Ciências ULisboa
Faculdade de Ciências da Universidade de Lisboa

2019/2020

Construção de Sistemas de Software

Aplicações Web Java

—

Padrões

Ciências ULisboa | Informática

CSS (2019/2020)
Caderno de Exercícios
Folha 8

Camada de Apresentação das Aplicações Web Java

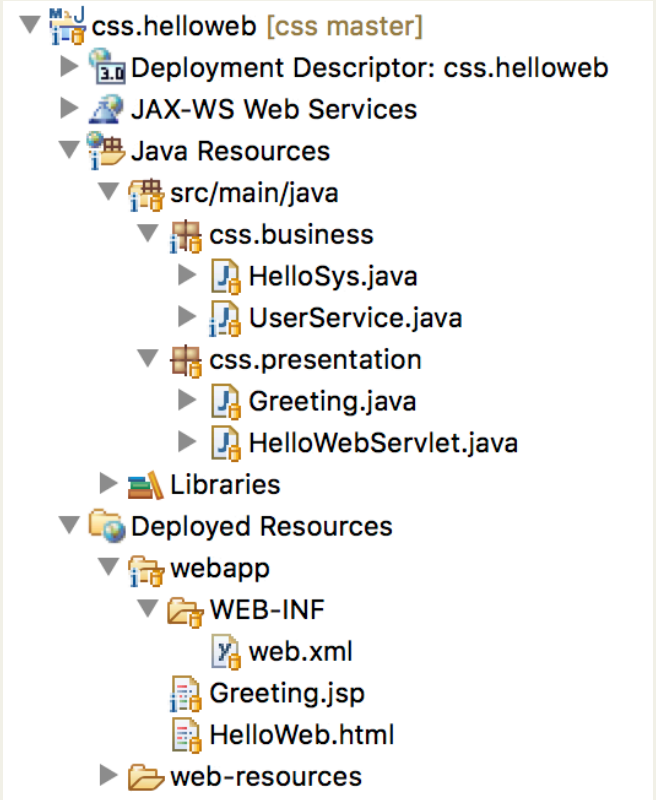
1. Considere de novo a aplicação *HelloWeb*, mais precisamente a versão da aplicação desenhada anteriormente que valida os dados de entrada —i.e., que o nome e o título fornecidos são válidos— e que obtém os pontos correspondentes a esses dados da camada de negócio, recorrendo a um objeto do tipo *UserService* que o objeto inicial do sistema, do tipo *HelloWebSys*, sabe fornecer.

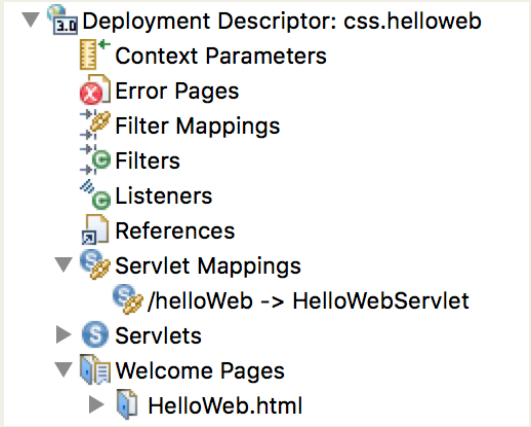
Exercícios

1. Considere de novo a aplicação *HelloWeb*, mais precisamente a versão da aplicação desenhada anteriormente que valida os dados de entrada —i.e., que o nome e o título fornecidos são válidos— e que obtém os pontos correspondentes a esses dados da camada de negócio, recorrendo a um objeto do tipo *UserService* que o objeto inicial do sistema, do tipo *HelloWebSys*, sabe fornecer.

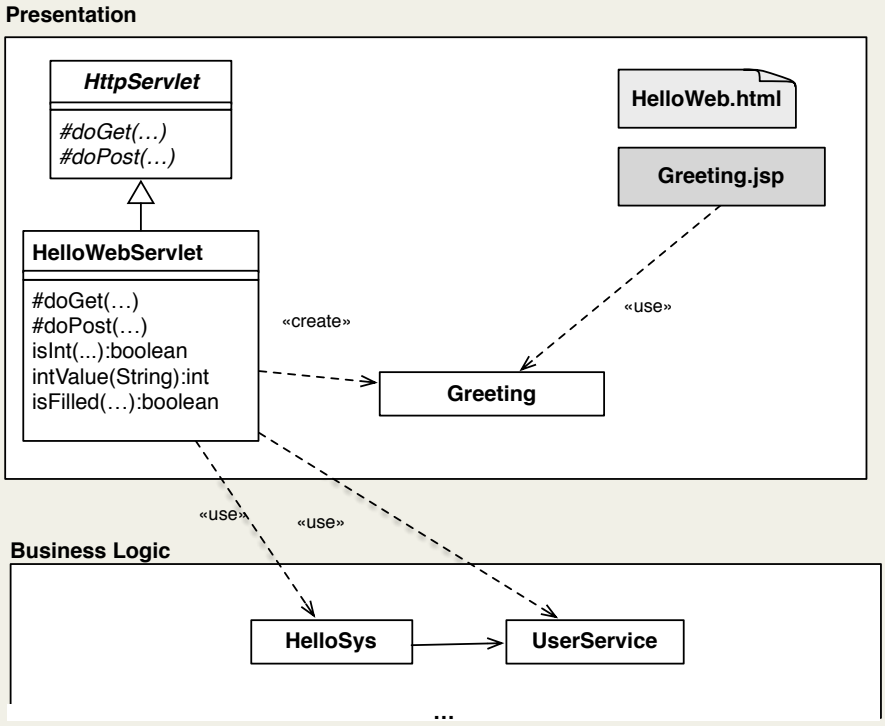
Explique porque o desenho da interface da aplicação segue o padrão MVC e descreva a organização do código da camada de apresentação da aplicação.

Exercícios

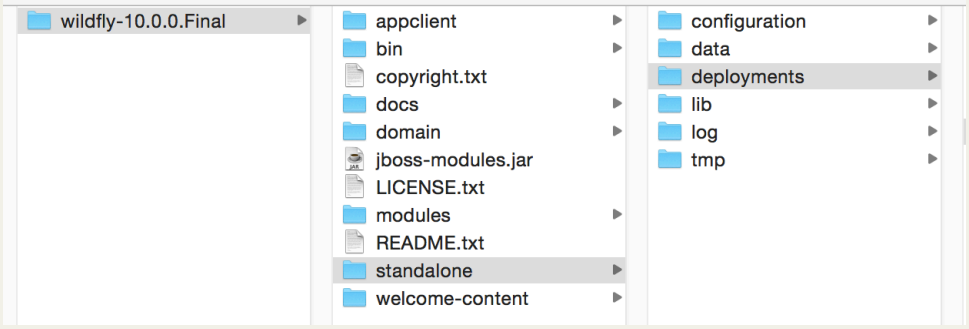


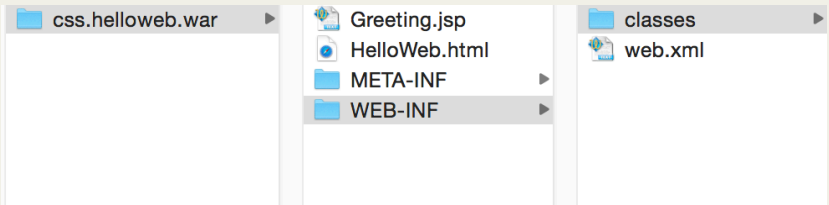


Exercícios

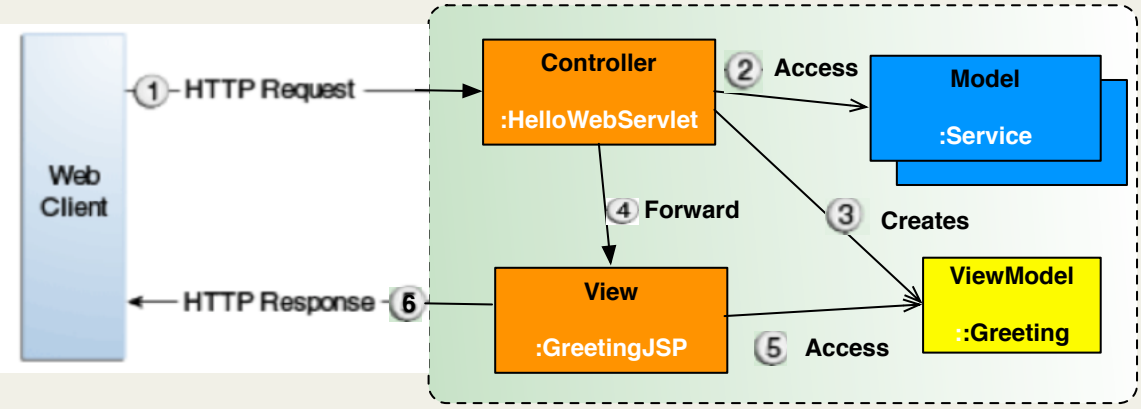


Exercícios





Exercícios

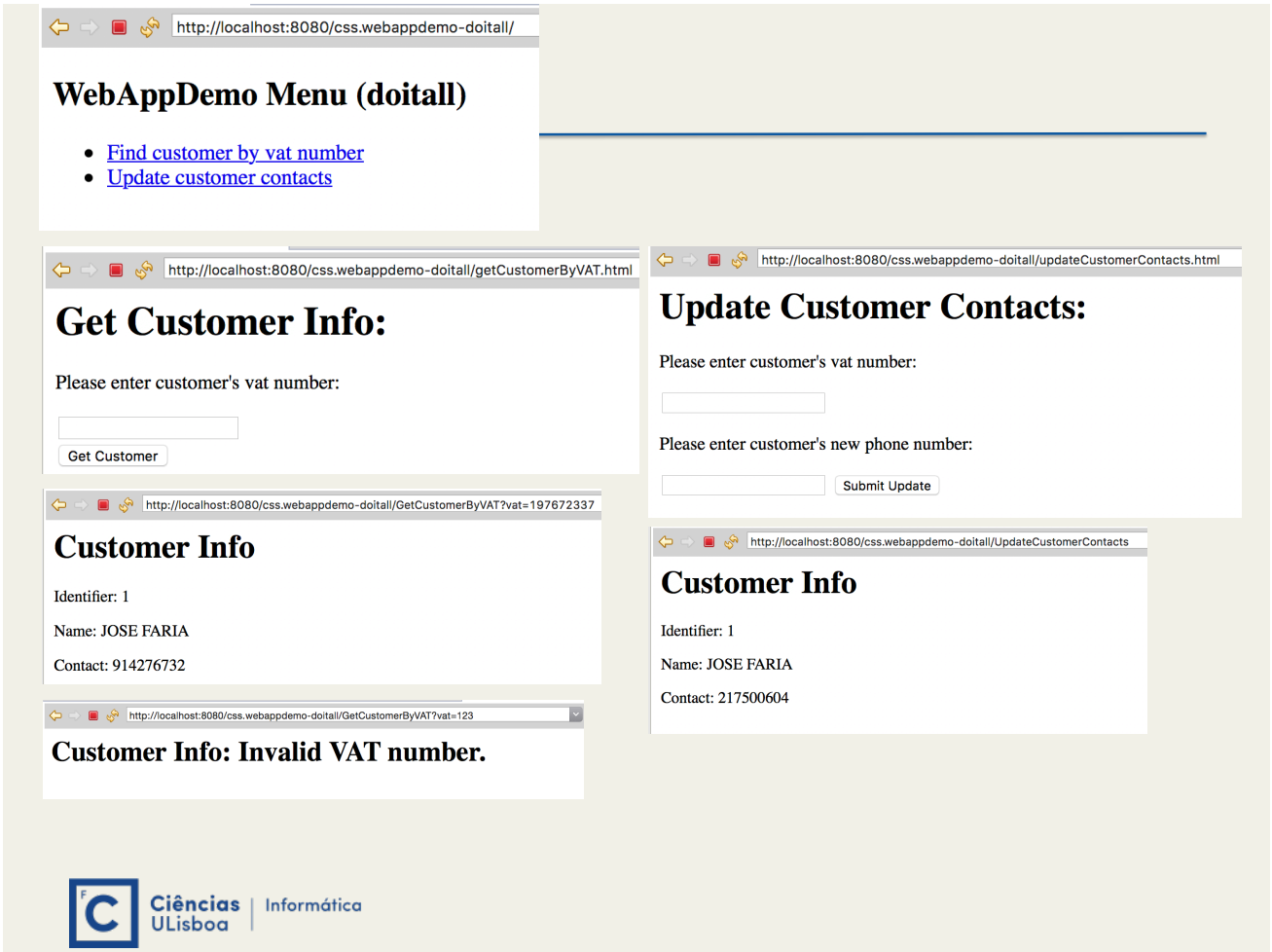


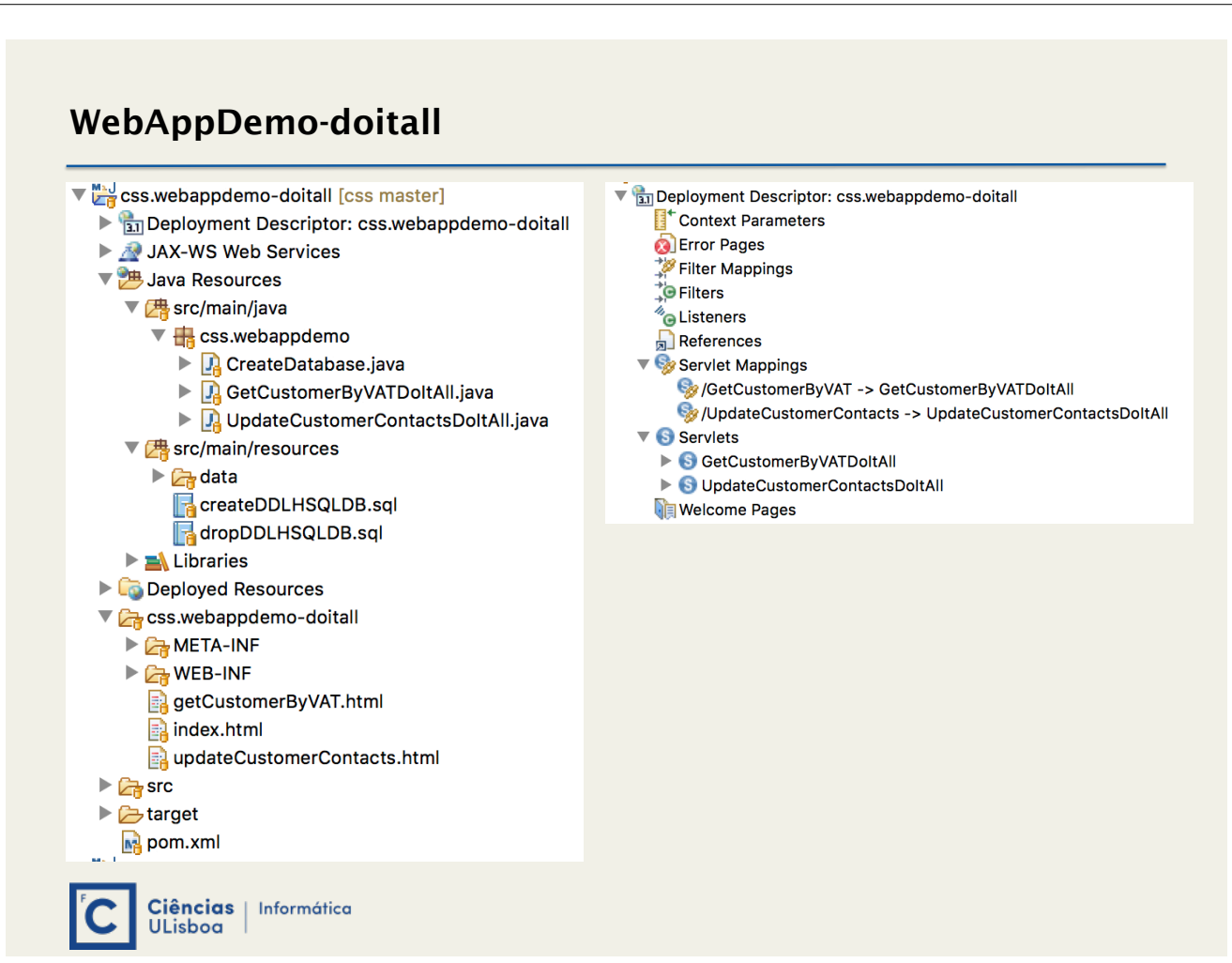
Exercícios

2. Considere a aplicação Java web cujo UI é o ilustrado abaixo e o projeto Eclipse Maven *webappdemo-doitall* que se encontra na página da disciplina, o qual contém uma implementação desta aplicação.

   (a) Analise o código fornecido e a forma como está organizado.

webapp demo doitall (recurso para aula TP)


http://localhost:8080/css.webappdemo-doitall/
WebAppDemo Menu (doitall)
Find customer by vat number
Update customer contacts
http://localhost:8080/css.webappdemo-doitall/getCustomerByVAT.html
Get Customer Info:
Please enter customer's vat number:
Get Customer
http://localhost:8080/css.webappdemo-doitall/GetCustomerByVAT?vat=197672337
Customer Info
Identifier: 1
Name: JOSE FARIA
Contact: 914276732
http://localhost:8080/css.webappdemo-doitall/GetCustomerByVAT?vat=123
Customer Info: Invalid VAT number.
http://localhost:8080/css.webappdemo-doitall/updateCustomerContacts.html
Update Customer Contacts:
Please enter customer's vat number:
Please enter customer's new phone number:
Submit Update
http://localhost:8080/css.webappdemo-doitall/UpdateCustomerContacts
Customer Info
Identifier: 1
Name: JOSE FARIA
Contact: 217500604
Ciências ULisboa | Informática


WebAppDemo-doitall
css.webappdemo-doitall [css master]
Deployment Descriptor: css.webappdemo-doitall
JAX-WS Web Services
Java Resources
src/main/java
css.webappdemo
CreateDatabase.java
GetCustomerByVATDoItAll.java
UpdateCustomerContactsDoItAll.java
src/main/resources
data
createDDLHSQLDB.sql
dropDDLHSQLDB.sql
Libraries
Deployed Resources
css.webappdemo-doitall
META-INF
WEB-INF
getCustomerByVAT.html
index.html
updateCustomerContacts.html
src
target
pom.xml
Deployment Descriptor: css.webappdemo-doitall
Context Parameters
Error Pages
Filter Mappings
Filters
Listeners
References
Servlet Mappings
/GetCustomerByVAT -> GetCustomerByVATDoItAll
/UpdateCustomerContacts -> UpdateCustomerContactsDoItAll
Servlets
GetCustomerByVATDoItAll
UpdateCustomerContactsDoItAll
Welcome Pages
Ciências ULisboa | Informática

WebAppDemo-doitall

- GetCustomerByVATDoItAll
  - serialVersionUID : long
  - doGet(HttpServletRequest, HttpServletResponse) : void
  - doPost(HttpServletRequest, HttpServletResponse) : void
  - isValidVAT(String) : boolean

getCustomerByVAT.html

```
<HTML>
<HEAD>
<TITLE>Enter Name (version: doitall)</TITLE>
</HEAD>
<BODY>
<H1>Get Customer Info:</H1>

<FORM ACTION="GetCustomerByVAT" METHOD="GET">
<P>Please enter customer's vat number:</P>
<INPUT TYPE="TEXT" NAME="vat">
<BR>
<INPUT TYPE="SUBMIT" VALUE="Get Customer">
</FORM>

</BODY>
</HTML>
```

- UpdateCustomerContactsDoItAll
  - serialVersionUID : long
  - doGet(HttpServletRequest, HttpServletResponse) : void
  - doPost(HttpServletRequest, HttpServletResponse) : void
  - isValidPhone(String) : boolean
  - isValidVAT(String) : boolean

updateCustomerContacts.html

```
<HTML>
<HEAD>
<TITLE>Enter Name (version: doitall)</TITLE>
</HEAD>
<BODY>
<H1>Update Customer Contacts:</H1>

<FORM ACTION="UpdateCustomerContacts" METHOD="POST">
<P>Please enter customer's vat number:</P>
<INPUT TYPE="TEXT" NAME="vat">
<BR>
<P>Please enter customer's new phone number:</P>
<INPUT TYPE="TEXT" NAME="phone">
<INPUT TYPE="SUBMIT" VALUE="Submit Update">
</FORM>

</BODY>
</HTML>
```

Ciências ULisboa | Informática

```
@WebServlet("/GetCustomerByVAT")
public class GetCustomerByVATDoItAll extends HttpServlet {
    private static final long serialVersionUID = 1L;
    /* ... */
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String vat = request.getParameter("vat");
        if (!isValidVAT(vat)) {
            PrintWriter out = response.getWriter();
            response.setContentType("text/html");
            out.println("<h1>Customer Info: Invalid VAT number.</h1>");
        } else {
            URL f = getClass().getResource("/data/hsqldb");
            try (Connection dbc = DriverManager.getConnection ("jdbc:hsqldb:file:" +
                                                    f.getPath() + "cssdb", "SA", "")) {
                String findSql = "select id, vatnumber, designation, phonenumber " +
                        "from customer where vatnumber = ?";
                PreparedStatement statement = dbc.prepareStatement(findSql);
                statement.setInt(1, Integer.parseInt(vat));
                ResultSet rs = statement.executeQuery();

                PrintWriter out = response.getWriter();
                response.setContentType("text/html");

                if (!rs.next())
                    out.println("Customer with vat number "+ vat + " does not exist");
                else {
                    int id = rs.getInt("id");
                    String designation = rs.getString("designation");
                    int phN = rs.getInt("phoneNumber");

                    out.println("<h1>Customer Info </h1>");
                    out.println("<p>Identifier: "+ id + " </p> ");
                    out.println("<p>Name: "+ designation + " </p> ");
                    out.println("<p>Contact: "+ phN + " </p> ");
                }
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
```

WebAppDemo-doitall

```
@WebServlet("/UpdateCustomerContacts")
public class UpdateCustomerContactsDoItAll extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /* ... */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws

        String vat = request.getParameter("vat");
        if (!isValidVAT(vat)) {
            PrintWriter out = response.getWriter();
            response.setContentType("text/html");
            out.println("<h1>Customer Info: Invalid VAT number.</h1>");
        } else {
            String phone = request.getParameter("phone");
            if (!isValidPhone(phone)) {
                PrintWriter out = response.getWriter();
                response.setContentType("text/html");
                out.println("<h1>Customer Info: Invalid Phone number.</h1>");
            } else {

                URL f = getClass().getResource("/data/hsqldb");
                try (Connection dbc = DriverManager.getConnection ("jdbc:hsqldb:file:" +
                                                    f.getPath() + "cssdb", "SA", "")) {

                    String findSql = "update customer set phonenumber = ? " +
                            "where vatnumber = ?";
                    PreparedStatement statement = dbc.prepareStatement(findSql);
                    statement.setInt(1, Integer.parseInt(phone));
                    statement.setInt(2, Integer.parseInt(vat));
                    statement.executeUpdate();
```


WebAppDemo-doitall

```
                    findSql = "select id, vatnumber, designation, phonenumber " +
                            "from customer where vatnumber = ?";
                    statement = dbc.prepareStatement(findSql);
                    statement.setInt(1, Integer.parseInt(vat));
                    ResultSet rs = statement.executeQuery();

                    PrintWriter out = response.getWriter();
                    response.setContentType("text/html");

                    if (!rs.next()) {
                        out.println("Customer with vat number "+ vat + " does not exist");
                    } else {
                        int id = rs.getInt("id");
                        String designation = rs.getString("designation");
                        int phN = rs.getInt("phoneNumber");

                        out.println("<h1>Customer Info </h1>");
                        out.println("<p>Identifier: "+ id + " </p> ");
                        out.println("<p>Name: "+ designation + " </p> ");
                        out.println("<p>Contact: "+ phN + " </p> ");
                    }

                } catch (SQLException e) {
                    e.printStackTrace();
                }
            }
        }
    }
```

Exercícios

WebAppDemo-doitall

- Não segue qualquer padrão de desenho, nem sequer o mais básico de ter o código responsável pela apresentação, negócio e acesso aos dados separado
- O código tem vários *anti-padrões*

  Anti-padrão: Uma solução comum para um problema recorrente que é sabida ser ineficaz e altamente contra-produtiva.

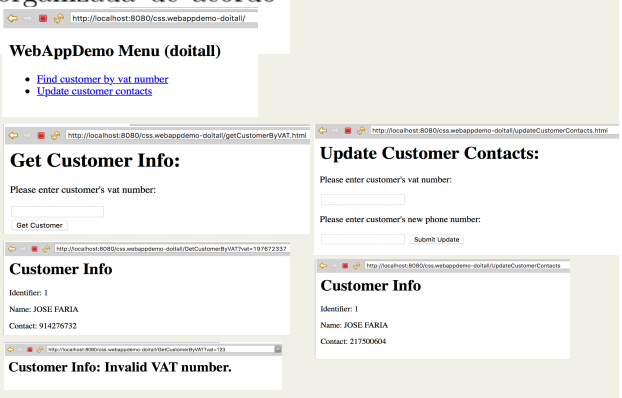
- Facilmente identificáveis:
  - ***God Object:*** *Concentrating too many functions in a single part of the design (class)*
  - ***ClassesWithoutOO:*** *Putting lots and lots of methods on one big massive "do it all" class.*

Esta solução é muito má e, portanto, nada recomendável

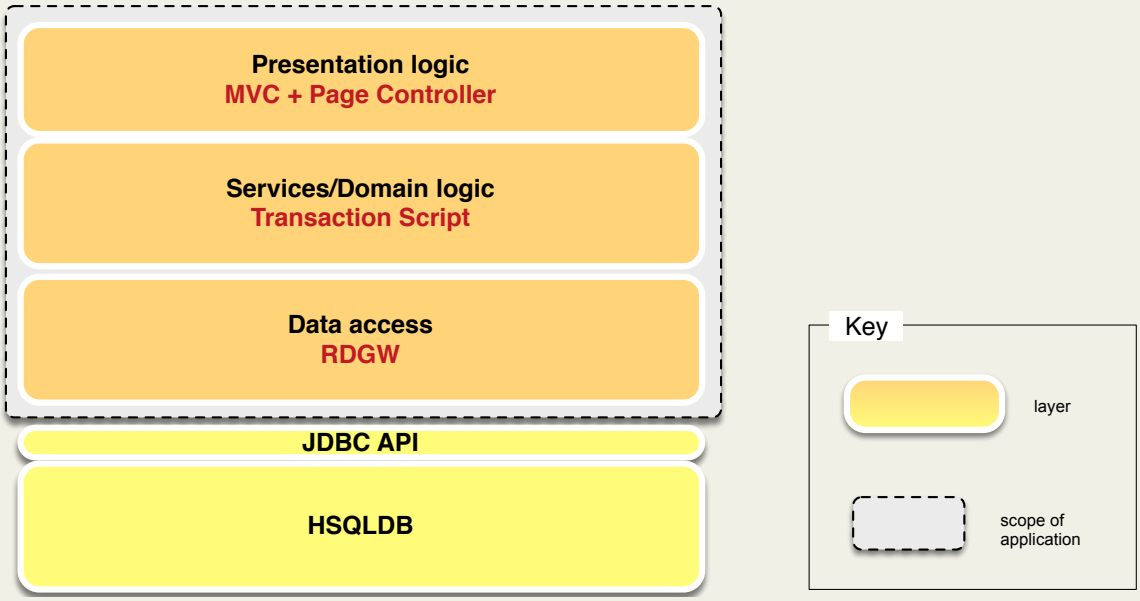
Exercícios

(b) Conceba uma solução de desenho alternativa para a aplicação que

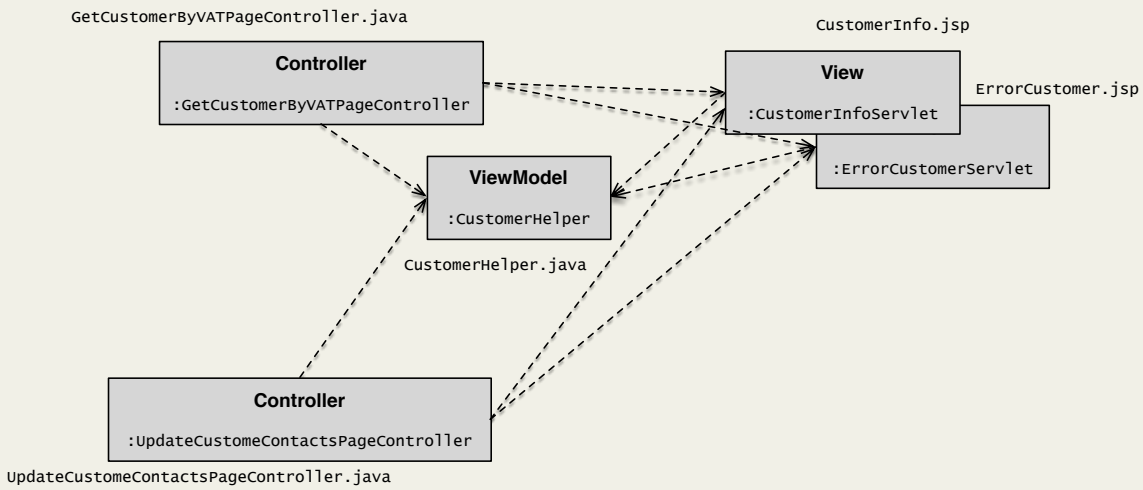
- utilize o padrão *MVC* e o padrão *Page Controller* na camada de apresentação
- tenha a camada de negócio organizada de acordo com o padrão *Transaction Script*
- tenha a camada de acesso aos dados organizada de acordo com o padrão *Row Data Gateway*



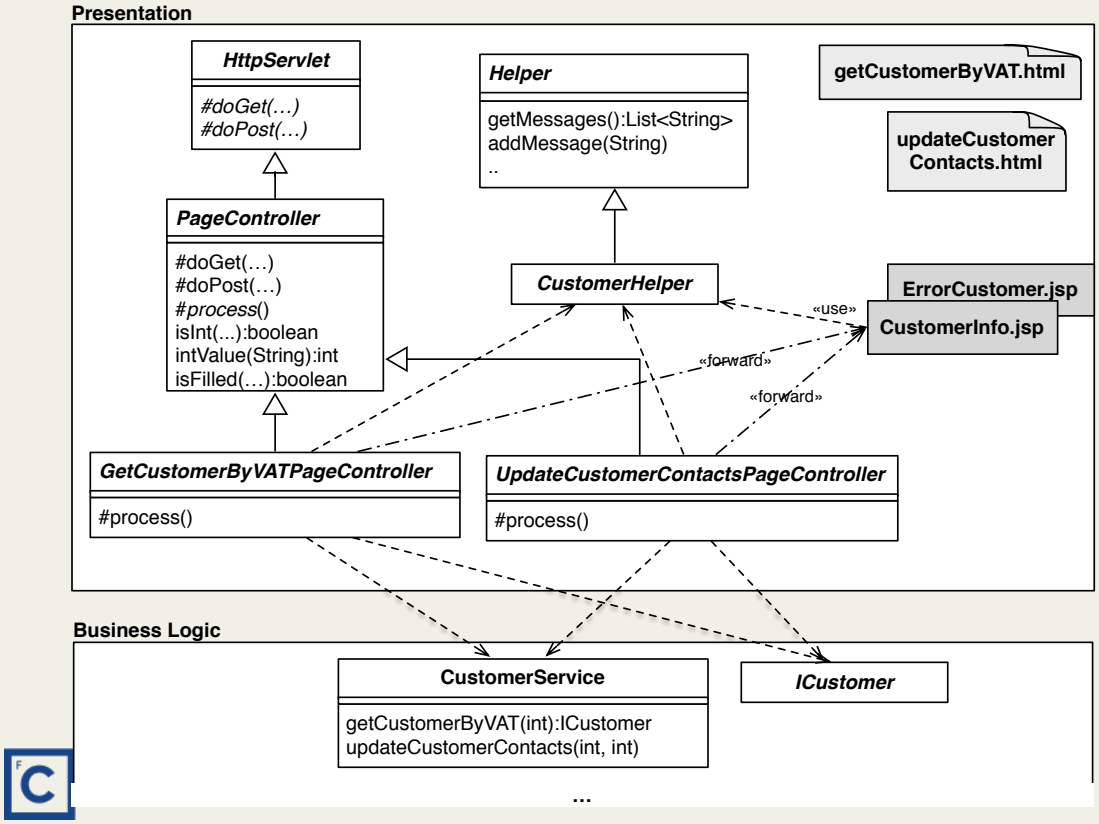
Exercícios



Refabricação da WebAppDemo: *MVC + Page Controller*



Refabricação da WebAppDemo: Camada de Apresentação



Refabricação da WebAppDemo: *Page Controller*

```
public abstract class PageController extends HttpServlet {

    private static final long serialVersionUID = -7066373204929867189L;

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws Serv
        doGet(request, response);
    }

    @Override
    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws Servl
        process(request,response);
    }

    /**
     * Strategy method for processing each request
     * @throws ServletException
     * @throws IOException
     */
    protected abstract void process(HttpServletRequest request, HttpServletResponse response) t
```

Refabricação da WebAppDemo: *Page Controller*

```java
@WebServlet("/GetCustomerPageController")
public class GetCustomerPageController extends PageController {

    private static final long serialVersionUID = 1L;

    @Override
    protected void process(HttpServletRequest request, HttpServletResponse response) throws Ser
        CustomerService cs = CustomerService.INSTANCE;

        CustomerHelper ch = new CustomerHelper();
        request.setAttribute("helper", ch);
        try {
            String vat = request.getParameter("vat");
            if (isInt(ch, vat, "Invalid VAT number")) {
                int vatNumber = intValue(vat);
                ch.fillWithCustomer(cs.getCustomerByVat(vatNumber));
                request.getRequestDispatcher("CustomerInfo.jsp").forward(request, response);
            }
        } catch (ApplicationException e) {
            ch.addMessage("It was not possible to fulfill the request: " + e.getMessage());
            request.getRequestDispatcher("CustomerError.jsp").forward(request, response);
        }

    }
```


Refabricação da WebAppDemo: *Page Controller*

getCustomerByVAT.html

```html
<HTML>
<HEAD>
<TITLE>Enter Name</TITLE>
</HEAD>
<BODY>
<H1>Get Customer Info (version: pagecontroller):</H1>

<FORM ACTION="GetCustomerPageController" METHOD="GET">
<P>Please enter customer's vat number:</P>
<INPUT TYPE="TEXT" NAME="vat">
<BR>
<INPUT TYPE="SUBMIT" VALUE="Get Customer">
</FORM>

</BODY>
</HTML>
```

*updateCustomerContacts.html

```html
<HTML>
<HEAD>
<TITLE>Enter Name</TITLE>
</HEAD>
<BODY>
<H1>Get Customer Info (version: pagecontroller):</H1>

<FORM ACTION="UpdateCustomerContactsPageController" METHOD="POST">
<P>Please enter customer's vat number:</P>
<INPUT TYPE="TEXT" NAME="vat">
<BR>
<P>Please enter customer's new phone number:</P>
<INPUT TYPE="TEXT" NAME="phone">
<INPUT TYPE="SUBMIT" VALUE="Submit Update">
</FORM>

</BODY>
</HTML>
```

Refabricação da WebAppDemo: *Page Controller*

*CustomerInfo.jsp

```jsp
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/lo
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<jsp:useBean id="helper" class="css.webappdemopagecontroller.webpresentation.CustomerHelper"
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Customer Info</title>
</head>
<body>
<body>
<h2>Client Info</h2>
Identifier: <c:out value ="${helper.id}"></c:out>
Name: <c:out value ="${helper.designation}"></c:out>
Contact Info: <c:out value ="${helper.phNumber}"></c:out>
</body>
</html>
```

*CustomerError.jsp

```jsp
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/lo
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<jsp:useBean id="helper" class="css.webappdemopagecontroller.webpresentation.CustomerHelper"
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Customer Info</title>
</head>
<body>
<body>
    <p>Error Messages: </p>
    <ul>
    <c:forEach var="message" items="${helper.messages}">
        <li>${message}
    </c:forEach>
    </ul>
</body>
</html>
```

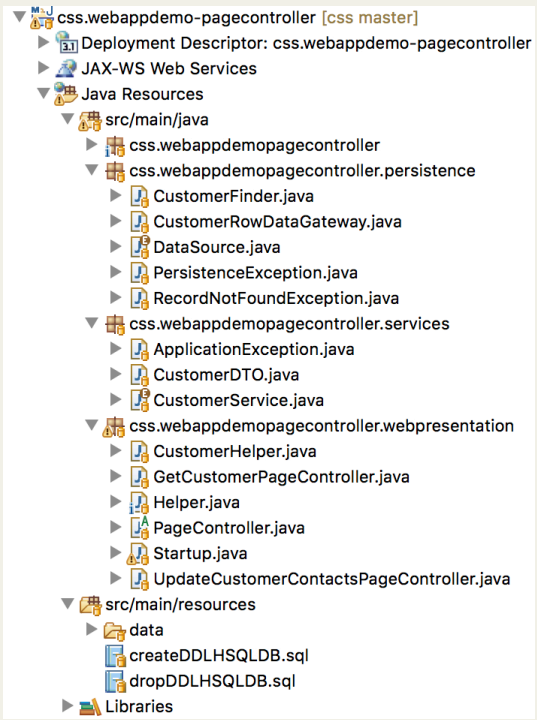

Inicialização e Finalização

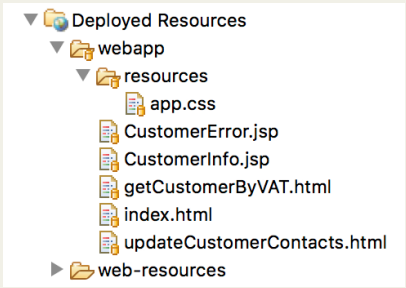
```java
/**
 * Application life cycle Listener implementation class
 *
 */
@WebListener
public class Startup implements ServletContextListener {

  /**
   * @see ServletContextListener#contextInitialized(ServletContextEvent)
   */
  public void contextInitialized(ServletContextEvent event)  {
    // Connects to the database

    URL f = getClass().getClassLoader().getResource("/data/hsqldb");
    try {
      DataSource.INSTANCE.connect("jdbc:hsqldb:file:" + f.getPath() + "cssdb", "SA", "");
    } catch (PersistenceException e) {
      System.out.println("Error connecting database");
      System.out.println("Application Message: " + e.getMessage());
      System.out.println("SQLException: " + e.getCause().getMessage());
      System.out.println("SQLState: " + ((SQLException) e.getCause()).getSQLState());
      System.out.println("VendorError: " + ((SQLException) e.getCause()).getErrorCode());
      return;
    }
    catch (NullPointerException e){
      System.out.println("Error connecting database");
      System.out.println("Not able to find the resource.");
      return;
    }
  }
```

Inicialização e Finalização

```
/**
 * @see ServletContextListener#contextDestroyed(ServletContextEvent)
 */
public void contextDestroyed(ServletContextEvent event)  {
    try {
        System.out.println("closing HSQLDB connection.");
        DataSource.INSTANCE.close();
    } catch (Exception e) {
        System.out.println("bem tentei, mas...");
        e.printStackTrace();
    }
}
```


Refabricação da WebAppDemo: *Page Controller*

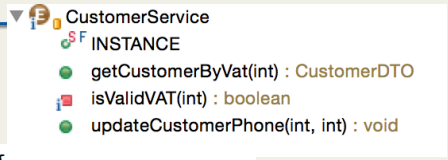




Refabricação da WebAppDemo: *Page Controller*

```
@WebServlet("/UpdateCustomerContactsPageController")
public class UpdateCustomerContactsPageController extends PageController {

    private static final long serialVersionUID = 1L;

    @Override
    protected void process(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        CustomerService cs = CustomerService.INSTANCE;

        CustomerHelper ch = new CustomerHelper();
        request.setAttribute("helper", ch);
        try{
            String vat = request.getParameter("vat");
            String phone = request.getParameter("phone");
            if (isInt(ch, vat, "Invalid VAT number") && isInt(ch, phone, "Invalid phone number")) {
                int vatNumber = intValue(vat);
                int phoneNumber = intValue(phone);
                cs.updateCustomerPhone(vatNumber,phoneNumber);
                ch.fillWithCustomer(cs.getCustomerByVat(vatNumber));
                request.getRequestDispatcher("CustomerInfo.jsp").forward(request, response);
            }
        } catch (ApplicationException e) {
            ch.addMessage("It was not possible to fulfill the request: " + e.getMessage());
            request.getRequestDispatcher("CustomerError.jsp").forward(request, response);
        }
```


Refabricação da WebAppDemo: *Page Controller*

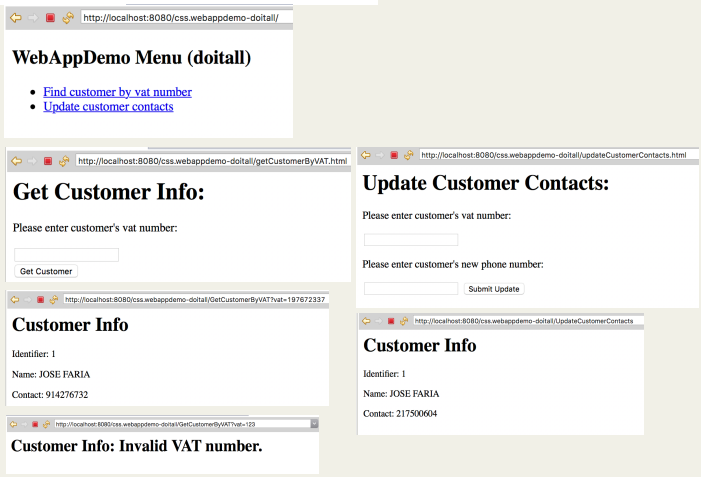



```
public CustomerDTO getCustomerByVat (int vat) throws ApplicationException {
    if (!isValidVAT (vat))
        throw new ApplicationException ("Invalid VAT number: " + vat);
    else try {
        CustomerRowDataGateway customer = new CustomerFinder().getCustomerByVATNumber(vat);
        return new CustomerDTO(customer.getCustomerId(), customer.getVAT(),
                customer.getDesignation(), customer.getPhoneNumber());
    } catch (PersistenceException e) {
            throw new ApplicationException ("Customer with vat number " + vat + " not found.", e);
    }
}

public void updateCustomerPhone(int vat, int phoneNumber) throws ApplicationException {
    if (!isValidVAT (vat))
        throw new ApplicationException ("Invalid VAT number: " + vat);
    else try {
        CustomerRowDataGateway customer = new CustomerFinder().getCustomerByVATNumber(vat);
        customer.setPhoneNumber(phoneNumber);
        customer.updatePhoneNumber();
    } catch (PersistenceException e) {
            throw new ApplicationException ("Customer with vat number " + vat + " not found.", e);
    }
}
```

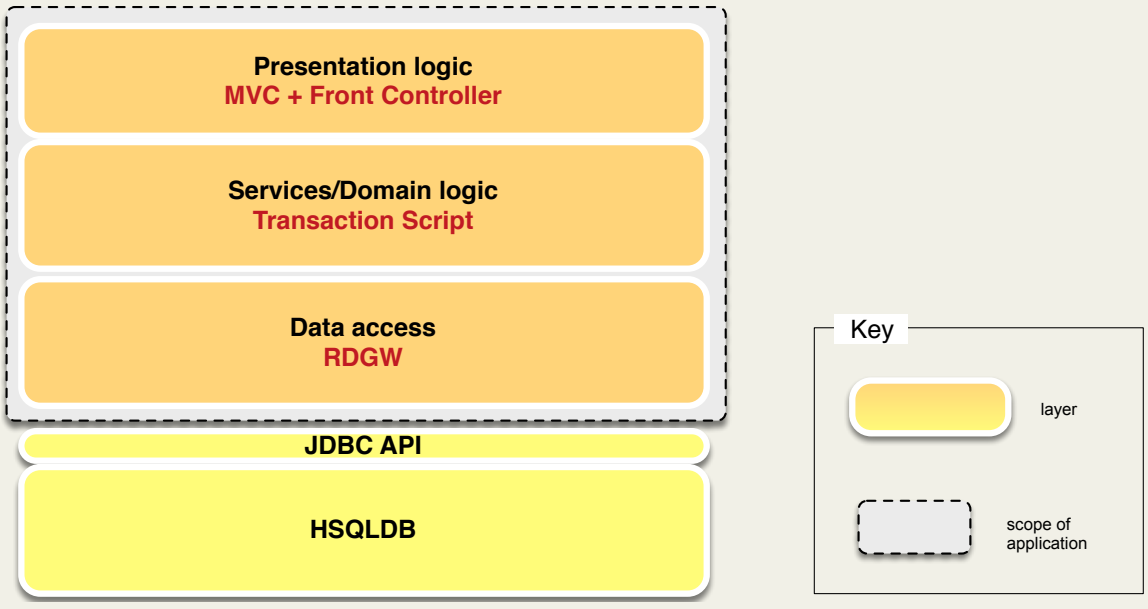
Exercícios

Conceba uma solução de desenho alternativa para a aplicação que

- utilize o padrão *MVC* e o padrão *Front Controller* na camada de apresentação,
- tenha a camada de negócio organizada de acordo com o padrão *Transaction Script*
- tenha a camada de acesso aos dados organizada de acordo com o padrão *Row Data Gateway*.



Exercício 2 (c)



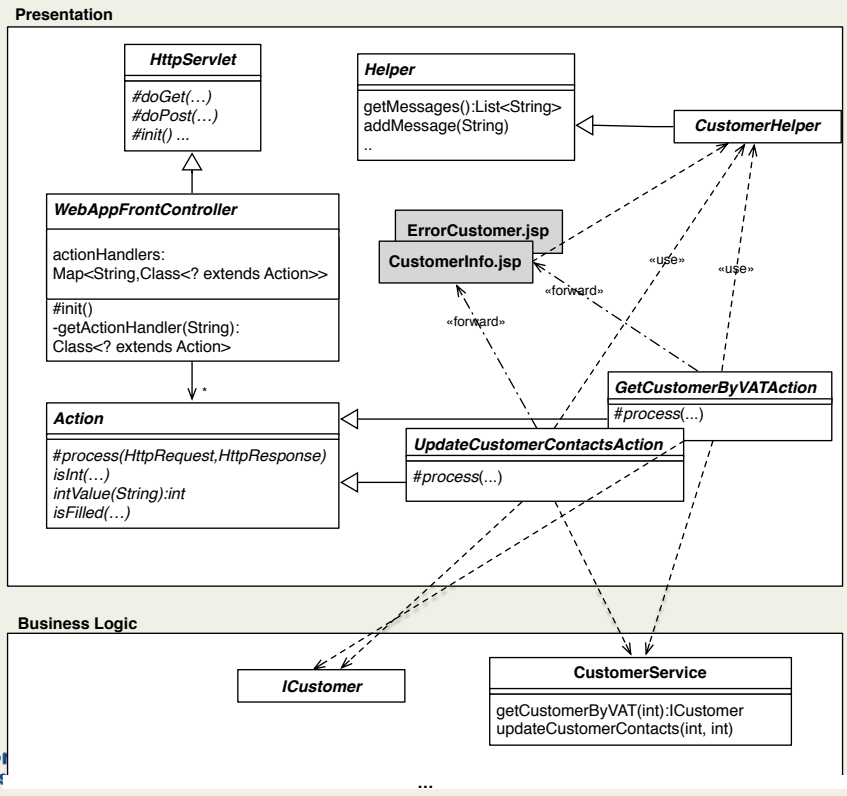
URLs no Page Controller e Front Controller

- **Page Controller**
  - .../css.webappdemo/
  - .../css.webappdemo/getCustomerByVAT?vat=...
  - .../css.webappdemo/updateCustomerContacts
- **Front Controller**
  - .../css.webappdemo/
  - .../css.webappdemo/action/getCustomerByVAT?vat=...
  - .../css.webappdemo/action/updateCustomerContacts

ou

  - .../css.webappdemo?action=getCustomerByVAT&vat=...
  - .../css.webappdemo?action=updateCustomerContacts

Refabricação da WebAppDemo: *Front Controller*



```java
 */
@WebServlet(WebAppFrontController.ACTION_PATH + "/*")
public class WebAppFrontController extends HttpServlet {

    private static final long serialVersionUID = 1L;

    static final String ACTION_PATH = "/action";

    /**
     * Maps http actions to the objects that are going to handle them
     */
    protected HashMap<String, Class<? extends Action>> actionHandlers;

    private Properties appProperties;

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws Servl
        String actionURL = request.getPathInfo();
        Class<?> actionClass = getActionHandlerAddress(ACTION_PATH + actionURL);
        Action actionCommand;
        try {
            actionCommand = (Action) actionClass.newInstance();
        } catch (Exception e) {
            actionCommand = new UnknownAction();
        }
        actionCommand.process(request, response);
    }
```



```java
private Class<?> getActionHandlerAddress(String action) {
    Class<?> result = actionHandlers.get(action);
    if (result == null)
        result = actionHandlers.get("unknownAction");
    return result;
}


/* (non-Javadoc)
 * @see javax.servlet.GenericServlet#init()
 */
@Override
public void init() {
    String propertiesFileName = "/app.properties";
    actionHandlers = new HashMap<>();
    actionHandlers.put("unknownAction", UnknownAction.class);
    appProperties = new Properties();
    try {
        appProperties.load(getClass().getResourceAsStream(propertiesFileName));
        for (Entry<Object, Object> keyValue : appProperties.entrySet()) {
            if (keyValue.getKey() instanceof String) {
                String key = (String) keyValue.getKey();
                if(key.startsWith("appRoot"))
                    actionHandlers.put(key.substring(7), loadClass(keyValue.getValue()));
            }
        }
    } catch (Exception e) {
        // It was not able to load properties file.
        // Bad luck, all action will be dispatched to the UnknownAction
    }
}
```

Refabricação da WebAppDemo: Front Controller

getCustomerByVAT.html

```
<HTML>
<HEAD>
<TITLE>Enter Name (version: frontcontroller)</TITLE>
</HEAD>
<BODY>
<H1>Get Customer Info:</H1>

<FORM ACTION="action/getCustomerByVAT" METHOD="GET">
<P>Please enter customer's vat number:</P>
<INPUT TYPE="TEXT" NAME="vat">
<BR>
<INPUT TYPE="SUBMIT" VALUE="Get Customer">
</FORM>

</BODY>
</HTML>

```

app.properties

```
# URL wiring

appRoot/action/getCustomerByVAT = css.webappdemo.webpresentation.actions.GetCustomerByVATAction
appRoot/action/updateCustomerContacts = css.webappdemo.webpresentation.actions.UpdateCustomerContactsAction

```


Refabricação da WebAppDemo: Front Controller

```
public class GetCustomerByVATAction extends Action {

    @Override
    public void process(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        CustomerService cs = CustomerService.INSTANCE;

        CustomerHelper ch = new CustomerHelper();
        request.setAttribute("helper", ch);
        try {
            String vat = request.getParameter("vat");
            if (isInt(ch, vat, "Invalid VAT number")) {
                int vatNumber = intValue(vat);
                ch.fillWithCustomer(cs.getCustomerByVat(vatNumber));
                request.getRequestDispatcher("../CustomerInfo.jsp").forward(request, response);
            }
        }catch (ApplicationException e) {
            ch.addMessage("It was not possible to fulfill the request: " + e.getMessage());
            request.getRequestDispatcher("../CustomerError.jsp").forward(request, response);
        }
    }
}
```

Casos de uso com estado

- Efetuar uma venda no *SaleSys*
  - o utilizador fornece primeiro o vat do cliente, que o sistema valida
  - e depois acrescenta à venda uma ou mais entradas, cada uma com determinada quantidade de um produto

```
public class ProcessSaleHandler {

    /**
     * Entity manager factory for accessing the persistence service
     */
    private EntityManagerFactory emf;

    /**
     * The current sale
     */
    private Sale currentSale;
```

- O *SaleService* usa um objeto *ProcessSaleHandler* que mantém no estado a venda corrente
- É preciso garantir que o mesmo objeto *SaleService* é usado nos vários pedidos que pertencem à mesma sessão HTTP

```
public class CreateSaleAction extends Action {

    SaleService saleService = app.getSaleService();
    request.getSession().setAttribute("saleService", saleService);
```