Java JFrame Büyüklüğünün Ve Koordinatlarının Restore Edilmesi

Java **GUI** uygulamasında, **Frame**'in **büyüklüğünün ve ekranda gösterildiği lokasyonun**, uygulama kapatıldıktan sonra, **aynı değerlere** sahip olması için aşağıdaki kodları kullanabiliriz:

```
1   import javax.swing.*;
2   import java.awt.*;
3   import java.awt.event.WindowAdapter;
4   import java.awt.event.WindowEvent;
5   import java.io.IOException;
6
7   public class Main extends JFrame {
8       private Rectangle bounds;
9
10      public Main() {
11          setTitle("Continue From Last Location");
12          setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
13          setLayout(new BorderLayout());
14          try {
15              Rectangle dimension = (Rectangle) FileOp.readObject(FileOp.tempDirectory +
16  "lastLocation.tmp");
17              if (dimension != null) {
18                  setSize(dimension.getSize());
19                  setLocation((int) dimension.getX(), (int) dimension.getY());
20              }
21
22          } catch (IOException e) {
23              e.printStackTrace();
24          } catch (ClassNotFoundException e) {
25              e.printStackTrace();
26          }
27          setCloseListener();
28          setVisible(true);
29
30      }
31
32      /*
33       * When JFrame is closed, save last location and dimension to a file
34      */
35      private void setCloseListener() {
36          this.addWindowListener(new WindowAdapter() {
37              @Override
38              public void windowClosing(WindowEvent e) {
39                  //this method returns Rectangle object which contains x, y coordinate values and Dimension
40  object
41                  bounds = getBounds();
42                  try {
43                      FileOp.writeObject(bounds, FileOp.tempDirectory + "lastLocation.tmp");
44                  } catch (IOException e1) {
45                      e1.printStackTrace();
46                  }
47              }
48          });
49
50      }
51
52      public static void main(String[] args) {
53          new Main();
54      }
55  }
```

FileOp Class

```
/*
* Write and Read File Operations
*/
public class FileOp {
    public static final String tempDirectory = System.getProperty("java.io.tmpdir");

    public static void writeObject(Object content, String filePath) throws IOException {
```

```
        FileOutputStream fos = new FileOutputStream(filePath);
        ObjectOutputStream oos = new ObjectOutputStream(fos);
        oos.writeObject(content);
        oos.close();
    }

    public static Object readObject(String filePath) throws IOException, ClassNotFoundException {
        File file = new File(filePath);
        if (file.exists()) {
            FileInputStream fis = new FileInputStream(filePath);
            ObjectInputStream ois = new ObjectInputStream(fis);
            Object content = ois.readObject();
            ois.close();
            return content;
        } else {
            return null;
        }

    }
}
```