8 Integração com Bancos de Dados

Helder da Rocha (helder@acm.org)
www.argonavis.com.br

Objetivos

- *Este módulo apresenta estratégias para conectar servlets com a camada de dados usando um DAO - Data Access Object*
- *Um DAO é a alternativa ideal porque isola o servlet do conhecimento dos detalhes da implementação de banco de dados usada*
  - *Pode-se trocar a implementação por outra (usando XML, arquivos, outros bancos, etc.)*
- *Para escrever um DAO para um banco relacional, será necessário usar a interface JDBC, mas o servlet não terá nenhuma linha de código JDBC*

Acesso a bancos de dados

- *Servlets são aplicações Java e, como qualquer outra aplicação Java, podem usar JDBC e integrar-se com um banco de dados relacional*
- *Pode-se usar java.sql.DriverManager e obter a conexão da forma tradicional*

```
Class.forName("nome.do.Driver");
Connection con =
     DriverManager.getConnection("url", "nm", "ps");
```

- *Pode-se obter as conexões de um pool de conexões através de javax.sql.DataSource via JNDI (J2EE)*

```
DataSource ds = (DataSource)ctx.lookup("jbdc/Banco");
Connection con = ds.getConnection();
```

Servlet que faz um SELECT em banco

- *Para tornar um servlet capaz de acessar bancos de dados, basta incluir código JDBC dentro dele*

```
import java.sql.*; // ... outros pacotes
public class JDBCServlet extends HttpServlet {
   String jdbcURL;   String driverClass;
   String nome = ""; String senha = "";  Connection con;
   public void init() {
      // ... obter os dados via initParameter()
      try {
        Class.forName(driverClass);
        con = DriverManager.getConnection(jdbcURL, nome, senha);
     } catch (Exception e) { throw new UnavailableException(e); }
   }
   public void doGet(... request, ... response) ... {
      try {  // ... codigo de inicializacao do response
        Statement stmt = con.createStatement();
        ResultSet rs = stmt.executeQuery("SELECT * FROM nomes");
        out.println("<table border>");
        while(rs.next()) {
          out.println("<tr><td>"+rs.getString("nome")+"</tr></td>");
        } // ... resto do codigo
      } catch (SQLException e) {
        throw new ServletException(e);
      }
   }
}
```

Servlet que usa DAO

- *Misturar código de servlet com banco de dados não é uma boa idéia. O ideal é disponibilizar uma interface independente de banco para o servlet*

```
public class DataAccessServlet extends HttpServlet {
   DataAccessDAO dao;
   public void init() {
      dao = JDBCDataAccessDAO.getInstance();
   }
   public void doGet(... request, ... response) ... {
      try {  // ... codigo de inicializacao do response
        String[] nomes = dao.listarTodosOsNomes();
        out.println("<table border>");
        for(int i = 0; i < nomes.length; i++) {
          out.println("<tr><td>"+nomes[i]+"</tr></td>");
        } // ... resto do codigo
      } catch (AcessoException e) {
        throw new ServletException(e);
      }
   }
}
```

Exemplo de DAO

- *O DAO é um objeto que isola o servlet da camada de dados, deixando-o à vontade para mudar a implementação*
- *Faça-o sempre implementar uma interface Java*

```
import java.sql.*; // ... outros pacotes
public class JDBCDataAccessDAO implements DataAccessDAO {
   // ...
   public void JDBCDataAccessDAO() {
      try {
        Class.forName(driverClass);
        con = DriverManager.getConnection(jdbcURL, nome, senha);
     } catch (Exception e) { ... }
   }
   public String[] listarTodosOsNomes() throws AcessoException {
      try {
        Statement stmt = con.createStatement();
        ResultSet rs = stmt.executeQuery("SELECT * FROM nomes");
        List nomesList = new ArrayList();
        while(rs.next()) {
          nomesList.add(rs.getString("nome"));
        }
        return (String[]) nomesList.toArray(new String[nomesList.size()];
      } catch (SQLException e) {
        throw new AcessoException(e);
      }
   }
}
```

Exercicio

- 1. Crie um DAO para isolar a gravação de produtos (atualmente gravados em arquivo) do resto da aplicação
  - Todo o conhecimento sobre gravação deve ficar no DAO
  - Use uma interface ou classe abstrata ProdutosDAO e implemente o DAO em uma classe ArquivoProdutosDAO
- 2. Crie uma tabela Produto no seu banco de dados e implemente as funções do DAO usando JDBC
  - Crie uma classe JDBCProdutosDAO
  - Passe os dados de configuração do banco como parâmetros de contexto e recupere-os no seu init()
- 3. Implemente, na sua aplicação, uma opção administrativa para remover produtos e outra para editar produtos
  - Crie as novas funções no DAO e implemente-as com JDBC
- 4. Implemente a função adicionar autor e adicionar editor do DAO da aplicação Biblioteca disponível no diretório cap08

helder@ibpinet.net

argonavis.com.br