EL

EL 概述

什么是EL

JSP2.0要把html和css分离、要把html和javascript分离、要把Java代码块替换成标签。标签的好处是非Java人员都可以使用。

JSP2.0 – 纯标签页面，即：不包含<% ... %>、<%! ... %>，以及<%= ... %>

EL（Expression Language）是一门表达式语言，它对应<%=...%>。我们知道在JSP中，表达式会被输出，所以EL表达式也会被输出。

EL表达式的格式：${...}，例如：${1+2}。

在使用EL表达式时，要求page指令的isELIgnored属性为false。这样JSP在编译成.java时，才不会忽略EL表达式。

如果你希望某个EL表达式被JSP编译器忽略，那么可以在EL表达式之前添加"\"，例如：\${1+2}。

EL隐藏对象

在EL表达式中，无需创建就可以使用的对象称之为EL隐藏（隐含、内置）对象。在EL中一共有11个隐藏对象，它们都与Map相似。其中10 个 是Map，一个是PageContext

EL操作操作对象的方式

- 操作List和数组：${list[0]}、${arr[0]}；
- 操作bean的属性：${person.name}、${person['name']}，对应person.getName()方法；
- 操作Map的值：${map.key}、${map['key']}，对应map.get(key)。

注意，在使用EL获取参数时，如果参数不存在，返回的是空字符串，而不是null。这一点与使用request.getParameter()方法是不同的。

EL 运算符

```
+       加
-       减
*       乘
/或div  除
%或mod  取余
==或eq  等于
!=或ne  不等于
<或lt   小于
>或gt   大于
<=或le  小于等于
>=或ge  大于等于
&&或and 并且
!或not  非
||或or  或者
empty   是否为空  ${empty ""}，可以判断字符串、数组、集合的长度是否为0，为0返回true。empty还可以与not或!一起使用。${not empty ""}
```

EL隐藏对象

参数隐藏对象

这些隐藏对象都是Map类型！

1. param：param是Map<String,String>类型！param对象可以用来获取参数，与request.getParameter()方法相同. 例: ${param.password}
2. paramValues：paramValues是Map<String, String[]>类型，当一个参数名，对应多个参数值时可以使用它。例:checkbox 选中的值
3. header: Map<String,String> ${header.referer}
4. headerValues：headerValues是Map<String,String[]>类型。当一个请求头名称，对应多个值时，使用该对象，这里就不在赘述。
5. initParam：initParam是Map<String,String>类型。它对应web.xml文件中的 "context-param"参数。
6. cookie：cookie是Map<String,Cookie>类型，其中key是Cookie的名字，而值是Cookie对象本身。 例: Cookie对象 ("username","ethan") ("password","123") ${cookie.username} 获取的是 键为"username" 的 cookie 对象，cookie.password 获取的是键为"password"的cookie对象

   ${cookie.username.value} "ethan" ${cookie.username.username} "username"

域对象

1. pageScope
2. requestScope
3. sessionScope
4. applicationScope

> 都是Map<String,Object>类型，${pageScope.xxx}的功能相等与pageContext.getAttribute("xxx")。两者的区别在于，前者在数据不存在时返回空字符串，而后者返回null。

页面上下文隐藏对象

pageContext ： pageContext是PageContext类型！可以使用pageContext对象调用getXXX()方法，例如pageContext.getRequest()，可以${pageContext.request}。

```
`${ pageContext.request.contextPath }` 项目目录
`${pageContext.request.requestURL}`

`${pageContext.session.new}` pageContext.getSession().isNew();
`${pageContext.session.id}` pageContext.getSession().getId();
`${pageContext.servletContext.serverInfo}`
```

EL 函数库

什么EL函数库

EL函数库是由第三方对EL的扩展，我们现在学习的EL函数库是由JSTL添加的。下面我们会学习JSTL标签库。 EL函数库就是定义一些有返回值的静态方法。然后通过EL语言来调用它们！当然，不只是JSTL可以定义EL函数库，我们也可以自定义EL函数库。　　EL函数库中包含了很多对字符串的操作方法，以及对集合对象的操作。例如：${fn:length("abc")}会输出3，即字符串的长度。

使用

1. 导入函数库　　因为是第三方的东西，所以需要导入。导入需要使用taglib指令！ <%@ taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions" %>
2. ${fn: 函数名(参数)}

${fn:length(arr) } ${fn:contains("abc", "a")}

EL函数库具体函数

- toUpperCase(String input)：String
- toLowerCase(String input)：String
- indexOf(String input, String substring)：int
- contains(String input, String substring)：boolean
- containsIgnoreCase(String input, String substring)：boolean
- startsWith(String input, String substring)：boolean
- endsWith(String input, String substring)：boolean
- substring(String input, int beginIndex, int endIndex)：String
- substringAfter(String input, String substring)：String
- substringBefore(String input, String substring)：String
- escapeXml(String input)：String
- trim(String input)：String
- replace(String input, String substringBefore, String substringAfter)：String
- split(String input, String delimiters)：String[]
- length(Object obj)：int
- join(String array[], String separator)：String

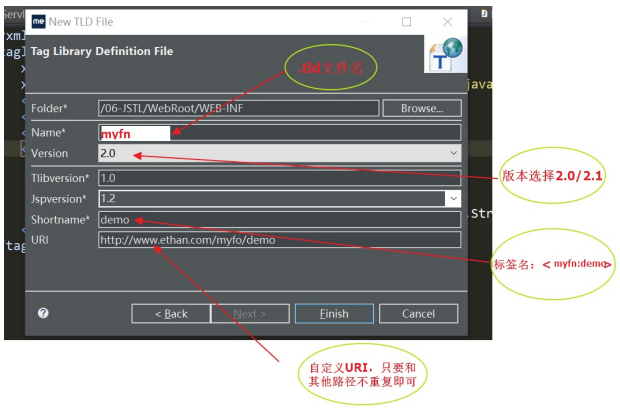
自定义函数库

1. 编写一个类，方法必须是静态方法。 例:

```
public class ELDemo {
        public static String sayHello(String name){
                return "Hello!"+name;
        }

}
```

2. 在WEB-INF目录下创建tld的文件，配置。(版本选择2.0 或以上)



.tld文件 配置

```
        <!-- 配置自定义的EL函数 -->
        <function>
        <!-- 配置方法名称 -->
        <name>sayHello</name>

        <!-- 方法所在的类 -->
        <function-class>com.example.ELDemo</function-class>

        <!-- 配置方法的签名 -->
        <function-signature>java.lang.String sayHello(java.lang.String)</function-signature>
                                                返回值类型          方法名    参数类型
        </function>
```

3. 使用

- 引入 uri <%@ taglib prefix="myfn" uri="http://www.ethan.com/myfn/demo" %>
- ${myfn:sayHello("ethan") 返回结果: Hello!ethan