Meu passo a passo sobre JAVA

Passo a passo iniciado a partir do curso de Java na prática - #1 à 27 do professor NAZARÉ do site BLSoft no link [www.blsoft.com.br](http://www.blsoft.com.br)

Princípio básico para codificar com netBeans, sempre ir aceitando as dicas do netBeans, pois 80% das vezes ele vai vincular o código a algo, sempre digitar as primeiras letras e caso não apareça algum complemente, segurar CTRL e apertar espaço. Lembre-se disso do começo ao fim.

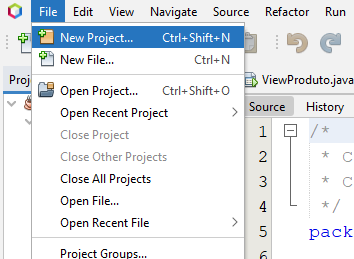
Aula 01 – Descreve sobre o projeto, inclusive mostra o programa e funcionalidades.

Aula 02 – Descreve sobre a ESPECIFICAÇÃO DE SOFTWARE.docx, que esta junto ao GitHub, fala ainda sobre a documentação tabelas e diagramas, e o material que ele vende, ajudem o homem pois as aulas são boas, já digo que não ganhei nada, nem conheço o professor, mas como a aula dele foi o diferencial para voltar a querer estudar Java, estou promovendo ele.

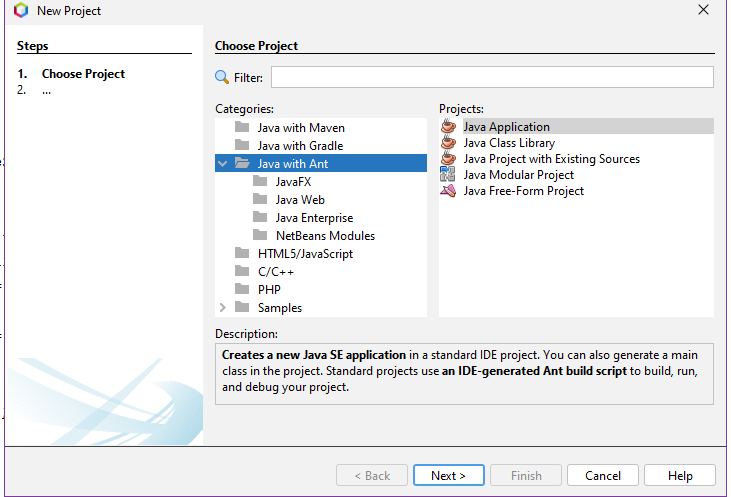
Aula 03 – Inicia falando sobre a criação da interface, os aplicativos usados: JDK, NetBeans e SqlLite Studio. “faz a mercham do código fonte que está vendendo”.

Aula 04 – “Criando a interface gráfica”, explica sobre padrão MVC.

Primeiro cria um novo projeto, clicando em, file/New Project:

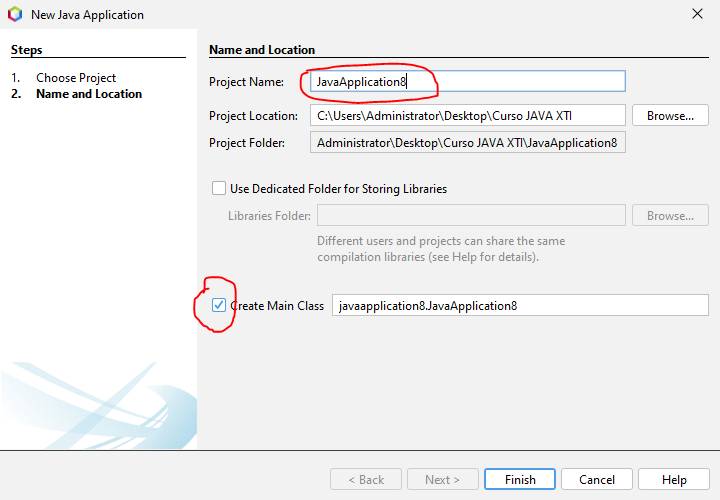


Vai abrir a janela abaixo, usar categoria Java With Ant e Projects: Java Application.

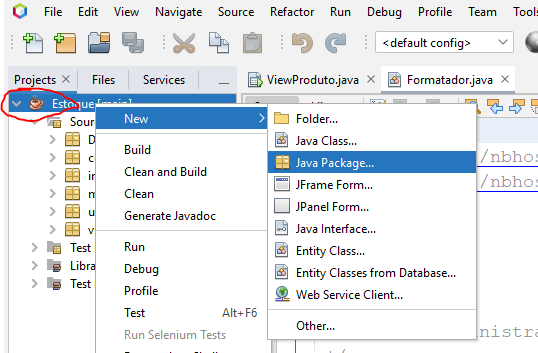


Clicar em Next, vai aparecer a tela abaixo, Sempre lembra de seguir os padrões modelo, entre letras maiúscula e minúscula, não precisa decorar, só seguir o modelo que já vai estar no padrão MVC, basta digitar no campo marcado que ele adiciona de forma automática nos outros.

Desmarcar a caixa Create Main Class, pois a classe principal vai ser criada na mão mesmo.



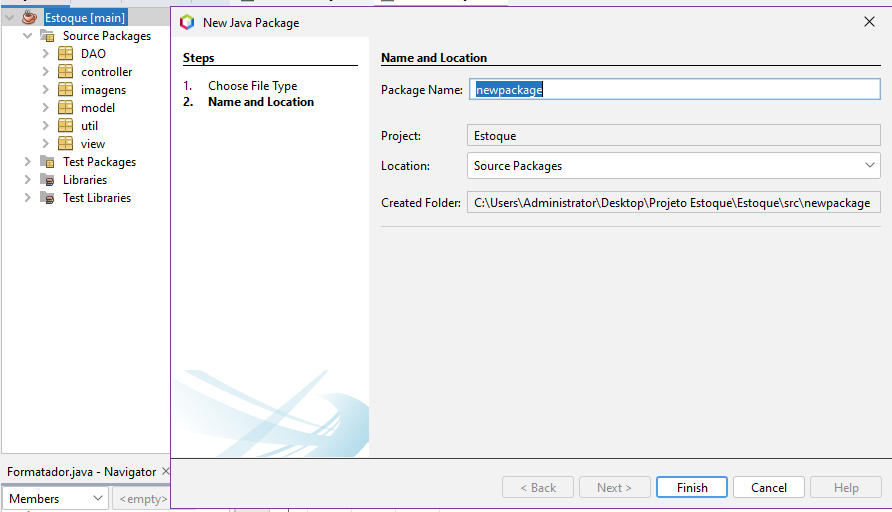
Clicar com o botão direito no nome do seu projeto, depois em New/Java Package.



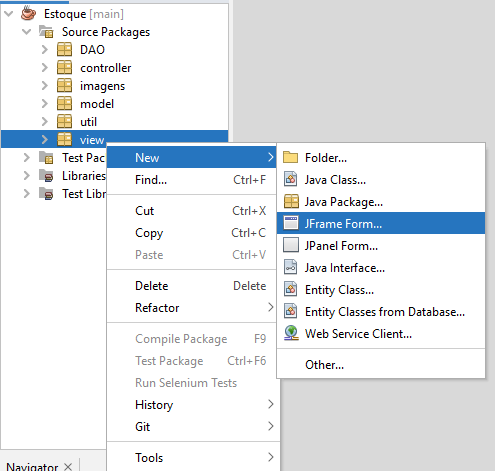
Criar os pacotes padrões que geralmente são usados em boa parte dos projetos, quais sejam:

* DAO 🡪 pacotes relacionado ao Banco de Dados
* controller 🡪Pacote dos controladores
* imagens 🡪Pacote de imagens/ícones
* model 🡪
* util 🡪Pacote usado para
* view 🡪 Pacote usado para criar as telas, ou seja a parte visual

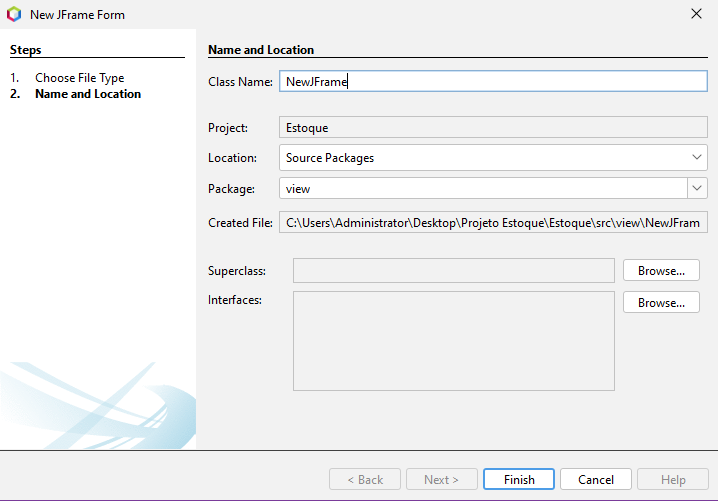
Com exceção do DAO que é a referencia ao banco de dados, escrever tudo em minúsculo igual no modelo.



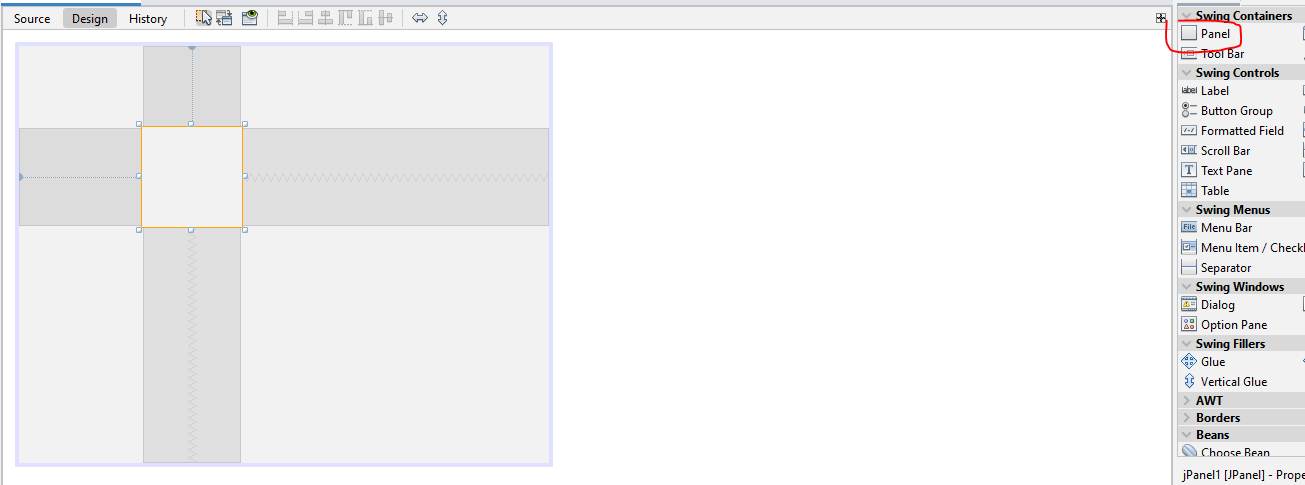
Criados os pacotes, é dado início a criação das view, Clica com o botão direito, New/JFrame Form..., muito cuidado para selecionar outro pois a função é diferente.



Pode-se observar que novamente o próprio Netbeans segue o modelo MVC maiúsculo e minúsculo onde deve ser, nomear a o que quiser, no projeto vão ser criadas várias telas, tais como ViewEstoque, ViewLogin, ViewProduto e assim vai.

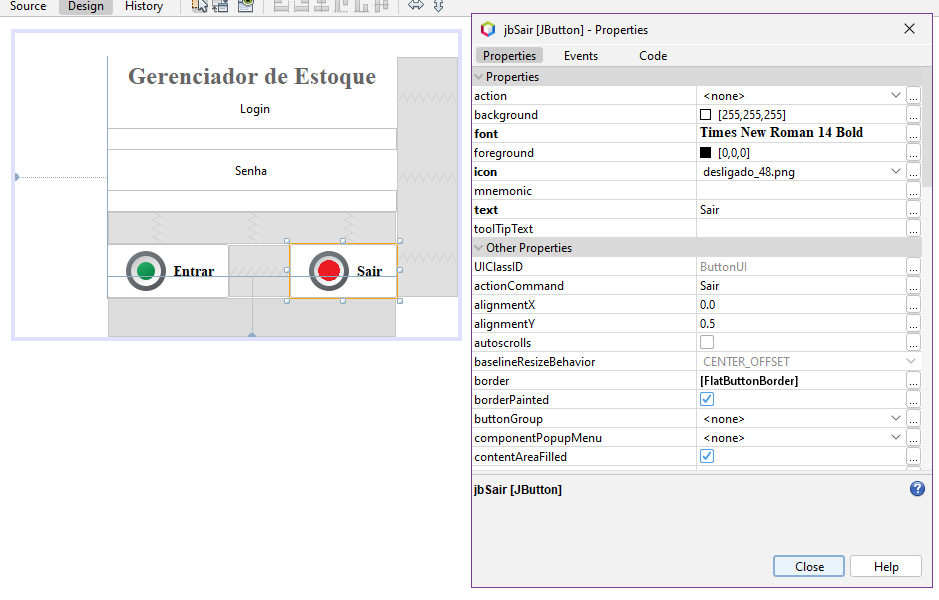


Após clicar em finish, vai aparecer a tela com o painel chamado de JFrame, que para facilitar o trabalho é bom adicionar um painel dentro dele e redimensionar.

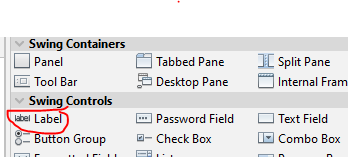


Adicionado, redimensionar arrastando para o tamanho do JFrame, pois assim ele ficara meio que responsível.

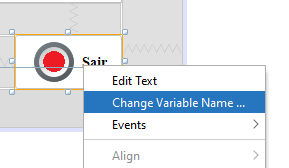
Arruma-lo à gosto, lembrando que ao selecionar algo, você pode modificar o nome selecionando o que for adicionar e clicar F2, que dará a opção de renomear, nas propriedades vc pode mudar, fonte e inserir ícones.



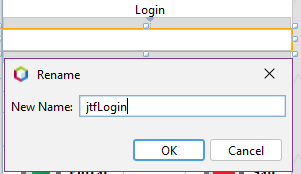
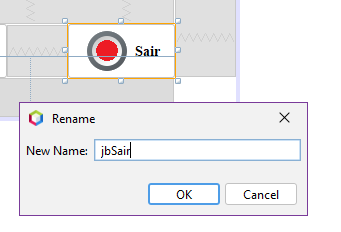
Adicionar os Label, que é onde vão ficar os textos que irão nomear as coisas como placas indicativas.



Uma boa prática é nomear todos os componentes, para facilitar na hora da programação, clicando com o botão direito e posterior Change variabele Name, à exemplo o campo abaixo.

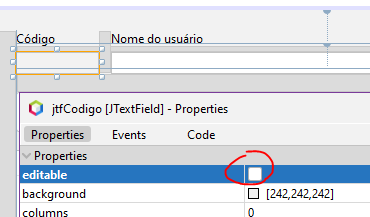


Foi renomeado para jbSair, onde jb significa JButton, jtfLogin onde jtf significa JTextField

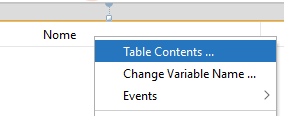


E fazer assim para todos os botões, campos de texto, campos de senha etc.

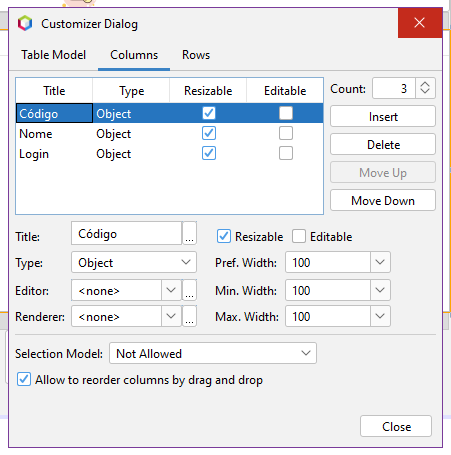
Ao fazer outras view, tais como o view usuário, no Código desmarcar a opção de edita-lo, pois o código vai ser inserido de forma automática pelo sistema.



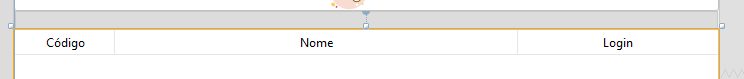
Já nas tabelas, clicar com o botão direito e depois Table Contents, para modifica-la



Você tem a opção de arrumar o tamanho das colunas e das linhas da forma que quiser, conforme imagem abaixo.



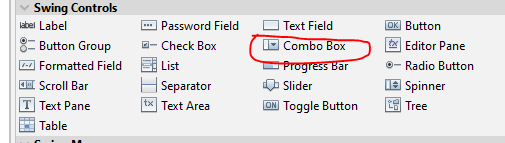
Deixando conforme a tela abaixo, sendo que basta colocar o tamanho para o campo que quiser e deixar default para o campo que o Netbeans usará o tamanho disponível.



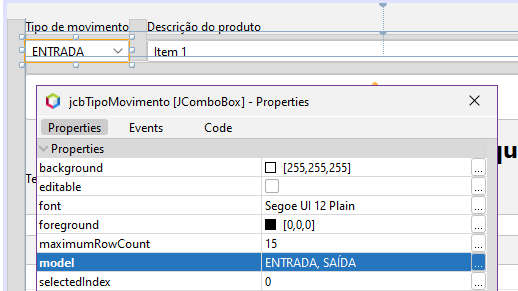
Neste caso foi deixado o código com 100 e o login com 200, o nome ficou como Default, consumindo o espaço restante de forma automática.

Aula 05 - Já na aula 5, foi iniciado a ViewEstoque, que nela foi adicionar um Combo Box, onde da opção de acionar itens pré-definidos “estáticos”.

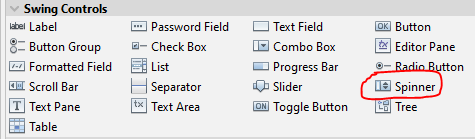
Adicionar o Combo Box igual na imagem abaixo.



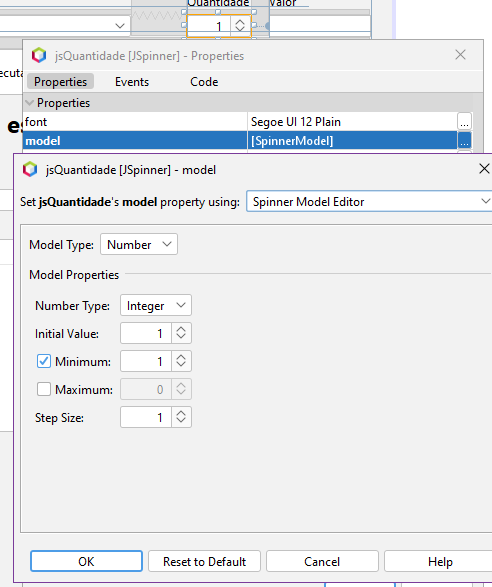
Após adiciona-lo modificar o model, como será somente duas opções colocar ENTRADA, SAÍDA ou seja, as duas opções separadas por uma vírgula.



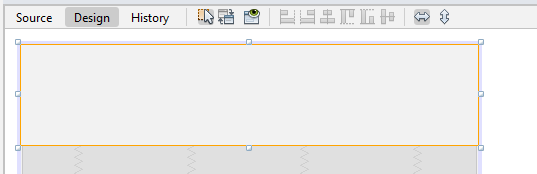
Já na opção de quantidade, o certo é colocar um Spinner.



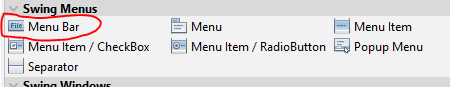
Nas propriedades dele definir o model, marcando Type como Number, e iniciar em 1 conforme imagem abaixo.



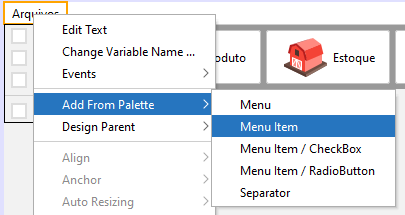
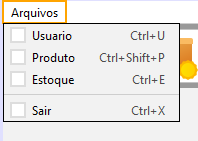
Aula 06 – Criando interface principal, “base para as demais”, criar a View conforme as demais, e dentro dele o painel “panel” porém somente em um trecho retangular na parte de cima, isso para poder delimitar o menu e botões a serem adicionados.



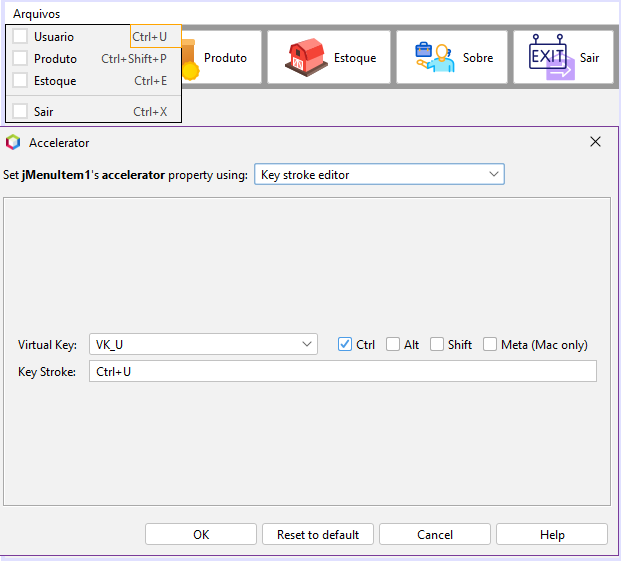
Adicionar o Menu Bar, que dará opção de selecionar as outras view.



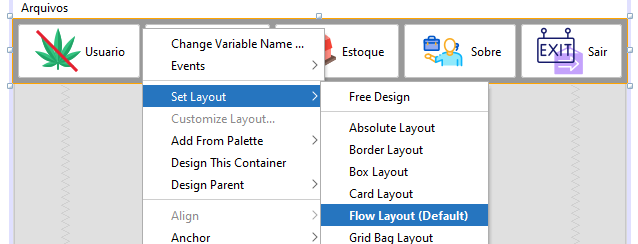
Após arrastar o menu para o Label, clicar com o botão direito, depois ir adicionando paletas à seu critério, clicando em Menu Item, que pode ser renomeado com o F2, também após resolver fazer uma paleta diferente pode-se adicionar um separador.

também tem como colocar atalho para cada botão bastando dar clique duplo, onde por exemplo foi definido que CTRL+U ira acionar o menu usuário.



Após criado o menu, só inserir botões e nomeá-los da maneira que quiser, após adicionado os botões basta definir um layout que ira redimensionar os botões de forma automática, basta clicar no Panel com o botão direito e depois em Set Layout, no caso foi usado o Flow Layout (Default), e posteriormente foram adicionados ícones.



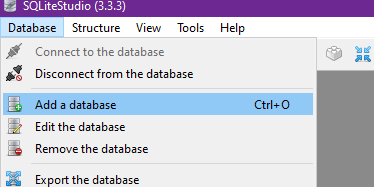
Aula 07 - Começa a falar sobre as classes de conexão com o banco de dados, ensina que não precisa decorar, basta saber usar o google.

Basta achar o código, salvar em formato Java e colar na classe, no caso foi colado na classe útil, ele até disponibilizou no link abaixo, onde deixei como comentário o código completo que deverá ser salvo necessariamente com o nome: ConexaoSQLite.java

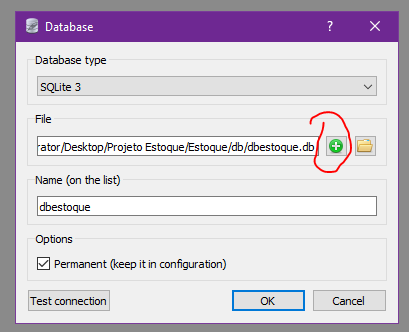
🡪 <https://drive.google.com/file/d/1BuUbqawtrnAvAeZr2XPaHB0rskY6GuPa/view>

Aula 08 - Inicia a aula falando sobre o Banco de Dados SQLite.

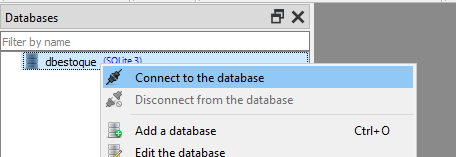
Abrir o SQLite , clicar em DataBase, depois Add a database.



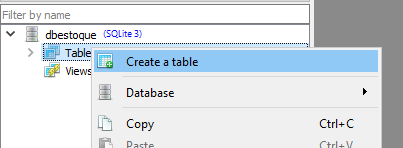
Antes de tudo ir na pasta onde salvou seu projeto e criar uma pasta chamada db, feito isso voltar para o SQLite, clicar na opção + e selecionar a pasta que criou, e adicionar o nome que quiser para o arquivo do banco de dados no caso foi incluíso o nome dbestoque, tudo minusculo.



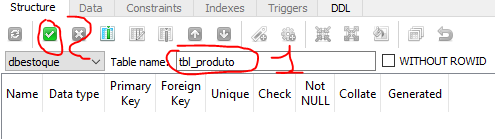
Feito isso, conectar no banco de dados, clicando com o botão direito no banco de dados criado e clicar em conectar.



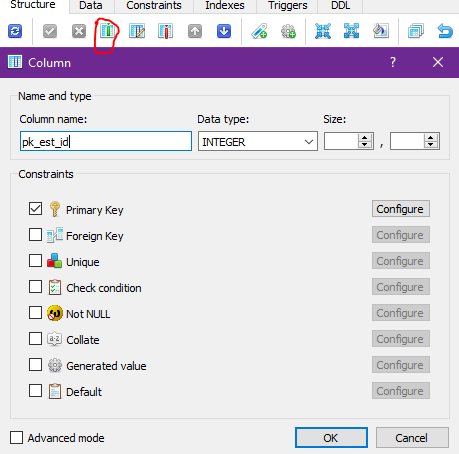
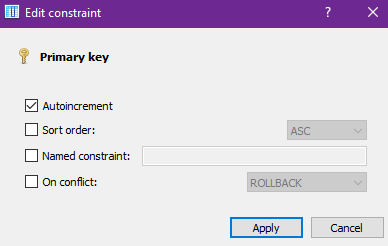
Feito isso, vai ser modela o banco de dados conforme necessidade, bastando clicar com o botão direito em Table depois Create a table.



Primeiro digitar o nome da tabela, seguindo o padrão tbl\_nome, depois clicar no botão verde para adicionar.

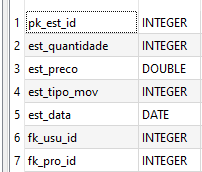


Criada a primeira tabela, vamos adicionar colunas nela, sendo sempre a primeira o nome padrão pk\_pro\_id, onde pk = primary key, pro = produto, id = id do produto, devido ser número o Data Type é INTEGER “inteiro” marcando os Constrainsts igual o modelo abaixo.

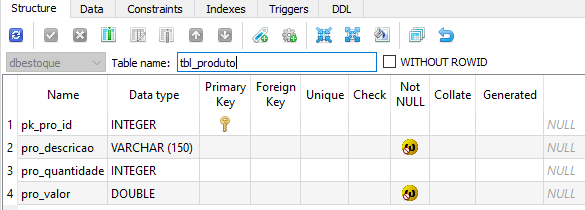
 

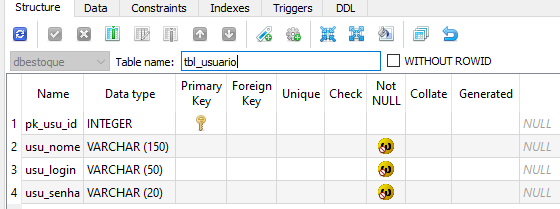
Ants de terminar, configurar a Primary Key, clicando em configure, e marcar AutoIncrement para incrementar sozinho a cada item adicionado.

Feito isso clicar no botão verde  para confirmar a inclusão, feito isso criar os demais itens da tabela, sendo todos not null.



Para o banco de dados ficar funcional, criar também as tabelas: tbl\_produto e tbl\_usuario com os campos abaixo.

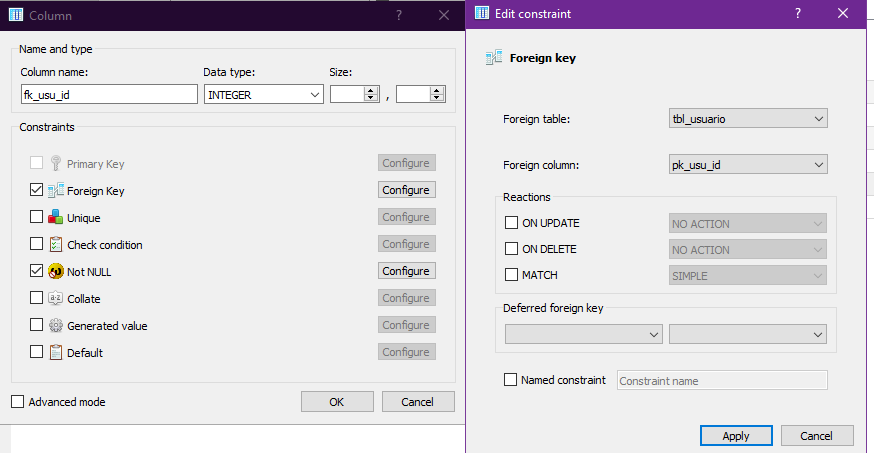


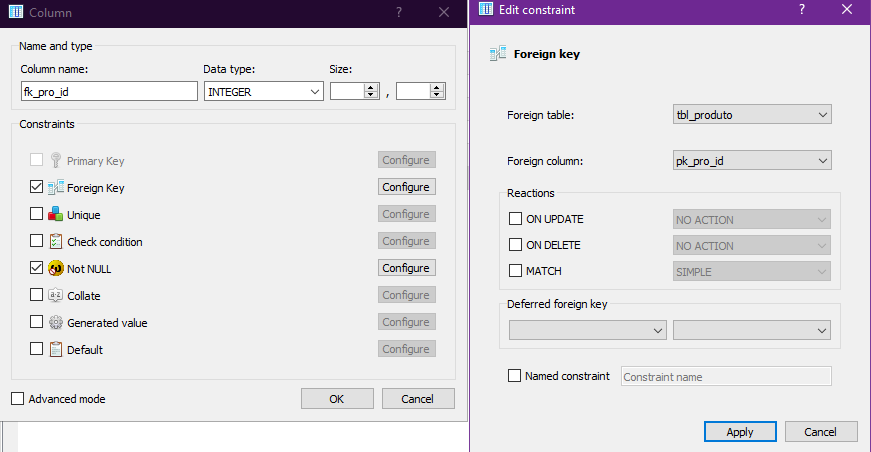


Aula 09 - Feito isso voltamos à tabela principal “tbl\_estoque” fazendo ligação entre elas, usando a chave estrangeira para vincular uma à outra para evitar problemas, por exemplo ao criar um item chamado maça, e este era vinculado a o outro produto, sem a chave estrangeira se alguém apagasse o item maça ia bugar tudo que fosse vinculado à maça.

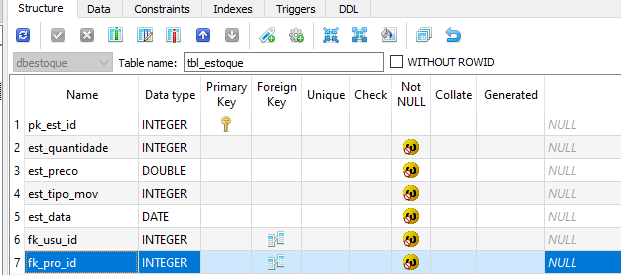
O padrão para criação da chave estrangeira é o nome fk “foreing Key” e a referencia que ela está vinculando, tudo conforme imagem abaixo.

No caso vamos vincula todas as tabelas



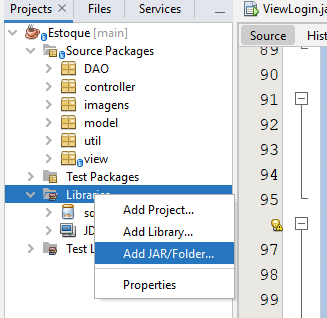


Por fim a tabela tbl\_estoque vai ficar conforme a imagem abaixo.

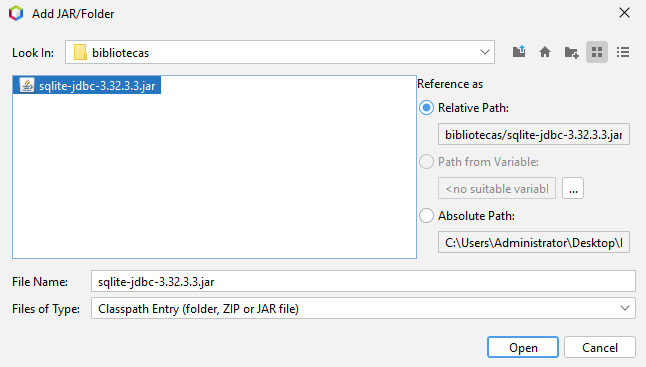


Feito isso, temos que procurar o drive do banco de dados SQLite referente à versão do netbeans que vc tiver, basta dar um google digitando a versão do SQLite e a do seu netbeans, no meu caso achei com o nome “sqlite-jdbc-3.32.3.3.jar”, que deve ser salvo na pasta do seu projeto, mais especificamente deve criar uma pasta chamada biblioteca, e salvar este arquivo lá.

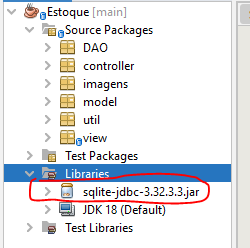
Feito isso, vai no Netbean clica com o botão direito em “libraries” Add JAR/Folder.



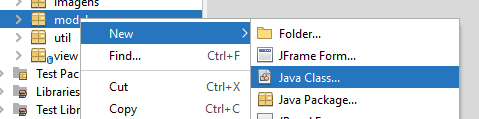
Feito isso, basta procurar o drive na pasta biblioteca dentro do seu projeto, seleciona-lo e clicar em open.



Feito isso ira ser adicionado o drive possibilitando a utilização do banco de dados SQLite.



Aula 10 - De incio vai ser criado a Java Class ModelUsuario.



Após criado o ModeUsuario.java, vamos criar uma classe publica chamada ModelUsuario, e dentro delas vamos criar 4 classes privadas, ficando assim:

public class ModelUsuario {

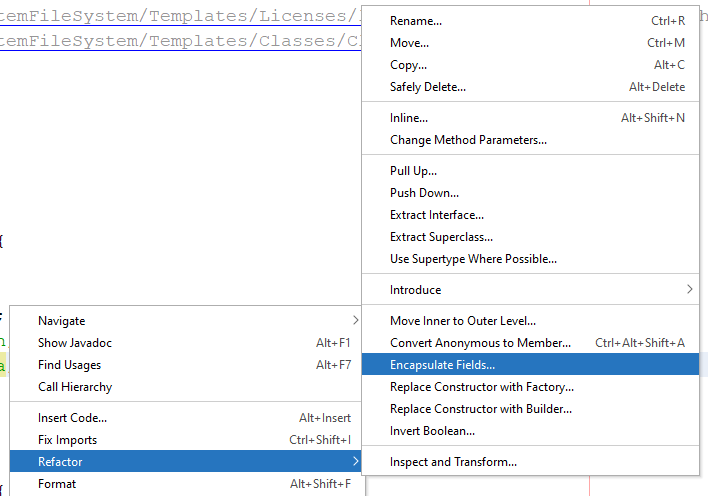
private int usuId;

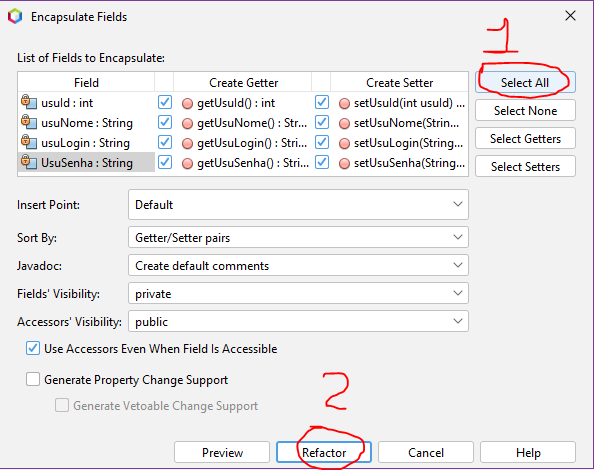
private String usuNome;

private String usuLogin;

private String UsuSenha;

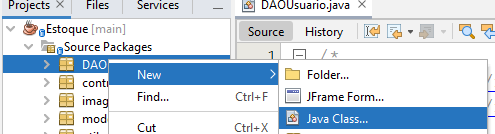
Após feito isso vamos encapsular os dados conforme imagem abaixo:

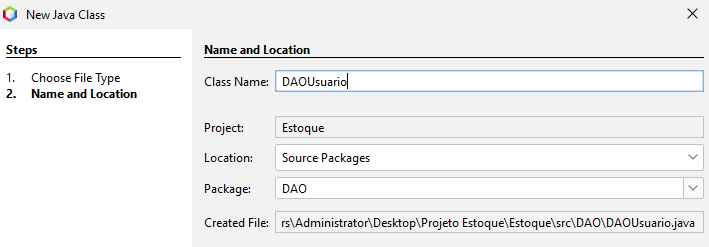
 Feito isso ira abrir a janela para mostrais quais dados encapsular, primeiro clicar em Select All para selecionar todos, depois em Refactor, no caso da imagem abaixo vai ter uns campos e umas bolinhas vermelhas a mais pois já é o projeto pronto.



Após clicar Refactor, ira criar or Gets and Sets.

Feito isso vamos para no DAO a JavaClass chamada DAOUsuario que serve para a vinculação ao banco de dados.





Para ela ser funcional devemos extender ela criando um extends e o nome da conexão ao SQLite.

public class DAOUsuario extends ConexaoSQLite{

Após isso sempre que quiser acessar o banco de dados basta digitar os comandos:

* conectar();
* desconectar();

Para iniciar e encerrar a conexão ao Banco.

Após detalhado isso, o código será:

public boolean salvarUsuarioDAO(ModelUsuario pModelUsuario){

conectar();

//executar SQL

String sql = "INSERT INTO tbl\_usuario("

+ "usu\_nome, "

+ "usu\_login, "

+ "usu\_senha) "

+ "VALUES (?,?,?)";

PreparedStatement preparedStatement = criarPreparedStatement(sql, Statement.RETURN\_GENERATED\_KEYS);

try {

preparedStatement.setString(1, pModelUsuario.getUsuNome());

preparedStatement.setString(2, pModelUsuario.getUsuLogin());

preparedStatement.setString(3, pModelUsuario.getUsuSenha());

preparedStatement.executeUpdate();

} catch (SQLException ex) {

Logger.getLogger(DAOUsuario.class.getName()).log(Level.SEVERE, null, ex);

return false;

}

desconectar();

return true;

}

Detalhando o código acima, será publico e o retorno será verdadeiro ou falso “true ou false”, pois deverá devolver a informação se foi salvo ou não, o método será nomeado salvarUsuarioDAO onde Salvar Usuario é o nome e DAO é para referenciar a classe dele entre parêntese devemos demonstrar qual objeto e para onde ele retorna, que será pModelUsuario, sendo o objeto apresentado mais abaixo no mesmo código.

Aula 11 - Continuação da outra aula “conectando ao “BD”.

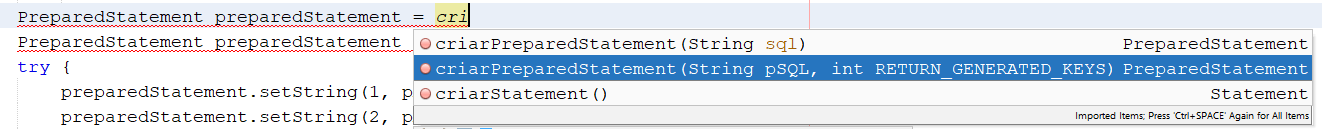
Para guardar o SQL cria-se uma variável String que colocamos o nome de sql, que recebe os dados, para boas práticas, ao usar comandos para SQL sempre se escreve em caixa alta.

Como estamos querendo inserir algo no banco de dados, colocamos o comando INSERT, INTO para indicar o local, que é a tbl\_usuario, e após demonstrar as colunas a serem inseridas e quanto aos valores, como vão ser inseridos pelo usuário, para cada coluna os VALUES colocamos uma interrogação, ficando conforme abaixo:

String sql = "INSERT INTO tbl\_usuario(usu\_nome, usu\_login, usu\_senha) VALUES (?,?,?)";

Pode ser feito da forma da aula anterior ou tudo na mesma linha igual acima.

Feito isso, vamos instanciar o PreparedStatement que cria um ID, bastando digitar parte do trecho, segurar ALT + espaço, que ira aparecer essa opção.



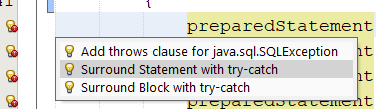
PreparedStatement preparedStatement = criarPreparedStatement(sql, Statement.RETURN\_GENERATED\_KEYS);

Basta completar com o que queremos que ele retorne, que é a geração da chave ID.

Feito isso passar quais campos a serem executados, que será o pModelUsuario.

No trecho da linha de comando preparedStatement.setString(1, pModelUsuario.getUsuNome()); o 1 indica a linha do código que queremos buscar, podendo ser colocar diretamente o que queremos, ficando assim preparedStatement.setString(usu\_nome, pModelUsuario.getUsuNome());

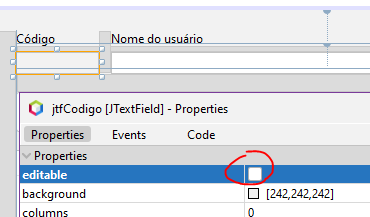
Vai dar um erro, e basta clicar na lâmpada e mandar o Netbeans criar o try catch.



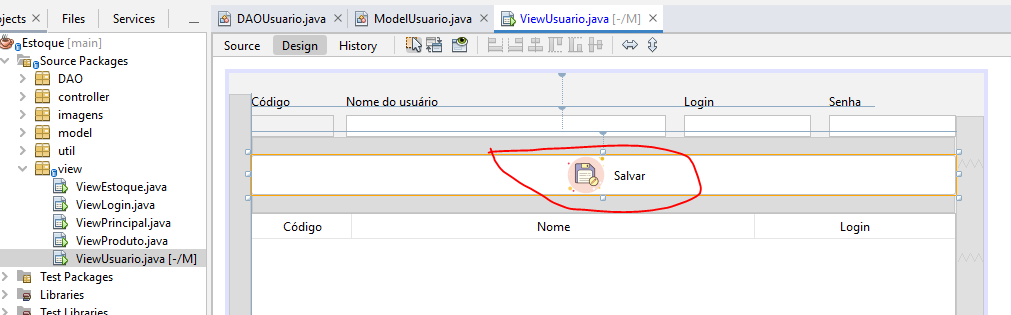
Feito a primeira linha, do PreparedStatement basta segurar CTRL + SHIFT + seta pra baixo que ele copia a linha que você está, bastando corrigir os trechos diferentes.

E para finalizar o preparedStatement, basta digitar o código preparedStatement.executeUpdate();

Feito isso, basta salvar o método na view.

Aula 12 - Inicia a aula falando da ViewUsuario e sobre o campo código que devemos desmarcar a opção do usuário edita-lo conforme imagem abaixo, também disponível no resumo da aula 04. 

Posteriormente, Vamos configurar o botão salvar, bastar dar clique duplo no mesmo.



Se clicou corretamente, ira para este trecho de código:

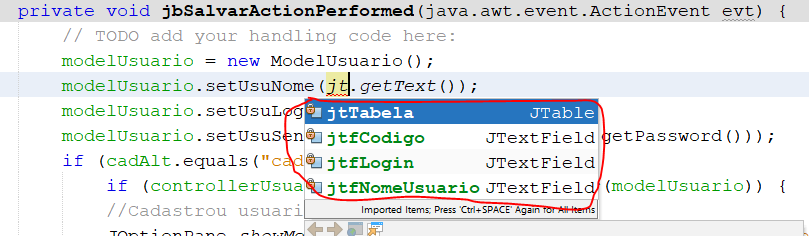
private void jbSalvarActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

Iremos precisar instanciar o objeto para estar salvando os dados, devendo primeiro instanciar o ModelUsuario, conforme código abaixo:

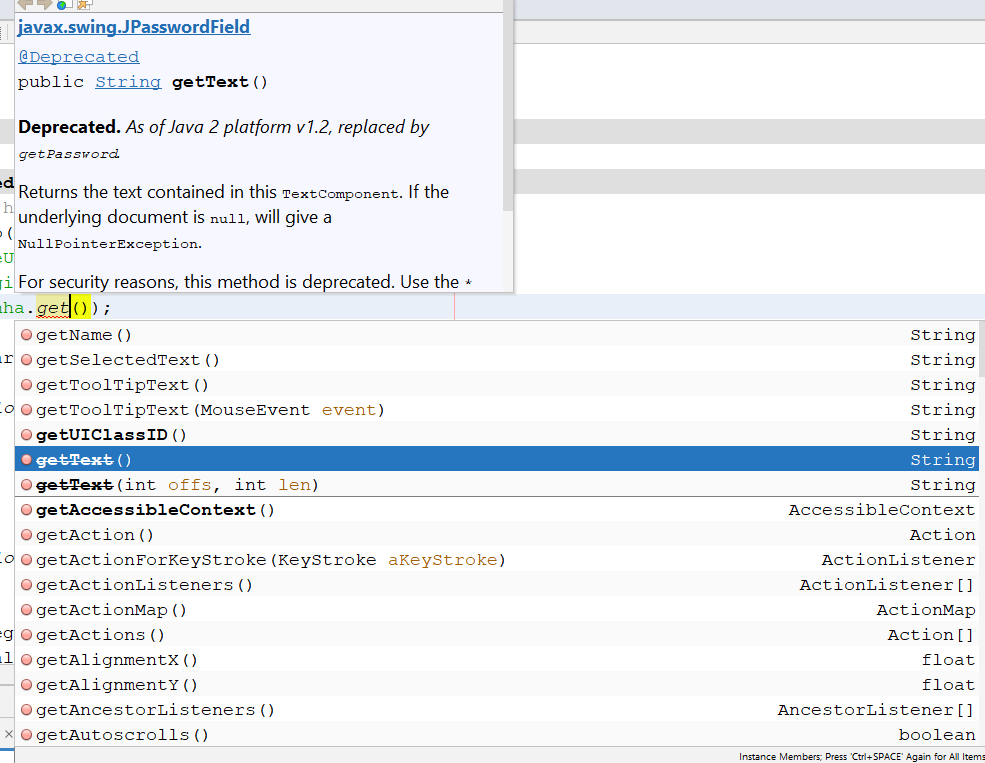
modelUsuario = new ModelUsuario();

Após feito isso vamos inserir os campos que renomeamos.



Fazer isso para os 4 campos, pois para salvar o sistema ira mandar para o banco de dados as informações de todos os campos necessários.

Em casos que aparecem igual o caso abaixo, ou seja o código aparece riscado, quer dizer que o mesmo já ficou obsoleto, devendo ser substituído para um novo modelo, que geralmente ao digitar o comando e apertar o CRTL + espaço, o netbeans já lhe da as opções..

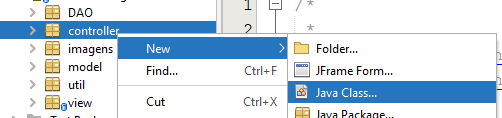


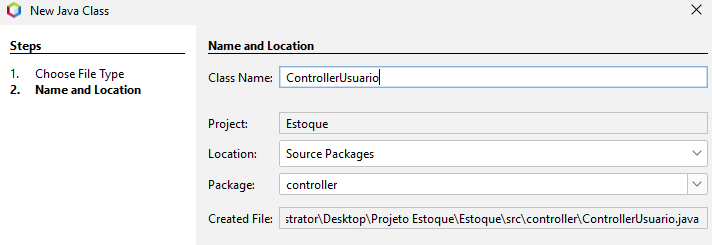
Neste caso foi usado o comando abaixo, que após digitar o getPassword, ele pede par converter em String, ficando da forma abaixo:



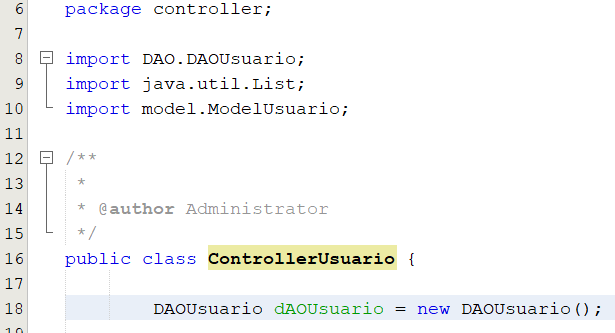
Codigo pronto, precisamos instância-lo pelo controller, e como ainda não o temos, vamos criar:

Botão direito em controller, New/JavaClasse, e vamos nomea-lo como ControllerUsuario.





Criado o controller, vamos iniciar o DAOUsuario, e ele já vai importa-lo.

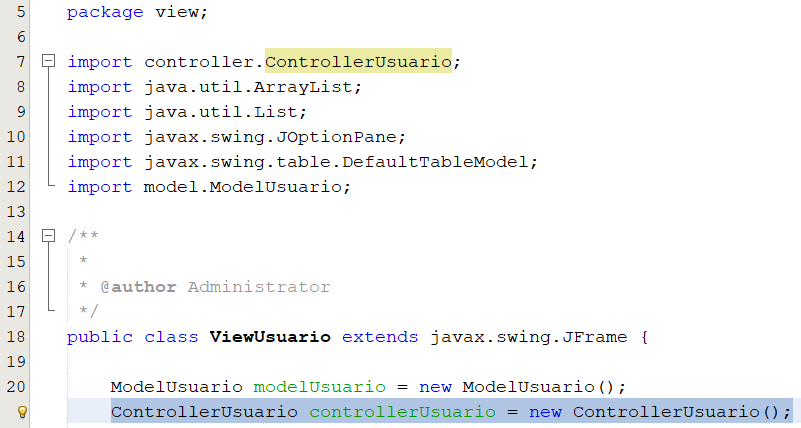


Lembrando que com o atalho CTRL+SHIFT+i o NetBeans vai analisar seu código e vai importar ou apagar os import des/necessários.

Feito o controller, vamos instância-lo na ViewUsuario, e para inicia-lo vamos praticamente no começo da View digitando o código abaixo:

ControllerUsuario controllerUsuario = new ControllerUsuario();

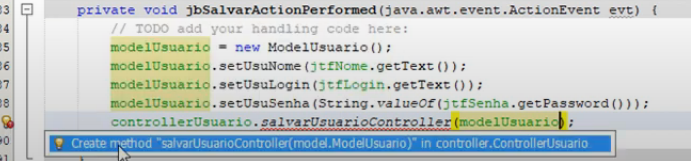
E importando os pacotes necessário, no caso o do ControllerUsuario que acabamos de criar.



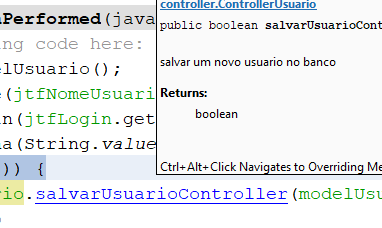
Depois, voltamos à onde estávamos instanciando o controller, logo abaixo de mostrar-mos ao botão Salvar quais campos deveria ser salvos, e criamos o seguinte código:

controllerUsuario.salvarUsuarioController(modelUsuario);

O NetBeans vai reclamar, e basta clicar na lâmpada e mandar ele criar o método.



Criado o método Segure CTRL e passe o mouse sobre o método criado que ele se tornara um link que ao clicar, leva direto ao método criado, faça isso.



Essa linha de baixo que foi criada é padrão, pode apagar, altere o código de void para bollean pois queremos que retorne algo sendo verdadeiro ou falso “public boolean”, e na linha abaixo, faça o retorno que queremos, ou seja, return this.dAOUsuario.salvarUsuarioDAO(modelUsuario); this remetendo a classe dAOUsuario, lembrar de sempre codificar usando CTRL + espaço.

Sempre comentar os métodos, e para faze-lo é fácil, basta ir logo antes do método, digitar /\*\* e dar enter, e digitar o que achar necessário.

Por fim, criado o necessário no controller, voltemos ao método, bastando segurar CTRL e clicar no salvarUsuarioDAO.

Aula 13 -

Aula 14 -

Aula 15 -

Aula 16 -

Aula 17 -

Aula 18 -

Aula 19 -

Aula 20 -

Aula 21 -

Aula 22 -

Aula 23 -

Aula 24 -

Aula 25 -

Aula 26 -

Aula 27 -