Projeto Cook Tok

Alunos: Ester Toja e Igor Bandasz

Objetivo:

O trabalho tem como objetivo atingir todos os requisitos solicitados para a finalização da disciplina de Desenvolvimento de Aplicações Orientado a Componentes. O projeto Cook Tok é um Interface com objetivo de cronometrar o passo a passo de uma receita, como o próprio nome da aplicação Cook significa "cozinhar" e Tok em relação ao "Tik Tok" do relógio.

Tecnologias utilizadas:

O projeto foi desenvolvido utilizando a IDE NetBeans utilizando a linguagem Java. Para armazenamento de dados foi utilizado o Banco de dados MySQL. Utilizamos o GitHub para compartilhamento do código, e para organização do desenvolvimento utilizamos o método Kanban com o framework Trello.

Detalhamento do Código:

public class ConexaoFactory

Método:

public static Connection getConexao():

Tem como objetivo fazer uma conexão com o banco de dados.

public static void close(Connection connection):

Tem como objetivo fechar a conexão com o banco de dados.

public static void close(Connection conn, Statement stat):

Tem como objetivo fechar uma conexão e um statiment com o banco de dados.

public class Aplicação

Tem como objetivo instanciar a Tela de Início dando assim início a aplicação.

Método:

public static void main(String[] args):

Tem como objetivo chamar uma tela de início.

public class Cronômetro

Os atributos foram declarados como privados: int segundos, int segundosFixo, AtualizaLabel myThread e Thread thread.

Método:

public Cronometro():

O Método construtor tem o objetivo de inicializa todos os componentes da interface gráfica.

private void initComponents()

Tem o objetivo de modificar o texto do jLabel e adicionar os Listenend aos buttons que controlam o cronômetro.

private void segundos()

Tem o objetivo de decrementar o tempo.

private String tempoFormatado()

Tem o objetivo de realizar o cálculo para deixar o tempo no formato do cronômetro.

private class AtualizaLabel implements Runnable

Os atributos foram declarados como privados: boolean begin, boolean stop e boolean paused.
E foram declarados os métodos getters e setters.

Método:

public AtualizaLabel()

O Método construtor tem o objetivo de controlar o comportamento do cronômetro.

public void run()

Tem como objetivo controlar as Threads.

private class StartListener implements ActionListener

Método:

public void actionPerformed(ActionEvent e)

Tem como objetivo instanciar a Thread assim dando início ao cronômetro, alterando os valores dos atributos assim iniciando a Thread.

private class PauseListener implements ActionListener

Método:

public void actionPerformed(ActionEvent e)

Tem como objetivo alterar o valor do atributo, assim pausando a Thread.

private class StopListener implements ActionListener

Método:

public void actionPerformed(ActionEvent e)

Tem como objetivo alterar o valor do atributo, assim parando a Thread.

public class Ingrediente

Os atributos foram declarados como privados: String nome_Ingred e private int cod_Ingred.
E foram declarados os métodos getters e setters.

Método:

public Ingrediente( )

public Ingrediente(String nome_Ingred)

public Ingrediente(int cod, String nome_Ingred)

O método construtor foi sobrescrito.

public class Instrucao

Os atributos foram declarados como privados: int cod_Instru, String nome_Instru, int tempo_Instru e int cod_Receita. E foram declarados os métodos getters e setters.

Método:

public Instrucao()

public Instrucao(String nome_Instru, int tempo_Instru, int cod_Receita)

public Instrucao(int cod_Instru, String nome_Instru, int tempo_Instru, int cod_Receita)

O método construtor foi sobrescrito.

public class Receita

Os atributos foram declarados como privados: int cod_Receita, String nome_Receita e int tempo_Preparo. E foram declarados os métodos getters e setters.

Método:

public Receita()

public Receita(String nome_Receita, int tempo_Preparo)

public Receita(int cod_Receita, String nome_Receita, int tempo_Preparo)

O método construtor foi sobrescrito.

public class Rel_ReceitaIngrediente

Os atributos foram declarados como privados: int cod_Relacao, int cod_Receita, int cod_Ingred, int quantidade e String unidade . E foram declarados os métodos getters e setters.

Método:

public Rel_ReceitaIngrediente( )

public Rel_ReceitaIngrediente(int cod_Receita, int cod_Ingred, int quantidade, String unidade)

public Rel_ReceitaIngrediente(int cod_Relacao, int cod_Receita, int cod_Ingred, int quantidade, String unidade)

O método construtor foi sobrescrito.

public class TbIngrediente

Método:

public static void save(Ingrediente ingre)

Tem como objetivo salvar os novos ingredientes no banco de dados.

public class TbInstrucao

Método:

public void save(Instrucao instru)

Tem como objetivo salvar as instruções que foram preenchidos na jTable de instruções para o banco de dados.

public class TbReceita

Método:

public static void save(Receita receita)

Tem como objetivo salvar uma receita no banco de dados.

public static int ultimo()

Tem como objetivo trazer o código da última receita criada.

public class TbRelIngreRec

Método:

public static void save(Rel_ReceitaIngrediente relacao)

Tem como objetivo salvar os códigos dos ingredientes, código da quantidade e código da unidade relaciona-se com as receitas de acordo com o preenchidos na jTable de ingredientes para o banco de dados.

public class Tela_Inicio.java

Tem como objetivo ser a tela inicial da aplicação, o usuário pode escolher entre adicionar uma nova receita e executar uma receita.

A classe tem os atributos privados: int[] vetorCodigo, Tela_Exec telaExec, int pagina e ArrayList<Receita> lista.

Métodos:

private void btPesquisarActionPerformed(java.awt.event.ActionEvent evt):

Tem como objetivo realizar a pesquisa de acordo com a ordem e o filtro escolhido.

private void btAdicionarReceitaActionPerformed(java.awt.event.ActionEvent evt):

Tem como objetivo chamar a jFrame (TelaAdicionar_Receita.java).

private void formWindowActivated(java.awt.event.WindowEvent evt):

Tem como objetivo chamar o método que busca informações do banco de dados.

private void btAvancarActionPerformed(java.awt.event.ActionEvent evt):

Tem como objetivo avançar as opções de receitas para executar.

private void lbReceita0MouseClicked(java.awt.event.MouseEvent evt):

private void lbReceita1MouseClicked(java.awt.event.MouseEvent evt):

private void lbReceita2MouseClicked(java.awt.event.MouseEvent evt):

private void lbReceita3MouseClicked(java.awt.event.MouseEvent evt):
private void lbReceita4MouseClicked(java.awt.event.MouseEvent evt):
private void lbReceita5MouseClicked(java.awt.event.MouseEvent evt):
private void lbReceita6MouseClicked(java.awt.event.MouseEvent evt):
private void lbReceita7MouseClicked(java.awt.event.MouseEvent evt):
private void lbReceita8MouseClicked(java.awt.event.MouseEvent evt):
private void lbReceita9MouseClicked(java.awt.event.MouseEvent evt):
Tem como objetivo ser as opções de receitas, ao serem selecionadas um dos componentes deve chamar a tela de Tela_Execu.java e executar a receita referente.
private void btVoltarActionPerformed(java.awt.event.ActionEvent evt):
Tem como objetivo retroceder as opções de receitas para executar.
private void cbFiltroItemStateChanged(java.awt.event.ItemEvent evt):
Tem como objetivo limpar o campo textField de Pesquisar.
public void carregaReceitas(String sql):
Tem como objetivo carregar todas as receitas do banco de dados.
public void exibeLista():
Tem como objetivo listar todas as receitas para que seja exibido lista de receitas.
public void formataTempo(int t):
Tem como objetivo que o tempo seja exibido em horas, minutos e segundos e que seja adicionado ao lado do nome das receitas.
public void limpaVetor():
Tem como objetivo realizar a limpeza dos vetor mudando todos as posições para números negativos.
public static void main(String args[]):
Tem como objetivo instanciar uma Tela de Inicio.

public class TelaAdicionar_Receita.java
Tem como objetivo permitir que o usuário possa adicionar novas receitas a aplicação, por meio de preenchimento de componentes textfield e combo box que visualmente ficam apresentadas em jtables.

Métodos:
public TelaAdicionar_Receita()
Tem como objetivo inicializar todos os componentes da interface gráfica.
private void Button_SalvarActionPerformed(java.awt.event.ActionEvent evt) :
Tem como objetivo salvar todas informações referentes a receitas (ingredientes e instruções), preenchidas nas jtables para o banco de dados.
private void btAdicionarIngredienteActionPerformed(java.awt.event.ActionEvent evt):
Tem como objetivo salvar todas as informações referentes ao ingrediente preenchido nos Combobox e nos textField para a jtable de ingredientes.
private void formWindowOpened(java.awt.event.WindowEvent evt):
Tem como objetivo chamar o método que busca os ingredientes do banco de dados.
private void txtCombo_Box_ingredientesFocusLost(java.awt.event.FocusEvent evt):
Tem como objetivo selecionar o código do ingrediente para posteriormente inserir no banco de dados.
private void btExcluirIngredienteActionPerformed(java.awt.event.ActionEvent evt)
Tem como objetivo excluir uma linha selecionada na jtables de ingrediente.
private void btAdicionarInstrucaoActionPerformed(java.awt.event.ActionEvent evt):*
Tem como objetivo salvar todas as informações referente as instruções preenchidas nos textFields para as jtables de instruções.
private void btExcluirInstrucaoActionPerformed(java.awt.event.ActionEvent evt) :
Tem como objetivo excluir uma linha selecionada na jtables de instruções.

private void OKActionPerformed(java.awt.event.ActionEvent evt):
Tem como objetivo fechar a janela mudando a visibilidade para “false” do jDialog chamado de Poupop.
private void txtCombo_Box_ingredientesItemStateChanged(java.awt.event.ItemEvent evt):
Tem como objetivo chamar o JDialog de Adicionar Ingrediente caso seja selecionado “Adicionar novo” no combobox.
private void cancelarPoupopActionPerformed(java.awt.event.ActionEvent evt):
Tem como objetivo fechar o JDialog de Adicionar Ingrediente.
private void salvarPoupopActionPerformed(java.awt.event.ActionEvent evt):
Tem como objetivo salvar o novo ingrediente.
private void btSairActionPerformed(java.awt.event.ActionEvent evt):
Tem como objetivo fechar a jFrame.
public void carregaIngredientes() ;
Tem como objetivo buscar as informações do banco de dados referente a ingredientes para os combo box.
public static void main(String args[]);
Tem como objetivo instanciar a TelaAdicionar_Receita.java.

Tela_Exec.java =
Tem como objetivo permitir a visualização do usuário de todos os ingredientes e instruções para a execução da receita selecionada. A tela é composta por duas jtables que mostram todos os ingredientes e instruções necessárias para a execução da receita. Também possui um cronômetro e a instrução do passo que está sendo cronometrado.

Métodos:
private void btIniciarActionPerformed(java.awt.event.ActionEvent evt) :
Tem como objetivo iniciar o cronômetro do passo das receitas.
private void formWindowActivated(java.awt.event.WindowEvent evt):
Tem como objetivo carregar todos os itens necessários de ingredientes e instruções para a execução da receita.
private void btProximoActionPerformed(java.awt.event.ActionEvent evt):
Tem como objetivo permitir que o usuário possa passar para próximo passo.
private void btSairActionPerformed(java.awt.event.ActionEvent evt):
Tem como objetivo permitir que o usuário possa fechar o jFrame.
private void btPausarActionPerformed(java.awt.event.ActionEvent evt):
Tem como objetivo permitir que o usuário possa pausar o tempo do cronômetro.
private void btPararActionPerformed(java.awt.event.ActionEvent evt):
Tem como objetivo permitir que o usuário possa parar o tempo do cronômetro.

OBS: A diferença entre pausar e parar, é porque o parar permite que o tempo seja inicializado novamente. Enquanto o pausar, continua a cronometrar o tempo de onde parou.

public void proximaInstrucao():
Tem como objetivo passar para o próximo passo(instrução).
public void formatarTempo():
Tem como objetivo de fazer a formatação do tempo para o cronômetro.
public static void controleBotoes(int acao):
Tem como objetivo controlar todas as ações dos botões do cronômetro (iniciar, pausar, próximo e parar), tornando disponível ou indisponível os buttons conforme a ação.
public static void main(String args[]):
Tem como objetivo instalar uma tela de Execução.