

DATA ACCESS OBJECT (DAO)

Wellington Moreira de Oliveira



INTRODUÇÃO

- Padrão de projeto
- “Middleware” entre a aplicação e o banco de dados
- Separa o acesso aos dados persistidos (BD ou arquivo)
- Gera independência e facilita mudanças
- Deve ser criado um pacote DAO para separar as classes de conexão e acesso aos dados
- Deve ser criar uma classe DAO para cada classe de domínio (do pacote model)
 - Ex.: AlunoDAO, ProfessorDAO, etc



INTRODUÇÃO

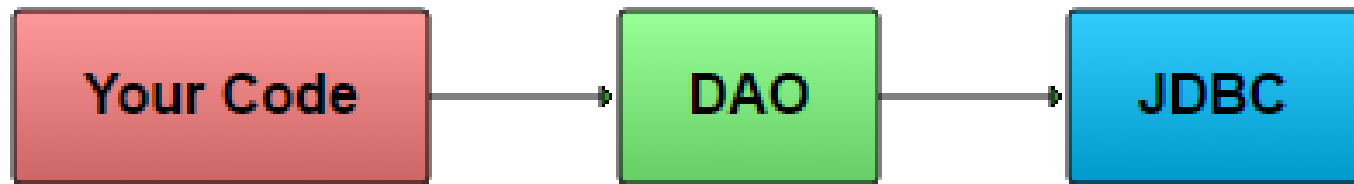
- Definição mais formal

É um padrão de projeto para persistência de dados que permite separar regras de negócio das regras de acesso a banco de dados.

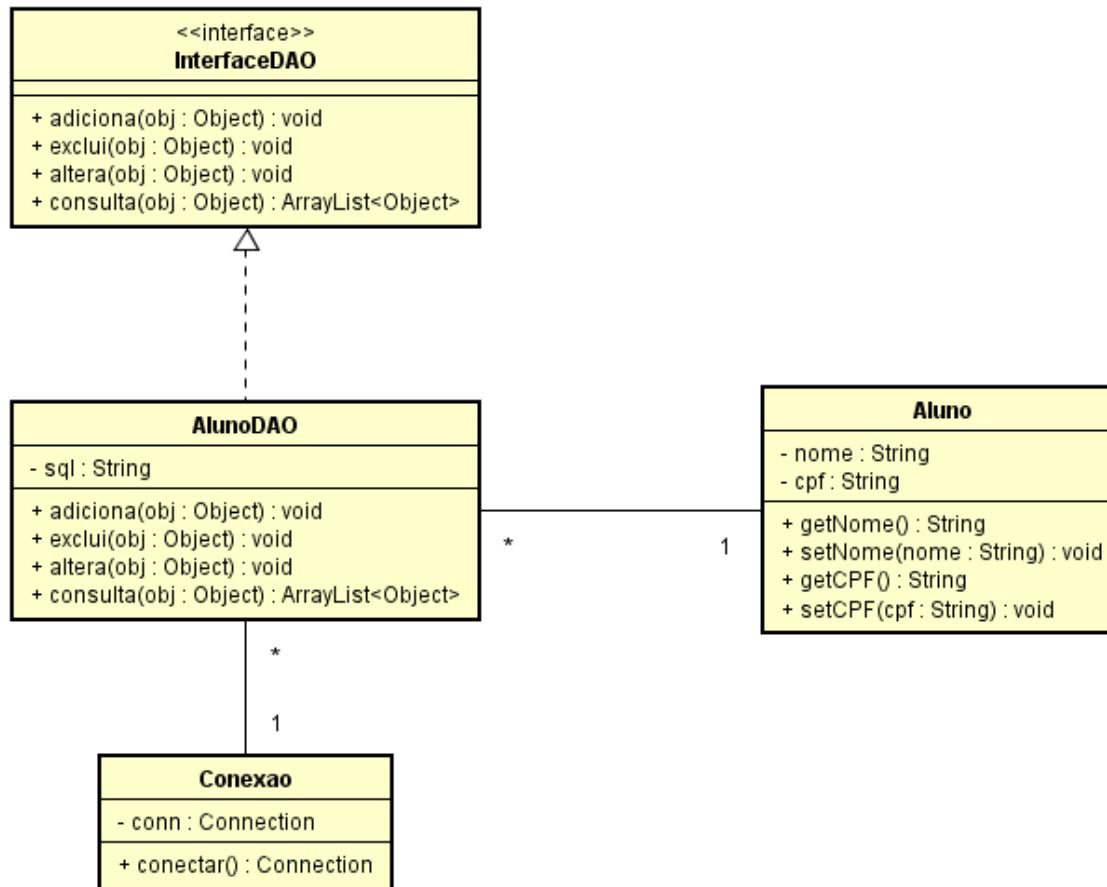
Numa arquitetura MVC, todas as funcionalidades de BD, tais como obter conexões, mapear objetos Java para tipos de dados SQL ou executar comandos SQL, devem ser feitas por classes DAO.



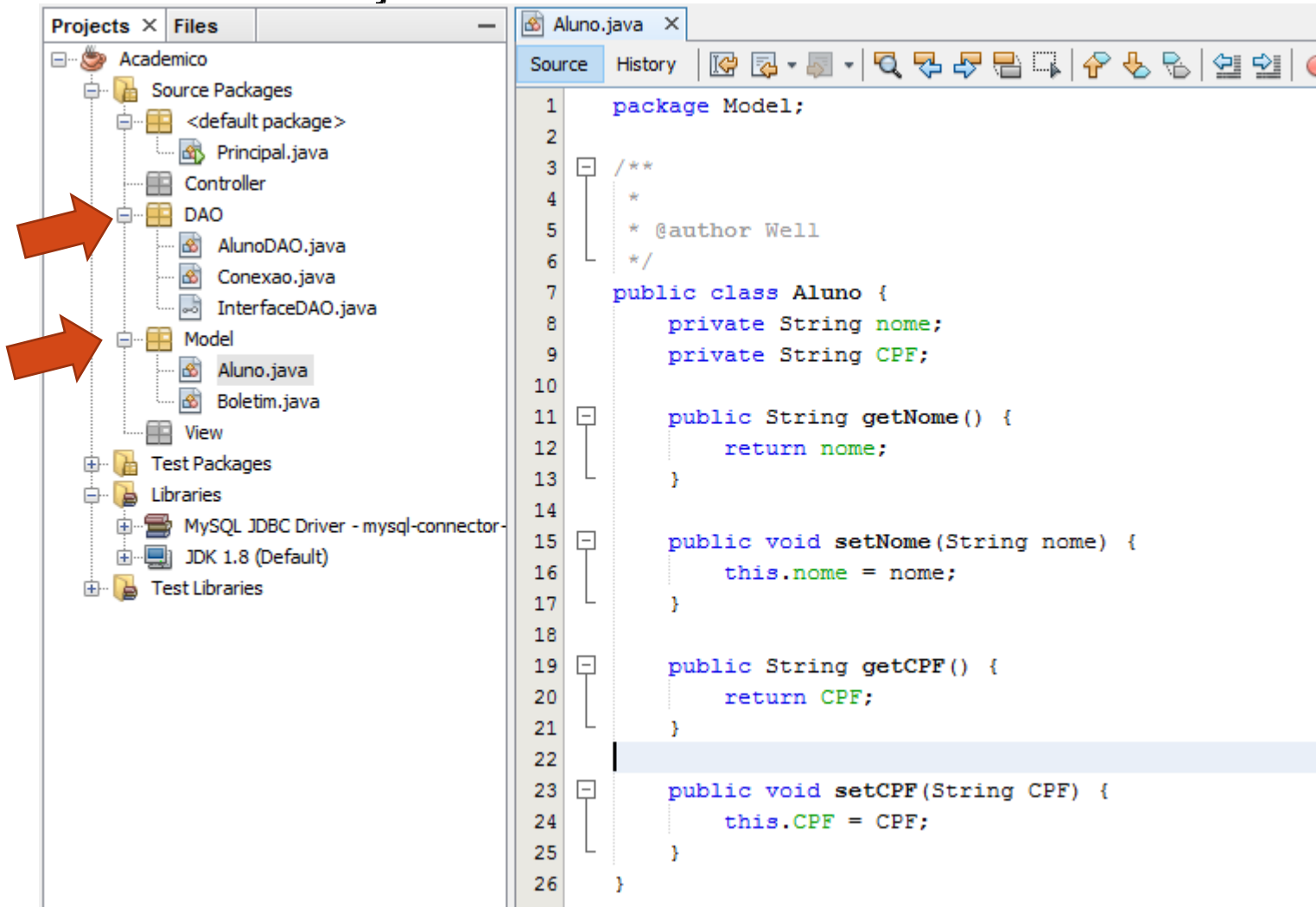
DAO



MODELAGEM (EX.: CLASSE ALUNO)



UMA APLICAÇÃO SIMPLES COM DAO - ALUNO



The screenshot displays an IDE interface with a project named 'Academico'. The 'Projects' tab is active, showing the following structure:

- Academico
 - Source Packages
 - <default package> (containing Principal.java)
 - Controller
 - DAO (containing AlunoDAO.java, Conexao.java, InterfaceDAO.java)
 - Model (containing Aluno.java, Boletim.java)
 - View
 - Test Packages
 - Libraries
 - MySQL JDBC Driver - mysql-connector-
 - JDK 1.8 (Default)
 - Test Libraries

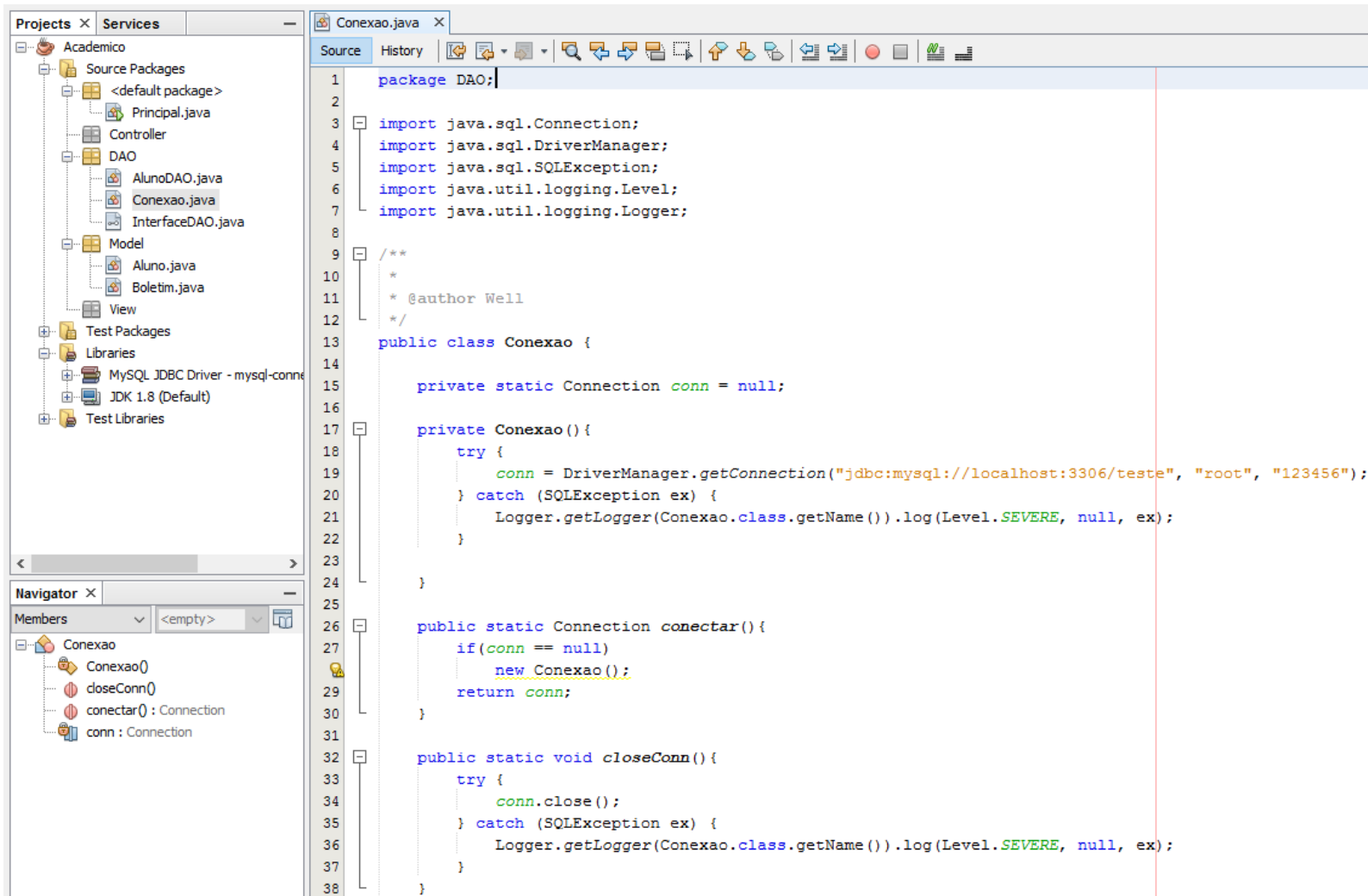
Two red arrows point to the 'DAO' and 'Model' packages in the 'Source Packages' folder.

The 'Aluno.java' file is open in the editor, showing the following code:

```
1 package Model;
2
3 /**
4  *
5  * @author Well
6  */
7 public class Aluno {
8     private String nome;
9     private String CPF;
10
11     public String getNome() {
12         return nome;
13     }
14
15     public void setNome(String nome) {
16         this.nome = nome;
17     }
18
19     public String getCPF() {
20         return CPF;
21     }
22
23     public void setCPF(String CPF) {
24         this.CPF = CPF;
25     }
26 }
```



UMA APLICAÇÃO SIMPLES COM DAO — CONEXÃO SINGLETON



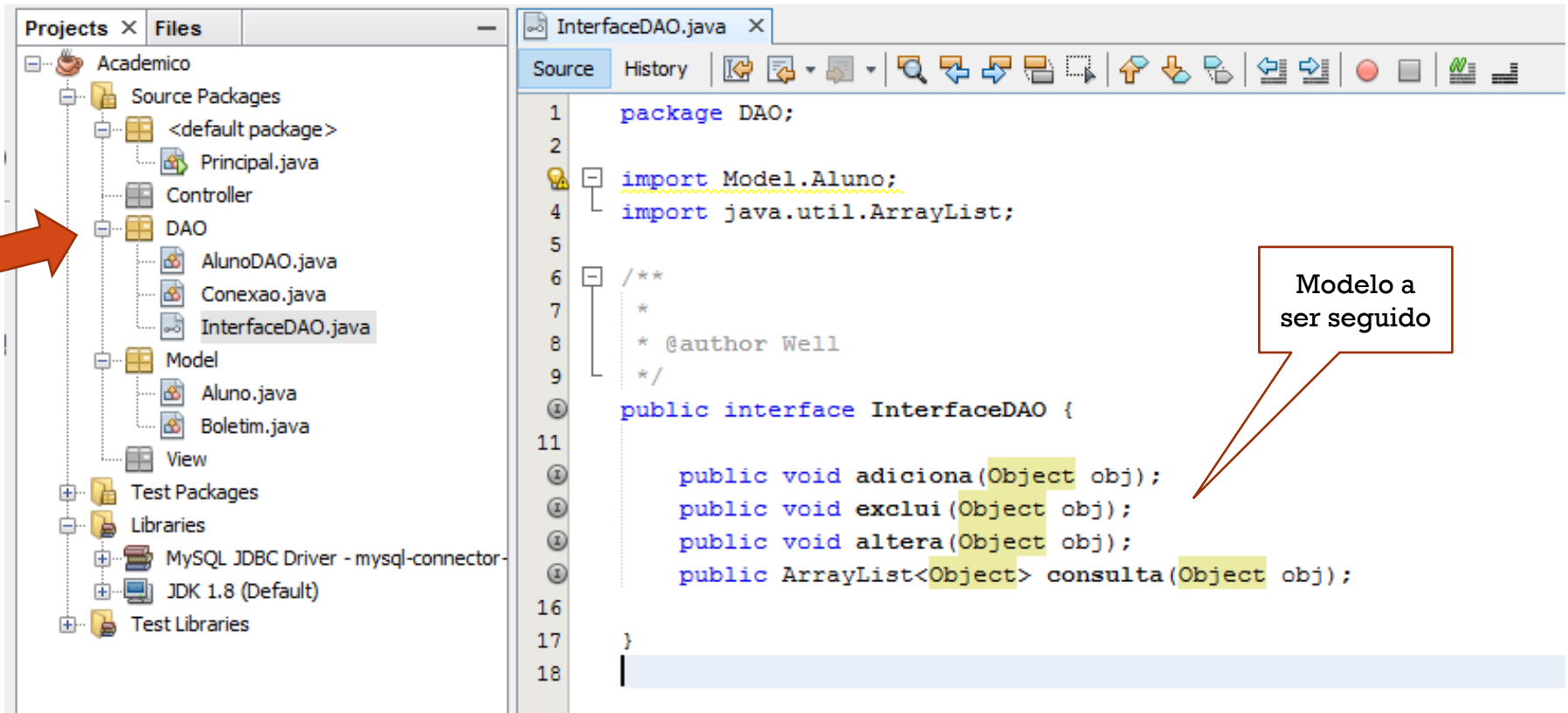
The screenshot displays an IDE with the following components:

- Projects Panel:** Shows a project named 'Academico' with a package structure including 'Source Packages', 'Controller', 'DAO' (containing 'AlunoDAO.java', 'Conexao.java', and 'InterfaceDAO.java'), 'Model' (containing 'Aluno.java' and 'Boletim.java'), and 'View'.
- Navigator Panel:** Shows the 'Members' of the 'Conexao' class, including 'Conexao()', 'closeConn()', 'conectar() : Connection', and 'conn : Connection'.
- Source Editor:** Displays the code for 'Conexao.java'.

```
1 package DAO;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.SQLException;
6 import java.util.logging.Level;
7 import java.util.logging.Logger;
8
9 /**
10  *
11  * @author Well
12  */
13 public class Conexao {
14
15     private static Connection conn = null;
16
17     private Conexao() {
18         try {
19             conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/teste", "root", "123456");
20         } catch (SQLException ex) {
21             Logger.getLogger(Conexao.class.getName()).log(Level.SEVERE, null, ex);
22         }
23     }
24
25     public static Connection conectar() {
26         if (conn == null)
27             new Conexao();
28         return conn;
29     }
30
31     public static void closeConn() {
32         try {
33             conn.close();
34         } catch (SQLException ex) {
35             Logger.getLogger(Conexao.class.getName()).log(Level.SEVERE, null, ex);
36         }
37     }
38 }
```



UMA APLICAÇÃO SIMPLES COM DAO - INTERFACE



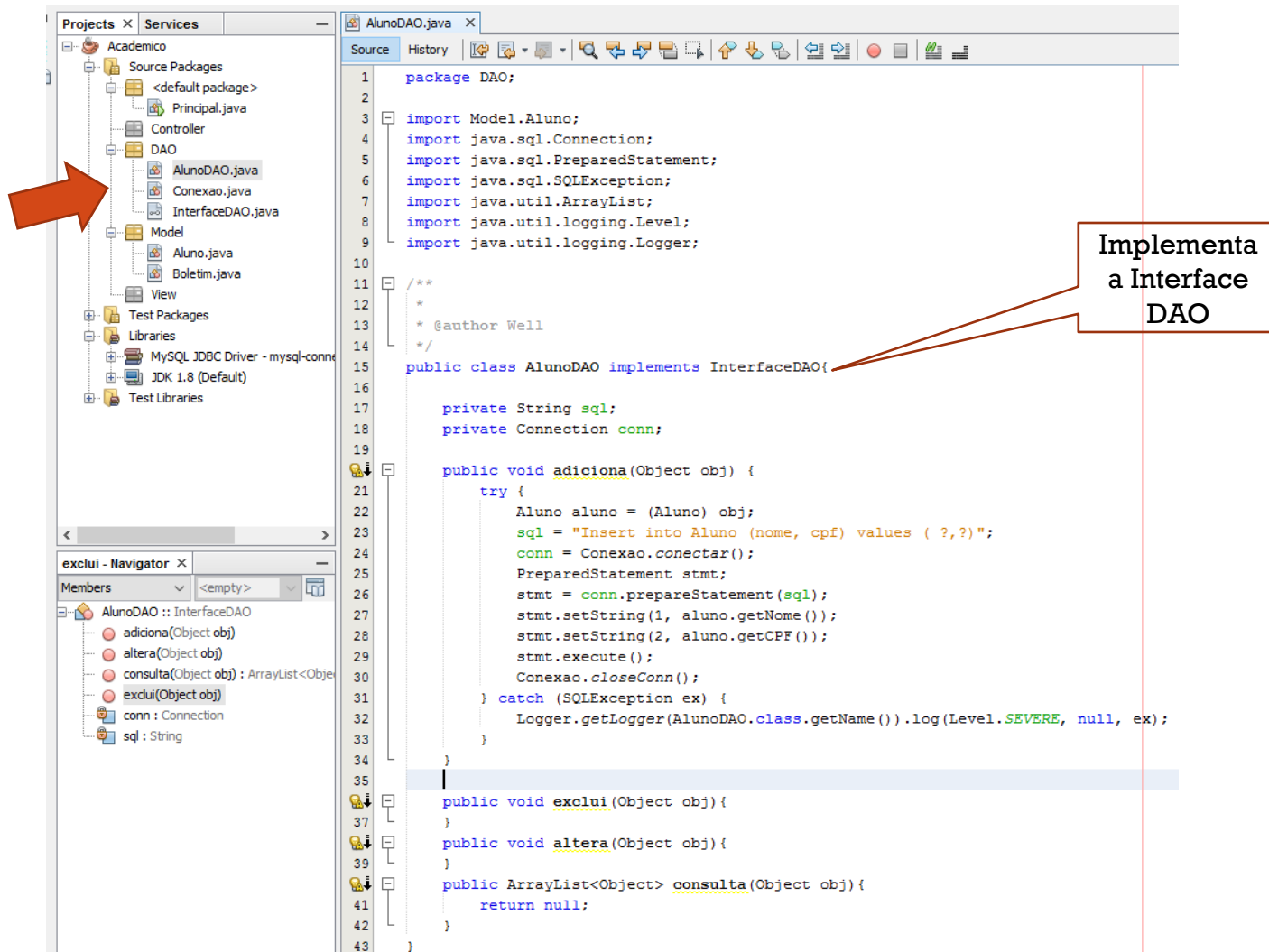
The screenshot displays an IDE with two main panels. The left panel, titled 'Projects' and 'Files', shows a project named 'Academico'. Under 'Source Packages', there is a 'DAO' package containing 'AlunoDAO.java', 'Conexao.java', and 'InterfaceDAO.java'. An orange arrow points to the 'DAO' package. Below 'Source Packages' are 'Test Packages', 'Libraries' (including 'MySQL JDBC Driver - mysql-connector-' and 'JDK 1.8 (Default)'), and 'Test Libraries'. The right panel shows the 'InterfaceDAO.java' file with the following code:

```
1 package DAO;
2
3 import Model.Aluno;
4 import java.util.ArrayList;
5
6 /**
7  *
8  * @author Well
9  */
10 public interface InterfaceDAO {
11
12     public void adiciona(Object obj);
13     public void exclui(Object obj);
14     public void altera(Object obj);
15     public ArrayList<Object> consulta(Object obj);
16
17 }
18
```

A callout box with a red border and a pointer to the 'Object' parameter in the 'consulta' method contains the text: 'Modelo a ser seguido'.



UMA APLICAÇÃO SIMPLES COM DAO - ALUNODAO



The screenshot displays an IDE with the following components:

- Project Explorer:** Shows a project named 'Academico' with a package structure including 'Source Packages', 'Controller', 'DAO', 'Model', 'Test Packages', and 'Libraries'. The 'DAO' package contains 'AlunoDAO.java', 'Conexao.java', and 'InterfaceDAO.java'. An orange arrow points to the 'DAO' package.
- Source Editor:** Displays the code for 'AlunoDAO.java'. The code implements the 'InterfaceDAO' and includes imports for 'Model.Aluno', 'java.sql.Connection', 'java.sql.PreparedStatement', 'java.sql.SQLException', 'java.util.ArrayList', 'java.util.logging.Level', and 'java.util.logging.Logger'. The methods implemented are 'adiciona', 'exclui', 'altera', and 'consulta'.
- Members View:** Shows the members of the 'AlunoDAO' class, including the methods 'adiciona', 'altera', 'consulta', and 'exclui', and the variables 'conn' and 'sql'.

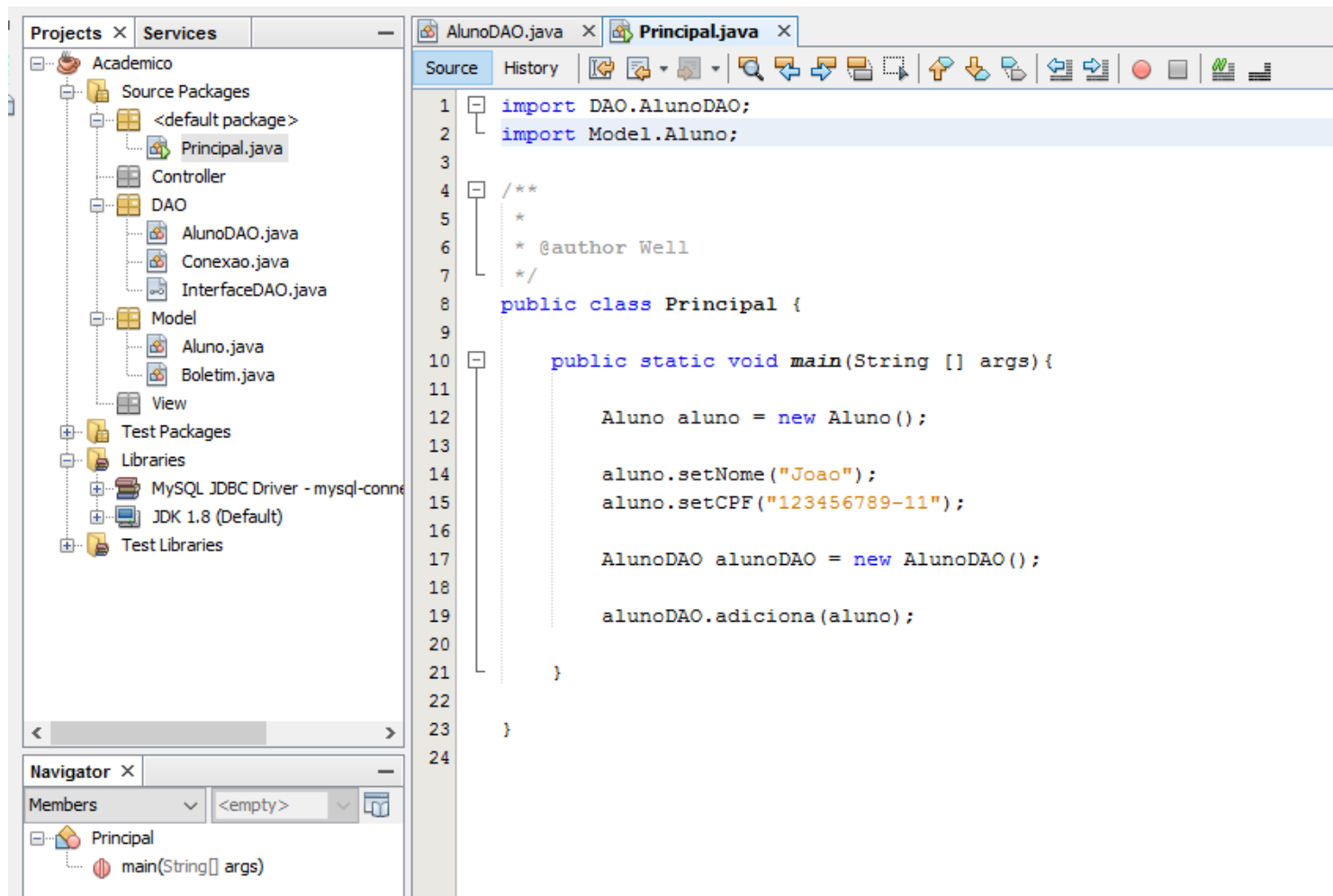
Code Snippet:

```
1 package DAO;
2
3 import Model.Aluno;
4 import java.sql.Connection;
5 import java.sql.PreparedStatement;
6 import java.sql.SQLException;
7 import java.util.ArrayList;
8 import java.util.logging.Level;
9 import java.util.logging.Logger;
10
11 /**
12  *
13  * @author Well
14  */
15 public class AlunoDAO implements InterfaceDAO{
16
17     private String sql;
18     private Connection conn;
19
20     public void adiciona(Object obj) {
21         try {
22             Aluno aluno = (Aluno) obj;
23             sql = "Insert into Aluno (nome, cpf) values ( ?,?)";
24             conn = Conexao.conectar();
25             PreparedStatement stmt;
26             stmt = conn.prepareStatement(sql);
27             stmt.setString(1, aluno.getNome());
28             stmt.setString(2, aluno.getCPF());
29             stmt.execute();
30             Conexao.closeConn();
31         } catch (SQLException ex) {
32             Logger.getLogger(AlunoDAO.class.getName()).log(Level.SEVERE, null, ex);
33         }
34     }
35
36     public void exclui(Object obj){
37     }
38
39     public void altera(Object obj){
40     }
41
42     public ArrayList<Object> consulta(Object obj){
43         return null;
44     }
45 }
```

Callout: Implementa a Interface DAO



UMA APLICAÇÃO SIMPLES COM DAO - PRINCIPAL



REFERÊNCIAS

- <http://tutorials.jenkov.com/java-persistence/dao-design-pattern.html>
- <https://www.ime.usp.br/~kon/MAC5715/PLoP/2006/refact/RoupaSujaSeLavaEmCasa-ref.pdf>
- <https://prezi.com/k8pvincplplu/padrao-de-projeto-dao/>

