Assignment-1

COMPUTER GRAPHICS LAB

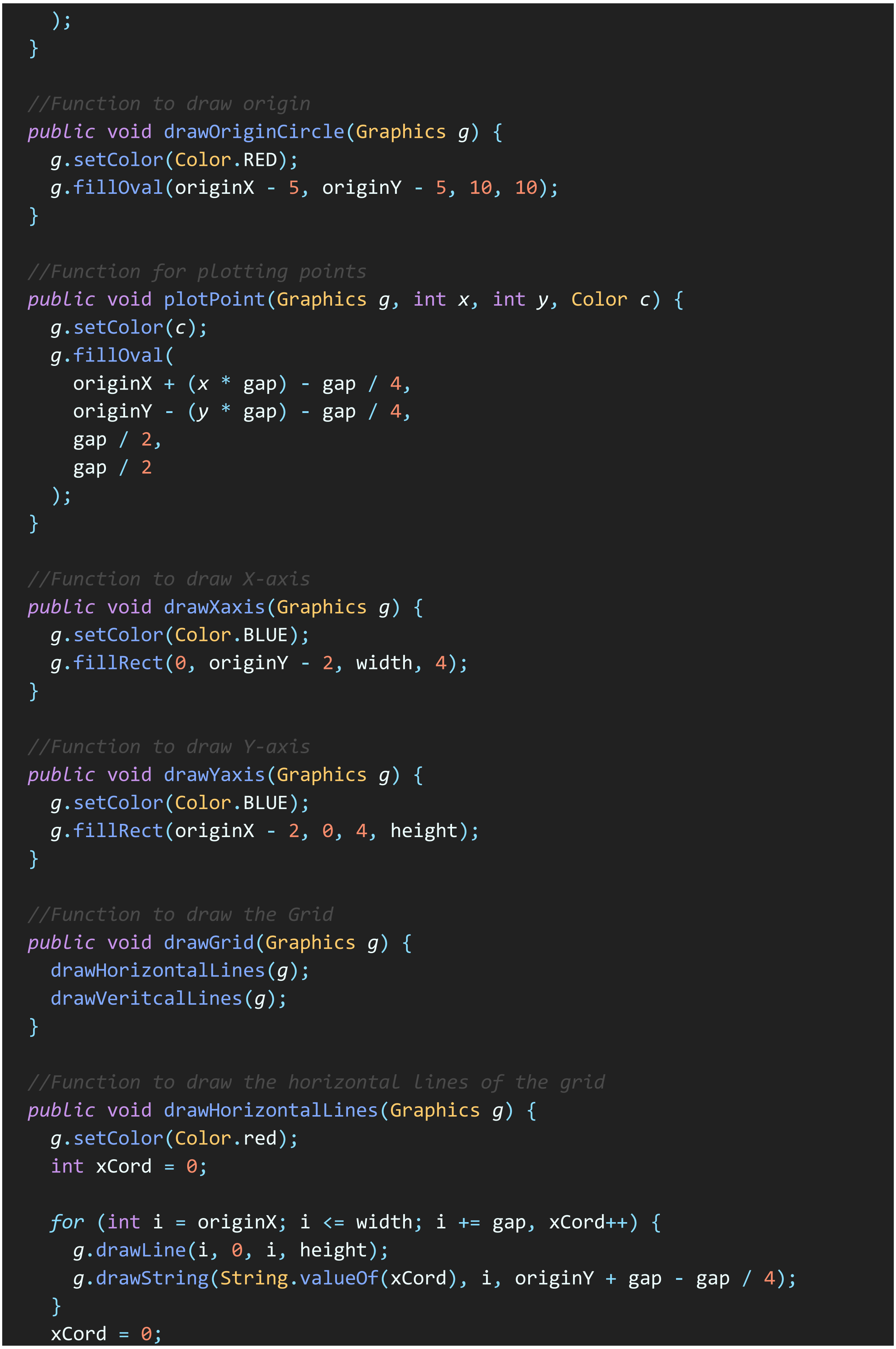
Subhajit Samanta

2020CSB046

➢ PART-1

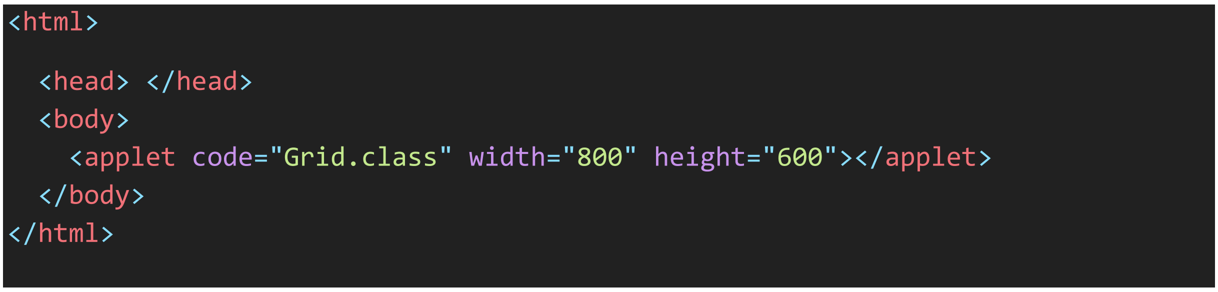
○ Grid.java

```
 import java.applet.*;

import java.awt.*;
import java.awt.event.*;

public class Grid extends Applet implements ActionListener, MouseWheelListener
{

  int originX, originY;
  int height, width;
  int gap = 20;
  Button b1 = new Button(" + ");
  Button b2 = new Button(" - ");

  public void init() {
    setBackground(new Color(232, 249, 253));
    b1.setBackground(new Color(31, 70, 144));
    b2.setBackground(new Color(255, 229, 180));
    add(b1);
    add(b2);
    addMouseWheelListener(this);
    b1.addActionListener(this);
    b2.addActionListener(this);
  }

  public void paint(Graphics g) {
    g.setColor(Color.BLACK);
    height = getHeight();
    width = getWidth();
    originX = (getX() + width) / 2;
    originY = (getY() + height) / 2;
    drawXaxis(g);
    drawYaxis(g);
    drawOriginCircle(g);

    drawGrid(g);
    g.setColor(Color.green);
    g.fillOval(
      originX + (1 * gap) - gap / 4,
      originY - (1 * gap) - gap / 4,
      gap / 2,
      gap / 2
```



```
  );
}

//Function to draw origin
public void drawOriginCircle(Graphics g) {
  g.setColor(Color.RED);
  g.fillOval(originX - 5, originY - 5, 10, 10);
}

//Function for plotting points
public void plotPoint(Graphics g, int x, int y, Color c) {
  g.setColor(c);
  g.fillOval(
    originX + (x * gap) - gap / 4,
    originY - (y * gap) - gap / 4,
    gap / 2,
    gap / 2
  );
}

//Function to draw X-axis
public void drawXaxis(Graphics g) {
  g.setColor(Color.BLUE);
  g.fillRect(0, originY - 2, width, 4);
}

//Function to draw Y-axis
public void drawYaxis(Graphics g) {
  g.setColor(Color.BLUE);
  g.fillRect(originX - 2, 0, 4, height);
}

//Function to draw the Grid
public void drawGrid(Graphics g) {
  drawHorizontalLines(g);
  drawVeritcalLines(g);
}

//Function to draw the horizontal lines of the grid
public void drawHorizontalLines(Graphics g) {
  g.setColor(Color.red);
  int xCord = 0;

  for (int i = originX; i <= width; i += gap, xCord++) {
    g.drawLine(i, 0, i, height);
    g.drawString(String.valueOf(xCord), i, originY + gap - gap / 4);
  }
  xCord = 0;
  for (int i = originX; i >= 0; i -= gap, xCord--) {
   g.drawLine(i, 0, i, height);
   g.drawString(String.valueOf(xCord), i, originY + gap - gap / 4);
  }
 }

 //Function to draw the vertical lines of the grid
 public void drawVeritcalLines(Graphics g) {
  g.setColor(Color.red);
  int yCord = 0;
  for (int i = originY; i <= height; i += gap, yCord--) {
   g.drawLine(0, i, width, i);
   // add coordinate text
   if (yCord != 0) g.drawString(String.valueOf(yCord), originX, i + gap -
5);
  }
  yCord = 0;
  for (int i = originY; i >= 0; i -= gap, yCord++) {
   g.drawLine(0, i, width, i);
   if (yCord != 0) g.drawString(String.valueOf(yCord), originX, i + gap -
5);
  }
 }

 //Function for the buttons
 public void actionPerformed(ActionEvent e) {
  if (e.getSource() == b1) zoom(10);
  if (e.getSource() == b2) zoom(-10);
 }

 //Function for the mousewheel
 public void mouseWheelMoved(MouseWheelEvent e) {
  int z = e.getWheelRotation();
  zoom(z);
 }

 //Function for the zoom in feature
 public void zoom(int i) {
  if (gap + i >= 5 && gap + i <= 300) {
   gap += i;
   repaint();
  }
 }
}
```

○ Grid.html



```
<html>

  <head> </head>
  <body>
    <applet code="Grid.class" width="800" height="600"></applet>
  </body>
</html>
```

○ Screenshots

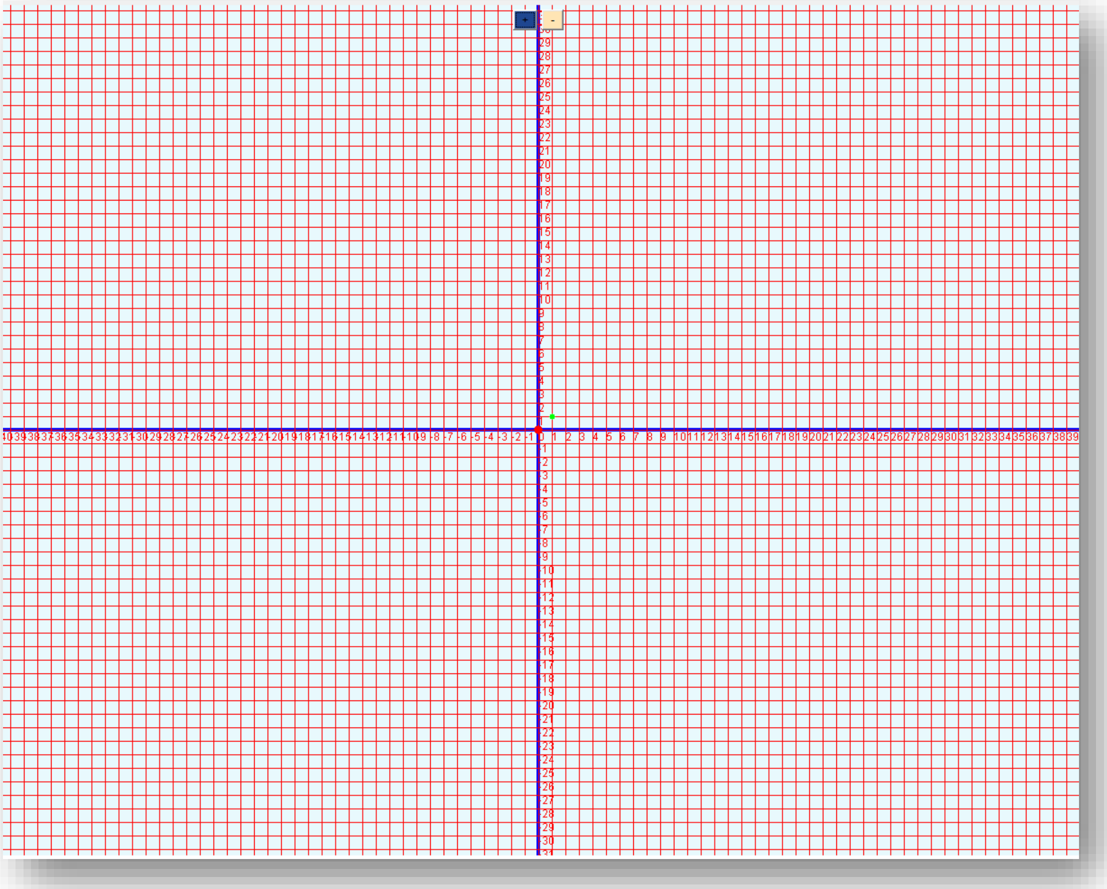


Figure 1-Zoom-Out View

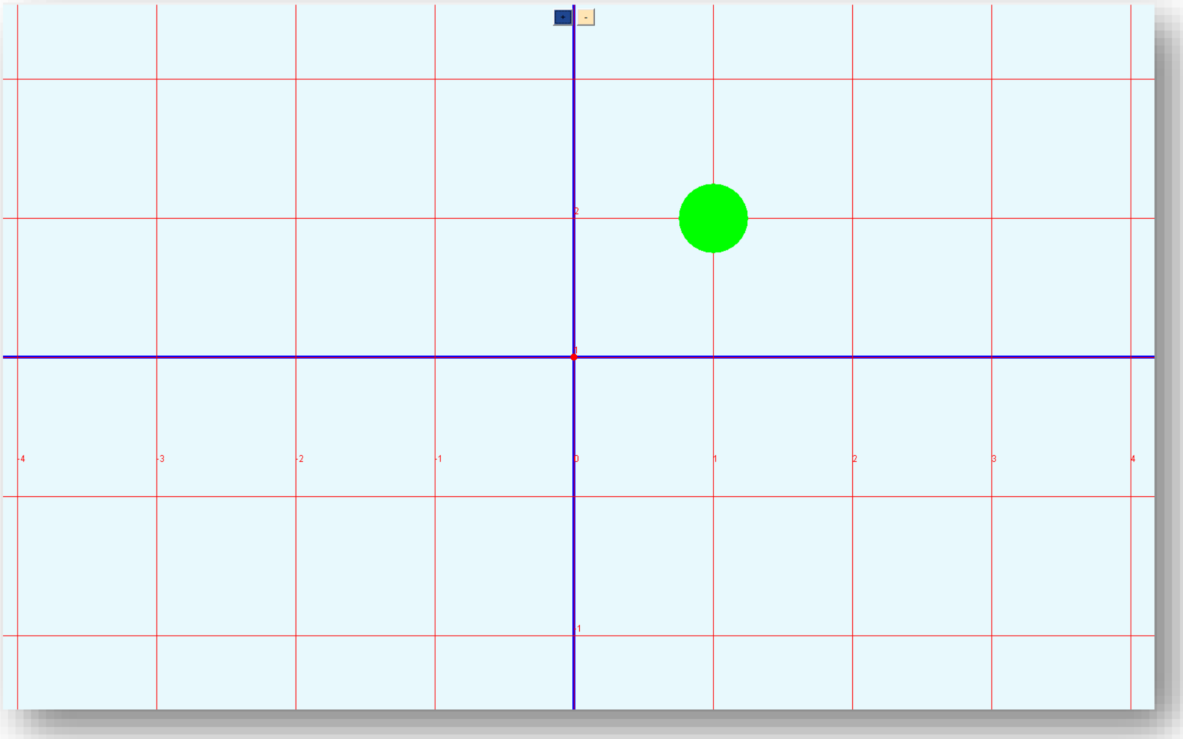
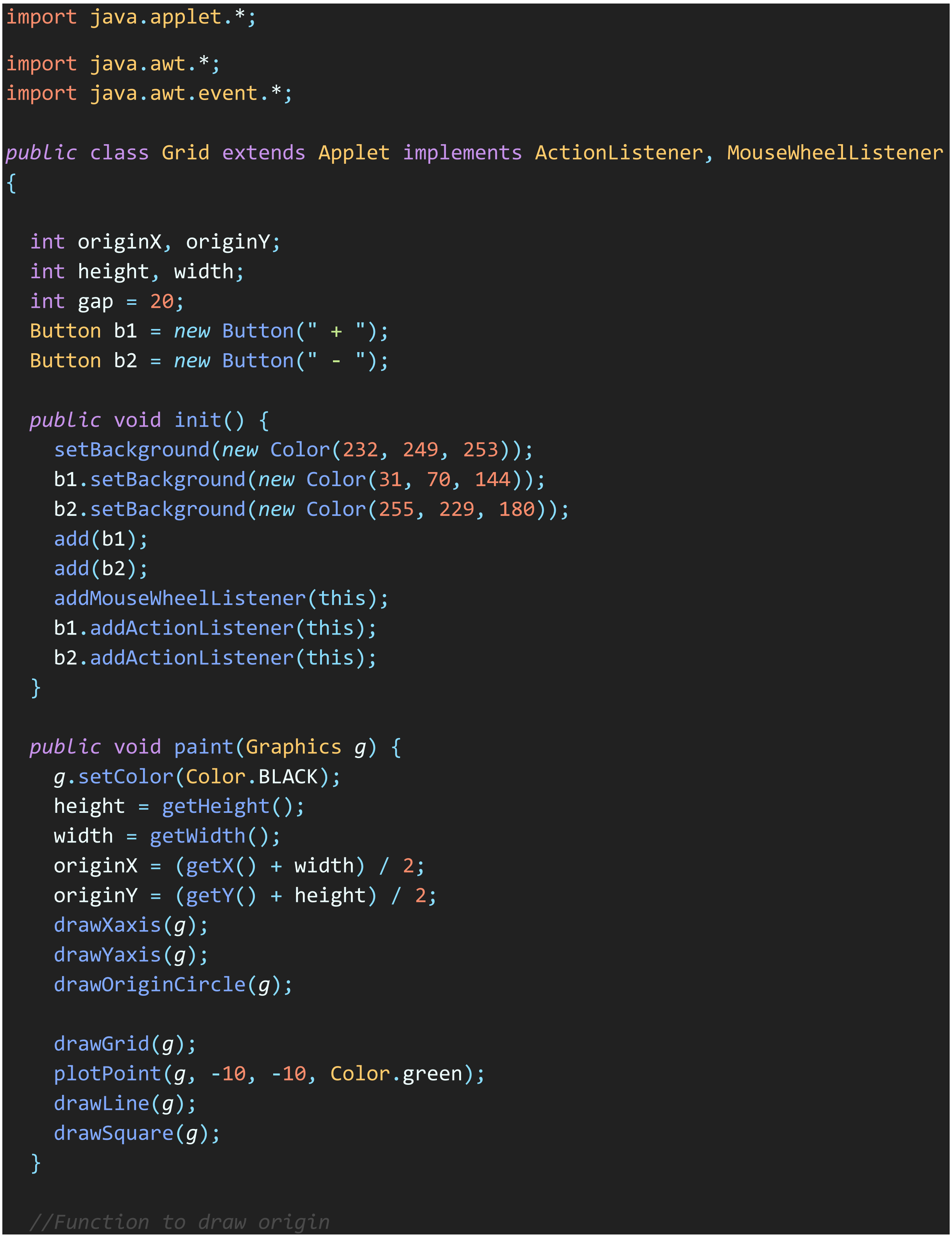


Figure 2-Zoomed-In View

➢ PART-2

○ Grid.java



```java
import java.applet.*;

import java.awt.*;
import java.awt.event.*;

public class Grid extends Applet implements ActionListener, MouseWheelListener
{

  int originX, originY;
  int height, width;
  int gap = 20;
  Button b1 = new Button(" + ");
  Button b2 = new Button(" - ");

  public void init() {
    setBackground(new Color(232, 249, 253));
    b1.setBackground(new Color(31, 70, 144));
    b2.setBackground(new Color(255, 229, 180));
    add(b1);
    add(b2);
    addMouseWheelListener(this);
    b1.addActionListener(this);
    b2.addActionListener(this);
  }

  public void paint(Graphics g) {
    g.setColor(Color.BLACK);
    height = getHeight();
    width = getWidth();
    originX = (getX() + width) / 2;
    originY = (getY() + height) / 2;
    drawXaxis(g);
    drawYaxis(g);
    drawOriginCircle(g);

    drawGrid(g);
    plotPoint(g, -10, -10, Color.green);
    drawLine(g);
    drawSquare(g);
  }

  //Function to draw origin
```

```
public void drawOriginCircle(Graphics g) {
  g.setColor(Color.RED);
  g.fillOval(originX - 5, originY - 5, 10, 10);
}

//Function for plotting points
public void plotPoint(Graphics g, int x, int y, Color c) {
  g.setColor(c);
  g.fillOval(
    originX + (x * gap) - gap / 4,
    originY - (y * gap) - gap / 4,
    gap / 2,
    gap / 2
  );
}

//Function to draw X-axis
public void drawXaxis(Graphics g) {
  g.setColor(Color.BLUE);
  g.fillRect(0, originY - 2, width, 4);
}

//Function to draw Y-axis
public void drawYaxis(Graphics g) {
  g.setColor(Color.BLUE);
  g.fillRect(originX - 2, 0, 4, height);
}

//Function to draw the Grid
public void drawGrid(Graphics g) {
  drawHorizontalLines(g);
  drawVeritcalLines(g);
}

//Function to draw the horizontal lines of the grid
public void drawHorizontalLines(Graphics g) {
  g.setColor(Color.yellow);
  int xCord = 0;

  for (int i = originX; i <= width; i += gap, xCord++) {
    g.drawLine(i, 0, i, height);
    g.drawString(String.valueOf(xCord), i, originY + gap - gap / 4);
  }
  xCord = 0;
  for (int i = originX; i >= 0; i -= gap, xCord--) {
    g.drawLine(i, 0, i, height);
    g.drawString(String.valueOf(xCord), i, originY + gap - gap / 4);
  }
  }

  //Function to draw the vertical lines of the grid
  public void drawVeritcalLines(Graphics g) {
    g.setColor(Color.yellow);
    int yCord = 0;
    for (int i = originY; i <= height; i += gap, yCord--) {
      g.drawLine(0, i, width, i);
      // add coordinate text
      if (yCord != 0) g.drawString(String.valueOf(yCord), originX, i + gap -
5);
    }
    yCord = 0;
    for (int i = originY; i >= 0; i -= gap, yCord++) {
      g.drawLine(0, i, width, i);
      if (yCord != 0) g.drawString(String.valueOf(yCord), originX, i + gap -
5);
    }
  }

  //Function for the buttons
  public void actionPerformed(ActionEvent e) {
    if (e.getSource() == b1) zoom(10);
    if (e.getSource() == b2) zoom(-10);
  }

  //Function for the mousewheel
  public void mouseWheelMoved(MouseWheelEvent e) {
    int z = e.getWheelRotation();
    zoom(z);
  }

  //Function for the zoom in feature
  public void zoom(int i) {
    if (gap + i >= 5 && gap + i <= 300) {
      gap += i;
      repaint();
    }
  }

  public void drawLine(Graphics g) {
    int x1 = 10;
    int y1 = 10;
    int x2 = 100;
    int y2 = 100;
    while (x1 != x2 && y1 != y2) {
      plotPoint(g, x1 + 1, y1 + 1, Color.red);
      x1 = x1 + 1;
      y1 = y1 + 1;
    }
  }

  public void drawSquare(Graphics g) {
    int x1 = 10;
    int y1 = 10;
    int x2 = 10;
    int y2 = 10;
    int x3 = 50;
    int y3 = 10;
    int x4=10;
    int y4=50;
    plotPoint(g,10,10,Color.green);
    while (x1 != 50) {
      plotPoint(g, x1 + 1, y1, Color.green);
      x1 = x1 + 1;
    }
    while (y2 != 50) {
      plotPoint(g, x2, y2 + 1, Color.green);
      y2 = y2 + 1;
    }

    while (y3 != 50) {
      plotPoint(g, x3, y3 + 1, Color.green);
      y3 = y3 + 1;
    }
    while (x4 != 50) {
        plotPoint(g, x4+1, y4, Color.green);
        x4 = x4 + 1;
      }
  }
}
```

○ Grid.html

```
 <html>

  <head> </head>
  <body>
    <applet code="Grid.class" width="800" height="600"></applet>
  </body>
</html>
```

○ Screenshots

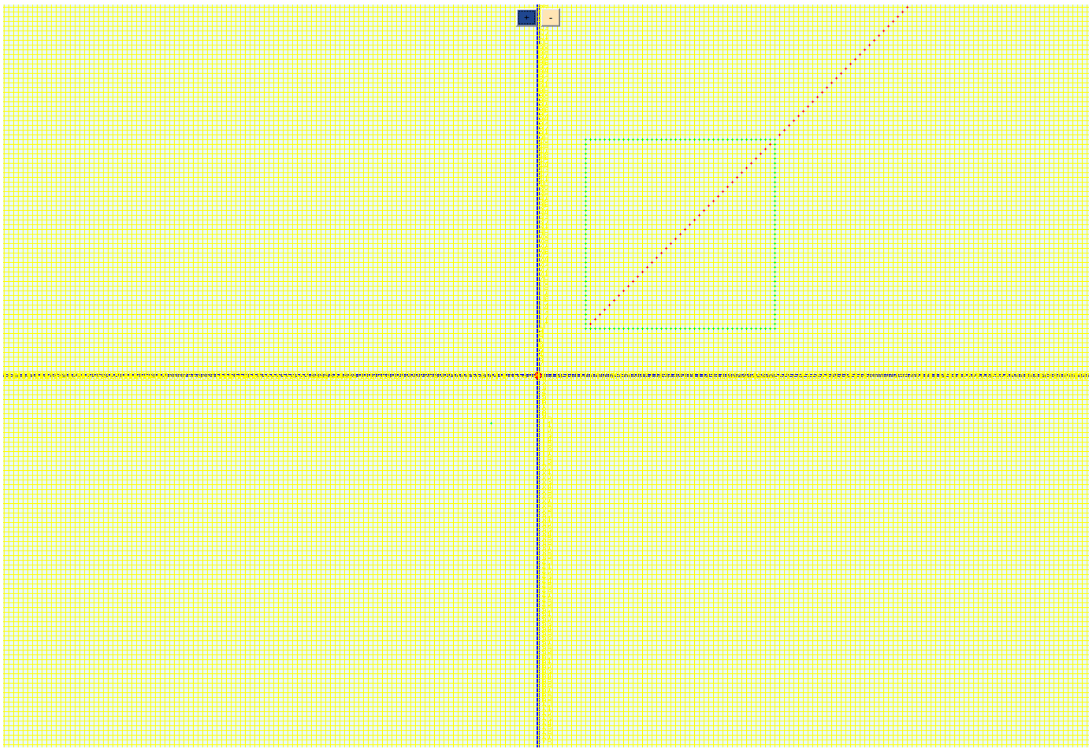


Figure 3-Square & St. Line View