

Expression Language (EL) and JSP standard Tag Library (JSTL)

- **Using scriptlet is discouraged:** We want to generate dynamic content without embedding Java code in html
- **EL** was introduced in JSP 2.0 as an alternative to the scripting elements (scriptlets), to make writing JSP easier by non-programmers and JSP pages more readable.
- Simplify the presentation by replacing hard-to-maintain scriptlets with short entries of form `${expression}`
- The goal: use scriptlets as less as possible.
- For example, the following JSP with scriptlets is hard to write, and equally hard to read:

```
<% String username = request.getParameter("username");
  if (username != null) { %>
    <p>Hello, <%= username %>!</p>
  <% } else { %>
    <p>Hello, everyone!</p>
  <% } %>
```

40



40

EL language

`${expr}` `# { . . . }`

- The main construct of the EL is the expression evaluator `$(ELExpression)`, which evaluates the `ELExpression` within the braces.
- For example, `${1+2*3}`, `${index == 0}`, `${user.firstName}`, `#{aBean.aProperty}`, `#{aMap["aKey"]}`.
- Operators
 - Arithmetic Operators: Addition `+`, subtraction `-`, multiplication `*`, division `/` (or **div**), and modulus `'%'` (or **mod**).
 - Comparison Operators: `==` (or **eq**), `!=` (or **ne**), `>` (or **gt**), `<` (or **lt**), `>=` (or **ge**), `<=` (or **le**). The equality operator (`==` or **eq**), when applied to string, compare the contents of two strings (similar to Java's `aString.equals(anotherString)`).
 - Boolean (Logical) Operators: `&&` (or **and**), `||` (or **or**), `!` (or **not**).
 - Validation Operator `empty`: `empty` is a boolean unary operator checking for null value or empty string, e.g., `#{empty param["user"]}` returns true if the request parameter "user" is null or an empty string. You could also use `#{not empty}`.
 - Shorthand if-else `? : :` Similar to Java/C/C++, (test ? trueExpression : falseExpression) returns the value of trueExpression if test resulted in true; or falseExpression otherwise.

41



41

Basic Operators in EL:

JSP Expression Language (EL) supports most of the arithmetic and logical operators supported by Java. Below is the list of most frequently used operators:

Operator	Description
.	Access a bean property or Map entry
[]	Access an array or List element
()	Group a subexpression to change the evaluation order
+	Addition
-	Subtraction or negation of a value
*	Multiplication
/ or div	Division
% or mod	Modulo (remainder)
== or eq	Test for equality
!= or ne	Test for inequality
< or lt	Test for less than
> or gt	Test for greater than
<= or le	Test for less than or equal
>= or ge	Test for greater than or equal
&& or and	Test for logical AND
or or	Test for logical OR
! or not	Unary Boolean complement
empty	Test for null, empty String, array or Collection.
func(arg)	A function call



42

EL Expression	Result
<code>\${1}</code>	1
<code>\${1 + 2}</code>	3
<code>\${1.2 + 2.3}</code>	3.5
<code>\${1.2E4 + 1.4}</code>	12001.4
<code>\${-4 - 2}</code>	-6
<code>\${21 * 2}</code>	42
<code>\${3/4}</code>	0.75
<code>\${3 div 4}</code>	0.75
<code>\${3/0}</code>	Infinity
<code>\${10%4}</code>	2
<code>\${10 mod 4}</code>	2
<code>\${(1==2) ? 3 : 4}</code>	4

From Tomcat JSP examples

- Less-than (< or lt)
- Greater-than (> or gt)
- Less-than-or-equal (<= or le)
- Greater-than-or-equal (>= or ge)
- Equal (== or eq)
- Not Equal (!= or ne)

Numeric

EL Expression	Result
<code>\${1 < 2}</code>	true
<code>\${1 lt 2}</code>	true
<code>\${1 > (4/2)}</code>	false
<code>\${1 gt (4/2)}</code>	false
<code>\${4.0 >= 3}</code>	true
<code>\${4.0 ge 3}</code>	true
<code>\${4 <= 3}</code>	false
<code>\${4 le 3}</code>	false
<code>\${100.0 == 100}</code>	true
<code>\${100.0 eq 100}</code>	true
<code>\${(10*10) != 100}</code>	false
<code>\${(10*10) ne 100}</code>	false

Alphabetic

EL Expression	Result
<code>\${'a' < 'b'}</code>	true
<code>\${'hip' > 'hit'}</code>	false
<code>\${'4' > 3}</code>	true



43

43

- EL Implicit objects

Implicit object & Description	
pageScope Scoped variables from page scope	param Request parameters as strings A key-value map of request parameters to their <i>first</i> value. E.g., <code>\${param["username"]}</code> or <code>\${param.username}</code>
requestScope Scoped variables from request scope	paramValues Request parameters as collections of strings
sessionScope Scoped variables from session scope	header HTTP request headers as strings
applicationScope Scoped variables from application scope	headerValues HTTP request headers as collections of strings
	initParam Context-initialization parameters A key-value map of the application's initialization parameter declared in "web.xml". E.g., <code>\${initParam["username"]}</code> or <code>\${initParam.username}</code>
	cookie Cookie values
	pageContext The JSP PageContext object for the current page

All except pageContext, are type *map*

44

44

EL get attribute value by name?

- In MVC, a servlet invokes code that creates the data, then uses **RequestDispatcher.forward** or **response.sendRedirect** to transfer control to the appropriate JSP page.
- To permit the JSP page to access the data, the servlet needs to use **setAttribute** to store the data in one of the standard locations: the **HttpServletRequest**, the **HttpSession**, or the **ServletContext**.
- Objects in these locations are known as "scoped variables," and the expression language has a quick and easy way to access them.
- `\${name}`**
- Search in the order of
 - page scope
 - request scope,
 - session scope
 - application (context) scope
- Can specify **`\${requestScope.name}`** or **`\${requestScope["name"]}`**
`\${sessionScope.name}` or **`\${sessionScope["name"]}`**
`\${applicationScope.name}` or **`\${applicationScope["name"]}`**

45

45

JSP old	EL
parameter <code>request.getParameter(" ")</code>	<code>\${param.name}</code> <code>\${param['name']}</code> <code>"name"</code>
initParamet <code>application.getInitParam("config")</code>	<code>\${initParam.name}</code> <code>initParam["name"]</code> <code>"name"</code>
attributes <code>request.getAttribute()</code> <code>session.getAttribute()</code> <code>application.getAttribute()</code>	<code>`\${name}`</code> <code>`\${requestScope.name}`</code> <code>`\${sessionScope.name}`</code> <code>`\${applicationScope.name}`</code> <code>["name"]</code> <code>["name"]</code>

46

YORK
UNIVERSITY

46

JSTL -- JSP standard tag Library

- EL does not contain programming constructs such as if-else, loops, and setting variables. They are provided by the JSP standard tag Library (JSTL) core tags, such as **<c:set>**, **<c:out>**, **<c:if>**, and **<c:forEach>**. For examples:

For examples,

```
<c:set var="name" value="Peter"/>
<c:set var="name" scope="session" value="${param['user']}" />
<c:set var="name" scope="session" value="${param.user}" />
<c:set var="area" value="${param['radius']*param['radius']*3.1416}" />
```

For example,

```
// Prints a literal value
<c:out value="Hello, world" />
// Prints the result of an EL Expression
<c:out value="${user.firstname}" />
// Can concatenate literals and EL expressions
<c:out value="Hello, ${user.firstname} ${user.lastname}!" />
// Default for null value
<c:out default="everybody" value="${param['name']}" />
```

47

YORK
UNIVERSITY

47

Conditional (or Selection): <c:if>, <c:choose>, <c:when>, <c:otherwise>

```
// Perform the body only if the expression returns true
<c:if test="expression" > ... </c:if>

// Assign the result of test to the variable
<c:if test="expression" var="varName" scope="scope" > ... </c:if>

// Similar to switch-case statement in Java/C/C++
<c:choose>
  <c:when test="ELExpression1" > ... </c:when>
  <c:when test="ELExpression2" > ... </c:when>
  .....
  <c:otherwise > ... </c:otherwise>
</c:choose>
```

Loops: <c:forEach>, <c:forEachToken>

```
// Similar to for-each loop in JDK 1.5, to iterate thru a collection or array
<c:forEach var="varName" items="ELExpressionMapArray" >
  .....
</c:forEach>

// Similar to for-loop in Java
<c:forEach var="varName" begin="intExpression" end="intExpression" step="1|intExpression">
  .....
</c:forEach>
```

- To use the JSTL core library, we need to
 - include this JSP taglib directive in the JSP page, similar to using any custom tag library.

```
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
```

- 48
- make sure the tag library JAR-files e.g., "standard.jar" and "jstl.jar" shall be accessible by the web application

48

- Now see some JSP examples that use the EL and JSTL.

49

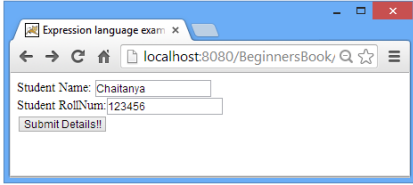
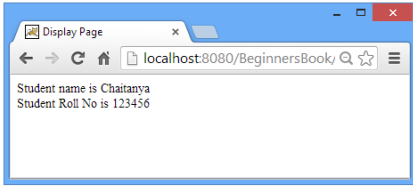
49

Index.html

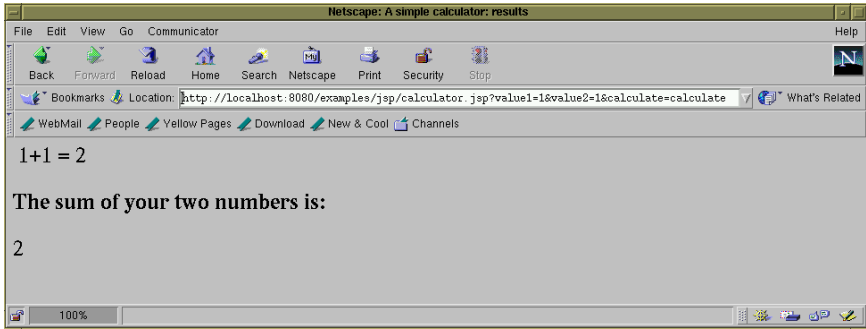
```
<html>
<head>
  <title>Expression language example2</title>
</head>
<body>
  <form action="display.jsp">
    Student Name: <input type="text" name="stuname" /><br>
    Student RollNum:<input type="text" name="rollno" /><br>
    <input type="submit" value="Submit Details!!"/>
  </form>
</body>
</html>
```

display.jsp

```
<html>
<head>
  <title>Display Page</title>
</head>
<body>
  Student name is ${ param.stuname } <br>
  Student Roll No is ${ param.rollno }
</body>
</html>
```





50

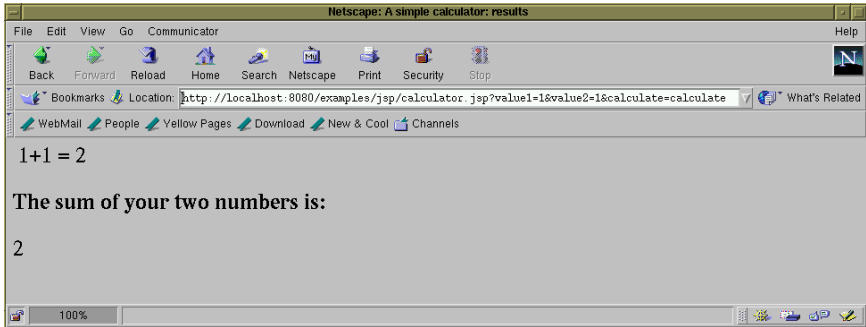


```
<html>
<head><title>A simple calculator: results</title></head>
<body>
  <!-- A simpler example 1+1=2 -->
  1+1 = <%= 1+1 %> // or <% out.print(1+1) %>
  <!-- A simple calculator -->
  <h2>The sum of your two numbers is:</h2>
  <%= Integer.parseInt(request.getParameter("value1")) +
    Integer.parseInt(request.getParameter("value2"))
  %>
</body>
</html>
```

calculator.jsp



51




```

<html>
<head><title>A simple calculator: results</title></head>
<body>
<!-- A simpler example 1+1=2 -->
1+1 = ${1+1}
<!-- A simple calculator -->
<h2>The sum of your two numbers is:</h2>
    ${ param["value1"] + param.value2 }
    <!-- auto convert string to int -->
</body>
</html>

```

52 calculator.jsp



Get attributes (scoped variables)

```

public class ScopedVars extends HttpServlet {
    public void doGet(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException {
        request.setAttribute("attribute1", "First Value");

        HttpSession session = request.getSession();
        session.setAttribute("attribute2", "Second Value");

        ServletContext application = getServletContext();
        application.setAttribute("attribute3", new java.util.Date());

        request.setAttribute("repeated", "by Request");
        session.setAttribute("repeated", "by Session");
        application.setAttribute("repeated", "by ServletContext");

        RequestDispatcher dispatcher =
            request.getRequestDispatcher("scoped-vars.jsp");
        dispatcher.forward(request, response);
    }
}

```

53



scoped-vars.jsp

```
<TABLE BORDER=5 ALIGN="CENTER">
  <TR><TH CLASS="TITLE">
    Accessing Scoped Variables
  </TH>
</TABLE>
<P>
  Traditional:
  <UL>
    <LI><B>attribute1: </B> <%= request.getAttribute("attribute1") %>
    <LI><B>attribute2: </B> <%= session.getAttribute("attribute2") %>
    <LI><B>attribute3: </B> <%= application.getAttribute("attribute3") %>
  </UL>

  EL:
  <UL>
    <LI><B>attribute1:</B> ${attribute1}
    <LI><B>attribute2:</B> ${attribute2}
    <LI><B>attribute3:</B> ${attribute3}
    <LI><B>Source of "repeated" attribute:</B> ${repeated}
  </UL>
</UL>
```



54

← → ↻ ⚠ Not secure | 192.168.0.12:8080/try/SP-CoreServletBook/ScopedVars

New Tab Data Structures & A... Leisure reading C Unix Programming (JAVA) Sth Else (formally ot...

Accessing Scoped Variables

Traditional:

- attribute1: First Value
- attribute2: Second Value
- attribute3: Fri Jun 23 12:13:25 EDT 2023

EL:

- attribute1: First Value
- attribute2: Second Value
- attribute3: Fri Jun 23 12:13:25 EDT 2023
- Source of "repeated" attribute: by Request

```
<%= request.getAttribute("attribute1") %>
<%= session.getAttribute("attribute2") %>
<%= application.getAttribute("attribute3") %>

${attribute1}
${attribute2}
${attribute3}
${repeated}
```

55



55

scoped-vars.jsp

```
<TABLE BORDER=5 ALIGN="CENTER">
  <TR><TH CLASS="TITLE">
    Accessing Scoped Variables
  </TH>
</TABLE>
<P>
Traditional:
<UL>
  <LI><B>attribute1: </B> <%= request.getAttribute("attribute1") %>
  <LI><B>attribute2: </B> <%= session.getAttribute("attribute2") %>
  <LI><B>attribute3: </B> <%= application.getAttribute("attribute3") %>
</UL>
EL:
<UL>
  <LI><B>attribute1:</B> ${attribute1}
  <LI><B>attribute2:</B> ${attribute2}
  <LI><B>attribute3:</B> ${attribute3}
  <LI><B>Source of "repeated" attribute:</B> ${repeated}
    <ul>
      <LI><B>Source of "repeated reqScope" attribute:</B> ${requestScope.repeated}
      <LI><B>Source of "repeated sessScope" attribute:</B> ${sessionScope.repeated}
      <LI><B>Source of "repeated appScope" attribute:</B> ${applicationScope.repeated}
    </ul>
  </LI>
</UL>
```



56

← → ↻ ⚠ Not secure | 192.168.0.12:8080/try/SP-CoreServletBook/ScopedVars

New Tab Data Structures & A... Leisure reading C Unix Programming (JAVA) Sth Else (formally ot...

Accessing Scoped Variables

Traditional:

- attribute1: First Value
- attribute2: Second Value
- attribute3: Fri Jun 23 12:13:25 EDT 2023

```
<%= request.getAttribute("attribute1") %>
<%= session.getAttribute("attribute2") %>
<%= application.getAttribute("attribute3") %>
```

EL:

- attribute1: First Value
- attribute2: Second Value
- attribute3: Fri Jun 23 12:13:25 EDT 2023
- Source of "repeated" attribute: by Request
 - Source of "repeated reqScope: "" attribute: by Request
 - Source of "repeated sessScope" attribute: by Session
 - Source of "repeated appScope" attribute: by ServletContext

```
· ${attribute1}
· ${attribute2}
· ${attribute3}
· ${repeated}
  ${requestScope.repeated}
  > ${sessionScope.repeated}
  > ${applicationScope.repeated}
```

57



57

Example if else

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<html>
<head>
<title><c:choose> Tag Example</title>
</head>
<body>

<c:choose>
  <c:when test="${sessionScope.visitCount} gt 1">
    <p>Welcome back. This is your visit ${visitCount} </p>
  </c:when>
  <c:otherwise> <p>Welcome a new visitor!</p>
  </c:otherwise>
</c:choose>

</body>
</html>
```

58



58

Example loops

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ page import="java.util.Vector" %>

<html>
<head>
  <title>Tag Plugin Examples: forEach</title>
</head>
<body>
  <h3>Iterating over a range</h3>
  <c:forEach var="currentItem" begin="1" end="10">
    ${currentItem}
  </c:forEach>

  <% Vector<String> v = new Vector<>();
     v.add("One"); v.add("Two"); v.add("Three"); v.add("Four");
     pageContext.setAttribute("vector", v);
  %>

  <h3>Iterating over a Vector</h3>
  <c:forEach items="${vector}" var="currentItem" >
    ${currentItem}
  </c:forEach>
</body>
</html>
```

59

Iterating over a range

1 2 3 4 5 6 7 8 9 10

Iterating over a Vector

One Two Three Four

59

Display Bean properties

`${name.property}`
or
`${name["property"]}`

```
public class NameBean {
    private String firstName = "Missing first name";
    private String lastName = "Missing last name";

    public NameBean() {}

    public NameBean(String firstName, String lastName) {
        setFirstName(firstName);
        setLastName(lastName);
    }

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

    public void setFirstName(String newFirstName) {
        firstName = newFirstName;
    }

    public String getLastName() {
        return(lastName);
    }

    public void setLastName(String newLastName) {
        lastName = newLastName;
    }

    public String toString() {
        return "NameBean [ firstName="+ firstName + " , lastName="
            + lastName + "]";
    }
}
```

```
public class CompanyBean {
    private String companyName;
    private String business;

    public CompanyBean(String companyName, String business) {
        setCompanyName(companyName);
        setBusiness(business);
    }

    public String getCompanyName() { return(companyName); }

    public void setCompanyName(String newCompanyName) {
        companyName = newCompanyName;
    }

    public String getBusiness() { return(business); }

    public void setBusiness(String newBusiness) {
        business = newBusiness;
    }
}
```

```
public class EmployeeBean {
    private NameBean name;
    private CompanyBean company;

    public EmployeeBean(NameBean name, CompanyBean company) {
        setName(name);
        setCompany(company);
    }

    public NameBean getName() { return(name); }

    public void setName(NameBean newName) {
        name = newName;
    }

    public CompanyBean getCompany() { return(company); }

    public void setCompany(CompanyBean newCompany) {
        company = newCompany;
    }
}
```

62

```
public class BeanProperties extends HttpServlet {
    public void doGet(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException {
        NameBean nameB = new NameBean("Marty", "Hall");
        CompanyBean companyB = new CompanyBean("ABC.com",
            "J2EE Training and Consulting");

        request.setAttribute("name", nameB);
        request.setAttribute("company", companyB);

        RequestDispatcher dispatcher =
            request.getRequestDispatcher("bean-properties.jsp");
        dispatcher.forward(request, response);
    }
}
```

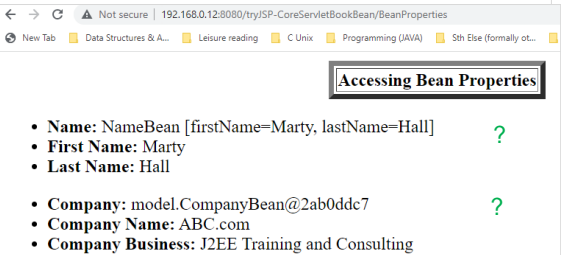
}
63

63

bean-properties.jsp

```
<TABLE BORDER=5 ALIGN="CENTER">
<TR><TH CLASS="TITLE">
Accessing Bean Properties
</TABLE>
<P>
<UL>
<LI><B>Name:</B> ${name}           // invoke toString()
<LI><B>First Name:</B> ${name.firstName}
<LI><B>Last Name:</B> ${name.lastName}
</UL><UL>
<LI><B>Company:</B> ${company}      // invoke toString()
<LI><B>Company Name:</B> ${company.companyName}
<LI><B>Company Business:</B> ${company.business}
</UL>
</BODY></HTML>
```

66



66


```
public class BeanProperties extends HttpServlet {
    public void doGet(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException {
        NameBean nameB = new NameBean("Marty", "Hall");
        CompanyBean companyB = new CompanyBean("ABC.com",
            "J2EE Training and Consulting");

        EmployeeBean employeeB = new EmployeeBean(nameB, companyB);

        request.setAttribute("employee", employeeB);

        RequestDispatcher dispatcher =
            request.getRequestDispatcher("bean-properties.jsp");
        dispatcher.forward(request, response);
    }
}
```

67



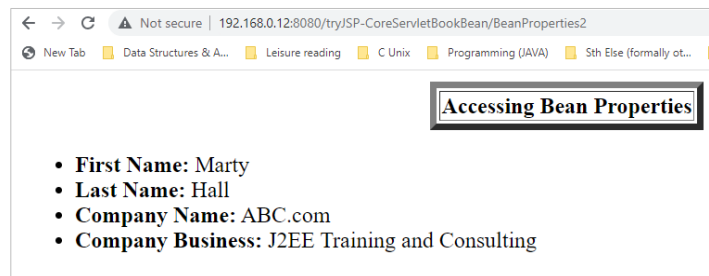
67

bean-properties.jsp

```

<TABLE BORDER=5 ALIGN="CENTER">
<TR><TH CLASS="TITLE">
Accessing Bean Properties
</TABLE>
<P>
<UL>
<LI><B>First Name:</B>  ${employee.name.firstName}
<LI><B>Last Name:</B>  ${employee.name.lastName}
<LI><B>Company Name:</B>  ${employee.company.companyName}
<LI><B>Company Business:</B>  ${employee.company.business}
</UL>
</BODY></HTML>

```



70

70

Accessing collections

- EL lets you access different types of collection (arrays, lists, maps)
- Items are accessed in the same way: using array notation.
 - array, list: `${collectionName[0]}`
 - Map: `${collectionName['key']}` Or `${collectionName.key}`
- Can use JSTL **forEach** tag to iterate over the collections


```

<c:forEach items="${collec}" var="currValue">
  ${currValue}
  ${currValue.property} // if item is bean object
</c:forEach>

```

71

71

```

public class Collections extends HttpServlet {
    public void doGet(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException {
        String[] firstNames = { "Bill", "Scott", "Larry" };

        ArrayList lastNames = new ArrayList();
        lastNames.add("Ellison");
        lastNames.add("Gates");
        lastNames.add("McNealy");

        HashMap companyNames = new HashMap();
        companyNames.put("Ellison", "Sun");
        companyNames.put("Gates", "Oracle");
        companyNames.put("McNealy", "Microsoft");

        request.setAttribute("firstN", firstNames);
        request.setAttribute("lastN", lastNames);
        request.setAttribute("companyN", companyNames);

        RequestDispatcher dispatcher =
            request.getRequestDispatcher("collections.jsp");
        dispatcher.forward(request, response);
    }
}

```



72

```

<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<UL>
  <LI> ${firstN[0]} ${lastN[0]} (${companyN["Ellison"]})
  <LI> ${firstN[1]} ${lastN[1]} (${companyN["Gates"]})
  <LI> ${firstN[2]} ${lastN[2]} (${companyN["McNealy"]})
</UL>
first names:    <!-- array -->
<ul>
  <c:forEach items="${firstN}" var="itemName">
    <li> ${itemName} </li>
  </c:forEach>
</ul>

last names:     <!-- arrayList -->
<ul>
  <c:forEach items="${lastN}" var="itemName">
    <li> ${itemName} </li>
  </c:forEach>
</ul>

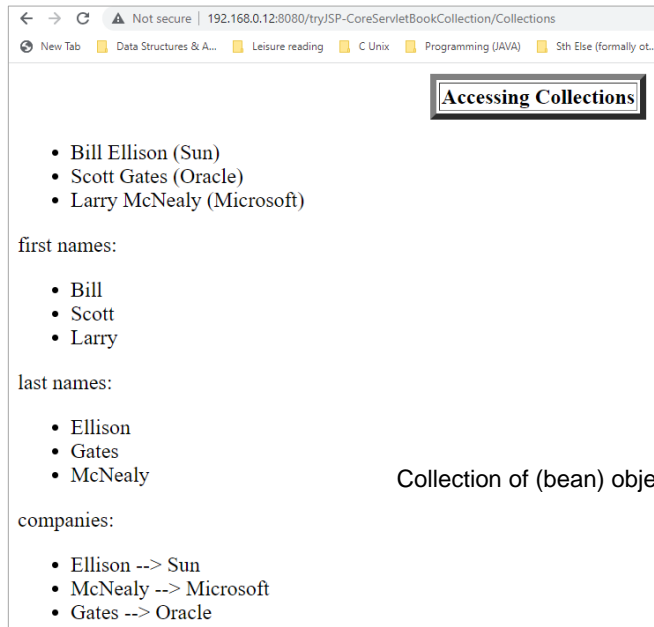
companies:      <!-- hashMap -->
<ul>
  <c:forEach items="${companyN}" var="item">
    <li> ${item.key} --> ${item.value} </li>
  </c:forEach>
</ul>

```



73

Add jar files



Accessing Collections

- Bill Ellison (Sun)
- Scott Gates (Oracle)
- Larry McNealy (Microsoft)

first names:

- Bill
- Scott
- Larry

last names:

- Ellison
- Gates
- McNealy

companies:

- Ellison --> Sun
- McNealy --> Microsoft
- Gates --> Oracle

Collection of (bean) objects?

74



Some examples we seen before

75



```

<form action="loginPage" method="post">
  User Name:<input type="text" name="uname"/><br/>
  Password:<input type="password" name="upass"/><br/>
  <input type="submit" value="SUBMIT"/>
</form>

```

index.html

```

@WebServlet("/loginPage")
public class Validation extends HttpServlet
{
    public void doPost(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException
    {
        response.setContentType("text/html");
        PrintWriter pwriter = response.getWriter();
        String name=request.getParameter("uname");
        String pass=request.getParameter("upass");
        if(name.equals("Chaitanya") && pass.equals("beginnersbook"))
        {
            request.setAttribute("upperName", name.toUpperCase());

            RequestDispatcher dis=request.getRequestDispatcher("welcome.jsp");
            dis.forward(request, response);
        }
        else
        {
            pwriter.print("User name or password is incorrect!");

            RequestDispatcher dis=request.getRequestDispatcher("index.html");
            dis.include(request, response);
        }
    }
}

```

Revisit the forward program

76

YORK UNIVERSITY

76

```

@WebServlet("/welcome")
public class WelcomeUser extends HttpServlet {
    public void doPost(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException
    {
        response.setContentType("text/html");
        PrintWriter pwriter = response.getWriter();
        pwriter.println("<html><body>");

        String name= request.getAttribute("upperName");

        pwriter.print("Hello " + name + "!");
        pwriter.print(" Welcome to Beginnersbook.com");
    }
}

```

welcome.jsp

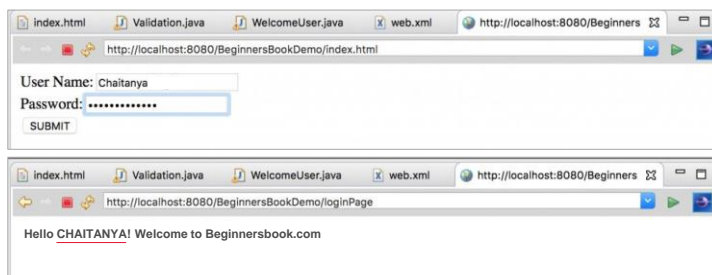
```

<%@ page language="java" contentType="text/html;
charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<p> Hello ${upperName}! Welcome to beginnersbook.com
</p>
</body>
</html>

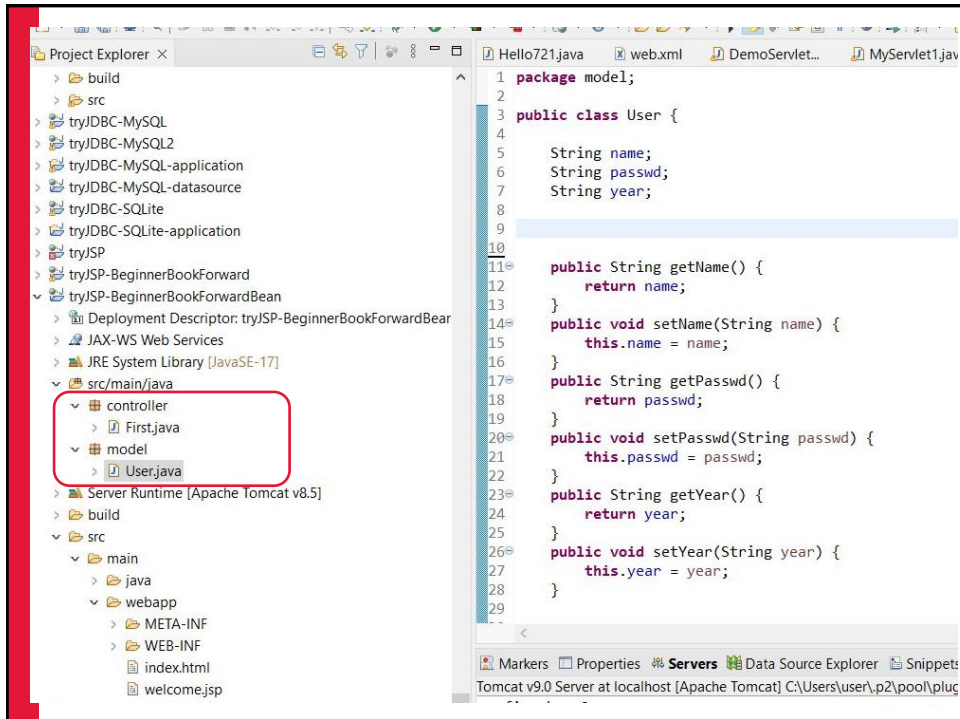
```

MVC model?

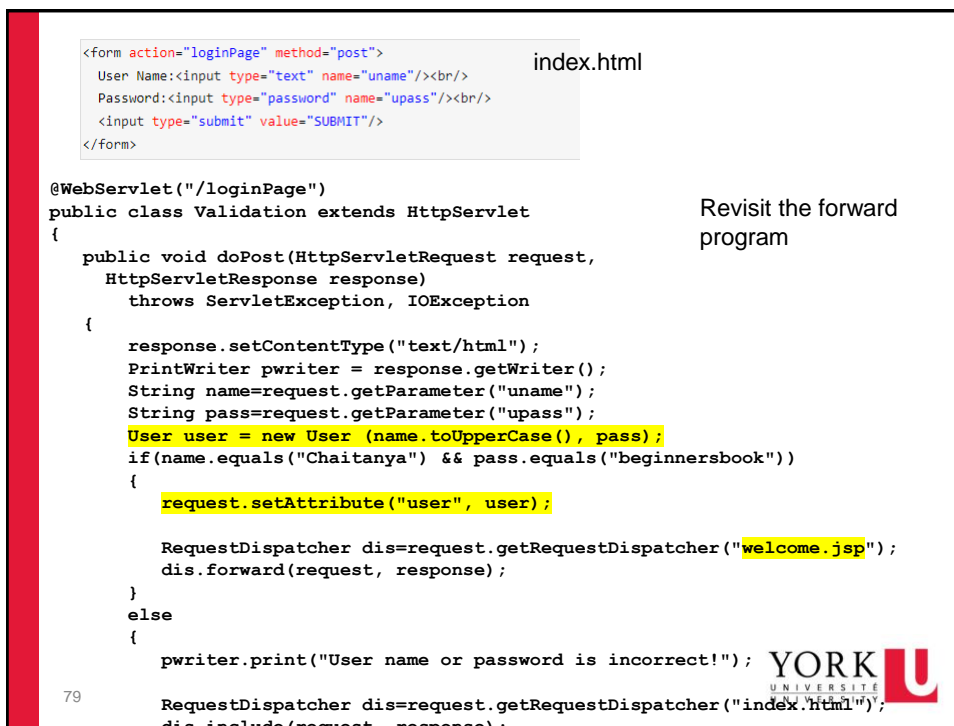
77



77



78



79

79

```

@WebServlet("/welcome")
public class WelcomeUser extends HttpServlet {

    public void doPost(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException
    {
        response.setContentType("text/html");
        PrintWriter pwriter = response.getWriter();
        pwriter.println("<html><body>");

        String name= request.getAttribute("upperName");

        pwriter.print("Hello " + name + "!");
        pwriter.print(" Welcome to Beginnersbook.com");
    }
}

```

welcome.jsp

```

<%@ page language="java" contentType="text/html;
charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<p> Hello ${user.name}! Welcome to beginnersbook.com
</p>
</body>
</html>

```

The screenshot shows two browser windows. The top window is at `http://localhost:8080/Beginners` and displays a login form with 'User Name: Chaitanya' and a masked password. The bottom window is at `http://localhost:8080/BeginnersBookDemo/loginPage` and displays the message 'Hello CHAITANYA! Welcome to Beginnersbook.com'.

80

80

Loan Calculator by 4413-23s

Calculator

Principal (total loan amount after studies)

Annual Interest Rate

Payment Period (total number of months)

[Calculate](#)

Student Loan Application

Client Protocol: HTTP/1.1
Client Method: GET
Query String: principal=3500&interest=6&period=12&out=html

Entered principal: 3500
Entered interest: 6
Entered period: 12

Welcome back. Calculation count: 3

Monthly Payment: 301.2

Calculation used:

principal: 3500 Interest: 6 Period: 12

[Re-calculate](#)

lab5

81

81

resultView.java

```
String pay = (String)request.getAttribute("payment");
String principle = (String)request.getAttribute("principal"); // used
String inter = (String)request.getAttribute("interest"); // used
String period = (String)request.getAttribute("period"); // used
out.println("<head>");
out.println("<link rel='StyleSheet' href='res/mc.css' type='text/css' ">");
out.println("</head>");

out.println("<body>");
out.println("<header>Student Loan Application</header>");

out.println("<p class='subtitle'>Client Protocol: " + request.getProtocol() + "</p>\n");
out.println("<p class='subtitle'> Client Method: " + request.getMethod() + "</p>");
out.println("<p class='subtitle'> Query String:" + request.getQueryString() + "</p>");

String originalPrinciple = request.getParameter("principal");
out.println("<p class='subtitle'> Entered principal: " + originalPrinciple + "</p>");
String originalInterest = request.getParameter("interest");
out.println("<p class='subtitle'> Entered interest: " + originalInterest + "</p>");
String originalPeriod = request.getParameter("period");
out.println("<p class='subtitle'> Entered period: " + originalPeriod + "</p> <hr>");

int c = (Integer)request.getSession().getAttribute("count");
if (c == 1)
    out.println("<p class='subtitle'>Welcome. This is first time calculation used");
else
    out.println("<p class='subtitle'>Welcome back. Calculation count: " + c);

out.println("<form method='get' action='UI.html' class='resultForm'>");
out.println("<h2>Monthly Payment: " + pay + "</h2>");
out.println("<br><br>");

out.println("Calculation used: </h2> <ul>");
out.println("<li> principal: " + principle + "</li>");
out.println("<li> Interest: " + inter + "</li>");
```

Student Loan Application

Client Protocol: HTTP/1.1
Client Method: GET
Query String: principal=3500&interest=6&period=12
Entered principal: 3500
Entered interest: 6
Entered period: 12
Welcome back. Calculation count: 3

Monthly Payment: 301.2

Calculation used:

principal: 3500 Interest: 6 Period: 12

Re-calculate

82

resultView.jsp

```
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<header>Student Loan Application</header>

<p class='subtitle'>Client Protocol: <%= request.getProtocol() %> </p>
<p class='subtitle'> Client Method: <%= request.getMethod() %> </p>
<p class='subtitle'> Query String: <%=request.getQueryString() %> </p>

<p class='subtitle'> Entered principal: ${param.principal }</p> // param ["principal"]
// or <%= request.getParameter("principal")%>
<p class='subtitle'> Entered interest: ${param.interest }</p>
<p class='subtitle'> Entered period: ${param.period }</p>

<hr>
<c:choose>
    <c:when test= "${sessionScope.count > 1}">
        <p class='subtitle'>Welcome back, Calculation count: ${count} </p></c:when>
    <c:otherwise> <p class='subtitle'>Welcome. This is first time calculation for you.
        </p></c:otherwise>
</c:choose>

<form method='get' action='UI.html' class='resultForm'>

    <h2>Monthly Payment: ${payment} <br><br>
    Calculation used: </h2>
    <ul>
        <li> principal: ${principal} </li>
        <li> Interest: ${requestScope.interest} </li>
        <li> Period: ${period} </li>
    </ul>

    <input type='submit' value='Re-calculate' />
</form>
```

Student Loan Application

Client Protocol: HTTP/1.1
Client Method: GET
Query String: principal=3500&interest=6&period=12
Entered principal: 3500
Entered interest: 6
Entered period: 12
Welcome back. Calculation count: 3

Monthly Payment: 301.2

Calculation used:

principal: 3500 Interest: 6 Period: 12

Re-calculate

MVC model?

83