

Expression Language, EL

What we can do with EL:

1. Accessing bean properties
 1. Direct
 2. Nested
2. Submitting bean properties
 1. Expressions in output values
 2. Expressions in submission values
 3. Expressions for action controllers
3. Accessing collection elements
4. Using implicit objects and operators
5. Conditionally rendering output
6. Passing arguments to methods
7. Has set of simple operators (arithmetic, relational, logical)
8. We can do conditional (IFs) for output
9. Predefined variables (implicit objects)
 1. To access request params, cookies, HTTP headers, and other standard types of request data, you can use one of several predefined implicit objects.
10. Passing arguments
 1. We pass arbitrary arguments to methods. Works only in Java EE 6 or other servers that support EL 2.1. Not part of JSF 2 itself.

Examples, accessing beans:

– `#{employee.firstName}`

Call `getFirstName` on bean named `employee`. Output it.

– `<h:inputText value="#{employee.firstName}"/>`

–

– When form displayed, call `getFirstName`, and if non-empty, fill it in as initial value of textfield.

– When form submitted, validate value and if it is OK, pass value to the `setFirstName` method

•

– `#{employee.addresses[0].zip}`

– Call `getAddresses` on bean named `employee` (which should return an array or list), then take first entry, then call `getZip` on that, then output it

Advantages of the Expression Language

– Shorthand notation for bean properties

– To reference the result of the `getCompanyName()` method of a managed bean named `company`, you use `#{company.companyName}`.

- To reference the firstName property of the president property of a managed bean named company, you use `#{company.president.firstName}`.
- Simple access to collection elements
 - To reference an element of an array, List, or Map, you use `#{someBean.someProperty[indexOrKey]}`: `#{person.friends[2]}`

1. Create a new Web App, LabEL1

2. Create a SimpleBean

```

1 package edu.slcc.asdv.beans;
2
3 import ...2 lines
4
5
6 @Named(value = "simpleBean")
7 @RequestScoped
8 public class SimpleBean
9 {
10     private String[] colors =
11     {
12         "red", "orange", "yellow"
13     };
14     public String getMessage()
15     {
16         return ("Hello, Friend!");
17     }
18     public String[] getColors()
19     {
20         return colors;
21     }
22 }

```

3. Test the bean from your index. Clean build and run.

```

1 <?xml version='1.0' encoding='UTF-8' ?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Tra
3 <html xmlns="http://www.w3.org/1999/xhtml"
4       xmlns:h="http://xmlns.jcp.org/jsf/html">
5   <h:head>
6       <title>Test Page</title>
7   </h:head>
8   <h:body>
9       Message: #{simpleBean.message}
10      <br/>
11      First color: #{simpleBean.colors[0]}
12      <br/>
13   </h:body>
14 </html>
15

```

4. Create a TestBean1, Application scope.

```

1 package edu.slcc.asdv.beans;
2 import ...3 lines
5 @Named(value = "testBean1")
6 @ApplicationScoped
7 public class TestBean1
8 {
9     private Date creationTime = new Date();
10    private String greeting = "Hello";
11
12    public Date getCreationTime()
13    {
14        return creationTime;
15    }
16    public String getGreeting()
17    {
18        return greeting;
19    }
20    public double getRandomNumber()
21    {
22        return (Math.random());
23    }
24

```

5. Test it from index. Lines 13-21.

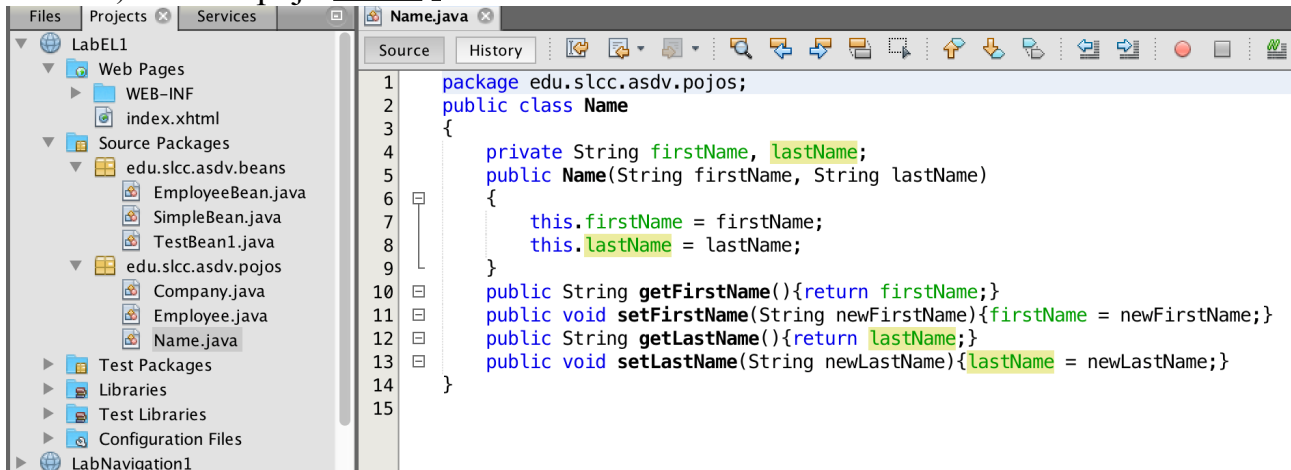
```

5 <h:head>
6   <title>Test Page</title>
7 </h:head>
8 <h:body>
9   Message: #{simpleBean.message}
10  <br/>
11  First color: #{simpleBean.colors[0]}
12  <br/>
13  -----
14  <br/>
15  Creation time: #{testBean1.creationTime}
16  <br/>
17  Greeting: #{testBean1.greeting}
18  <br/>
19  Random number: #{testBean1.randomNumber}
20  <br/>
21  -----
22  <br/>

```

6. Accessing Nested properties of POJOs

a) Create pojo Name. Use Netbeans Insert code.



```

1 package edu.slcc.asdv.pojos;
2 public class Name
3 {
4     private String firstName, lastName;
5     public Name(String firstName, String lastName)
6     {
7         this.firstName = firstName;
8         this.lastName = lastName;
9     }
10    public String getFirstName(){return firstName;}
11    public void setFirstName(String newFirstName){firstName = newFirstName;}
12    public String getLastName(){return lastName;}
13    public void setLastName(String newLastName){lastName = newLastName;}
14 }
15

```

b) Create pojo Company

```

1 package edu.slcc.asdv.pojo;
2 public class Company
3 {
4     private String companyName, business;
5
6     public Company(String companyName, String business)
7     {
8         this.companyName = companyName;
9         this.business = business;
10    }
11
12    public String getCompanyName(){return companyName;}
13    public void setCompanyName(String newCompanyName){companyName = newCompanyName;}
14    public String getBusiness(){return business;}
15    public void setBusiness(String newBusiness){business = newBusiness;}
16 }
17

```

c) Create pojo Employee

```

1 package edu.slcc.asdv.pojo;
2
3 public class Employee
4 {
5     private Name name;
6     private Company company;
7     public Employee(Name name, Company company)
8     {
9         this.name = name;
10        this.company = company;
11    }
12    public Name getName(){return name;}
13    public Company getCompany(){return company;}
14    public String processEmployee()
15    {
16        if (Math.random() < 0.5) return ("accepted");
17        else return ("rejected");
18    }
19 }

```

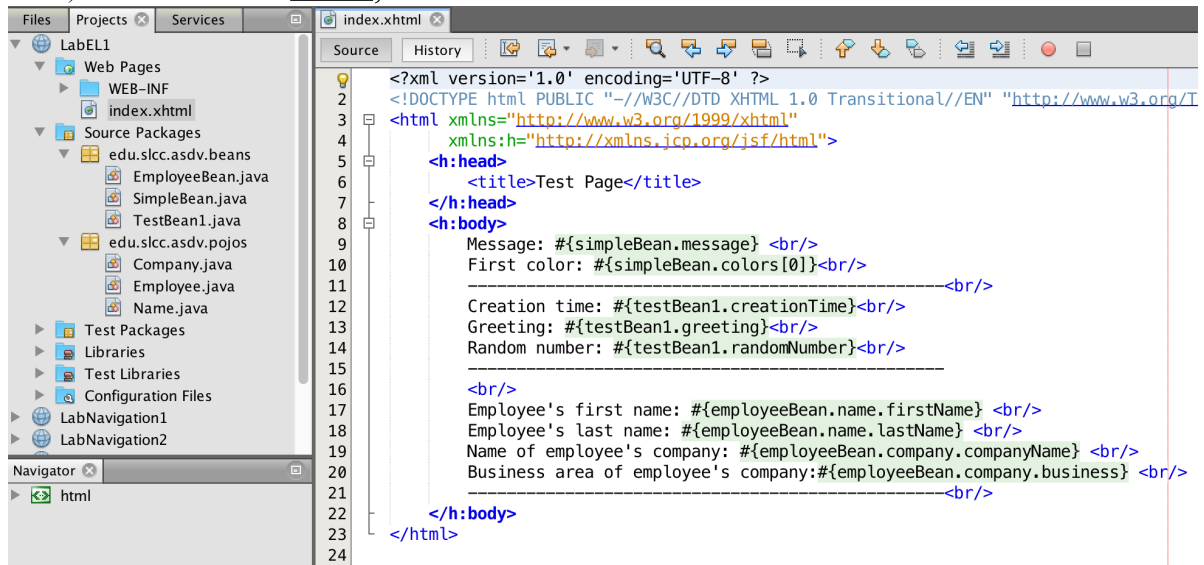
d) Create bean EmployeeBean

```

1 package edu.slcc.asdv.beans;
2
3 import ..5 lines
4
5 @Named(value = "employeeBean")
6 @RequestScoped
7 public class EmployeeBean extends Employee
8 {
9     public EmployeeBean()
10    {
11        super( new Name( "Jan", "Databaseux"),
12              new Company("DBaseSoft.com",
13                          "Developing Database Solutions")
14            );
15    }
16 }
17
18
19
20
21
22

```

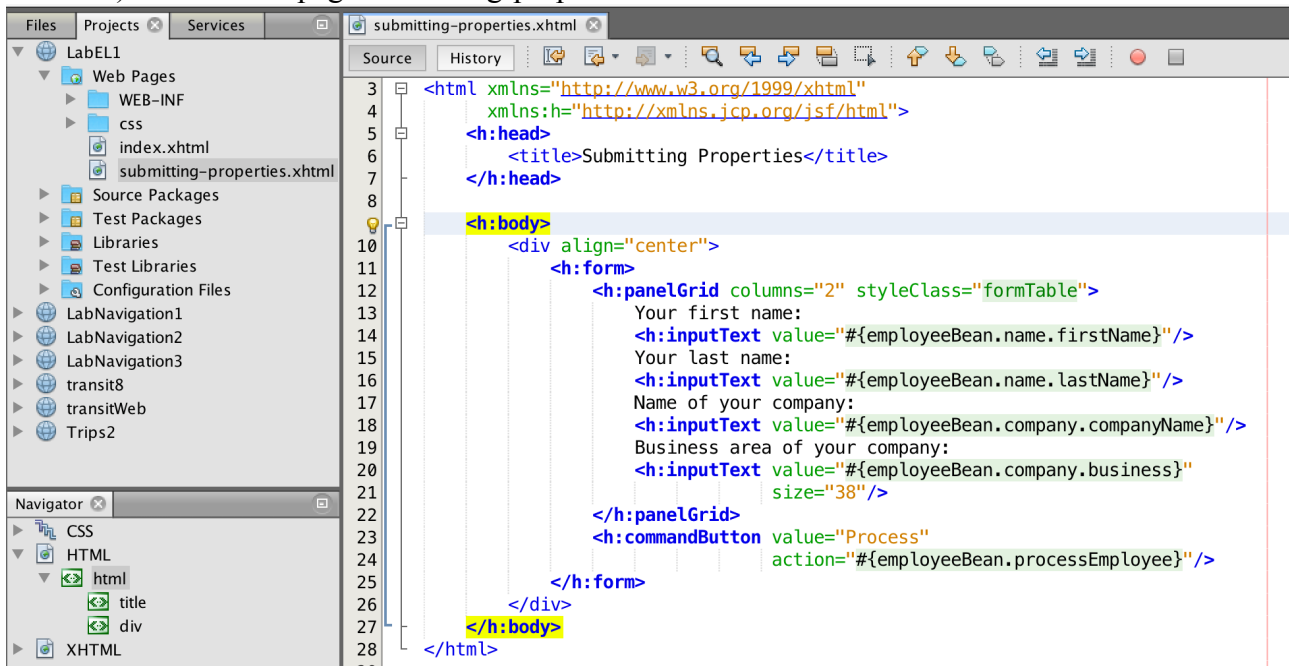
e) Test it from index, lines 17-20.



```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/T
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://xmlns.jcp.org/jsf/html">
  <h:head>
    <title>Test Page</title>
  </h:head>
  <h:body>
    Message: #{simpleBean.message} <br/>
    First color: #{simpleBean.colors[0]}<br/>
    -----<br/>
    Creation time: #{testBean1.creationTime}<br/>
    Greeting: #{testBean1.greeting}<br/>
    Random number: #{testBean1.randomNumber}<br/>
    -----<br/>
    Employee's first name: #{employeeBean.name.firstName} <br/>
    Employee's last name: #{employeeBean.name.lastName} <br/>
    Name of employee's company: #{employeeBean.company.companyName} <br/>
    Business area of employee's company:#{employeeBean.company.business} <br/>
    -----<br/>
  </h:body>
</html>
```

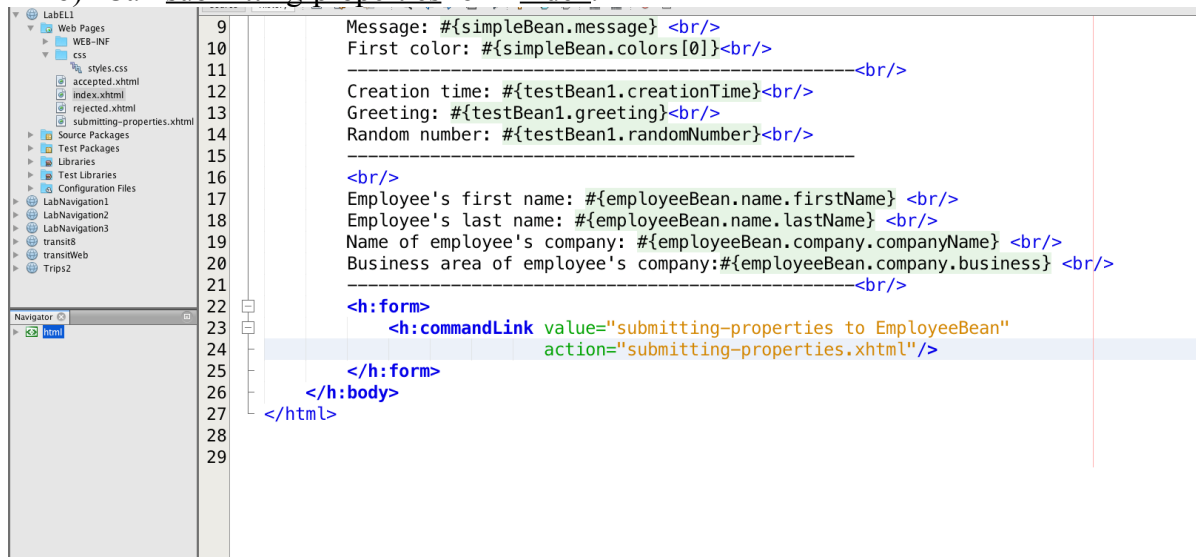
7. Submitting Bean properties

a) Create JSF page submitting-properties.xhtml



```
3 <html xmlns="http://www.w3.org/1999/xhtml"
4       xmlns:h="http://xmlns.jcp.org/jsf/html">
5   <h:head>
6       <title>Submitting Properties</title>
7   </h:head>
8
9   <h:body>
10      <div align="center">
11          <h:form>
12              <h:panelGrid columns="2" styleClass="formTable">
13                  Your first name:
14                  <h:inputText value="#{employeeBean.name.firstName}"/>
15                  Your last name:
16                  <h:inputText value="#{employeeBean.name.lastName}"/>
17                  Name of your company:
18                  <h:inputText value="#{employeeBean.company.companyName}"/>
19                  Business area of your company:
20                  <h:inputText value="#{employeeBean.company.business}"
21                              size="38"/>
22              </h:panelGrid>
23              <h:commandButton value="Process"
24                              action="#{employeeBean.processEmployee}"/>
25          </h:form>
26      </div>
27  </h:body>
28 </html>
```

b) Call submitting-properties form index.



```
9 Message: #{simpleBean.message} <br/>
10 First color: #{simpleBean.colors[0]}<br/>
11 -----<br/>
12 Creation time: #{testBean1.creationTime}<br/>
13 Greeting: #{testBean1.greeting}<br/>
14 Random number: #{testBean1.randomNumber}<br/>
15 -----<br/>
16 Employee's first name: #{employeeBean.name.firstName} <br/>
17 Employee's last name: #{employeeBean.name.lastName} <br/>
18 Name of employee's company: #{employeeBean.company.companyName} <br/>
19 Business area of employee's company:#{employeeBean.company.business} <br/>
20 -----<br/>
21
22 <h:form>
23     <h:commandLink value="submitting-properties to EmployeeBean"
24                   action="submitting-properties.xhtml"/>
25 </h:form>
26 </h:body>
27 </html>
```


c) Add JSF page `accepted` (observe the link for css in header)

```

1 <?xml version='1.0' encoding='UTF-8' ?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1"
3 <html xmlns="http://www.w3.org/1999/xhtml"
4     xmlns:h="http://xmlns.jcp.org/jsf/html">
5     <h:head>
6         <title>Employee Accepted </title>
7         <link href="./css/styles.css" rel="stylesheet" type="text/css"/>
8     </h:head>
9     <h:body>
10        <h1>Employee Accepted</h1>
11        <div align="center">
12            <ul>
13
14                <li> Employee's first name: #{employeeBean.name.firstName}</li>
15                <li> Employee's last name: #{employeeBean.name.lastName} </li>
16                <li> Name of employee's company: #{employeeBean.company.companyName} </li>
17                <li> Business area of employee's company:#{employeeBean.company.business} </li>
18            </ul>
19        </div>
20    </h:body>
21 </html>
22
  
```

d) Add JSF page `rejected` (observe the link for css in header)

```

1 <?xml version='1.0' encoding='UTF-8' ?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/
3 <html xmlns="http://www.w3.org/1999/xhtml"
4     xmlns:h="http://xmlns.jcp.org/jsf/html">
5     <h:head>
6         <title>Employee Rejected</title>
7         <link href="./css/styles.css" rel="stylesheet" type="text/css"/>
8     </h:head>
9     <h:body>
10        <h1>Employee Rejected</h1>
11        <br/>
12        <div align="center">
13
14            <h:panelGrid columns="1" styleClass="error">
15
16                Employee's first name: #{employeeBean.name.firstName} <br/>
17                Employee's last name: #{employeeBean.name.lastName} <br/>
18                Name of employee's company: #{employeeBean.company.companyName} <br/>
19                Business area of employee's company:#{employeeBean.company.business} <br/>
20            </h:panelGrid>
21        </div>
22    </h:body>
23 </html>
24
25
26
  
```

8. Clean build and run.

How getters setters are invoked

– `<h:inputText value="#{myBean.a.b.c.d}"/>`

- When **displaying** form

- Find or instantiate myBean. Call getA. Call getB on result. Call getC on that result. Call getD on that result. If non-empty use as initial value of textfield.

- When **submitting** form

- Find myBean (instantiate new version if in request scope). Call getA. Call getB on result. Call getC on that result. Then pass submitted value to the setD method of that result.