

# Linguagem de Programação III

---

Java Swing + Banco de dados

# Projeto com banco MySQL

---

- E agora, vamos fazer a consulta de todos as pessoas cadastradas no banco?
- Para isso, vamos lidar novamente na classe PessoaDAO:

```
public List<Pessoa> getPessoas(){

    String sql = "SELECT * FROM pessoa";
    try {

        PreparedStatement stmt = conn.prepareStatement(string:sql,i: ResultSet.TYPE_SCROLL_INSENSITIVE,
            if: ResultSet.CONCUR_UPDATABLE);

        ResultSet rs = stmt.executeQuery(); //obtenho o retorno da consulta e armazeno no ResultSet
        List<Pessoa> listaPessoas = new ArrayList(); //Preparo uma lista de objetos que vou armazenar a consulta
        //Percorre rs e salvar as informações dentro de um objeto Pessoa e depois adiciona na lista
        while (rs.next()){
            Pessoa p = new Pessoa();
            p.setId(id: rs.getInt(string:"id"));
            p.setNome(nome: rs.getString(string:"nome"));
            p.setSexo(sexo: rs.getString(string:"sexo"));
            p.setIdioma(idioma:rs.getString(string:"idioma"));
            listaPessoas.add(e: p);
        }
        return listaPessoas;
    } catch (SQLException ex) {
        System.out.println("Erro ao consultar todas as pessoas: "+ex.getMessage());
        return null;
    }
}
```

# Projeto com banco MySQL

---

```
public List<Pessoa> getPessoas() {

    String sql = "SELECT * FROM pessoa";
    try {

        PreparedStatement stmt = conn.prepareStatement(string:sql,i: ResultSet.TYPE_SCROLL_INSENSITIVE,
            il: ResultSet.CONCUR_UPDATABLE);

        ResultSet rs = stmt.executeQuery(); //obtenho o retorno da consulta e armazeno no ResultSet
        List<Pessoa> listaPessoas = new ArrayList(); //Preparo uma lista de objetos que vou armazenar a consulta
        //Percorre rs e salvar as informações dentro de um objeto Pessoa e depois adiciona na lista
        while (rs.next()){
            Pessoa p = new Pessoa();
            p.setId(id: rs.getInt(string:"id"));
            p.setNome(nome: rs.getString(string:"nome"));
            p.setSexo(sexo: rs.getString(string:"sexo"));
            p.setIdioma(idioma:rs.getString(string:"idioma"));
            listaPessoas.add(e: p);
        }
        return listaPessoas;
    } catch (SQLException ex) {
        System.out.println("Erro ao consultar todas as pessoas: "+ex.getMessage());
        return null;
    }
}
```

# Projeto com banco MySQL

---

- Depois, precisaremos ter uma tela (Jframe Form) desta forma (chamada RelatorioPessoas):

Relatório das Pessoas			
ID	Nome	Sexo	Idioma

# Projeto com banco MySQL

---

- Depois, precisaremos de um método para preencher a tabela com uma lista de Pessoas que é retornada do método que temos no DAO:

```
public void preencheTabela() {
    PessoaDAO pDAO = new PessoaDAO();
    List<Pessoa> listaPessoas = pDAO.getPessoas();

    DefaultTableModel tabelaPessoas = (DefaultTableModel) tbl_Pessoas.getModel();

    //Agora, precisamos percorrer a lista de pessoas que foi consultada e adicionar na tabela:
    for(Pessoa p: listaPessoas){
        Object[] obj = new Object[]{
            p.getId(),
            p.getNome(),
            p.getSexo(),
            p.getIdioma() };
        tabelaPessoas.addRow(rowData: obj);
    }
}
```

# Projeto com banco MySQL

---

- Depois, no construtor do JFrame Form criado, precisamos chamar este método:

```
public RelatorioPessoas() {  
    initComponents();  
    preencheTabela();  
}
```



ID	Nome	Sexo	Idioma
2	Ricardo Frohlich	M	PORTUGUÊS
5	Teste	F	Inglês
6	Rafael	M	Espanhol
7	Sabrina	F	Português
8	Ana Paula	F	Inglês

## Projeto com banco MySQL

---

- Agora, vamos unir todas as telas a partir de um menu principal?
  - Fica com vocês

## Projeto com banco MySQL

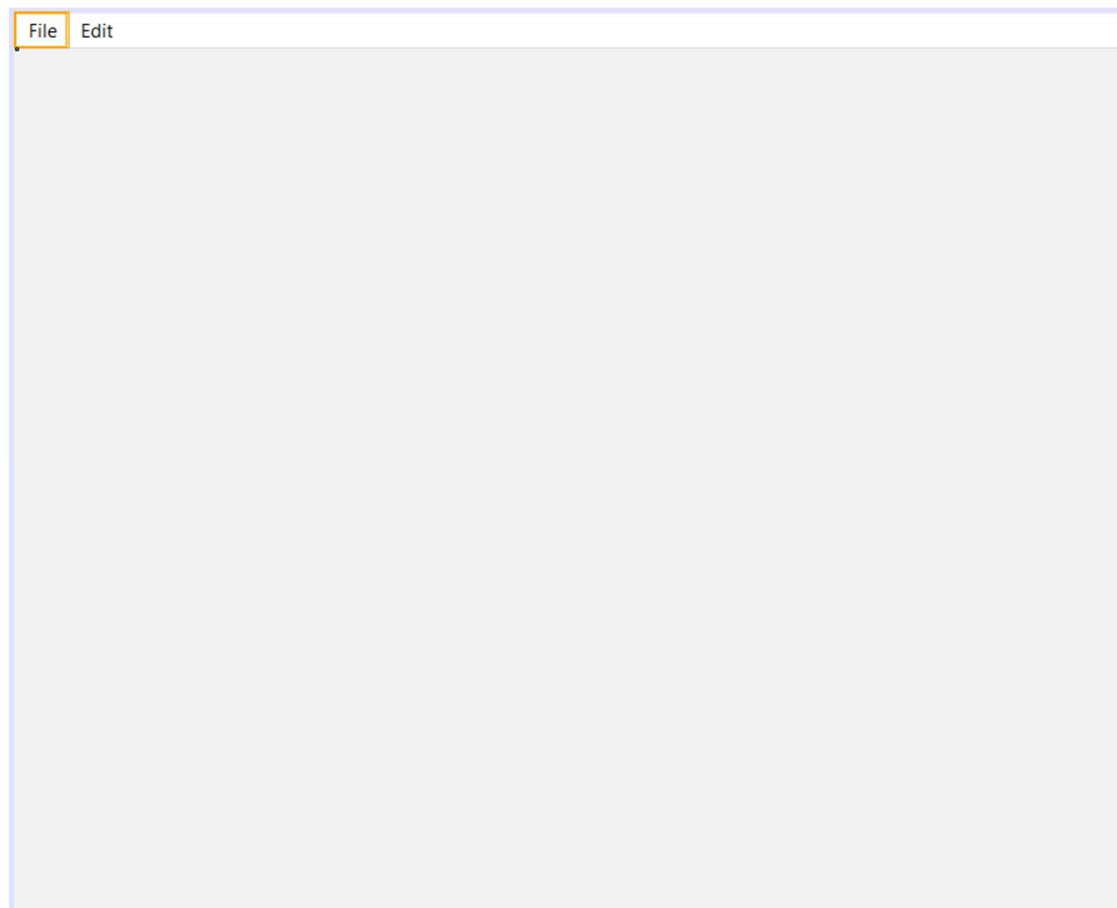
---

- Agora, vamos unir todas as telas a partir de um menu principal?
  - Crie um formulário novo (Jframe Form) chamado MenuPrincipal
    - Adicione um componente chamado MenuBar



# Projeto com banco MySQL

---



## Projeto com banco MySQL

---

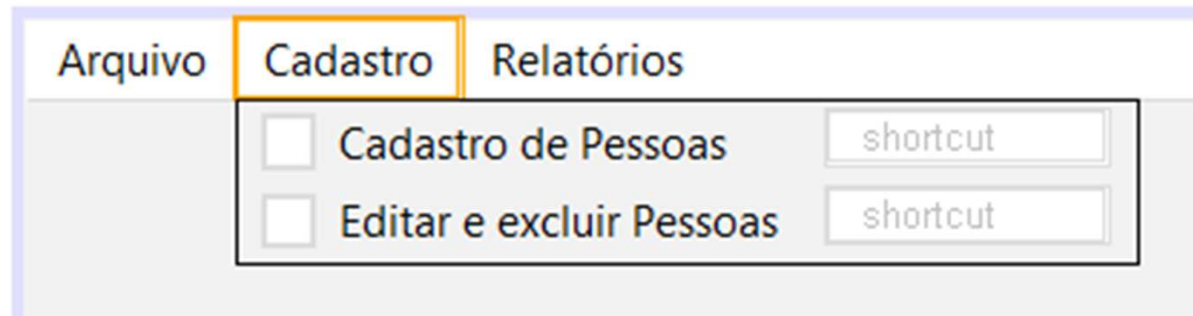
- Aí vocês podem renomear e/ adicionar outros itens
  - Para adicionar novo itens, basta adicionar outros itens de Menu
    - Adicione um Menu chamado Cadastro
    - Adicione um Menu chamado Relatórios



## Projeto com banco MySQL

---

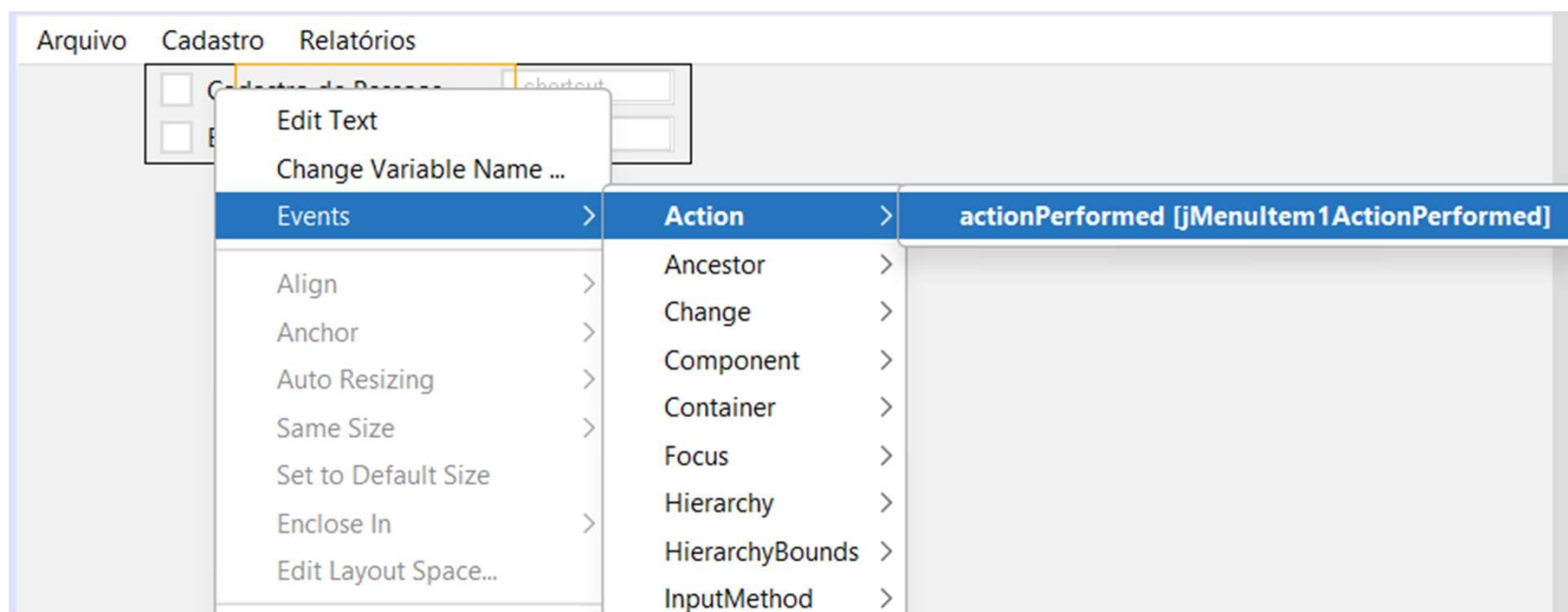
- Após, dentro do Cadastro, adicione dois “Menu Item” do cadastro chamados:



# Projeto com banco MySQL

---

- Após, vamos chamar a tela de cadastro de pessoas:



## Projeto com banco MySQL

---

- Após, vamos chamar a tela de cadastro de pessoas:

```
private void menu_CadPessoaActionPerformed(java.awt.event.ActionEvent evt) {  
    CadastraPessoa cadPessoa = new CadastraPessoa();  
    cadPessoa.setVisible(b: true);  
}
```

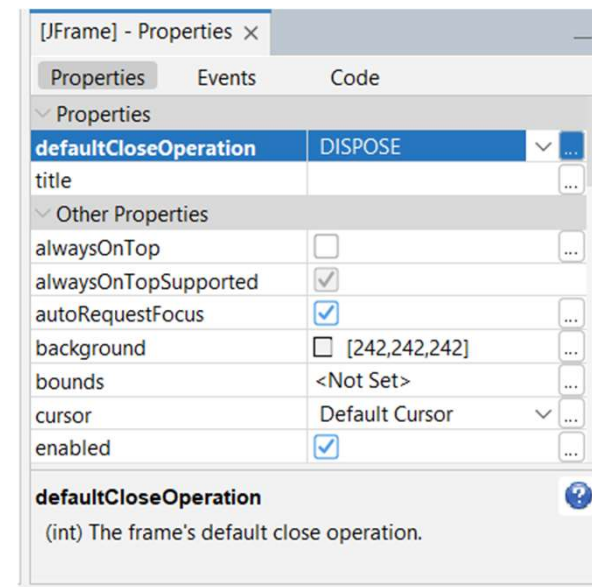
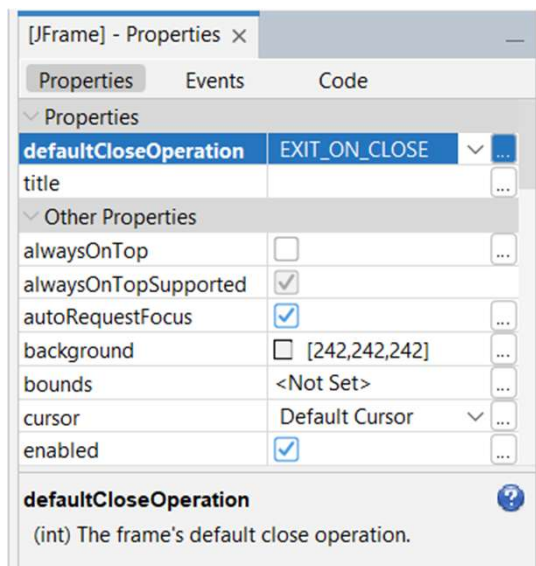
## Projeto com banco MySQL

---

- Testaram? Funcionou?
  - Quando fecha o cadastro, o que acontece?

# Projeto com banco MySQL

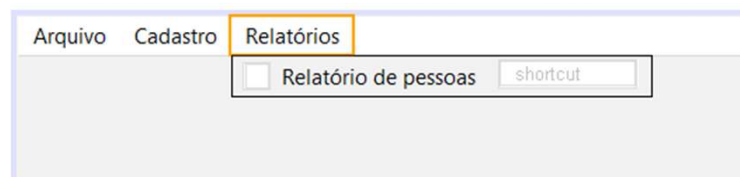
- Vamos resolver isso:
- Vai na tela do Cadastro de pessoas e vá em propriedades e procure pela propriedade DefaultCloseOperation e mude para DISPOSE



# Projeto com banco MySQL

---

- Agora, façam com o restante:



```
private void menu_EditarActionPerformed(java.awt.event.ActionEvent evt) {  
    ConsultaPessoa consPessoa = new ConsultaPessoa();  
    consPessoa.setVisible(b: true);  
}
```

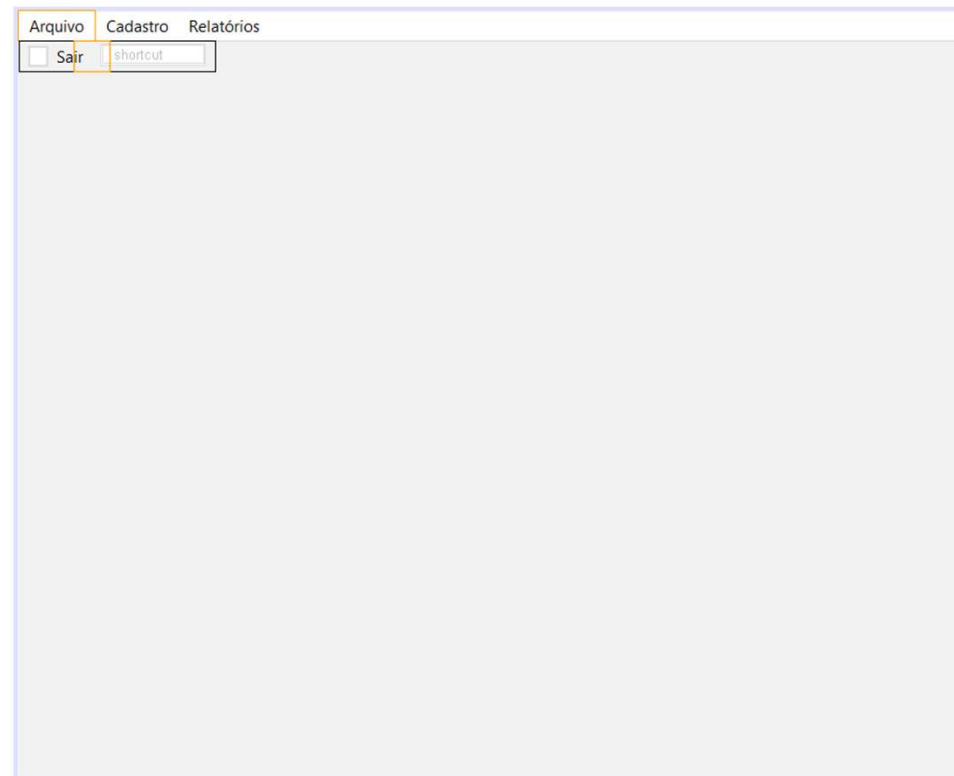
```
private void menu_RelatorioActionPerformed(java.awt.event.ActionEvent evt) {  
    RelatorioPessoas relPessoa = new RelatorioPessoas();  
    relPessoa.setVisible(b: true);  
}
```



# Projeto com banco MySQL

---

- Agora, no arquivo, adicione o botão item de menu Sair:



## Projeto com banco MySQL

---

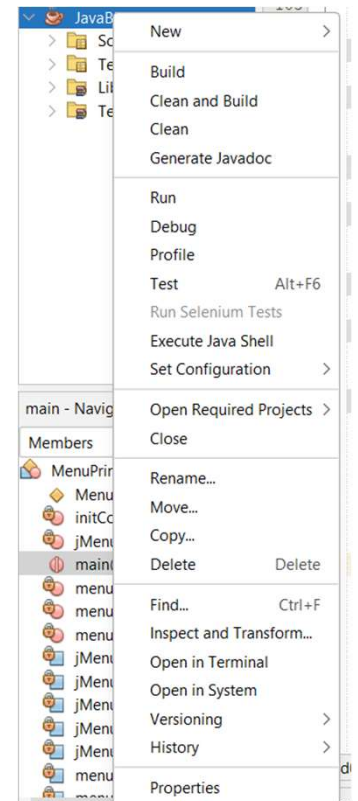
- Agora, no arquivo, adicione o botão item de menu Sair:

```
private void jMenuItemActionPerformed(java.awt.event.ActionEvent evt) {  
    //this.dispose(); - fecha a tela  
    System.exit(status: 0);  
}
```

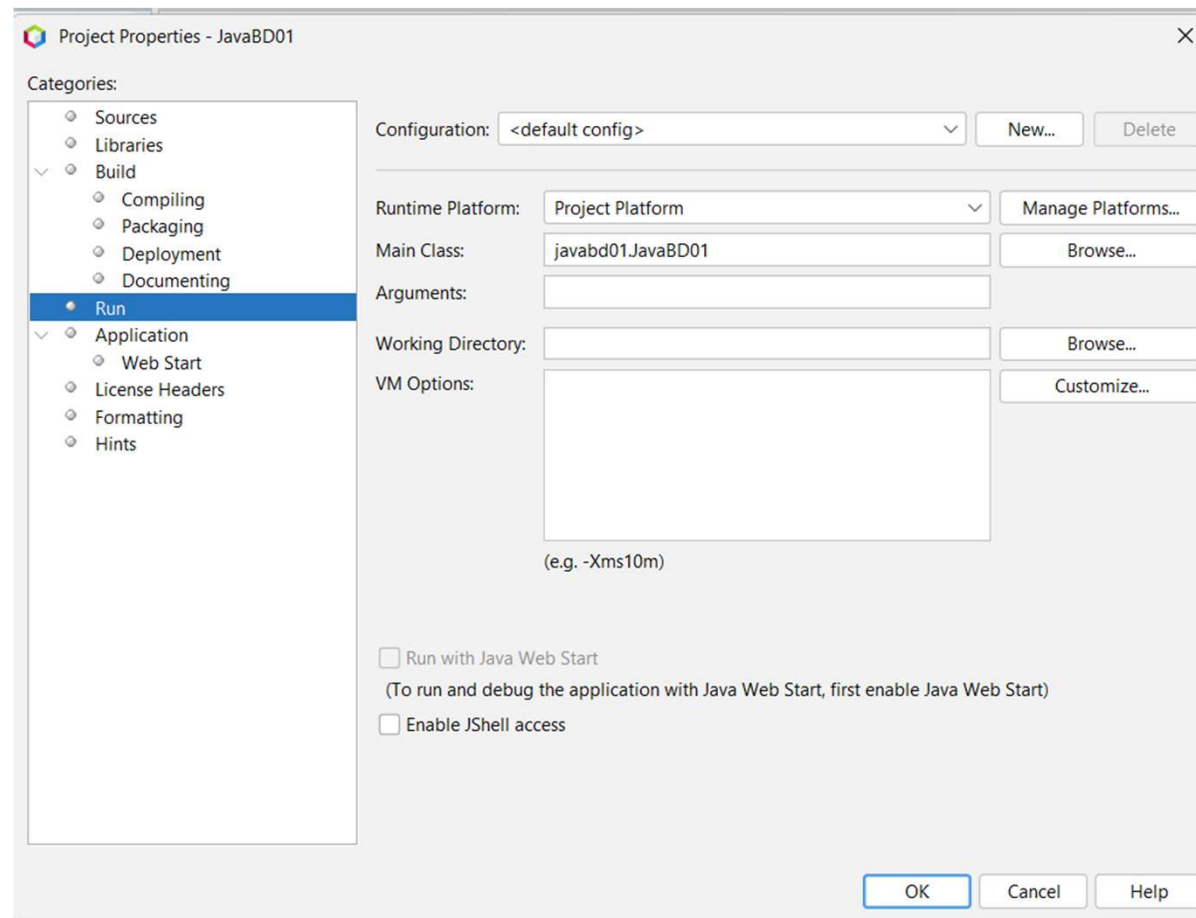
# Projeto com banco MySQL

---

- Outra coisa que podemos melhorar é definir qual é a classe padrão do nosso programa, ou seja, qual será a classe que será a primeira a ser executada
- Para isso, clique com o botão direito no projeto
- Vá em propriedades

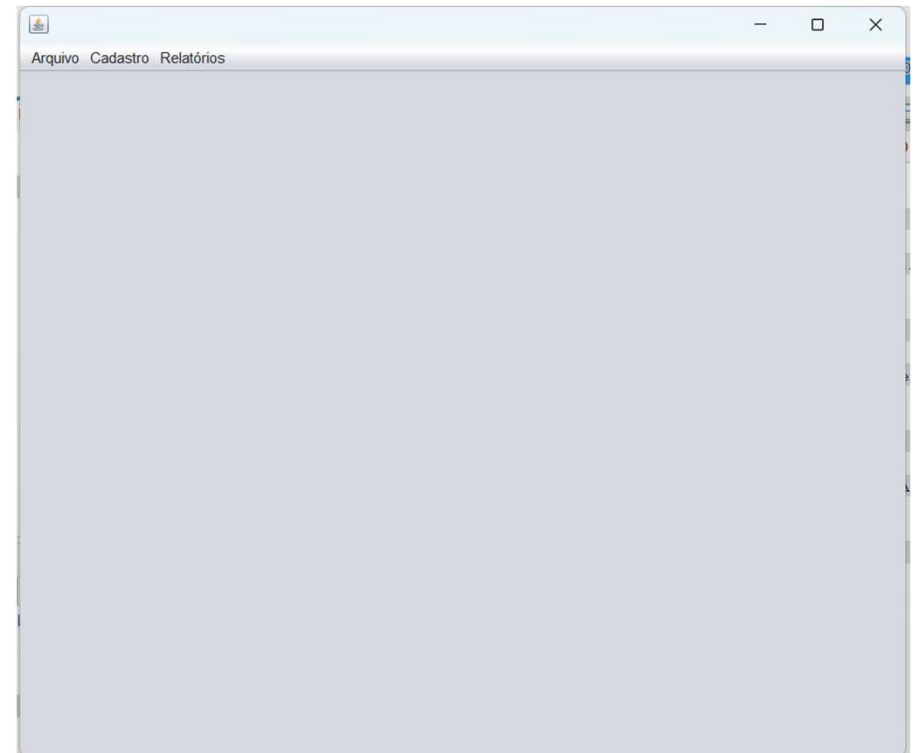
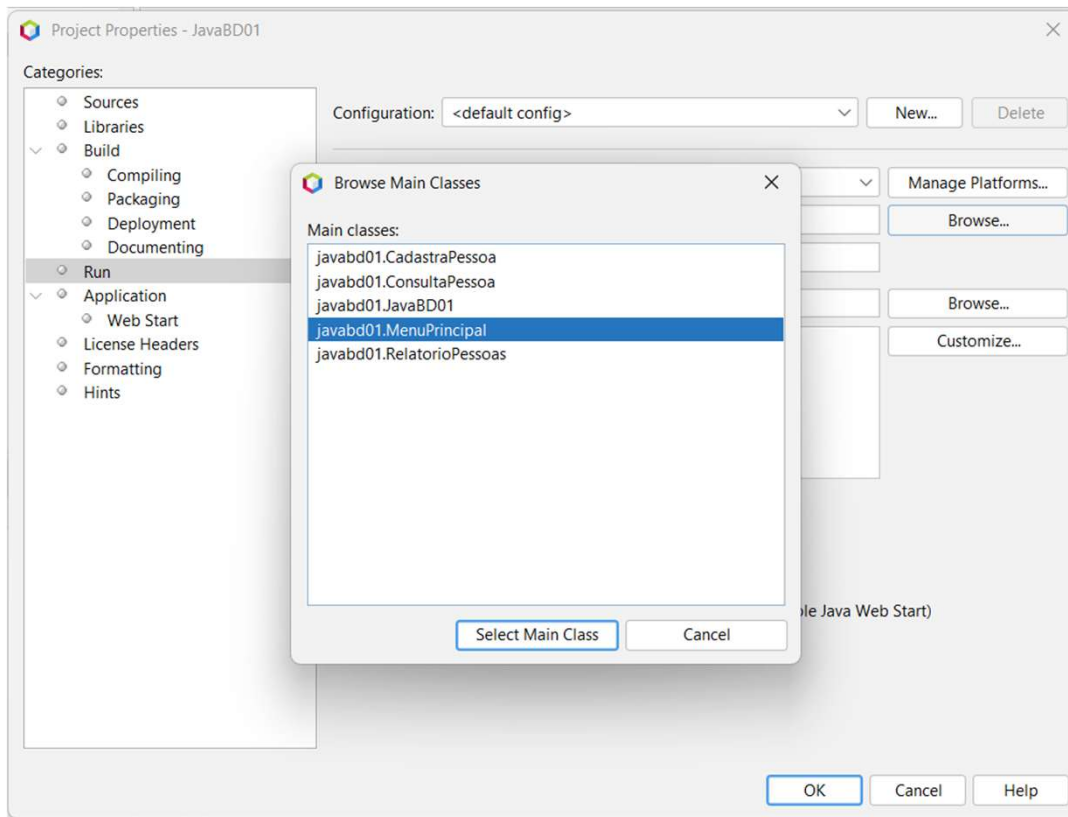


# Projeto com banco MySQL



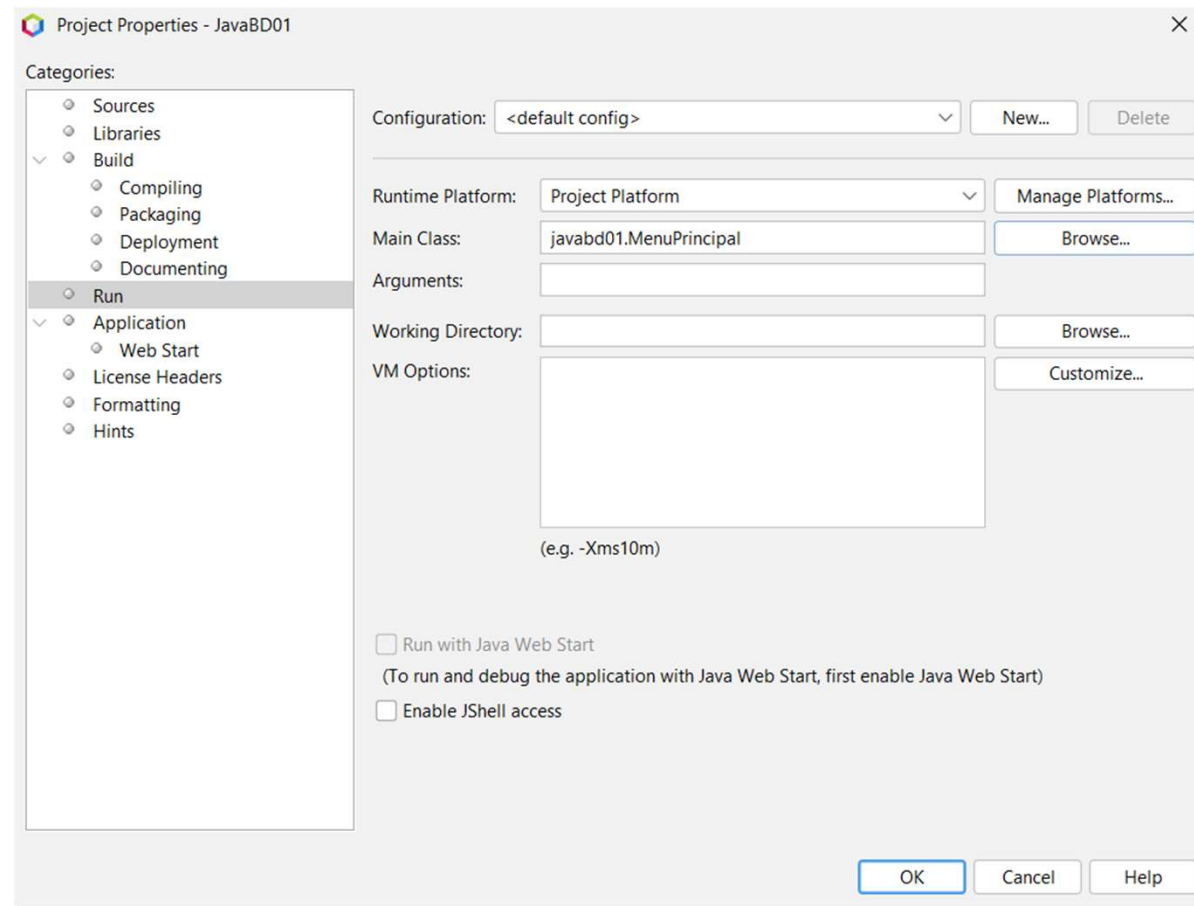
# Projeto com banco MySQL

- E compilem:



# Projeto com banco MySQL

---



## Agora, vamos fazer uma pesquisa pelo nome:

---

- Precisamos criar um método que receba um parâmetro para pesquisa e alterar a consulta e adicionar o parâmetro na consulta SQL.

```
public List<Pessoa> getPessoasNome(String nome){
    // "SELECT * FROM pessoa WHERE nome LIKE ?"
    String sql = "SELECT * FROM pessoa WHERE nome LIKE ?";
    try {

        PreparedStatement stmt = conn.prepareStatement(string:sql,i: ResultSet.TYPE_SCROLL_INSENSITIVE,
            i1: ResultSet.CONCUR_UPDATABLE);
        stmt.setString(i: 1 ,"%"+ nome +"%");
        ResultSet rs = stmt.executeQuery(); //obtenho o retorno da consulta e armazeno no ResultSet
        List<Pessoa> listaPessoas = new ArrayList(); //Preparo uma lista de objetos que vou armazenar a consulta
        //Percorre rs e salvar as informações dentro de um objeto Pessoa e depois adiciona na lista
        while (rs.next()){
            Pessoa p = new Pessoa();
            p.setId(id: rs.getInt(string:"id"));
            p.setNome(nome: rs.getString(string:"nome"));
            p.setSexo(sexo: rs.getString(string:"sexo"));
            p.setIdioma(idioma:rs.getString(string:"idioma"));
            listaPessoas.add(e: p);
        }
        return listaPessoas;
    } catch (SQLException ex) {
        System.out.println("Erro ao consultar todas as pessoas: "+ex.getMessage());
        return null;
    }
}
```

# Agora, vamos fazer uma pesquisa pelo nome:

---

```
public List<Pessoa> getPessoasNome(String nome) {  
    // "SELECT * FROM pessoa WHERE nome LIKE ?"  
    String sql = "SELECT * FROM pessoa WHERE nome LIKE ?";  
    try {  
  
        PreparedStatement stmt = conn.prepareStatement(string:sql,i: ResultSet.TYPE_SCROLL_INSENSITIVE,  
            i1: ResultSet.CONCUR_UPDATABLE);  
        stmt.setString(i: 1 ,"%"+ nome +"%");  
        ResultSet rs = stmt.executeQuery(); //obtenho o retorno da consulta e armazeno no ResultSet  
        List<Pessoa> listaPessoas = new ArrayList(); //Preparo uma lista de objetos que vou armazenar a consulta  
        //Percorre rs e salvar as informações dentro de um objeto Pessoa e depois adiciona na lista  
        while (rs.next()){  
            Pessoa p = new Pessoa();  
            p.setId(id: rs.getInt(string:"id"));  
            p.setNome(nome: rs.getString(string:"nome"));  
            p.setSexo(sexo: rs.getString(string:"sexo"));  
            p.setIdioma(idioma:rs.getString(string:"idioma"));  
            listaPessoas.add(e: p);  
        }  
        return listaPessoas;  
    } catch (SQLException ex) {  
        System.out.println("Erro ao consultar todas as pessoas: "+ex.getMessage());  
        return null;  
    }  
}
```



E precisaremos colocar uma entrada de dados na tela

---

Relatório das Pessoas

Nome:

ID	Nome	Sexo	Idioma
----	------	------	--------

## E para fazer a consulta?

---

- O que precisaremos?
  - Como vamos fazer a chamada do método para pesquisar após digitar?

## Precisamos alterar o método preencheTabela

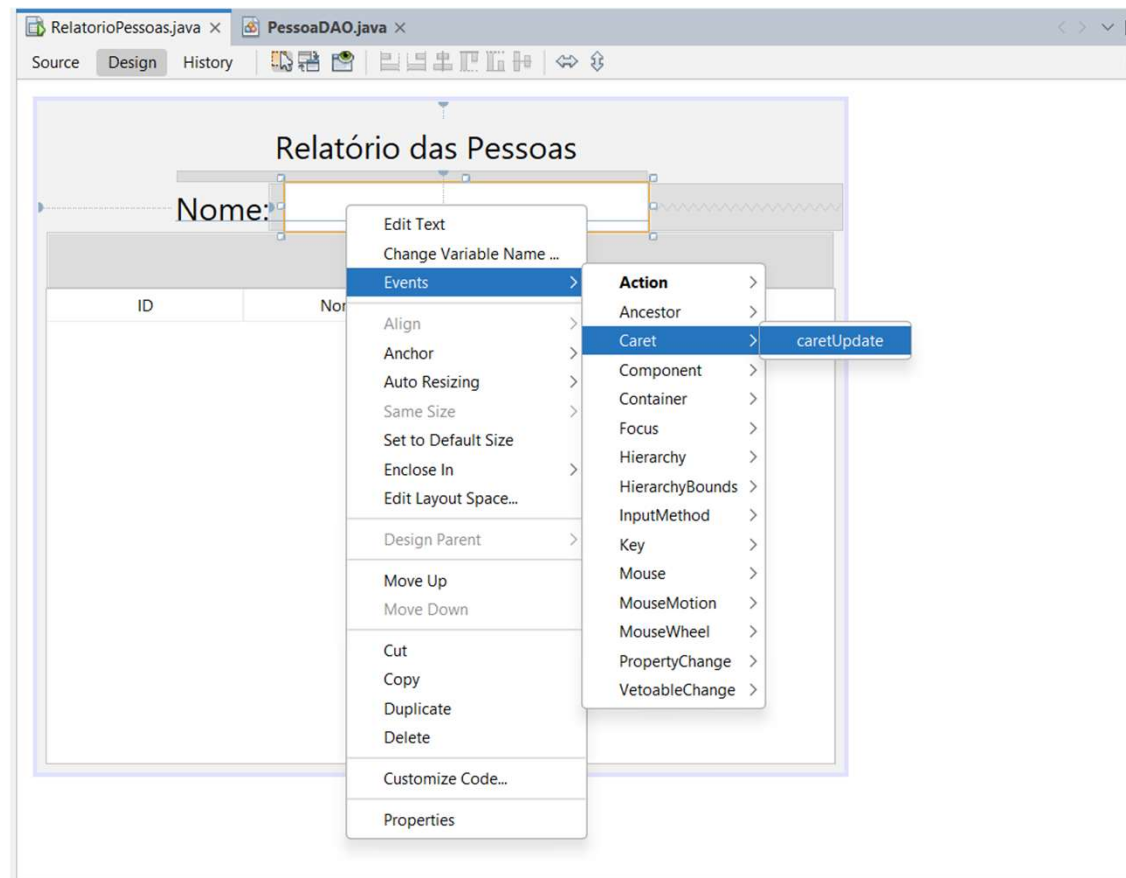
---

```
public void preencheTabela() {
    PessoaDAO pDAO = new PessoaDAO();
    //List<Pessoa> listaPessoas = pDAO.getPessoas();

    List<Pessoa> listaPessoas = pDAO.getPessoasNome(nome: txtNome.getText());
    DefaultTableModel tabelaPessoas = (DefaultTableModel) tbl_Pessoas.getModel();

    //Agora, precisamos percorrer a lista de pessoas que foi consultada e adicionar na tabela:
    for(Pessoa p: listaPessoas){
        Object[] obj = new Object[]{
            p.getId(),
            p.getNome(),
            p.getSexo(),
            p.getIdioma() };
        tabelaPessoas.addRow(rowData: obj);
    }
}
```

Agora, vamos fazer uma pesquisa pelo nome:



## Consulta com filtro

---

- E neste evento, chamaremos o método:

```
private void txtNomeCaretUpdate(javax.swing.event.CaretEvent evt) {  
    preencheTabela();  
}
```

## Consulta com filtro

---

- Funcionou?

## Consulta com filtro

---

- Funcionou?
  - Não né, precisamos limpar a tabela..
    - E como fizemos isso?

## Consulta com filtro

---

- Funcionou?
  - Não né, precisamos limpar a tabela..
    - E como fizemos isso?



## Consulta com filtro

---

```
public void preencheTabela() {
    PessoaDAO pDAO = new PessoaDAO();
    //List<Pessoa> listaPessoas = pDAO.getPessoas();
    List<Pessoa> listaPessoas = pDAO.getPessoasNome(nome: txtNome.getText());
    DefaultTableModel tabelaPessoas = (DefaultTableModel) tbl_Pessoas.getModel();
    //Agora, com a consulta com filtro de nome, para não irmos só adicionando, precisamos limpar a tabela
    // para isso, vamos limpar excluindo todas as linhas dela
    tabelaPessoas.setNumRows(rowCount: 0);
    //Agora, precisamos percorrer a lista de pessoas que foi consultada e adicionar na tabela:
    for(Pessoa p: listaPessoas){
        Object[] obj = new Object[]{
            p.getId(),
            p.getNome(),
            p.getSexo(),
            p.getIdioma()};
        tabelaPessoas.addRow(rowData: obj);
    }
}
```

## Consulta com filtro

---

- Agora, vamos filtrar pelo sexo?
- É com vocês!

Relatório das Pessoas

Nome:

Sexo: ☐ Masculino ☐ Feminino

ID	Nome	Sexo	Idioma
----	------	------	--------

# Consulta com filtro

---

```
public List<Pessoa> getPessoasNome(String nome, String sexo){
    // "SELECT * FROM pessoa WHERE nome LIKE ?"
    String sql = "SELECT * FROM pessoa WHERE nome LIKE ? and sexo LIKE ?";
    try {

        PreparedStatement stmt = conn.prepareStatement(string:sql,i: ResultSet.TYPE_SCROLL_INSENSITIVE,
            i1: ResultSet.CONCUR_UPDATABLE);
        stmt.setString(i: 1 ,"%"+ nome +"%");
        stmt.setString(i: 2 ,"%"+ sexo +"%");
        ResultSet rs = stmt.executeQuery(); //obtenho o retorno da consulta e armazeno no ResultSet
        List<Pessoa> listaPessoas = new ArrayList(); //Preparo uma lista de objetos que vou armazenar a consulta
        //Percorre rs e salvar as informações dentro de um objeto Pessoa e depois adiciona na lista
        while (rs.next()){
            Pessoa p = new Pessoa();
            p.setId(id: rs.getInt(string:"id"));
            p.setNome(nome: rs.getString(string:"nome"));
            p.setSexo(sexo: rs.getString(string:"sexo"));
            p.setIdioma(idioma:rs.getString(string:"idioma"));
            listaPessoas.add(e: p);
        }
        return listaPessoas;
    } catch (SQLException ex) {
        System.out.println("Erro ao consultar todas as pessoas: "+ex.getMessage());
        return null;
    }
}
```

## Consulta com filtro

---

```
public void preencheTabela() {
    String sexo = "";
    PessoaDAO pDAO = new PessoaDAO();
    //List<Pessoa> listaPessoas = pDAO.getPessoas();
    if (rdo_Masculino.isSelected())
        sexo = "M";
    else if (rdo_Feminino.isSelected())
        sexo = "F";
    else
        sexo = "";
    List<Pessoa> listaPessoas = pDAO.getPessoasNome(nome: txtNome.getText(), sexo);
    DefaultTableModel tabelaPessoas = (DefaultTableModel) tbl_Pessoas.getModel();
    //Agora, com a consulta com filtro de nome, para não irmos só adicionando, precisamos limpar a tabela
    // para isso, vamos limpar excluindo todas as alinhas dela
    tabelaPessoas.setNumRows(rowCount: 0);
    //Agora, precisamos percorrer a lista de pessoas que foi consultada e adicionar na tabela:
    for (Pessoa p: listaPessoas) {
        Object[] obj = new Object[] {
            p.getId(),
            p.getNome(),
            p.getSexo(),
            p.getIdioma() };
        tabelaPessoas.addRow(rowData: obj);
    }
}
```

## Consulta com filtro

---

```
private void rdo_MasculinoStateChanged(javax.swing.event.ChangeEvent evt) {  
    preencheTabela();  
}
```

```
private void rdo_FemininoStateChanged(javax.swing.event.ChangeEvent evt) {  
    preencheTabela();  
}
```

# Consulta com filtro

---

Relatório das Pessoas

Nome:

Sexo: ☐ Masculino ☐ Feminino

ID	Nome	Sexo	Idioma
----	------	------	--------

```
private void btn_LimparActionPerformed(java.awt.event.ActionEvent evt) {  
    txtNome.setText("");  
    btnGrp_Sexo.clearSelection();  
}
```

## Atividade

---

- Agora, no sistema que vocês estão trabalhando com alunos e professores, faça:
  - Um menu com os cadastros e telas criadas
  - E crie uma tela para cada tabela, com pelo menos um filtro (de nome), conforme visto em aula