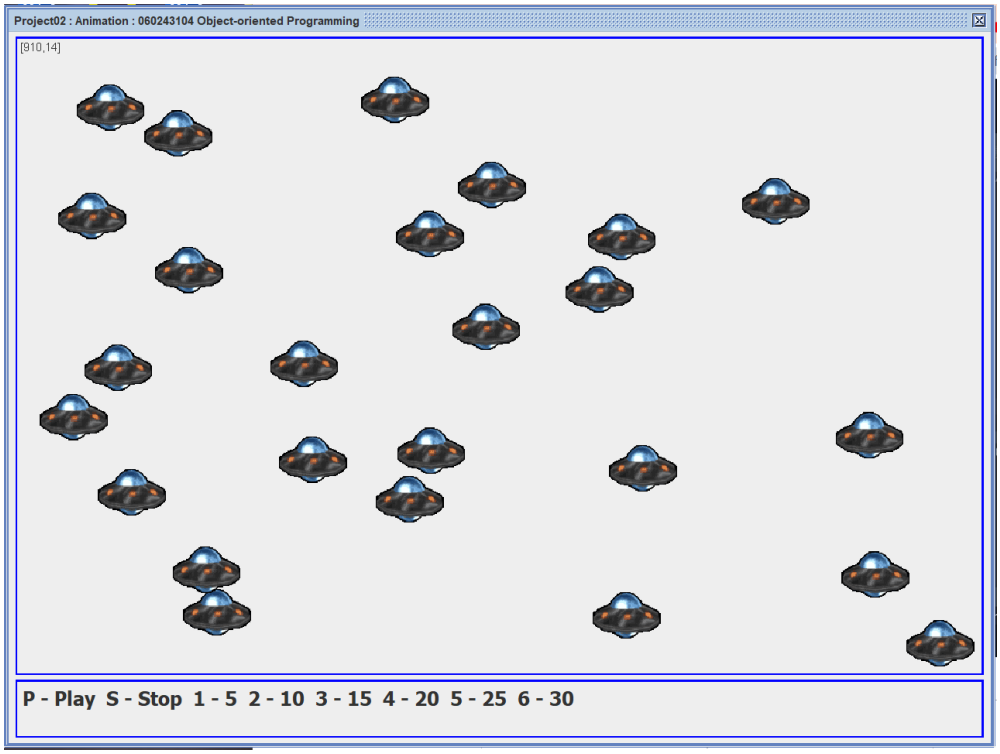
ภาพที่ 34 กดปุ่ม 1 เพื่อกำหนดจำนวน UFO 5 ลำ



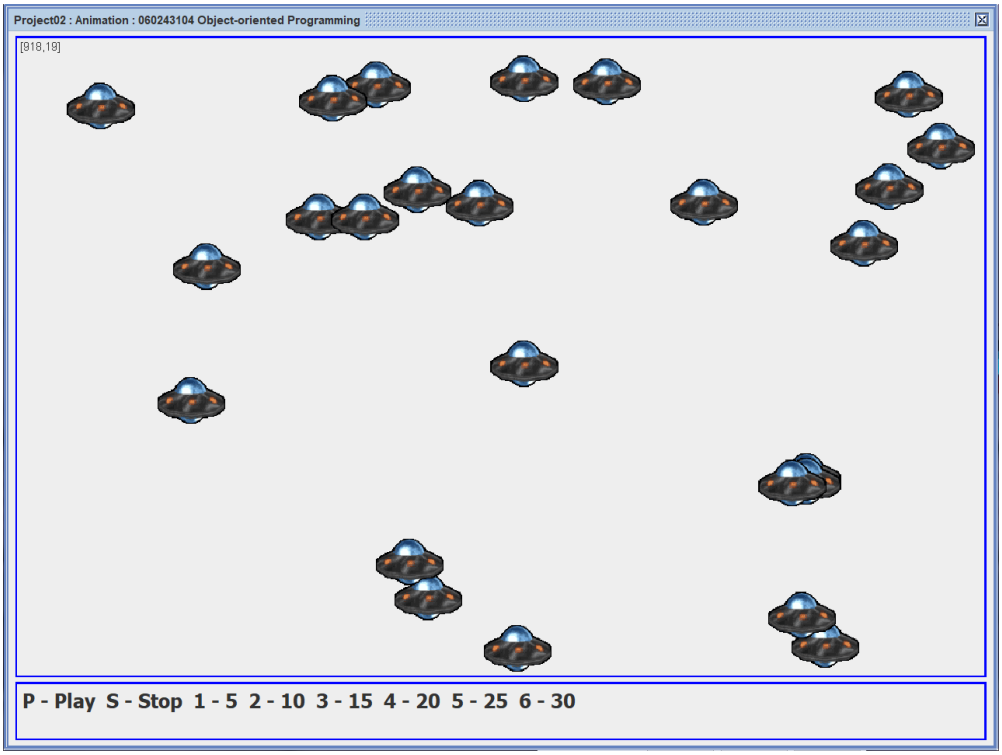
ภาพที่ 35 กดปุ่ม 2 เพื่อกำหนดจำนวน UFO 10 ลำ



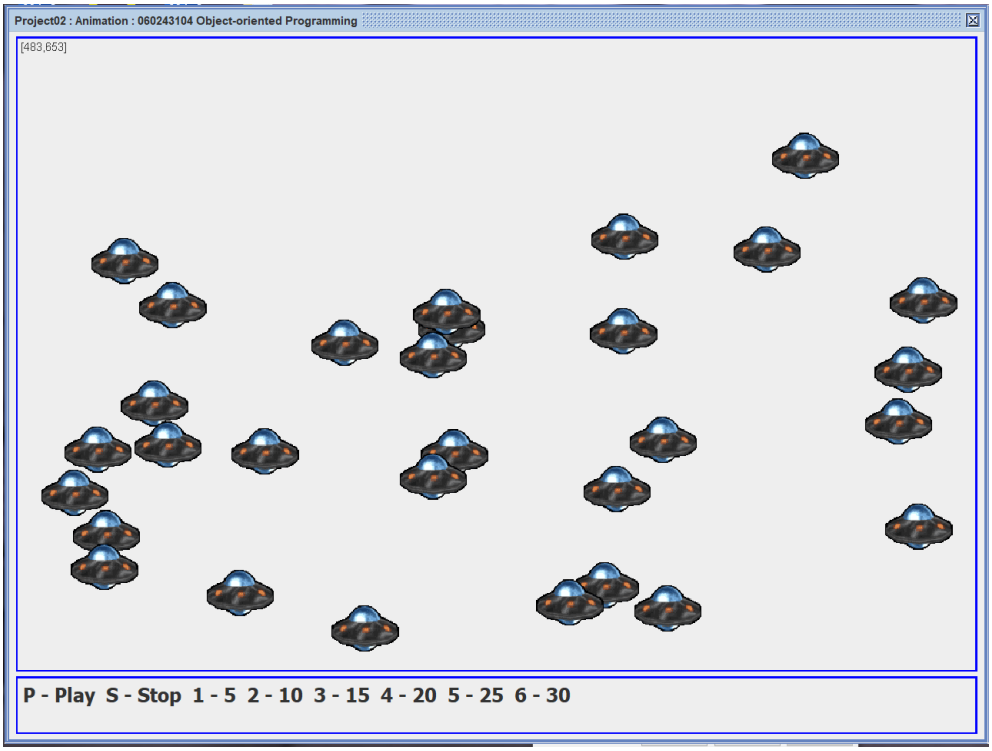
ภาพที่ 36 กดปุ่ม 3 เพื่อกำหนดจำนวน UFO 15 ลำ



ภาพที่ 37 กดปุ่ม 4 เพื่อกำหนดจำนวน UFO 20 ลำ

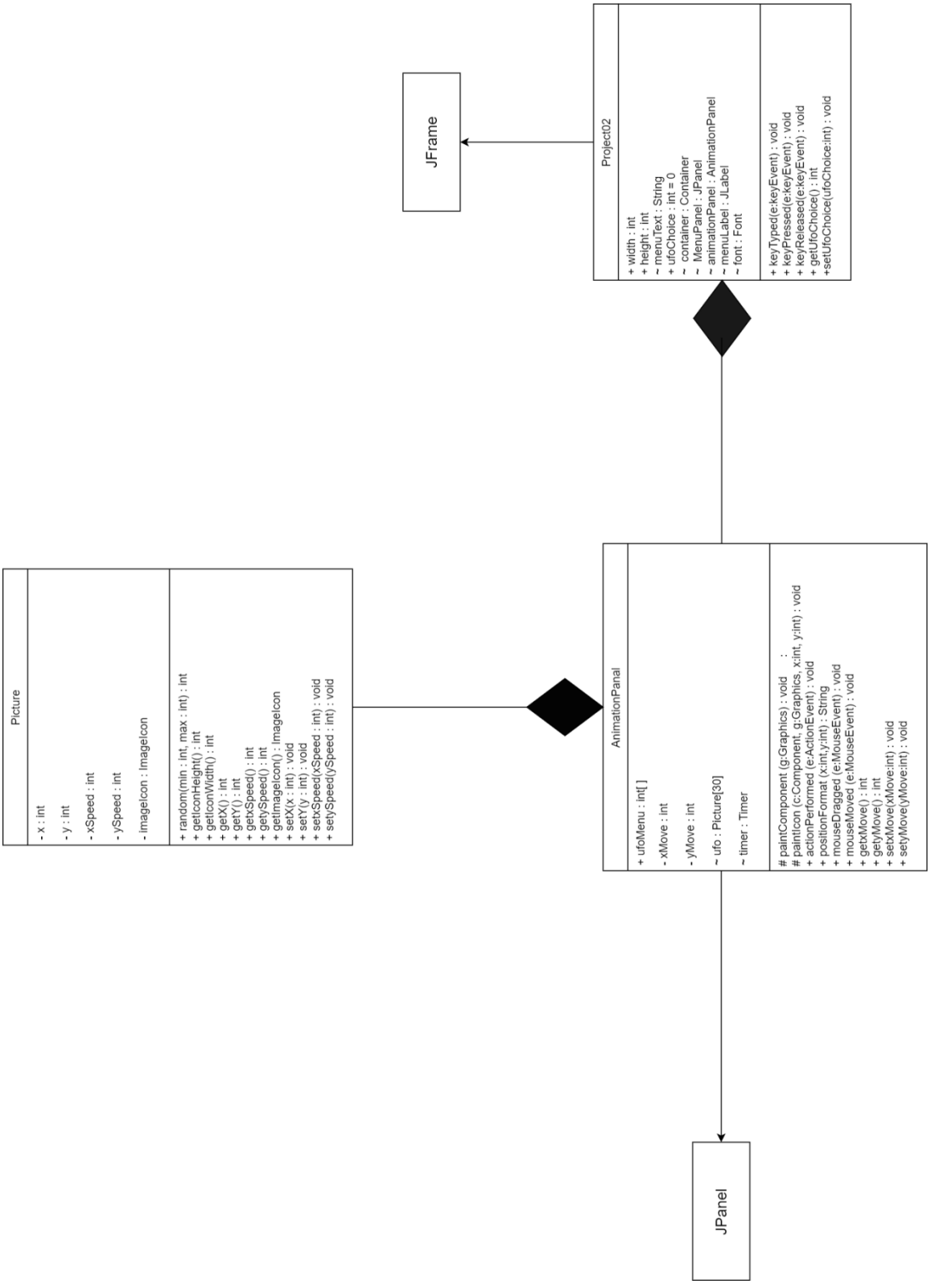


ภาพที่ 38 กดปุ่ม 5 เพื่อกำหนดจำนวน UFO 25 ลำ



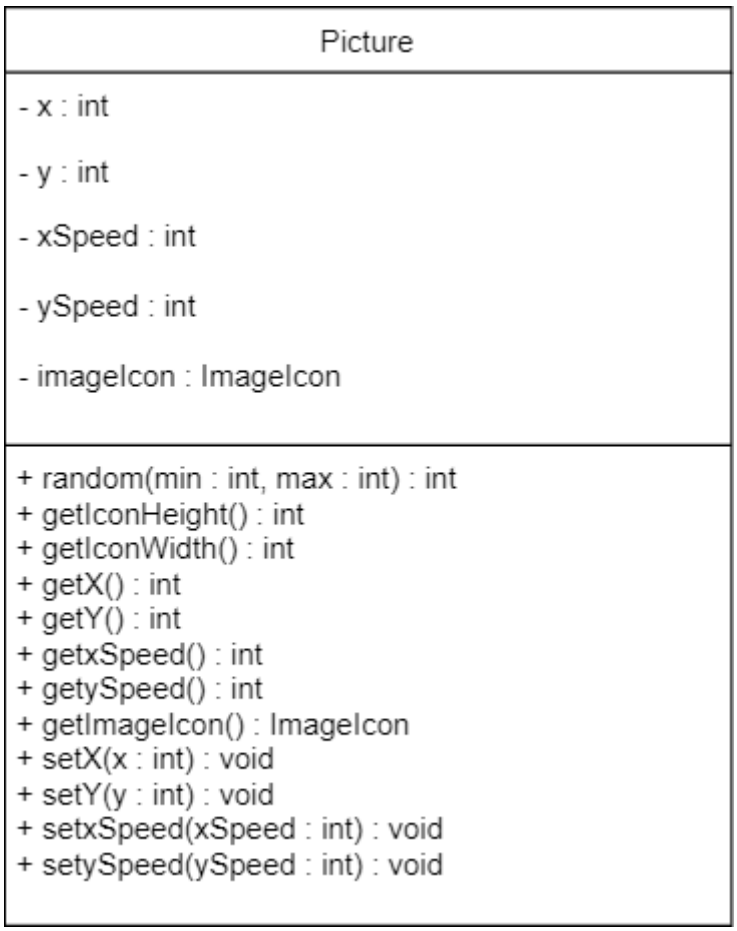
ภาพที่ 37 กดปุ่ม 6 เพื่อกำหนดจำนวน UFO 30 ลำ

Class Diagram โปรแกรม Java Application สำหรับการแสดงการเคลื่อนที่ของรูปภาพ



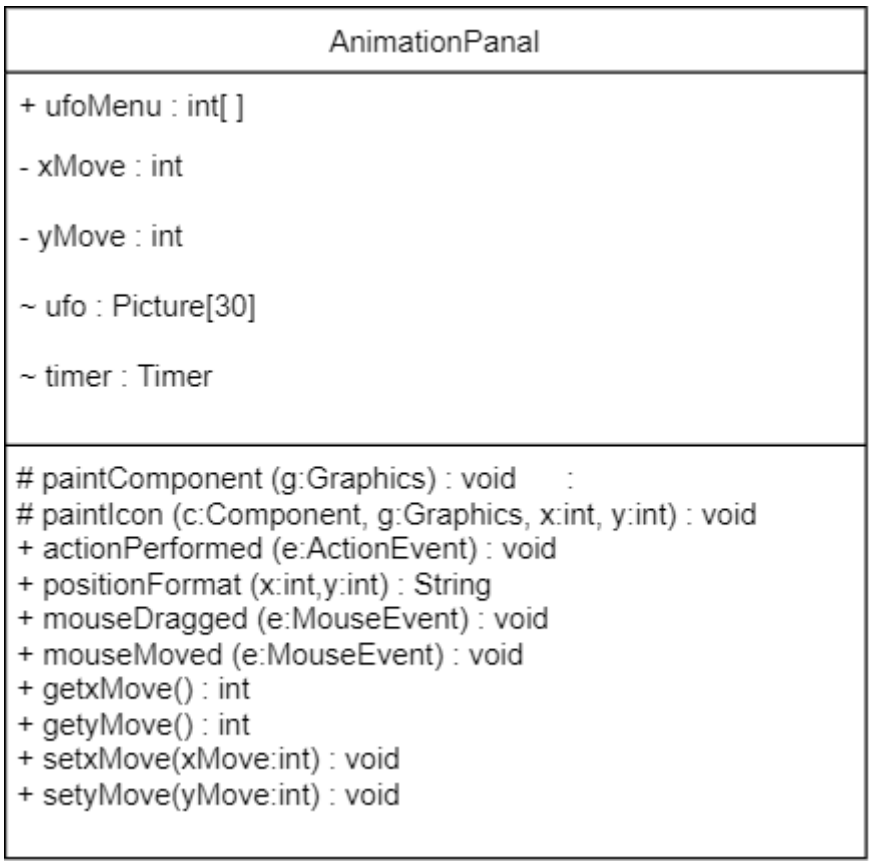
ภาพที่ 38 Class Diagram โปรแกรม Java Application สำหรับการแสดงการเคลื่อนที่ของรูปภาพ

1. Class Picture



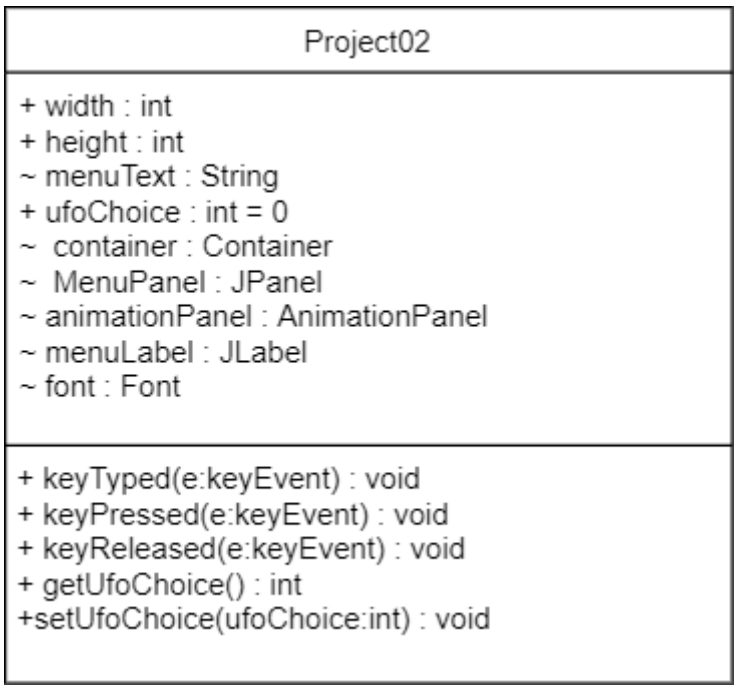
ภาพที่ 39 Class Diagram ของ Class Picture

2. Class AnimationPanel



ภาพที่ 40 Class Diagram ของ Class AnimationPanel

3. Class Project02



ภาพที่ 41 Class Diagram ของ Class Project02

Source Program Java Application สำหรับการแสดงการเคลื่อนที่ของรูปภาพ

Source Program Class Picture

```
1.  import javax.swing.*;
2.  import java.util.Random;
3.  import java.awt.*;
4.  public class Picture {
5.      private int x,y;
6.      private int xSpeed, ySpeed;
7.
8.      private final ImageIcon imageIcon;
9.
10.     //width and height is from JPanel height
11.     Picture(int width, int height){
12.         imageIcon = new ImageIcon("images//ufo.png");
13.
14.         // Set UFO Random position
15.         // random( gap, width - iconSize, gap);
16.         setX(random(5, width-imageIcon.getIconWidth() - 5 ));
17.         setY(random(5, height-imageIcon.getIconHeight() - 5 ));
18.
19.         // Set random speed. For make our UFO look COOL! :D
20.         setxSpeed(random(1,7));setySpeed(random(1,7));
21.     }
22.
23.     public int random(int min, int max){
24.         Random rand = new Random();
25.         return rand.nextInt((max - min) + 1) + min;
26.     }
27.
28.
29.
30.
31.     public int getIconHeight(){
32.         return imageIcon.getIconHeight();
33.     }public int getIconWidth(){
34.         return imageIcon.getIconWidth();
35.     }
36.
37.     public int getX() {
38.         return x;
39.     }
40.
41.     public int getY() {
42.         return y;
43.     }
44.
45.     public int getxSpeed() {
46.         return xSpeed;
47.     }
48.
49.     public int getySpeed() {
50.         return ySpeed;
51.     }
52.
53.     public ImageIcon getImageIcon() {
54.         return imageIcon;
55.     }
56.
57.     public void setX(int x) {
58.         this.x = x;
59.     }
60.
61.     public void setY(int y) {
62.         this.y = y;
```

```
63.     }
64.
65.     public void setxSpeed(int xSpeed) {
66.         this.xSpeed = xSpeed;
67.     }
68.
69.     public void setySpeed(int ySpeed) {
70.         this.ySpeed = ySpeed;
71.     }
72. }
73.
```

Source Program Class AnimationPanel

```
1.  import javax.swing.*;
2.  import java.awt.*;
3.  import java.awt.event.*;
4.  import javax.swing.Timer;
5.  import javax.swing.border.LineBorder;
6.
7.  public class AnimationPanel extends JPanel implements MouseMotionListener, ActionListene
    r {
8.      final public int[] ufoMenu = {0, 5, 10, 15, 20, 25, 30};
9.      private int xMove, yMove;
10.
11.     Picture[] ufo = new Picture[30];
12.     static Timer timer;
13.
14.     AnimationPanel(){
15.         super(true);
16.         super.setLayout(new FlowLayout());
17.         super.setPreferredSize(new Dimension(1000, 660));
18.         super.setBorder(new LineBorder(Color.BLUE,2));
19.         addMouseMotionListener(this);
20.
21.         timer = new Timer(30,this);
22.
23.         // Create UFO
24.         for(int i = 0; i < ufo.length; i++){
25.             // Picture ( PanelWidth, PanelHeight)
26.             ufo[i] = new Picture(super.getPreferredSize().width, super.getPreferredSize(
    ).height);
27.         }
28.
29.     }
30.
31.     protected void paintComponent(Graphics g){
32.         super.paintComponent(g);
33.         // Draw Mouse position in AnimationPanel.
34.         g.drawString(positionFormat(getxMove(),getyMove()),5,15);
35.
36.         // Paint ufo
37.         for(int i =0; i < ufoMenu[Project02.ufoChoice]; i++){
38.             ufo[i].getImageIcon().paintIcon(this, g, ufo[i].getX(), ufo[i].getY());
39.         }
40.
41.         super.repaint();
42.     }
43.
44.     protected void paintIcon(Component c, Graphics g, int x,int y){ }
45.
46.     // Ufo Animation Update
47.     public void actionPerformed(ActionEvent e) {
48.         for(int i =0; i < ufoMenu[Project02.ufoChoice]; i++){
49.
50.             // For make UFO don't fly out from Animation Panel.
51.
52.             //////////////////////////// Set speed x,y to positive or negative Speed  ///
    ////////////////////////////
53.             // For make UFO don't move more than width, height limit.
54.             if (ufo[i].getX() + ufo[i].getxSpeed() > super.getPreferredSize().width-
    ufo[i].getIconWidth()) {
55.                 ufo[i].setxSpeed(ufo[i].getxSpeed() * -1);
56.             }if (ufo[i].getY() + ufo[i].getySpeed() > super.getPreferredSize().height-
    ufo[i].getIconHeight() ) {
57.                 ufo[i].setySpeed(ufo[i].getySpeed() * -1);
58.             }
59.             // For make UFO don't move less than width, height limit.
60.             // 5 is mean gap.
61.             if (ufo[i].getX() + ufo[i].getxSpeed() < 5) {
```

```
62.                 ufo[i].setxSpeed(ufo[i].getxSpeed() * -1);
63.             }
64.             if (ufo[i].getY() + ufo[i].getySpeed() < 5) {
65.                 ufo[i].setySpeed(ufo[i].getySpeed() * -1);
66.             }
67.             ////////////////////////////////////////////////////////////////////////////
    ////////////////////////
68.
69.             // Move UFO position (position + speed)l
70.             ufo[i].setX( ufo[i].getX() + ufo[i].getxSpeed());
71.             ufo[i].setY( ufo[i].getY() + ufo[i].getySpeed());
72.
73.         }
74.     }
75.
76.     public static String positionFormat(int x, int y){
77.         return "["+ x + "," + y + "]";
78.     }
79.
80.     public void mouseDragged(MouseEvent e) {
81.
82.     }
83.
84.     public void mouseMoved(MouseEvent e) {
85.         setxMove(e.getX());
86.         setyMove(e.getY());
87.     }
88.
89.     public int getxMove() {
90.         return xMove;
91.     }
92.
93.     public int getyMove() {
94.         return yMove;
95.     }
96.
97.     public void setxMove(int xMove) {
98.         this.xMove = xMove;
99.     }
100.
101.      public void setyMove(int yMove) {
102.          this.yMove = yMove;
103.      }
104.
105.  }
106.
```

Source Program Class Project02

```
1.  import javax.swing.*;
2.  import java.awt.*;
3.  import java.awt.event.KeyEvent;
4.  import java.awt.event.KeyListener;
5.  import java.awt.Font;
6.  import javax.swing.border.LineBorder;
7.
8.  public class Project02 extends JFrame implements KeyListener {
9.      public int width, height;
10.     final String menuText = "P - Play  S - Stop  1 - 5  2 - 10  3 - 15  4 - 20  5 -
     25  6 - 30";
11.     public static int ufoChoice = 0;
12.
13.     // Get user screen resolution
14.     Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
15.
16.     Container container;
17.     JPanel menuPanel;
18.     AnimationPanel animationPanel;
19.     JLabel menuLabel;
20.     Font font = new Font("Tahoma",Font.BOLD,20);
21.
22.     Project02(){
23.         super("Project02 : Animation : 060243104 Object-oriented Programming");
24.         //set default app size
25.         width = 1024; height = 768;
26.
27.         super.setUndecorated(true);
28.         super.getRootPane().setWindowDecorationStyle(2);
29.         super.getRootPane().setFont(font);
30.         JFrame.setDefaultLookAndFeelDecorated(true);
31.         addKeyListener(this);
32.
33.         container = super.getContentPane();
34.         container.setLayout(new FlowLayout());
35.
36.         //Create animationPanel
37.         animationPanel = new AnimationPanel();
38.
39.         //Create Menu Panel
40.         menuPanel = new JPanel();
41.         menuPanel.setLayout(new FlowLayout(FlowLayout.LEFT));
42.         menuPanel.setPreferredSize(new Dimension(1000,60));
43.         menuPanel.setBorder(new LineBorder(Color.BLUE,2));
44.
45.         menuLabel = new JLabel();
46.         menuLabel.setText(menuText);
47.         menuLabel.setFont(font);
48.         menuPanel.add(menuLabel);
49.
50.         container.add(animationPanel);
51.         container.add(menuPanel);
52.
53.         super.setSize(width,height);
54.         setLocation(
55.                 //set window position to center
56.                 //screenSize/2 - windowWidth/2
57.                 (int)screenSize.getWidth()/2-(super.getWidth()/2),
58.                 (int)screenSize.getHeight()/2-(super.getHeight()/2) );
59.         super.setVisible(true);
60.         super.setResizable(false);
61.         super.setDefaultCloseOperation(EXIT_ON_CLOSE);
62.     }
63.
64.     public void keyTyped(KeyEvent e) {
65.
```

```
66.     }
67.
68.     // Press Key Event
69.     public void keyPressed(KeyEvent e) {
70.         System.out.println("\n-------------------------------------");
71.         System.out.println("keyPressed Code: " + e.getKeyCode() );
72.         System.out.println("keyPressed Char: " + e.getKeyChar() );
73.         System.out.println("-------------------------------------");
74.
75.         // Set how many UFO in screen.
76.         if(     // 49-54 are number on keyboard, 97-105 are number on numpad
77.                 // This is key number 1-6
78.                 ( e.getKeyCode() >= 49 ) && ( e.getKeyChar() <= 54) ||
79.                 ( e.getKeyCode() >= 97 ) && ( e.getKeyChar() <= 105))
80.         setUfoChoice( Integer.parseInt(String.valueOf(e.getKeyChar())) );// end of if
81.
82.         // Set UFO start and stop.
83.         if (Character.toString(e.getKeyChar()).toUpperCase().equals("P")){
84.             AnimationPanel.timer.start();
85.         }else if (Character.toString(e.getKeyChar()).toUpperCase().equals("S")){
86.             AnimationPanel.timer.stop();
87.         }
88.     }
89.
90.     public void keyReleased(KeyEvent e) {
91.
92.     }
93.
94.     public int getUfoChoice() {
95.         return ufoChoice;
96.     }
97.
98.     public void setUfoChoice(int ufoChoice) {
99.         Project02.ufoChoice = ufoChoice;
100.      }
101.
102.     public static void main(String[] args) {
103.         new Project02();
104.     }
105. }
106.
```