JDBC
API Java Database Connectivity

Professor Vinícius Costa

JDBC

- É a API Java para se conectar e interagir com o banco de dados
- Os principais sistemas de gerenciamento de banco de dados fornecem drives JDBC
- Utilizando JDBC é fácil trocar o banco de dados de uma aplicação

Iniciando uma Conexão

- Connection com = DriverManager.getConnection(url, usuário, senha);
- O método estático getConnection da classe DriverManager estabelece a conexão com o banco. Recebe como parâmetro a url informando qual é o banco de dados, o nome do usuário e sua senha. Retorna uma conexão
- DriverManager tenta escolher o driver correto, se ele estiver registrado
- Se não for possível escolher o drive correto é necessário utilizar o método estático forName da classe Class para indicar qual é o drive. Isso deve acontecer antes de criar uma conexão

Exemplo com MySql

```
Class.forName("com.mysql.cj.jdbc.Driver");
Connection com =
    DriverManager.getConnection(
    "jdbc:mysql://localhost:3306/nomeDoBanco?
    serverTimezone=UTC","usuario","senha");
```

Statement

- O objeto do tipo Statement serve para enviar instruções sql ao banco de dados
- Para criar um objeto do tipo Statement deve se utilizar o método createStatement da conexão
- Exemplo
  - Statement statement = con.createStatement();

Executando uma Instrução SQL

- executeUpdate()
  - Para inserir, deletar ou atualizar dados no banco de dados uma das possibilidades é utilizar o método executeUpdate do objeto do tipo Statement
  - Recebe como parâmetro um comando sql e retorna a quantidade de linhas afetadas
- executeQuery()
  - Uma das possiblidades para selecionar dados no banco
  - Recebe como parâmetro um comando sql e retorna um objeto do tipo ResultSet

Executando uma Instrução SQL

- Exemplos
  - ResultSet result = statement.executeQuery("select * from tabela");
  - Int quant = statement. executeUpdate("update tabela set coluna1=valor1, coluna2=valor2 where coluna=algo");

ResultSet

- O método executeQuery do objeto Statement retorna um objeto ResultSet
  - ResultSet result = statement.executeQuery("select * from tabela");
- ResultSet é um conjunto de dados que representa uma tabela
- O ResultSet tem um cursor que aponta para a linha de dados atual
- Inicialmente o cursor está posicionado antes da primeira linha
- Para ir para a próxima linha utilize o método next() do objeto ResultSet
- Quando as linhas acabarem o método next() retornará 0 (zero)

ResultSet

- O ResultSet tem vários métodos get para pegar os dados das colunas da linha atual
- Os métodos get podem receber o nome da coluna por parâmetro ou um inteiro que indica a posição da coluna
- Exemplo
  - getInt()
  - getString()
  - getDate()

ResultSet

- Exemplo

```
while(result.next())
{
    for(int i=1;i<=quantColunas;i++)
        System.out.println(result.getString(i));
}
```

- Observe que a primeira coluna está no índice 1.

Meta Dados do ResultSet

- O método getMetaData() do objeto ResultSet retorna um objeto do tipo ResultSetMetaData
- O ResultSetMetaData fornece informações sobre o tipo e propriedades das colunas em um ResultSet
- Exemplo
  - ResultSetMetaData meta=result.getMetaData();
  - Int quantColunas = meta.getColumnCount();

Fechando Tudo

- result.close();//um statement pode ter apenas um resultset por vez, se você abrir outro resultset o anterior será fechado. Fechar um statement irá descartar o resultset
- statement.close();
- connection.close();

PreparedStatements

- PreparedStatments é mais eficiente pois as instruções sql são compiladas
- PreparedStatements é mais flexível para realizar instruções sql pois pode receber parâmetros que serão passados ao comando sql
- Para informar os parametros utilize os métodos set (setString, setInt, etc)

Preparando uma Declaração

- PreparedStatement prestatement = con.prepareStatement("select * from tabela where coluna1=**?** And coluna2=**?**");
- prestatement.setString(1,"valorDaColuna1");
- prestatement.setString(2,"valorDaColuna2");

- Os métodos set substituem os valores pelo ponto de interrogação no comando sql, sendo o primeiro ponto de interrogação 1 o segundo 2, etc.
- Por fim utilize um dos métodos executeQuery() ou executeUpdate()

- ResultSet result = Prestatement.executeQuery();