CSE 510
Web Data Engineering

Java Beans

cse@buffalo

What is a Java Bean?

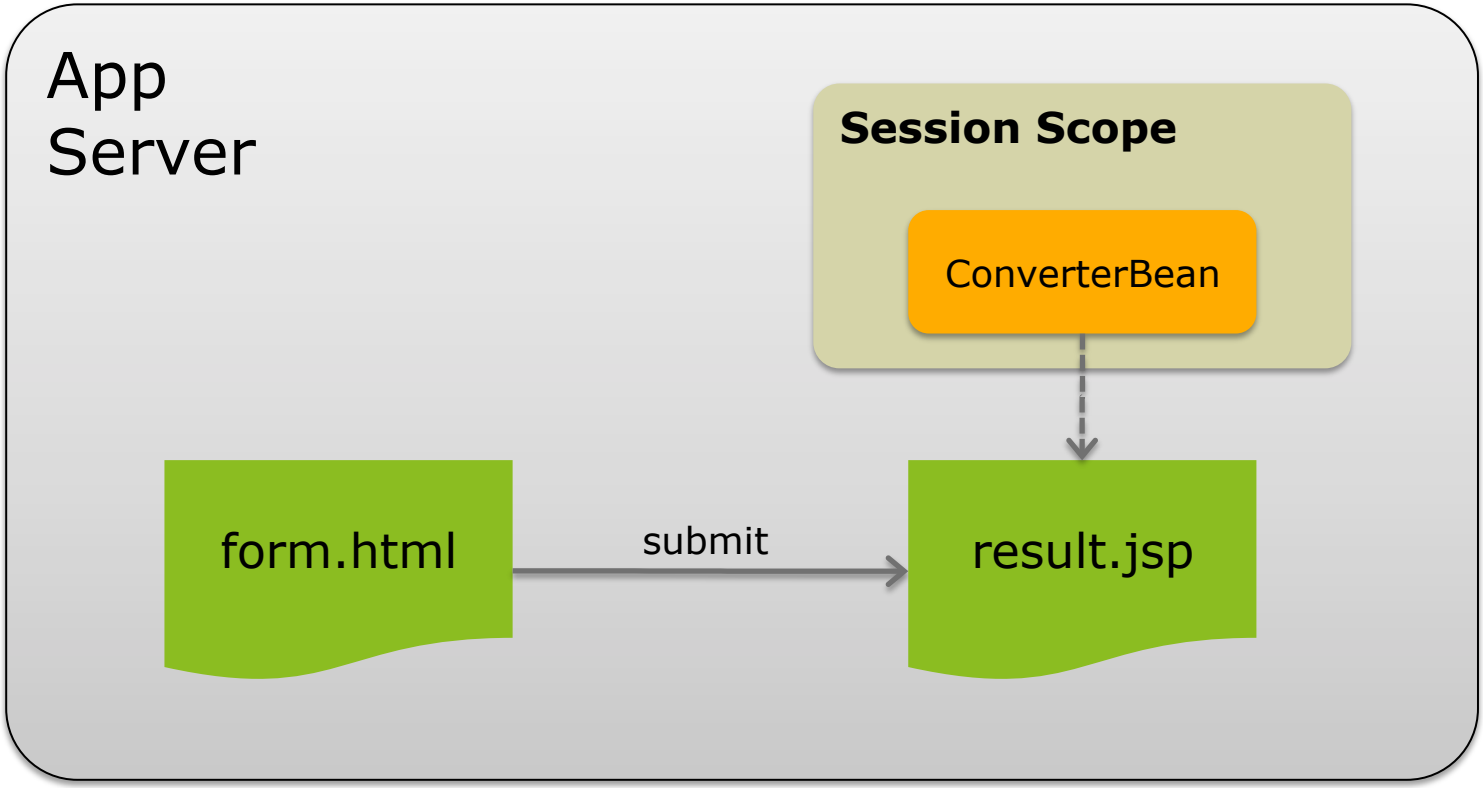
- Simply a class wrapping some data…

  …and meets certain restrictions
- Three components:
  - **Default constructor** (no arguments)
  - **Private properties** (instance variables) only, no public ones!
  - For each field, provide a **getter** method to retrieve it and a **setter** method to modify it

Why Java Beans?

- **Reusable and Portable Component Model**
  - No need to repeat the same code/queries within multiple JSPs
  - Code maintenance becomes much easier
  - Better use of resources
- Managed by container (Apache Tomcat)
  - Bean scope can be page, request, session or application
- Easy to assign HTTP request parameters to bean properties
- Easy to share beans among multiple HTTP requests, servlets and JSPs
- Can be introspected, persisted and customized

Currency Converter Example



form.html

```
<html>
<head>
   <title>Currency Conversion Form</title>
</head>
<body>
   <h1>Currency Conversion Form</h1>

   <p>Enter an amount to convert:</p>
   <form action="result.jsp" method="GET">
        <input type="text" name="usdAmount" size="25"><br />
        <input type="submit" value="Submit">
   </form>
</body>
</html>
```

result.jsp (Without Java Bean)

```
<html>
<head><title>Currency Conversion Result</title></head>
<body>
   <h1>Currency Conversion Result</h1>
   <%
        String usdAmount = request.getParameter("usdAmount");
        BigDecimal yenRate = new BigDecimal("88.75");
        BigDecimal yenAmount =
               new BigDecimal(usdAmount).multiply(yenRate);
        yenAmount = yenAmount.setScale(2, BigDecimal.ROUND_UP);
   %>
   <p><%= usdAmount %> USD are <%= yenAmount %> Yen</p>
</body>
</html>
```

Converter Bean

```
package converter;
public class ConverterBean {
   // private properties
   private String usdAmount = "0.0";
   private String yenRate = "88.75";
   private BigDecimal yenAmount;
   // Getters and setters automatically generated by Eclipse
   // Menu Source -> Generate Getters and Setters...
   public String getUsdAmount() { return usdAmount; }
   public void setUsdAmount(String usdAmountStr) {
        usdAmount = usdAmountStr;
        ...
   }
   ...
}
```

Naming Convention

- Connect property names with getter/setter method names
- Capitalize the first letter of the property and add the word “get” (“set” respectively)
  - Property: **`usdAmount`**
  - Getter: **`getUsdAmount`**
  - Setter: **`setUsdAmount`**

How to Use a Java Bean in a JSP

- Compile bean and place under WEB-INF/classes

```
<jsp:useBean id="<beanName>" class="<beanClass>"
             scope="<scope>"/>
```

- Equivalent to

```
<% <beanClass> <beanName> = new <beanClass>(); %>
```

Always?

```
<jsp:getProperty name="<beanName>"
                 property="<propertyName>"/>
```

- Equivalent to `<%= <beanName>.getPropertyName() %>`

```
<jsp:setProperty name="<beanName>"
                 property="<propertyName>"
                 value="<value>"/>
```

- Can be initialized when subelement of `jsp:useBean`

result.jsp (With Java Bean)

```
<%@ page import="converter.*"%>
<jsp:useBean id="conv" scope="session"
             class="converter.ConverterBean"/>
<jsp:setProperty name="conv" property="*"/>
<html>
<head><title>Currency Conversion Result</title></head>
<body>
   <h1>Currency Conversion Result</h1>
   <p>
   <jsp:getProperty name="conv" property="usdAmount"/> USD
       are
   <jsp:getProperty name="conv" property="yenAmount"/> Yen
   </p>
</body>
</html>
```

resultInitRate.jsp

- Initialize bean properties

Why not session?

```
<jsp:useBean id="conv" scope="request"
             class="converter.ConverterBean">
  <jsp:setProperty name="conv" property="yenRate"
                               value="100" />
</jsp:useBean>
<jsp:setProperty name="conv" property="*"/>
<html>
<head><title>Currency Conversion Result</title></head>
<body>
   ...
</body>
</html>
```