

Code：

import javax.swing.\*;  
import javax.swing.tree.DefaultMutableTreeNode;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.io.IOException;  
import java.nio.file.Files;  
import java.nio.file.Path;  
import java.util.Objects;  
  
*/\*\*  
 \* 组合模式实现的文件浏览器  
 \* 支持浏览计算机目录结构，以树状图形式展示  
 \*/*public class Main {  
 // 抽象组件接口  
 interface FileSystemComponent {  
 String getName();  
 long getSize();  
 default String getInfo() {  
 return getName() + " (" + FileBrowser.*formatSize*(getSize()) + ")";  
 }  
 }  
  
 // 文件类（叶子节点）  
 static class File implements FileSystemComponent {  
 private final Path path;  
  
 public File(Path path) {  
 this.path = path;  
 }  
  
 @Override  
 public String getName() {  
 return path.getFileName().toString();  
 }  
  
 @Override  
 public long getSize() {  
 try {  
 return Files.*size*(path);  
 } catch (IOException e) {  
 System.*err*.println("无法获取文件大小: " + path);  
 return 0;  
 }  
 }  
 }  
  
 // 目录类（容器节点）  
 static class Directory implements FileSystemComponent {  
 private final Path path;  
 private FileSystemComponent[] children;  
  
 public Directory(Path path) {  
 this.path = path;  
 loadChildren();  
 }  
  
 @Override  
 public String getName() {  
 return path.getFileName().toString();  
 }  
  
 @Override  
 public long getSize() {  
 long totalSize = 0;  
 for (FileSystemComponent child : getChildren()) {  
 totalSize += child.getSize();  
 }  
 return totalSize;  
 }  
  
 public FileSystemComponent[] getChildren() {  
 if (children == null) {  
 loadChildren();  
 }  
 return children;  
 }  
  
 private void loadChildren() {  
 try {  
 children = Files.*list*(path)  
 .map(p -> Files.*isDirectory*(p) ? new Directory(p) : new File(p))  
 .toArray(FileSystemComponent[]::new);  
 } catch (IOException | SecurityException e) {  
 System.*err*.println("无法读取目录: " + path + " - " + e.getMessage());  
 children = new FileSystemComponent[0];  
 }  
 }  
 }  
  
 // 文件浏览器主窗口  
 static class FileBrowser extends JFrame {  
 private final JTextField pathField;  
 private final JTree fileTree;  
 private final DefaultMutableTreeNode rootNode;  
  
 public FileBrowser() {  
 setTitle("文件浏览器 - 组合模式示例");  
 setSize(800, 600);  
 setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 setLayout(new BorderLayout());  
  
 // 顶部面板：路径输入和加载按钮  
 JPanel topPanel = new JPanel();  
 pathField = new JTextField(System.*getProperty*("user.home"), 50);  
 JButton loadButton = new JButton("加载目录");  
 loadButton.addActionListener(this::loadDirectory);  
  
 topPanel.add(new JLabel("目录路径:"));  
 topPanel.add(pathField);  
 topPanel.add(loadButton);  
 add(topPanel, BorderLayout.*NORTH*);  
  
 // 中央面板：文件树  
 rootNode = new DefaultMutableTreeNode("根目录");  
 fileTree = new JTree(rootNode);  
 JScrollPane treeScrollPane = new JScrollPane(fileTree);  
 add(treeScrollPane, BorderLayout.*CENTER*);  
  
 // 状态栏  
 JLabel statusLabel = new JLabel("就绪");  
 add(statusLabel, BorderLayout.*SOUTH*);  
  
 // 初始加载  
 loadDirectory(null);  
 }  
  
 private void loadDirectory(ActionEvent e) {  
 String path = pathField.getText();  
 try {  
 Path dirPath = Path.*of*(path);  
 if (!Files.*exists*(dirPath) || !Files.*isDirectory*(dirPath)) {  
 JOptionPane.*showMessageDialog*(this, "路径不存在或不是目录", "错误", JOptionPane.*ERROR\_MESSAGE*);  
 return;  
 }  
  
 rootNode.removeAllChildren();  
 Directory rootDir = new Directory(dirPath);  
 rootNode.setUserObject(rootDir.getInfo());  
 buildTree(rootNode, rootDir);  
  
 fileTree.updateUI();  
 // 展开前两级节点  
 for (int i = 0; i < fileTree.getRowCount(); i++) {  
 fileTree.expandRow(i);  
 }  
 } catch (Exception ex) {  
 JOptionPane.*showMessageDialog*(this, "加载目录失败: " + ex.getMessage(), "错误", JOptionPane.*ERROR\_MESSAGE*);  
 }  
 }  
  
 private void buildTree(DefaultMutableTreeNode parentNode, Directory directory) {  
 for (FileSystemComponent child : directory.getChildren()) {  
 DefaultMutableTreeNode childNode = new DefaultMutableTreeNode(child.getInfo());  
 parentNode.add(childNode);  
  
 if (child instanceof Directory) {  
 buildTree(childNode, (Directory) child);  
 }  
 }  
 }  
  
 // 工具方法：格式化文件大小显示  
 static String formatSize(long size) {  
 if (size < 1024) return size + " B";  
 int exp = (int) (Math.*log*(size) / Math.*log*(1024));  
 String pre = "KMGTPE".charAt(exp - 1) + "i";  
 return String.*format*("%.1f %sB", size / Math.*pow*(1024, exp), pre);  
 }  
 }  
  
 public static void main(String[] args) {  
 SwingUtilities.*invokeLater*(() -> {  
 try {  
 // 设置系统外观  
 UIManager.*setLookAndFeel*(UIManager.*getSystemLookAndFeelClassName*());  
 } catch (Exception e) {  
 // 忽略外观设置失败  
 }  
 new FileBrowser().setVisible(true);  
 });  
 }  
}