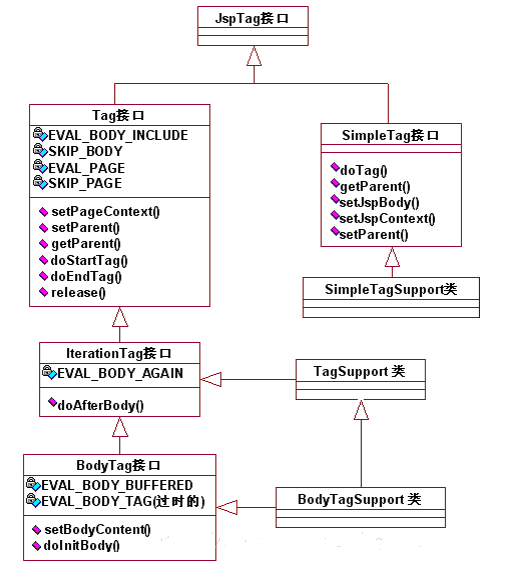
**前面我们在jsp页面中,我们对集合或数组的显示,都是通过java的循环遍历来做的!**

**这个样子做,太麻烦,而且不利于前端美工和后端java程序员的分离,程序修改和维护起来都非常麻烦,我们希望的,java代码和前端页面完全分离!这个时候,EL标签就不够用了,比如循环遍历,if分支等等EL是做不到的!**

**首先我们想到的,能不能将循环,if分支也做成一个像EL的标签:**

**JDK为我们提供了自定义标签的接口**



**这是Java中标签规范的继承体系，实现Tag接口的我们叫做传统式标签库开发，这种开发模式略显发复杂，基本已经被SimpleTag式的简单式开发标签库给取代了。Java中提供了一个默认的实现类SimpleTagSupport来实现自定义标签，我们只要继承此类即可。**

**接下来我们就看自定义标签的开发步骤:**

**①编写标签功能的java类(标签处理器)**

**②编写标签库描述文件(.tld)**

**③中jsp页面中导入和使用自定义的标签**

**根据步骤,我们先做一个简单的标签出来:**

**首先我们先从自定义标签处理类开始，正如上文所说，这个类只有继承了SimpleTagSupport这个类可以省去省去重写SimpleTag接口中的一些方法。我们说个doTag()这个方法很重要，这个方法类似于java中的main方法一样，当jsp页面加载到我们定义的标签的时候就会过来调用这个方法。**

**public class MyTag extends SimpleTagSupport {**

**@Override**

**public void doTag() throws JspException, IOException {**

**getJspContext().getOut().write("hello tag");**

**}**

**}**

**这是一个简单的标签处理类，具体的细节暂时不用关心，只需要知道，它负责向jsp页面输出字符串即可。下面我们看看第二步，创建\*tld文件。这个文件我们没有必要重新写一遍，到Tomcat服务器上的webapps/examples/WEB-INF/jsp2中复制一个过来，修改名字存放到我们的项目中WEB-INF的任意子路径下。删除一些标签成如下内容：**



**我们看到这是一个XML文件，根元素为taglib，而taglib主要有以下几个子元素：**

**description //描述信息**

**tlib-version //指定标签库的版本号，基本不用我们操心**

**short-name //指定标签库的短名字，也是没什么用**

**uri //这是一个重要的子元素，是该标签库的唯一标识**

**tag //看名字就知道，这是定义标签的子元素，很重要**

**对于taglib这个根元素，我们主要关心他下面的uri和tag两个子元素，一个标签库可以由多个标签，也就是可以有多个tag标签。关于tag标签，主要有以下几个子元素：**

**name //该标签的唯一标识，很重要**

**tag-class //指定了处理该标签的类，也就是使用该标签谁给我返回结果**

**body-content //标签体，后面详说，很重要**

**还可以有attribute //属性，后面介绍，很重要**

**对于以上标签大家可能已经知道什么意思，但是具体用在什么地方可能不清楚,先不急,我们先来看看这么使用自定义的标签库:**

**使用标签库也是有两个步骤，首先导入标签库，然后引用标签。我们使用taglib编译指令导入标签库，具体格式如下：**

**<%@ taglib uri="tld文件中指定的唯一标识" prefix="指定标签前缀"%>**

**我们看到这个导入标签库的编译指令主要有两个属性，一个是用于定位我们已经写好的标签库，定位的方法就是读取每个tld文件中的URI元素的值，prefix用于指定我们使用标签时的前缀，等用的时候就很容易理解了，现在解释反而不容易说清楚。我们使用标签的格式如下：<刚刚指定的前缀 :标签名 />标签名就是我们标签库中每个tag都会有的name的值.**

**以上就是一个最简单的自定义标签的过程，为了更好的理解后续的较复杂的自定义标签方式，上述内容值得感受体会。**