Chapter 2

How to structure a web application with the MVC pattern
Objectives

Knowledge
1. Describe the Model 1 pattern.
2. Describe the Model 2 (MVC) pattern.
3. Explain how the MVC pattern can improve application development.
4. Distinguish between the HTML and CSS for a web page.
5. Distinguish between the code for a servlet and a JSP.
6. Explain why you typically use both servlets and JSPs in a Java web application.
7. Describe the purpose of the deployment descriptor in a web application.
8. Describe the purpose of a JavaBean within a web application.
The Model 1 pattern

Note no servlet here, just JSP OK for single-page apps, but not recommended for multi-page apps
Example: basicjsp/date_el.jsp

<jsp useBean id="date" class = "java.util.Date"/>

...HTML...
Today’s date is ${date}
...HTML...

Browser

HTTP request

JSP
thanks.jsp

HTTP response

Java classes
User.class
UserDB.class

Data store
The Model 2 (MVC) pattern

The servlet is the controller, JSP provides the view. This is the recommended way for non-trivial web apps.
Concepts and terminology

- The *Model 1 pattern* uses JSPs to handle all of the processing and presentation for the application.
- The *Model 2 pattern* separates the code into a model, a view, and a controller. As a result, it’s also known as the *Model-View-Controller (MVC)* pattern.
- The *model* consists of business objects like the User object.
- The *view* consists of HTML pages and JSPs.
- The *controller* consists of servlets.
- The *data access layer* consists of classes like the UserDB class that read and write business objects like the User object to and from the data store.
- Try to construct each layer so it’s as independent as possible.
The HTML page that gets data from the user
The JSP that displays the data

Thanks for joining our email list

Here is the information that you entered:

Email: joel@murach.com
First Name: Joel
Last Name: Murach

To enter another email address, click on the Back button in your browser or the Return button shown below.

Return
Two view files: index.html and thanks.jsp

Servlet: processes form submission, forwards to thanks.jsp
User requests app at ch02email/emailList

User: fills in form
Browser: put user input and hidden field into params in GET/POST request to ch02email/emailList

User: see response

Server

tomcat: process GET /ch02email/, Call servlet’s doGet method, Servlet forwards to index.html

tomcat: process (GET or POST) GET /ch02email/emailList?... Call servlet’s doGet method with params in request object Servlet sets up user variable and forwards to thanks.jsp
The index.html file

<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Murach's Java Servlets and JSP</title>
    <link rel="stylesheet" href="styles/main.css" type="text/css"/>
</head>
<body>
    <h1>Join our email list</h1>
    <p>To join our email list, enter your name and email address below.</p>
<form action="emailList" method="post">
    <input type="hidden" name="action" value="add">

    <label>Email:</label>
    <input type="email" name="email" required><br>

    <label>First Name:</label>
    <input type="text" name="firstName" required><br>

    <label>Last Name:</label>
    <input type="text" name="lastName" required><br>

    <label>&nbsp;</label>
    <input type="submit" value="Join Now" id="submit">
</form>
</body>
</html>
The main.css file

```css
body {
    font-family: Arial, Helvetica, sans-serif;
    font-size: 11pt;
    margin-left: 2em;
    margin-right: 2em;
}

h1 {
    color: teal;
}

label {
    float: left;
    width: 6em;
    margin-bottom: 0.5em;
}

input[type="text"], input[type="email"] {
    width: 15em;
    margin-left: 0.5em;
    margin-bottom: 0.5em;
}

br {
    clear: both;
}

#submit {
    margin-left: 0.5em;
}
```

Note: we’re not covering details of CSS
The EmailListServlet class

```java
package murach.email;

import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import murach.business.User;
import murach.data.UserDB;

public class EmailListServlet extends HttpServlet {

    @Override
    protected void doPost(HttpServletRequest request,
                          HttpServletResponse response)
        throws ServletException, IOException {

        String url = "index.html";

        // get current action
        String action = request.getParameter("action");
        if (action == null) {
            action = "join";  // default action
        }
    }
```
The EmailListServlet class (continued)

    // perform action and set URL to appropriate page
    if (action.equals("join")) {
        url = "/index.html";    // the "join" page
    }
    else if (action.equals("add")) {
        // get parameters from the request
        String firstName = request.getParameter("firstName");
        String lastName = request.getParameter("lastName");
        String email = request.getParameter("email");

        // store data in User object and save User object in db
        User user = new User(firstName, lastName, email);
        UserDB.insert(user);

        // set User object in request object and set URL
        request.setAttribute("user", user);
        url = "/thanks.jsp";   // the "thanks" page
    }

    // forward request and response objects to specified URL
    getServletContext()
        .getRequestDispatcher(url)
        .forward(request, response); 

    }
The EmailListServlet class (continued)

```java
@Overrride
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
        doPost(request, response);
    }
```

Parameters

- `pageContext`
- `request`
- `session`
- `Request attributes`
- `Session attributes`
The web.xml file

```xml
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="3.1"
    xmlns="http://xmlns.jcp.org/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
                         http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd">

    <servlet>
        <servlet-name>EmailListServlet</servlet-name>
        <servlet-class>murach.email.EmailListServlet</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>EmailListServlet</servlet-name>
        <url-pattern>/emailList</url-pattern>
    </servlet-mapping>

    <session-config>
        <session-timeout>30</session-timeout>
    </session-config>

</web-app>
```
The web.xml file (continued)

```xml
<welcome-file-list>
  <welcome-file>index.html</welcome-file>
  <welcome-file>index.jsp</welcome-file>
</welcome-file-list>

</web-app>
```
The User class

```java
package murach.business;

import java.io.Serializable;

public class User implements Serializable {
    private String firstName;
    private String lastName;
    private String email;

    public User() {
        firstName = "";
        lastName = "";
        email = "";
    }

    public User(String firstName, String lastName, String email) {
        this.firstName = firstName;
        this.lastName = lastName;
        this.email = email;
    }
}
```
The User class (continued)

```java
    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }
```
The thanks.jsp file

```html
<!doctype html>
<html>
<head>
    <meta charset="utf-8">
    <title>Murach's Java Servlets and JSP</title>
    <link rel="stylesheet" href="styles/main.css" type="text/css"/>
</head>

<body>
    <h1>Thanks for joining our email list</h1>
    <p>Here is the information that you entered:</p>
    
    <label>Email:</label>
    <span>${user.email}</span>
    <br>
    <label>First Name:</label>
    <span>${user.firstName}</span>
    <br>
    <label>Last Name:</label>
    <span>${user.lastName}</span>
</body>
```

The thanks.jsp file (continued)

<p>To enter another email address, click on the Back button in your browser or the Return button shown below.</p>

<form action="" method="get">
    <input type="hidden" name="action" value="join">
    <input type="submit" value="Return">
</form>

</body>
</html>
Types of files in the MVC pattern

- An HTML file contains tags that define the content of the web page.
- A CSS (Cascading Style Sheet) file contains the formatting for the web pages.
- Servlets contain Java code for a web application. When a servlet controls the flow of the application, it’s known as a controller.
- The web.xml file, or deployment descriptor (DD), describes how the web application should be configured when it’s deployed.
- A JavaBean, or bean, is a Java class that (1) provides a zero-argument constructor, (2) provides get and set methods for all of its instance variables, and (3) implements the Serializable or Externalizable interface.
- A JavaServer Page (JSP) consists of special Java tags such as Expression Language (EL) tags that are embedded within HTML code. An EL tag begins with a dollar sign ($).
Making this app into a Model 1 app i.e. use only JSP, no servlet

We can use almost the same index.html, just change action= target

The index.html file

```html
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Murach's Java Servlets and JSP</title>
    <link rel="stylesheet" href="styles/main.css" type="text/css"/>
</head>
<body>
    <h1>Join our email list</h1>
    <p>To join our email list, enter your name and email address below.</p>
</body>
</html>
```
The index.html file (continued)

Pure JSP: action=thanks.jsp

```html
<form action="emailList" method="post">
    <input type="hidden" name="action" value="add">

    <label>Email:</label>
    <input type="email" name="email" required><br>

    <label>First Name:</label>
    <input type="text" name="firstName" required><br>

    <label>Last Name:</label>
    <input type="text" name="lastName" required><br>

    <label>&nbsp;</label>
    <input type="submit" value="Join Now" id="submit">
</form>
</body>
</html>
```
The EmailListServlet class

```java
package murach.email;

import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import murach.business.User;
import murach.data.UserDB;

public class EmailListServlet extends HttpServlet {

    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

        String url = "/index.html";

        // get current action
        String action = request.getParameter("action");
        if (action == null) {
            action = "join";  // default action
        }
    }
```
The EmailListServlet class (continued)

```java
// perform action and set URL to appropriate page
if (action.equals("join")) {
    url = "/index.html"; // the "join" page
}
else if (action.equals("add")) {
    // get parameters from the request
    String firstName = request.getParameter("firstName");
    String lastName = request.getParameter("lastName");
    String email = request.getParameter("email");

    // store data in User object and save User object in db
    User user = new User(firstName, lastName, email);
    UserDB.insert(user);

    // set User object in request object and set URL
    request.setAttribute("user", user);
    url = "/thanks.jsp"; // the "thanks" page
}

// forward request and response objects to specified URL
getServletContext()
    .getRequestDispatcher(url)
    .forward(request, response);
```

### Just start from index.html

```xml
<c:useBean with setProperty>
```

### Tricky part

```java
// set User object in request object and set URL
request.setAttribute("user", user);
url = "/thanks.jsp"; // the "thanks" page
```

### Not needed

```java
getServletContext()
    .getRequestDispatcher(url)
    .forward(request, response);
```
The EmailListServlet class (continued)

```java
@Override
protected void doGet(HttpServletRequest request,
                       HttpServletResponse response)
                       throws ServletException, IOException {
        doPost(request, response);
    }
```
The web.xml file

```xml
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="3.1"
    xmlns="http://xmlns.jcp.org/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
    http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd">

    <servlet>
        <servlet-name>EmailListServlet</servlet-name>
        <servlet-class>murach.email.EmailListServlet</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>EmailListServlet</servlet-name>
        <url-pattern>/emailList</url-pattern>
    </servlet-mapping>

    <session-config>
        <session-timeout>30</session-timeout>
    </session-config>

</web-app>
```
The web.xml file (continued)

```xml
<welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.jsp</welcome-file>
</welcome-file-list>
</web-app>
```
Still in use, created by `<c:useBean>`

**The User class**

```java
package murach.business;

import java.io.Serializable;

public class User implements Serializable {
    private String firstName;
    private String lastName;
    private String email;

    public User() {
        firstName = "";
        lastName = "";
        email = "";
    }

    public User(String firstName, String lastName, String email) {
        this.firstName = firstName;
        this.lastName = lastName;
        this.email = email;
    }
}
```
The User class (continued)

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }
This changes a lot, does the controller work as well as the second page view…

The thanks.jsp file

```html
<!doctype html>
<html>
<head>
    <meta charset="utf-8">
    <title>Murach's Java Servlets and JSP</title>
    <link rel="stylesheet" href="styles/main.css" type="text/css"/>
</head>

<body>
    <h1>Thanks for joining our email list</h1>
    <p>Here is the information that you entered:

    <label>Email:</label>
    <span>${user.email}</span>
    <br>
    <label>First Name:</label>
    <span>${user.firstName}</span>
    <br>
    <label>Last Name:</label>
    <span>${user.lastName}</span>
</body>
```

Add `<c:useBean>` to create `User` object with properties from request parameters and establish it as a variable “user”

How do we do the call `UserDB.insert(user)`???
The thanks.jsp file (continued)

To enter another email address, click on the Back button in your browser or the Return button shown below.

```html
<form action="" method="get">
    <input type="hidden" name="action" value="join">
    <input type="submit" value="Return">
</form>
```

Ways to call UserDB.insert(user):

- Use a scriptlet: `<%=UserDB.insert(user)%>` (old JSP!!)
- Add a method to User: getInsertStatus() that does the action and add $user.insertStatus to page (not nice: getters shouldn’t modify things!)
- Make the setters smart about detecting all the data is available and if so, doing the action in the setter.
- None of these ways is really a good solution.
JSP-only (Model 1) Implementation
Two files: index.html and thanks.jsp

Form’s action=thanks.jsp to submit form to JSP

Symbolizing controller code in the JSP
Conclusion: MVC is the way to go for any app that uses a DB, our concentration