Banco de Dados – Atividade Hands-on em Laboratório - 09
Implementação da Camada de Persistência – Aplicação Java
Prof. Dr. Aparecido Freitas

1. Introdução

Este laboratório assume que o Kit de Desenvolvimento Java (jdk), O Web Container Tomcat versão 9 e a IDE Eclipse for JEE (Photon). Configurar o Apache Tomcat para responder requisições na porta 8888. Criar no Eclipse um projeto **Maven** chamado **Maven_Qualit** e considerar a seguinte configuração para o arquivo **pom.xml.**

```
<properties>
		<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
		<failOnMissingWebXml>false</failOnMissingWebXml>
	</properties>

	<dependencies>
		<dependency>
			<groupId>javax.servlet</groupId>
			<artifactId>javax.servlet-api</artifactId>
			<version>4.0.1</version>
			<scope>provided</scope>
		</dependency>
	</dependencies>

	<build>
		<plugins>
			<plugin>
				<artifactId>maven-compiler-plugin</artifactId>
				<version>3.0</version>
				<configuration>
					<source>1.8</source>
					<target>1.8</target>
				</configuration>
			</plugin>
		</plugins>
	</build>


</project>
```

Para testar se o ambiente está operacional, botão direito no projeto e vamos criar uma página HTML, de nome **index.html**, com o seguinte **conteúdo**:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Atividade de Implementacao da Camada de Persistencia</title>

</head>
<body>

<h1> Seja Bem vindo !  </h1>

<h2><a href="JSP_01.jsp"> Data e Hora</a></h2>


</body>
</html>
```

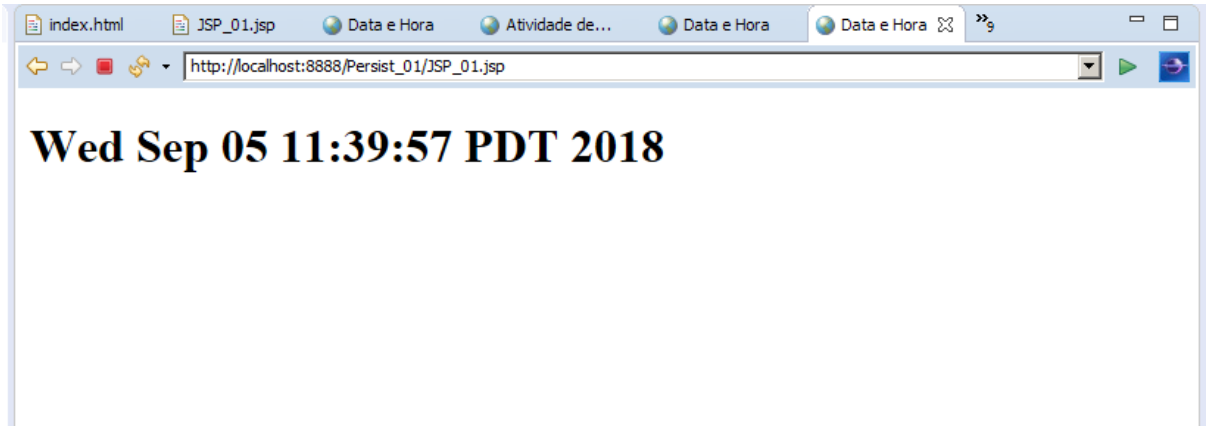
Para a execução, botão direito no arquivo **index.hml** e **Run On Server.** Vamos agora criar uma página JSP – Java Server Page para mostrar o servidor Tomcat em execução. Botão direito no projeto, **New JSP File**, de nome **JSP_01.jsp.**

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Data e Hora </title>
</head>
<body>

<h1><%=  new java.util.Date() %> </h1>

</body>
</html>
```

Agora, vamos atualizar a página index.html para incluir o link para chamada da página **JSP**:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Atividade de Implementacao da Camada de Persistencia</title>

</head>
<body>

<h1> Seja Bem vindo !  </h1>

<h2><a href="JSP_01.jsp"> Data e Hora</a></h2>


</body>
</html>
```

Para a execução, botão direito no arquivo **index.hml** e **Run On Server.**



2. Criação de Servlets

Vamos criar uma simples aplicação que irá manter um cadastro de **Funcionários**. Inicialmente, vamos criar um arquivo **Form12.html** para exibir um formulário de inserção de Funcionários.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Implementacao da Camada de Persistencia...</title>
</head>
<body>

<body    BGCOLOR="#6699FF">

<h1> Cadastro de Funcionários...  </h1>

<form name = "frmCadastro" method = "post"  action =
"/Maven_Qualit/CadastrarFuncionarioServlet">

      <p> Nome:   <input type = "text" name ="nome" size ="30"/> </p>
      <p> Data de nascimento: <input type = "text" name = "nascimento" size =
"15"/> (dd/mm/aa) </p>
      <p> Salário: R$ <input type ="text" name = "salario" size = "15" /> </p>
      <p> Sexo: <input type = "radio" name ="sexo" value ="M" /> Masculino <input
type ="radio" name = "sexo" value ="F" /> Feminino</p>

      <p> <input type ="checkbox"  name = "temporario" value ="true" />
Funcionário Temporário? </p>

      <p> <input type="submit" Value = "Cadastrar"/> </p>


</form>

</body>
</html>
```

Vamos agora criar a classe de Domínio que representa um **Funcionário**, no package **br.com.qualitsys.dominio.**

```
package br.com.qualitsys.dominio;
import java.util.Date;
public class Funcionario {

	private String nome;
	private Date nascimento;
	private Double salario;
	private Character sexo;
	private Boolean temporario;

	public Funcionario()  { }

	public Funcionario(String nome, Date nascimento, Double salario, Character
sexo, Boolean temporario) {
		super();
		this.nome = nome;
		this.nascimento = nascimento;
		this.salario = salario;
		this.sexo = sexo;
		this.temporario = temporario;
	}
	public String getNome() {
		return nome;
	}
	public void setNome(String nome) {
		this.nome = nome;
	}
	public Date getNascimento() {
		return nascimento;
	}
	public void setNascimento(Date nascimento) {
		this.nascimento = nascimento;
	}
	public Double getSalario() {
		return salario;
	}
	public void setSalario(Double salario) {
		this.salario = salario;
	}
	public Character getSexo() {
		return sexo;
	}
	public void setSexo(Character sexo) {
		this.sexo = sexo;
	}
	public Boolean getTemporario() {
		return temporario;
	}
	public void setTemporario(Boolean temporario) {
		this.temporario = temporario;
	}


}
```

Vamos agora criar uma classe que irá **simular** o Banco de Dados da aplicação. Vamos gravar esta classe de Dados no package **br.com.qualitsys.modelo.**

```
package br.com.qualitsys.modelo;

import java.util.ArrayList;
import java.util.List;

import br.com.qualitsys.dominio.Funcionario;

public class Dados {

	private static List<Funcionario> funcionarios = new
ArrayList<Funcionario>();

	public static void cadastrarFuncionario(Funcionario funcionario) {

		funcionarios.add(funcionario);

	}

	public static List<Funcionario> listarFuncionarios() {

		return funcionarios;
	}

}
```

Vamos agora criar um **Servlet** que irá receber os dados enviados pelo usuário para o cadastramento de funcionários. Botão direito no Projeto, vamos criar um Servlet,
chamado **CadastrarFuncionarioServlet** no package **br.com.qualitsys.controle**.

O Eclipse automaticamente criará o código abaixo descrito:

```
package br.com.qualitsys.controle;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * Servlet implementation class CadastrarFuncionarioServlet
 */
@WebServlet("/CadastrarFuncionarioServlet")
public class CadastrarFuncionarioServlet extends HttpServlet {
	private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public CadastrarFuncionarioServlet() {
        super();
        // TODO Auto-generated constructor stub
    }

	/**
	 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse
response)
	 */
	protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
		// TODO Auto-generated method stub
		response.getWriter().append("Served at:
").append(request.getContextPath());
	}

	/**
	 * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse
response)
	 */
	protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
		// TODO Auto-generated method stub
		doGet(request, response);
	}

}
```

Vamos agora criar um método chamado **doService()** que irá ser chamado tanto pelo método **doGet()** quanto pelo método doPost(). Com este procedimento, o **Servlet** executará a funcionalidade de tratamento da requisição HTTP tanto se esta for feita pelo método GET ou pelo método POST.

```
private void doService(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

                    String nome = request.getParameter("nome");
                    String sNascimento = request.getParameter("nascimento");
                    String sSalario = request.getParameter("salario");
                    String sSexo = request.getParameter("sexo");
                    String sTemporario = request.getParameter("temporario");

                    Funcionario funcionario = null;

                    DateFormat df = new SimpleDateFormat("dd/MM/yyyy");

                    try {
                          Date nascimento = df.parse(sNascimento);
                          Double salario = Double.parseDouble(sSalario);
                          Character sexo = sSexo.charAt(0);
                          Boolean temporario = Boolean.parseBoolean(sTemporario);

                    funcionario = new Funcionario(nome, nascimento, salario, sexo,
temporario);

                    }
                    catch (Exception e) {

                          //throw new ServletException(e) ;
                          System.out.println("Erro no metodo doService() .....");
                    }

                    if (funcionario != null ) {
                          Dados.cadastrarFuncionario(funcionario);
                          response.setContentType("text/html");
                          PrintWriter out = response.getWriter();
                          out.write("<html> <head><title> Funcionarios
Cadastrados</title></head></body>");
                          out.write(("<h1>Funcionários cadastrados</h1><ol>"));

                          List<Funcionario> lista = Dados.listarFuncionarios();

                          for (Funcionario f : lista ) out.write("<li><p>"  +
f.getNome() + "</p></li>" );

                          out.write("</ol>");

                          out.write("<p> <hr>   </p> <p> <a  href = 'Form12.html'>
Formulário de Cadastro</a></p></body></html> ");
                          out.close();

                    }


      }
```

Ao final da execução do método **doService()**, o formulário Form12.html poderá ser novamente chamado para inserção de um novo Funcionário.

As telas abaixo exibem a operação da aplicação:

Após a entrada de dados, o usuário deve clicar no botão Cadastrar para chamada do Servlet:
O servlet exibirá os dados cadastrados e permitirá a inclusão de novos funcionários.

A aplicação deste exercício está **simulando** um banco de dados, por meio da gravação dos dados em uma lista implementada em memória.

Implementar a camada de persistência, substituindo o procedimento de gravação em lista pelo emprego de um **Sistema Gerenciador de Banco de Dados**. Fique a vontade para utilizar o **Gerenciador de Banco de Dados** que lhe form mais conveniente.