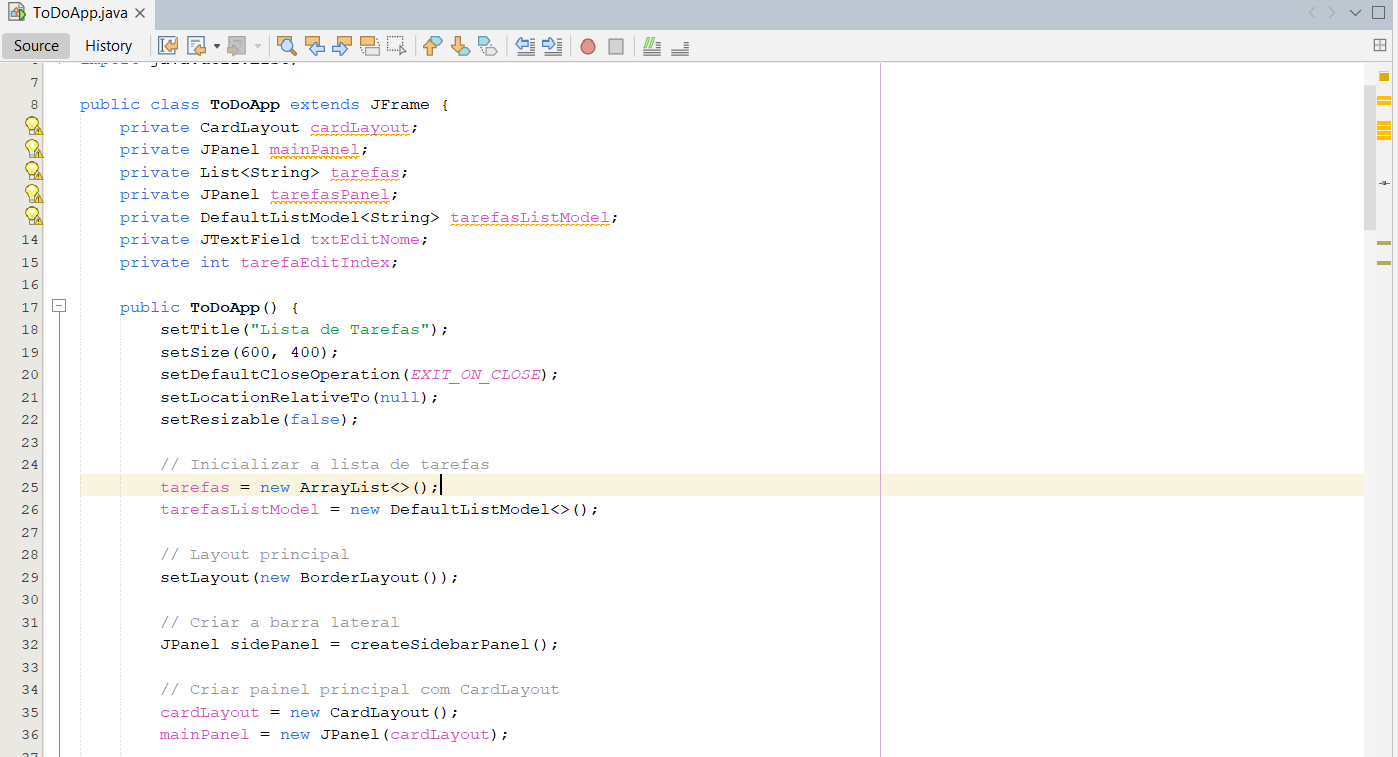
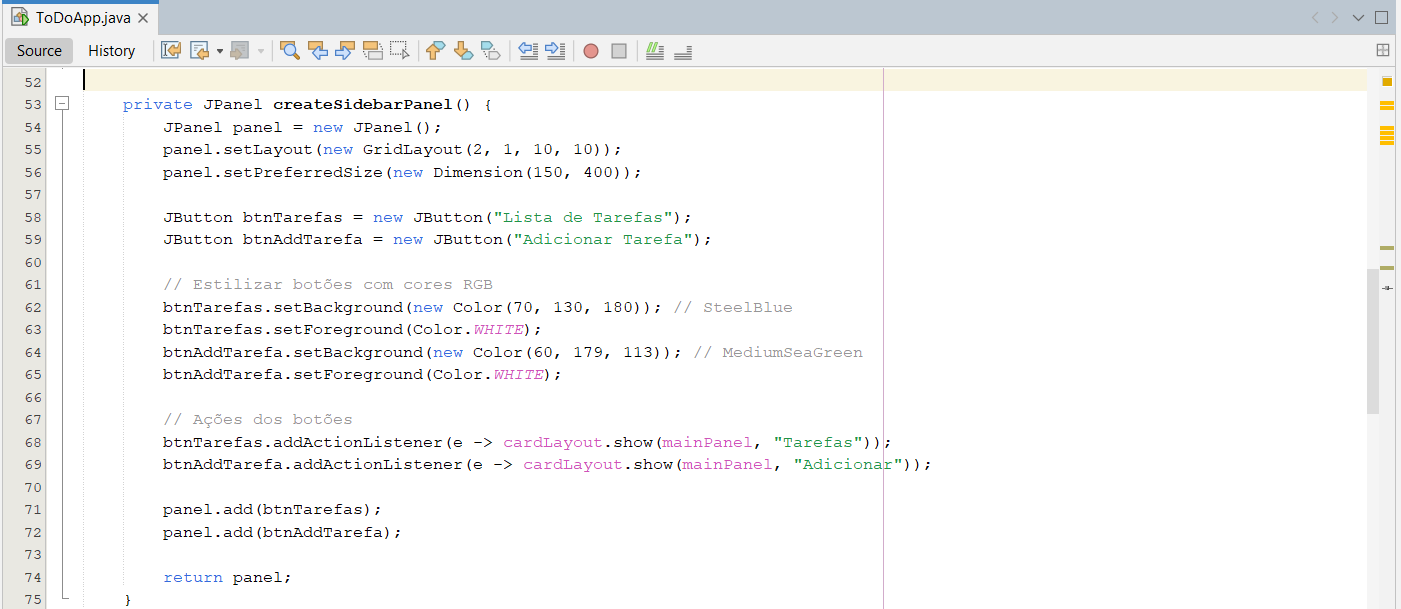
O código representa uma aplicação de gerenciamento de tarefas usando Java e Swing. Aqui estão os trechos mais relevantes e sua explicação:

### 1. **Estrutura Geral (ToDoApp como JFrame)**



* ToDoApp estende JFrame, tornando-se a janela principal da aplicação.
* Utiliza CardLayout para alternar entre diferentes telas (lista de tarefas, adicionar tarefa e editar tarefa).
* Gerencia uma lista de tarefas com ArrayList<String> e um modelo DefaultListModel<String> para exibição na interface gráfica.

### 2. **Barra Lateral (Menu de Navegação)**



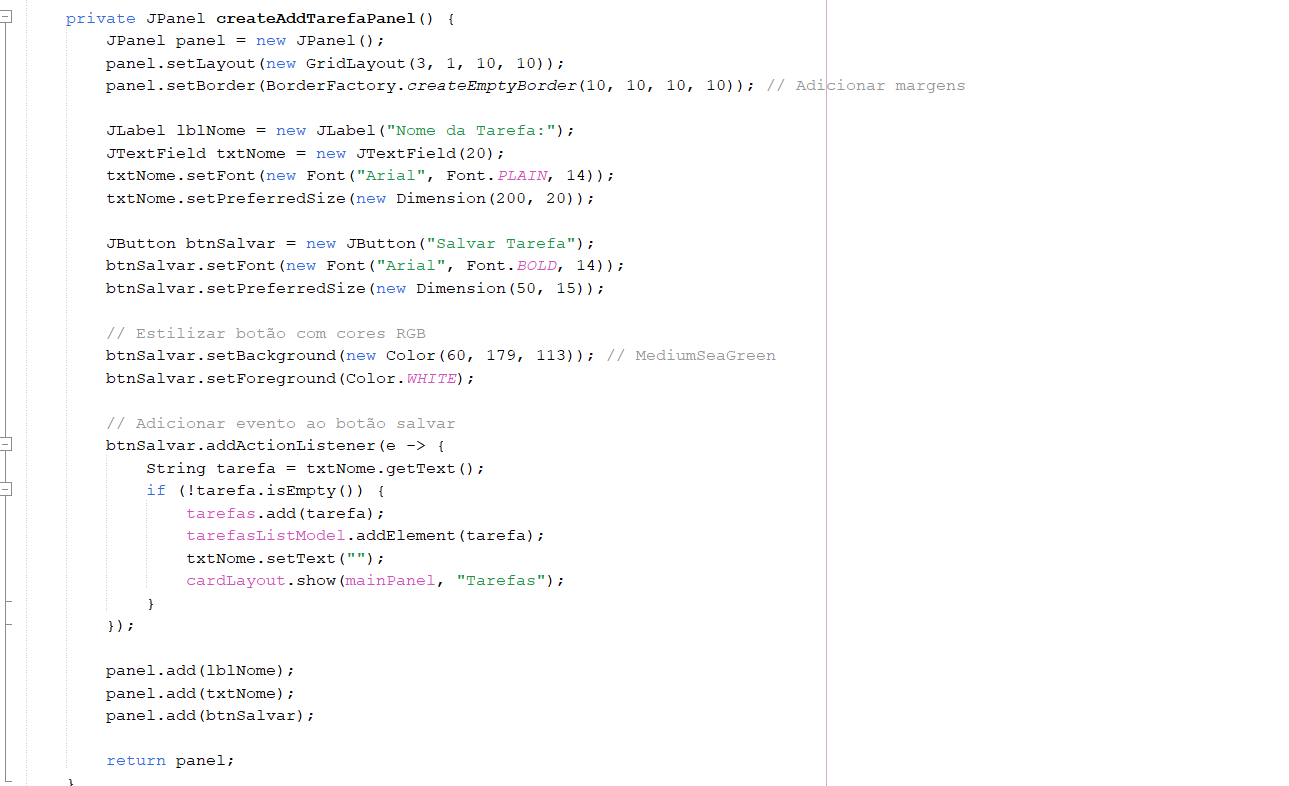
* createSidebarPanel(): Cria um painel com dois botões:
  + **"Lista de Tarefas"**: Mostra a tela principal com a lista de tarefas.
  + **"Adicionar Tarefa"**: Alterna para a tela de adição de tarefas.
* Usa GridLayout(2,1) para organizar os botões verticalmente.

### 3. **Tela Principal - Lista de Tarefas**



* createTarefasPanel(): Cria a interface que exibe a lista de tarefas usando JList<String>.
* Dois botões importantes:
  + **"Editar Tarefa"**: Permite modificar o nome da tarefa selecionada.
  + **"Apagar Tarefa"**: Remove a tarefa selecionada.

### 4. **Adicionar uma Nova Tarefa**



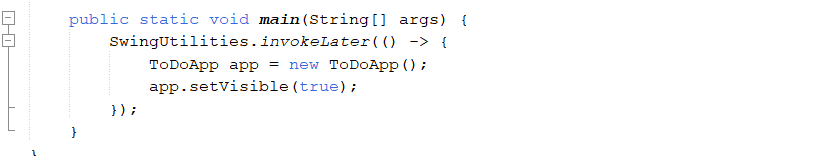
* createAddTarefaPanel(): Contém um campo de texto (JTextField) e um botão "Salvar Tarefa".
* Quando o botão é pressionado:
  + O texto digitado é adicionado à lista de tarefas (tarefas.add(tarefa)).
  + A lista visível é atualizada (tarefasListModel.addElement(tarefa)).
  + Retorna para a tela principal.

### 5. **Editar uma Tarefa**



* createEditTarefaPanel(): Permite modificar o nome de uma tarefa existente.
* Quando uma tarefa é selecionada para edição:
  + O índice é armazenado (tarefaEditIndex).
  + O campo de edição (txtEditNome) recebe o valor atual da tarefa.
* Após a edição:
  + A tarefa na lista é atualizada (tarefas.set(tarefaEditIndex, tarefa)).
  + A interface gráfica é sincronizada (tarefasListModel.set(tarefaEditIndex, tarefa)).
  + Retorna para a tela de lista.

### 6. **Método main (Inicialização)**



* SwingUtilities.invokeLater(() -> { new ToDoApp().setVisible(true); });
* Garante que a interface seja criada na thread do Swing para evitar problemas de concorrência.

### **Resumo**

* A aplicação usa CardLayout para alternar entre telas.
* JList e DefaultListModel são usados para exibir tarefas.
* A adição, edição e remoção de tarefas são manipuladas de forma interativa.
* A interface gráfica é estilizada com cores e layouts para melhor organização.