Select em uma tabela do banco oracle

Esta classe foi enviada pelo usuário Alan para a lista soujava e modificada por mim.
Incluí diversos recursos que os novos programadores devem ter em mente, tal como, sempre fechar os statements e connections. ;-)

Neste exemplo o nosso banco se chamará "portaljava".
Username:portaljava
Senha:pj123

******* Código *******

```
import java.sql.*;
import java.util.Vector;
import oracle.jdbc.driver.*;

public class oracle {

	private	String url = "jdbc:oracle:thin:@192.168.1.207:1521:portaljava";
	private	String user = "portaljava";
	private	String pass = "pj123";


	/*
	Método que retorna um objeto Connection
	para a utilização somente dentro desta classe.
	*/
	private Connection getMyConnection() {
		Connection conn = null;
		try {
			DriverManager.registerDriver(new OracleDriver());
			conn = DriverManager.getConnection(url, user, pass);
		} catch (Exception e) {
			System.out.println("Erro ao tentar se conectar ao banco");
		}

		return conn;
	}

	/*
	Este método retorna um Array contendo os nomes dos usuários
	do portal.
	*/
	public Vector selectTable()
		throws SQLException, Exception{
		Connection conn = getMyConnection();

		Vector vtUsrs = new Vector();

		//Testa para ver se a conexão está nula.
```

```
                if (conn == null)
                        throw new Exception("Conexão está nula");
                Statement stmt = null;

                        try {
                                stmt = conn.createStatement();
                                ResultSet rs =
                                        stmt.executeQuery(
                                                "SELECT * FROM USUÁRIOS
WHERE NOME = 'DALTON'");

                                while (rs.next()) {
                                        /*
                                        O objeto resultset, permite retornar
valores primitivos conforme os

                                        mesmos encontram-se no banco.
                                        Aqui usaremos números para recuperar
os campos conforme

                                        a ordem que eles se encontram na tabela
do banco, por exemplo:

                                        A tabela do banco está assim
                                        ID | Nome | Job
                                        Se usarmos getString(2), estaremos
recuperando a coluna nome.

                                        O mesmo aconteceria se utilizássemos

getString("Nome")

                                        */
                                        int empno = rs.getInt(1); //Retornando

um valor inteiro

                                        String nome = rs.getString(2); //

Retornando uma String

                                        String job = rs.getString(3); //

Retornando uma String

                                        int mgr = rs.getInt(4); //Retornando um

valor inteiro

                                        Date data = rs.getDate(5); //Retornando

um campo Date

                                        long sal = rs.getLong(6); //Retornando

um valor long

                                        long comm = rs.getLong(7);

//Retornando um valor long

                                        int dept = rs.getInt(8);  //Retornando um

valor inteiro


                                        //Imprimindo no console os dados obtidos
                                        System.out.print(empno + " ");
                                        System.out.print(nome + " ");
                                        System.out.print(job + " ");
                                        System.out.print(mgr + " ");
                                        System.out.print(data + " ");
                                        System.out.print(sal + " ");
                                        System.out.print(comm + " ");
                                        System.out.println(dept);
```

```
                                        //Utilizando o método add para incluir a
String nome no vector
                                        vtUsrs.add(nome);
                                }

                        } catch (SQLException e) {
                                //Aconteceram erros na execução do sql
                                System.out.println("Erro = " + e.getMessage());
                } finally {
                        try {
                                /* Geralmente usamos o o finally para fechar
statements e connections.
                                O finally sempre será chamado, mesmo que um
return venha ser invocado
                                antes do escopo.
                                ps: É altamente recomendado que em todas as
requisição que forem utilizado
                                os objetos Statement e Connection, sejam
fechados através do finally.
                                */
                                if(statement != null) statement.close();
                                if(connection != null) connection.close();
                        } catch (SQLException e) {
                                throw new Exception(e.toString());
                        }
                }
                //Retornando o Vector
                return vtUsrs;
        }
}
```

[]'s

Dalton
dalton@portaljava.com