

Expression Language, EL II

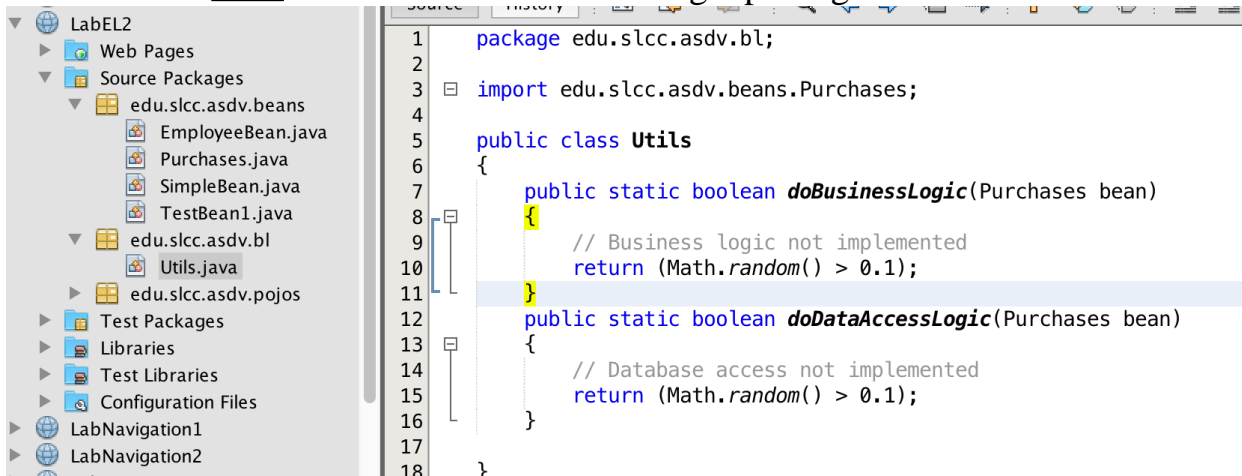
Accessing Collections

Equivalence of Dot and Array

Notations

- Equivalent forms
 - `#{name.property}`
 - Legal ONLY if “property” is **LEGAL** Java variable name
 - `#{name["property"]}`
- Reasons for using bracket notation
 - To access arrays, lists, and other collections
 - To calculate the property name at request time.
 - `#{name1[name2]}` (no quotes around name2)
 - To use names that are **ILLEGAL** as Java variable names
 - `#{foo["bar-baz"]}`
 - `#{foo["bar.baz"]}`

1. Copy LabeEL1 and rename it Label2
2. Create a Utils class under the business logic package



The screenshot shows an IDE with a project structure on the left and a Java source file on the right. The project structure includes a package named `edu.slcc.asdv.bl` containing a class `Utils.java`. The source file `Utils.java` contains the following code:

```
1 package edu.slcc.asdv.bl;
2
3 import edu.slcc.asdv.beans.Purchases;
4
5 public class Utils
6 {
7     public static boolean doBusinessLogic(Purchases bean)
8     {
9         // Business logic not implemented
10        return (Math.random() > 0.1);
11    }
12    public static boolean doDataAccessLogic(Purchases bean)
13    {
14        // Database access not implemented
15        return (Math.random() > 0.1);
16    }
17 }
18 }
```

3. Create a request scoped bean `Purchases` that has an Array, a List and a Map.

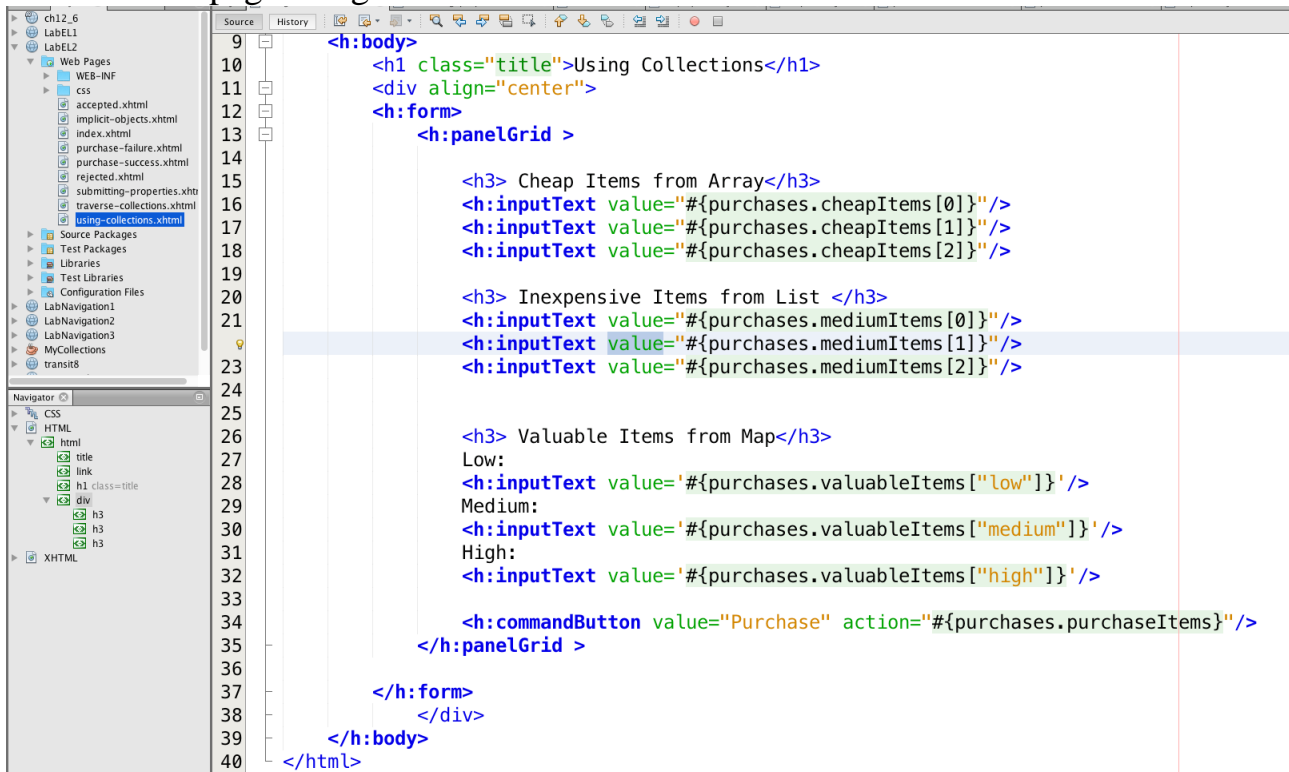
```
6 import java.util.List;
7 import java.util.Map;
8 import javax.enterprise.context.RequestScoped;
9 import javax.inject.Named;
10
11 @Named(value = "purchases")
12 @RequestScoped
13 public class Purchases
14 {
15     private String[] cheapItems = {"Gum", "Yo-yo", "Pencil"};
16     private List<String> mediumItems= new ArrayList<>();
17     private Map<String, String> valuableItems = new HashMap<>();
18     private boolean isEverythingOK = true;
19
20     public Purchases()
21     {
22         mediumItems.add("iPod");
23         mediumItems.add("GameBoy");
24         mediumItems.add("Cell Phone");
25         valuableItems.put("low", "Porche");
26         valuableItems.put("medium", "Yacht");
27         valuableItems.put("high", "Oracle, Training Course ");
28     }
29     public String[] getCheapItems(){return cheapItems;}
30     public List<String> getMediumItems(){return mediumItems;}
31     public Map<String, String> getValuableItems(){return valuableItems;}
32     public String purchaseItems()
33     {
34         isEverythingOK = Utils.doBusinessLogic(this);
35         isEverythingOK = Utils.doDataAccessLogic(this);
36         if (isEverythingOK) return "purchase-success";
37         else return "purchase-failure";
38     }
39 }
40
```

4. Add JSF pages `purchase-success.xhtml` and `purchase-failure.xhtml`.

```
3 <html xmlns="http://www.w3.org/1999/xhtml"
4     xmlns:h="http://xmlns.jcp.org/jsf/html">
5     <h:head>
6         <title>Success</title>
7     </h:head>
8     <h:body>
9         <h1 class="title">Successful Purchase</h1>
10        <h3>Your account was successfully charged.
11        Thanks for shopping with us.</h3>
12        <ul>
13            <li><b>Cheap Items</b></li>
14            <ol>
15                <li>#{purchases.cheapItems[0]}</li>
16                <li>#{purchases.cheapItems[1]}</li>
17                <li>#{purchases.cheapItems[2]}</li>
18            </ol></li>
19            <li><b>Medium Items</b></li>
20            <ol>
21                <li>#{purchases.mediumItems[0]}</li>
22                <li>#{purchases.mediumItems[1]}</li>
23                <li>#{purchases.mediumItems[2]}</li>
24            </ol></li>
25            <li><b>Valuable Items</b></li>
26            <ul>
27                <li>Low: #{purchases.valuableItems["low"]}</li>
28                <li>Medium: #{purchases.valuableItems["medium"]}</li>
29                <li>High: #{purchases.valuableItems["high"]}</li>
30            </ul></li>
31        </ul>
32    </h:body>
33 </html>
```

```
3 <html xmlns="http://www.w3.org/1999/xhtml"
4     xmlns:h="http://xmlns.jcp.org/jsf/html">
5     <h:head>
6         <title>Failure</title>
7     </h:head>
8     <h:body>
9         <h1 class="title">Failure</h1>
10        <h3>Insufficient balance. Go away, cheapskate.</h3>
11        <ul>
12            <li><b>Cheap Items</b></li>
13            <ol>
14                <li>#{purchases.cheapItems[0]}</li>
15                <li>#{purchases.cheapItems[1]}</li>
16                <li>#{purchases.cheapItems[2]}</li>
17            </ol></li>
18            <li><b>Medium Items</b></li>
19            <ol>
20                <li>#{purchases.mediumItems[0]}</li>
21                <li>#{purchases.mediumItems[1]}</li>
22                <li>#{purchases.mediumItems[2]}</li>
23            </ol></li>
24            <li><b>Valuable Items</b></li>
25            <ul>
26                <li>Low: #{purchases.valuableItems["low"]}</li>
27                <li>Medium: #{purchases.valuableItems["medium"]}</li>
28                <li>High: #{purchases.valuableItems["high"]}</li>
29            </ul></li>
30        </ul>
31    </h:body>
32 </html>
```

5. Create page using-collections.



The screenshot shows an IDE with a project structure on the left and a source code editor on the right. The project structure includes 'Web Pages' with sub-folders 'WEB-INF', 'css', and 'source Packages'. The 'source Packages' folder contains several files, including 'using-collections.xhtml'. The source code editor displays the following code:

```
9 <h:body>
10 <h1 class="title">Using Collections</h1>
11 <div align="center">
12 <h:form>
13 <h:panelGrid >
14
15 <h3> Cheap Items from Array</h3>
16 <h:inputText value="#{purchases.cheapItems[0]}" />
17 <h:inputText value="#{purchases.cheapItems[1]}" />
18 <h:inputText value="#{purchases.cheapItems[2]}" />
19
20 <h3> Inexpensive Items from List </h3>
21 <h:inputText value="#{purchases.mediumItems[0]}" />
22 <h:inputText value="#{purchases.mediumItems[1]}" />
23 <h:inputText value="#{purchases.mediumItems[2]}" />
24
25
26 <h3> Valuable Items from Map</h3>
27 Low:
28 <h:inputText value="#{purchases.valuableItems["low"]}" />
29 Medium:
30 <h:inputText value="#{purchases.valuableItems["medium"]}" />
31 High:
32 <h:inputText value="#{purchases.valuableItems["high"]}" />
33
34 <h:commandButton value="Purchase" action="#{purchases.purchaseItems}" />
35 </h:panelGrid >
36
37 </h:form>
38 </div>
39 </h:body>
40 </html>
```

6. Call it from index

```
-----<br />
<h:form>
  <h:commandLink value="submitting-properties to EmployeeBean"
    action="submitting-properties" />
</h:form>
-----<br />
<h:form>
  <h:commandLink value="Using Collections"
    action="using-collections" />
</h:form>
-----<br />
<h:form>
```