**Conteúdo da disciplina:**

**Java Standard Edition (SE)**

**- Fundamentos**

- Tipos de dados

- Variáveis e constantes

- Estruturas de seleção e repetição

**- Criação de formulários (swing)**

- Caixas de texto, rótulos, caixas de seleção, botões

**- Interação com banco de dados**

- Driver JDBC (ponte com SGBD)

- CRUD (Create, Read, Update e Delete)

- **Conceitos de Orientação a objetos**

- Métodos

- Operadores de visibilidade

- Encapsulamento

- Construtores

- Herança

- Interface e classe abstrata

- Polimorfismo

**- UML**

- Engenharia de requisitos

- Diagrama de caso de uso

- Diagramas de classe

**Aula de hoje (27/02/2023):**

* Conteúdo da disciplina;
* Instalação dos recursos necessários;
* Criação do primeiro projeto Java.
* Tendências tecnológicas.
* Oportunidades de trabalho.

Link das vídeo aulas de Java no youtube:

<https://www.youtube.com/watch?v=Sv9saLJ8psE&list=PLmhTO6CBiifN2lUEIin5dLUGoJoWP3SVn&index=2>

Livro de Java gratuito:

<https://www.uninove.br/biblioteca-ebooks>

Link para conteúdo sobre blockchain e criptomoedas:

<https://www.youtube.com/@ecteruel/videos>

**Recursos necessários (downloads):**

Ou baixe a versão para 64 bits OU para 32 bits. Nunca baixe as duas.

**64 bits - x64 (para computadores mais potentes):**

Java SE Development Kit 19.0.2

<https://www.oracle.com/java/technologies/downloads/>

Ambiente de desenvolvimento para Java Netbeans

<https://netbeans.apache.org/download/index.html>

**32 bits - x86 (para computadores com potência discreta):**

Java Development Kit

<https://drive.google.com/file/d/1Nii2XPMAuvtA6LOhJ3JEure0CU1FlylE/view?usp=sharing>

<https://www.oracle.com/java/technologies/downloads/#java8-windows>

NetBeans 8.2 - 32 bits x86

<https://drive.google.com/file/d/1WxL90SxHNfsRvMy_t0yQDVlpikRgeFvA/view>

**Como instalar o Java Development Kit e o Netbeans no Linux (Ubuntu, Kali, Debian):**

**Atualizar o Linux:**

sudo su

cd Downloads

sudo apt-get update

sudo apt-get update

**Baixar o Java Development Kit:**

<https://www.oracle.com/java/technologies/downloads/#jdk20-linux>

**Instalar a JDK:**

sudo dpkg -i jdk-20\_linux-x64\_bin.deb

sudo apt-get install -f

**Baixar o Netbeans:**

https://netbeans.apache.org/download/index.html

**Instalar o Netbeans:**

sudo dpkg -i apache-netbeans\_17-1\_all.deb

sudo apt-get install -f

**Executar o Netbeans:**

netbeans

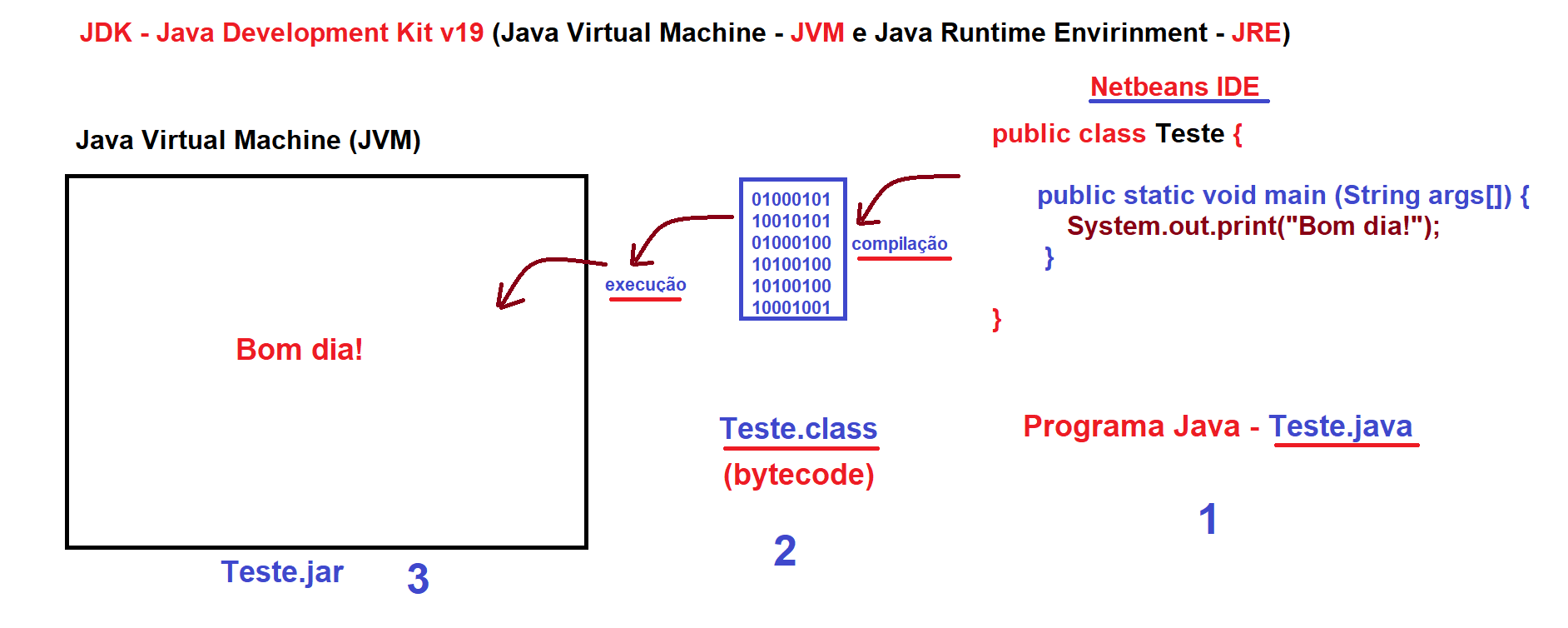
dpkg --list | grep netbeans

sudo apt-get remove --purge apache-netbeans

**Aula de hoje (28/02/2023):**

* Tipos de dados em Java (variáveis)
* Ajustes no ambiente Java e Netbeans

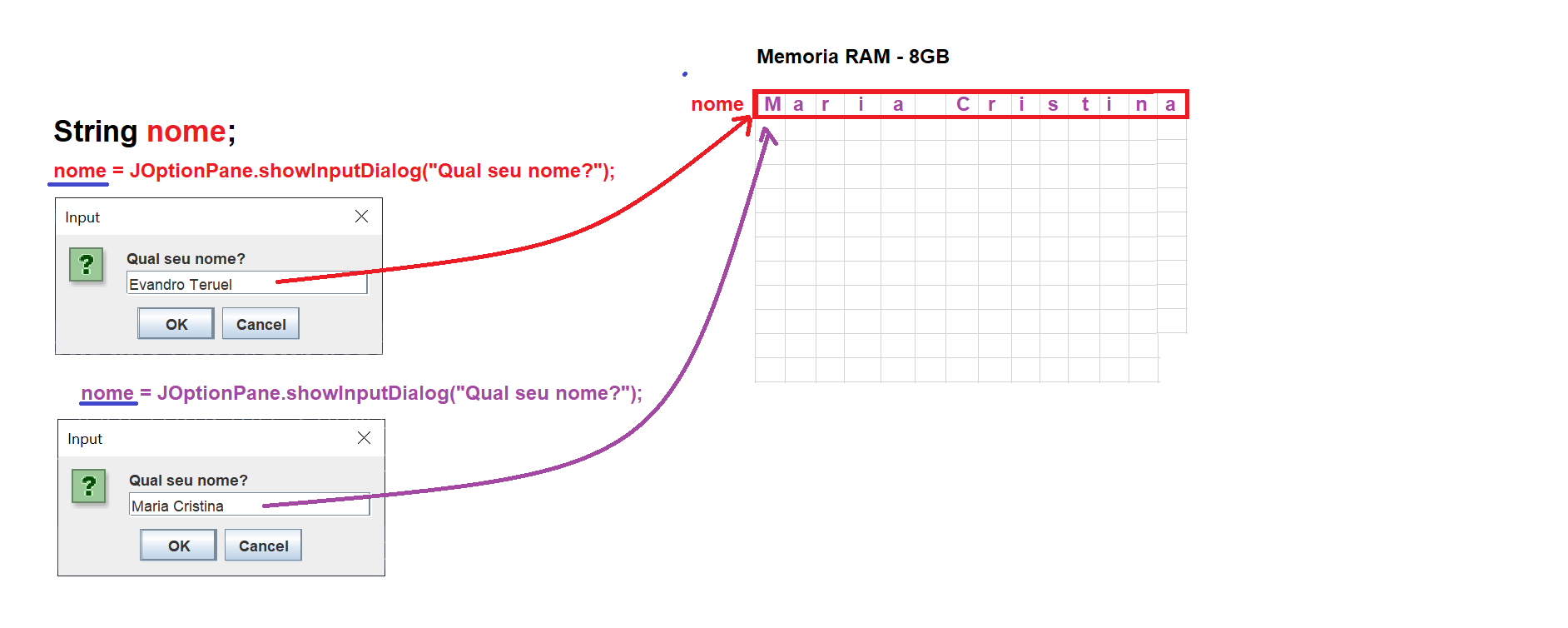
Iniciaremos às 9h00 em ponto.



Para quem está sem computador, acesso o ambiente de programação online:

<https://replit.com/login>

O que é variável?



import javax.swing.JOptionPane;

public class Teste {

public static void main(String[] args) {

String nome;

nome = JOptionPane.showInputDialog("Qual seu nome?");

JOptionPane.showMessageDialog(null,"Alo Mundo!");

}

}

**Aula de hoje (06/03/2023):**

* Tipos de dados em Java (variáveis)
* Exemplos

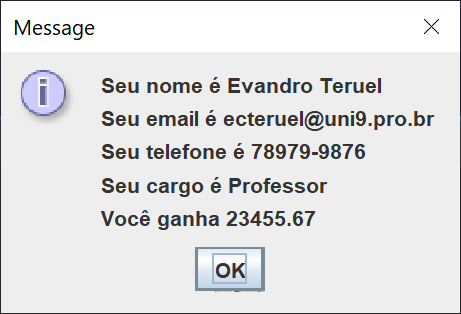
import javax.swing.JOptionPane;

javax

swing

JOptionPane.java

JOptionPane.showMessageDialog(null, "Seu nome é " + nome + "\nSeu email é " + email + "\nSeu telefone é " + telefone + "\nSeu cargo é " + cargo + "\nVocê ganha " + salario);



**Programa feito em aula:**

import javax.swing.JOptionPane;

public class Exemplo {

public static void main(String[] args) {

// Cadastrar nome, email, telefone, idade, cargo, salário

//Declaração de variáveis

String nome, email, cargo, telefone;

byte idade; //Inteiros - byte, short, int e long

double salario; //Número real - float, double

//Entrada de dados - cadastro

nome = JOptionPane.showInputDialog("Qual seu nome?");

email = JOptionPane.showInputDialog("Qual seu e-mail?");

cargo = JOptionPane.showInputDialog("Qual seu cargo?");

telefone = JOptionPane.showInputDialog("Qual seu telefone?");

idade = Byte.parseByte(JOptionPane.showInputDialog("Qual sua idade?"));

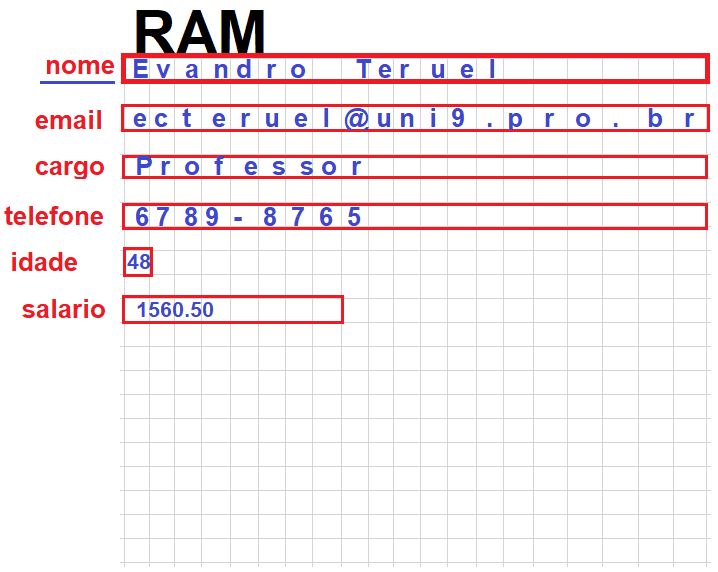
salario = Double.parseDouble(JOptionPane.showInputDialog("Qual seu salário?"));

//Exibir os dados armazenados

JOptionPane.showMessageDialog(null, "Seu nome é " + nome + "\nSeu email é " + email + "\nSeu telefone é " + telefone + "\nSeu cargo é " + cargo + "\nVocê ganha " + salario + "\nVocê tem " + idade + " anos");

}

}

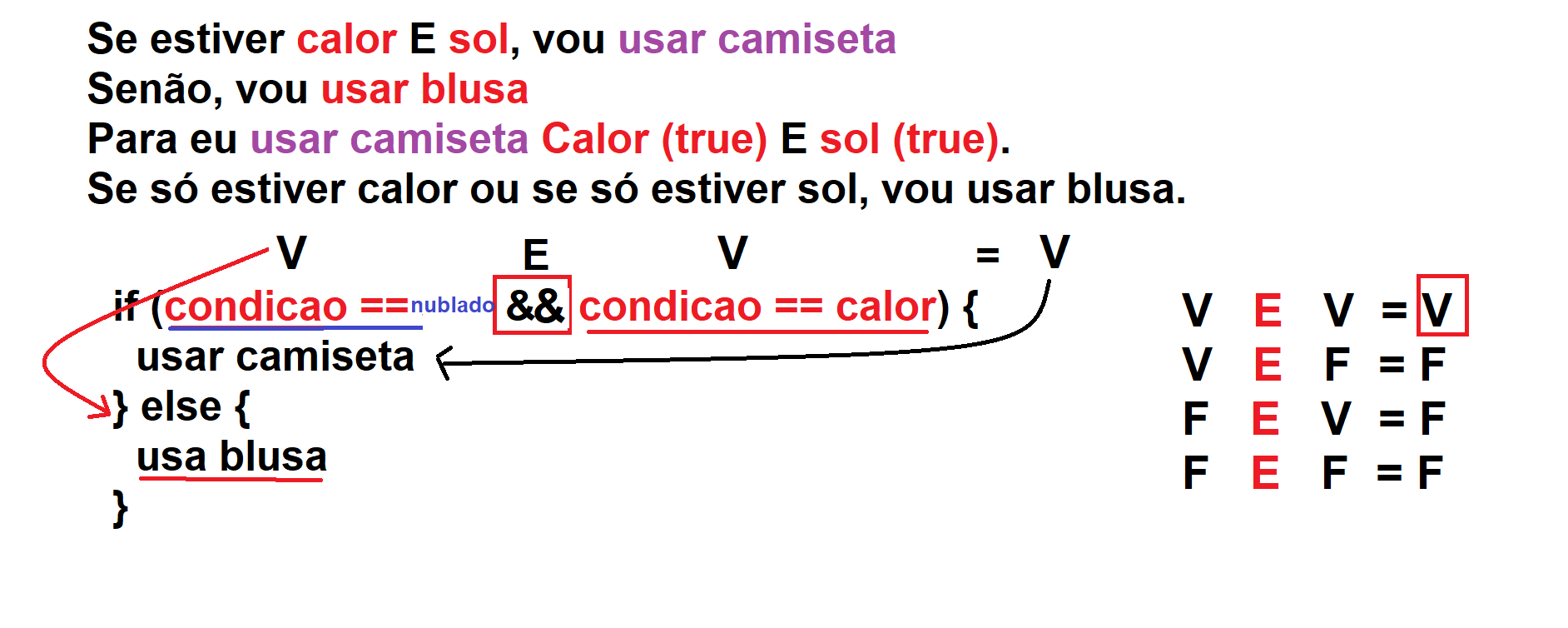


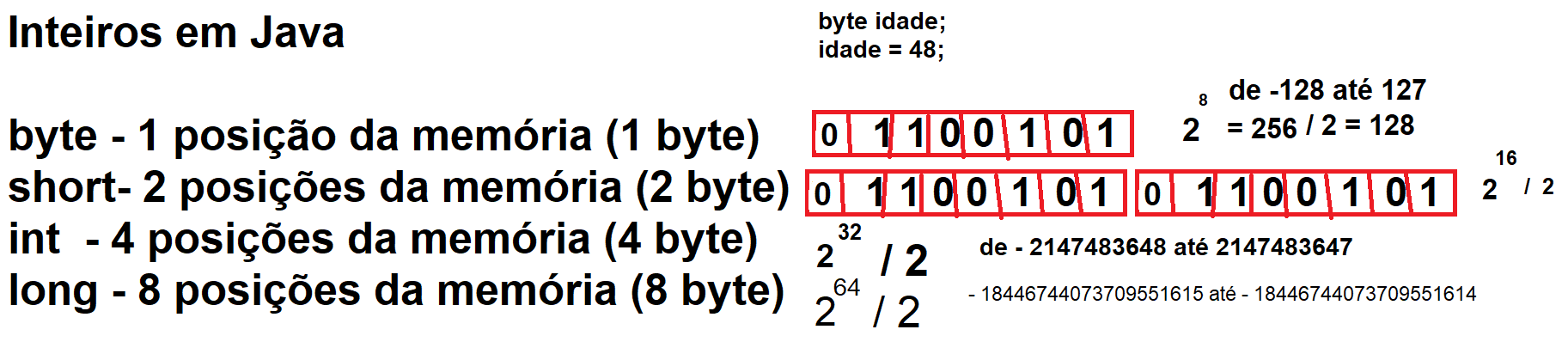
Material da disciplina no Google Drive:

<https://drive.google.com/drive/folders/1WcaRE6UhIFX0C4pJY2CmxDxVRNs-UTATh3nixRPLzJ861djx4cxyPI_wfNNH_tP-pNQaBTPU?usp=sharing>

**Aula de hoje (07/03/2023):**

* Tipos de dados em Java (variáveis)
* Comparações (estruturas de seleção)
* Exemplos





**Reais em Java**

float - 4 posições da memória (4 bytes)

double - 8 posições da memória (8 bytes)

O tipo recomendado para inteiro é **int**.

O tipo recomendado para real é **double**.

import javax.swing.JOptionPane;

public class Exemplo {

public static void main(String[] args) {

// Cadastrar nome, email, telefone, idade, cargo, salário

//Declaração de variáveis

String nome, email, cargo, telefone;

short idade; //Inteiros - byte, short, int e long

float salario; //Número real - float, double

long ra;

//Entrada de dados - cadastro

ra = Long.parseLong(JOptionPane.showInputDialog("Qual seu RA?"));

nome = JOptionPane.showInputDialog("Qual seu nome?");

email = JOptionPane.showInputDialog("Qual seu e-mail?");

cargo = JOptionPane.showInputDialog("Qual seu cargo?");

telefone = JOptionPane.showInputDialog("Qual seu telefone?");

idade = Short.parseShort(JOptionPane.showInputDialog("Qual sua idade?"));

salario = Float.parseFloat(JOptionPane.showInputDialog("Qual seu salário?"));

//Exibir os dados armazenados

JOptionPane.showMessageDialog(null, "Seu nome é " + nome + "\nSeu email é " + email + "\nSeu telefone é " + telefone + "\nSeu cargo é " + cargo + "\nVocê ganha " + salario + "\nVocê tem " + idade + " anos");

//Classificar a pessoa de acordo com a classe social A, B, C, D, E

if (salario <= 2900) {

JOptionPane.showMessageDialog(null, nome + " você pertence à classe D");

} else if (salario > 2900 && salario <= 7100) {

JOptionPane.showMessageDialog(null, nome + " você pertence à classe C");

} else if (salario > 7100 && salario <= 22000) {

JOptionPane.showMessageDialog(null, nome + " você pertence à classe B");

} else {

JOptionPane.showMessageDialog(null, nome + " você pertence à classe A");

}

}

}

**Outra forma:**

import javax.swing.JOptionPane;

public class Exemplo {

public static void main(String[] args) {

// Cadastrar nome, email, telefone, idade, cargo, salário

//Declaração de variáveis

String nome, email, cargo, telefone;

short idade; //Inteiros - byte, short, int e long

float salario; //Número real - float, double

long ra;

//Entrada de dados - cadastro

ra = Long.parseLong(JOptionPane.showInputDialog("Qual seu RA?"));

nome = JOptionPane.showInputDialog("Qual seu nome?");

email = JOptionPane.showInputDialog("Qual seu e-mail?");

cargo = JOptionPane.showInputDialog("Qual seu cargo?");

telefone = JOptionPane.showInputDialog("Qual seu telefone?");

idade = Short.parseShort(JOptionPane.showInputDialog("Qual sua idade?"));

salario = Float.parseFloat(JOptionPane.showInputDialog("Qual seu salário?"));

//Exibir os dados armazenados

JOptionPane.showMessageDialog(null, "Seu nome é " + nome + "\nSeu email é " + email + "\nSeu telefone é " + telefone + "\nSeu cargo é " + cargo + "\nVocê ganha " + salario + "\nVocê tem " + idade + " anos");

//Classificar a pessoa de acordo com a classe social A, B, C, D, E

if (salario <= 2900) {

JOptionPane.showMessageDialog(null, nome + " você pertence à classe D");

} else if (salario <= 7100) {

JOptionPane.showMessageDialog(null, nome + " você pertence à classe C");

} else if (salario <= 22000) {

JOptionPane.showMessageDialog(null, nome + " você pertence à classe B");

} else {

JOptionPane.showMessageDialog(null, nome + " você pertence à classe A");

}

}

}

**Exercício**

Faça um programa Java que pergunte o **nome** e a **idade**. Use a variável idade do tipo int.

Se a idade for até 18 anos, exiba a mensagem "Você é Jovem”

Se a idade for maior que 18 anos e menor ou igual a 30 anos, exiba a mensagem "Você está na meia idade”.

Se a idade for maior que 30 anos e menor que 60, exiba a mensagem "Você está velho”

Se a idade for acima de 60 exiba a mensagem “Ihhh, ferrou”.

**Aula de hoje (13/03/2023):**

* Criação de telas (formulários);
* Exemplos.

Gera janelas pop-up.

**JOptionPane.java**

JOptionPane.showInputDialog("");

JOptionPane.showMessageDialogDialog(null,"");

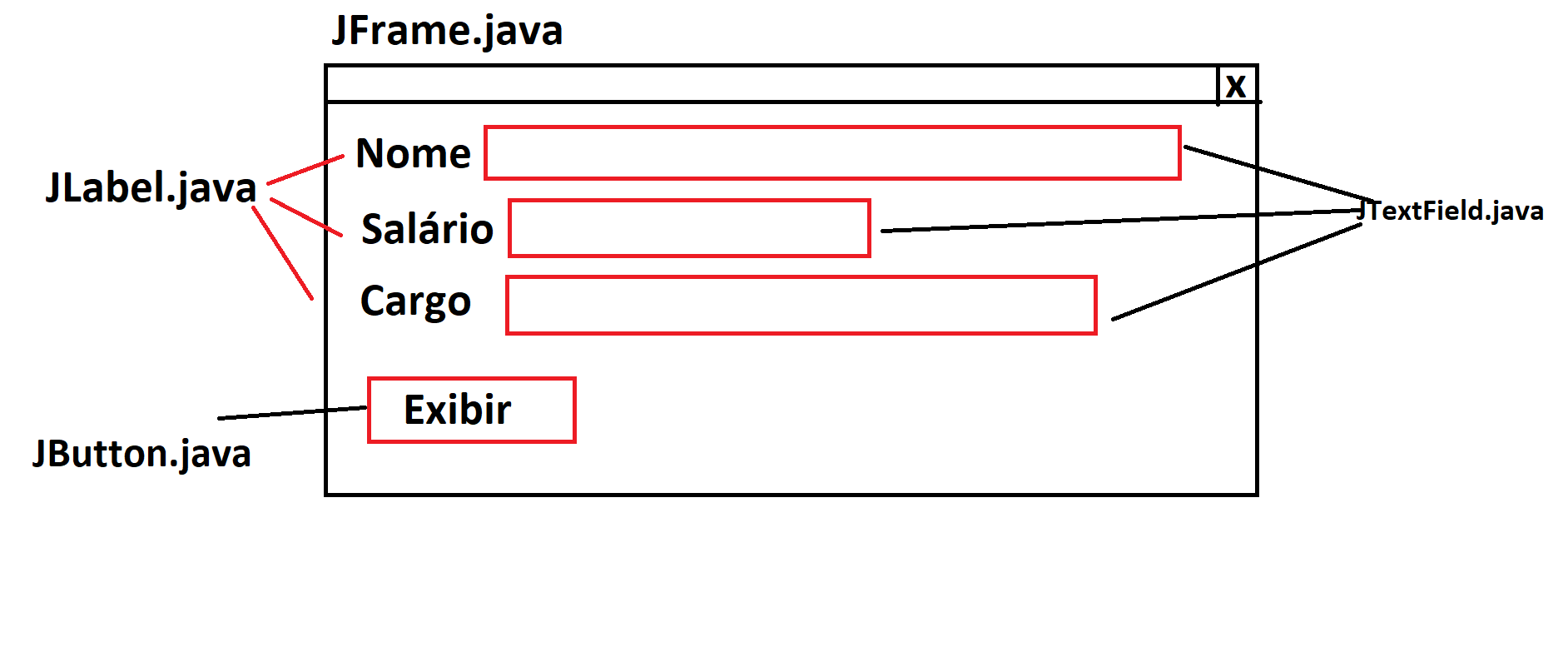
Criação de formulários

**JFrame.java**

**JTextField.java**

**JLabel.java**

**JButton.java**

****

**import javax.swing.JFrame;**

**javax**

**swing**

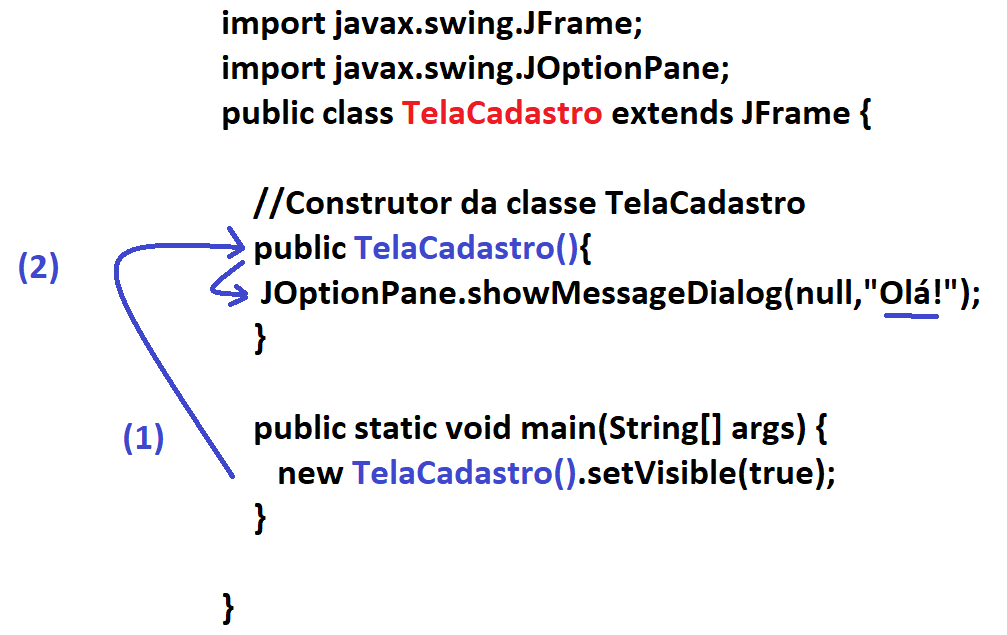
**JFrame.java**

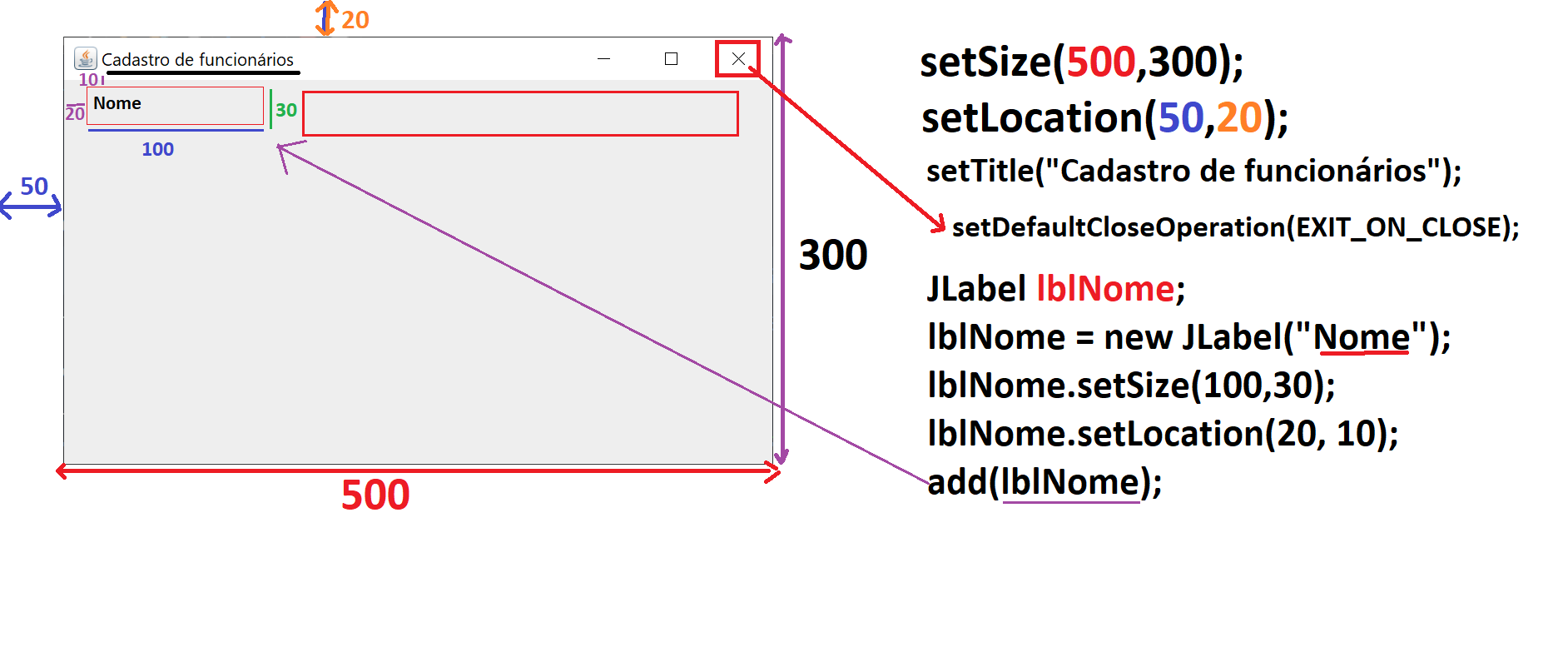
**import javax.swing.JOptionPane;**

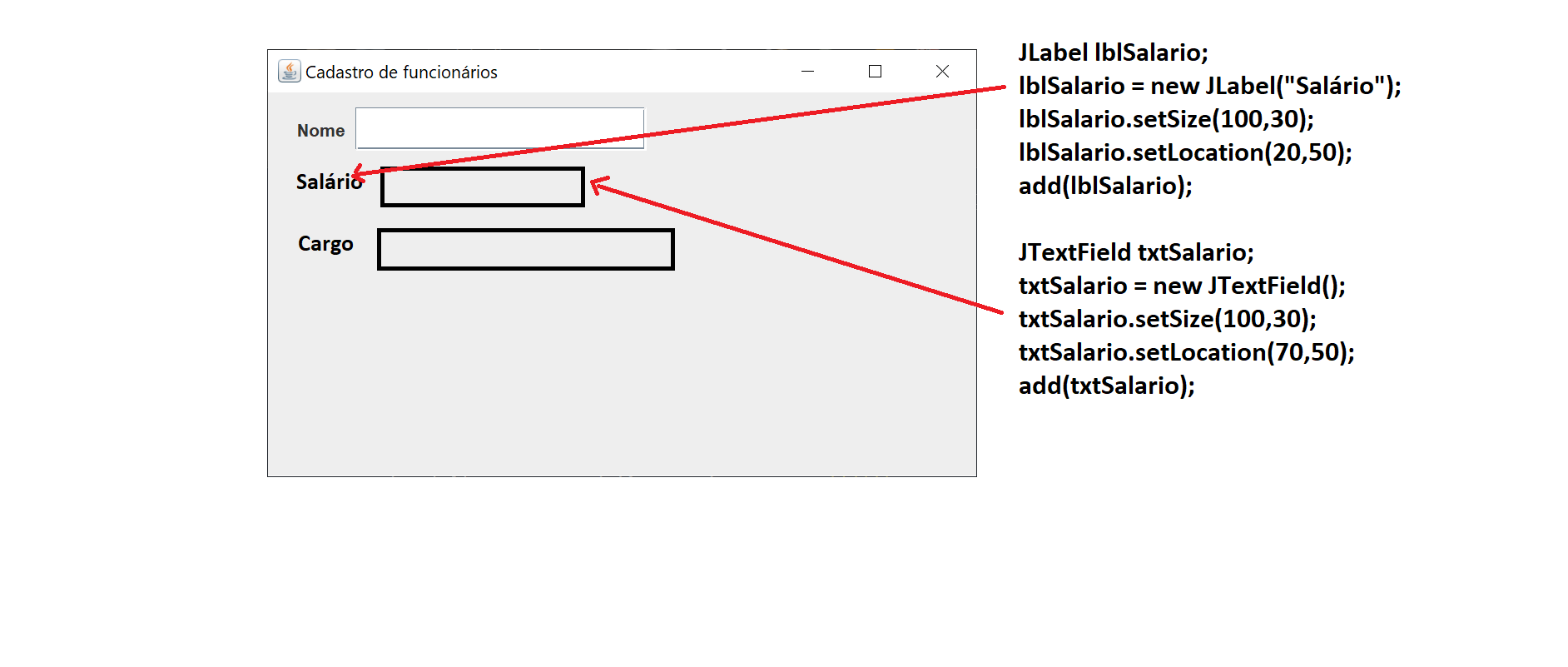
**javax**

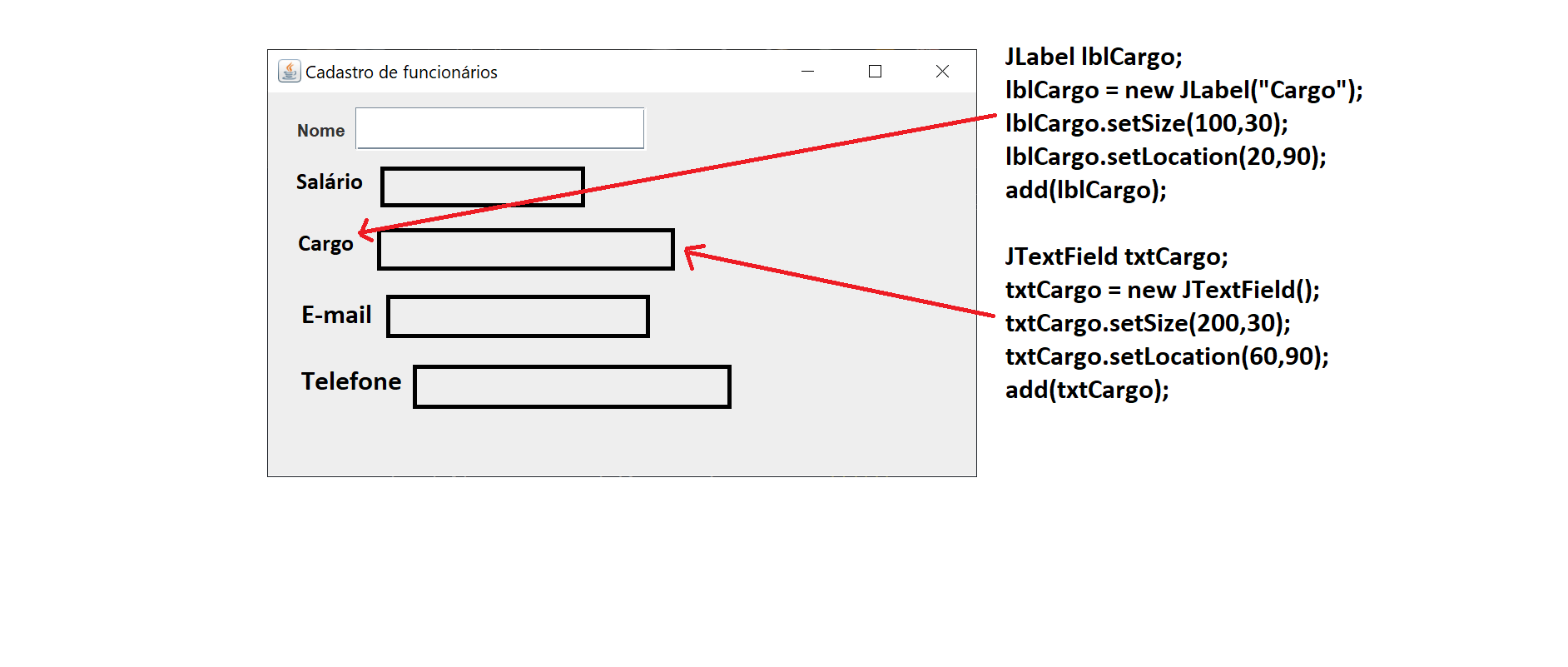
**swing**

**JOptionPane.java**









**Programa da aula de hoje:**

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JTextField;

public class TelaCadastro extends JFrame {

//Construtor da classe TelaCadastro

public TelaCadastro() {

//Comandos da janela

setSize(500, 300); //largura,altura

setLocation(50, 20); //margem sup, margem esq

setTitle("Cadastro de funcionários");

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setLayout(null);

//Conteúdo da janela

JLabel lblNome;

lblNome = new JLabel("Nome");

lblNome.setSize(100, 30);

lblNome.setLocation(20, 10);

add(lblNome);

JTextField txtNome;

txtNome = new JTextField();

txtNome.setSize(300, 30);

txtNome.setLocation(60, 10);

add(txtNome);

JLabel lblSalario;

lblSalario = new JLabel("Salário");

lblSalario.setSize(100, 30);

lblSalario.setLocation(20, 50);

add(lblSalario);

JTextField txtSalario;

txtSalario = new JTextField();

txtSalario.setSize(100, 30);

txtSalario.setLocation(70, 50);

add(txtSalario);

JLabel lblCargo;

lblCargo = new JLabel("Cargo");

lblCargo.setSize(100, 30);

lblCargo.setLocation(20, 90);

add(lblCargo);

JTextField txtCargo;

txtCargo = new JTextField();

txtCargo.setSize(200, 30);

txtCargo.setLocation(60, 90);

add(txtCargo);

}

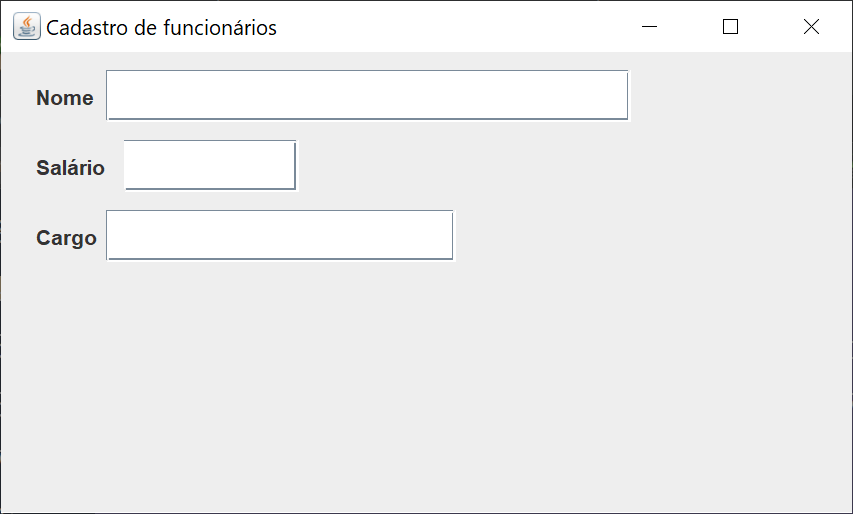
public static void main(String[] args) {

new TelaCadastro().setVisible(true);

}

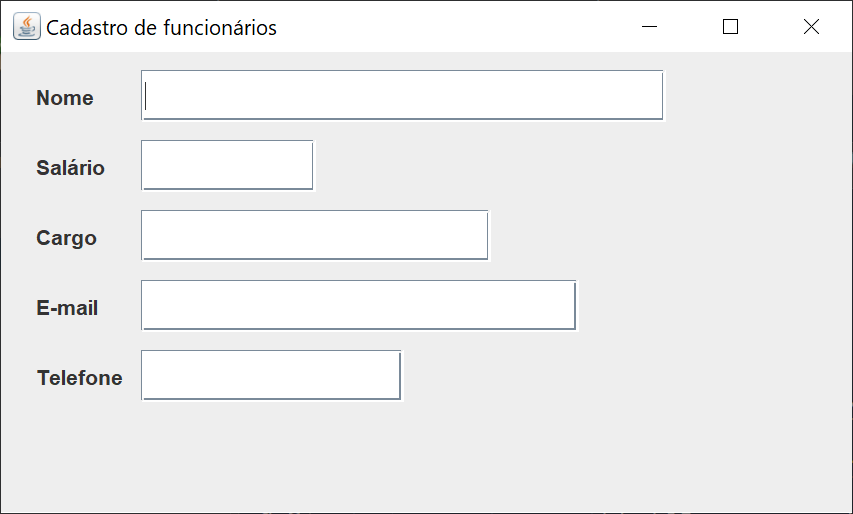
}

**Programa da aula de hoje:**



**Exercício**

Adicionar e-mail e telefone na tela que criamos em aula. Você deve gerar a tela abaixo.



**Aula de hoje (14/03/2023):**

* Criação de telas (formulários);
* Programação de eventos;
* Criação de tela com arrastar/soltar (modo nutella);
* Exemplos.

Correção do exercício:

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JTextField;

public class TelaCadastro extends JFrame implements ActionListener {

//Declaração das variáveis

JLabel lblNome;

JTextField txtNome;

JLabel lblSalario;

JTextField txtSalario;

JLabel lblCargo;

JTextField txtCargo;

JLabel lblEmail;

JTextField txtEmail;

JLabel lblTelefone;

JTextField txtTelefone;

JButton btnSalvar;

JButton btnPesquisar;

//Construtor da classe TelaCadastro

public TelaCadastro() {

//Comandos da janela

setSize(500, 300); //largura,altura

setLocation(50, 20); //margem sup, margem esq

setTitle("Cadastro de funcionários");

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setLayout(null);

//Conteúdo da janela

lblNome = new JLabel("Nome");

lblNome.setSize(100, 30);

lblNome.setLocation(20, 10);

add(lblNome);

txtNome = new JTextField();

txtNome.setSize(300, 30);

txtNome.setLocation(80, 10);

add(txtNome);

lblSalario = new JLabel("Salário");

lblSalario.setSize(100, 30);

lblSalario.setLocation(20, 50);

add(lblSalario);

txtSalario = new JTextField();

txtSalario.setSize(100, 30);

txtSalario.setLocation(80, 50);

add(txtSalario);

lblCargo = new JLabel("Cargo");

lblCargo.setSize(100, 30);

lblCargo.setLocation(20, 90);

add(lblCargo);

txtCargo = new JTextField();

txtCargo.setSize(200, 30);

txtCargo.setLocation(80, 90);

add(txtCargo);

lblEmail = new JLabel("E-mail");

lblEmail.setSize(100, 30);

lblEmail.setLocation(20, 130);

add(lblEmail);

txtEmail = new JTextField();

txtEmail.setSize(250, 30);

txtEmail.setLocation(80, 130);

add(txtEmail);

lblTelefone = new JLabel("Telefone");

lblTelefone.setSize(100, 30);

lblTelefone.setLocation(20, 170);

add(lblTelefone);

txtTelefone = new JTextField();

txtTelefone.setSize(150, 30);

txtTelefone.setLocation(80, 170);

add(txtTelefone);

btnSalvar = new JButton("Salvar");

btnSalvar.setSize(120, 30);

btnSalvar.setLocation(20, 210);

//Chama o método actionPerformed abaixo

btnSalvar.addActionListener(this);

add(btnSalvar);

btnPesquisar = new JButton("Pesquisar");

btnPesquisar.setSize(120, 30);

btnPesquisar.setLocation(150, 210);

//Chama o método actionPerformed abaixo

btnPesquisar.addActionListener(this);

add(btnPesquisar);

}

public static void main(String[] args) {

new TelaCadastro().setVisible(true);

}

@Override

public void actionPerformed(ActionEvent e) {

if(e.getSource() == btnSalvar){

JOptionPane.showMessageDialog(null,"Clicou salvar");

} else if (e.getSource()== btnPesquisar){

JOptionPane.showMessageDialog(null,"Clicou pesquisar");

}

}

}

**Aula de hoje (20/03/2023):**

* Criação de telas (formulários);
* Revisão.

Material da disciplina no Google Drive:

<https://drive.google.com/drive/folders/1WcaRE6UhIFX0C4pJY2CmxDxVRNs-UTATh3nixRPLzJ861djx4cxyPI_wfNNH_tP-pNQaBTPU?usp=sharing>

Link das vídeo aulas de Java no youtube:

<https://www.youtube.com/watch?v=Sv9saLJ8psE&list=PLmhTO6CBiifN2lUEIin5dLUGoJoWP3SVn&index=2>

**Criação de tela - Modo 1 - Usando a ferramenta JOptionPane.java**

import javax.swing.JOptionPane;

public class Teste{

public static void main(String[] args) {

String nome, cidade, bairro, email, telefone, endereco;

byte idade;

long ra;

double rendaFamiliar;

nome = JOptionPane.showInputDialog("Nome:");

endereco = JOptionPane.showInputDialog("Endereço:");

cidade = JOptionPane.showInputDialog("Cidade:");

bairro = JOptionPane.showInputDialog("Bairro:");

email = JOptionPane.showInputDialog("E-mail");

telefone = JOptionPane.showInputDialog("Telefone");

ra = Long.parseLong(JOptionPane.showInputDialog("RA:"));

rendaFamiliar = Double.parseDouble(JOptionPane.showInputDialog("Renda familiar:"));

idade = Byte.parseByte(JOptionPane.showInputDialog("Idade:"));

JOptionPane.showMessageDialog(null,

"RA: " + ra +

"\nNome: " + nome +

"\nEndereço: " + endereco +

"\nCidade: " + cidade +

"\nBairro: " + bairro +

"\nE-mail: " + email +

"\nTelefone: " + telefone +

"\nRenda familiar: " + rendaFamiliar +

"\nIdade: " + idade

);

}

}

**Criação de cadastro - Modo 2 - Usando a ferramenta Scanner.java**

import java.util.Scanner;

public class Teste {

public static void main(String[] args) {

Scanner sc;

sc = new Scanner(System.in);

String nome, endereco, cidade, bairro, email, telefone;

byte idade;

double rendaFamiliar;

long ra;

System.out.print("Nome:");

nome = sc.nextLine();

System.out.print("Endereco:");

endereco = sc.nextLine();

System.out.print("Cidade:");

cidade = sc.nextLine();

System.out.print("Bairro:");

bairro = sc.nextLine();

System.out.print("E-mail:");

email = sc.nextLine();

System.out.print("Telefone:");

telefone = sc.nextLine();

System.out.print("Idade:");

idade = sc.nextByte();

System.out.print("Renda familiar:");

rendaFamiliar = sc.nextDouble();

System.out.print("RA:");

ra = sc.nextLong();

System.out.print(

"RA: " + ra

+ "\nNome: " + nome

+ "\nEndereço: " + endereco

+ "\nCidade: " + cidade

+ "\nBairro: " + bairro

+ "\nE-mail: " + email

+ "\nTelefone: " + telefone

+ "\nIdade: " + idade

+ "\nRenda Familiar: " + rendaFamiliar

);

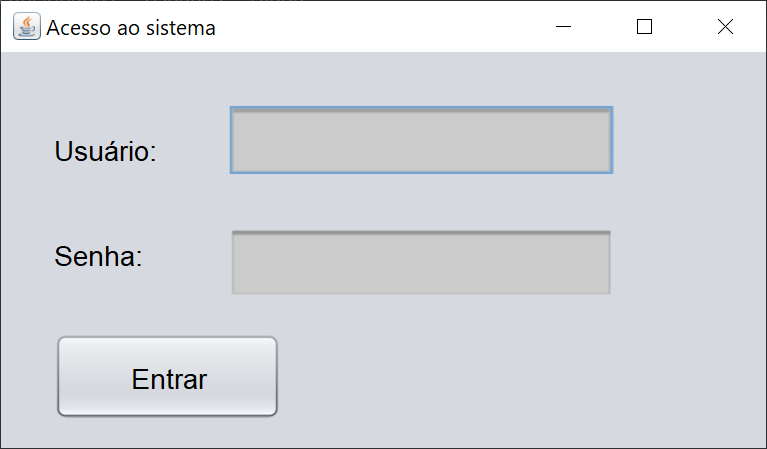
}

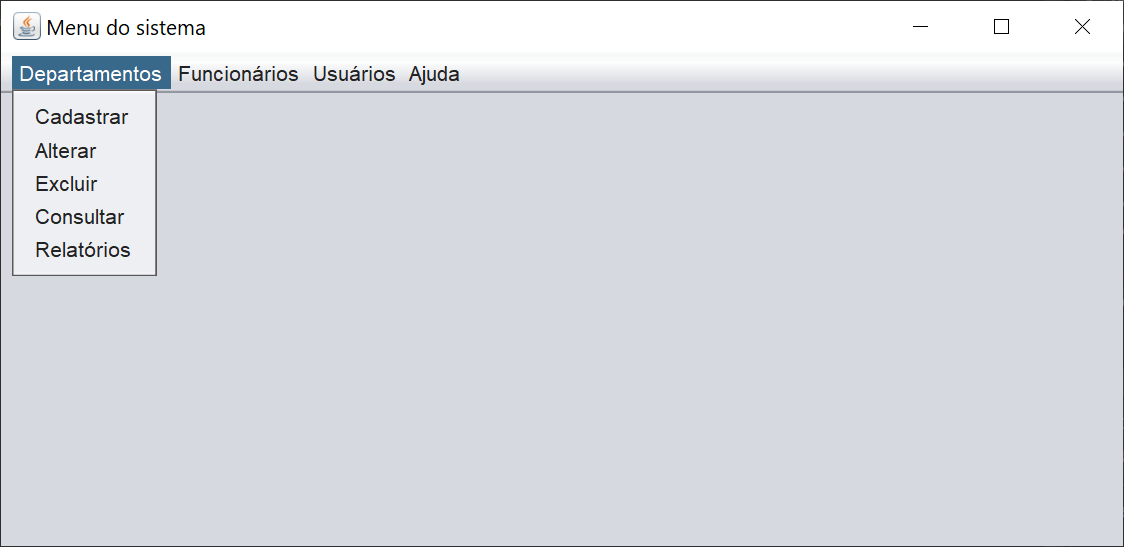
}

**Aula de hoje (21/03/2023):**

* Criação de telas - modo low code (formulários);
* Programação de botões para abrir outras classes de tela.

**Telas geradas hoje:**



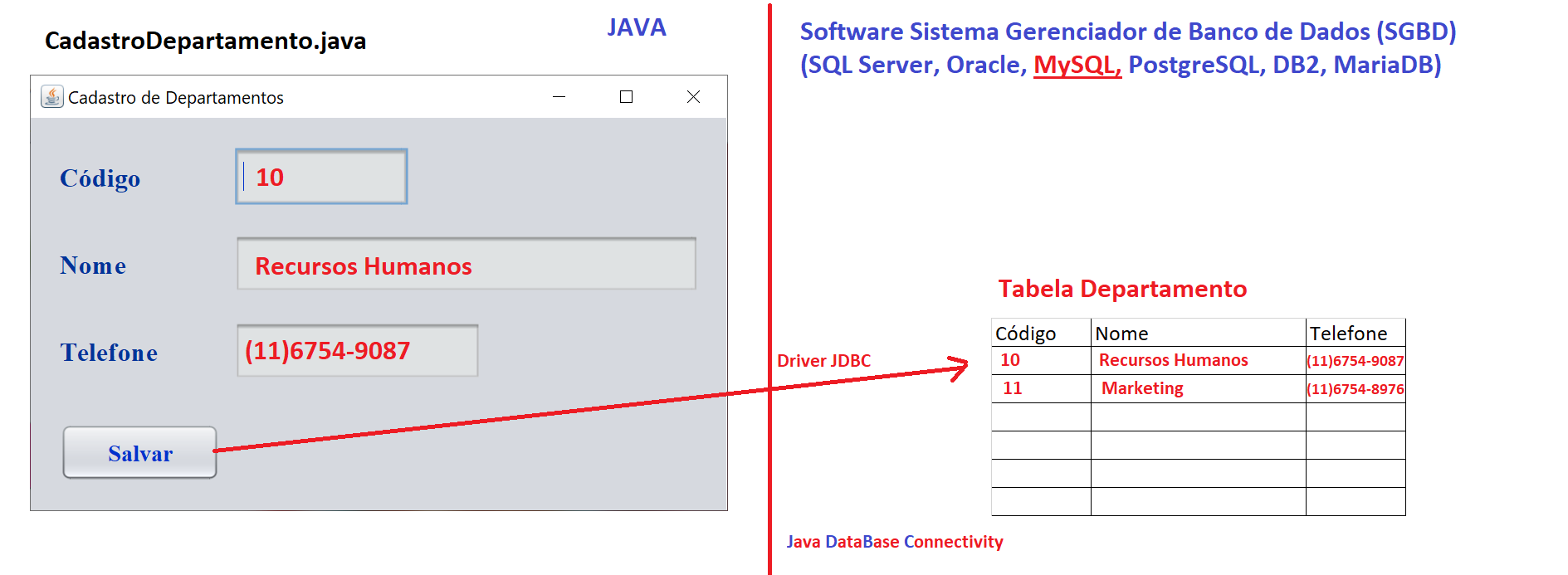


**Link para baixar o projeto do Netbeans criado na aula de hoje:**

<https://drive.google.com/file/d/1xav46upbZLNBDU29lpNk3BpSwP8s7y6A/view?usp=sharing>

**Aula de hoje (27/03/2023):**

* Baixar e instalar o Servidor de Banco de Dados MySQL.
* Criar um banco de dados e tabelas.
* Baixar o driver JDBC e adicionar à library do projeto.
* Ajustar as telas criadas.



Baixar o MySQL Server

<https://dev.mysql.com/downloads/mysql/>

Baixar o Driver JDBC para MySQL

<https://dev.mysql.com/downloads/connector/j/>

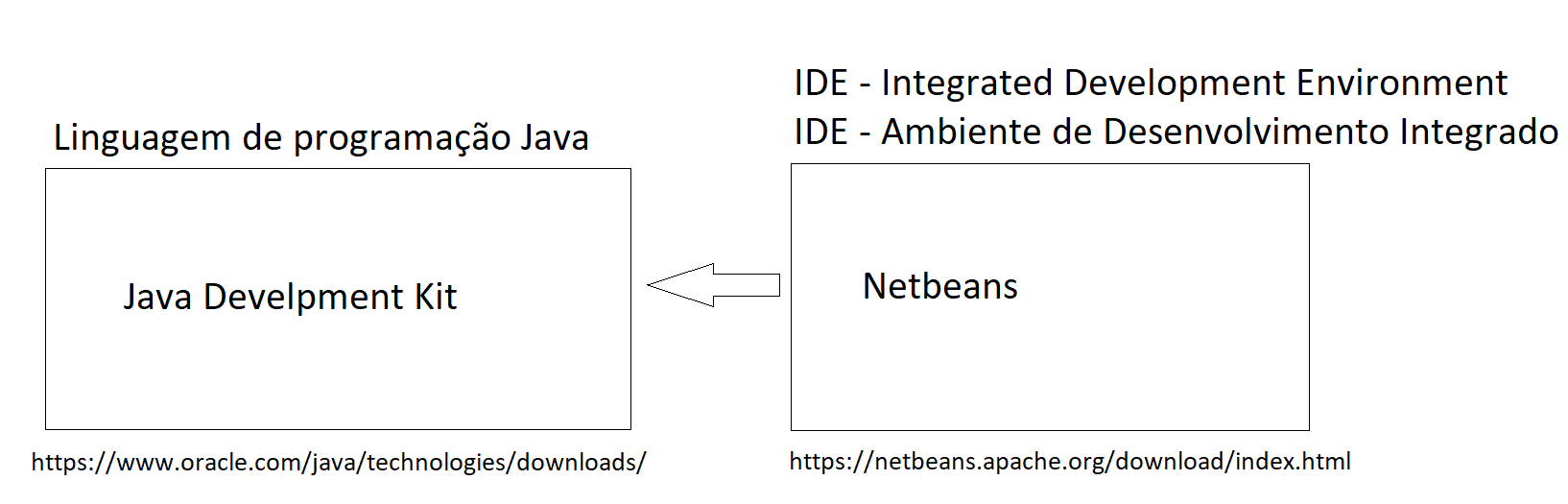
Vídeo Tutorial que mostra como instalar o MySQL

<https://youtu.be/jsX64sXFzxE>

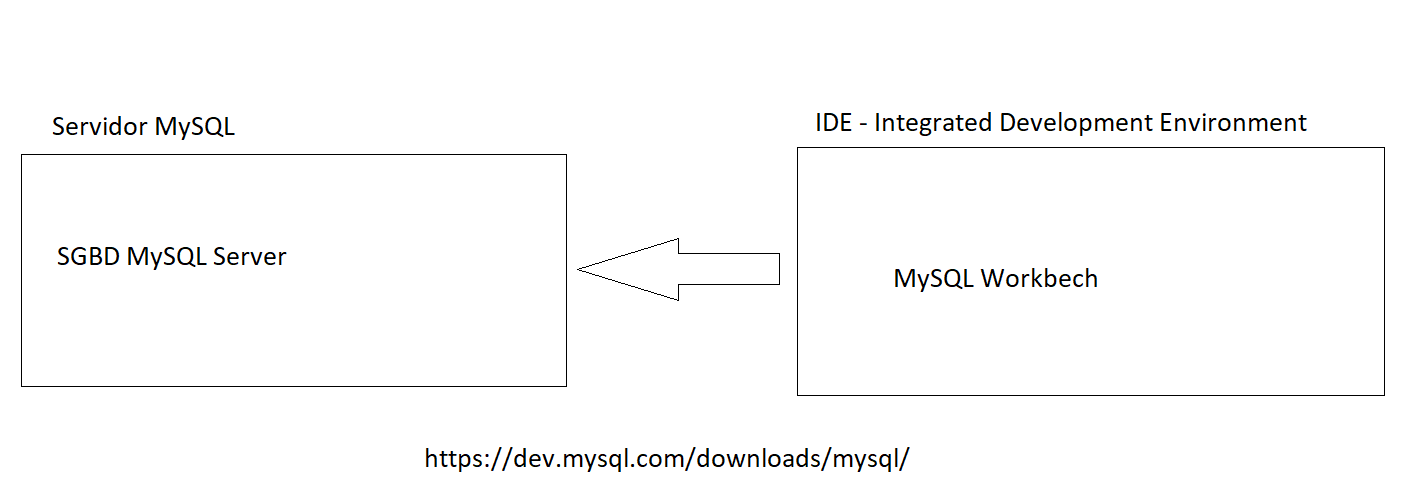
**Aula de hoje (28/03/2023):**

* Ajustes de instalação de ambiente.
* Criação de banco de dados e tabelas no MySQl Workbench.

**Ambiente para programar em Java**



**Ambiente para desenvolver e manipular banco de dados**

****

**Criamos o banco de dados bd\_empresa e as tabelas departamento e usuario, abaixo:**

****

**Criação do banco de dados e das tabelas no modo Raiz (samurai):**

create schema bd\_empresa;

use bd\_empresa;

create table usuario (

usuario varchar(40) not null,

senha varchar(40) null,

cargo varchar (50) null,

primary key (usuario)

);

create table departamento (

codigo int not null,

nome varchar(50) null,

telefone varchar(15) null,

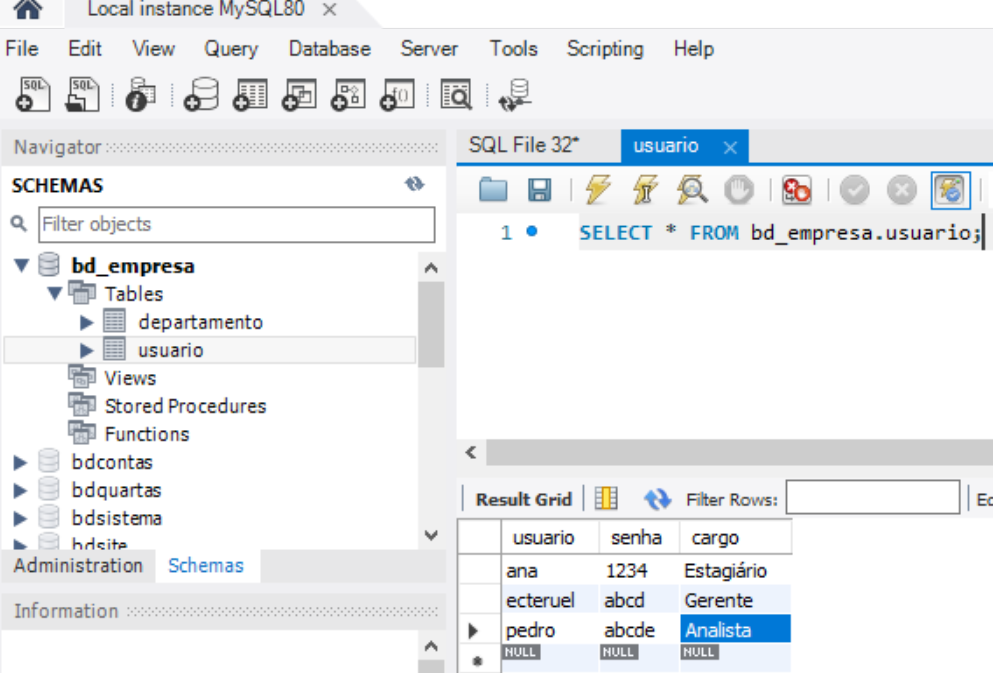
primary key (codigo)

);

insert into departamento values(10, 'Recuros Humanos','(11)675407654');

select \* from departamento;

**Cadastro de alguns usuários.**

****

Material da disciplina no Google Drive:

<https://drive.google.com/drive/folders/1WcaRE6UhIFX0C4pJY2CmxDxVRNs-UTATh3nixRPLzJ861djx4cxyPI_wfNNH_tP-pNQaBTPU?usp=sharing>

**O que você precisa ter feito até aqui para não se enrolar na próxima aula?**

1. Criado o projeto **sistema21-03** no netbeans e as telas **TelaLogin.java**, **TelaMenu.java** e **CadastroDepartamento.java**;
2. Adicionado o driver JDBC (**mysql-connector-j-8.0.32**) na pasta Library (biblioteca) do projeto **sistema21-03** do Netbeans;
3. Criado o banco de dados **bd\_empresa** e as tabelas **usuario** e **departamento**.
4. Cadastrado pelo menos um usuário na tabela **usuario** do banco de dados.

**Aula de hoje (03/04/2023):**

* Revisão de Banco de Dados - Linguagem SQL;
* Programação do botão Salvar do formulário de controle de departamentos.

O projeto **sistema21-03** está disponível para download no link abaixo.

<https://drive.google.com/file/d/1WzG67x40waZs-aG5z7at8TvPbY6r2URz/view?usp=sharing>

Material da disciplina no Google Drive:

<https://drive.google.com/drive/folders/1WcaRE6UhIFX0C4pJY2CmxDxVRNs-UTATh3nixRPLzJ861djx4cxyPI_wfNNH_tP-pNQaBTPU?usp=sharing>

**CRUD - Com Java**

Create (INSERT) - BOTÃO SALVAR

Read (SELECT) - BOTÃO CONSULTAR

Update (UPDATE) - BOTÃO ALTERAR

Delete (DELETE) - BOTÃO EXCLUIR

Como apontar o driver para cada Sistema Gerenciador de Banco de Dados

Oracle

Class.forName("oracle.jdbc.driver.OracleDriver");

Class.forName("oracle.driver.OracleDriver");

SQL Server

Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");

Class.forName("com.microsoft.sqlserver.jdbc.DriverJDBCVersion");

MySQL

Class.forName("com.mysql.cj.jdbc.Driver");

mysql-connector-j-8.0.32.jar

com

mysql

cj

jdbc

Driver.class

Para tratar a possibilidade da classe Driver.class não existir na library, utiliza-se a estrutura Try/catch abaixo.

try {

Class.forName("com.mysql.cj.jdbc.Driver");

} catch (ClassNotFoundException x){

JOptionPane.showMessageDialog(null,"Driver JDBC não encontrado");

}

**Linhas para fazer a conexão com nosso banco de dados**

try {

Class.forName("com.mysql.cj.jdbc.Driver");

Connection conectado = DriverManager.getConnection("jdbc:mysql://localhost:3306/bd\_empresa", "root", "teruel");

} catch (ClassNotFoundException x) {

JOptionPane.showMessageDialog(null, "Driver JDBC não encontrado " + x.getMessage());

} catch (SQLException x) {

JOptionPane.showMessageDialog(null,"Erro na conexão com o banco de dados " + x.getMessage());

}

**Aula de hoje (10/04/2023):**

* Professor afastado (atestado médico)

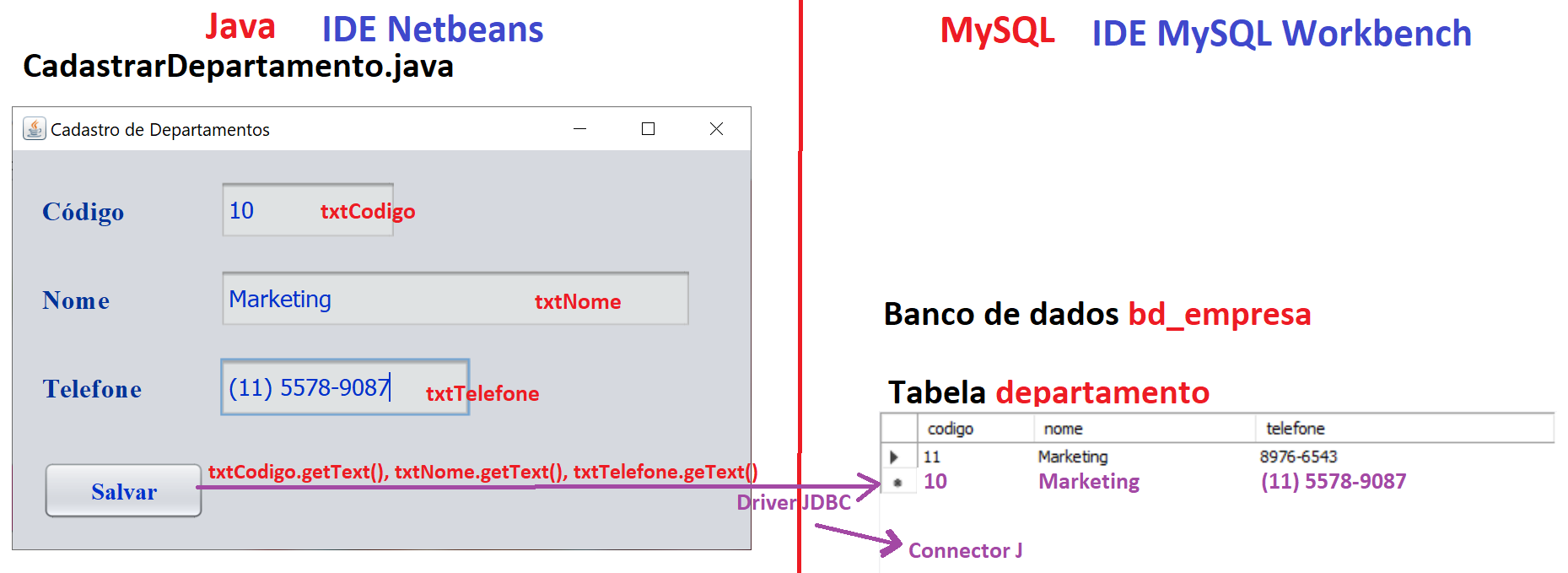
**Aula de hoje (17/04/2023):**

* Programação do botão **Salvar** do formulário de controle de departamentos.

O projeto **sistema21-03** está disponível para download no link abaixo.

<https://drive.google.com/file/d/1Dv7nqo2fdtwt1wWVj41_yOQftIg48Pfq/view?usp=sharing>

Ilustração do processo de salvar dados do formulário Java para a tabela do Banco de dados.



Toda vez que for executar operação em Banco de dados precisará destas linhas:

try {

Class.forName("com.mysql.cj.jdbc.Driver");

Connection conectado = DriverManager.getConnection("jdbc:mysql://localhost:3306/bd\_empresa", "root", "teruel");

//Aqui outros comandos de banco de dados

} catch (ClassNotFoundException x) {

JOptionPane.showMessageDialog(null, "Driver JDBC não encontrado " + x.getMessage());

} catch (SQLException x) {

JOptionPane.showMessageDialog(null,"Erro na conexão com o banco de dados " + x.getMessage());

}

E tem que importar as classes abaixo:

**import javax.swing.JOptionPane;**

//Para guardar a conexão com o Banco de dados

**import java.sql.Connection;**

//Para indicar o banco de dados ao qual vamos nos conectar

**import java.sql.DriverManager;**

//Para tratar qualquer exceção de comando SQL digitado errado

**import java.sql.SQLException;**

//Para executar os comandos SQL INSERT, UPDATE, DELETE, SELECT

**import java.sql.PreparedStatement;**

Passos para **salvar dados** na tabela do banco de dados

1. Indicar o Driver JDBC que está na biblioteca (library) do projeto;

Class.forName("com.mysql.cj.jdbc.Driver");

1. Conectar ao Banco de dados;

Connection conectado = DriverManager.getConnection("jdbc:mysql://localhost:3306/bd\_empresa", "root", "teruel");

1. Preparar o comando INSERT para inserir os dados na tabela banco de dados.

PreparedStatement st = conectado.prepareStatement("INSERT INTO departamento VALUES(?,?,?)");

st.setInt(1,Integer.parseInt(txtCodigo.getText()));

st.setString(2,txtNome.getText());

st.setString(3, txtTelefone.getText());

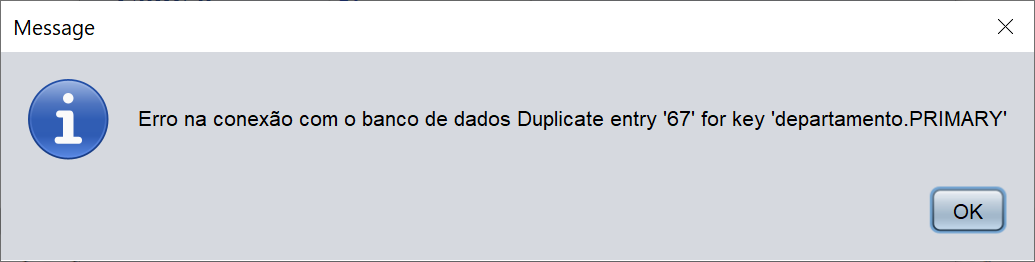
1. Executar o comando INSERT.

st.executeUpdate();

1. Exibir a mensagem indicando que a operação foi realizada.

JOptionPane.showMessageDialog(null,"Departamento cadastrado com sucesso");

**Se tentar cadastrar um código de departamento já cadastrado, aparecerá a mensagem abaixo.**



**Para que seja mostrada uma mensagem mais descritiva do problema, dentro do catch (SQLException x) coloque as linhas abaixo.**

if (ex.getMessage().contains("Duplicate entry")){

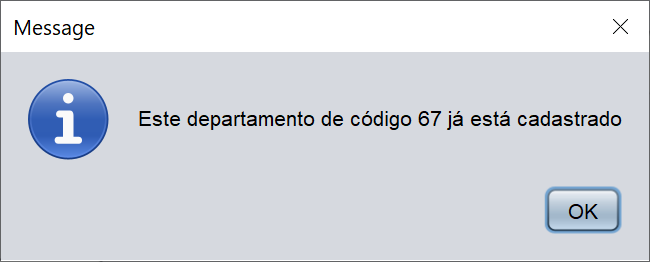
JOptionPane.showMessageDialog(null, "Este departamento de código " + txtCodigo.getText() + " já está cadastrado");

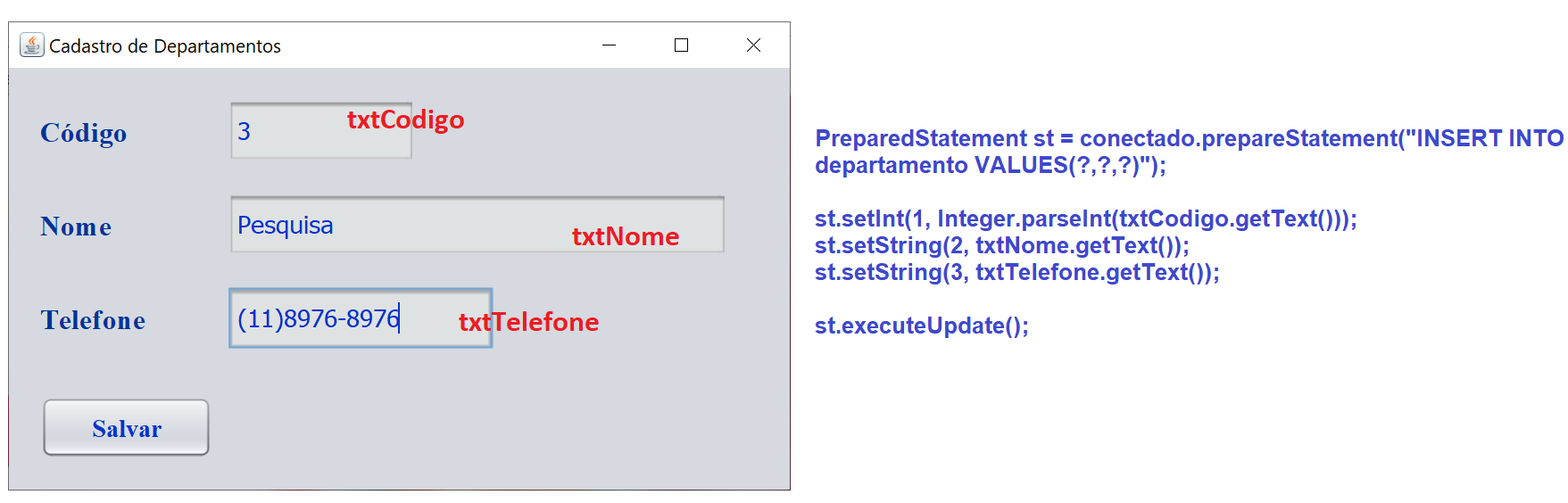
} else {

JOptionPane.showMessageDialog(null,"Erro na conexão com o banco de dados " + x.getMessage());

}

**Agora a mensagem gerada será a mostrada abaixo.**





**Código completo do botão Salvar do formulário CadastrarDepartamento.java.**

try {

Class.forName("com.mysql.cj.jdbc.Driver");

Connection conectado = DriverManager.getConnection("jdbc:mysql://localhost:3306/bd\_empresa", "root", "**teruel**");

PreparedStatement st = conectado.prepareStatement("INSERT INTO departamento VALUES(?,?,?)");

st.setInt(1, Integer.parseInt(txtCodigo.getText()));

st.setString(2, txtNome.getText());

st.setString(3, txtTelefone.getText());

st.executeUpdate(); //Executa o INSERT

JOptionPane.showMessageDialog(null, "Departamento cadastrado com sucesso");

// limpar os campos

txtCodigo.setText("");

txtNome.setText("");

txtTelefone.setText("");

// posiciona o cursor dentro do campo código

txtCodigo.requestFocus();

} catch (ClassNotFoundException x) {

JOptionPane.showMessageDialog(null, "Driver JDBC não encontrado " + x.getMessage());

} catch (SQLException x) {

if (x.getMessage().contains("Duplicate entry")) {

JOptionPane.showMessageDialog(null, "Este departamento de código " + txtCodigo.getText() + " já está cadastrado");

} else {

JOptionPane.showMessageDialog(null, "Erro na conexão com o banco de dados " + x.getMessage());

}

}

**Aula de hoje (18/04/2023):**

* Botão Salvar Depatamento.

**Código completo do botão Salvar do formulário CadastrarDepartamento.java**

**try {**

//Indica qual é o nome do driver JDBC que foi colocado na pasta library do projeto

**Class.forName("com.mysql.cj.jdbc.Driver");**

// Faz a conexão com o banco de dados bd\_empresa e guarda na variável conectado

**Connection conectado = DriverManager.getConnection("jdbc:mysql://localhost:3306/bd\_empresa", "root", "teruel");**

//Prepara o comando INSERT para inserir os dados na tabela do banco de dados

**PreparedStatement st = conectado.prepareStatement("INSERT INTO departamento VALUES(?,?,?)");**

//Coloca o código digitado no comando INSERT acima

**st.setInt(1, Integer.parseInt(txtCodigo.getText()));**

//Coloca o nome digitado no comando INSERT acima

**st.setString(2, txtNome.getText());**

//Coloca o telefone digitado no comando INSERT acima

**st.setString(3, txtTelefone.getText());**

//Executa o comando INSERT que insere os dados na tabela departamento do banco de dados

**st.executeUpdate();**

//Mostra a mensagem informando que o departamento foi cadastrado

**JOptionPane.showMessageDialog(null, "Departamento cadastrado com sucesso");**

//limpa os campos código, nome e telefone

**txtCodigo.setText("");**

**txtNome.setText("");**

**txtTelefone.setText("");**

//posiciona o cursor dentro do campo código

**txtCodigo.requestFocus();**

**} catch (ClassNotFoundException x) {**

//Se o driver JDBC não estiver na library, mostra a mensagem abaixo

**JOptionPane.showMessageDialog(null, "Driver JDBC não encontrado " + x.getMessage());**

**} catch (SQLException x) {**

//Se alguma informação do banco de dados estiver incorreta nos comandos acima

**if (x.getMessage().contains("Duplicate entry")) {**

//Se vocês estiver tentando cadastrar um código que já foi cadastrado

**JOptionPane.showMessageDialog(null, "Este departamento de código " + txtCodigo.getText() + " já está cadastrado");**

**} else {**

//Se ocorrer outro tipo de erro nas informações fornecidas relativas ao banco de dados

**JOptionPane.showMessageDialog(null, "Erro na conexão com o banco de dados " + x.getMessage());**

**}**

**} catch (NumberFormatException x) {**

//Se o código informado não for um número inteiro

**JOptionPane.showMessageDialog(null, "O código precisa ser um número inteiro");**

**txtCodigo.requestFocus(); // posiciona o cursor no campo código**

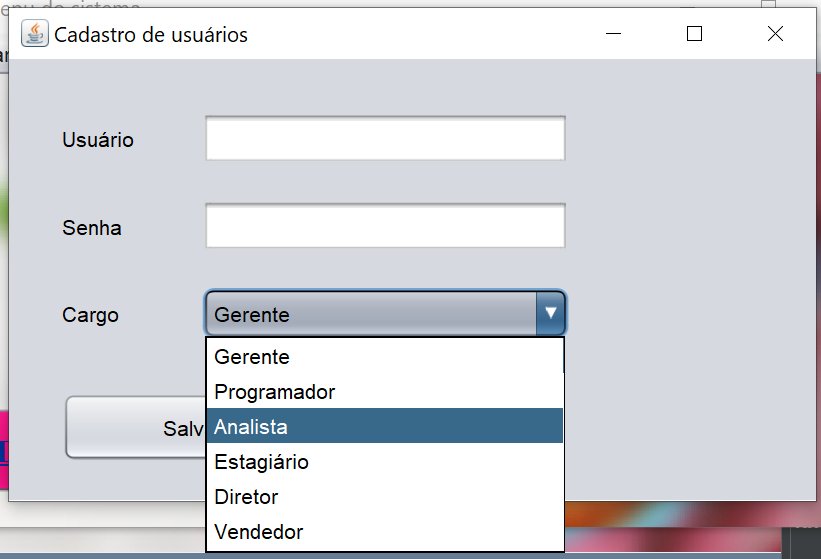
**}**

**Vídeo–aulas de apoio:**

<https://youtu.be/0pwEVDSCxEc>

<https://youtu.be/OQbWbevcCj4>

**CadastrarUsuario.java**



**Programa do botão Salvar do formulário CadastrarUsuario.java.**

**try {**

//Indica qual é o nome do driver JDBC que foi colocado na pasta library do projeto

**Class.forName("com.mysql.cj.jdbc.Driver");**

// Faz a conexão com o banco de dados bd\_empresa e guarda na variável conectado

**Connection conectado = DriverManager.getConnection("jdbc:mysql://localhost:3306/bd\_empresa", "root", "teruel");**

//Prepara o comando INSERT para inserir os dados na tabela do banco de dados

**PreparedStatement st = conectado.prepareStatement("INSERT INTO usuario VALUES(?,?,?)");**

//Coloca o usuário digitado no comando INSERT acima

**st.setString(1, txtUsuario.getText());**

//Coloca a senha digitado no comando INSERT acima

**st.setString(2, txtSenha.getText());**

//Coloca o telefone digitado no comando INSERT acima

**st.setString(3, cmbCargo.getSelectedItem().toString());**

//Executa o comando INSERT que insere os dados na tabela departamento do banco de dados

**st.executeUpdate();**

//Mostra a mensagem informando que o departamento foi cadastrado

**JOptionPane.showMessageDialog(null, "Usuário cadastrado com sucesso");**

//limpa os campos usuario, senha e cargo

**txtUsuario.setText("");**

**txtSenha.setText("");**

//Coloca a 1a. opção do combo visivel

**cmbCargo.setSelectedIndex(0);**

//posiciona o cursor dentro do campo usuário

**txtUsuario.requestFocus();**

**} catch (ClassNotFoundException x) {**

//Se o driver JDBC não estiver na library, mostra a mensagem abaixo

**JOptionPane.showMessageDialog(null, "Driver JDBC não encontrado " + x.getMessage());**

**} catch (SQLException x) {**

//Se alguma informação do banco de dados estiver incorreta nos comandos acima

**if (x.getMessage().contains("Duplicate entry")) {**

//Se vocês estiver tentando cadastrar um código que já foi cadastrado

**JOptionPane.showMessageDialog(null, "Este usuário " + txtUsuario.getText() + " já está cadastrado");**

**} else {**

//Se ocorrer outro tipo de erro nas informações fornecidas relativas ao banco de dados

**JOptionPane.showMessageDialog(null, "Erro na conexão com o banco de dados " + x.getMessage());**

**}**

**}**

**CRUD - Com Java**

Create (INSERT) - BOTÃO SALVAR

Read (SELECT) - BOTÃO CONSULTAR ou ENTRAR

Update (UPDATE) - BOTÃO ALTERAR

Delete (DELETE) - BOTÃO EXCLUIR

**Link do projeto sistema21-03 atualizado:**

<https://drive.google.com/file/d/1g5f5XX8wc2e36GOvO09HixWgbOtFp8K0/view?usp=sharing>

**Exercício**

Fazer o módulo de cadastro de funcionários no sistema que estamos desenvolvendo com Netbeans. Você deve cadastrar na tabela **funcionario** do banco de dados **bd\_empresa** o que for digitado no formulário **CadastrarFuncionario.java** da aplicação Java.

A tabela **funcionario** deve conter os seguintes campos:

matricula int primary key

nome varchar(50)

cargo varchar(40)

salario double

No formulário Java o cargo deve ser um ComboBox onde o cargo será apenas selecionado e não digitado.

**Aula de hoje (24/04/2023):**

* Validação de Login ao entrar no sistema.

**Link do projeto sistema21-03 atualizado:**

<https://drive.google.com/file/d/1g5f5XX8wc2e36GOvO09HixWgbOtFp8K0/view?usp=sharing>

**Toda vez que for executar operação em Banco de dados precisará destas linhas:**

try {

Class.forName("com.mysql.cj.jdbc.Driver");

Connection conectado = DriverManager.getConnection("jdbc:mysql://localhost:3306/bd\_empresa", "root", "**teruel**");

//Aqui outros comandos de banco de dados

} catch (ClassNotFoundException x) {

JOptionPane.showMessageDialog(null, "Driver JDBC não encontrado " + x.getMessage());

} catch (SQLException x) {

JOptionPane.showMessageDialog(null,"Erro na conexão com o banco de dados " + x.getMessage());

}

E importar as classes abaixo:

import javax.swing.JOptionPane;

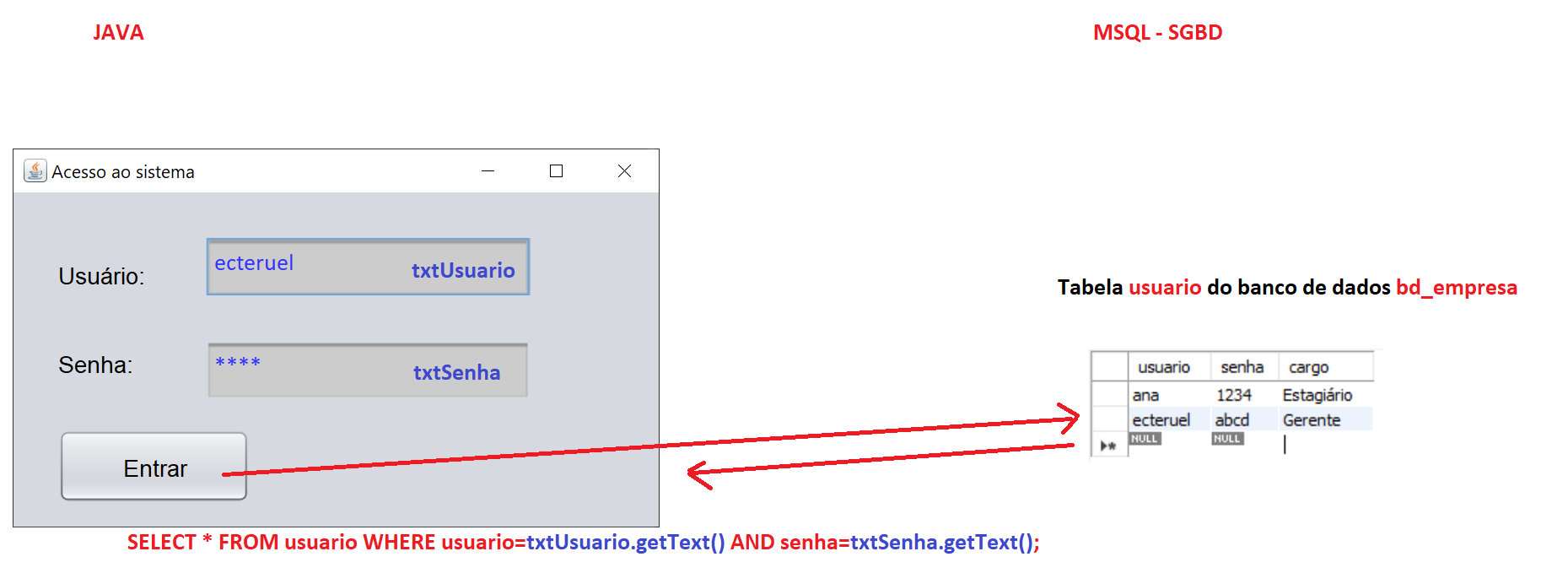
import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

import java.sql.PreparedStatement;

import java.sql.ResultSet;





Programa do botão Entrar que só permite acesso de usuários cadastrados na tabela usuario do banco de dados

try {

Class.forName("com.mysql.cj.jdbc.Driver");

Connection conectado = DriverManager.getConnection("jdbc:mysql://localhost:3306/bd\_empresa", "root", "teruel");

PreparedStatement st = conectado.prepareStatement("SELECT \* FROM usuario WHERE usuario = ? AND senha = ?");

st.setString(1, txtUsuario.getText());

st.setString(2, txtSenha.getText());

ResultSet resultado = st.executeQuery();

if (resultado.next()){

dispose();

new TelaMenu().setVisible(true);

} else {

JOptionPane.showMessageDialog(null,"Usuário e/ou senha inválido");

}

} catch (ClassNotFoundException x) {

JOptionPane.showMessageDialog(null, "Driver JDBC não encontrado " + x.getMessage());

} catch (SQLException x) {

JOptionPane.showMessageDialog(null, "Erro na conexão com o banco de dados " + x.getMessage());

}

**CRUD - Com Java**

Create (INSERT) - BOTÃO SALVAR

Read (SELECT) - BOTÃO CONSULTAR ou ENTRAR

Update (UPDATE) - BOTÃO ALTERAR

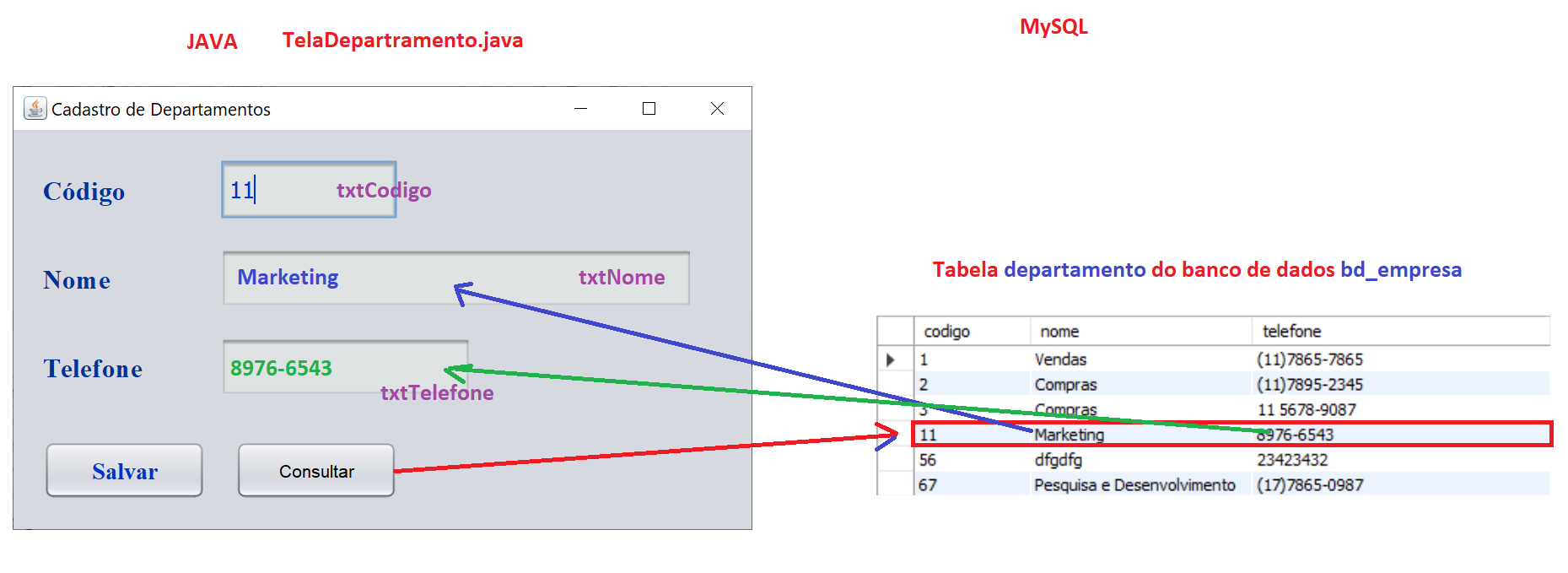
Delete (DELETE) - BOTÃO EXCLUIR

**Link do projeto sistema21-03 atualizado:**

<https://drive.google.com/file/d/1z8zni8g9aGf6ZD7gEg0xMONmYBzqTFh6/view?usp=sharing>

**Aula de hoje (25/04/2023):**

* Consultar Departamentos.
* Excluir Departamentos.



**Código do botão Consultar departamento**

try {

//Indica o nome da classe Driver.class colocada na pasta library do projeto

Class.forName("com.mysql.cj.jdbc.Driver");

//Faz a conexão com o banco de dados bd\_empresa e guarda na variável conectado

Connection conectado = DriverManager.getConnection("jdbc:mysql://localhost:3306/bd\_empresa", "root", "teruel");

//Cria o comando SQL SELECT incompleto (?) que será executado e armazena esse comando na variável st

PreparedStatement st = conectado.prepareStatement("SELECT \* FROM departamento WHERE codigo = ?");

//Completa o comando SELECT criado, colocando o txtCodigo no lugar d ponto de interrogação

st.setInt(1, Integer.parseInt(txtCodigo.getText()));

//Executa o comando SQL SELECT e guarda o retorno do banco de dados na variável resultado

ResultSet resultado = st.executeQuery();

//Verifica se encontrou o departamento na tabela do banco de dados

if (resultado.next()) {

//Se o depatamento foi encontrado, carrega o nome do departamento no campo txtNome do formulário

txtNome.setText(resultado.getString("nome"));

//Se o depatamento foi encontrado, carrega o telefone do departamento no campo txtTelefone do formulário

txtTelefone.setText(resultado.getString("telefone"));

//Torna o botão excluir visível

btnExcluir.setVisible(true);

} else {

//Se o departamento não foi encontrado na tabela, exibe uma mensagem indicando isto.

JOptionPane.showMessageDialog(null, "Código de departamento não encontrado");

//Coloca o cursor no campo txtCodigo

txtCodigo.requestFocus();

}

} catch (ClassNotFoundException x) {

//Se o a classe Driver.class não foi encontrada na pasta library do projeto, exibe a mensagem indicando isto.

JOptionPane.showMessageDialog(null, "Driver JDBC não encontrado " + x.getMessage());

} catch (SQLException x) {

//Se alguma informação relacionda ao banco de dados foi igita incorretamente, exibe a mensagem indicando isso

JOptionPane.showMessageDialog(null, "Erro na conexão com o banco de dados " + x.getMessage());

} catch (NumberFormatException x) {

//Se foi digitado no campo txtCodigo qualquer coisa que não seja número inteiro, exibe a mensagem abaixo.

JOptionPane.showMessageDialog(null, "Digite um código que seja um número inteiro válido");

txtCodigo.requestFocus();

}

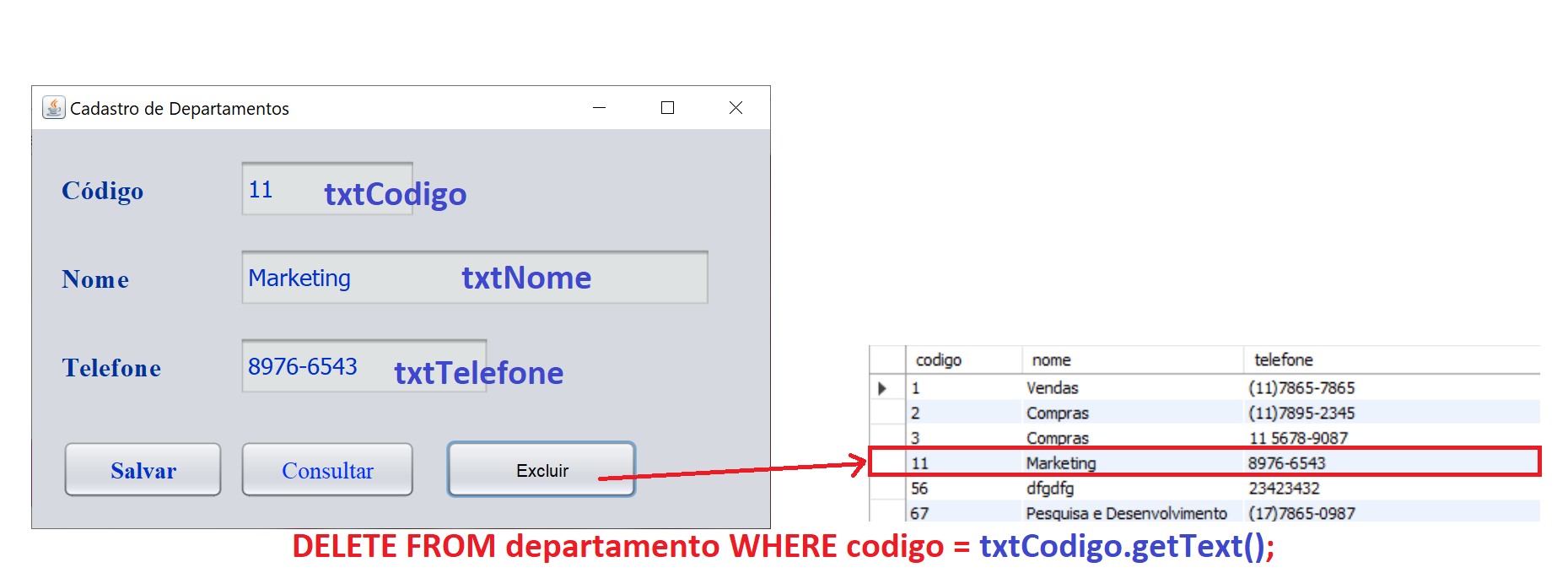
**CRUD - Com Java**

Create (INSERT) - BOTÃO SALVAR

Read (SELECT) - BOTÃO CONSULTAR ou ENTRAR

Update (UPDATE) - BOTÃO ALTERAR

Delete (DELETE) - BOTÃO EXCLUIR



st.executeUpdate();

Executa comandos SQL INSERT, DELETE e UPDATE

st.executeQuery();

Executa um comando SQL SELECT

Código do botão Excluir departamento

try {

Class.forName("com.mysql.cj.jdbc.Driver");

Connection conectado = DriverManager.getConnection("jdbc:mysql://localhost:3306/bd\_empresa", "root", "teruel");

PreparedStatement st = conectado.prepareStatement("DELETE FROM departamento WHERE codigo = ?");

st.setInt(1, Integer.parseInt(txtCodigo.getText()));

st.executeUpdate();

JOptionPane.showMessageDialog(null, "Departamento excluído com sucesso");

btnExcluir.setVisible(false);

btnAlterar.setVisible(false);

txtCodigo.setText("");

txtNome.setText("");

txtTelefone.setText("");

txtCodigo.requestFocus();

} catch (ClassNotFoundException x) {

JOptionPane.showMessageDialog(null, "Driver JDBC não encontrado " + x.getMessage());

} catch (SQLException x) {

JOptionPane.showMessageDialog(null, x.getMessage());

} catch (NumberFormatException x) {

JOptionPane.showMessageDialog(null, "O código precisa ser um número inteiro");

txtCodigo.requestFocus();

}

Na classe TelaDepartamento.java, incluir a linha, em vermelho, abaixo.

public TelaDepartamento() {

initComponents();

//Oculta o botão excluir ao carregar a tela (formulário)

btnExcluir.setVisible(false);

}

**CRUD - Com Java**

Create (INSERT) - BOTÃO SALVAR

Read (SELECT) - BOTÃO CONSULTAR ou ENTRAR

Update (UPDATE) - BOTÃO ALTERAR

Delete (DELETE) - BOTÃO EXCLUIR

**Exercício**

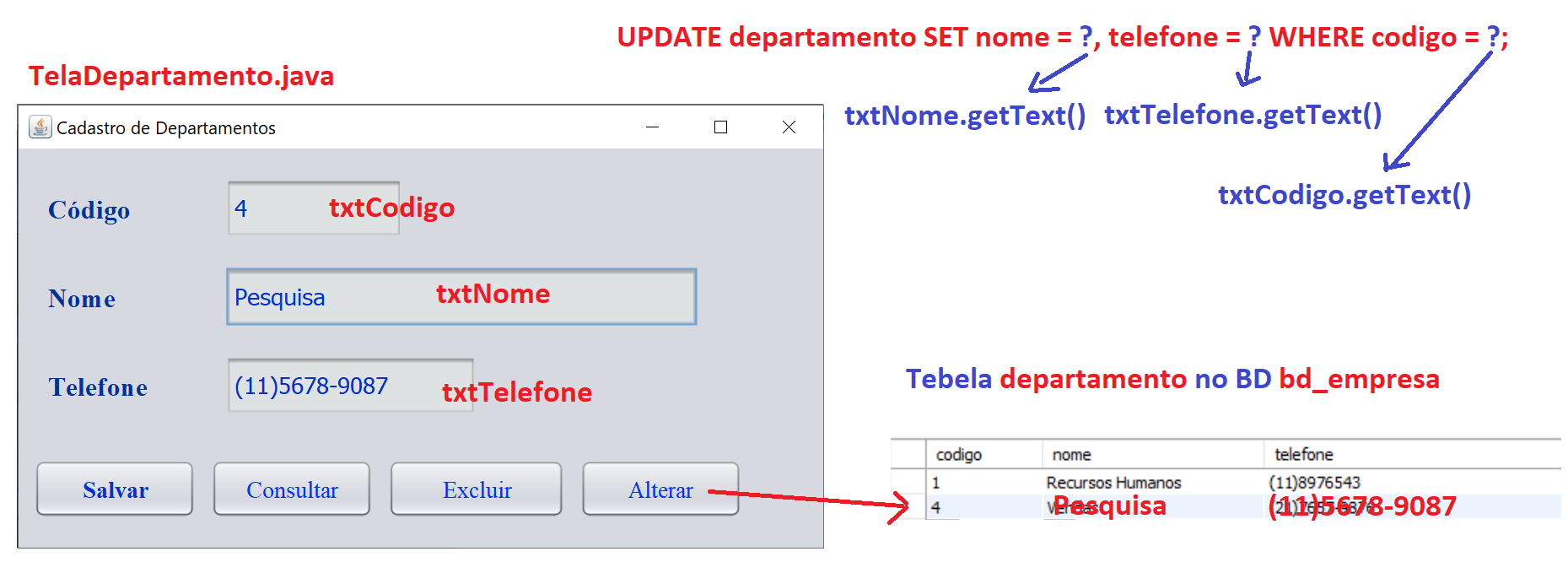
Desenvolver as funcionalidades Excluir e Consultar de Usuários e de Funcionários.

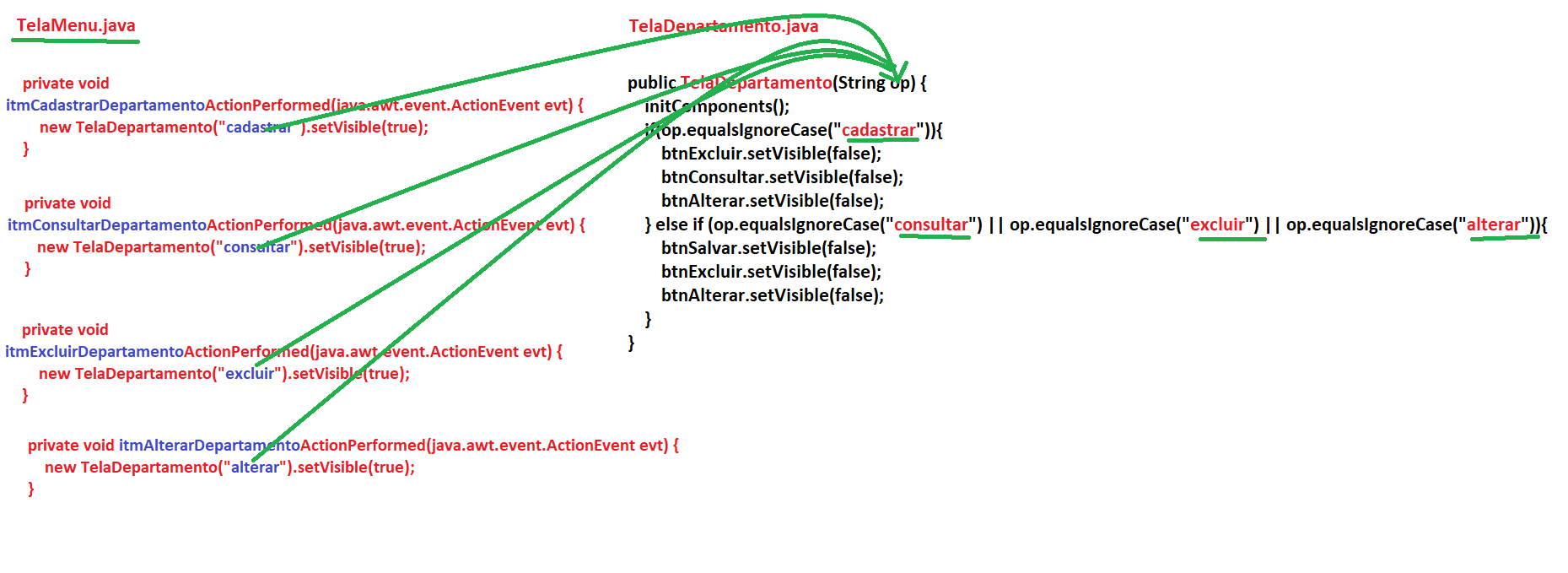
Link do projeto **sistema21-03**, atualizado, criado no netbeans

<https://drive.google.com/file/d/108tbGXFrSTwGTJ1foZMNB7E_R6tk3fDz/view?usp=sharing>

**Aula de hoje (02/05/2023):**

* Alterar Departamentos.





Programa do botão Alterar Departamento

try {

Class.forName("com.mysql.cj.jdbc.Driver");

Connection conectado = DriverManager.getConnection("jdbc:mysql://localhost:3306/bd\_empresa", "root", "teruel");

PreparedStatement st = conectado.prepareStatement("UPDATE departamento SET nome = ?, telefone = ? WHERE codigo = ?");

st.setString(1, txtNome.getText());

st.setString(2, txtTelefone.getText());

st.setInt(3, Integer.parseInt(txtCodigo.getText()));

st.executeUpdate();

JOptionPane.showMessageDialog(null, "Departamento Alterado com sucesso");

btnExcluir.setVisible(false);

btnAlterar.setVisible(false);

txtCodigo.setText("");

txtNome.setText("");

txtTelefone.setText("");

txtCodigo.requestFocus();

} catch (ClassNotFoundException x) {

JOptionPane.showMessageDialog(null, "Driver JDBC não encontrado " + x.getMessage());

} catch (SQLException x) {

JOptionPane.showMessageDialog(null, x.getMessage());

} catch (NumberFormatException x) {

JOptionPane.showMessageDialog(null, "O código precisa ser um número inteiro");

txtCodigo.requestFocus();

}

Programação do construtor da classe Departamento, para identificar a opção clicada.

public TelaDepartamento(String op) {

initComponents();

if(op.equalsIgnoreCase("cadastrar")){

btnExcluir.setVisible(false);

btnConsultar.setVisible(false);

btnAlterar.setVisible(false);

} else if (op.equalsIgnoreCase("consultar") || op.equalsIgnoreCase("excluir") || op.equalsIgnoreCase("alterar")){

btnSalvar.setVisible(false);

btnExcluir.setVisible(false);

btnAlterar.setVisible(false);

}

}

**Exercício**

Fazer o botão Alterar do formulário de usuários e de funcionários para dia 08/05. Vale ponto adicional na Av1.

**Aula de hoje (08/05/2023):**

* Relatórios.

Link do projeto do Netbeans atualizado:

<https://drive.google.com/file/d/1szxA-rRk9emLI5GNresE17CfJMx3qPNY/view?usp=sharing>

Código base para selecionar todos os departamentos cadastrados

try {

Class.forName("com.mysql.cj.jdbc.Driver");

Connection conectado = DriverManager.getConnection("jdbc:mysql://localhost:3306/bd\_empresa", "root", "**teruel**");

PreparedStatement st = conectado.prepareStatement("SELECT \* FROM departamento");

ResultSet resultado = st.executeQuery();

while (resultado.next()) {

//Jogar os valores na tabela do formulário

}

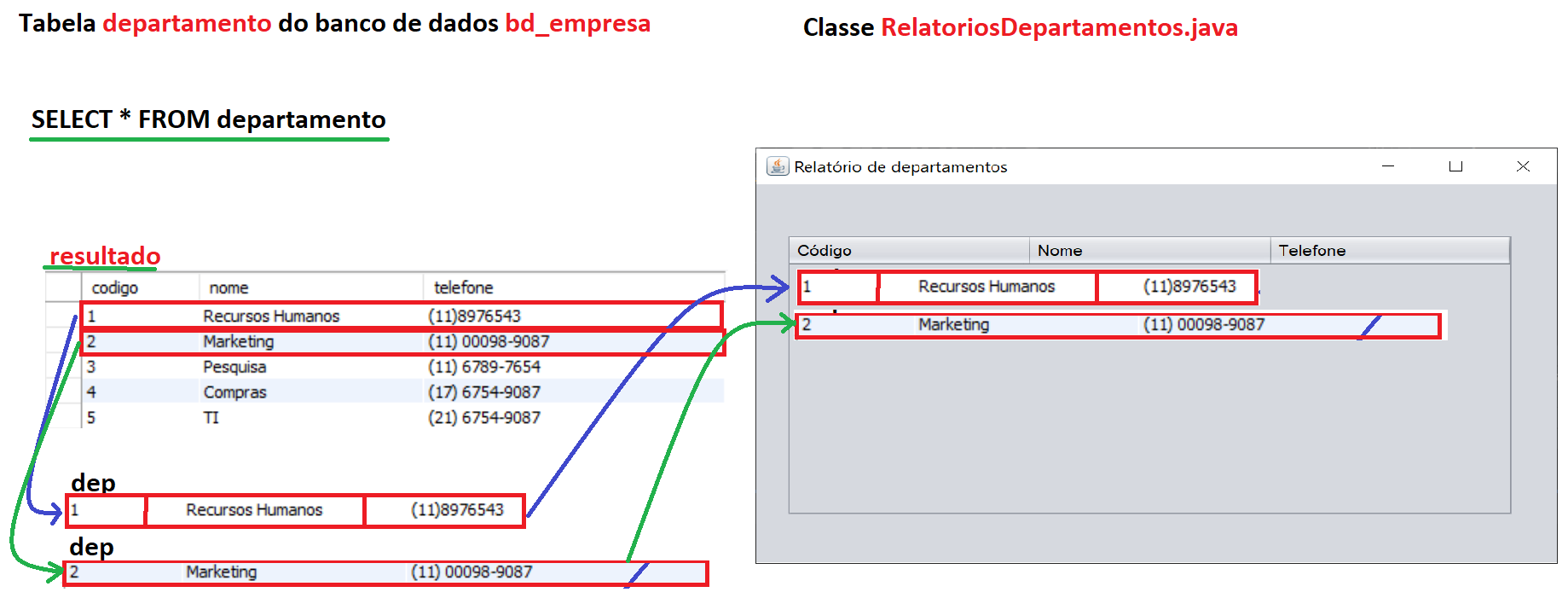
} catch (ClassNotFoundException x) {

JOptionPane.showMessageDialog(null, "Driver JDBC não encontrado " + x.getMessage());

} catch (SQLException x) {

JOptionPane.showMessageDialog(null, "Erro na conexão com o banco de dados " + x.getMessage());

}



Programa do construtor da classe RelatorioDepartamentos.java

//Construtor da classe

public RelatoriosDepartamentos() {

initComponents();

try {

Class.forName("com.mysql.cj.jdbc.Driver");

Connection conectado = DriverManager.getConnection("jdbc:mysql://localhost:3306/bd\_empresa", "root", "teruel");

PreparedStatement st = conectado.prepareStatement("SELECT \* FROM departamento");

ResultSet resultado = st.executeQuery();

DefaultTableModel moldeTabela;

moldeTabela = (DefaultTableModel) tblDepartamentos.getModel();

moldeTabela.setRowCount(0);

while (resultado.next()) {

Object dep[] = {

resultado.getInt("codigo"),

resultado.getString("nome"),

resultado.getString("telefone")

};

moldeTabela.addRow(dep);

}

} catch (ClassNotFoundException x) {

JOptionPane.showMessageDialog(null, "Driver JDBC não encontrado " + x.getMessage());

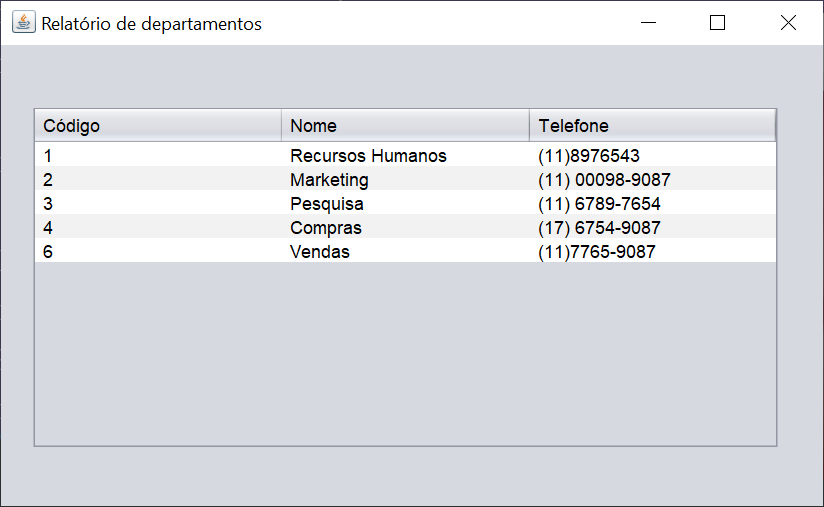
} catch (SQLException x) {

JOptionPane.showMessageDialog(null, "Erro na conexão com o banco de dados " + x.getMessage());

}

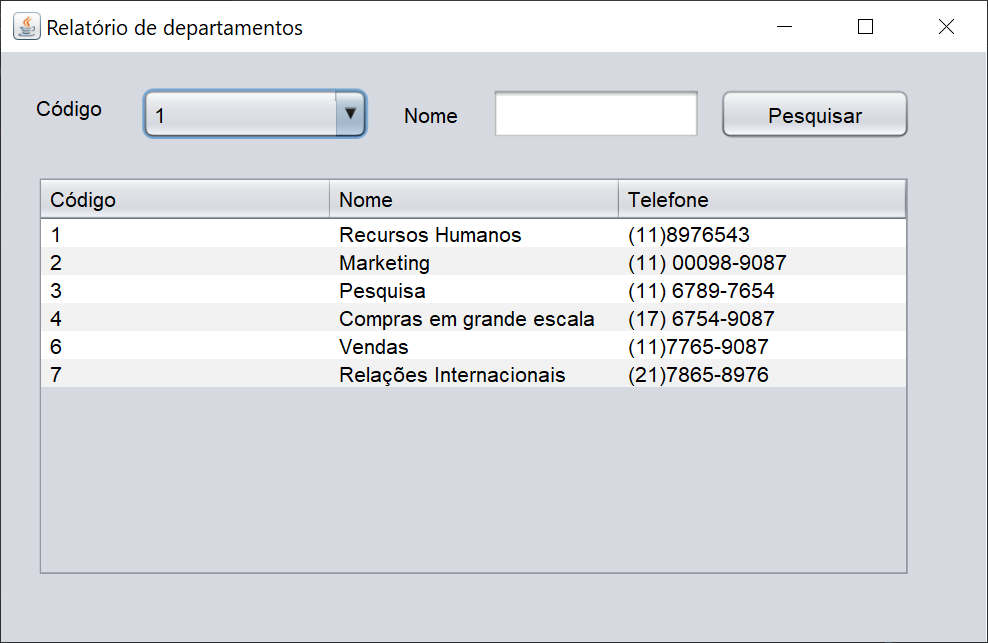
}

Tela gerada:



**Aula de hoje (09/05/2023):**

* Relatórios.
* Segregação de funções.



**Programa do combobox cmbCodigo**

try {

Class.forName("com.mysql.cj.jdbc.Driver");

Connection conectado = DriverManager.getConnection("jdbc:mysql://localhost:3306/bd\_empresa", "root", "**teruel**");

PreparedStatement st = conectado.prepareStatement("SELECT \* FROM departamento WHERE codigo = ?");

st.setString(1, cmbCodigo.getSelectedItem().toString());

ResultSet resultado = st.executeQuery();

DefaultTableModel moldeTabela;

moldeTabela = (DefaultTableModel) tblDepartamentos.getModel();

moldeTabela.setRowCount(0);

while (resultado.next()) {

Object dep[] = {

resultado.getInt("codigo"),

resultado.getString("nome"),

resultado.getString("telefone")

};

moldeTabela.addRow(dep);

}

} catch (ClassNotFoundException x) {

JOptionPane.showMessageDialog(null, "Driver JDBC não encontrado " + x.getMessage());

} catch (SQLException x) {

JOptionPane.showMessageDialog(null, "Erro na conexão com o banco de dados " + x.getMessage());

}

**Programa do botão Pesquisar**

try {

Class.forName("com.mysql.cj.jdbc.Driver");

Connection conectado = DriverManager.getConnection("jdbc:mysql://localhost:3306/bd\_empresa", "root", "**teruel**");

PreparedStatement st = conectado.prepareStatement("SELECT \* FROM departamento WHERE nome LIKE ?");

st.setString(1,"%" + txtNome.getText() + "%" );

ResultSet resultado = st.executeQuery();

DefaultTableModel moldeTabela;

moldeTabela = (DefaultTableModel) tblDepartamentos.getModel();

moldeTabela.setRowCount(0);

while (resultado.next()) {

Object dep[] = {

resultado.getInt("codigo"),

resultado.getString("nome"),

resultado.getString("telefone")

};

moldeTabela.addRow(dep);

}

} catch (ClassNotFoundException x) {

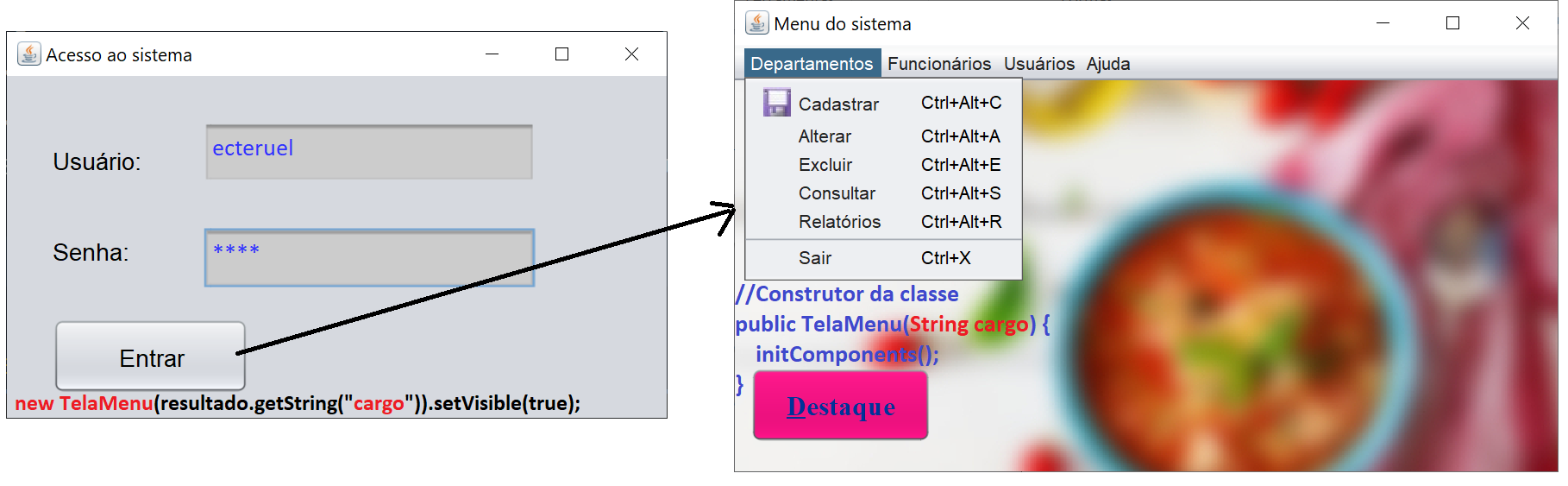
JOptionPane.showMessageDialog(null, "Driver JDBC não encontrado " + x.getMessage());

} catch (SQLException x) {

JOptionPane.showMessageDialog(null, "Erro na conexão com o banco de dados " + x.getMessage());

}

**Segregação de funções - as funcionalidades são disponibilizadas de acordo com o cargo.**

****

**Ajuste no programa do botão Entrar do TelaLogin**

try {

Class.forName("com.mysql.cj.jdbc.Driver");

Connection conectado = DriverManager.getConnection("jdbc:mysql://localhost:3306/bd\_empresa", "root", "teruel");

PreparedStatement st = conectado.prepareStatement("SELECT \* FROM usuario WHERE usuario = ? AND senha = ?");

st.setString(1, txtUsuario.getText());

st.setString(2, txtSenha.getText());

ResultSet resultado = st.executeQuery();

if (resultado.next()){

dispose();//esconde a tela de login

**new TelaMenu(resultado.getString("cargo")).setVisible(true);**

} else {

JOptionPane.showMessageDialog(null,"Usuário e/ou senha inválido");

}

} catch (ClassNotFoundException x) {

JOptionPane.showMessageDialog(null, "Driver JDBC não encontrado " + x.getMessage());

} catch (SQLException x) {

JOptionPane.showMessageDialog(null, "Erro na conexão com o banco de dados " + x.getMessage());

}

**Construtor da classe TelaMenu.java - Só restringindo o gerente**

//Construtor da classe

public TelaMenu(String cargo) {

initComponents();

//Se o cargo recebido NÃO for "Gerente"

if (!cargo.equalsIgnoreCase("gerente")) {

mnuUsuarios.setVisible(false);

itmCadastrarDepartamento.setVisible(false);

itmAlterarDepartamento.setVisible(false);

itmExcluirDepartamento.setVisible(false);

itmConsultarDepartamento.setVisible(false);

}

}

**Construtor da classe TelaMenu.java - Restringindo o gerente, chefe ou administrador**

//Construtor da classe

public TelaMenu(String cargo) {

initComponents();

//Se o cargo recebido NÃO for "Gerente" ou "Chefe" ou "Administrador"

if (!(cargo.equalsIgnoreCase("gerente") || cargo.equalsIgnoreCase("chefe") || cargo.equalsIgnoreCase("Administrador"))) {

mnuUsuarios.setVisible(false);

itmCadastrarDepartamento.setEnabled(false);

itmAlterarDepartamento.setEnabled(false);

itmExcluirDepartamento.setEnabled(false);

itmConsultarDepartamento.setEnabled(false);

}

}

Link para acesso ao projeto do Netbeans (sistema21-03) atualizado:

<https://drive.google.com/file/d/1LzyWl_ON-oTBAD28TqONVXTRCMAwH4N8/view?usp=sharing>

**Exercício**

Fazer o relatório com os filtros para funcionários e usuários para a próxima semana(15/05). Vale ponto adicional na Av1.

**Aula de hoje (15/05/2023):**

* Criação de executável.
* Backup de Banco de Dados.
* Instalação e execução do software no cliente.

Para habilitar o pressionamento da tecla ENTER na tela de login

private void txtUsuarioKeyPressed(java.awt.event.KeyEvent evt) {

**if(evt.getKeyCode() == java.awt.event.KeyEvent.VK\_ENTER){**

**txtSenha.requestFocus();**

**}**

}

private void txtSenhaKeyPressed(java.awt.event.KeyEvent evt) {

**if(evt.getKeyCode() == java.awt.event.KeyEvent.VK\_ENTER){**

**btnEntrar.requestFocus();**

**}**

}

private void btnEntrarKeyPressed(java.awt.event.KeyEvent evt) {

**if(evt.getKeyCode() == java.awt.event.KeyEvent.VK\_ENTER){**

**btnEntrar.doClick();**

**}**

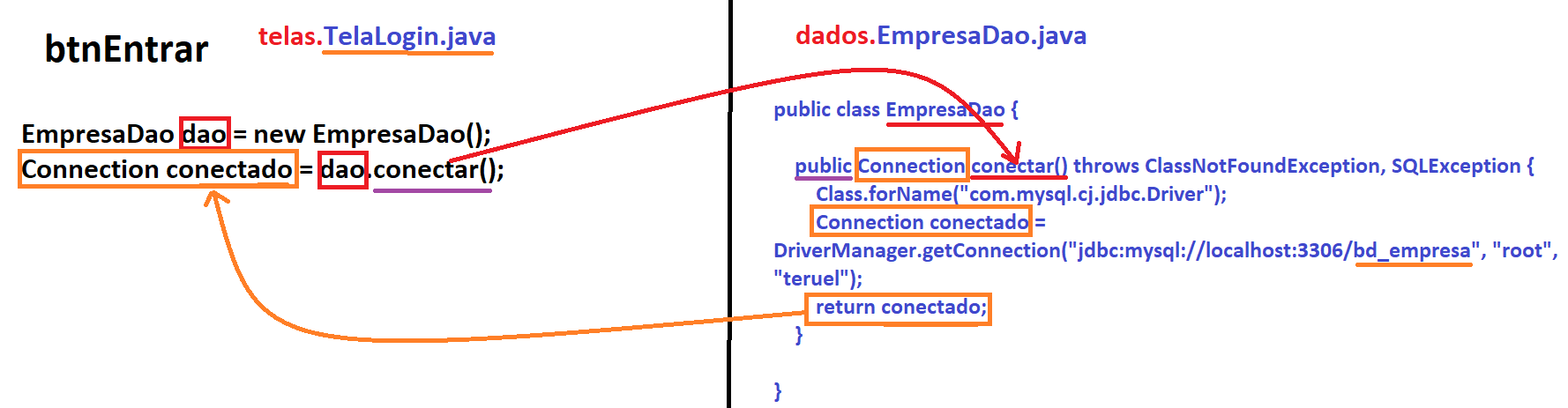
}

Link para baixar o projeto atualizado:

<https://drive.google.com/file/d/1IyUHC-hpiWe7FLB0KQcY-AAgOLSB806y/view?usp=sharing>

**Aula de hoje (16/05/2023):**

* Métodos



**EmpresaDao.java**

package dados;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import java.sql.ResultSet;

public class EmpresaDao {

static Connection conectado;

public static Connection conectar() throws ClassNotFoundException, SQLException {

Class.forName("com.mysql.cj.jdbc.Driver");

conectado = DriverManager.getConnection("jdbc:mysql://localhost:3306/bd\_empresa", "root", "teruel");

return conectado;

}

public static void conectarInterno() throws ClassNotFoundException, SQLException {

Class.forName("com.mysql.cj.jdbc.Driver");

conectado = DriverManager.getConnection("jdbc:mysql://localhost:3306/bd\_empresa", "root", "teruel");

}

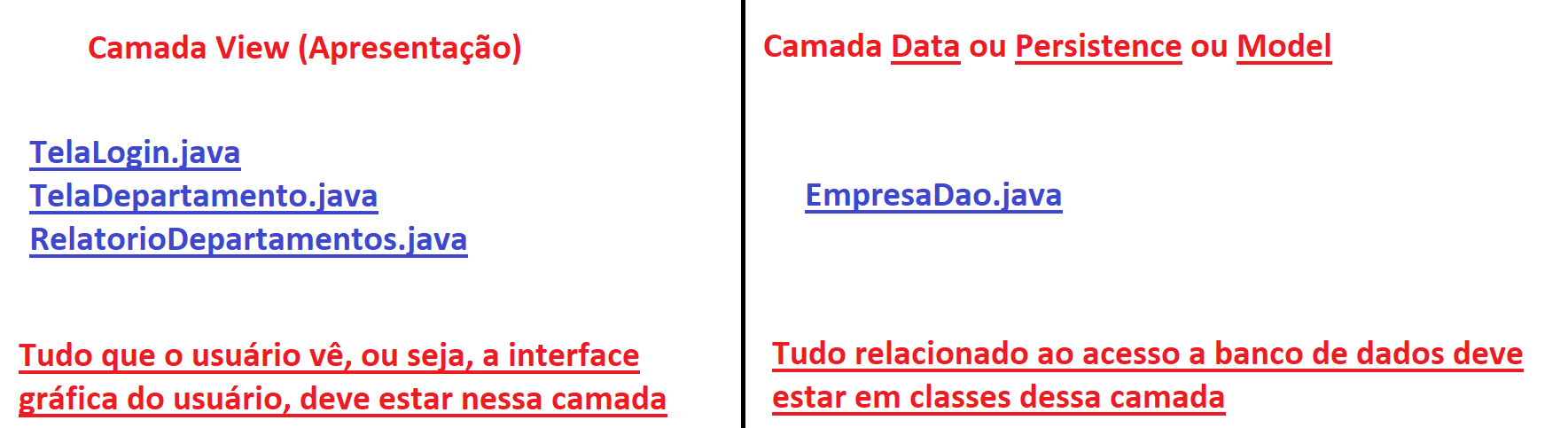
}

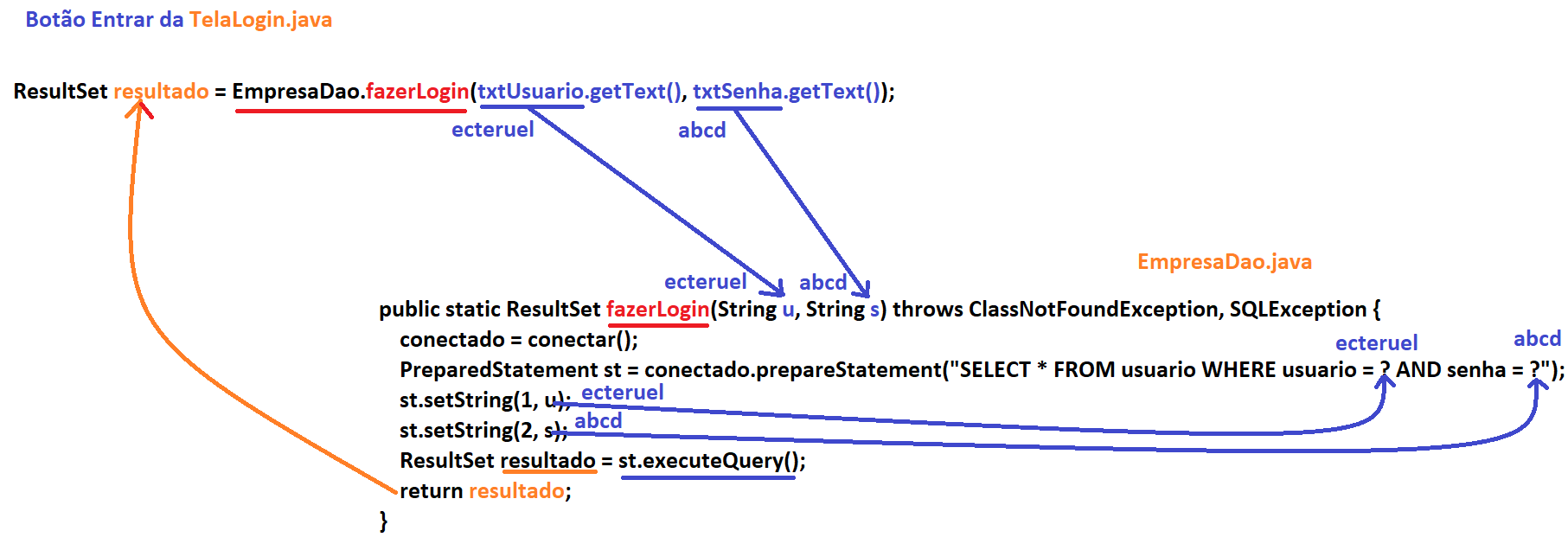
Link para baixar o projeto atualizado:

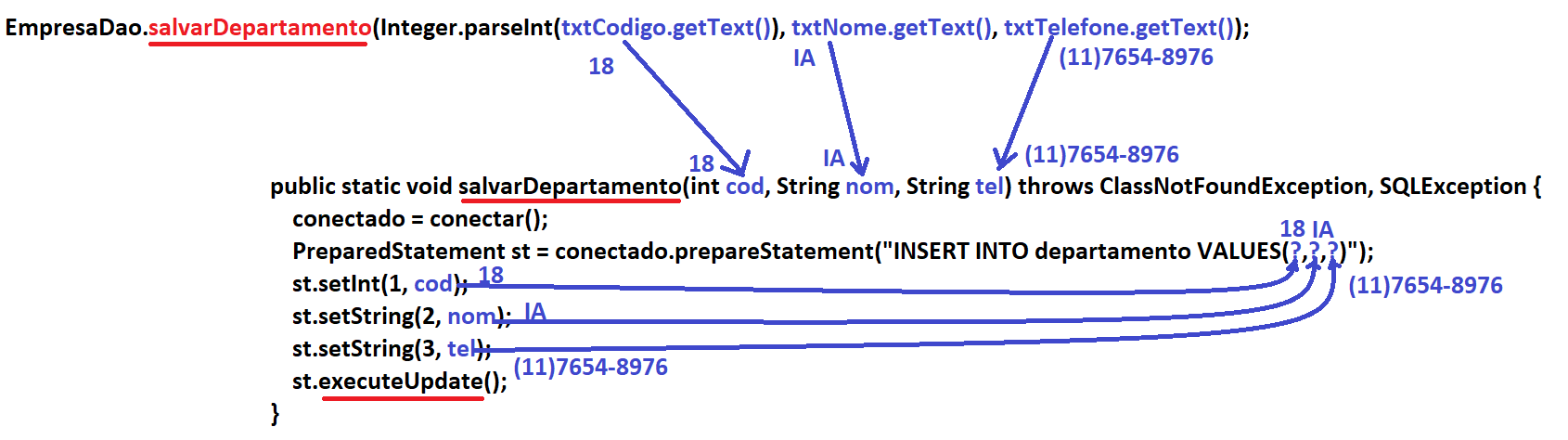
<https://drive.google.com/file/d/1e8RdXhuupOoIKWv_W3dU1_0gpnscoSuT/view?usp=sharing>

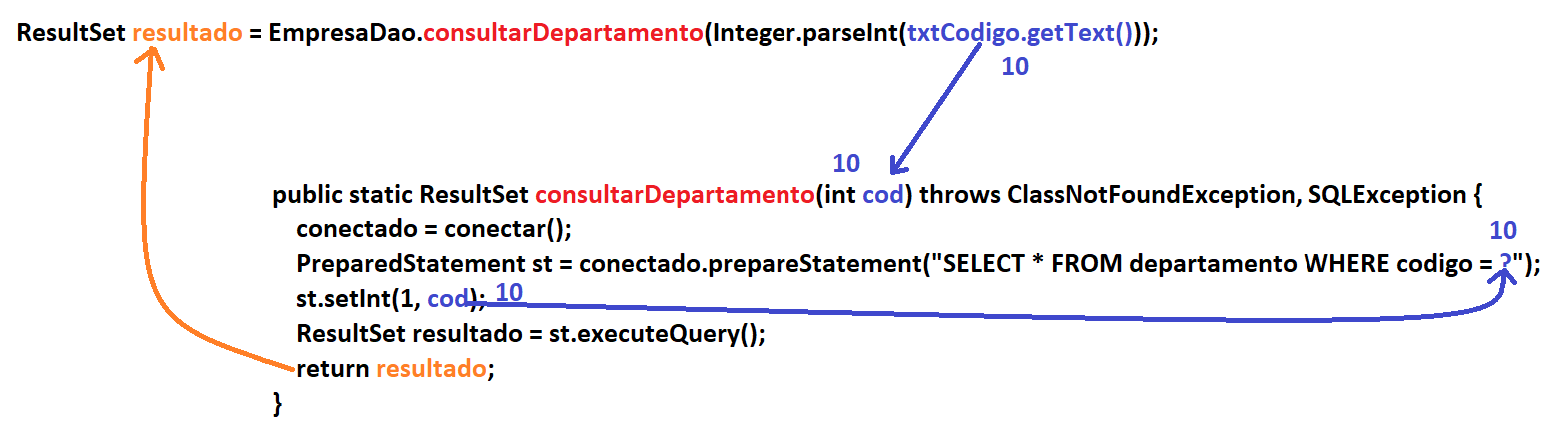
**Aula de hoje (22/05/2023):**

* Métodos
* Operadores de visibilidade









**Classe EmpresaDao.java**

package dados;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import java.sql.ResultSet;

public class EmpresaDao {

static Connection conectado;

public static Connection conectar() throws ClassNotFoundException, SQLException {

Class.forName("com.mysql.cj.jdbc.Driver");

conectado = DriverManager.getConnection("jdbc:mysql://localhost:3306/bd\_empresa", "root", "teruel");

return conectado;

}

public static ResultSet fazerLogin(String u, String s) throws ClassNotFoundException, SQLException {

conectado = conectar();

PreparedStatement st = conectado.prepareStatement("SELECT \* FROM usuario WHERE usuario = ? AND senha = ?");

st.setString(1, u);

st.setString(2, s);

ResultSet resultado = st.executeQuery();

return resultado;

}

public static void salvarDepartamento(int cod, String nom, String tel) throws ClassNotFoundException, SQLException {

conectado = conectar();

PreparedStatement st = conectado.prepareStatement("INSERT INTO departamento VALUES(?,?,?)");

st.setInt(1, cod);

st.setString(2, nom);

st.setString(3, tel);

st.executeUpdate();

}

public static ResultSet consultarDepartamento(int cod) throws ClassNotFoundException, SQLException {

conectado = conectar();

PreparedStatement st = conectado.prepareStatement("SELECT \* FROM departamento WHERE codigo = ?");

st.setInt(1, cod);

ResultSet resultado = st.executeQuery();

return resultado;

}

public static void excluirDepartamento(int cod) throws ClassNotFoundException, SQLException {

conectado = conectar();

PreparedStatement st = conectado.prepareStatement("DELETE FROM departamento WHERE codigo = ?");

st.setInt(1, cod);

st.executeUpdate();

}

public static void alterarDepartamento(String nom, String tel, int cod) throws ClassNotFoundException, SQLException {

conectado = conectar();

PreparedStatement st = conectado.prepareStatement("UPDATE departamento SET nome = ?, telefone = ? WHERE codigo = ?");

st.setString(1, nom);

st.setString(2, tel);

st.setInt(3, cod);

st.executeUpdate();

}

public static ResultSet pesquisarDepartamentoPorNome(String nom) throws ClassNotFoundException, SQLException {

conectado = conectar();

PreparedStatement st = conectado.prepareStatement("SELECT \* FROM departamento WHERE nome LIKE ?");

st.setString(1, "%" + nom + "%");

ResultSet resultado = st.executeQuery();

return resultado;

}

public static ResultSet pesquisarDepartamentoPorCodigo(String cod) throws ClassNotFoundException, SQLException {

conectado = conectar();

PreparedStatement st = conectado.prepareStatement("SELECT \* FROM departamento WHERE codigo = ?");

st.setInt(1, Integer.parseInt(cod));

ResultSet resultado = st.executeQuery();

return resultado;

}

public static ResultSet listaDepartamentos() throws ClassNotFoundException, SQLException {

conectado = conectar();

PreparedStatement st = conectado.prepareStatement("SELECT \* FROM departamento");

ResultSet resultado = st.executeQuery();

return resultado;

}

}

**Aula de hoje (23/05/2023):**

* Métodos.
* Operadores de visibilidade (private e public).
* Encapsulamento.

Link para baixar o projeto sistema16-05 do Netbeans:

<https://drive.google.com/file/d/183SbZ4I4_GPjFHC3nQH-tvaTeYFM2Kv5/view?usp=sharing>

**Projetos criados no Netbeans**

CRUD forma 1 - sistema21-03

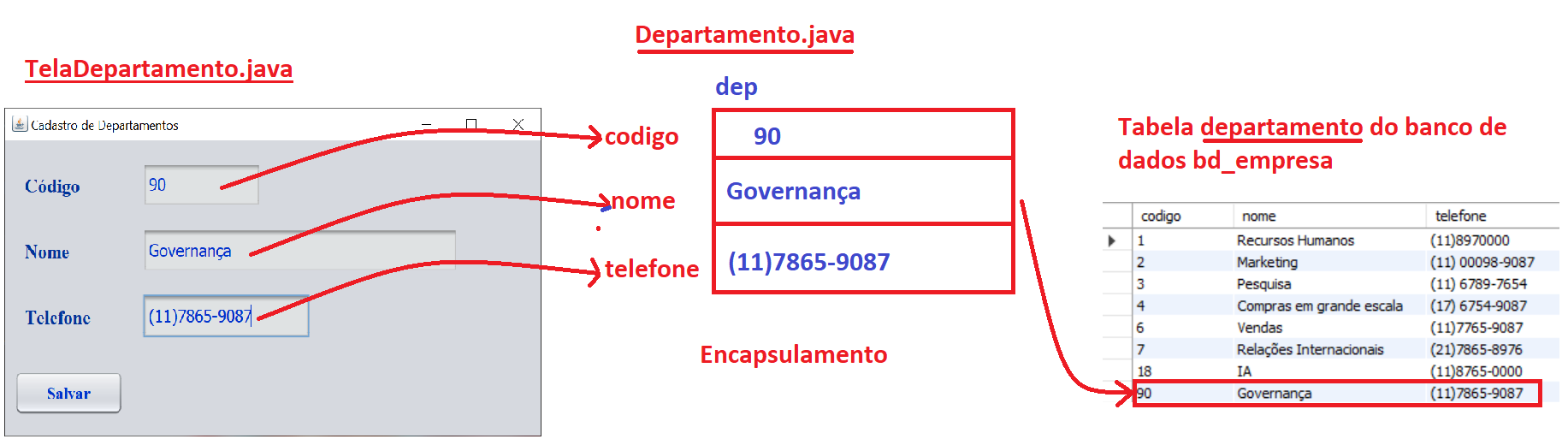
CRUD forma 2 - sistema16-05

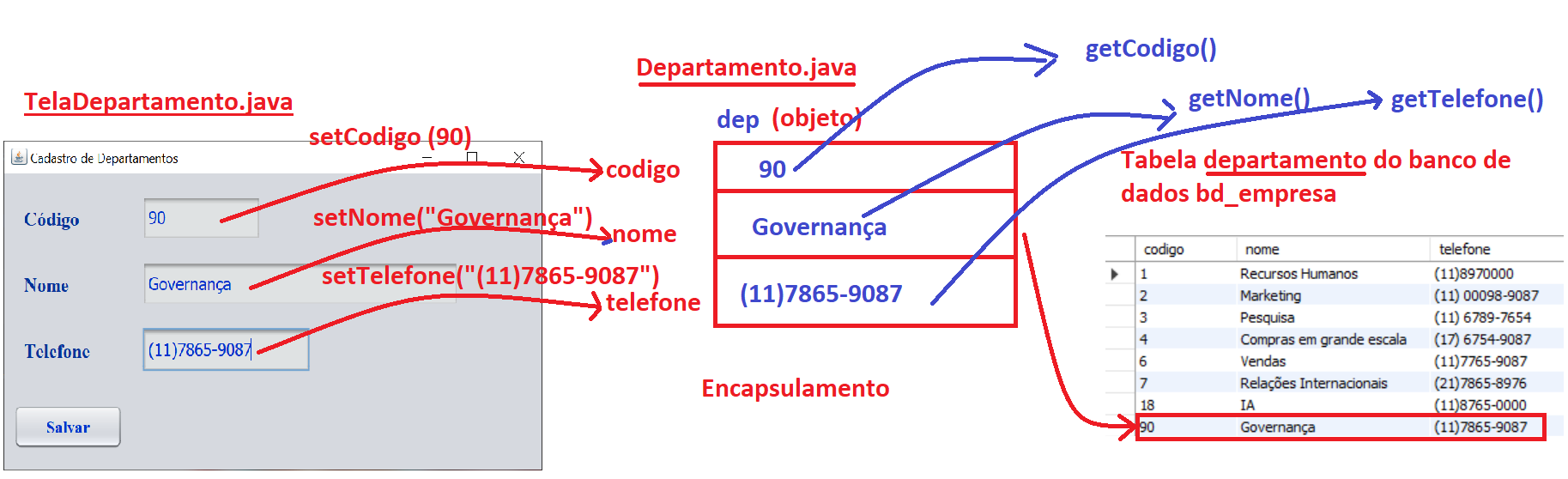
CRUD forma 3 - sistema23-05

No **CRUD forma 1** e **CRUD forma 2**, fizemos desta maneira:



No **CRUD forma 3** vamos fazer desta maneira:





**Departamento.java**

package dados;

//Modelo da tabela deprtamento do banco de dados (classe de entidade)

public class Departamento {

//Atributos da classe (variáveis privadas)

private int codigo;

private String nome;

private String telefone;

public Departamento() {

}

public void setCodigo(int codigo) {

this.codigo = codigo;

}

public void setNome(String nome) {

this.nome = nome;

}

public void setTelefone(String telefone) {

this.telefone = telefone;

}

public int getCodigo() {

return codigo;

}

public String getNome() {

return nome;

}

public String getTelefone() {

return telefone;

}

}

**Atividade para 29/05**

Incluir na classe **EmpresaDao** do projeto **sistema16-05** os métodos para os módulos de usuários e funcionários e fazer as chamadas adequadas nos botões destes módulos.

**Aula de hoje (29/05/2023):**

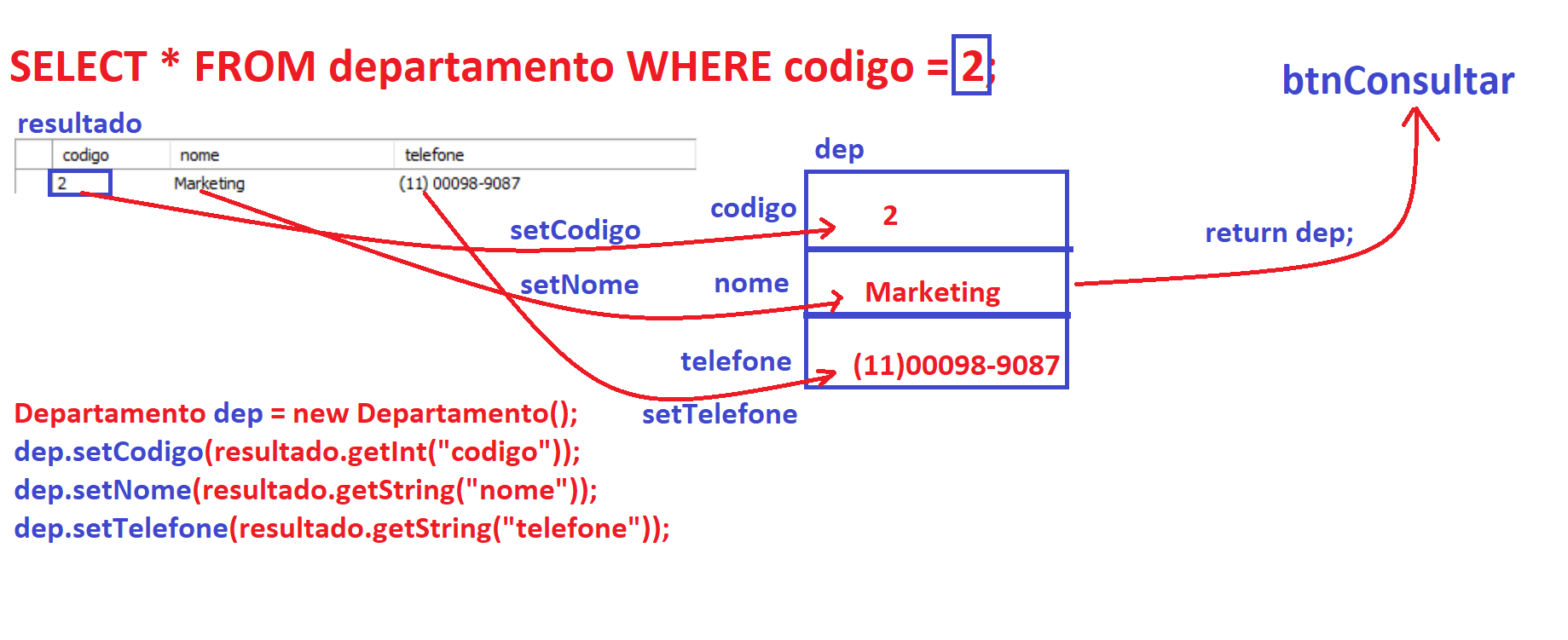
* Vista da atividade
* Encapsulamento e métodos

**Projetos criados no Netbeans - CRUD (Create, Read, Update e Delete)**

**CRUD forma 1 - sistema21-03** - operações em banco de dados diretamente no botões.

**CRUD forma 2 - sistema16-05** - operações em banco de dados foram para uma classe DAO.

**CRUD forma 3 - sistema23-05** - operações em banco de dados passam pelo encapsulamento.



public static Departamento **consultarDepartamento**(int cod) throws ClassNotFoundException, SQLException {

conectado = conectar();

st = conectado.prepareStatement("SELECT \* FROM departamento WHERE codigo = ?");

st.setInt(1, cod);

resultado = st.executeQuery();

if (resultado.next()){ //se ele achou o dep buscado

Departamento dep = new Departamento();

dep.setCodigo(resultado.getInt("codigo"));

dep.setNome(resultado.getString("nome"));

dep.setTelefone(resultado.getString("telefone"));

return dep;

} else {

return null;

}

}

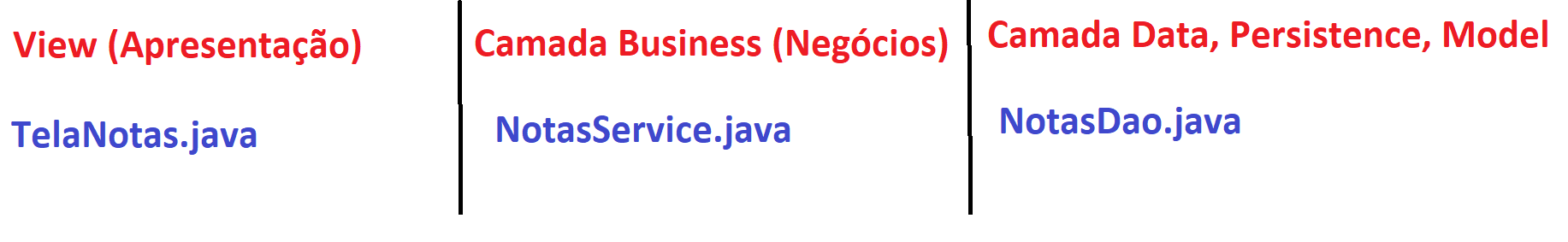
Link do projeto sistema23-05:

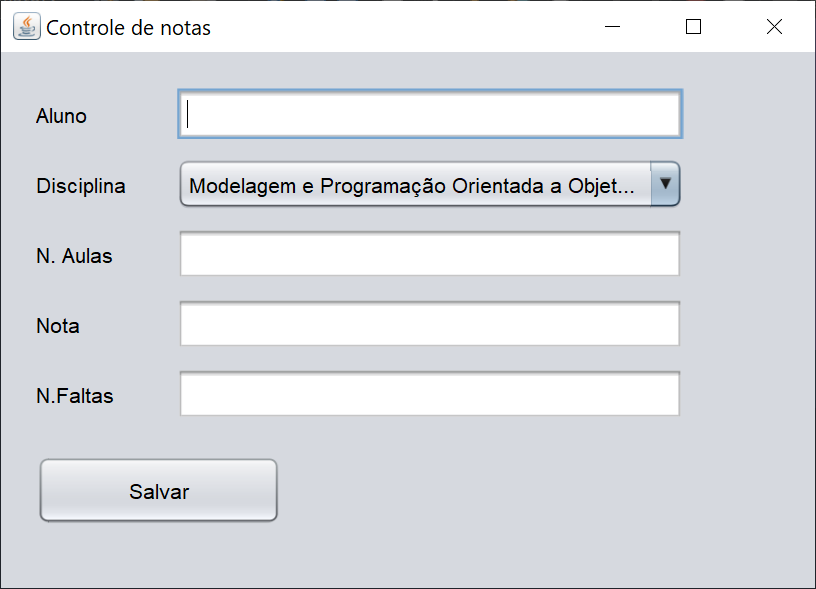
<https://drive.google.com/file/d/1kzjvcEz29fSUkN7Fr8Tty6g_mus4Q66_/view?usp=sharing>

**Prova Av1 - 05/06/2023 - 9h30**

**Aula de hoje (30/05/2023):**

* Sobrecarga de Métodos.
* Métodos estáticos.
* Exemplo.
* Programação em 3 camadas.





**NotaService.java**

public class NotasService {

//O aluno só é aprovado de tiver nota maior do que 6 e menos de 25% de faltas do total de aulas em cada disciplina.

public static String verificarAprovacao(int nAulas, int nFaltas, double nota) {

double limiteFaltas = nAulas \* 25 / 100;

if (nota < 6 || nFaltas > limiteFaltas) {

return "Reprovado";

} else {

return "Aprovado";

}

}

//O aluno só é aprovado de tiver nota maior do que 6.

public static String verificarAprovacao(double nota) {

if (nota < 6) {

return "Reprovado";

} else {

return "Aprovado";

}

}

public static String verificarAprovacao(int nota) {

if (nota < 6) {

return "Reprovado";

} else {

return "Aprovado";

}

}

}

**Aula de hoje (05/06/2023):**

* Prova Av1.
* Link da prova: <https://forms.gle/Hwc421R4S8tio4id6>

Bom dia!

A prova AV1 terá **40 questões de múltipla escolha com 5 alternativas cada**.

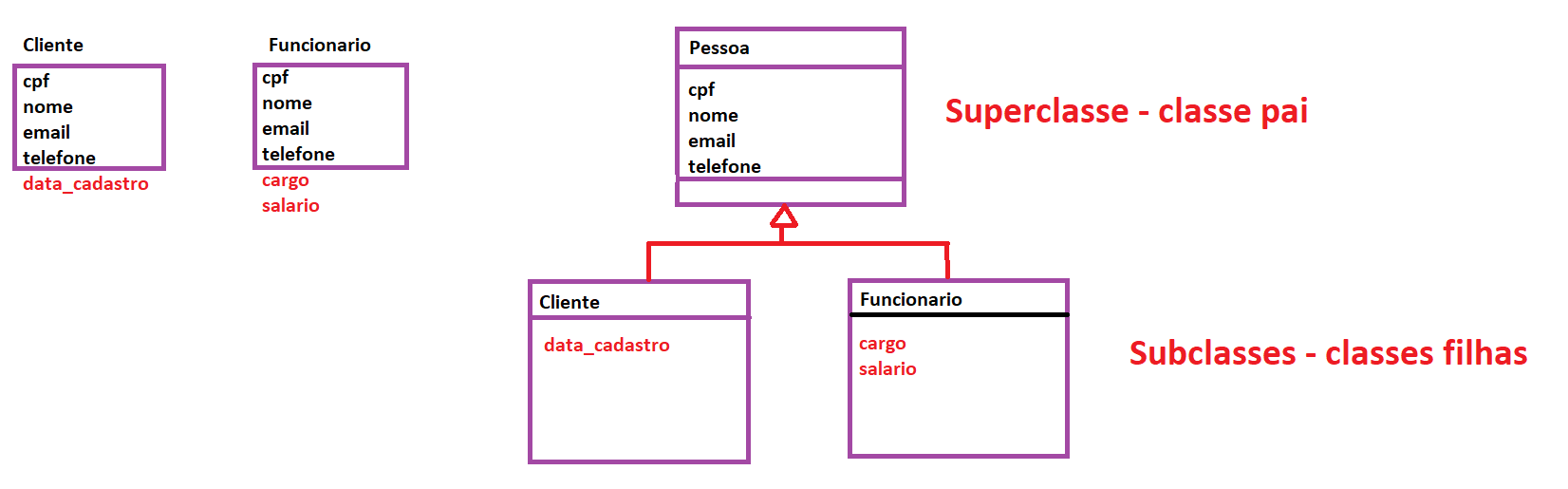
Você terá **uma hora para fazer a prova**, ou seja, 1,5 minutos para responder cada questão.

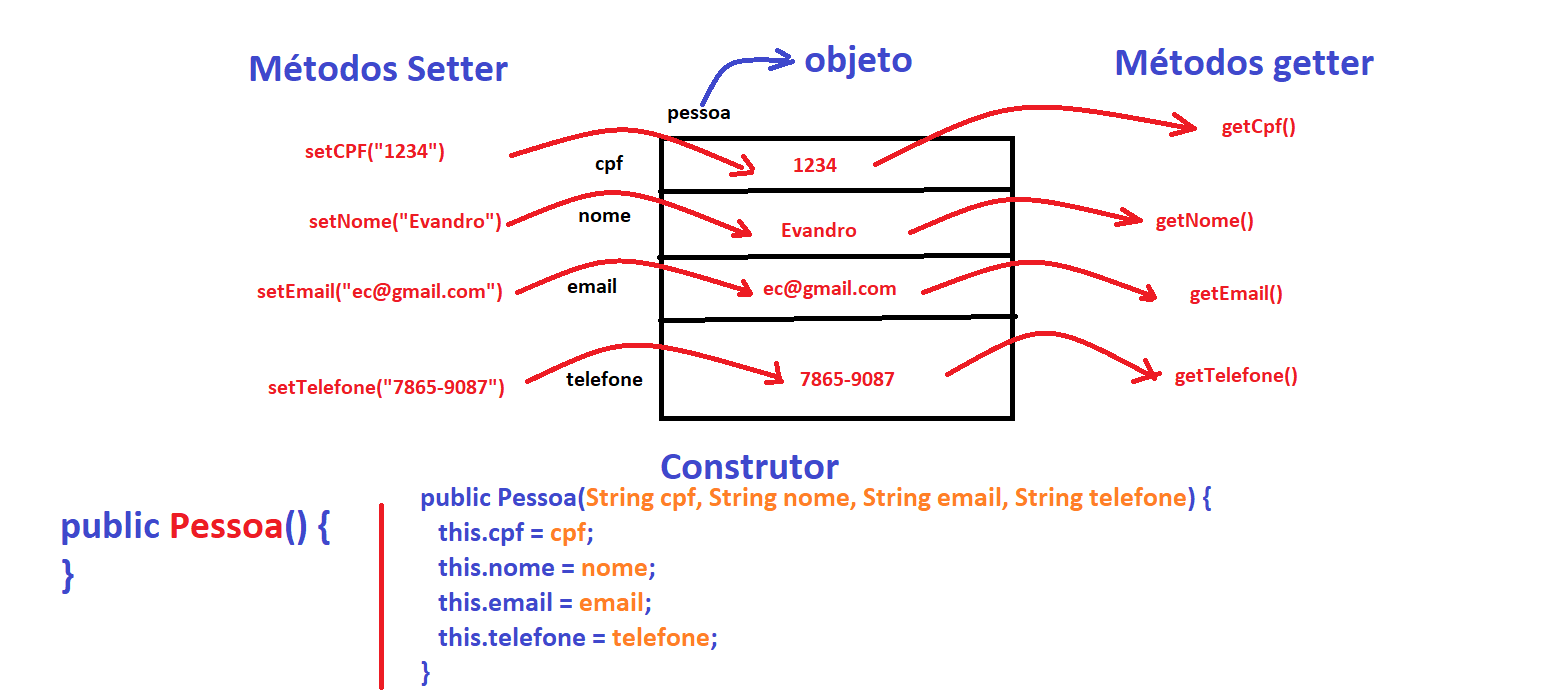
Início 10 horas.

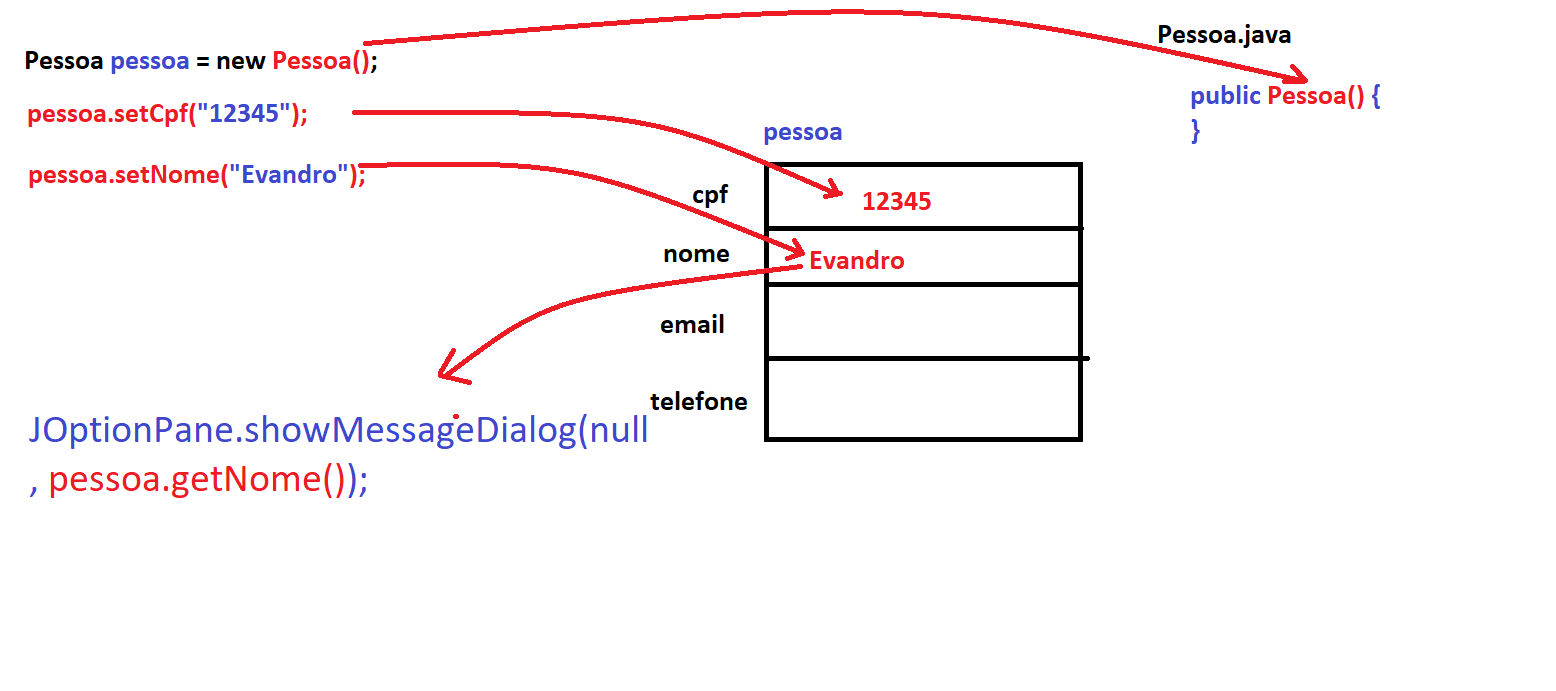
Aguardem.

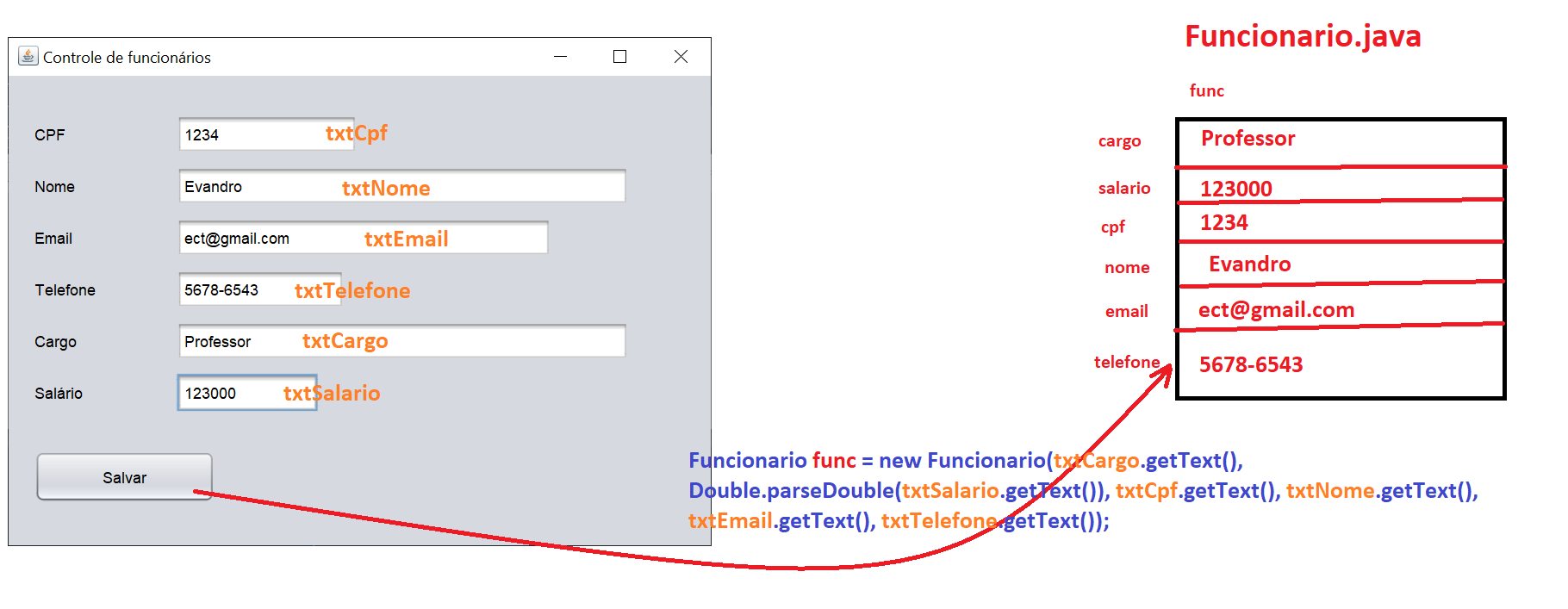
**Aula de hoje (06/06/2023):**

* Herança.









**Classe Pessoa.java**

package data;

//Superclasse, ou seja, classe pai

public class Pessoa {

//Atributos da classe

private String cpf;

private String nome;

private String email;

private String telefone;

//Construtor vazio (gera objetos vazios)

public Pessoa() {

}

//Construtor preenchido (gera objetos com os dados)

public Pessoa(String cpf, String nome, String email, String telefone) {

this.cpf = cpf;

this.nome = nome;

this.email = email;

this.telefone = telefone;

}

//Métodos setter

public void setCpf(String cpf) {

this.cpf = cpf;

}

public void setNome(String nome) {

this.nome = nome;

}

public void setEmail(String email) {

this.email = email;

}

public void setTelefone(String telefone) {

this.telefone = telefone;

}

//Métodos getter

public String getCpf() {

return cpf;

}

public String getNome() {

return nome;

}

public String getEmail() {

return email;

}

public String getTelefone() {

return telefone;

}

}

**Classe Funcionario.java**

package data;

public class Funcionario extends Pessoa {

private String cargo;

private double salario;

//Construtor preenchido

public Funcionario(String cargo, double salario, String cpf, String nome, String email, String telefone) {

super(cpf, nome, email, telefone);

this.cargo = cargo;

this.salario = salario;

}

//Métodos setter

public void setCargo(String cargo) {

this.cargo = cargo;

}

public void setSalario(double salario) {

this.salario = salario;

}

//Métodos getter

public String getCargo() {

return cargo;

}

public double getSalario() {

return salario;

}

}

**Classe Cliente.java**

package data;

public class Cliente extends Pessoa {

private String dataCadastro;

public Cliente(String dataCadastro, String cpf, String nome, String email, String telefone) {

super(cpf, nome, email, telefone);

this.dataCadastro = dataCadastro;

}

public void setDataCadastro(String dataCadastro) {

this.dataCadastro = dataCadastro;

}

public String getDataCadastro() {

return dataCadastro;

}

}

**Aula de hoje (12/06/2023):**

* Herança e CRUD

**Criação do banco de dados e das tabelas no MySQL Workbench**

create schema bd\_heranca;

use bd\_heranca;

create table funcionario(

cpf varchar(15) not null primary key,

nome varchar(70),

email varchar(30),

telefone varchar(15),

cargo varchar(30),

salario double

);

create table cliente(

cpf varchar(15) not null primary key,

nome varchar(70),

email varchar(30),

telefone varchar(15),

dataCadastro varchar(9)

);

**Classe EmpresaDao.java**

package data;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.DriverManager;

import java.sql.SQLException;

import java.sql.PreparedStatement;

public class EmpresaDao {

static String url = "jdbc:mysql://localhost:3306/bd\_heranca";

static String usuario = "root";

static String senha = "teruel";

private static Connection conectar() throws ClassNotFoundException,SQLException{

Class.forName("com.mysql.cj.jdbc.Driver");

Connection conectado = DriverManager.getConnection(url, usuario, senha);

return conectado;

}

public static void salvarFuncionario(Funcionario func) throws ClassNotFoundException, SQLException{

Connection conectado = conectar();

String sql = "INSERT INTO funcionario VALUES(?,?,?,?,?,?)";

PreparedStatement st = conectado.prepareStatement(sql);

st.setString(1, func.getCpf());

st.setString(2, func.getNome());

st.setString(3, func.getEmail());

st.setString(4, func.getTelefone());

st.setString(5, func.getCargo());

st.setDouble(6, func.getSalario());

st.executeUpdate(); // executa o INSERT

}

}

**Botão Salvar Funcionário**

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

Funcionario func = new Funcionario(txtCargo.getText(), Double.parseDouble(txtSalario.getText()), txtCpf.getText(), txtNome.getText(), txtEmail.getText(), txtTelefone.getText());

try {

EmpresaDao.salvarFuncionario(func);

JOptionPane.showMessageDialog(null,"Cadastro realizado com sucesso");

limpar();

} catch (ClassNotFoundException ex) {

JOptionPane.showMessageDialog(null,ex.getMessage());

} catch (SQLException ex) {

JOptionPane.showMessageDialog(null,ex.getMessage());

}

}

private void limpar(){

txtCpf.setText("");

txtNome.setText("");

txtEmail.setText("");

txtTelefone.setText("");

txtCargo.setText("");

txtSalario.setText("");

txtCpf.requestFocus();

}

**Exercício:** Fazer o módulo para **Salvar Clientes.**

**Aula de hoje (12/06/2023):**

* Polimorfismo
* Interface

Correção do exercício:

**Classe EmpresaDao.java**

package data;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.DriverManager;

import java.sql.SQLException;

import java.sql.PreparedStatement;

public class EmpresaDao {

static String url = "jdbc:mysql://localhost:3306/bd\_heranca";

static String usuario = "root";

static String senha = "teruel";

private static Connection conectar() throws ClassNotFoundException, SQLException {

Class.forName("com.mysql.cj.jdbc.Driver");

Connection conectado = DriverManager.getConnection(url, usuario, senha);

return conectado;

}

public static void salvarFuncionario(Funcionario func) throws ClassNotFoundException, SQLException {

Connection conectado = conectar();

String sql = "INSERT INTO funcionario VALUES(?,?,?,?,?,?)";

PreparedStatement st = conectado.prepareStatement(sql);

st.setString(1, func.getCpf());

st.setString(2, func.getNome());

st.setString(3, func.getEmail());

st.setString(4, func.getTelefone());

st.setString(5, func.getCargo());

st.setDouble(6, func.getSalario());

st.executeUpdate(); // executa o INSERT

}

public static void salvarCliente(Cliente cliente) throws ClassNotFoundException, SQLException {

Connection conectado = conectar();

String sql = "INSERT INTO cliente VALUES(?,?,?,?,?)";

PreparedStatement st = conectado.prepareStatement(sql);

st.setString(1, cliente.getCpf());

st.setString(2, cliente.getNome());

st.setString(3, cliente.getEmail());

st.setString(4, cliente.getTelefone());

st.setString(5, cliente.getDataCadastro());

st.executeUpdate(); // executa o INSERT

}

}

**Botão Salvar Cliente**

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

Cliente cliente = new Cliente(txtDataCadastro.getText(),txtCpf.getText(),txtNome.getText(),txtEmail.getText(),txtTelefone.getText());

try {

EmpresaDao.salvarCliente(cliente);

JOptionPane.showMessageDialog(null, "Cliente gravado com sucesso");

limpar();

} catch (ClassNotFoundException ex) {

JOptionPane.showMessageDialog(null, ex.getMessage());

} catch (SQLException ex) {

JOptionPane.showMessageDialog(null, ex.getMessage());

}

}

private void limpar() {

txtCpf.setText("");

txtNome.setText("");

txtEmail.setText("");

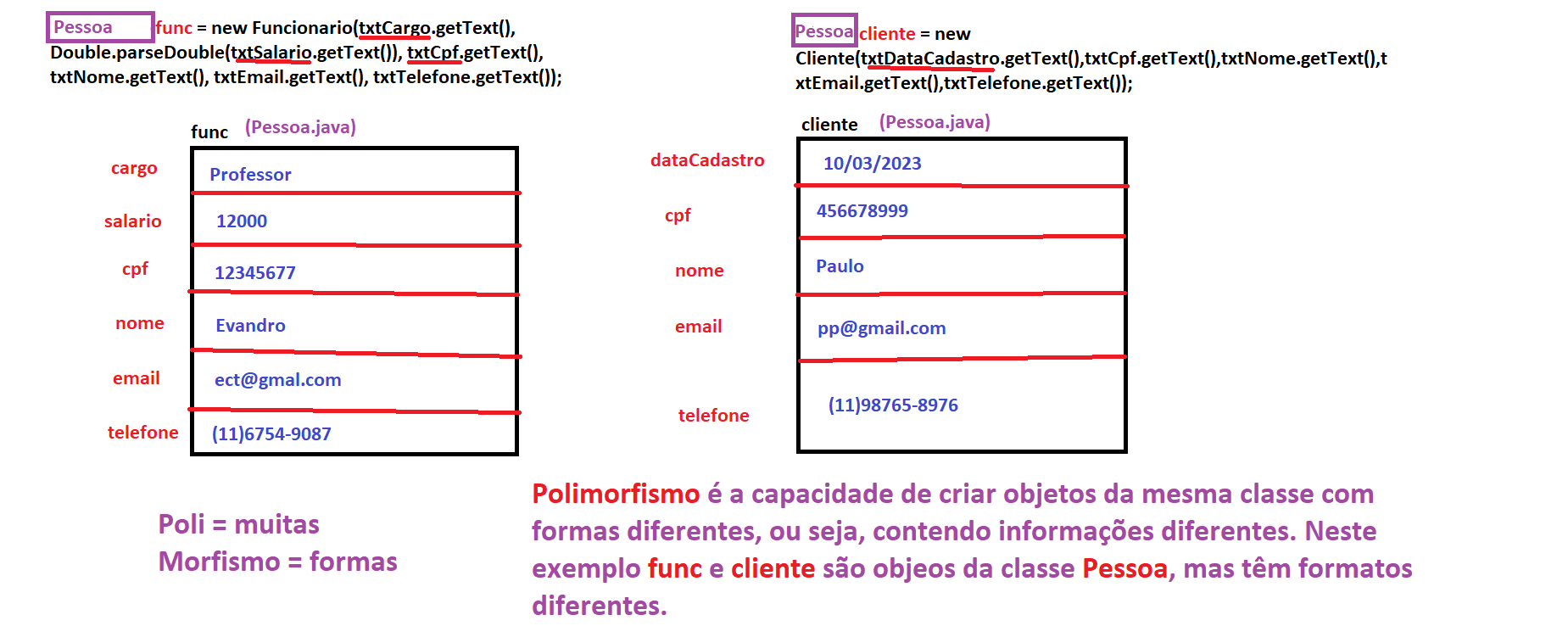
txtTelefone.setText("");

txtDataCadastro.setText("");

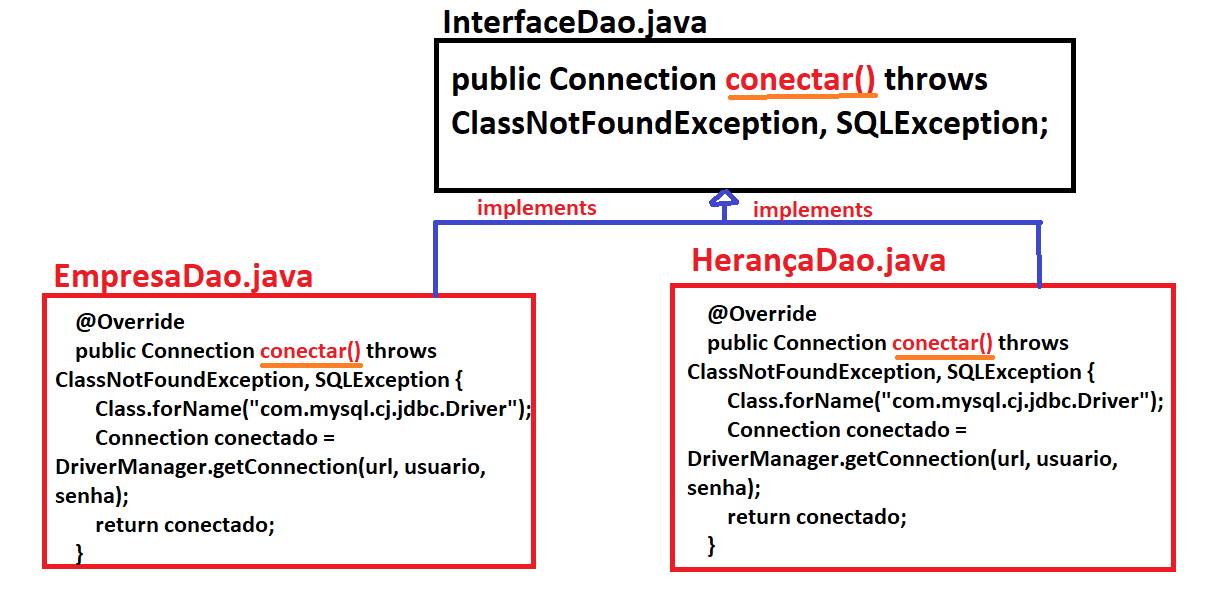
txtCpf.requestFocus();

}

**Polimorfismo**

****

**Interface Java**

****

**Interface InterfaceDao.java**

package data;

import java.sql.Connection;

import java.sql.SQLException;

public interface InterfaceDao {

public Connection conectar() throws ClassNotFoundException, SQLException;

}

**Interface EmpresDao.java**

package data;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

import java.sql.PreparedStatement;

public class EmpresaDao implements InterfaceDao {

static String url = "jdbc:mysql://localhost:3306/bd\_empresa";

static String usuario = "root";

static String senha = "teruel";

@Override

public Connection conectar() throws ClassNotFoundException, SQLException {

Class.forName("com.mysql.cj.jdbc.Driver");

Connection conectado = DriverManager.getConnection(url, usuario, senha);

return conectado;

}

public void salvarDepartamento(Departamento dep) throws ClassNotFoundException, SQLException {

Connection conectado = conectar();

String insert = "INSERT INTO departamento VALUES (?,?,?)";

PreparedStatement st = conectado.prepareStatement(insert);

st.setInt(1, dep.getCodigo());

st.setString(2, dep.getNome());

st.setString(3, dep.getTelefone());

st.executeUpdate();

}

}

**Classe HerancaDao.Java**

package data;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.DriverManager;

import java.sql.SQLException;

import java.sql.PreparedStatement;

public class HerancaDao implements InterfaceDao{

String url = "jdbc:mysql://localhost:3306/bd\_heranca";

String usuario = "root";

String senha = "teruel";

@Override

public Connection conectar() throws ClassNotFoundException, SQLException {

Class.forName("com.mysql.cj.jdbc.Driver");

Connection conectado = DriverManager.getConnection(url, usuario, senha);

return conectado;

}

public void salvarFuncionario(Funcionario func) throws ClassNotFoundException, SQLException {

Connection conectado = conectar();

String sql = "INSERT INTO funcionario VALUES(?,?,?,?,?,?)";

PreparedStatement st = conectado.prepareStatement(sql);

st.setString(1, func.getCpf());

st.setString(2, func.getNome());

st.setString(3, func.getEmail());

st.setString(4, func.getTelefone());

st.setString(5, func.getCargo());

st.setDouble(6, func.getSalario());

st.executeUpdate(); // executa o INSERT

}

public void salvarCliente(Cliente cliente) throws ClassNotFoundException, SQLException {

Connection conectado = conectar();

String sql = "INSERT INTO cliente VALUES(?,?,?,?,?)";

PreparedStatement st = conectado.prepareStatement(sql);

st.setString(1, cliente.getCpf());

st.setString(2, cliente.getNome());

st.setString(3, cliente.getEmail());

st.setString(4, cliente.getTelefone());

st.setString(5, cliente.getDataCadastro());

st.executeUpdate(); // executa o INSERT

}

}

**Classe Departamento.java**

package data;

public class Departamento {

private int codigo;

private String nome;

private String telefone;

public Departamento(int codigo, String nome, String telefone) {

this.codigo = codigo;

this.nome = nome;

this.telefone = telefone;

}

public int getCodigo() {

return codigo;

}

public void setCodigo(int codigo) {

this.codigo = codigo;

}

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;

}

}

**Botão Salvar Departamento**

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

Departamento dep = new Departamento(Integer.parseInt(txtCodigo.getText()), txtNome.getText(), txtTelefone.getText());

try {

new EmpresaDao().salvarDepartamento(dep);

JOptionPane.showMessageDialog(null, "Cliente gravado com sucesso");

limpar();

} catch (ClassNotFoundException ex) {

JOptionPane.showMessageDialog(null, ex.getMessage());

} catch (SQLException ex) {

JOptionPane.showMessageDialog(null, ex.getMessage());

}

}

private void limpar(){

txtCodigo.setText("");

txtNome.setText("");

txtTelefone.setText("");

txtCodigo.requestFocus();

}

**Aula de hoje (19/06/2023):**

* Modelagem (diagramas de caso de uso e classe da UML)

Ferramenta para desenho de diagramas da UML

<https://www.lucidchart.com/>

UML - Unified Modeling Language

<https://www.uml-diagrams.org/uml-25-diagrams.html>

**UML** é uma linguagem para desenhar (modelar) sistemas de software.

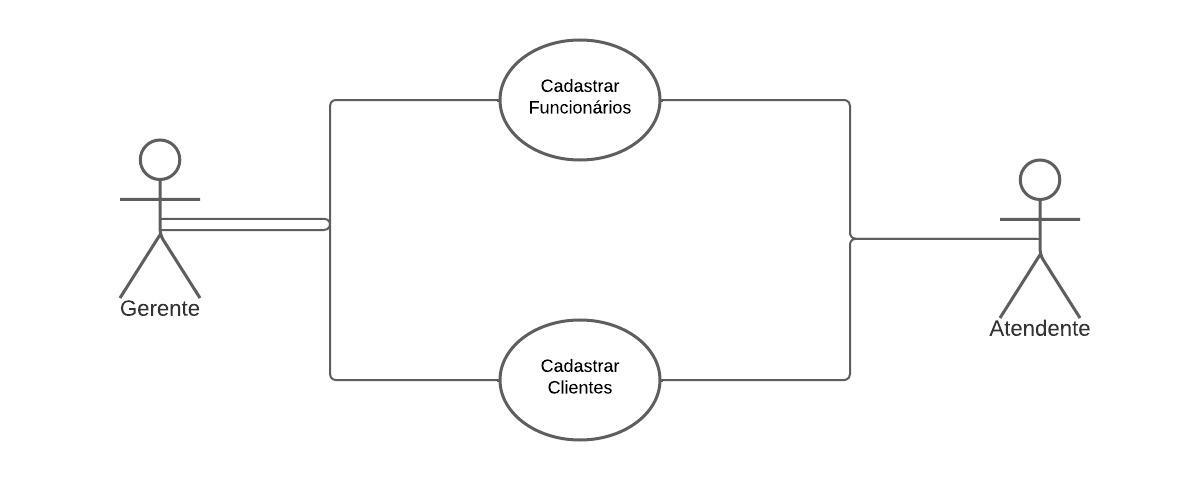
**Requisitos funcionais ( o que o software vai fazer)**

* cadastrar funcionários.
* cadastrar clientes.

**Requisitos não-funcionais (como o software vai ser)**

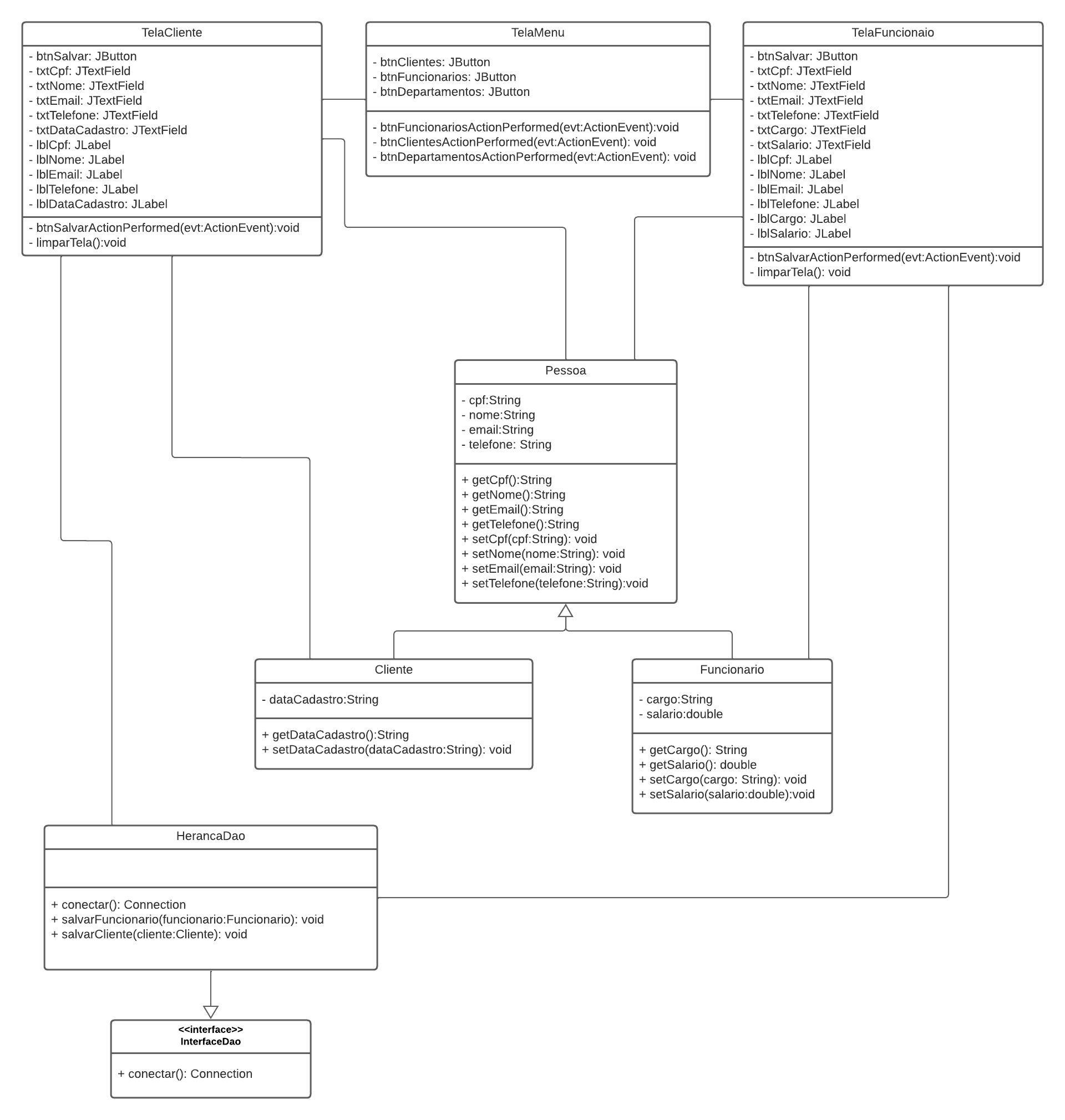
* fácil de usar (boa usabilidade).
* desenvolvido em Java com banco de dados MYSQL.
* adequado à Lei Geral de Proteção de Dados.
* para ambiente desktop.

**Diagrama de caso de uso (baseado nos requisitos funcionais)**



O diagrama de casos de uso mostra quem (quais atores) se relacionam com quais funcionalidades da aplicação.

**Diagrama de classes (programas)**



O **diagrama de classes** mostra os atributos e métodos das classes da aplicação e com quais outras classes elas se relacionam.

**EXERCÍCIO/Atividade:** Criar o botão **Salvar** do módulo Controle de Departamentos:

