Passo 1: Crie a tabela no Netbeans com as informações abaixo:

CREATE TABLE usuarios (

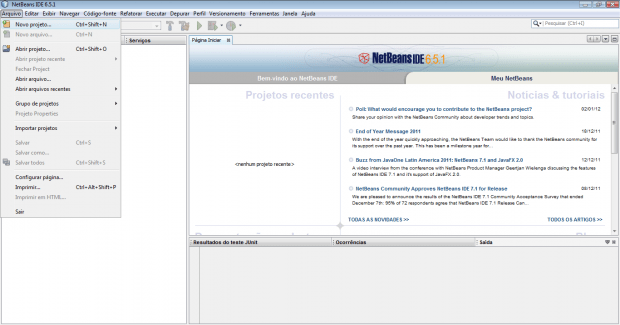
nome VARCHAR(255),

cpf VARCHAR(255),

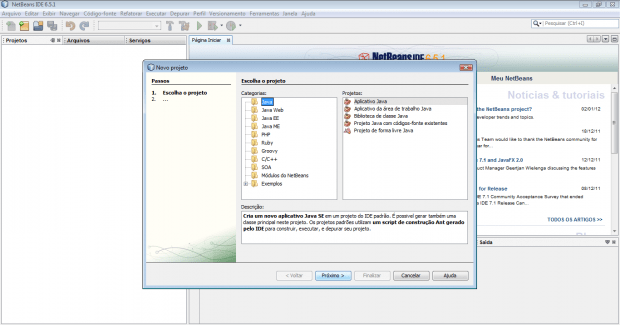
email VARCHAR(255),

telefone VARCHAR(255),

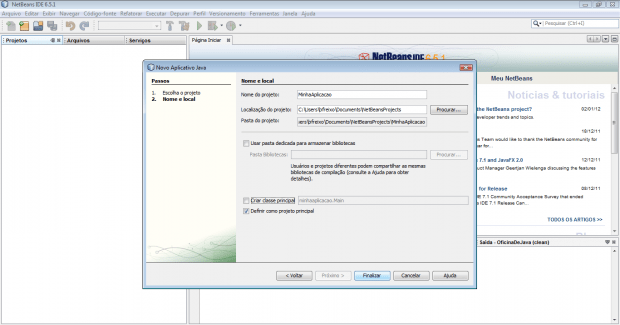
Passo 2: O Projeto Arquivo > Novo projeto;



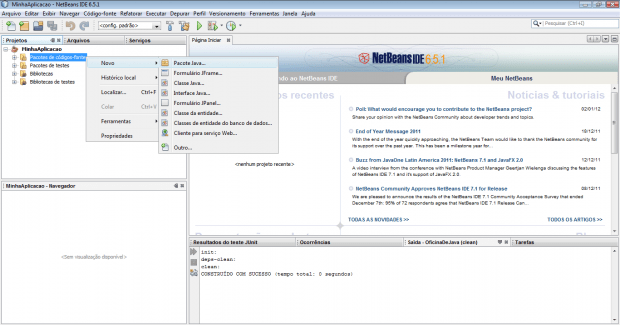
Na aba Novo projeto, clique em Java e em Aplicativo Java. Clique em próximo.



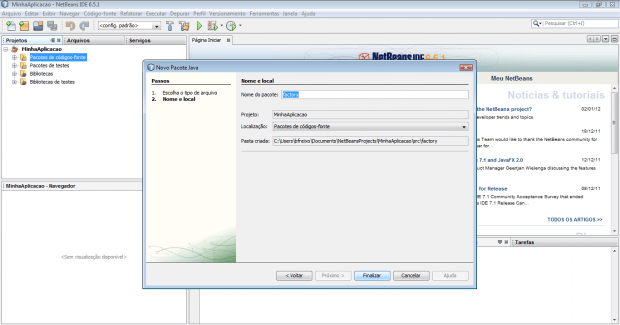
Na aba Novo aplicativo Java, em “Nome do projeto” digite MinhaAplicacao. Desmarque a opção “Criar classe principal”. Clique em finalizar.

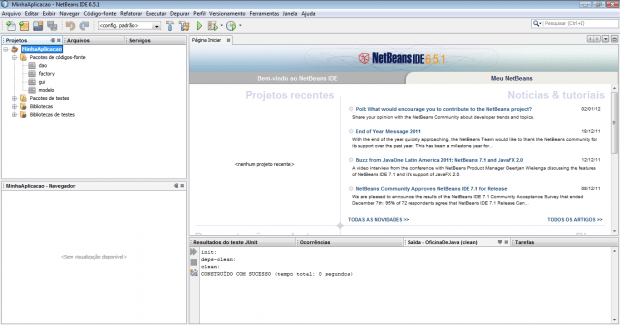


Agora vamos criar os pacotes ou packages. Clique com o botão direito em “Pacotes de códigos-fonte” e com o botão esquerdo do mouse escolha Novo > Pacote Java...

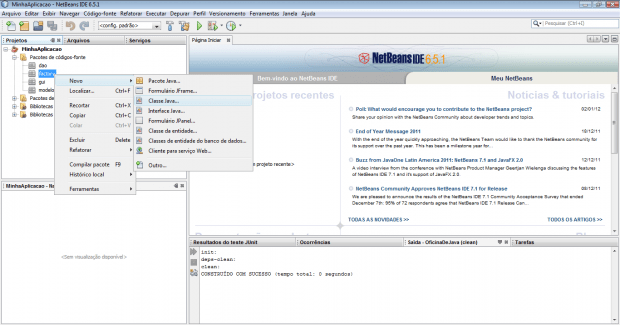


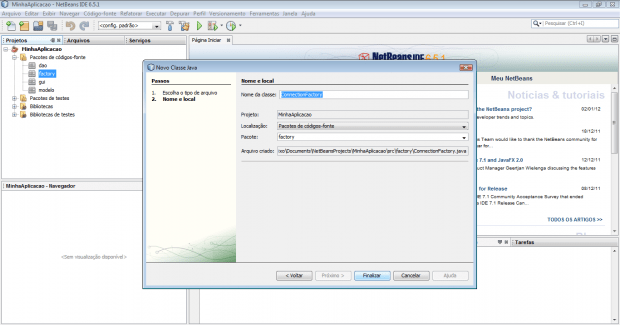
Na aba Novo Pacote Java digite "factory” para Nome do pacote. Clique em Finalizar.



Repita o processo de criação de pacote, criando os seguintes pacotes, além do pacote factory: modelo, dao, gui. Assim ficará a visão geral do projeto (no canto esquerdo do NetBeans):

Passo 3: Factory: Factory significa “fábrica” e ConnectionFactory significa fábrica de conexões. Factory será o nome do pacote e ConnectionFactory o nome da classe que fará a interface com o driver JDBC de conexão a qualquer banco que desejar. Por isso o nome “fábrica”, pois o JDBC permite a conexão a qualquer banco: MySQL, Postgree, Oracle, SQL Server, etc., somente alterando a linha do método “getConnection”. Vamos começar criando a classe ConnectionFactory no pacote factory. Vá com o botão direito até factory e clique com o botão esquerdo em Novo > Classe Java.



Na aba Novo Classe Java em Nome da Classe escolha o nome ConnectionFactory. Clique em Finalizar.

O script abaixo representa a classe de conexão ConnectionFactory. Copie e cole na classe ConnectionFactory:

package factory;/\*\*

\*

\* @author Simone

\*/

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

public class ConnectionFactory {

public Connection getConnection()

{

try

{

return DriverManager.getConnection("jdbc:mysql://localhost:3307/banco","root","1234");

}

catch(SQLException excecao)

{

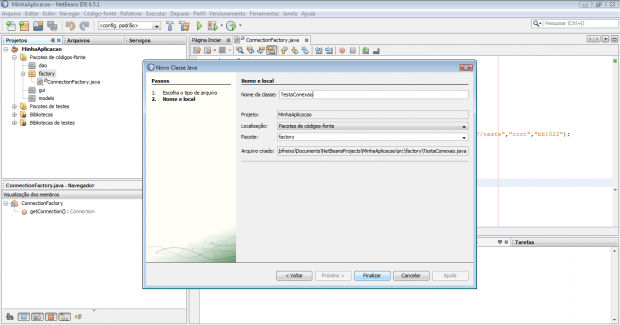
throw new RuntimeException(excecao);

}

}

}

Salve a alteração (CTRL+S). OBS.: não esqueça de salvar todas as alterações nos códigos ao decorrer do tutorial. OBS 2: altere "seu-nome-de-usuario" e "sua-senha" para as configurações do seu Banco de Dados. Vamos criar uma classe para testar a conectividade ao MySQL. Pode ser dentro do pacote factory mesmo... Coloquemos o nome TestaConexao:



Clique em Finalizar. Script da classe TestaConexao:

package factory;

/\*\*

\*

\* @author Simone

\*/

import java.sql.Connection;

import java.sql.SQLException;

public class TestaConexao {

public static void main(String[] args) throws SQLException {

try

(Connection connection = new ConnectionFactory().getConnection()) {

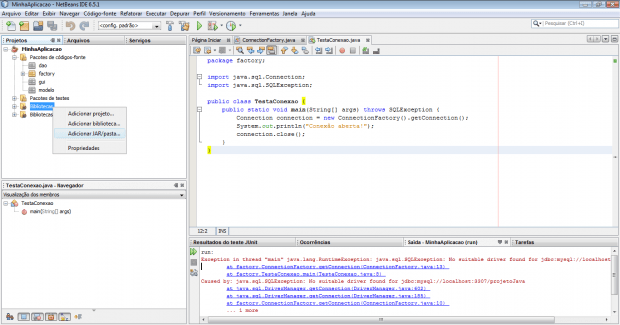
System.out.println("Conexão aberta!");

}

}

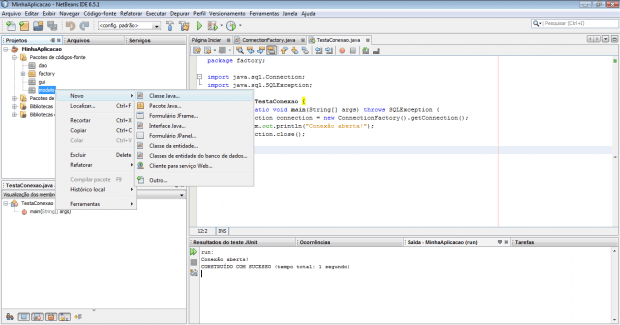
}

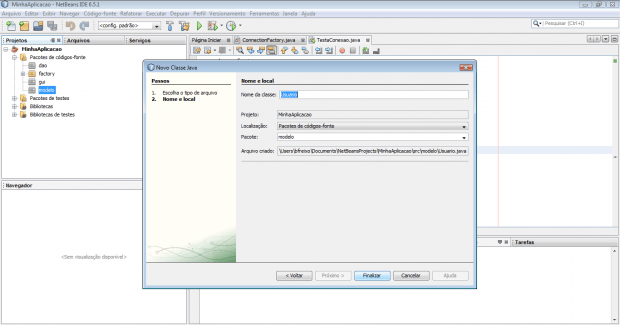
Depois de baixar o driver JDBC, vá em: Bibliotecas > Adicionar JAR/pasta...



Escolha o diretório onde instalou o driver JDBC e clique em Open.

Passo 4: Modelo: Agora, criemos a classe Usuario, dentro do pacote modelo: modelo > Novo > Classe Java > Usuario > Finalizar.





Crie as variáveis id (Long), nome, CPF, email, telefone (todas string) e os métodos getters e setters. Assim ficará o script da classe:

package modelo;

/\*\*

\*

\* @author Simone

\*/

public class Usuario {

String nome;

String cpf;

String email;

String telefone;

public String getCpf()

{

return cpf;

}

public void setCpf(String cpf)

{ this.cpf = cpf;

} public String getEmail() {

return email;

} public void setEmail(String email)

{ this.email = email;

}

public String getNome()

{ return nome;

}

public void setNome(String nome)

{ this.nome = nome;

}

public String getTelefone() {

return telefone;

}

public void setTelefone(String telefone)

{ this.telefone = telefone;

}

}

Passo 5: DAO: Crie no pacote DAO a classe UsuarioDAO: dao > Novo > Classe Java > UsuarioDAO > Finalizar. UsuarioDAO:

Eis o script abaixo da classe UsuarioDAO:

package dao;

/\*\*

\*

\* @author Simone

\*/

import factory.ConnectionFactory;

import modelo.Usuario;

import java.sql.\*;

import java.sql.PreparedStatement;

public class UsuarioDAO {

private Connection connection;

Long id;

String nome;

String cpf;

String email;

String telefone;

public UsuarioDAO(){

this.connection = new ConnectionFactory().getConnection();

} public void adiciona(Usuario usuario)

{

String sql = "INSERT INTO usuarios(nome,cpf,email,telefone) VALUES(?,?,?,?)";

try {

PreparedStatement stmt = connection.prepareStatement(sql);

stmt.setString(1, usuario.getNome());

stmt.setString(2, usuario.getCpf());

stmt.setString(3, usuario.getEmail());

stmt.setString(4, usuario.getTelefone());

stmt.execute();

stmt.close();

} catch (SQLException u)

{

throw new RuntimeException(u);

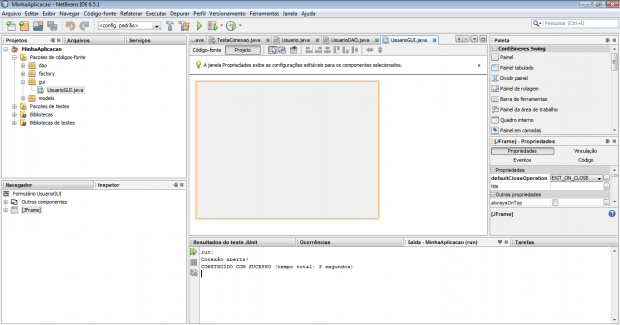
}

}

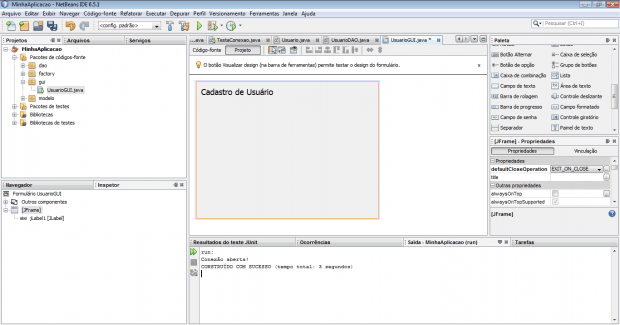
}

Passo 6: GUI (Graphical User Interface ou Interface Gráfica de Usuário) Nossa aplicação back-end está toda finalizada. Precisamos aprontar o front-end, isto é, a interface de usuário, a classe que será responsável pela interação com o usuário, ou seja, o formulário de entrada.

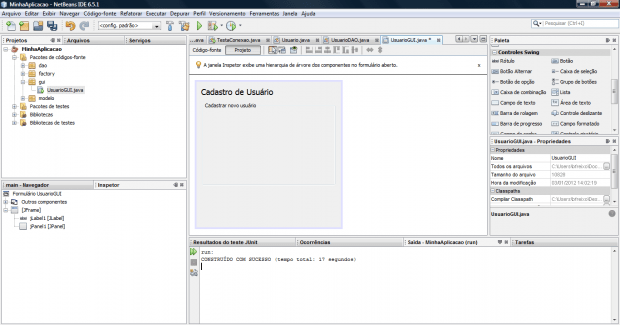
Vamos criar o formulário que será preenchido pelo usuário: gui > Novo > Formulario JFrame > UsuarioGUI > Finalizar. A seguinte tela aparecerá:



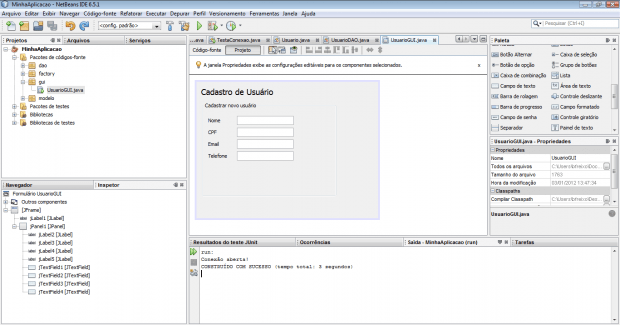
Para criarmos os elementos do formulário é necessário o arrastar e soltar do mouse. A esse processo, vou criar a sigla ASM para facilitar nosso entendimento. Arrasta-se os componentes SWING para o formulário. Portanto, quando eu chamar o nome do componente e colocar ao lado a sigla ASM, subentende-se que é para arrastar componentes da paleta e soltá-los no Formulário. À direita, na paleta de componentes, em Controles Swing, clique em Rótulo ASM. Escreva Cadastro de Usuário. Com o botão direito do mouse em cima do rótulo clique em Propriedades e em "font" escolha tamanho 18 e clique em OK. Veja:



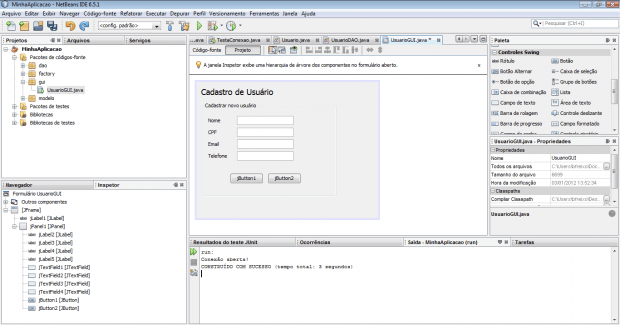
Agora na paleta de componentes, em Contêiners SWING, escolha Painel ASM. Clique com o botão direito do mouse e escolha Propriedades. Clique em border > Borda de título. Intitule “Cadastrar novo usuário”. Clique em OK e depois fechar. Veja:



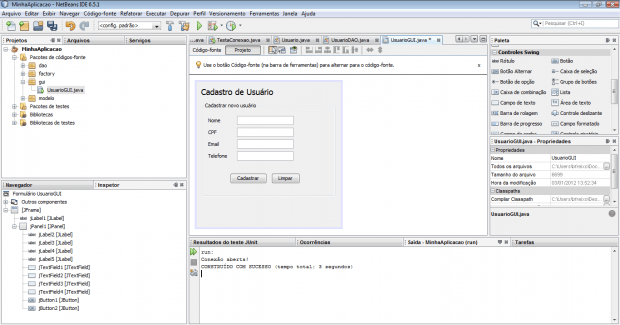
Escreva mais 4 rótulos dentro do painel: Nome, CPF, Email, Telefone. Agora escolha na paleta de componentes 4 campos de textos representando de forma respectiva cada um dos rótulos mencionados. Veja:



Finalmente vamos criar os botões. Na aba Paleta > Controles SWING, vá até Botão ASM. Crie dois botões, conforme mostra a imagem abaixo:



Escreva jButton1 como “Cadastrar” e jButton2 como “Limpar”. Veja:



Agora, fora do painel, crie o botão SAIR. Faça o mesmo processo: sobrescreva jButton3 para SAIR.

Passo 7: Evento SAIR Clique duas vezes no botão ”SAIR” para criarmos o evento. Na aba Código-Fonte, no método referente a jButton3, isto é, ao botão SAIR, digite: System.exit(0);

Este comando fecha a janela em execução. Dê um ALT+F6 e agora clique no botão SAIR. A janela será fechada. Passo 8: Evento LIMPAR Agora, na aba Projeto, dê dois cliques em Limpar.

No método jButton2ActionPerformed, na aba Código-fonte, escreva os seguintes scripts: jTextField1.setText("");

jTextField2.setText("");

jTextField3.setText("");

jTextField4.setText("");

Estes scripts são responsáveis por limpar ou apagar qualquer string escrita pelo usuário em cada um dos 4 campos de texto do formulário.

Passo 9: Evento CADASTRAR Precisamos criar o principal evento que é literalmente cadastrar o usuário. Para isso, vamos clicar duas vezes no botão “Cadastrar” e, na aba Código-fonte, no evento jButton1ActionPerformed ficará assim o código:

Usuario usuarios = new Usuario();

usuarios.setNome(jTextField1.getText());

usuarios.setCpf(jTextField2.getText());

usuarios.setEmail(jTextField3.getText());

usuarios.setTelefone(jTextField4.getText());

if ((jTextField1.getText().isEmpty()) || (jTextField2.getText().isEmpty()) || (jTextField3.getText().isEmpty()) || (jTextField4.getText().isEmpty()))

{

JOptionPane.showMessageDialog(null, "Os campos não podem retornar vazios");

}

else { UsuarioDAO dao = new UsuarioDAO();

dao.adiciona(usuarios);

JOptionPane.showMessageDialog(null, "Usuário "+jTextField1.getText()+" Inserido com sucesso! ");

}

jTextField1.setText("");

jTextField2.setText("");

jTextField3.setText("");

jTextField4.setText("");

}

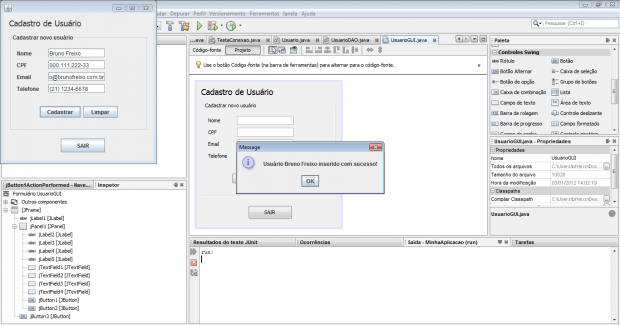
Certamente algumas mensagens de erro aparecerão. Isto porque temos que importar no início do código as classes Usuario (pacote modelo) e UsuarioDAO (pacote dao). Além destas, precisamos importar a classe JOptionPane, responsável pelas janelas de validação, aquelas que aparecem dizendo se o usuário foi ou não cadastrado, se os campos estão vazios, etc. Coloque estas linhas no início do código, abaixo de “package gui”, na aba Código-fonte:

import modelo.Usuario;

import dao.UsuarioDAO;

import javax.swing.JOptionPane;

Agora sim não aparecerá erro nenhum e o cadastro poderá ser feito. Faça um teste! Veja:



2. Crie o banco de dados escola e a tabela aluno, os campos são: RM, NOME, ENDERECO, TELEFONE, CEP, BAIRRO, CIDADE, ESTADO. O campo que será a chave primária será o RM do aluno.

- Crie o botão Inserir como no exemplo acima.

Desafio : Criar o botão Consulta.