Veri Yapıları Ödevi

Kodlarımı muzikcalar package içinde tuttum. 4 sınıf oluşturdum. SongNode sınfı, MusicPlayer sınıfı ve main sınıfı ve gui sınıfı.

1-SongNode sınıfı

SongNode sınıfı her şarkıyı temsil eden düğümleri içeriyor. Her bir SongNode nesnesi şarkı adı, dosya yolu, bir önceki şarkıya ve bir sonraki şarkıya referans tutuyor.

SongNode sınıfı için kodlarım:

```
package muzikcalar;

public class SongNode {
   String sarkiAdi; // Şarkının adı
   String dosyaYolu; // Şarkının dosya yolu
   SongNode oncekiSarki // Önceki şarkıyı işaret eden düğüm
   SongNode sonrakiSarki; // Sonraki şarkıyı işaret eden düğüm
   public SongNode(String sarkiAdi, String dosyaYolu) {    //yapıcı metod
      this.sarkiAdi = sarkiAdi;
      this.dosyaYolu = dosyaYolu;
      this.oncekiSarki = null;
      this.sonrakiSarki = null;
   }
}
```

2-MusicPlayer sınıfı

MusicPlayer sınıfı, çalma listesini yönetir ve şarkıları eklemek, çalmak, ileri/geri gitmek, çalma listesinin başına dönmek gibi işlemleri gerçekleştiriyor.

MusicPlayer sınıfı için kodlarım:

```
package muzikcalar;

public class MusicPlayer {
  private SongNode mevcutSarki; // Şu anda çalınan şarkıdır
  private SongNode ilkSarki; // Listenin başındaki şarkıdır
  public MusicPlayer() {    // Yapıcı metod: MusicPlayer nesnesini oluşturur
    this.mevcutSarki = null;
    this.ilkSarki = null;
  }

  // Şarkı ekleme metodu
  public void sarkiEkle(String sarkiAdi, String dosyaYolu) {
    SongNode yeniSarki = new SongNode(sarkiAdi, dosyaYolu);
    if (ilkSarki == null) {
      ilkSarki = yeniSarki;
      mevcutSarki = yeniSarki;
    }
    else {
      SongNode sonSarki = ilkSarki;
      while (sonSarki.sonrakiSarki != null) {
          sonSarki = sonSarki.sonrakiSarki;
      }
      sonSarki.sonrakiSarki = yeniSarki;
      yeniSarki.oncekiSarki = sonSarki;
    }
```

```
}

// Mevcut şarkıyı oynatma metodu
public void mevcutSarkiyiOynat() {
    if (mevcutSarki != null) {
        System.out.println("Çalan Şarkı: " + mevcutSarki.sarkiAdi);
    }
    else {
        System.out.println("Mevcut şarkı yok");
    }
}

// Sonraki şarkıya geçme metodu
public void sonrakiSarki() {
    if (mevcutSarki != null && mevcutSarki.sonrakiSarki != null) {
        mevcutSarki = mevcutSarki.sonrakiSarki;
        mevcutSarkiyiOynat();
    }
    else {
        System.out.println("Sonraki şarkı yok.");
    }
}
```

```
// Önceki şarkıya dönme metodu
  public void oncekiSarki() {
    if (mevcutSarki != null && mevcutSarki.oncekiSarki != null) {
      mevcutSarki = mevcutSarki.oncekiSarki;
      mevcutSarkiyiOynat();
    } else {
      System.out.println("Önceki şarkı yok.");
    }
  }

  // Listenin başına dönme metodu
  public void listeninBasinaAl() {
    mevcutSarki = ilkSarki;
    mevcutSarkiyiOynat();
  }
}
// Mevcut şarkı adını döndüren getter metodu
   public String getMevcutSarkiAdi() {
    if (mevcutSarki != null) {
      return mevcutSarki.sarkiAdi;
    } else {
      return "-";
    }
  }
```

3- Main Sınıfı

MusicPlayer nesnemizi bu sınıfta tanımlayıp kodumuzu bu sınıfta çalıştıracağız.

Main sınıfımızın kodları:

```
package muzikcalar;

public class Main {
   public static void main(String[] args) {
      new MusicPlayerGUI();
   }
}
```

4-GUI sınıfı

```
package muzikcalar;


import javax.swing.*;

import java.awt.*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.File;
```

```
public class MusicPlayerGUI extends JFrame {
    private MusicPlayer musicPlayer;
    private DefaultListModel<String> playlistModel;
    private JList<String> playlist;
    private JLabel currentSongLabel;

    public MusicPlayerGUI() {
        musicPlayer = new MusicPlayer();
        setTitle("Bağlı Liste ile Basit Müzik Çalar");
        setSize(500, 400);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        setLayout(new BorderLayout());

        // Playlist
        playlistModel = new DefaultListModel<>();
        playlist = new JList<>(playlistModel);
        add(new JScrollPane(playlist), BorderLayout.CENTER);

        //mevcut şarkı labelı
        currentSongLabel = new JLabel("Çalan Şarkı: Yok");
        currentSongLabel.setHorizontalAlignment(JLabel.CENTER);
        add(currentSongLabel, BorderLayout.NORTH);

        // kontrol paneli
        JPanel controlPanel = new JPanel();
        JButton playButton = new JButton("Oynat");
        JButton stopButton = new JButton("Durdur");
```

```
        JButton nextButton = new JButton("Sonraki");
        JButton previousButton = new JButton("Önceki");

        controlPanel.add(playButton);
        controlPanel.add(stopButton);
        controlPanel.add(nextButton);
        controlPanel.add(previousButton);
        add(controlPanel, BorderLayout.SOUTH);

        // Şarkı ekleme paneli (manuel giriş ve dosya seçici seçenekleri internetten dosya seçme kısmını öğrendim)
        JPanel addSongPanel = new JPanel();
        addSongPanel.setLayout(new GridLayout(3, 2));

        JLabel songNameLabel = new JLabel("Şarkı Adı:");
        JTextField songNameField = new JTextField();
        JLabel songPathLabel = new JLabel("Dosya Yolu:");
        JTextField songPathField = new JTextField();
        JButton addButton = new JButton("Şarkı Ekle");
        JButton browseButton = new JButton("Dosya Seç");

        addSongPanel.add(songNameLabel);
        addSongPanel.add(songNameField);
        addSongPanel.add(songPathLabel);
        addSongPanel.add(songPathField);
        addSongPanel.add(browseButton);
        addSongPanel.add(addButton);
```

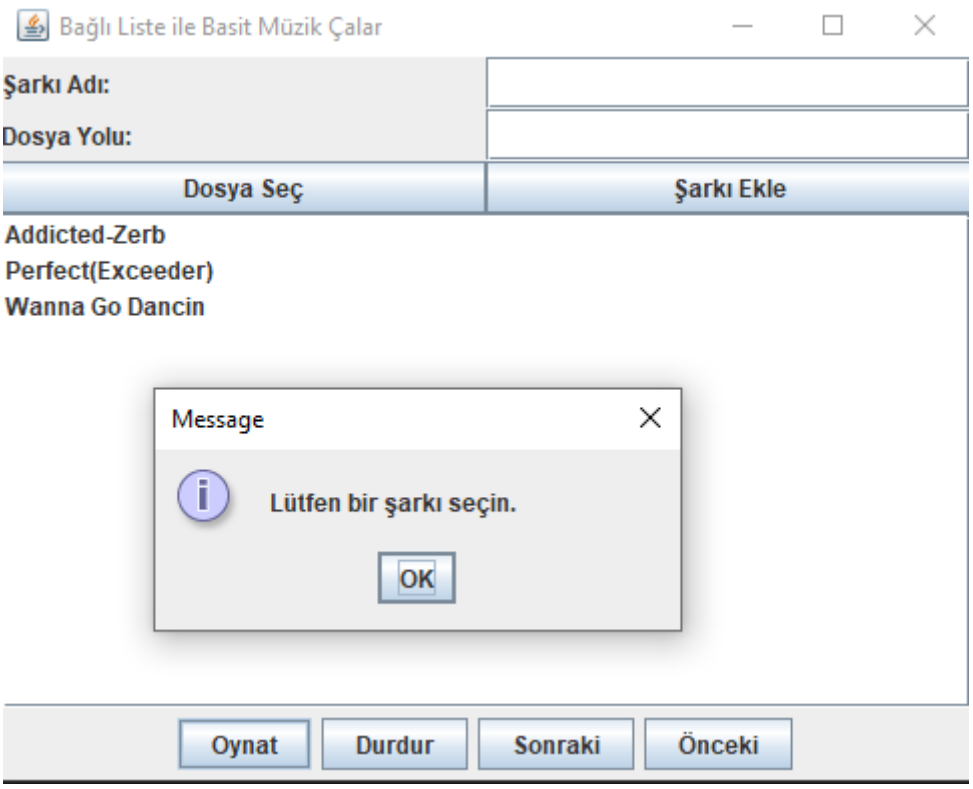
```
        add(addSongPanel, BorderLayout.NORTH);


        // "Dosya Seç" butonuna basıldığında dosya seçici açılıyor ve dosya yolu otomatik dolduruyor kendisi
        browseButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                JFileChooser fileChooser = new JFileChooser();
                fileChooser.setDialogTitle("Şarkı Seçin");
                int selection = fileChooser.showOpenDialog(null);
                if (selection == JFileChooser.APPROVE_OPTION) {
                    File selectedFile = fileChooser.getSelectedFile();
                    songNameField.setText(selectedFile.getName()); // Dosya adını şarkı adı alanına ekle
                    songPathField.setText(selectedFile.getAbsolutePath()); // Dosya yolunu dosya yolu alanına ekle
                }
            }
        });


        // Şarkı ekle butonuna basıldığında manuel giriş veya dosya seçici ile ekleme yapılmasını sağlıyor
        addButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                String songName = songNameField.getText().trim();
                String songPath = songPathField.getText().trim();
```

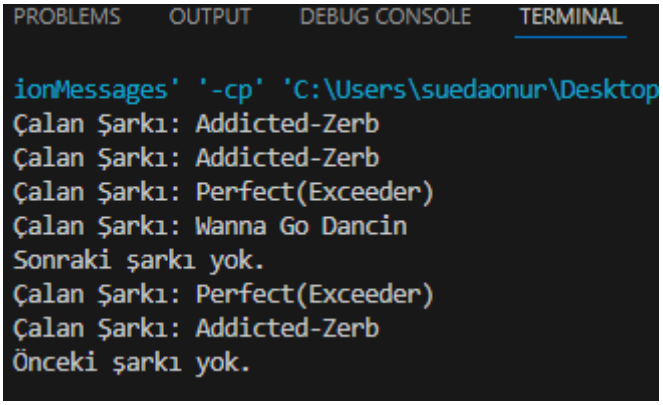
```
  if (!songName.isEmpty() && !songPath.isEmpty()) {
                playlistModel.addElement(songName); // Şarkıyı listeye ekle
                musicPlayer.sarkiEkle(songName, songPath); // Şarkıyı MusicPlayer'a ekle
                songNameField.setText(""); // Alanları temizle
                songPathField.setText("");
            } else {
                JOptionPane.showMessageDialog(null, "Lütfen şarkı adı ve dosya yolunu
doldurun.");
            }
        }
    });

    // Oynat butonu
    playButton.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            String selectedSong = playlist.getSelectedValue();
            if (selectedSong != null) {
                currentSongLabel.setText("Çalan Şarkı: " + selectedSong);
                musicPlayer.mevcutSarkiyiOynat();
            } else {
                JOptionPane.showMessageDialog(null, "Lütfen bir şarkı seçin.");
            }
        }
    });
```

```
// Durdur butonu
    stopButton.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            currentSongLabel.setText("Çalan Şarkı: Yok");
            // Durdurma işlemleri burada yapılabilir
        }
    });

    // Sonraki butonu
    nextButton.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            musicPlayer.sonrakiSarki();
            updateCurrentSongLabel();
        }
    });

    // Önceki butonu
    previousButton.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            musicPlayer.oncekiSarki();
            updateCurrentSongLabel();
        }
    });

    setVisible(true);
```

```
    }

    // Mevcut şarkı etiketini güncelleyen metot
    private void updateCurrentSongLabel() {
        SongNode currentSong = musicPlayer.getMevcutSarki();
        if (currentSong != null) {
            currentSongLabel.setText("Çalan Şarkı: " + currentSong.sarkiAdi);
        } else {
            currentSongLabel.setText("Çalan Şarkı: Yok");
        }
    }

    public static void main(String[] args) {
        new MusicPlayerGUI();
    }
}
```

Kodumuzun çıktısı:





Sueda Onur 230260145