SERVLETS E JSP (JEE) - JSP - Banco de Dados (JDBC)

Diogo Cezar Teixeira Batista

diogo@diogocezar.com.br

http://www.diogocezar.com

Universidade Tecnológica Federal do Paraná - UTFPR

Cornélio Procópio - 2012

JSP Banco de Dados (JDBC) I

- *Java Database Connectivity* ou JDBC é um conjunto de classes e interfaces (API) escritas em Java que fazem o envio de instruções SQL para qualquer banco de dados relacional;
- Api de baixo nível e base para api's de alto nível;
- Amplia o que você pode fazer com Java;
- Possibilita o uso de bancos de dados já instalados;
- Para cada banco de dados há um driver JDBC;

JSP Banco de Dados (JDBC) II

- exemplo separado em arquivos:
  - bd:
    - ConnectionBase - É a base para a conexão ao banco de dados;
    - DataBase - Herda a ConnectionBase e configura o banco de dados baseado nas constantes definidas na classe ConfigSite;
  - config:
    - ConfigSite - Seta as configurações do banco e do site em questão;
  - errors:
    - Error - Seta as mensagens de erro do sistema;
  - util:
    - Library - Biblioteca com funções do sistema;

Exemplos I

Código 1: Exemplo ConnectionBase

```
public ConnectionBase(String url, String bd, String user, String key,
String path, String driver, int tipo, int concorrencia){
...
try {
  Class.forName(getDriver());
  setConnection(DriverManager.getConnection(getUrl(), getUser(),
  getKey()));
  this.connection.setAutoCommit(isAutoCommit());
  setStatement(this.connection.createStatement(getTipo(),
  getConcorrencia()));
  if(!Library.empty(getPath())){
      this.statement.execute("SET SEARCH_PATH TO " + getPath());
  }
  setConectado(true);
      } catch (Exception ex) {
  Error.addError(Error.E_CONNECTION + ex);
  setConectado(false);
      }
  }
```

Exemplos II

Código 2: Exemplo DataBase

```
  public class DataBase extends ConnectionBase{
    public DataBase(){
      super(ConfigSite.url, ConfigSite.bd,
ConfigSite.user, ConfigSite.key, ConfigSite.path,
ConfigSite.driver, ConfigSite.tipo, ConfigSite.concorrencia);
    }
    public ResultSet query(String sql){}
    public int write(String sql){}
    public void close(){}
    public int insert(String tabela, String campos[], String
valores[]){}
    public int update(String tabela, String condicao, String campos[],
    String valores[]){}
    public int delete(String tabela, String condicao){}
    public boolean conectado(){}
    public void commit(){}
  }
```

Exemplos III

Código 3: Exemplo Error

```
public abstract class ConfigSite {
  public static final String TITULO_SISTEMA = "Alunos Online";
  public static String url = "jdbc:postgresql://localhost:5432/";
  public static String bd = "alunos";
  public static String user = "postgres";
  public static String key = "*****";
  public static String path = "public";
  public static String driver = "org.postgresql.Driver";
  public static boolean autoCommit = false;
  public static int tipo = ResultSet.TYPE_SCROLL_SENSITIVE;
  public static int concorrencia = ResultSet.CONCUR_UPDATABLE;
  private ConfigSite(){
  }
}
```

Exemplos IV

Código 4: Exemplo ConfigSite

```
  public abstract class Error {
    public static final String E_CONNECTION = "Erro ao se conectar com
o banco de dados: ";
    public static final String E_QUERY = "Erro ao executar uma
consulta no banco de dados: ";
    public static final String E_WRITE = "Erro ao executar uma escrita
no banco de dados: ";
    public static final String E_CLOSE = "Erro ao fechar a conexão com
o banco de dados: ";
    public static final String E_COMMIT = "Erro ao efetuar o commit: "↩
        ;
    private static String errors = "";
    public static void addError(String error){
    setErrors(errors+"<br>"+error);
    }
    public static void clearError(){
    setErrors("");
    }
    ...
  }
```

Exemplos V

Código 5: Exemplo Library

```
package util;
public abstract class Library {
    public static boolean empty(String vazia){
if(vazia.compareTo("") == 0){
    return true;
}
else{
    return false;
}
    }
    private Library() {
    }
}
```

Atividades

- Utilize esta estrutura de arquivos para fazer um crud do objeto *Produto*:
  - id (int);
  - Nome (string);
  - Categoria (string);
  - Preço (Float);
  - Promoção (Boolean);
  - Desconto (Float);