```java
import java.awt.*;
import java.awt.event.*;
import java.util.*;

import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;

import javax.swing.*;
import javax.swing.table.*;

/**
 *
 * @author Ayisha S.R. Sowkathali, Sifaben Vahora
 *
 *          ReminderFile.java
 *
 */
public class ReminderFile {

    // creates a window
    JFrame jfrm = new JFrame("Appointment Reminder Application");
    JLabel label;
    DefaultTableModel model;
    Calendar cal = new GregorianCalendar();
    JTable table;
    String month;
    int year;
    int day;
    JTextArea appointmentArea = new JTextArea(10, 10);
    JButton okButton = new JButton("Add Appointment");

    // list of appointments
    ArrayList<SetAppointment> appointmentList;
    String fileName = "data.txt";
    CellRenderer cr;

    ReminderFile() {
        loadData();
        jfrm.setLayout(new BorderLayout());

        // Making buttons
        okButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                saveAppointment();
            }

        });
        jfrm.add(okButton, BorderLayout.SOUTH);
        label = new JLabel();
        label.setHorizontalAlignment(SwingConstants.CENTER);

        JButton b1 = new JButton("<Previous");
        b1.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                cal.add(Calendar.MONTH, -1);
                updateMonth();
            }
        });
        JButton b2 = new JButton("Next>");
        b2.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                cal.add(Calendar.MONTH, +1);
                updateMonth();
            }
        });

        JPanel panel = new JPanel();
        panel.setLayout(new FlowLayout());
        panel.add(b1, FlowLayout.LEFT);
        panel.add(label, FlowLayout.CENTER);
        panel.add(b2, FlowLayout.RIGHT);

        String[] columns = { "Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat" };
        model = new DefaultTableModel(null, columns);
        table = new JTable(model);

        table.addMouseListener(new MouseAdapter() {
            @Override
            public void mouseClicked(final MouseEvent e) {
                if (e.getClickCount() == 1) {
                    final JTable jTable = (JTable) e.getSource();
                    try {
                        day = (Integer) jTable.getValueAt(jTable.getSelectedRow(),
jTable.getSelectedColumn());
                        displayAppointment();
                    } catch (Exception ex) {
                        day = -1;// empty cal cell
                    }
                }
            }

        });

        JScrollPane tablePane = new JScrollPane(table);
        JScrollPane appointmentAreaPane = new JScrollPane(appointmentArea);
        JPanel middlePanel = new JPanel(new GridLayout(1, 2, 2, 20));
        middlePanel.add(tablePane);
        middlePanel.add(appointmentAreaPane);

        panel.setBackground(Color.LIGHT_GRAY);

        jfrm.add(panel, BorderLayout.NORTH);
        jfrm.add(middlePanel, BorderLayout.CENTER);
```

```java

        this.updateMonth();

        jfrm.pack();
        jfrm.setResizable(true);
        jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        jfrm.setVisible(true);
    }

    /*
     * Method to display appointment
     */
    private void displayAppointment() {
        if (day == -1) {
            return;
        }
        appointmentArea.setText("");
        SetAppointment app = new SetAppointment(month, year, day, "");
        if (appointmentList.contains(app)) {
            int index = appointmentList.indexOf(app);
            appointmentArea.setText(appointmentList.get(index).appointment);
        }
    }

    /*
     * Method to save appointment set to the file for later use
     */
    private void saveAppointment() {
        String appData = appointmentArea.getText();

        if (appData == null || appData.trim().length() == 0) {
            JOptionPane.showMessageDialog(null, "Enter Appointment Details!");
            return;
        }
        if (day == -1) {
            JOptionPane.showMessageDialog(null, "Select correct date!");
            return;
        }

        SetAppointment app = new SetAppointment(month, year, day, "");
        int index = appointmentList.indexOf(app);

        if (index >= 0) {
            appointmentList.get(index).appointment = appointmentArea.getText();
        } else {
            app = new SetAppointment(month, year, day, appointmentArea.getText());
            appointmentList.add(app);
        }
        writeFile();
        table.repaint();
        JOptionPane.showMessageDialog(null, "Appointment Added!");

    }

    /*
     * Method to update month according to which button is clicked.
     */
    void updateMonth() {
        cal.set(Calendar.DAY_OF_MONTH, 1);
        month = cal.getDisplayName(Calendar.MONTH, Calendar.LONG, Locale.US);
        year = cal.get(Calendar.YEAR);
        label.setText(month + " " + year);

        int startDay = cal.get(Calendar.DAY_OF_WEEK);
        int numberOfDays = cal.getActualMaximum(Calendar.DAY_OF_MONTH);
        int weeks = cal.getActualMaximum(Calendar.WEEK_OF_MONTH);

        model.setRowCount(0);
        model.setRowCount(weeks);

        int date = startDay - 1;
        for (int day = 1; day <= numberOfDays; day++) {
            int r = date / 7;
            int c = date % 7;
            model.setValueAt(day, r, c);

            date++;
        }
        for (int i = 0; i < table.getColumnModel().getColumnCount(); i++) {
            TableColumn col = table.getColumnModel().getColumn(i);
            cr = new CellRenderer(appointmentList);
            SetAppointment app = new SetAppointment(month, year, day, "");
            cr.setCurrent(app);
            col.setCellRenderer(cr);
        }
    }

    /*
     * Method to load data from file
     */
    private void loadData() {
        try {

            FileInputStream fis = new FileInputStream(fileName);
            ObjectInputStream ois = new ObjectInputStream(fis);
            appointmentList = extracted(ois);
            ois.close();

        } catch (Exception fne) {
            appointmentList = new ArrayList<SetAppointment>();
        }
    }

    /*
     * Method to extract data from Array
     */
    @SuppressWarnings("unchecked")
    private ArrayList<SetAppointment> extracted(ObjectInputStream ois) throws IOException,
ClassNotFoundException {
        return (ArrayList<SetAppointment>) ois.readObject();
    }

    /*
     * Method to write data to the file
     */
    private void writeFile() {
        try {
            FileOutputStream fos = new FileOutputStream(fileName);
            ObjectOutputStream oos = new ObjectOutputStream(fos);
            oos.writeObject(appointmentList);
            oos.close();
        } catch (Exception e) {

        }
    }
}

```