© Copyright 2003, Sun Microsystems, Inc.

JavaServer Pages and the Expression Language

Bryan Basham
Sun Microsystems, Inc.
bryan.basham@sun.com

© Copyright 2003, Sun Microsystems, Inc.

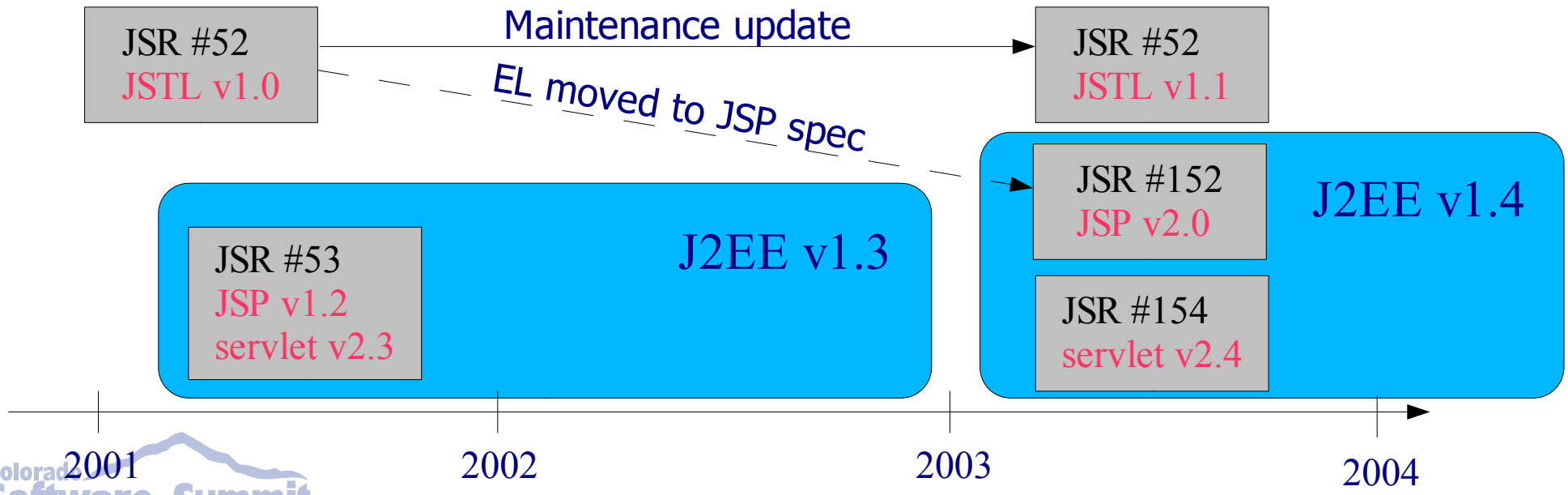
Topics Covered

- History of the Expression Language (EL)
- Overview of the EL
- EL Namespace
- EL Operators
- EL Functions

© Copyright 2003, Sun Microsystems, Inc.

History of the EL

- Proposed in the JSP Standard Tag Library (JSTL) specification (JSR #52)
- Incorporated into the JSP v2.0 specification (JSR #152)



© Copyright 2003, Sun Microsystems, Inc.

Topics Covered

- History of the Expression Language (EL)
- Overview of the EL
- EL Namespace
- EL Operators
- EL Functions

© Copyright 2003, Sun Microsystems, Inc.

Refactoring Example

- Before EL:

```
<jsp:useBean id="customer" scope="session" type="com.my.Customer"/>
```

Welcome, `<jsp:getProperty name="customer" property="firstName"/>`, to our web site...

- Using EL:

Welcome, `${sessionScope.customer.firstName}`, to our web site...

- Or simply:

Welcome, `${customer.firstName}`, to our web site...

© Copyright 2003, Sun Microsystems, Inc.

Definition of the EL

- Syntax: `${ expr1 }`
- Similar to: `<%= expr2 %>`
- But, the syntax of **`expr2`** is Java
- Whereas, the syntax of **`expr1`** is EL (not Java)

© Copyright 2003, Sun Microsystems, Inc.

Definition of the EL *(Continued)*

- Expressions can be used in:
  - Template text: `Welcome, ${customer.firstName}...`
  - Tag attributes: `<c:out value="${customer.firstName}"/>`
- Expressions cannot be used when:
  - In non-RT tag attributes, such as `jsp:useBean/id`
  - In the body of a tag with `tagdependent` body type
  - EL processing has been disabled
    - In deployment descriptor
    - Using `<%@ page isElIgnored="true" %>`

© Copyright 2003, Sun Microsystems, Inc.

Syntax of the EL

Expression ::= Expression BinOp Expression | UnaryExpression

UnaryExpression ::= UnaryOp UnaryExpression | Value

Value ::= ValuePrefix | Value ValueSuffix

ValuePrefix ::= Literal | '(' Expression ')' | Identifier
| ImplicitObject | FunctionInvocation

ValueSuffix ::= '.' Identifer | '[' Expression ']'

Literal ::= BooleanLit | IntegerLit | FloatLit | StringLit | NullLit

FunctionInvocation ::=
[Identifier ':'] Identifier '(' ParameterList ')'

ParameterList ::= Expression | Expression ',' ParameterList

© Copyright 2003, Sun Microsystems, Inc.

Operator Precedence

```
1  []   .
2  ()
3  not   !   -   empty
4  *   /   div   %   mod
5  +   -
6  <   lt   >   gt   <=   le   >=   ge
7  ==   eq   !=   ne
8  &&   and
9  !!   or
```

© Copyright 2003, Sun Microsystems, Inc.

Topics Covered

- History of the Expression Language (EL)
- Overview of the EL
- EL Namespace
- EL Operators
- EL Functions

© Copyright 2003, Sun Microsystems, Inc.

Implicit Objects

- `pageContext` the PageContext object
- `pageScope` a Map to page-scoped attributes
- `requestScope` a Map to request-scoped attributes
- `sessionScope` a Map for session-scoped attributes
- `ApplicationScope` a Map for appl-scoped attributes
- `initParam` a Map for context initialization params

© Copyright 2003, Sun Microsystems, Inc.

Implicit Objects

- `param` a Map from param name to one value
- `paramValues` a Map from param name to all values
- `header` a Map from header name to one value
- `headerValues` a Map from header name to all values
- `cookie` a Map from cookie name to one Cookie

© Copyright 2003, Sun Microsystems, Inc.

Implicit Object Examples

- Retrieve the request URI:
  `${pageContext.request.requestURI}`
- Retrieve a property of an attribute of the session:
  `${sessionScope.customer.firstName}`
- Alternate syntax:
  `${sessionScope['customer'].firstName}`
- Retrieve the `userName` parameter:
  `${param['userName']}`
- Alternate syntax:
  `${param.userName}`

© Copyright 2003, Sun Microsystems, Inc.

WebApp Attributes

- You can use an attribute name as the primary object:
  `${user.userName}`
- This is equivalent to:
  `<%= pageContext.findAttribute("user").getUserName() %>`
- It searches the scopes in this order:
  page, request, session, then application

© Copyright 2003, Sun Microsystems, Inc.

Topics Covered

- History of the Expression Language (EL)
- Overview of the EL
- EL Namespace
- EL Operators
- EL Functions

© Copyright 2003, Sun Microsystems, Inc.

Dot . and Bracket [ ] Operators

- Accessing maps:
  `${map['key']}` is equivalent to `${map.key}`
  `${pageScope['customer']}` is equiv. to `${pageScope.customer}`
- Accessing bean properties:
  `${bean['property']}` is equivalent to `${bean.property}`
  `${customer['firstName']}` is equiv. to `${customer.firstName}`
- Accessing lists and arrays:
  `${list[index]}`
  `${products[idx + 1]}`
- Can combine arbitrarily:
  `${user.address.city}`
  `${orders[productId].customer.address.city}`

© Copyright 2003, Sun Microsystems, Inc.

Arithmetic Operators

- Operators: `+ - * / div % mod`
- Examples:

```
${7 + 5}
${product.unitCost * orders[product.ID].quantity}
```

- Handling null values:

`${7 + null}` is evaluated to `7` (similar for the - and * ops)

`${7 div null}` is evaluated to `Infinity`

`${7 % null}` results in a runtime exception

© Copyright 2003, Sun Microsystems, Inc.

Relational Operators

- Operators: `== eq != ne < lt > gt <= le >= ge`
- Examples:

  `${7 < 5}` is equivalent to `${7 lt 5}`

  ```
  ${orders[product.ID].quantity != 0}
  <c:if test="${promotion.value gt 100.0}">...</c:if>
  ```

- Handling null values:

  `${7 < null}` is evaluated to `false` (as all comparator ops)

  `${7 == null}` is evaluated to `false`

  `${null == null}` is evaluated to `true`

  `${7 != null}` is evaluated to `true`

© Copyright 2003, Sun Microsystems, Inc.

Logical Operators

- Operators: `&&` and `||` or `!` not
- Examples:

  `${true && false}` is equivalent to `${true and false}`

  `${(product.cost > 100.0) and ! (quantity < 25)}`
- Handling null values:

  `${true and null}` is evaluated to `false`

  `${true or null}` is evaluated to `true`

  `${not null}` is evaluated to `true`
- Short-circut evaluation:

  `${A && B && C}` does not evaluate C if B is false

© Copyright 2003, Sun Microsystems, Inc.

The empty Operator

- This operator `${empty A}` evaluates to `true` if the operand A is null or "empty".
- The `true` cases:
  - A is null reference
  - A is an empty String
  - A is an empty Map or List or array

© Copyright 2003, Sun Microsystems, Inc.

Topics Covered

- History of the Expression Language (EL)
- Overview of the EL
- EL Namespace
- EL Operators
- EL Functions

© Copyright 2003, Sun Microsystems, Inc.

EL Function Syntax

- Syntax:

taglib prefix function name

FunctionInvocation ::=
  [Identifier ':'] Identifier '(' ParameterList ')'

ParameterList ::= Expression | Expression ',' ParameterList

- Examples:

```
There are ${fn:length(athletes[country])} athletes
representing ${country}.
```

```
<c:url var="myUrl" value="${base}/cust">
   <c:param name="custId" value="${fn:trim(custId)}"/>
</c:url>
```

© Copyright 2003, Sun Microsystems, Inc.

JSTL Pre-Defined Functions

- Length function: `length`
- String compare: `contains`, `containsIgnore`, `endsWith`, and `startsWith`
- String search: `indexOf`
- String modification: `replace`, `substring`, `substringAfter`, `substringBefore`, `toLowerCase`, `toUpperCase`, `trim`, and `escapeXml`
- String spliting: `split` and `join`

© Copyright 2003, Sun Microsystems, Inc.

How to Define a Function

- An EL function must be implemented as a static function in a Java class.
- You can define as many functions in a single class.
- For example:

```
package org.basham.util.web.el;
public final class MyFunctions {
  public static double sqrt(double value) {
    return Math.sqrt(value);
  }
}
```

© Copyright 2003, Sun Microsystems, Inc.

How to Declare the Function

Functions are defined in a Tag Library Desriptor file:

```
<taglib>
  <function>
    <name>sqrt</name>
    <function-class>
      org.basham.util.web.el.MyFunctions
    </function-class>
    <function-signature>
      double sqrt(double)
    </function-signature>
  </function>
</taglib>
```

© Copyright 2003, Sun Microsystems, Inc.

Summary

- History of the Expression Language (EL)
- Overview of the EL
- EL Namespace
- EL Operators
- EL Functions
- Questions?