|  |  |
| --- | --- |
|  | **JSP EL**  **(JSP Expression Language)** |
|  | we can use **scriptlets** and **JSP expressions** to retrieve attributes and parameters in JSP with java code and use it for view purpose. But for web designers, java code is hard to understand and that’s why JSP Specs 2.0 introduced **Expression Language** (EL) through which we can get attributes and parameters easily using HTML like tags. |
|  |  |
|  | JSP EL Implicit Objects |
|  | JSP Expression Language provides many implicit objects that we can use to get attributes from different scopes and parameter values. The list is given below. |
|  | 9.png |
|  | 7.png |
|  | 8.png |
|  | 9.png |
|  |  |
|  | JSP EL Operators |
|  |  |
| 1. | **EL Property Access Operator or Dot (.) Operator** |
|  | JSP EL Dot operator is used to get the attribute values. |
|  | ${firstObj.secondObj} |
|  | In above expression, firstObj can be EL implicit object or an attribute in page, request, session or application scope. For example, |
|  | ${requestScope.employee.address} |
|  | Note that except the last part of the EL, all the objects should be either Map or Java Bean, so in above example requestScope is a Map and employee should be a Java Bean or Map. If scope is not provided, the JSP EL looks into page, request, session and application scope to find the named attribute. |
|  |  |
| 2. | **EL [] Operator or Collection Access Operator** |
|  | [] operator is more powerful than dot operator. We can use it to get data from List and Array too. |
|  | Some examples; |
|  | ${myList[1]} and ${myList[“1”]} are same, we can provide List or Array index as String literal also. |
|  | ${myMap[expr]} – if the parameter inside [] is not String, it’s evaluated as an EL. |
|  | ${myMap[myList[1]]} – [] can be nested. |
|  | ${requestScope[“foo.bar”]} – we can’t use dot operator when attribute names have dots. |
|  |  |
| 3. | **EL Arithmetic Operators** |
|  | Arithmetic operators are provided for simple calculations in EL expressions. They are +, -, \*, / or div, % or mod. |
|  |  |
| 4. | **EL Logical Operators** |
|  | They are && (and), || (or) and ! (not). |
|  |  |
| 5. | **EL Relational Operators** |
|  | They are == (eq), != (ne), < (lt), > (gt), <= (le) and >= (ge). |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | JSP EL Operator Precedence |
|  | JSP EL expressions are evaluated from left to right. JSP EL Operator precedence is listed in below table from highest to lowest. |
|  | 4.png |
|  | 5.png |
|  |  |
|  | JSP EL Reserve Words |
|  | 6.png |
|  |  |
|  | JSP EL Important Points |
| 1. | EL expressions are always within curly braces prefixed with $ sign, for example ${expr} |
| 2. | We can disable EL expression in JSP by setting [JSP page directive](http://www.journaldev.com/2044/jsp-directives-page-include-and-taglib-example) isELIgnored attribute value to TRUE. |
| 3. | JSP EL can be used to get attributes, header, cookies, init params etc, but we can’t set the values. |
| 4. | JSP EL implicit objects are different from JSP implicit objects except pageContext, don’t get confused. |
| 5. | JSP EL pageContext implicit object is provided to get additional properties from request, response etc, for example getting HTTP request method. |
| 6. | JSP EL is NULL friendly, if given attribute is not found or expression returns null, it doesn’t throw any exception. For arithmetic operations, EL treats null as 0 and for logical operations, EL treats null as false. |
| 7. | The [] operator is more powerful than dot operator because we can access list and array data too, it can be nested and argument to [] is evaluated when it’s not string literal. |
| 8. | If you are using Tomcat, the EL expressions are evaluated using org.apache.jasper.runtime.PageContextImpl.proprietaryEvaluate() method. |
|  |  |