



**FPT POLYTECHNIC**



[www.poly.edu.vn](http://www.poly.edu.vn)



Conceive Design Implement Operate

**BEANUTIL, EMAIL, SESSIONS VÀ COOKIES**

**LẬP TRÌNH JAVA #3 (P5.1)**

- ☐ Sử dụng BeanUtils để đọc tham số
- ☐ Đọc và chuyển đổi tham số theo định dạng thời gian
- ☐ Xây dựng trang web gửi email



- ❑ **BeanUtils** là API do Apache phát triển hỗ trợ cho việc xử lý bean nhanh chóng và thuận lợi
- ❑ Hai phương thức tiện ích được sử dụng nhiều nhất
  - ❖ `BeanUtils.populate(bean, Map<String, Object[]>)`
    - Copy value từ Map vào các thuộc tính cùng tên của bean
  - ❖ `BeanUtils.copyProperties(destBean, srcBean)`
    - Copy value các thuộc tính của srcBean vào các thuộc tính cùng tên destBean

```
<dependency>
  <groupId>commons-beanutils</groupId>
  <artifactId>commons-beanutils</artifactId>
  <version>1.9.3</version>
</dependency>
```

```
// Dữ liệu của Map
Map<String, String[]> map = new HashMap<>();
map.put("fullname", new String[] {"Nguyễn Văn Tèo"});
map.put("salary", new String[] {"1500"});
map.put("hobbies", new String[] {"Music", "Travelling"});
// Bean sẽ chứa dữ liệu của map
Staff bean = new Staff();
BeanUtils.populate(bean, map);
```

```
public class Staff{
    String fullname;
    Double salary;
    String[] hobbies;
    getters/setters
}
```

❑ Sử dụng phương thức populate để đọc giá trị các tham số vào các thuộc tính cùng tên của bean

❖ Map<String, Object[]> map = **request.getParameterMap();**

❖ **BeanUtils.populate**(bean, map)

- ❑ Một lớp được gọi là Java Bean class khi thỏa mãn các quy ước sau:
  - ❖ Phải định nghĩa là public
  - ❖ Phải có constructor mặc định không tham số
  - ❖ Đọc ghi dữ liệu bên trong bằng các phương thức getter và setter
- ❑ Chiếu theo quy ước trên thì lớp Staff thỏa mãn là lớp Java Bean
  - ❖ Có định nghĩa public
  - ❖ Không định nghĩa constructor nào có nghĩa là có constructor mặc định không tham số
  - ❖ Các field của nó được đọc ghi thông qua các getter và setter
- ❑ Chú ý: Tên thuộc tính của bean có nghĩa là phần sau của get và set
  - ❖ Nếu chỉ có getter thì gọi là thuộc tính chỉ đọc (readonly)
  - ❖ Nếu chỉ có setter thì gọi là thuộc tính chỉ ghi (writeonly)

Fullname?

Age?

☐ Male ☐ Female

Recruitment Date?

Việt nam ▼

Chọn tệp Không có tệp nào được chọn

☐ Reading ☐ Traveling ☐ Music ☐ Other

Salary

Submit

StaffServlet

doPost()

BeanUtils

Staff

fullname: String  
age: **int**  
gender: boolean  
recruit: **Date**  
country: String  
photo: String  
hobbies: **int[]**  
salary: double

getters/setters

- ❑ Vấn đề với `getParameter()` là phải kiểm tra null và chuyển đổi sang kiểu phù hợp để xử lý. Vì vậy
  - ❖ Viết mã dài dòng, phức tạp
  - ❖ Dễ mắc lỗi
- ❑ Sử dụng BeanUtils
  - ❖ Tự chuyển đổi kiểu phù hợp
  - ❖ Code ngắn, đơn giản, rõ ràng
  - ❖ Tổ chức bài bản, dễ nâng cấp

# VÍ DỤ 1 VỀ SỬ DỤNG BEANUTILS

```
<%@ taglib uri="http://java.sun.com/jstl/core_rt" prefix="c" %>
<c:url var="url" value="/bean/simple/submit.php"/>
<form action="${url}" method="post">
  <input name="fullname" placeholder="Fullname?"> <br>
  <input name="age" placeholder="Age?"> <br>
  <input name="gender" type="radio" value="true"> Male
  <input name="gender" type="radio" value="false"> Female <br>
  <select name="country">
    <option value="VN">Việt nam</option>
    <option value="US">United States</option>
  </select> <br>
  <input name="salary" placeholder="Salary"> <hr>
  <button>Submit</button>
</form>
```

```
public class SimpleBean {
  String fullname;
  Integer age;
  Double salary;
  Boolean gender;
  String country;
  getters/setters
}
```

Nguyễn Nghiệm
Age?
<input checked="" type="radio"/> Male <input type="radio"/> Female
United States ▾
1000
Submit



# VÍ DỤ 1 VỀ SỬ DỤNG BEANUTILS

```
@WebServlet({"bean/simple/form.php", "bean/simple/submit.php"})
public class SimpleBeanServlet extends HttpServlet{
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {
        req.getRequestDispatcher("/views/bean/simple.jsp").forward(req, resp);
    }
    @Override
    protected void doPost(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {
        try {
            SimpleBean bean = new SimpleBean();
            BeanUtils.populate(bean, req.getParameterMap());
        } catch (Exception e) {
            throw new ServletException(e);
        }
        this.doGet(req, resp);
    }
}
```

```
System.out.println(">>Fullname: " + bean.getFullname());
System.out.println(">>Age: " + bean.getAge());
System.out.println(">>Salary: " + bean.getSalary());
System.out.println(">>Gender: " + bean.getGender());
System.out.println(">>Country: " + bean.getCountry());
```



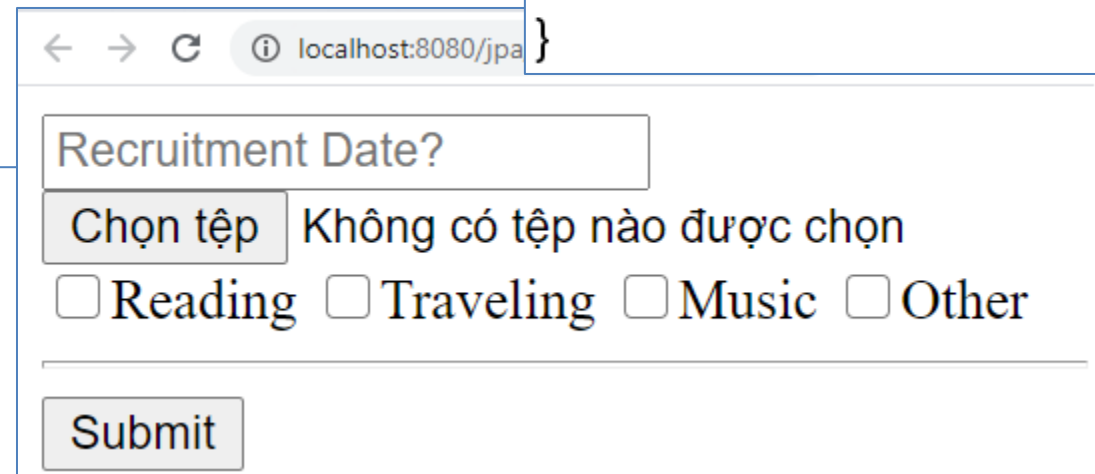
## VÍ DỤ 2 VỀ SỬ DỤNG BEANUTILS

```
<%@ taglib uri="http://java.sun.com/jstl/core_rt" prefix="c" %>
<c:url var="url" value="/bean/advance/submit.php"/>
<form action="${url}" method="post" enctype="multipart/form-data">
  <input name="recruitDate" placeholder="Recruitment Date?"> <br>
  <input name="photo_file" type="file"> <br>
  <input name="hobby" type="checkbox" value="0">Reading
  <input name="hobby" type="checkbox" value="1">Traveling
  <input name="hobby" type="checkbox" value="2">Music
  <input name="hobby" type="checkbox" value="3">Other
  <hr>
  <button>Submit</button>
</form>
```

```
public class AdvanceBean {
    Integer[] hobby;
    Date recruitDate;
    String photo;
    getters/setters
}
```

### ❑ Chú ý

- ❖ **Integer[]** chứa các hobby được chọn
- ❖ **java.util.Date** chứa thời gian
- ❖ String chứa tên của file upload (**photo\_file**)



The screenshot shows a web browser window with the address bar displaying 'localhost:8080/jpa'. The form contains a text input field labeled 'Recruitment Date?', a file upload button labeled 'Chọn tệp' followed by the text 'Không có tệp nào được chọn', four radio buttons labeled 'Reading', 'Traveling', 'Music', and 'Other', and a 'Submit' button at the bottom.

## VÍ DỤ 2 VỀ SỬ DỤNG BEANUTILS

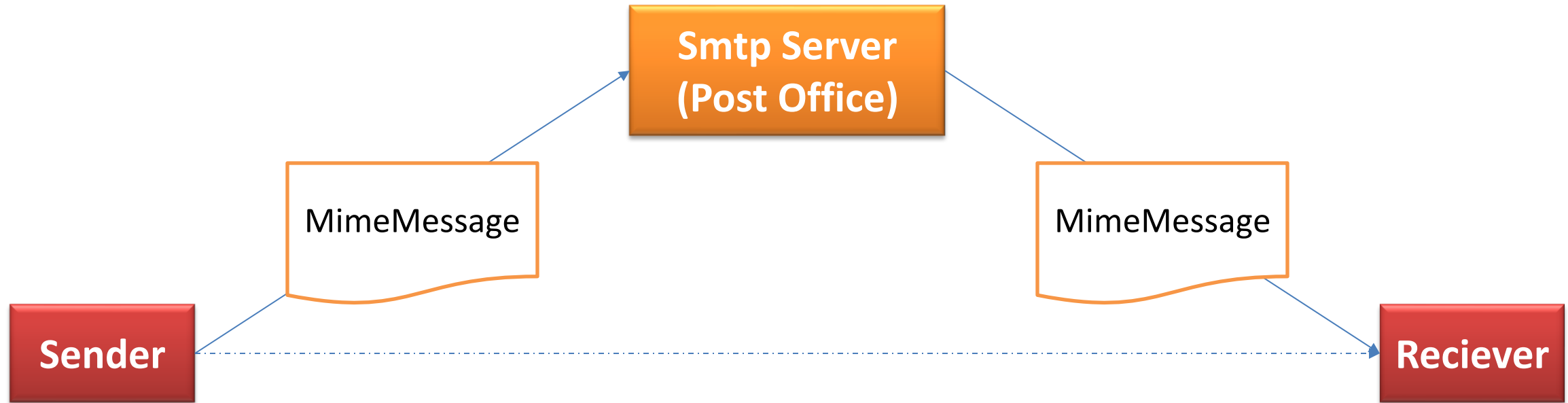
```
@MultipartConfig
@WebServlet({"/bean/advance/form.php", "/bean/advance/submit.php"})
public class AdvanceBeanServlet extends HttpServlet{
    @Override
    protected void doPost(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {
        try {
            DateTimeConverter dtc = new DateConverter(new Date());
            dtc.setPattern("MM/dd/yyyy");
            ConvertUtils.register(dtc, Date.class);
            AdvanceBean bean = new AdvanceBean();
            BeanUtils.populate(bean, req.getParameterMap());

            Part part = req.getPart("photo_file");
            bean.setPhoto(part.getSubmittedFileName());
        } catch (Exception e) {
            throw new ServletException(e);
        }
        this.doGet(req, resp);
    }
}
```

*Thiết lập định dạng thời gian để BeanUtils căn cứ trong việc chuyển đổi kiểu thời gian*

*Xử lý upload file*

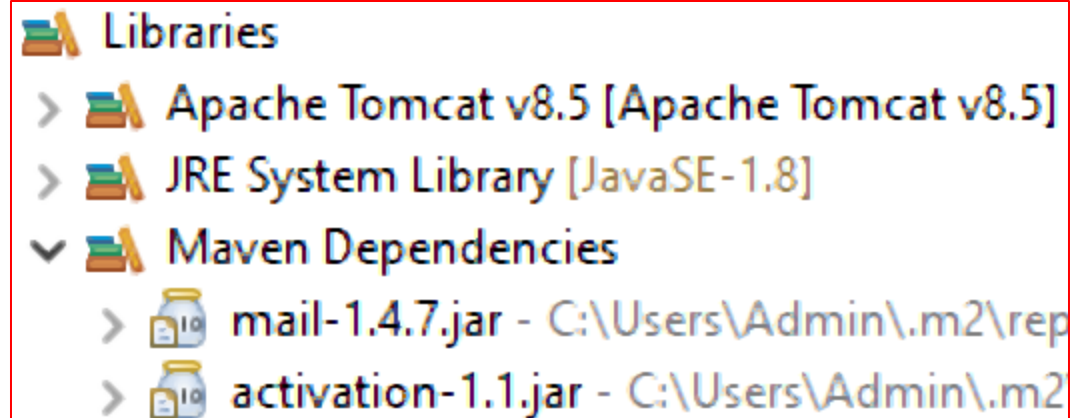
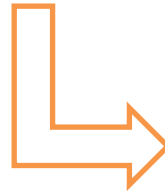




### ❑ Thông tin đầy đủ của một email gồm

- ❖ From: Người gửi
- ❖ To: Người nhận
- ❖ Subject: Tiêu đề
- ❖ Body: Nội dung
- ❖ CC: Những người đồng nhận
- ❖ BCC: Những người đồng nhận ẩn danh
- ❖ Attachment Files: Những file đính kèm

```
<dependency>  
  <groupId>javax.mail</groupId>  
  <artifactId>mail</artifactId>  
  <version>1.4.7</version>  
</dependency>
```



## KẾT NỐI SMTP SERVER (GMAIL SERVER)

// Thông số kết nối Smtplib Server

```
Properties props = new Properties();  
props.setProperty("mail.smtp.auth", "true");  
props.setProperty("mail.smtp.starttls.enable", "true");  
props.setProperty("mail.smtp.host", "smtp.gmail.com");  
props.setProperty("mail.smtp.port", "587");
```

// Kết nối Smtplib Server

```
Session session = Session.getInstance(props, new Authenticator() {  
    protected PasswordAuthentication getPasswordAuthentication() {  
        String username = "*****@gmail.com";  
        String password = "*****";  
        return new PasswordAuthentication(username, password);  
    }  
});
```

Tài khoản gmail này phải được kích hoạt cho phép gửi email từ ứng dụng (xem slide sau)



// Tạo message

```
MimeMessage message = new MimeMessage(session);  
message.setFrom(new InternetAddress("from@gmail.com"));  
message.setRecipients(Message.RecipientType.TO, "to@gmail.com");  
message.setSubject("Tiêu đề email", "utf-8");  
message.setText("Nội dung email", "utf-8", "html");  
message.setReplyTo(message.getFrom());
```

// Gửi message

```
Transport.send(message);
```

- ❑ Viết bổ sung đoạn mã sau đây trước khi gọi Transport.send() nếu muốn thực hiện đính kèm file

```
File file = new File("...path...");  
MimeBodyPart part = new MimeBodyPart();  
part.setDataHandler(new DataHandler(new FileDataSource(file)));  
part.setFileName(file.getName());
```

```
Multipart multipart = new MimeMultipart();  
multipart.addBodyPart(part);  
message.setContent(multipart);
```

- ❑ CC, BCC

- ❖ setRecipients(Message.RecipientType.CC, "email1, email2,...")
- ❖ setRecipients(Message.RecipientType.BCC, "email1, email2,...")



- ✓ Sử dụng BeanUtils để đọc tham số
- ✓ Đọc và chuyển đổi tham số theo định dạng thời gian
- ✓ Xây dựng trang web gửi email





**Cảm ơn**