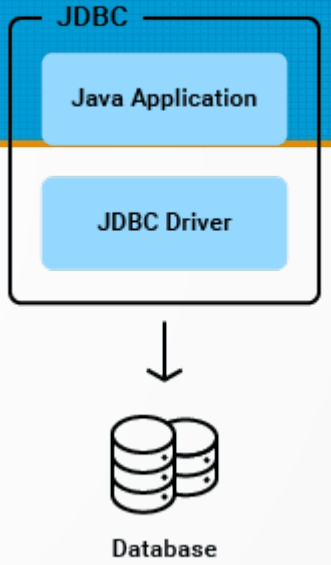
DESENVOLVIMENTO DE SOFTWARE PARA A WEB 1

Módulo 4
Java Database Connectivity (JDBC)

Delano Medeiros Beder
delano@dc.ufscar.br

Noção Geral



- A aplicação chama a biblioteca JDBC
- A biblioteca carrega o driver que “entende” o SGDB
- Após, a aplicação pode se conectar e enviar requisições ao SGBD
- Pacote principal: **java.sql**

JDBC

- Java Database Connectivity
- Padrão de acesso a BDs relacionais através de Java:
  - API comum
  - Os fabricantes de drivers JDBC implementam aspectos específicos
- Qualquer aplicação Java pode acessar um SGBD através do JDBC
- JDBC é semelhante ao ODBC, mas é escrito em Java.

JDBC

- 5 passos básicos:
  - Registrar o driver na aplicação
  - Conectar no SGBD
  - Executar sentenças SQL e procedures
  - Processar o resultado recebido
  - Fechar a conexão
- Principais classes da API:
  - **DriverManager, Connection, Statement, ResultSet**
- Referência: http://java.sun.com/javase/6/docs/api/java/sql/package-summary.html

JDBC – Passos Básicos

- Registro do driver: O driver é registrado automaticamente quando a classe é carregada na aplicação
  - // PostgreSQL
    Class.forName("org.postgresql.Driver");
  - // MySQL
    Class.forName("com.mysql.jdbc.Driver");

    - Inicializador estático que registra o driver

JDBC – Passos Básicos

- Conexão com o SGBD: Após o registro do driver, precisamos fornecer informações ao DriverManager para a conexão
  - **Connection con = DriverManager.getConnection(url, login, senha);**
    - url: URL de conexão JDBC
      - jdbc:postgresql://localhost: 5432/cursodsw1
      - jdbc:mysql://localhost:3306/cursodsw1
    - login: usuário com direitos de acesso ao banco de dados;
    - senha: senha para autenticação.
    - Sintaxe geral de urls: “jdbc:<subprotocol>://<server>:<port>/<database>”

JDBC – Passos Básicos

- Execução de sentenças SQL
  - Para a execução de sentenças devemos criar, por exemplo, um *Statement* e obter o resultado através de um *ResultSet*
    - **Statement stmt = con.createStatement();**
    - **ResultSet rs = stmt.executeQuery(“select * from db.empregados”);**
  - Neste caso, o resultado é armazenado num ResultSet e pode ser percorrido com métodos definidos nesta classe

JDBC – Passos Básicos

```
public class AcessaBD {
  public static void main(String[] args) {
    try {
      Class.forName("org.apache.derby.jdbc.ClientDriver");
      Connection con = (Connection) DriverManager.getConnection(""
          + "jdbc:derby://localhost:1527/Livraria", "root", "root");
      Statement stmt = con.createStatement();
      ResultSet rs = stmt.executeQuery("select * from Livro l");
      while (rs.next()) {
        System.out.print(rs.getString("Titulo"));
        System.out.print(", " + rs.getString("Autor"));
        System.out.print(", " + rs.getInt("Ano"));
        System.out.println(" (R$ " + rs.getFloat("Preco") + ")");
      }
      stmt.close(); con.close();
    } catch (ClassNotFoundException e) {
      System.out.println("A classe do driver de conexão não foi encontrada!");
    } catch (SQLException e) {
      System.out.println("O comando SQL não pode ser executado!");
    }
  }
}
```

Uma exemplo de acesso JDBC

Demonstração 1

Alternativas para Criação de Sentenças

- **Statement**: executa consultas simples, sem parâmetros
- **PreparedStatement**: executa consultas pré-compiladas com ou sem parâmetros

```
PreparedStatement modificaTabela = con.prepareStatement ("UPDATE TABELA
   SET CAMPO1 = ? WHERE CAMPO2 LIKE ? ");
modificaTabela.setInt(1, 75);
modificaTabela.setString(2, “Sirius”);
modificaTabela.executeUpdate();
```

- **CallableStatement**: executa chamadas à *stored procedures*

```
CallableStatement cs = con.prepareCall ("{call NOME_PROC}");
ResultSet rs = cs.executeQuery()
```

FIM