Software para Persistência de Dados - 2025.03

Prof. Marcelo Akira

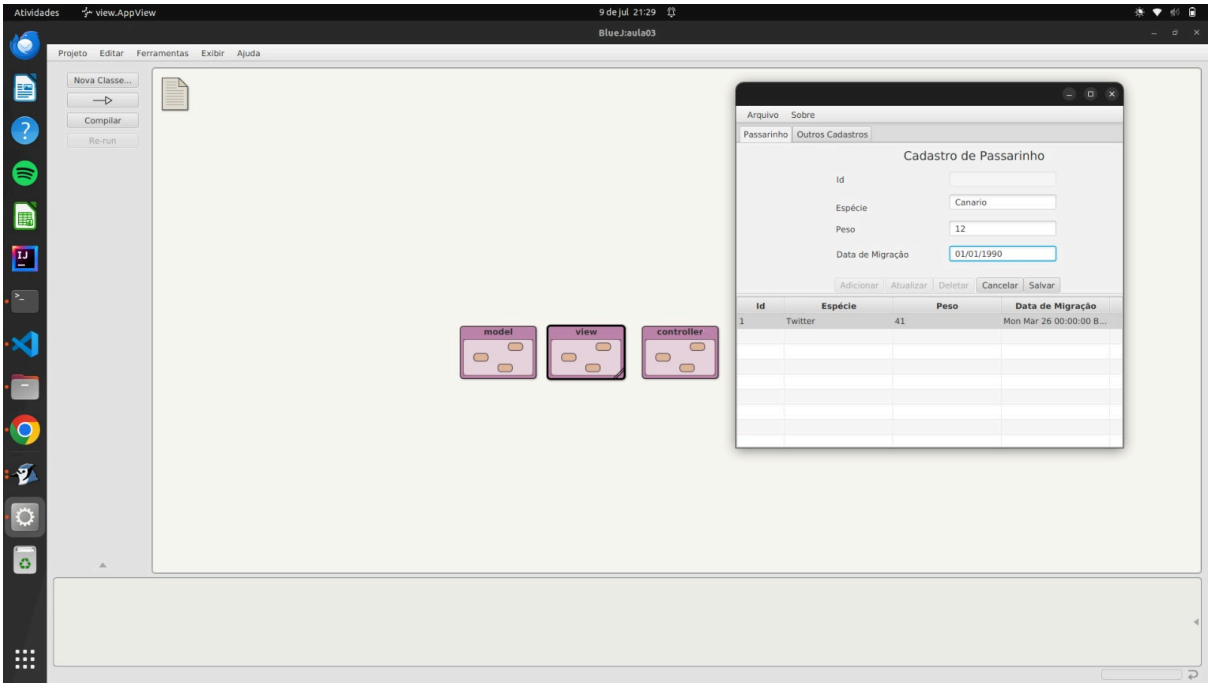
Atividade 0709 - Criar uma interface completa para uma entidade

Discentes:
Felipe Moreira Silva - 202201689
João Gabriel Cavalcante França - 202201695

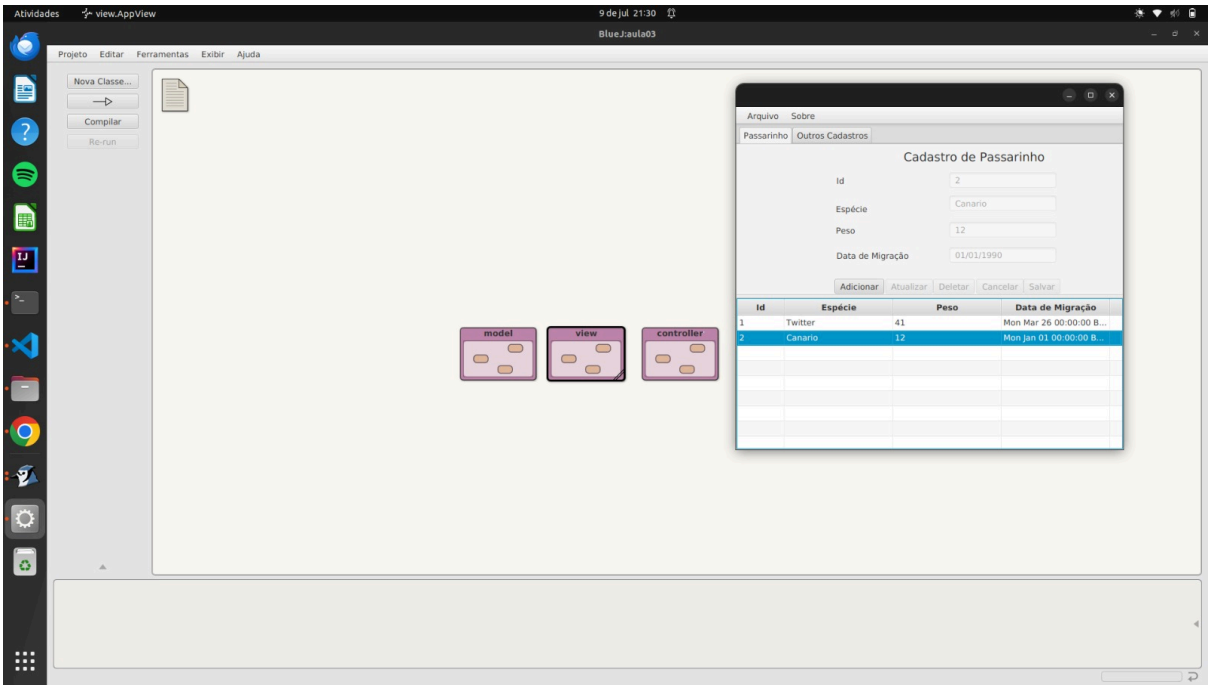
1. Método CREATE

O fluxo do Create no CRUD inicia quando o usuário clica no botão "Adicionar", que executa onAdicionarButtonAction() para habilitar os campos de entrada, desabilitar outros botões e limpar o formulário. Após o usuário preencher os dados (espécie, peso, data de migração) e clicar em "Salvar", o método onSalvarButtonAction() detecta que é uma operação CREATE (pois não há item selecionado na tabela), cria um objeto model.Passarinho convertendo os dados dos campos de texto para os tipos apropriados (String, BigDecimal, Date), chama passarinhoRepo.create() que utiliza ORMLite para gerar automaticamente o SQL INSERT e persistir no banco SQLite retornando o objeto com ID gerado, converte esse objeto model para view.Passarinho (compatível com JavaFX Properties) através do método modelToView(), adiciona o novo item na TableView para exibição imediata, seleciona o item recém-criado e por fim reseta a interface desabilitando os campos e retornando os botões ao estado inicial.

Antes de adicionar

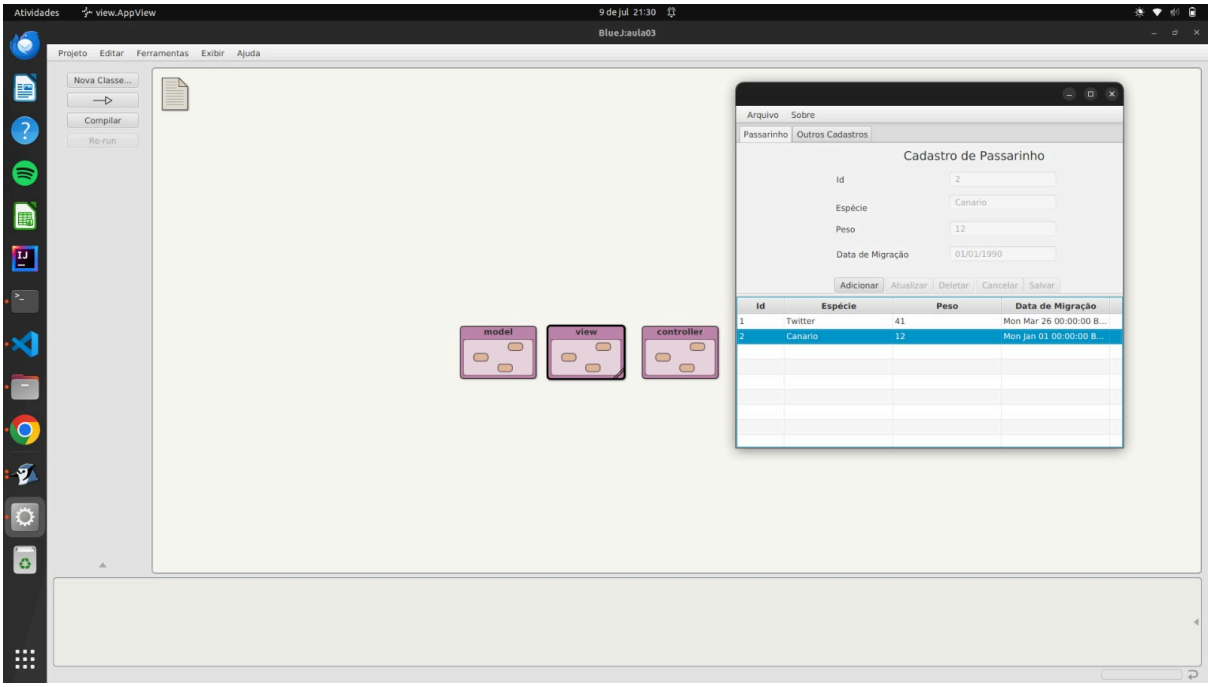


Depois de adicionar



2. Método RETRIEVAL

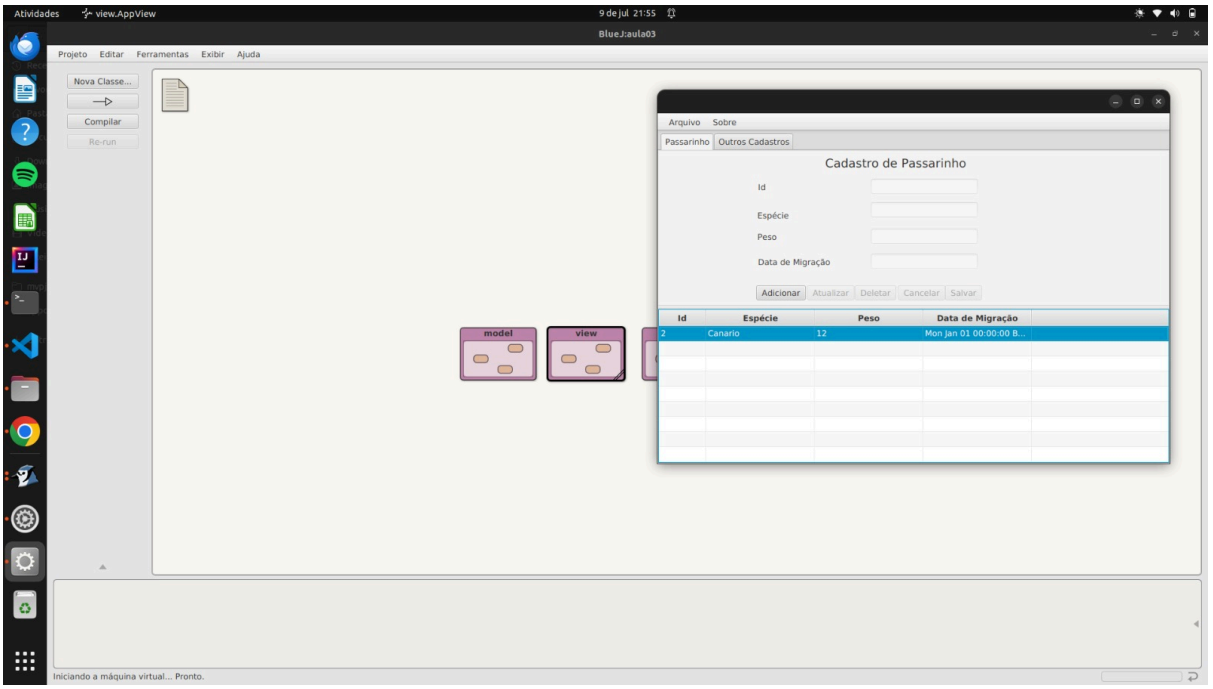
O Retrieval (Read) no CRUD acontece automaticamente durante a inicialização da aplicação através do método initialize() que é executado quando a interface JavaFX é carregada. Este método chama loadAllPassarinhos() que utiliza passarinhoRepo.getAll() para buscar todos os registros de passarinhos no banco de dados SQLite via ORMLite, converte cada objeto model.Passarinho retornado para view.Passarinho usando o método modelToView() (para compatibilidade com as propriedades JavaFX), cria uma ObservableList com todos os itens convertidos e define essa lista como fonte de dados da TableView através de tabela.setItems(). As colunas da tabela são configuradas com PropertyValueFactory para mapear automaticamente as propriedades dos objetos view.Passarinho (id, species, weight, migration) aos campos visuais correspondentes, garantindo que qualquer mudança na lista seja refletida instantaneamente na interface. Por isso, quando adicionamos um novo passarinho e ele é inserido diretamente na ObservableList da tabela via tabela.getItems().add(passarinhoView), a TableView automaticamente exibe o novo item junto com todos os outros já carregados, criando a impressão de que a lista completa está sempre sincronizada com o banco de dados.



3. Método Delete

O fluxo do Delete no CRUD inicia quando o usuário seleciona um item na TableView e clica no botão "Deletar", executando onDeletarButtonAction() que primeiro verifica se há um item selecionado através de tabela.getSelectionModel().getSelectedItem(). Caso haja seleção, o método cria um objeto model.Passarinho temporário copiando todos os dados do objeto view.Passarinho selecionado (incluindo o ID essencial para identificar o registro no banco), chama passarinhoRepo.delete() que utiliza ORMLite para gerar automaticamente o SQL DELETE baseado no ID do objeto e remove o registro do banco SQLite. Após a exclusão bem-sucedida no banco, o item é removido da interface visual através de tabela.getItems().remove(selectedPassarinho), que automaticamente atualiza a TableView devido à ObservableList, e por fim o método executa limparCampos() para limpar os campos de texto, desabilitarCampos(true) para desabilitar a edição e desabilitarBotoes() para retornar os botões ao estado inicial, garantindo que tanto a persistência quanto a interface estejam sincronizadas. Todo o processo é protegido por tratamento de exceções que exibe um Alert com mensagem de erro caso algo falhe durante a operação de exclusão.

Resultado da deleção da primeira instância:



4. Método Update

O Update no acontece em duas etapas: primeiro o usuário seleciona um item na TableView e clica no botão "Atualizar", que executa onAtualizarButtonAction() para habilitar os campos de edição (desabilitarCampos(false)) e ajustar o estado dos botões mantendo apenas "Cancelar" e "Salvar" habilitados, permitindo que o usuário modifique os dados nos campos de texto. Após as modificações, quando o usuário clica em "Salvar", o método onSalvarButtonAction() detecta que é uma operação UPDATE (pois tabela.getSelectionModel().getSelectedItem() retorna um objeto não-nulo), cria um novo objeto model.Passarinho copiando o ID do item selecionado e preenchendo com os novos dados dos campos de texto (convertendo adequadamente String para BigDecimal e Date), chama passarinhoRepo.update() que utiliza ORMLite para gerar automaticamente o SQL UPDATE baseado no ID e persiste as alterações no banco SQLite. Após a atualização no banco, o método atualiza o objeto view.Passarinho selecionado na interface com os novos valores através dos métodos set*(), executa tabela.refresh() para forçar a TableView a atualizar a exibição visual dos dados modificados e por fim reseta a interface desabilitando os campos e retornando os botões ao estado inicial, garantindo sincronização completa entre banco de dados e interface gráfica.

Atualizando a instância remanescente - alterando a espécie e o peso: