FTCE

Faculdade de Tecnologia e Ciências Exatas

USJT - 2017 - Práticas de Programação

Professores Teoria: Bonato, Hamilton
Laboratório: Calvetti, Hamilton, Keity e Rodrigo

Aula: 9
Assunto: Front Controller e Command

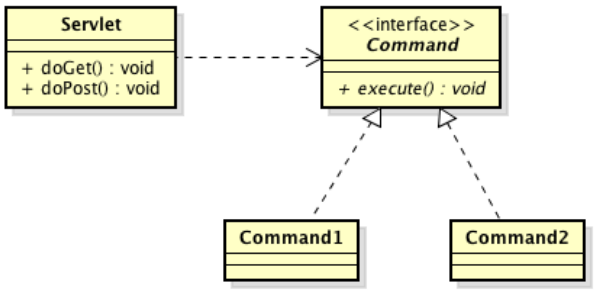
Conceitos básicos

Front Controller e Command

O que são

São padrões de projetos nos quais um único controller (Front Controller) trata todas as requisições de uma aplicação web e então delega esta ação para uma classe que implementa o padrão de projetos Command.

Estrutura:



Como Funciona

O Servlet do FrontController tem um método doExecute que recebe como parâmetros os objetos request e response. O doGet e o doPost deste servlet devem chamar o doExecute.

O doExecute então pega da request o parâmetro "Command", que deve ter o nome da classe Command a ser instanciada, e faz a instanciação via reflexão, chamando o Class.forName(String).newInstance().

A classe instanciada deve implementar a interface Command, que tem um método void execute que recebe por parâmetros os objetos request e response. O método execute, então, faz o que foi solicitado na requisição.

Os formulários que invocam o servlet controller devem, por sua vez, passar como argumento, no parâmetro command, o nome da classe concreta Command que deve ser instanciada.

Vantagens

A aplicação passa a ter um único ponto de entrada, facilitando a adição de novas funcionalidades por meio de decoradores e filtros.

Desvantagens

Dependendo do que for compartilhado nos Comandos pode haver problemas de controle de concorrência.

Exemplo

O CRUD de Clientes que vimos fazendo até o momento, que está no padrão MVC, tem dois servlets. Cada servlet tem uma série de desvios condicionais dentro dos seus métodos que executam as ações conforme o que vem no parâmetro ação. Estes dois servlets serão substituídos por um conjunto de classes Command.

Exemplo View: VisualizarCliente.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
        <!DOCTYPE html>
        <html lang="pt-br">

        <head>
            <meta charset="utf-8">
            <meta http-equiv="X-UA-Compatible" content="IE=edge">
            <meta name="viewport" content="width=device-width, initial-
scale=1">
            <title>cerveja.biz - Visualizar Cliente</title>

            <link href="css/bootstrap.min.css" rel="stylesheet">
            <link href="css/style.css" rel="stylesheet">
        </head>

        <body>
                <!-- Modal -->
                <div class="modal fade" id="delete-modal" tabindex="-1"
role="dialog" aria-labelledby="modalLabel">
                    <div class="modal-dialog" role="document">
                        <div class="modal-content">
                            <div class="modal-header">
                                <button type="button" class="close" data-
dismiss="modal" aria-label="Fechar"><span aria-hidden="true">&times;</span>
                                </button>
                                <h4 class="modal-title"
id="modalLabel">Excluir Cliente</h4>
                            </div>
                            <div class="modal-body">
                                Deseja realmente excluir este cliente?
                            </div>
                            <div class="modal-footer">
                                <form action="controller.do" method="post">
                                    <input type="hidden" name="id"
value="${cliente.id }" />
```

```
                                    <button type="submit" class="btn btn-
primary" name="command" value="ExcluirCliente">Sim</button>
                                    <button type="button" class="btn btn-
default" data-dismiss="modal">N&atilde;o</button>
                                </form>
                            </div>
                        </div>
                    </div>
                </div>
                <!-- /.modal -->
                <!-- Barra superior com os menus de navegação -->
            <c:import url="Menu.jsp"/>
                <!-- Container Principal -->
                <div id="main" class="container">
                    <h3 class="page-header">Visualizar Cliente #${cliente.id
}</h3>
                    <div class="row">
                        <div class="col-md-12">
                            <p><strong>Nome</strong>
                            </p>
                            <p>
                                ${cliente.nome }
                            </p>
                        </div>
                    </div>
                    <div class="row">
                        <div class="col-md-6">
                            <p><strong>Celular</strong>
                            </p>
                            <p>
                                ${cliente.fone }
                            </p>
                        </div>
                        <div class="col-md-6">
                            <p><strong>E-Mail</strong>
                            </p>
                            <p>
                                ${cliente.email }
                            </p>
                        </div>
                    </div>
                    <hr />
                    <div id="actions" class="row">
                        <div class="col-md-12">
                            <a
href="controller.do?command=EditarCliente&id=${cliente.id }" class="btn btn-
primary">Editar</a>
                            <a href="#" class="btn btn-danger" data-
toggle="modal" data-target="#delete-modal">Excluir</a>
                            <a href="ListarClientes.jsp" class="btn btn-
default">Voltar</a>
                        </div>
                    </div>
                </div>
                <script src="js/jquery.min.js"></script>
                <script src="js/bootstrap.min.js"></script>
        </body>
```

```
        </html>
```

Exemplo Controller: ServletController.java

```
package controller;

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;

import command.Command;

@WebServlet("/controller.do")
public class ServletController extends HttpServlet {
   private static final long serialVersionUID = 1L;

   protected void doExecute(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
      try {
         request.setCharacterEncoding("UTF-8");
         Command comando =
(Command)Class.forName("command."+request.getParameter("command")).newInstan
ce();
         comando.executar(request, response);
      } catch (InstantiationException | IllegalAccessException
            | ClassNotFoundException e) {
         e.printStackTrace();
         throw new ServletException(e);
      }
   }

   protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
      doExecute(request,response);
   }

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

}
```

Exemplo Command: EditarCliente.java

```
package command;

import java.io.IOException;
import java.util.ArrayList;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServletRequest;
```

```java
import javax.servlet.http.HttpServletResponse;
import model.Cliente;
import service.ClienteService;

public class EditarCliente implements Command {

   @Override
   public void executar(HttpServletRequest request,
         HttpServletResponse response) throws ServletException, IOException {

      String pId = request.getParameter("id");
      String pNome = request.getParameter("nome");
      String pFone = request.getParameter("fone");
      String pEmail = request.getParameter("email");
      int id = -1;
      try {
         id = Integer.parseInt(pId);
      } catch (NumberFormatException e) {

      }

      Cliente cliente = new Cliente();
      cliente.setId(id);
      cliente.setNome(pNome);
      cliente.setFone(pFone);
      cliente.setEmail(pEmail);
      ClienteService cs = new ClienteService();

      RequestDispatcher view = null;

      cliente = cs.carregar(cliente.getId());
      request.setAttribute("cliente", cliente);
      view = request.getRequestDispatcher("AlterarCliente.jsp");

      view.forward(request, response);

   }

   public int busca(Cliente cliente, ArrayList<Cliente> lista) {
      Cliente to;
      for(int i = 0; i < lista.size(); i++){
         to = lista.get(i);
         if(to.getId() == cliente.getId()){
            return i;
         }
      }
      return -1;
   }

}
```

Exercícios

1. Revisão da Teoria

a. Importe o arquivo exemplo_clientes.war. Faça deployment no Tomcat e rode. Depois analise o código para entendê-lo.

2. Exercício para Nota (correção em aula)

a. Implemente os padrões front-controller e command no CRUD de Pais implementado na semana passada, ficando com apenas um servlet.

Bibliografia

DEITEL, Harvey M.; DEITEL, Paul J; FURMANKIEWICZ, Edson. **Java: como programar.** 6. ed. atual. São Paulo: Pearson, 2005. xl, 1110 p. ISBN 8576050196 (Broch.)

BASHAM, Bryan; SIERRA, Kathy; BATES, Bert. **Use a cabeça!: Servlets & JSP**. 2. ed. Rio de Janeiro: Alta Books, 2008-2010. xxxii, 879 p. ISBN 9788576082941 (broch.)

FOWLER, Martin.; **Padrões de Arquitetura de Aplicações Corporativas.** Bookman, 2008.