

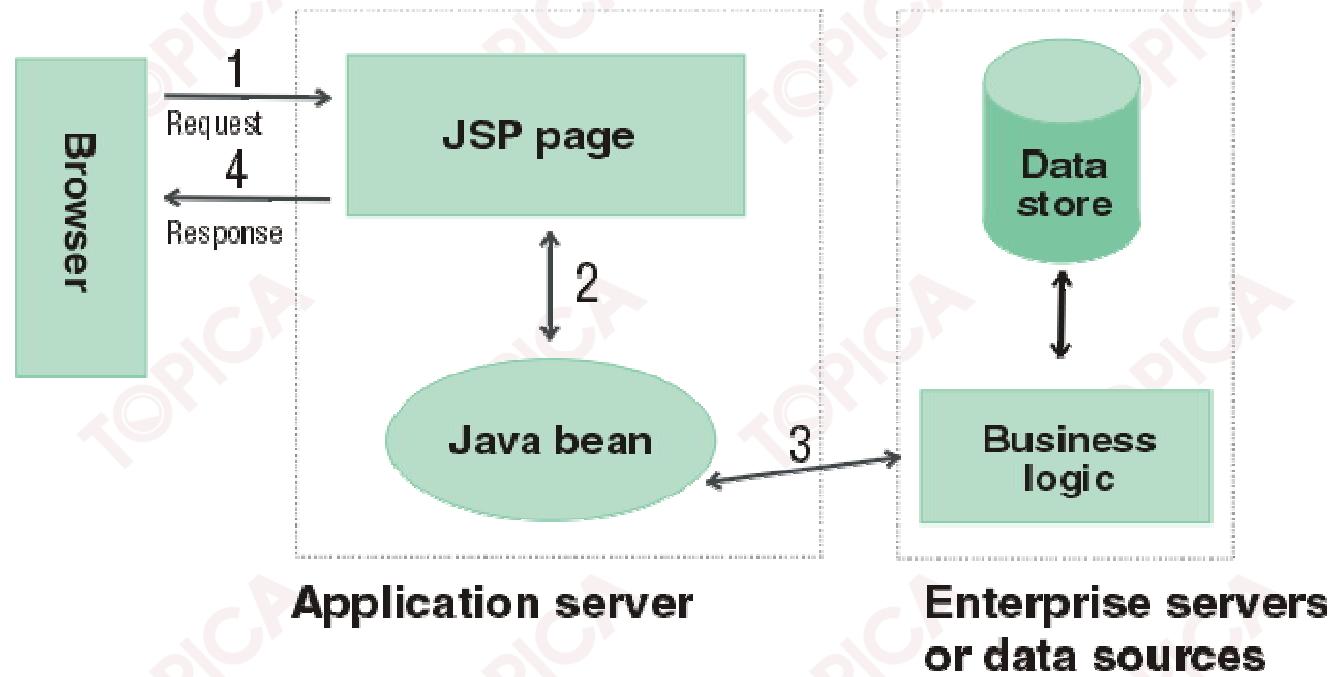
BÀI 4

SỬ DỤNG JAVABEAN VÀ JAVA MAIL TRONG JSP

ThS. Phan Thanh Toàn

TÌNH HUỐNG DẪN NHẬP

Nam đang cần thiết kế Website quản lý học tập bằng ngôn ngữ JSP, và đang xây dựng một lớp users để quản lý thông tin về những người sử dụng hệ thống.



Làm thế nào xây dựng được các lớp với các trường, thuộc tính, phương thức và đưa các lớp đã xây dựng vào sử dụng trong các trang JSP?

MỤC TIÊU

- Trình bày khái niệm, các thành phần của JavaBean.
- Trình bày các phương pháp tạo JavaBean trên các môi trường.
- Trình bày được các lợi điểm của JavaBean trong xây dựng các trang JSP.
- Sử dụng được JavaBean trong thiết kế và xây dựng ứng dụng web đơn giản.

NỘI DUNG

- 1 Giới thiệu về JavaBean
- 2 Các thành phần của JavaBean
- 3 Các cách sử dụng JavaBean
- 4 Lợi điểm của JavaBean
- 5 Sử dụng các thẻ JSP liên quan đến JavaBean
- 6 Phạm vi hoạt động và các loại biến trong JavaBean
- 7 JavaMail API

1. GIỚI THIỆU VỀ JAVABEAN

- JavaBean là Software Component được viết bởi ngôn ngữ Java.
- JavaBean tạo ra các component độc lập với platform.
- JavaBean có khả năng nhúng vào các component, application hay applet khác nhau.
- Sự khác biệt chủ yếu giữa JavaBean và JavaClass thông thường là JavaBean được vận dụng theo cơ chế Serializable (các giá trị của các thuộc tính trong Bean được đưa tới các phương thức instance của Bean).
- JSP truy cập JavaBean qua các tag action và nhận kết quả trả về mà không cần biết cấu trúc của JavaBean và cách thức xử lý của nó.
- JavaBean cài đặt các phương thức xử lý và không hiển thị khi thực hiện các xử lý.
- JavaBean là JavaClass tuân thủ 3 yếu tố sau:
 - Phải có một constructor không có tham số (mặc định có sẵn nếu không implement). Constructor này được gọi khi element của JSP tạo Bean.
 - Các thuộc tính (field) của Bean không được khai báo public.
 - Việc truy xuất các thuộc tính của Bean sẽ thông qua phương thức **getXxx** hay **setXxx** (accessor method) đối với các thuộc tính cần lưu trữ (persistent).
 - Lưu ý:

1. GIỚI THIỆU VỀ JAVABEAN (tiếp theo)

- Các thuộc tính khai báo với ký tự đầu là chữ thường và các accessor sẽ bắt đầu bằng chữ in hoa (ví dụ: length – getLength và setLength).
 - Các thuộc tính có kiểu dữ liệu là boolean thì phương thức gọi chúng sẽ có dạng **isXxx** thay vì **getXxx**.
 - JSP actions sẽ truy cập phương thức get/set/is để truy cập Bean.
- Khác biệt giữa JavaBean và JavaClass thông thường:
- JavaBean cũng là một lớp Java nhưng có thêm các đặc điểm sau:
 - JavaBean phải là lớp cụ thể (instantiable, concrete), không thể là lớp trừu tượng hay là interface.
 - Phải có constructor chuẩn (constructor không tham số) để IDE tạo ra đối tượng mặc định (vì trong IDE, không thể tạo ra một đối tượng với constructor không chuẩn).
 - Buộc phải tuân tự hóa bằng cách Implements Interface Serializable. Bằng cách tuân tự hóa, trạng thái đối tượng có thể được ghi lên đĩa (là một chuỗi các byte).
 - Trong JavaBean phải có 3 Property sau:
 - public void setPropertyName (<Property_Type> value);
 - public <Property_Type> getPropertyName ();
 - public boolean isPropertyName ().

1. GIỚI THIỆU VỀ JAVABEAN (tiếp theo)

- Ví dụ:

```
class Tower

{
    private float height;

    public Tower() { height = float (10.5); }

    public setHeight (float h) { height = h; }

    public getHeight () { return height; }

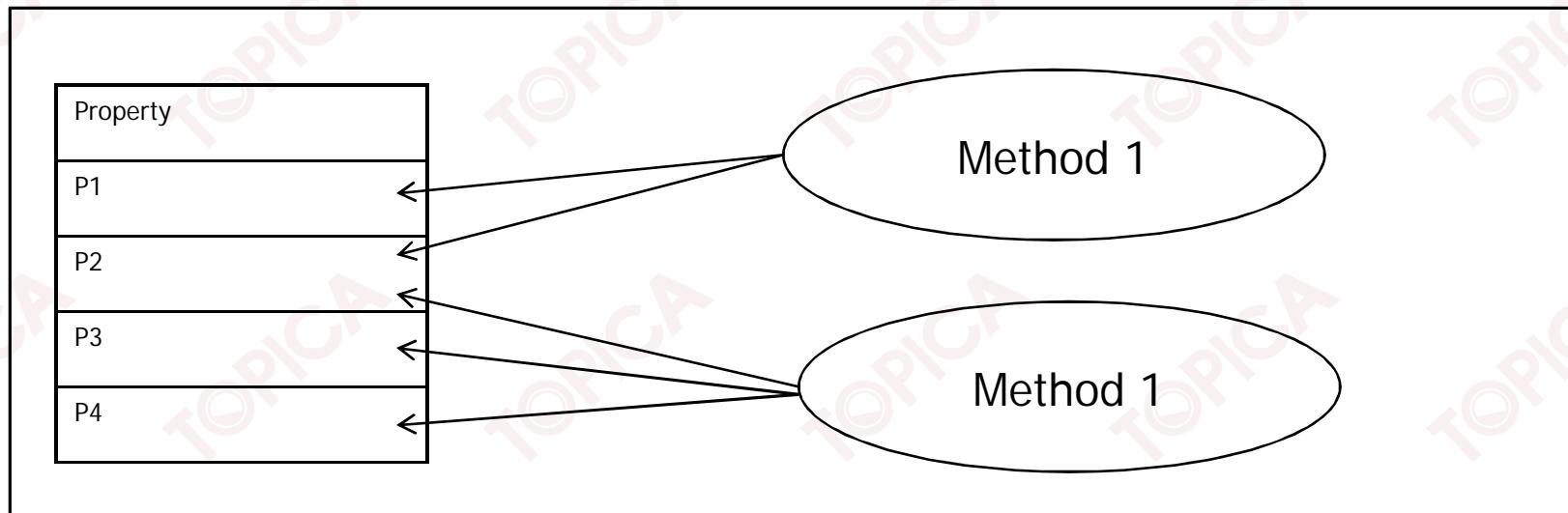
    public isGreaterHeight ( float initH, float finalH) {
        return (finalH-initH)>0 ? true : false;
    }
}
```

2. CÁC THÀNH PHẦN CỦA JAVABEAN

- Cấu trúc của JavaBean;
- Listener Bean;
- Sử dụng thuộc tính có chỉ số trong JavaBean.

2.1. CẤU TRÚC CỦA JAVABEAN

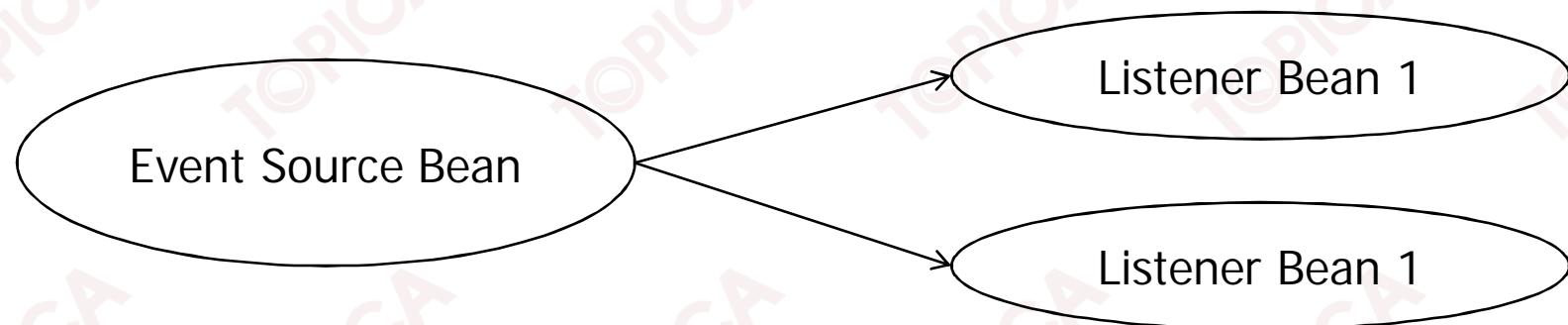
- Mỗi JavaBean thường gồm các thành phần là các trường (Field), các thuộc tính (Property), và các phương thức (Method).



- Các hành vi của Bean cũng như các lớp Java thông thường khác, cũng sử dụng các bổ từ truy cập: Private, protected, public.

2.2. LISTENER BEAN

- Bean giao tiếp với bên ngoài thông qua các sự kiện (event).
- Một Bean cũng sẽ ủy thác xử lý sự kiện cho các Listener. Các Listener này có thể là các Bean khác (gọi là Listener Bean).



Lớp hỗ trợ Bean: PropertyChangeSupport

- Lớp này được dùng để quản lý một danh sách các listeners khi trạng thái Bean thay đổi. Trong Bean thường có một đối tượng thuộc tính thuộc lớp này.
- Constructor: PropertyChangeSupport (Object source).
- Các hành vi của lớp này liên quan đến việc ủy thác xử lý sự kiện thường dùng:
 - public void addPropertyChangeListener (PropertyChangeListener);
 - public void removePropertyChangeListener (PropertyChangeListener);
 - public void firePropertyChange (String propertyName, Object oldVale, Object newValue);

2.2. LISTENER BEAN (tiếp theo)

Ví dụ: Xây dựng javaBean: SimpleBean với 2 field là message và num. Các Property: setMessage, getMessage và setNum, getNum như sau:

```
public class SimpleBean {  
    private String message = "First Bean";  
    private int num;  
  
    public String getMessage() {  
        return message; }  
  
    public void setMessage(String message) {  
        this.message=message; }  
  
    public int getNum() {  
        return num; }  
  
    public void setNum(int n) {  
        num=n; }  
}
```

2.2. LISTENER BEAN (tiếp theo)

Sử dụng scriptlet trong trang JSP:

```
<h1>JSP with Java Bean</h1>
<jsp:useBean id="msg" class="JBean.SimpleBean"/>
Init message (getProperty) :
<jsp:getProperty name="msg" property="message"/>
<br/>
(Scriptlet) : <%= msg.getMessage() %> <br/>
Set message (setProperty) :
<jsp:setProperty name="msg" property="message" value="Hello"/>
(Scriptlet) : <%=msg.getMessage()%>
```

2.3. SỬ DỤNG THUỘC TÍNH CÓ CHỈ SỐ TRONG JAVABEAN

- Khi một trường trong lớp JavaBean là một mảng ta phải sử dụng một chỉ số kèm với lớp và phải kiểm tra khi nào chỉ số vượt quá phạm vi giới hạn của mảng?
- Ví dụ: Xây dựng lớp JavaBean sử dụng thành phần là một mảng kiểu int:

```
public class ArrayClass
{
    private int items[]={1,2,3,4,5};
    private PropertyChangeSupport pcs;
    public ArrayClass()
    {
        pcs=new PropertyChangeSupport(this) ;
    }
    public int getItems (int i)
    {
        return items[i];
    }
    public void setItems(int i, int x)throws
        ArrayIndexOutOfBoundsException
    {
        this.items[i]=x;
        pcs.firePropertyChange( "Items" , i, x );
    }
}
```

2.3. SỬ DỤNG THUỘC TÍNH CÓ CHỈ SỐ TRONG JAVABEAN (tiếp theo)

```
public String ValuesString()
{
    String S= " ";
    for( int i=0; i<5;i++)
        S+=items[ i ] + ((i<4)?", ":" ");
    return S;
}

public void addPropertyChangeListener(PropertyChangeListener L)
{
    pcs.addPropertyChangeListener(L);
}

public void
removePropertyChangeListener(PropertyChangeListener L)
{
    pcs.removePropertyChangeListener(L);
}
}
```

2.3. SỬ DỤNG THUỘC TÍNH CÓ CHỈ SỐ TRONG JAVABEAN (tiếp theo)

- Sử dụng lớp JavaBean trong trang JSP như sau:

```
<h1>Example of array in JavaBean</h1>

<jsp:useBean id="arrayExample" class="Jbean.ArrayClass" />

<%
String x = arrayExample.ValuesString();
out.println(x);
%>
```

3. CÁC CÁCH SỬ DỤNG JAVABEAN

- Tạo JavaBean với môi trường phát triển ứng dụng:
 - Các môi trường phát triển ứng dụng Java trực quan như JBuilder (của công ty Borland), NetBeans (của Sun) cho phép tạo Bean với môi trường trực quan.
 - Có thể mua hoặc download từ Internet
- Tạo JavaBean bằng thủ công: Đây là cách tạo Bean bằng cách viết code thủ công với môi trường JDK thông dụng.
- Tạo Bean phụ thuộc platform:
 - Việc này nhằm tạo ra các Bean kết hợp được với các phần tử trong môi trường khác (thí dụ như Bean dùng chung với các phần tử ActiveX của Microsoft). Để tạo được các Bean có đặc điểm này, một lớp trung gian đóng vai trò cầu nối (component bridge) phải được dùng.
 - Tham khảo về cầu nối này tại:
<http://java.sun.com/products/javabeans/software/>

CÂU HỎI THẢO LUẬN



Cấu trúc JavaBean gồm những thành phần nào?

4. LỢI ĐIỂM CỦA JAVABEAN

- Viết một lần, sử dụng mọi nơi “Write once, run anywhere”.
- Các thuộc tính, sự kiện, hành vi của Bean được thể hiện trực quan trong các IDE và người lập trình có thể điều khiển chúng.
- Bean có thể được xây dựng để chạy tốt trên nhiều khu vực (locale) khác nhau để trở thành các phần tử toàn cục.
- Các phần mềm tiện ích có thể giúp cấu hình Bean. Các phần mềm này có thể cần thiết khi thiết kế ứng dụng nhưng không cần thiết trong môi trường thực thi (vì JVM mới là môi trường thực thi).
- Các thiết lập cấu hình cho Bean (thiết lập thuộc tính) cho phép tiết kiệm trong lưu trữ và phục hồi trạng thái của Bean.
- Cho phép đăng ký và nhận các sự kiện từ các đối tượng khác (Listener Bean) cũng như ủy thác xử lý sự kiện cho các đối tượng khác.
- Thuộc tính, sự kiện, hành vi của Bean được thể hiện trực quan trong các IDE và người lập trình có thể điều khiển chúng.

5. SỬ DỤNG CÁC THẺ JSP LIÊN QUAN ĐẾN JAVABEAN

- Khai báo sử dụng Bean;
- Áp trị thuộc tính cho Bean;
- Lấy trị thuộc tính của Bean.

5.1. KHAI BÁO SỬ DỤNG BEAN

- Cú pháp:

```
<jsp:useBean id="BeanName"  
    class="fully_qualified_classname" scope="scope" type= "type_spec" />
```

Hoặc:

```
<jsp:useBean id="BeanName"  
    class="fully_qualified_classname" scope="scope">  
    <jsp:setProperty .../>  
</jsp:useBean>
```

- id: Tên nhận diện cho Bean.
- fully_qualified_classname: Tên đầy đủ của lớp Bean này, có thể phải chỉ định gói chứa Bean này nằm trong một package (hoặc import gói này vào trang JSP).
- scope: chỉ định tầm vực cho Bean (vùng mà Bean có ý nghĩa = truy cập hợp lệ).
- scope = “page|session|request|application”.
- type: Là thuộc tính optional, đặc tả loại Class: superclass, interface hay là chính lớp này. Trị mặc định là trị của thuộc tính Class.
- Ví dụ:

```
<jsp:useBean id="locales" scope="application"  
    class="mypkg.MyLocales"/>
```

5.2. ÁP TRỊ THUỘC TÍNH CHO BEAN

Nguồn giá trị gán vào thuộc tính của Bean	Cú pháp
Hằng chuỗi	<jsp:setProperty name="beanName" Property="propName" value="string constant" />
Tham số của đối tượng request	<jsp:setProperty name="beanName" Property="propName" param="paramName" />
Tham số của đối tượng request trùng với một hoặc mọi thuộc tính của Bean	<jsp:setProperty name="beanName" Property="propName" <jsp:setProperty name="beanName" Property="*" />
Biểu thức	<jsp:setProperty name="beanName" Property="propName" value="expression" /> <jsp:setProperty name="beanName" Property="propName" <jsp:attribute name=" value " Expression </jsp:attribute> </jsp:setProperty>

5.2. ÁP TRỊ THUỘC TÍNH CHO BEAN (tiếp theo)

- Name: Tên của Bean đã được khai báo trong thuộc tính id trong thẻ <jsp:useBean>.
- Để gán trị vào thuộc tính của Bean, Bean phải có phương thức setPropertyName.
- Nếu param= " " thì việc gán trị không làm ảnh hưởng đến Bean.
- paramName trong là tên tham số của đối tượng request để lấy trị một tham số từ đối tượng request đưa vào một thuộc tính của Bean:

```
<jsp:setProperty name="BeanName"  
property="propName" value="<% = request.getParameter("FromParam")%>" />
```

5.2. ÁP TRỊ THUỘC TÍNH CHO BEAN (tiếp theo)

- Các kiểu dữ liệu trong JavaBean

Property Type	Conversion on String Value
Bean Property	Uses <code>setAsText(string-literal)</code>
boolean or Boolean	As indicated in <code>java.lang.Boolean.valueOf(String)</code>
byte or Byte	As indicated in <code>java.lang.Byte.valueOf(String)</code>
char or Character	As indicated in <code>java.lang.String.charAt(0)</code>
double or Double	As indicated in <code>java.lang.Double.valueOf(String)</code>
int or Integer	As indicated in <code>java.lang.Integer.valueOf(String)</code>
float or Float	As indicated in <code>java.lang.Float.valueOf(String)</code>
long or Long	As indicated in <code>java.lang.Long.valueOf(String)</code>
short or Short	As indicated in <code>java.lang.Short.valueOf(String)</code>
Object	New <code>String(string-literal)</code>

5.3. LẤY TRỊ THUỘC TÍNH CỦA BEAN

Cú pháp:

```
<jsp:getProperty name="BeanName" property="propName"/>
```

Ví dụ: Giả sử đã tạo JavaBean là BoxBean với các thuộc tính height, length, width ta sẽ khai báo và sử dụng các thuộc tính của JavaBean như sau:

```
<jsp:useBean id="BoxBean1" scope="request" class="JBean.BoxBean"/>
<jsp:setProperty name="BoxBean1" property="height" value="20"/>
<br>Truy cap cac thong tin cua BoxBean
<br>Chieu cao: <jsp:getProperty name="BoxBean1" property="height"/>
<br>Chieu dai: <jsp:getProperty name="BoxBean1" property="length"/>
<br>Chieu rong <jsp:getProperty name="BoxBean1" property="width"/>
```

6. TÂM VỰC CỦA BEAN

- Bean có tầm vực trong một trang (page scope, page context);
- Bean có tầm vực một yêu cầu (request scope);
- Bean có tầm vực một session (session scope);
- Bean có tầm vực cả ứng dụng (application scope).

6.1. BEAN CÓ TÂM VỰC TRONG MỘT TRANG

- Bean chỉ có ý nghĩa (truy xuất được) trong trang chứa Bean → Biến đổi tương này được khai báo cục bộ trong hành vi `_jspService(...)` của servlet tương ứng.
- Mọi tham khảo đến đổi tương có page context sẽ bị giải phóng khi đáp ứng được truyền cho client, nghĩa là hành vi `Servlet.service(...)` thực thi xong.
- Đổi tương bị ràng buộc là `javax.servlet.jsp.PageContext`.
- Các đổi tương có page context được lưu trong đổi tương `PageContext`.
- Thuộc tính `scope="page"` trong khai báo sử dụng Bean của JSP sẽ được thông dịch thành `PageContext.PAGE_SCOPE`.
- Ví dụ sử dụng lớp `BoxBean` để hiển thị thông tin:

```
<jsp:useBean id=«BoxBean1» scope=«page» class=«Jbean.BoxBean» />
<jsp:setProperty name=«BoxBean1» property=«length» value=«10» />
<br>Cac thuoc tinh cua Box la: <br>
Chieu dai: <jsp:getProperty name=«BoxBean1» property=«length» />
Chieu rong: <jsp:getProperty name=«BoxBean1» property=«width» />
Chieu cao: <jsp:getProperty name=«BoxBean1» property=«height» />
```

6.2. BEAN CÓ TÂM VỰC MỘT YÊU CẦU

- Bean chỉ có thể được truy xuất được trong quá trình xử lý cùng một yêu cầu, tham khảo đến Bean chỉ mất đi sau khi xử lý hoàn tất yêu cầu. Nghĩa là, trong quá trình xử lý mà có forward sang một tài nguyên khác lúc runtime thì Bean này vẫn còn trong tầm vực.
- Các tham khảo đến Bean được lưu trữ trong đối tượng request.
- Đối tượng bị ràng buộc là javax.servlet.jsp.PageContext.
- Được dùng để chia sẻ thông tin giữa các tài nguyên của cùng một request.
- Thuộc tính scope="request" trong khai báo sử dụng Bean của JSP sẽ được thông dịch thành PageContext.REQUEST_SCOPE.
- **Ví dụ:** Sử dụng Bean: BoxBean với phạm vi truy cập là request, trong ví dụ sẽ xây dựng 2 trang JSP là StartBoxBean.JSP, trang này sẽ chứa một đối tượng BoxBean và khởi tạo 2 thuộc tính chiều dài, chiều rộng sau đó truyền đối tượng BoxBean sang trang FinishBoxBean.JSP, trang FinishBoxBean sẽ khởi tạo thuộc tính chiều cao và in các tham số của BoxBean ra trình duyệt.

6.2. BEAN CÓ TÂM VỰC MỘT YÊU CẦU (tiếp theo)

Trang StartBoxBean.JSP:

```
<h1>Start BoxBean!</h1>

<jsp:useBean id=«BoxBean1» scope=«request»
class=«Jbean.BoxBean» />

<jsp:setProperty name=«BoxBean1» property=«length» value=«10» />

<%
BoxBean1.setWidth (5);

%>
```

Trang FinishBoxBean.JSP

```
<h1>Finish BoxBean!</h1>

<jsp:useBean id=«BoxBean1» scope=«page» class=«Jbean.BoxBean» />
<jsp:setProperty name=«BoxBean1» property=«height» value=«20» />
<br>Truy cap cac thong tin cua BoxBean
Chieu cao: <jsp:getProperty name=«BoxBean1» property=«height» />
Chieu dai: <jsp:getProperty name=«BoxBean1» property=«length» />
Chieu rong: <jsp:getProperty name=«BoxBean1» property=«width» />
```

6.3. BEAN CÓ TÂM VỰC MỘT SESSION

- Bean chỉ được truy cập trong các yêu cầu của cùng một session mà Bean này đã được tạo ra.
- Các tham khảo đến Bean này được lưu trong đối tượng session.
- Đối tượng bị ràng buộc là javax.servlet.jsp.PageContext.
- Thường được dùng để chia sẻ thông tin giữa các yêu cầu của cùng một user.
- Một session ứng với một đối tượng browser được mở và truy xuất một site → Khi mở một trình duyệt nữa tức là có một session khác.

6.4. BEAN CÓ TÂM VỰC CẢ ỨNG DỤNG

- Khi trang chứa Bean này được nạp thì Bean có thể được truy cập trong tất cả các trang của ứng dụng.
- Công cụ chia sẻ data trong toàn ứng dụng, công cụ chia sẻ data giữa nhiều user trong cùng Website.
- Các tham khảo đến Bean chỉ được giải phóng khi đối tượng ServletContext bị hủy. Nghĩa là Bean này chỉ mất đi khi JSP engine khởi động lại.
- Thuộc tính scope="application" trong khai báo useBean được dịch thành PageContext.APPLICATION_SCOPE.
- Đối tượng bị ràng buộc là javax.servlet.Servlet.

CÂU HỎI THẢO LUẬN



Một đối tượng Bean sinh ra sẽ có các phạm vi hoạt động nào?

7. JAVAMAIL API

Các ứng dụng J2EE sử dụng JavaMail API TM để gửi E-mail. JavaMail API có 2 phần:

- Interface mức ứng dụng – Application Level Interface được dùng bởi các application components để gửi thư;
- Interface mức dịch vụ của nhà cung cấp - Service Provider Interface.

7.1. MỘT SỐ KHÁI NIỆM VỀ THƯ ĐIỆN TỬ

- **Host:** Máy gửi thư đi.
- **To:** Danh sách địa chỉ các người nhận.
- **From:** Địa chỉ người gửi.
- **Subject:** Chủ đề, tiêu đề chính của thư.
- **Message:** Nội dung thư. Một message được tổ chức thành từng phần (Part), một thư có thể có nhiều phần (MultiPart).
- **Protocol gửi thư:** SMTP, Simple Mail Transfer Protocol, một TCP/IP protocol dành cho việc gửi các message từ một máy tính tới một máy khác trong Protocol này được dùng để định hướng một E-mail trên Internet.
- **Địa chỉ người gửi:** Một đối tượng thuộc lớp InternetAddress/Address.
- **Địa chỉ người nhận (Recipient):** Là nhóm địa chỉ InternetAddress/Address. Người nhận có thể thuộc loại (Recipient Type): "CC", "BCC", "TO".

7.1. MỘT SỐ KHÁI NIỆM VỀ THƯ ĐIỆN TỬ (tiếp theo)

- Sau khi gửi thư, thư được lưu trên server. Khi chúng ta truy cập mail server, chỉ có các thông tin cơ bản về thư (thuộc tính) được thể hiện ra màn hình (prefetch, lấy trước). Chỉ khi xem nội dung một thư cụ thể, nội dung message mới được tải về từ Server.
- Một thư thường được cất (lưu) trong một thư mục (folder) - người dùng có thể tự tạo thư mục này. Một user có thể có nhiều thư mục chứa thư. Một thư mục có một số thư và một lần người chủ hộp thư chỉ xem được một số thư (message number-msgnum), xem tiếp các thư khác phải kích Next.
- Một thư có thể được đánh dấu Read, Unread... các tính chất này được ghi nhận bằng một bộ các bit cờ (flag): ANSWERED (đã trả lời), DELETED (đã bị xóa), DRAFT (nháp), FLAGGED (đã đánh dấu), RECENT (mới nhận), SEEN (đã đọc - read).
- Mail session: Một phiên truy cập hộp thư từ server.

7.2. GÓI JAVAX.MAIL – CÁC LỚP MÔ HÌNH HÓA HỆ THỐNG THƯ

Đây là gói cung cấp cho người sử dụng hầu hết các xử lý cơ bản về E-mail.

Các lớp trong gói javax.mail:

MessageAware	Một interface được tùy ý hiện thực bởi một đối tượng DataSources để cung cấp thông tin cho một DataContentHandler về message context trong đó đối tượng nội dung đang hoạt động
MultipartDataSource	Interface cho DataSource có chứa các thân (body, nội dung) gồm nhiều phần
Part	Interface cơ bản cho Messages và BodyParts
UIDFolder	Interface sẽ được hiện thực bởi lớp Folders nhằm cung cấp tác vụ “disconnected” mode thông qua các unique-id cho mỗi message trong folder
Address	Lớp trùu tượng mô tả cho địa chỉ trong một message
Authenticator	Lớp biểu diễn cho việc xác nhận kết nối
BodyPart	Lớp mô tả cho một phần (Part) trong một Multipart
FetchProfile	Lớp giúp lấy trước các thuộc tính của một nhóm message từ server
FetchProfile.Item	Lớp inner này là lớp cơ bản cho các mục tin có thể được yêu cầu trong một FetchProfile
Flags	Biểu diễn cho một tập thông tin kiểm tra (cờ) được đánh dấu trên Message
Flags.Flag	Biểu diễn cho một System flag

7.2. GÓI JAVAX.MAIL – CÁC LỚP MÔ HÌNH HÓA HỆ THỐNG THƯ (tiếp theo)

Folder	Lớp trùu tượng mô tả cho một thư mục chứa thư
Header	Lớp mô tả cho một cặp <name/value> biểu diễn các headers
Message	Lớp mô tả cho một email message
Message.RecipientType	Lớp inner mô tả cho loại người nhận
MessageContext	Lớp mô tả cho ngữ cảnh chứa một đoạn nội dung (piece)
Multipart	Container cho nội dung nhiều đoạn
PasswordAuthentication	Lớp quản lý mật khẩu, được dùng bởi lớp Authenticator
Provider	Lớp mô tả cho loại protocol được dùng
ProviderType	Lớp inner định nghĩa loại Provider
Service	Lớp trùu tượng chứa các chức năng cơ bản phục vụ việc quản lý thư như lưu thư (store), truyền thư (transport)
Session	Biểu diễn cho một mail session, lớp final
Store	Lớp trùu tượng giúp mô hình hóa việc lưu trữ, truy xuất thư cùng với protocol đã dùng
Transport	Lớp trùu tượng mô hình hóa việc gửi thư
UIDFolder.FetchProfileItem	Một mục được lấy trước
URLName	Lớp mô tả cho tên một URL

7.2. GÓI JAVAX.MAIL – CÁC LỚP MÔ HÌNH HÓA HỆ THỐNG THƯ (tiếp theo)

Lớp javax.mail.Message

- Đây là lớp trừu tượng mô tả cho một thư.
- Các lớp con của lớp này sẽ hiện thực cụ thể cho từng loại thư.
- Lớp MimeMessage là lớp con của lớp Message cụ thể hóa các thư có kiểu MIME (Multipurpose Internet Mail Extensions, một protocol mở rộng của SMTP, cho phép truyền kết hợp text/video/sound...).

7.2. GÓI JAVAX.MAIL – CÁC LỚP MÔ HÌNH HÓA HỆ THỐNG THƯ (tiếp theo)

Dữ liệu của lớp Message

protected boolean	expunged	True nếu thư bị xóa (expunge).
protected Folder	folder	Thư mục chứa thư
protected int	msgnum	Số thư trong thư mục chứa
protected Session	session	Đối tượng Session cho thư này
Lớp inner	static class MessageType	– loại người nhận (BCC, CC, TO, một chuỗi)

Constructors của lớp Message

protected	Message()	No-arg version of the constructor.
protected	Message(Folder folder, int msgnum)	Constructor that takes a Folder and a message number.
protected	Message(Session session)	Constructor that takes a Session.

7.2. GÓI JAVAX.MAIL – CÁC LỚP MÔ HÌNH HÓA HỆ THỐNG THƯ (tiếp theo)

Các method của lớp Message

abstract void abstract Address[] abstract void	addFrom(Address[] addresses) – Thêm các địa chỉ cho "From" getFrom() – Lấy "From" attribute. setFrom(Address address) attribute
void abstract void abstract Address[]	addRecipient(Message.RecipientType type, Address address) addRecipients(Message.RecipientType type, Address[] addresses) getRecipients(Message.RecipientType type)
Address[]	getAllRecipients() – Lấy tập địa chỉ người nhận.
void	setRecipient(Message.RecipientType type, Address address)
abstract void	setRecipients(Message.RecipientType type, Address[] addresses)
abstract Flags	getFlags() – Lấy tập cờ đánh dấu thư, các cờ: Flags.Flag, Flags.Flag.ANSWERED, Flags.Flag.DELETED, Flags.Flag.DRAFT, Flags.Flag.FLAGGED, Flags.Flag.RECENT, Flags.Flag.SEEN

7.2. GÓI JAVAX.MAIL – CÁC LỚP MÔ HÌNH HÓA HỆ THỐNG THƯ (tiếp theo)

Folder	getFolder() – Lấy thư mục chứa thư
protected int void	getMessageNumber() - Lấy số thứ tự có trong thư mục chứa thư này setMessageNumber(int msgnum)
abstract Date	getReceivedDate() – Lấy thông tin ngày nhận thư
Address[]	getReplyTo() – Lấy tập địa chỉ cần trả lời
abstract Date abstract void	getSentDate() – Lấy ngày gửi setSentDate(java.util.Date date)
abstract String abstract void	getSubject() – Lấy chủ đề của thư setSubject(java.lang.String subject)
boolean	isExpunged() – Kiểm tra thư này đã bị xóa chưa?
boolean	isSet(Flags.Flag flag) – Kiểm tra 1 bit cờ
boolean	match(SearchTerm term) – Áp dụng 1 tiêu chuẩn tìm kiếm trong thư
abstract Message	reply(boolean replyToAll) – Lấy thư trả lời của thư này

7.2. GÓI JAVAX.MAIL – CÁC LỚP MÔ HÌNH HÓA HỆ THỐNG THƯ (tiếp theo)

abstract void	saveChanges () – Lưu các thay đổi của thư vào thư mục chứa thư khi thư mục này đóng
protected void	setExpunged(boolean expunged) Thiết lập cờ xóa
void	setFlag(Flags.Flag flag, boolean set) – Thiết lập 1 cờ
abstract void	setFlags(Flags flag, boolean set)
void	setReplyTo(Address[] addresses)

TÓM LƯỢC CUỐI BÀI

Sau khi học xong bài học học viên cần:

- Hiểu được khái niệm về JavaBean và cấu trúc của JavaBean.
- Biết cách sử dụng JavaBean trong các trang JSP.
- Hiểu được các tầm vực của Bean.
- Vận dụng được các kiến thức đã học để xây dựng các JavaBean và sử dụng các JavaBean trong trang JSP.