Linguagem de Programação III

Atividade 1

Aplicação WEB

Fevereiro 2017

Sumário

Objetivo

Através da implementação de uma aplicação WEB, consolidar o domínio de conceitos como:

- Servidores WEB.
- Java, Servlets, JSP.
- HTML.
- Protocolo HTTP.
- Persistência (JDBC).
- Controle de Sessão.
- Padrão de projeto.
- Estrutura de Dados.
- Mapeamento Objeto Relacional (JPA).
- UML.

Método

Execução de blocos de trabalho em um ambiente controlado e padronizado de implementação.

Ambiente de desenvolvimento e execução

- Plataforma Java versão 1.8.0
- Ambiente integrado de desenvolvimento: Netbeans 8.2.0
- Banco de dados relacional: Apache Derby 10.13.1.1
- Servidor WEB (Servlet Container): Apache Tomcat 9.0.0

Blocos de trabalho

1. Configuração do Apache Tomcat e implantação de uma aplicação WEB de exemplo. Configuração do Apache Derby e de uma aplicação cliente.
2. Implementação da Primeira parte da Aplicação WEB. (FrontController).

Bloco 1 - Configuração do ambiente de desenvolvimento e execução

Objetivo

Construir um ambiente de desenvolvimento e execução de fácil configuração nos equipamentos e plataforma disponível nos laboratórios. O ambiente deve ser construído novamente em toda aula. O aluno deve ser capaz de identificar o que deve guardar do ambiente para restaurar futuramente o trabalho realizado.

Método

Seguir tutorial disponibilizado no Moodle para configuração dos Servidores Web e de Banco de Dados.

Responder as questões dissertativas.

- Atenção: Seguir criteriosamente as instruções para evitar dificuldades futuras. O diretório C:\Temp é temporário e é apagado com frequência e é utilizado para a construção do ambiente.

Questões

1. Seguindo o tutorial disponibilizado para configuração do Apache Derby, o que você deve copiar para não perder dados e esquemas de tabelas persistidos no SGBD?
2. Você identificou algum erro nos tutoriais disponibilizados?

Bloco 2 – Aplicação (Parte I) – FrontController

Objetivo

No ambiente de desenvolvimento construído, implementar uma aplicação WEB fazendo uso de padrões de projeto, JSP, Servlet, Servlet Container e Java.

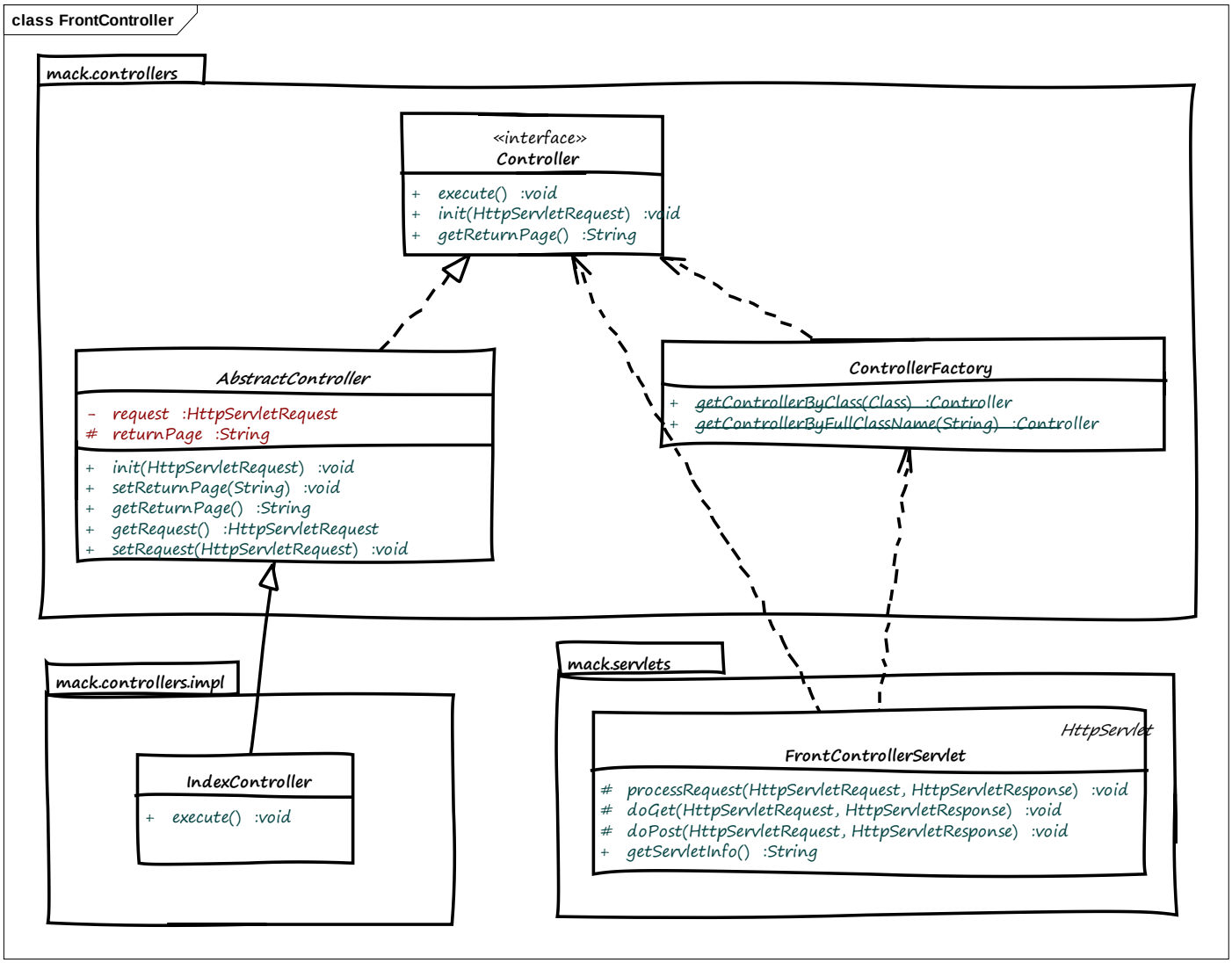
Método

1. Implementar a aplicação descrita no Anexo I.
    a. O novo projeto criado deve ser um projeto de uma aplicação WEB.
    b. Ao criar o Servlet, utilize o assistente (novo -> servlet) e certifique-se que o checkbox relacionado à criação e atualização do arquivo web.xml está selecionado.
    c. Notem que não iremos implementar o IndexController.java que faz uso de DAO. Implemente a versão alternativa.
    d. Não se esqueça de acrescentar à biblioteca do projeto as duas bibliotecas (arquivos com terminação jar) disponibilizados no Moodle.
2. Com o FrontController funcionando, modifique (refatore) o nome do IndexController para um nome mais adequado para um Controller que lista todos os usuários cadastrados no sistema, um nome como ListaController. Identifique quais alterações você deve realizar para que esta modificação mantenha o sistema funcional. Responda a questão 1.
3. Modifique sua aplicação para que ela seja capaz de retornar um usuário através de uma busca pelo nome. Responda a questão 2.
4. Para cada classe implementada, descreva o seu propósito.
5. Responda as questões remanescentes.

Questões

1. Quais foram as modificações realizadas para modificar o nome do IndexController? Indique classes e alterações.
2. Quais classes, páginas JSP e HTML foram criadas ou modificadas para que a busca pelo nome fosse possível?
3. Verifique o web.xml e explique como uma url é mapeada para uma classe que implementa um servlet.
4. Neste exemplo, como um controller (IndexController) consegue enviar dados para uma página JSP?
5. Descreva como você pode exportar seu projeto (completo) e restaura-lo em uma outra oportunidade.

ANEXO 1 - FrontController



Interface Controller

```
package mack.controllers;


import javax.servlet.http.HttpServletRequest;


public interface Controller {

	public void execute();

	public void init(HttpServletRequest request);

	public String getReturnPage();

}
```

AbstractController

```
package mack.controllers;

import javax.servlet.http.HttpServletRequest;


public abstract class AbstractController implements Controller{

    private HttpServletRequest request;

    protected String returnPage;

    public void init(HttpServletRequest request) {

        this.setRequest(request);

    }

    public void setReturnPage(String page) {

        returnPage = page;

    }

    public String getReturnPage() {

        return returnPage;

    }

    public HttpServletRequest getRequest() {

        return request;

    }

    public void setRequest(HttpServletRequest request) {

        this.request = request;

    }


}
```

ControllerFactory

```
package mack.controllers;


public class ControllerFactory {


    public static final Controller getControllerByClass(Class actionClass) {
        try {
            Controller controller = (Controller) actionClass.newInstance();
            return (Controller) actionClass.newInstance();
        } catch (java.lang.InstantiationException e) {
            e.printStackTrace();
        } catch (IllegalAccessException e) {
            e.printStackTrace();
        } catch (ClassCastException e) {
            e.printStackTrace();
        }
        return null;
    }
    public static final Controller getControllerByFullClassName(String className) {
        try {
            String name = "mack.controllers.impl." + className + "Controller";
            Class actionClass = Class.forName(name);
            return getControllerByClass(actionClass);
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
        return null;
    }


}
```

IndexController

```
package mack.controllers.impl;


import java.util.ArrayList;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;
import mack.controllers.AbstractController;
import mack.entities.Usuario;



public class IndexController extends AbstractController {


    public void execute() {
        try {
            List usuarios = new ArrayList<Usuario>();
            usuarios.add(new Usuario(1,"Mack","Junior"));
            usuarios.add(new Usuario(2, "Mack","Neto"));
            this.setReturnPage("/index.jsp");
            this.getRequest().setAttribute("usuarios", usuarios);

        }catch(Exception ex)  {
            Logger.getLogger(IndexController.class.getName()).log(Level.SEVERE, null, ex);
        }

    }


}
```

FrontControllerServlet

```
package mack.servlets;


import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import mack.controllers.Controller;
import mack.controllers.ControllerFactory;



public class FrontControllerServlet extends HttpServlet {


    /**
     * Processes requests for both HTTP <code>GET</code> and <code>POST</code>
     * methods.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
       response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            String controller = request.getParameter("control");
        Controller control = ControllerFactory.getControllerByFullClassName(controller);
         control.init(request);
            control.execute();
            RequestDispatcher requestDispatcher =
```

```
                    getServletContext().getRequestDispatcher(control.getReturnPage());
            requestDispatcher.forward(request, response);
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            out.close();
        }

    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign
on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Handles the HTTP <code>POST</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
```

```
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     *
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

Usuario

```
package mack.entities;

public class Usuario {

    private int id;

    private String nome;

    private String sobrenome;

    public Usuario(int id, String nome, String sobrenome) {

        this.id = id;

        this.nome = nome;

        this.sobrenome = sobrenome;

    }

    public String getNome() {

        return nome;

    }

    public void setNome(String nome) {

        this.nome = nome;

    }

    public String getSobrenome() {

        return sobrenome;

    }

    public void setSobrenome(String sobrenome) {

        this.sobrenome = sobrenome;

    }

    public int getId() {

        return this.id;

    }

    public String toString() {

        StringBuffer sbResult = new StringBuffer();

        sbResult.append("id = ");

        sbResult.append(id);

        sbResult.append(", nome = ");

        sbResult.append(nome);

        sbResult.append(", sobrenome = ");

        sbResult.append(sobrenome);

        return sbResult.toString();
```

```
    }
}
```

Index.html

```
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Pagina Principal</title>
    </head>
    <body>
        <h1>Exemplo de uso de FrontController</h1>


        <a href="FrontControllerServlet?control=Index">Lista</a>
    </body>
</html>
```

Index.jsp

```
%--
    Document   : index
    Created on : 07/04/2014, 20:49:49
--%>
<%@page import="mack.entities.Usuario"%>
<%@page import="java.util.List"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type"
            content="text/html; charset=UTF-8">
        <title>JSP Page</title>
    </head>
    <body>
        <h1>Usuarios!</h1>
        <%
            List<Usuario> usuarios =
             (List<Usuario>) request.getAttribute("usuarios");
         %>
         <%if (usuarios.size()>0) { %>
         <table>
            <% for (Usuario u:usuarios){%>
            <tr>
                <td><%=u.getNome()%></td>
                <td><%=u.getSobrenome()%></td>
            </tr>
            <%}%>
         </table>
         <%}%>
    </body>
</html>
```