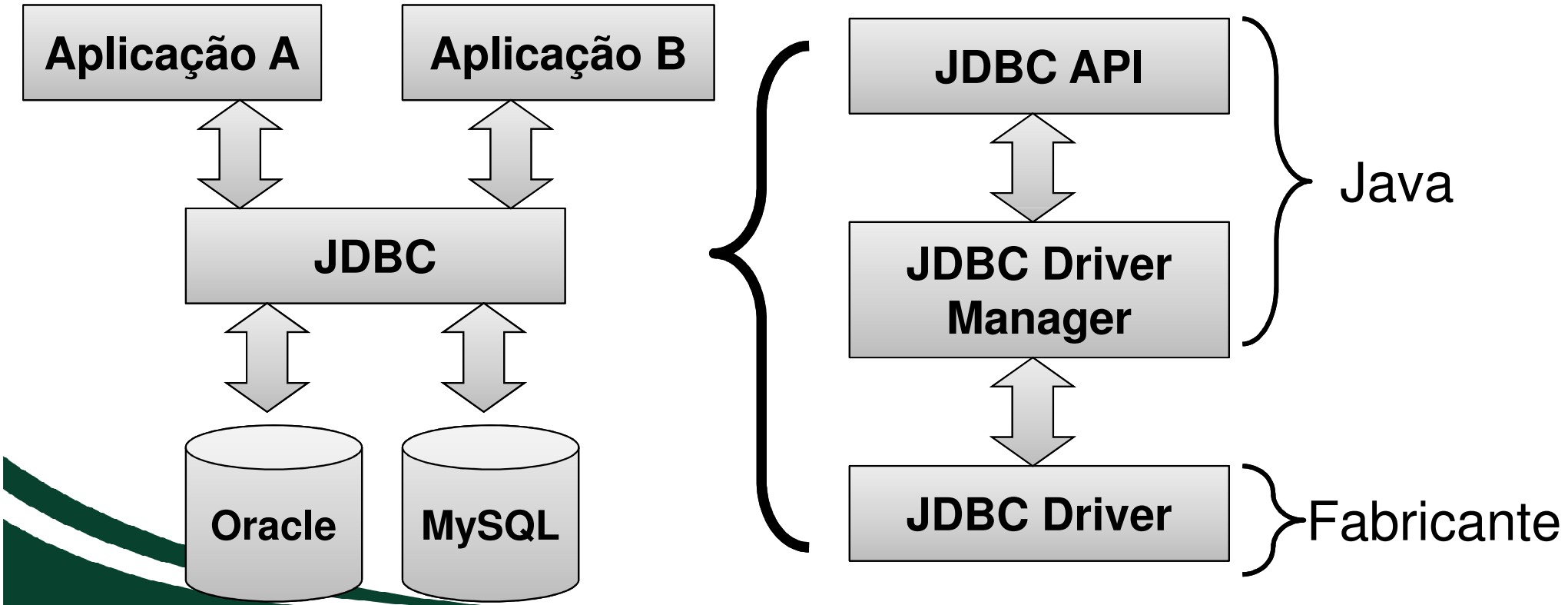
# Java DataBase Connectivity (JDBC)

Fernando dos Santos
fernando.santos@udesc.br

UDESC
Ibirama

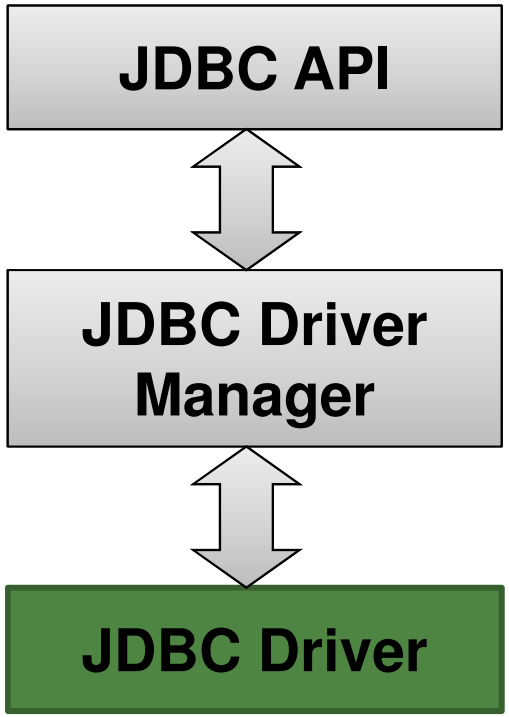
# Java DataBase Connectivity (JDBC)

- JDBC é uma API Java para acesso à banco de dados.
- O JDBC define um **conjunto de classes e métodos** para estabelecer conexão com um banco e executar comandos SQL.
- Cada fabricante (Oracle, MySQL, etc) fornece uma implementação específica para uso com o banco – **Driver JDBC**.

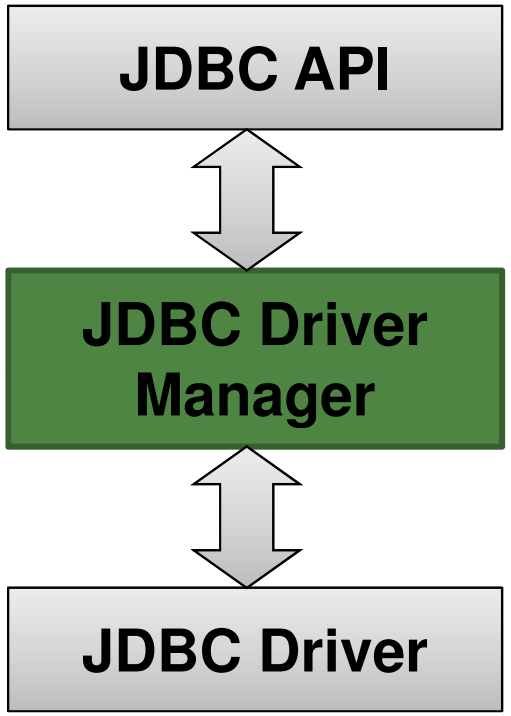


# JDBC Driver

- É a implementação da JDBC API, disponibilizada pelo fabricante do banco de dados, para conectar no banco e executar SQL.
- É disponibilizado em um arquivo **jar**, que deve ser incluido como uma biblioteca na aplicação.
- O driver específico do banco é carregado na aplicação, para depois ser utilizado.
- Comando para carregamento do driver:
  - Class.forName("classe do driver")
- Carregamento do driver MySQL
  - Class.forName("com.mysql.jdbc.Driver")



# JDBC Driver Manager



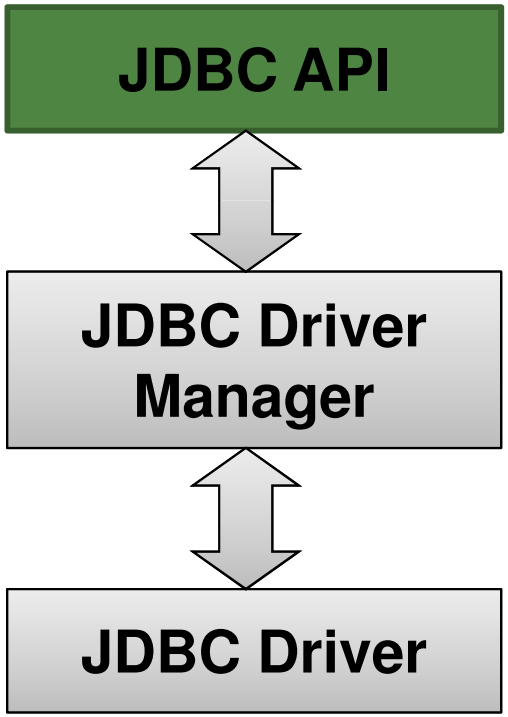

- É a classe Java que estabelece conexão com o banco de dados
  - java.sql.DriverManager
  - java.sql.Connection

- Estabelecendo conexão:

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

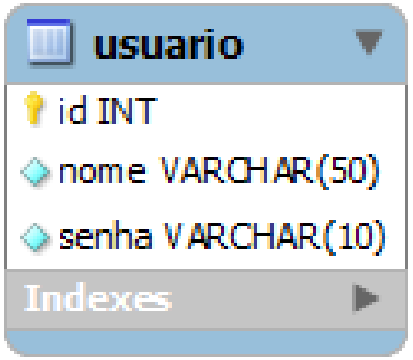
- **usuario**: usuário de acesso ao banco
- **senha**: senha de acesso ao banco
- **url**: identifica o **protocolo**, **banco**, **host do servidor**, **porta** e **esquema**.
  - **jdbc:mysql://localhost:3306/sistema_vendas**

# JDBC API




- Disponibiliza classes para interação com o banco:
  - envio de comandos SQL: Statement e PreparedStatement
  - obtenção de resultados: ResultSet
- Todas as classes estão no pacote **java.sql**

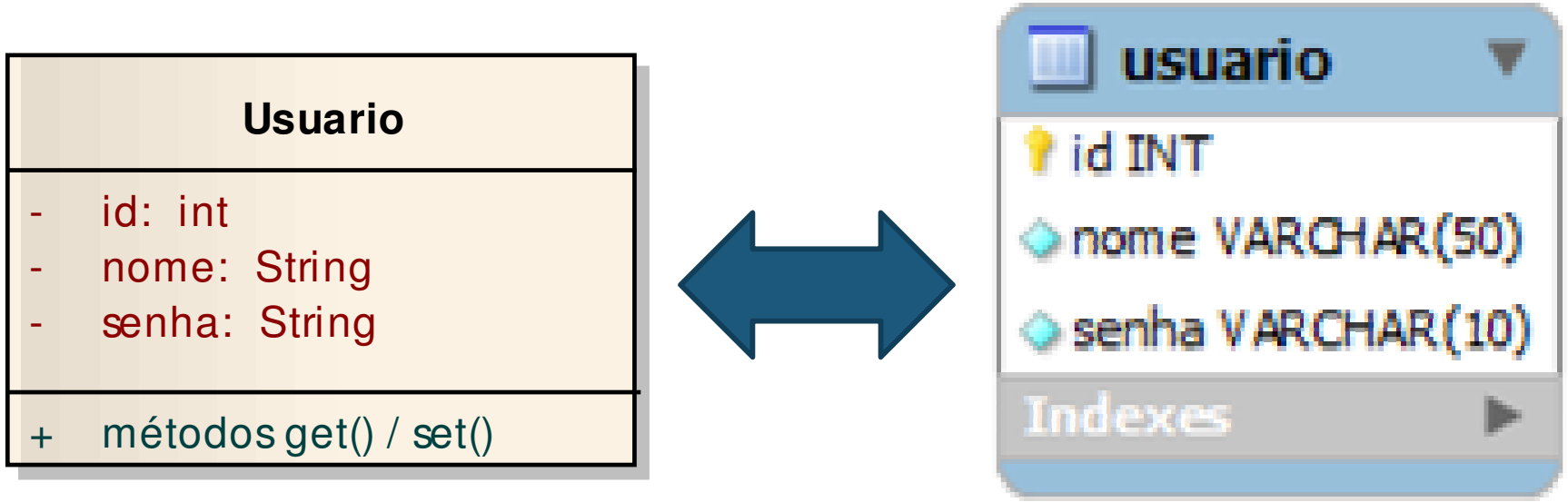
# JDBC API - Exemplo




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

Connection conexao =
DriverManager.getConnection( "jdbc:mysql://localhost:3306/sistema_vendas", "root", "root123");

Statement comando = conexao.createStatement();

ResultSet resultados = comando.executeQuery("select id, nome, senha from usuario");

// comando.execute("delete from usuario where id = 1"); // para comandos sem resultados

while (resultados.next()) {
    int id = resultados.getInt("id");
    String nome = resultados.getString("nome");
   String senha = resultados.getString("senha");
}
comando.close();
conexao.close();
```

# JDBC e o paradigma orientado a objetos

- As aplicações são desenvolvidas usando orientação a objetos
  - são modeladas classes e seus relacionamentos.
- Como mapear classes para tabelas, e os atributos para colunas?




- Escreva um método Java para buscar uma lista de usuários.
  - O método deve usar JDBC para fazer um select no banco;
  - Para cada registro, deve criar um objeto Usuário;
  - Retornar a lista de objetos Usuário

# Bibliografia


- DEITEL, Paul J; DEITEL, Harvey M. **Java**: como programar.8. ed. São Paulo: Pearson, 2010. xxix, 1144 p, il.
  - Capítulo 28

- **JDBC Overview**
  - http://www.oracle.com/technetwork/java/overview-141217.html