**自定义标签**

# 授课内容

授课科目：JavaEE

授课内容：自定义标签

授课类型：讲授+实践

授课时间：3+1学时

主讲教师：武永亮

# 教学目标要求

## 思想目标：

## 能力目标：

1. 了解定制标签的概念；
2. 掌握简单标签的创建方法；
3. 掌握标准标签的创建方法。

## 知识目标：

1. 定制标签简介；
2. 简单标签简介及使用；
3. 标准标签简介及使用；
4. JSP中使用标签文件。

# 课件分析

概 述：在之前的课程中讲到，JSP表示层中需要编写大量的JSP脚本代码，其他JSP中实现相同的代码，不能做到代码复用。JSP开发者可以通过自定义标签来根据实际需求实现方法，使用自定义标签可以实现页面表示层的代码重用。本章使用简单标签、标准标签、JSP标签文件三种方式来实现自定义标签效果。

教学重点：简单标签的实现；标准标签的实现。

教学难点：JSP标签文件的实现。

# 教学方法

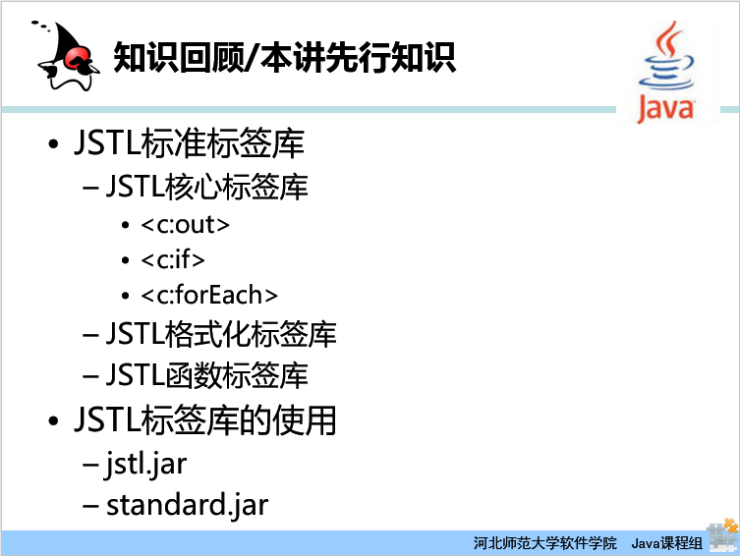
直观演示法、案例法等。

# 教学过程

## 课前补充

## 上节回顾

（2分钟）



上节课讲到JSTL标准标签库的基本用法，JSTL标签库主要包括：核心标签库、格式化标签库、函数标签库、SQL标签库、XML标签库。上节课主要内容讲解核心标签库的用法，其中常用标签有<c:out>、

<c:if>、<c:forEach>标签，在使用JSTL标签库前一定要导入jstl.jar、standard.jar文件。

## 作业点评

（0分钟）

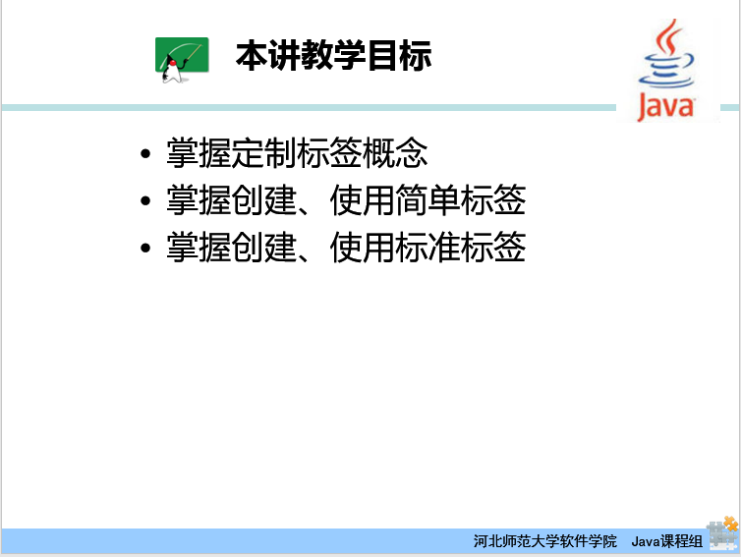
## 导入新课

（2分钟）

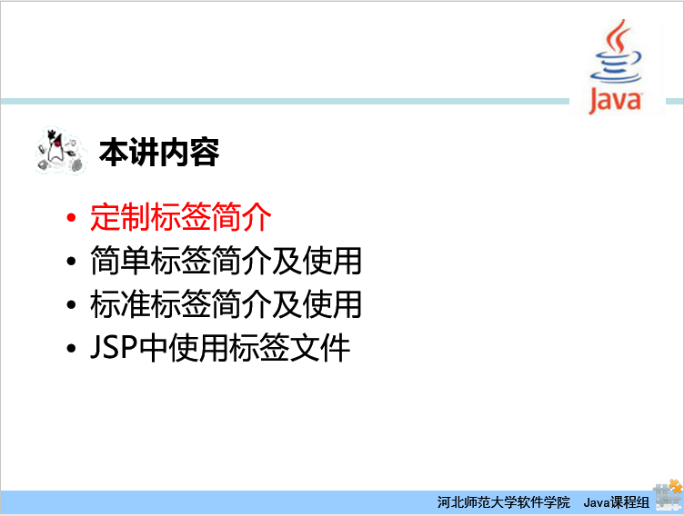
JSTL标准标签库封装了JSP前端需要的基本标签，这些标签是在不同项目中都共同需要的。但在不同的项目中可能有一些项目中通用的标签，这些标签是根据具体项目的需求来定义，解决项目中的实际需求。在JSP中可以通过自定义标签来实现项目中不同需求的JSP页面的封装。本章讲解自定义标签的用法。

## 讲授新课

（120分钟）

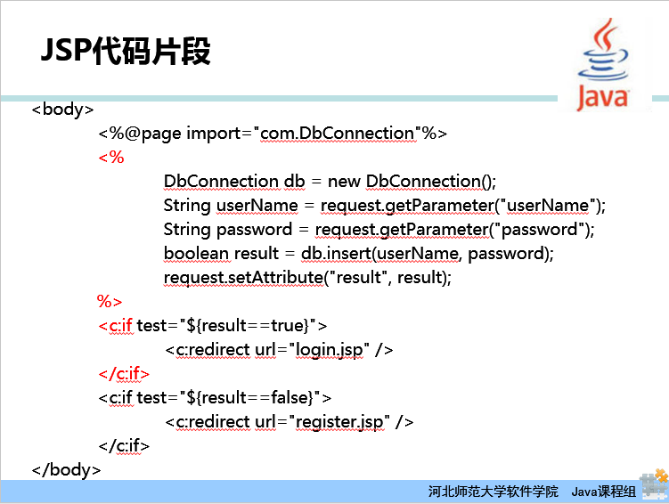


本章的教学目标：要求学生了解定制标签的基本概念，要求学生掌握简单标签的基本用法，要求学生掌握标准标签的基本用法。

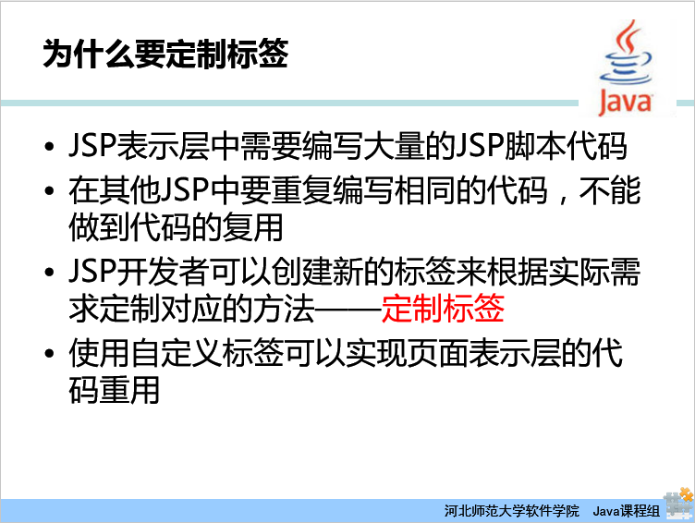


本节内容分为四个部分，第一部分介绍定制标签，也就是自定义标签的基本概念，第二部分介绍简单标签的基本概念和使用方法，第三部分介绍标准标签的基本概念和使用方法，第四部分介绍JSP标签文件的使用方法。

首先介绍第一部分内容—自定义标签的基本概念。

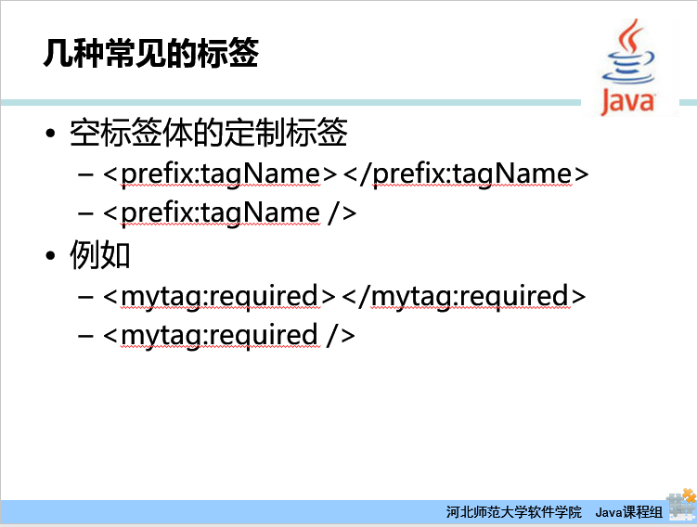


首先看一段JSP代码片段，在这段JSP代码中包含了Java标签、HTML标签，例如：<% %>、<body> </body>，通过JSTL标准标签库，可以实现在JSP页面中进行基本的逻辑判断，但是这些页面中的显示逻辑，无法进行复用。

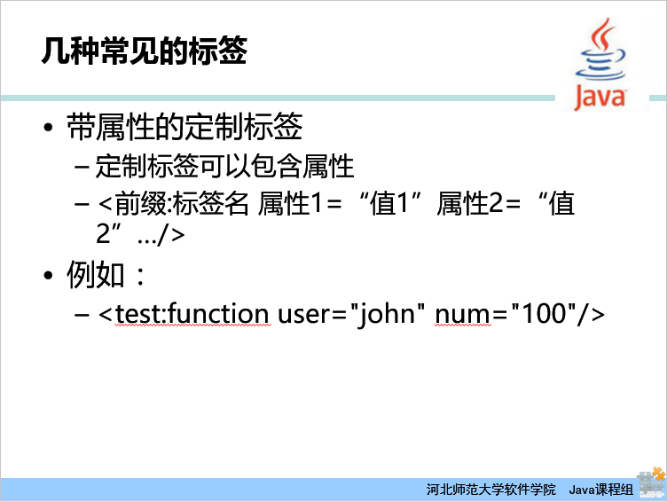


JSP表示层中包含大量的JSP脚本代码，但是在JSP中要重复编写相同的代码，而不能做到代码的复用。JSP开发者可以创建新的标签来根据实际需求定制对应的方法——定制标签，这种方式可以实现页面表示层的代码重用。

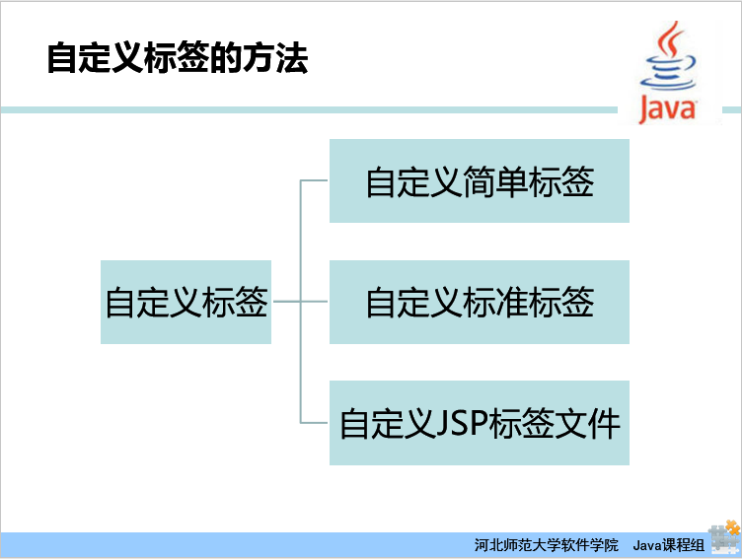
常见的几种标签样式如下：



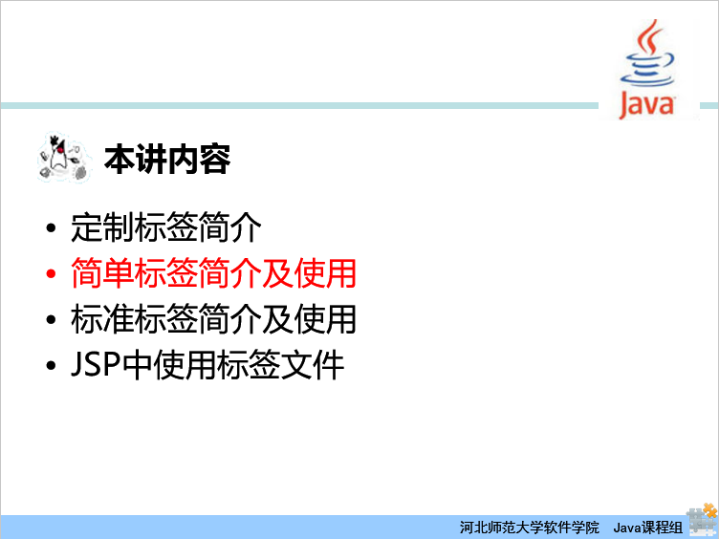
空标签体的定制标签，基本结构为<prefix:tagName></prefix:tagName>或者<prefix:tagName /> ，其中prefix为JSP文件头声明的标签前缀，tagName为标签名。具体示例代码参考requiredTag.jsp文件。其中<mytag:required></mytag:required>，mytag为定义好的标签前缀，required为标签名。



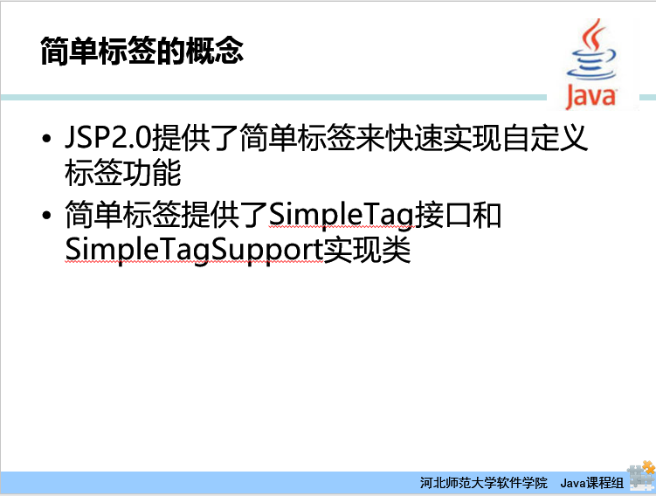
带属性的定制标签，顾名思义就是在标签中加入附加属性。标签结构为<前缀:标签名 属性1=“值1”属性2=“值2”…/>，具体示例代码参考sampleLib.jsp文件。其中<test:function user="john" num="100"/>，test为定义好的标签前缀，function为标签名，user为属性名，john为属性值。



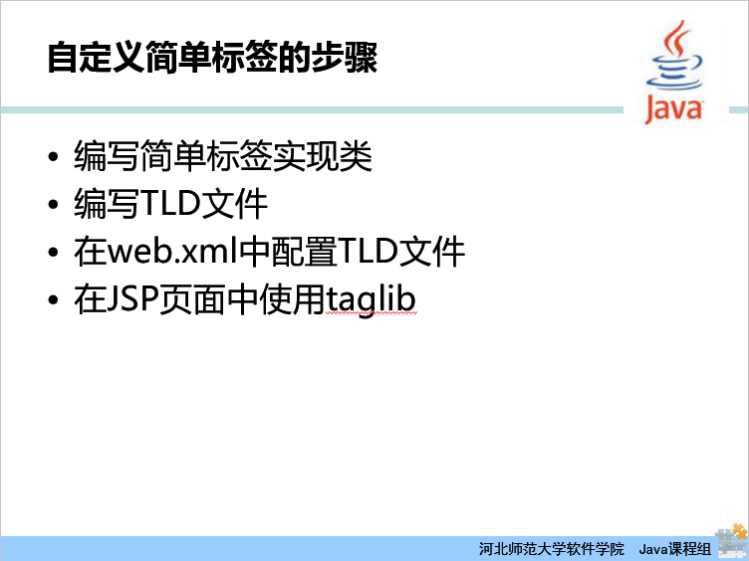
JSP中实现自定义标签的方法有三种：自定义简单标签，自定义标准标签，自定义JSP标签文件。



接下来介绍简单标签的使用。



在JSP2.0的标准中提供了简单标签，用来快速实现自定义标签的功能，其实现方法是自定义类实现SimpleTag接口或继承自SimpleTagSupport实现类。



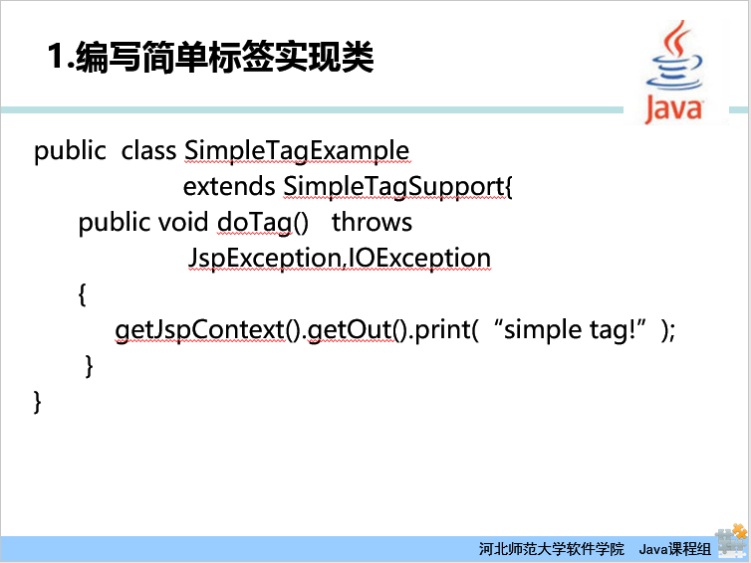
在JSP中自定义简单标签的步骤：

1.编写简单标签实现类；

2.编写TLD文件；

3.在web.xml中配置TLD文件；

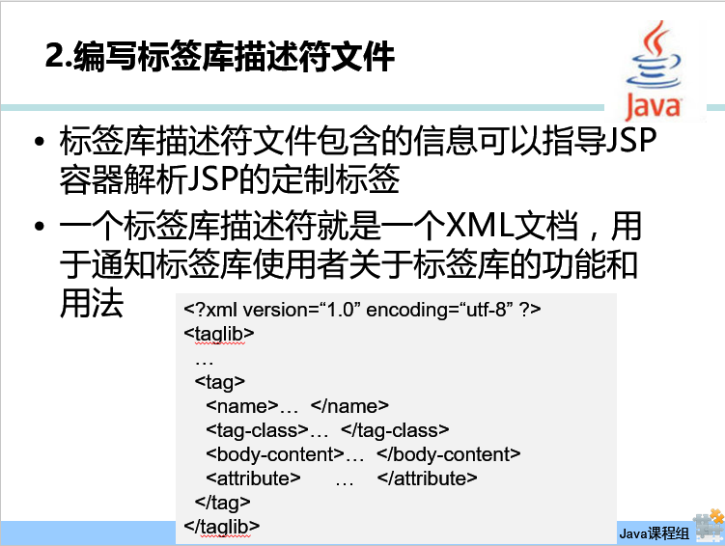
4.在JSP页面中使用taglib。



示例代码参考Demo04-03中的mathtag01.jsp文件。代码如下：

|  |
| --- |
| public class SimpleTagExample  extends SimpleTagSupport{  public void doTag() throws  JspException,IOException  {  getJspContext().getOut().print(“simple tag!”);  }  } |

在本段代码中定义了一个SimpleTagExample类，实现了SimpleTagSupport类，其中要求重写doTag方法，在此方法中通过getJspContext()方法得到调用标签的JSP上下文对象，然后通过getOut()方法得到Out对象，然后调用print()方法打印输出语句。

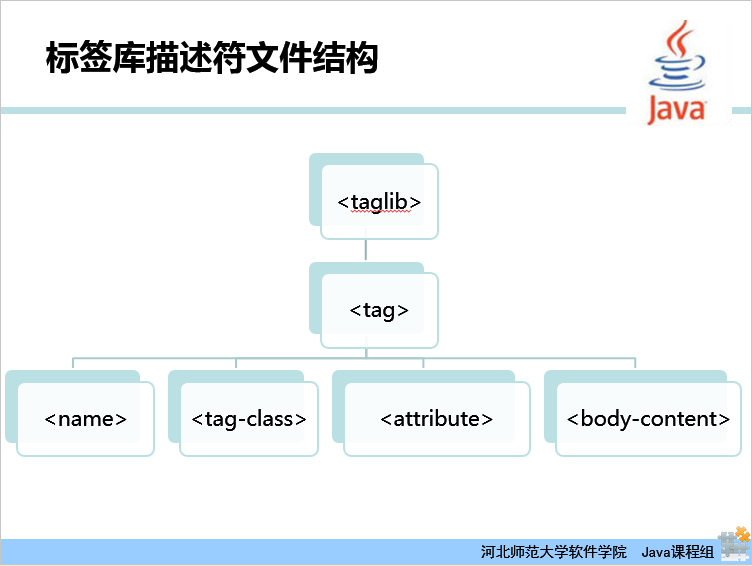


标签库描述符文件包含的信息可以指导JSP容器解析JSP的定制标签，一个标签库描述符就是一个XML文档，用于通知标签库使用者关于标签库的功能和用法

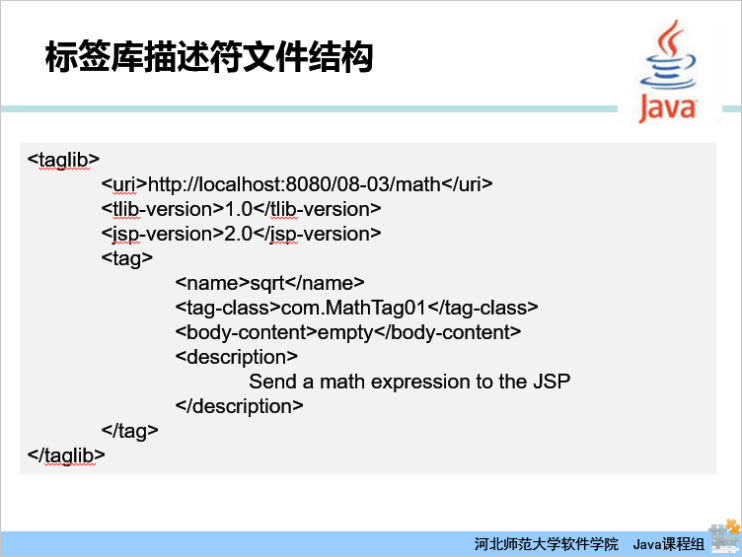
|  |
| --- |
| <?xml version=“1.0” encoding=“utf-8” ?>  <taglib>  …  <tag>  <name>… </name>  <tag-class>… </tag-class>  <body-content>… </body-content>  <attribute> … </attribute>  </tag>  </taglib> |

在此标签库描述符文件中，通过tag声明了自定义标签的信息，name结点表示自定义tag的名字，tag-class标签表示tag对应的Java类，body-content标签说明tag之间值的类型，attribute标签表示在tag标签调用时传入的参数。

标签库描述符的文件结构如下：



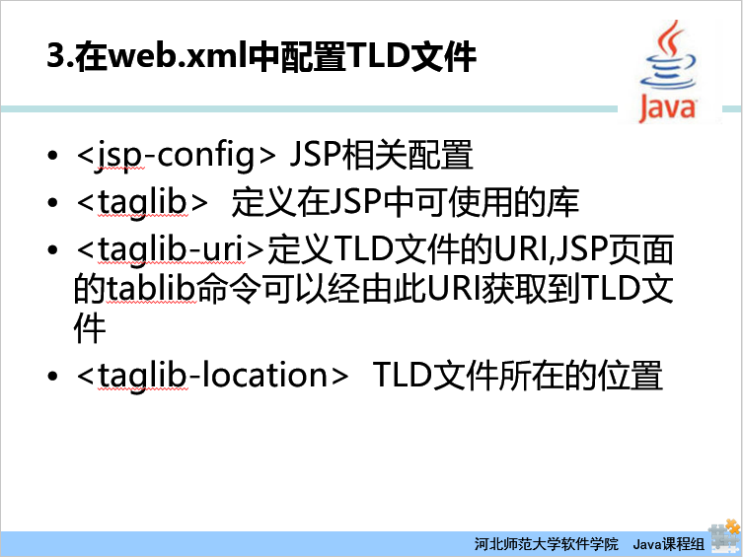
在标签库描述符编写过程中，一定要注意描述符的层次结构，否则可能会造成编译错误。



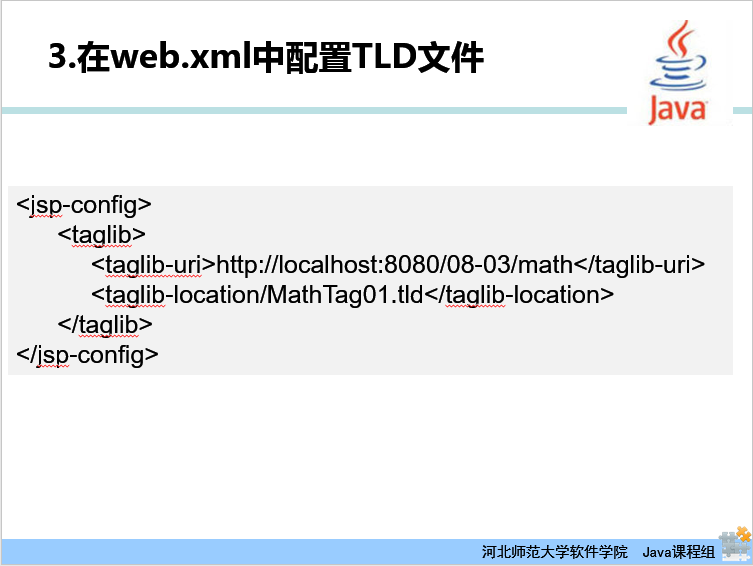
这是一段已经配置好的部署描述符文件代码：

|  |
| --- |
| <taglib>  <uri>http://localhost:8080/08-03/math</uri>  <tlib-version>1.0</tlib-version>  <jsp-version>2.0</jsp-version>  <tag>  <name>sqrt</name>  <tag-class>com.MathTag01</tag-class>  <body-content>empty</body-content>  <description>  Send a math expression to the JSP  </description>  </tag>  </taglib> |

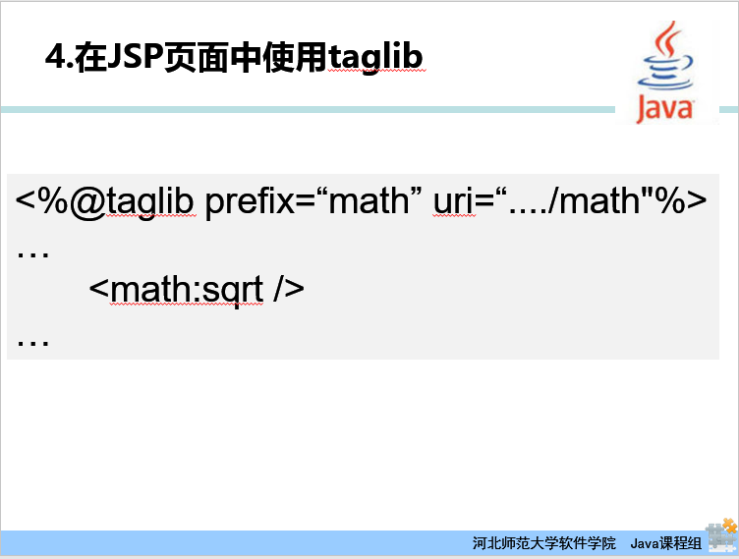
其标签库的uri定义为<http://localhost:8080/08-03/math>，定义了一个标签，名字为sqrt，对应的处理类为com.MathTag01类，body-content为empty，代表标签内部没有值。



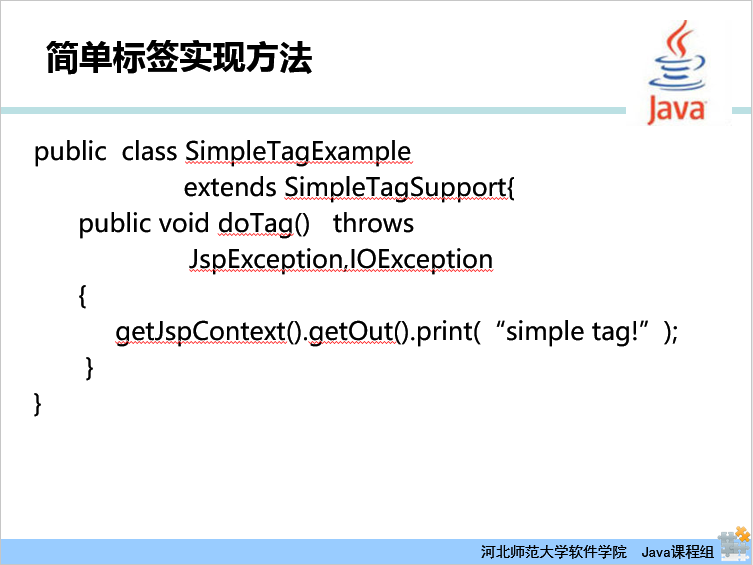
第三步在web.xml中配置TLD文件，其中<jsp-config> 配置JSP相关信息，<taglib> 定义在JSP中可使用的库，<taglib-uri>定义TLD文件的URI,JSP页面的tablib命令可以经由此URI获取到TLD文件，<taglib-location> TLD文件所在的位置。



这是一个web.xml的配置文件示例，其中声明了一个taglib-uri为http://localhost:8080/08-03/math 的标签库，对应的tld文件为/MathTag01.tld。



第四步 在JSP页面中使用taglib。其通过<%@taglib prefix=“math” uri=“..../math"%>在JSP页面中声明/math标签库，前缀定义为math。然后在当前JSP页面中就可以通过<math:sqrt />方式调用math库中的sqrt标签。



这是刚才实现的SimpleTagExample类的代码，这里只重写了doTag方法，其实SimpleTagSupport还有其他方法也可以被重写。



简单标签有如下接口：

1.setJspContext方法，用于把JSP页面的pageContext对象传递给标签处理器对象

2．setParent方法，用于把父标签处理器对象传递给当前标签处理器对象

3．getParent方法，用于获得当前标签的父标签处理器对象

4．setJspBody方法，用于把代表标签体的JspFragment对象传递给标签处理器对象

5．doTag方法，用于完成所有的标签逻辑，包括输出、迭代、修改标签体内容等。在doTag方法中可以抛出javax.servlet.jsp.SkipPageException异常，用于通知WEB容器不再执行JSP页面中位于结束标记后面的内容，这等效于在传统标签的doEndTag方法中返回Tag.SKIP\_PAGE常量的情况。

SimpleTag生命周期是：初始化，setJspContext()，处理标签内容体,setBody()，调用doTag()方法。

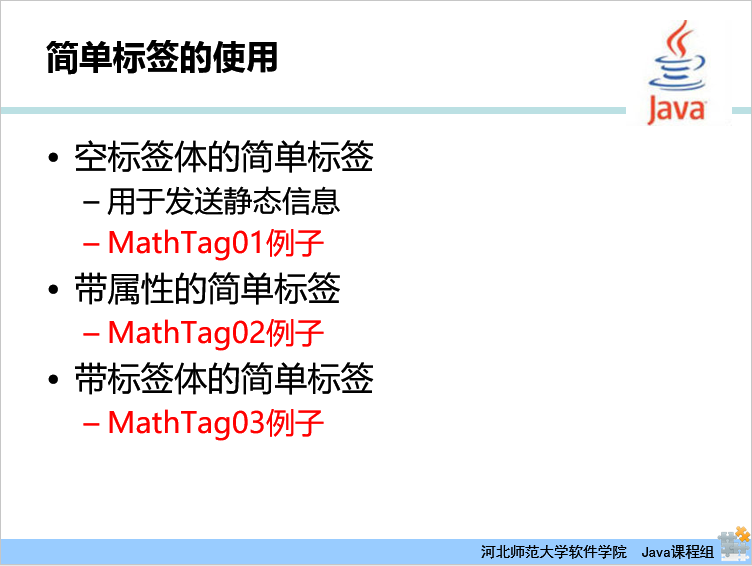


简单标签类有如下接口方法：

1. getJspContext()获取JspContext对象

2. getJspBody()获取JspFragment对象

3. findAncestorWithClass()获取指定类的祖先标签

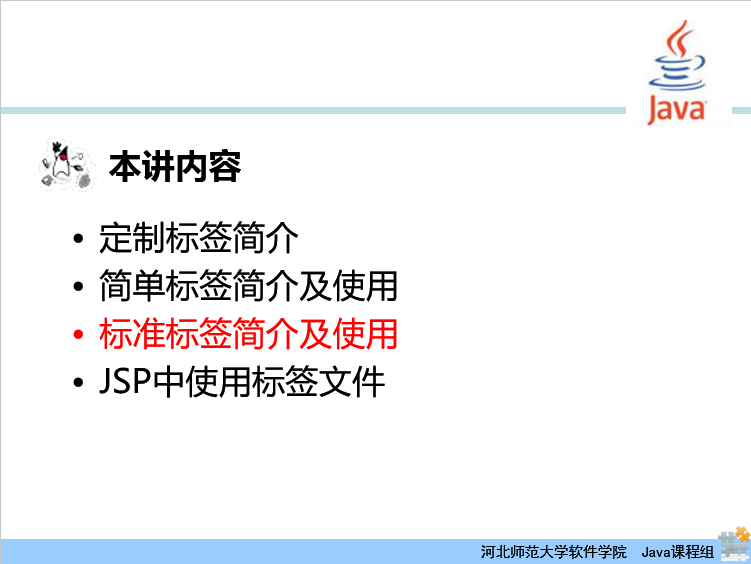


简单标签的实现示例如下：

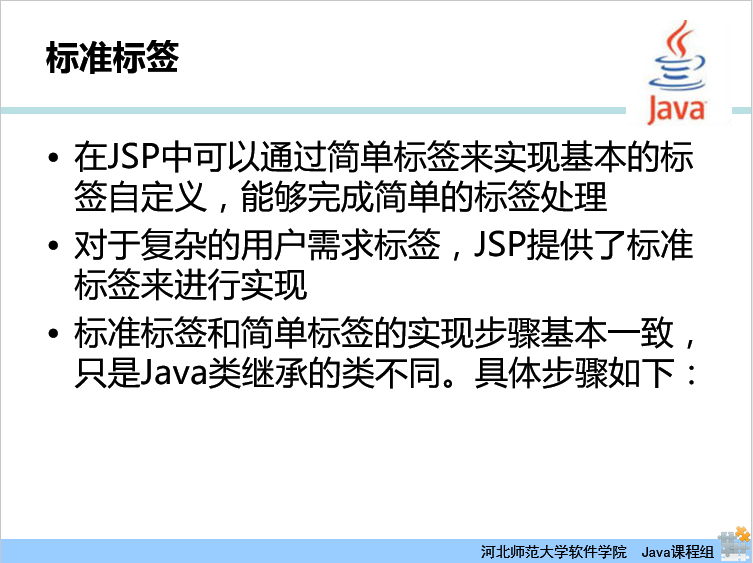
空标签体的简单标签：用于发送静态信息，MathTag01.jsp

带属性的简单标签:MathTag02.jsp

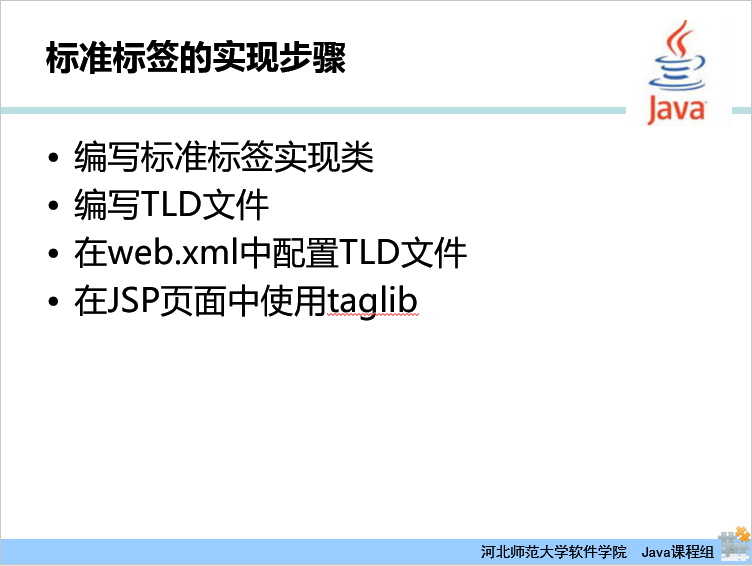
带标签体的简单标签,MathTag03.jsp



接下来讲解JSP中的标准标签用法。



在JSP中可以通过简单标签来实现基本的标签自定义，能够完成简单的标签处理，对于复杂的用户需求，JSP提供了标准标签来进行实现，标准标签和简单标签的实现步骤基本一致，只是Java类继承的类不同。具体步骤如下：



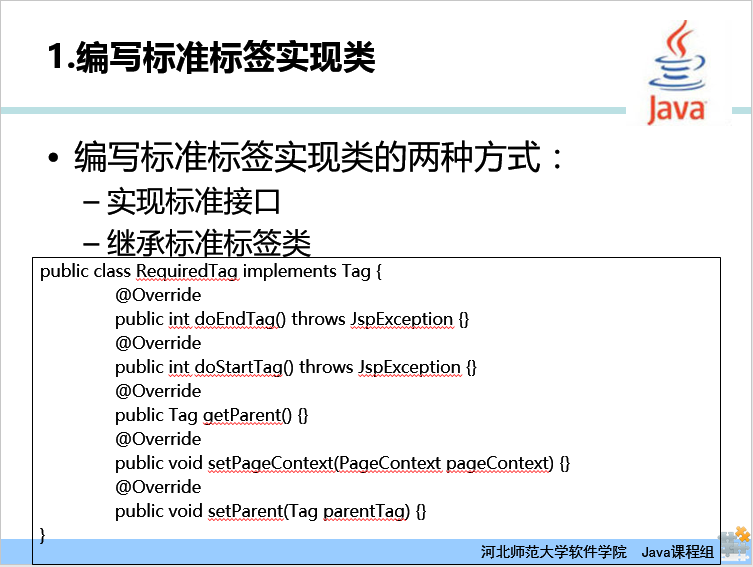
在JSP中自定义标准标签的步骤：

1.编写标准标签实现类

2.编写TLD文件

3.在web.xml中配置TLD文件

4.在JSP页面中使用taglib



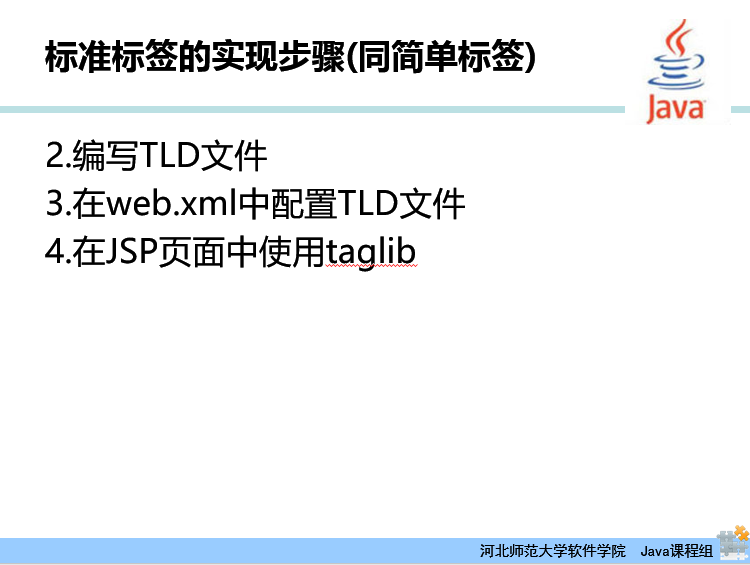
标准标签实现类的两种方式：

实现标准接口：public class RequiredTag implements Tag { }

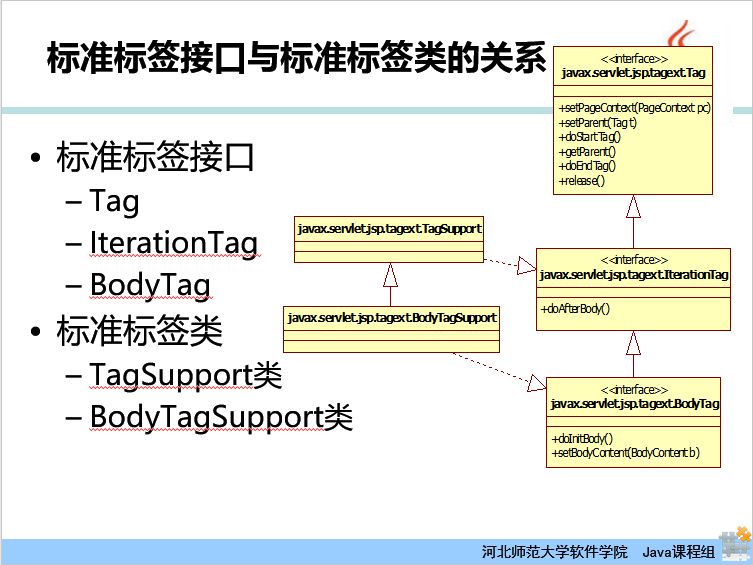
public class MarkTag implements BodyTag { }

public class LoopTag implements IterationTag { }

继承标准标签类：



标准标签的其他实现步骤与简单标签相同。



标准标签接口与标准标签类的关系如图，标准标签接口Tag，IterationTag，BodyTag，依次继承。标准标签类：TagSupport类，BodyTagSupport类依次继承。其中TagSupport类实现了IterationTag接口，BodyTagSupport类实现了BodyTag接口。



Tag接口的方法如下：

int doStartTag()当标签被处理时调用；

int doEndTag()当标签结束处理时调用；

Tag getParent()获取此标签的直接父标签类；

void release()释放标签资源；

void setPageContext(pageContext) 设置当前上下文；

void setParent(Tag)设置直接父标签类。



Tag接口常量属性如下：

EVAL\_BODY\_INCLUDE 指明标签内容体需要处理；

SKIP\_BODY 指明标签内容体不需要处理；

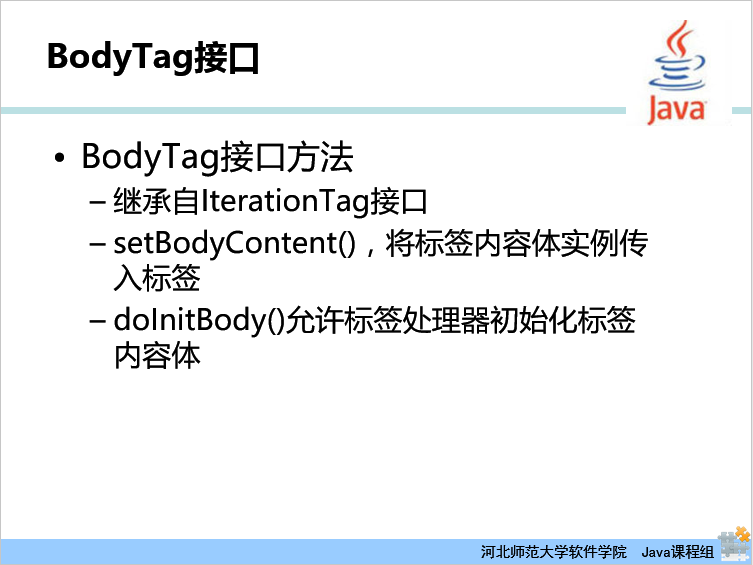
EVAL\_PAGE 指明继续处理页面剩余部分；

SKIP\_PAGE 指明不继续处理页面剩余部分。



IterationTag接口方法，继承了Tag接口所有的方法，doAfterBody()执行完标签体内容调用此方法。

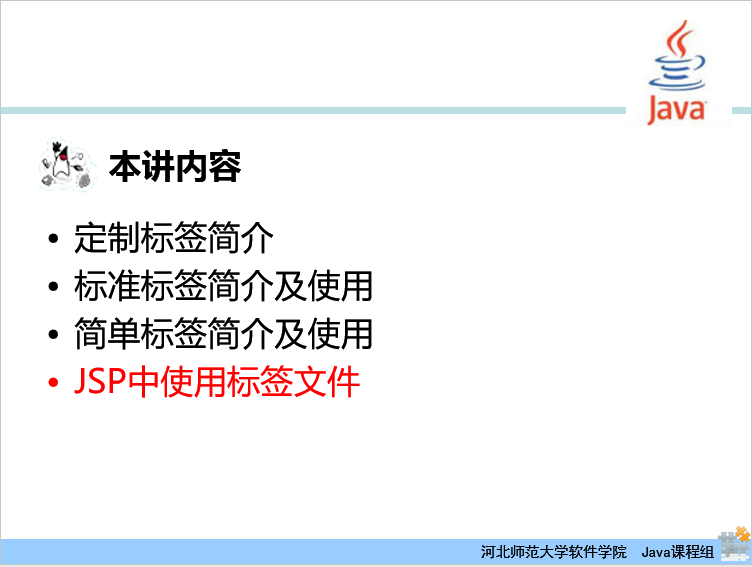
IterationTag接口属性，EVAL\_BODY\_AGAIN 指明标签内容体循环处理。



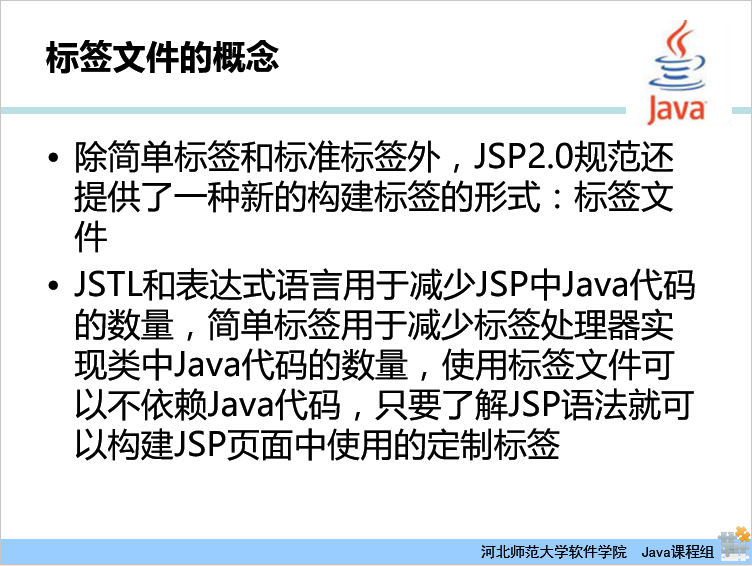
BodyTag接口方法，继承自IterationTag接口，setBodyContent()，将标签内容体实例传入标签，doInitBody()允许标签处理器初始化标签内容体。



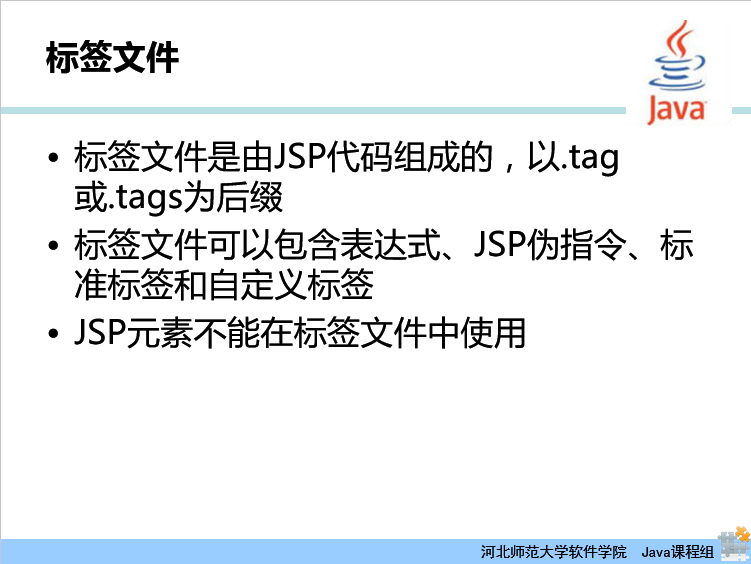
TagSupport类，实现了IterationTag接口，提供Tag和IterationTag接口所有方法的默认实现BodyTagSupport类，继承自TagSupport类、实现了BodyTag接口，提供setBodyContent()和doInitBody()方法的默认实现。 新增getBodyContent()和getPreviousOut()处理缓冲区输出。



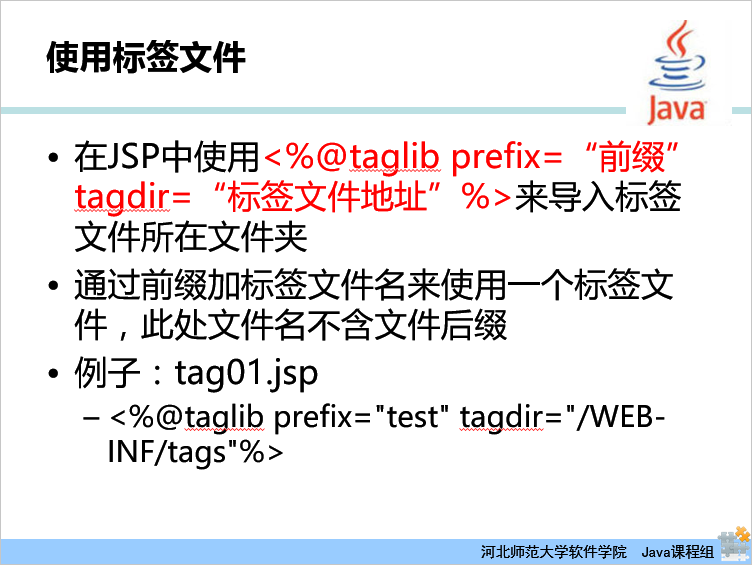
接下讲解JSP中使用标签文件自定义标签。



除简单标签和标准标签外，JSP2.0规范还提供了一种新的构建标签的形式：标签文件。JSTL和表达式语言用于减少JSP中Java代码的数量，简单标签用于减少标签处理器实现类中Java代码的数量，使用标签文件可以不依赖Java代码，只要了解JSP语法就可以构建JSP页面中使用的定制标签。

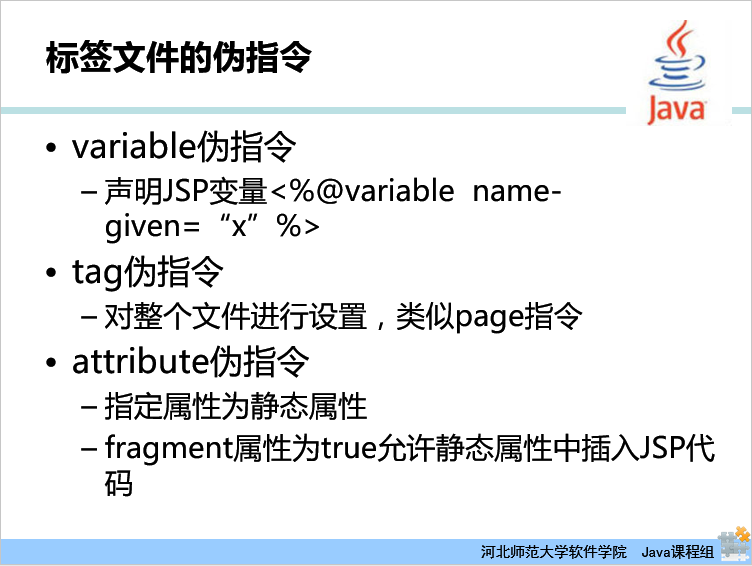


标签文件是由JSP代码组成的，以.tag或.tags为后缀，标签文件可以包含表达式、JSP伪指令、标准标签和自定义标签，JSP元素不能在标签文件中使用。



在JSP中使用<%@taglib prefix=“前缀” tagdir=“标签文件地址”%>来导入标签文件所在文件夹，通过前缀加标签文件名来使用一个标签文件，此处文件名不含文件后缀。

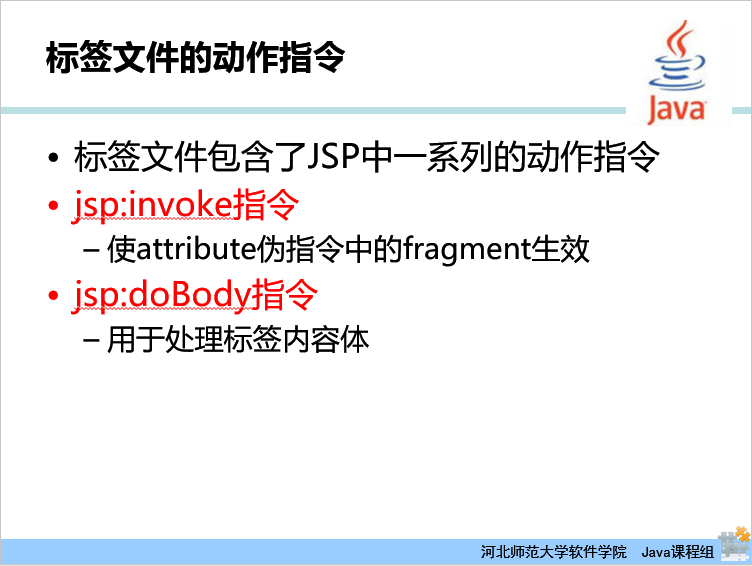
例子：参考tag01.jsp文件，其中通过该<%@taglib prefix="test" tagdir="/WEB-INF/tags"%>来引入JSP标签文件。



variable伪指令，声明JSP变量<%@variable name-given=“x”%>

tag伪指令，对整个文件进行设置，类似page指令

attribute伪指令，指定属性为静态属性，fragment属性为true允许静态属性中插入JSP代码。



标签文件包含了JSP中一系列的动作指令：

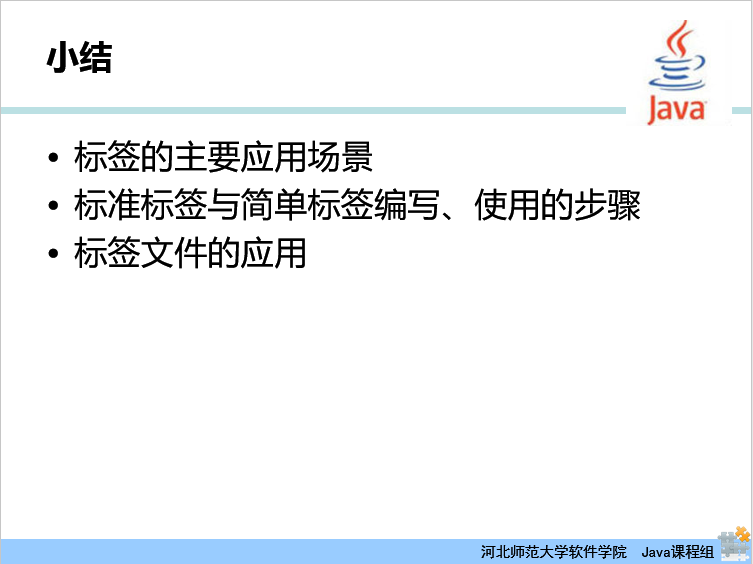
jsp:invoke指令，使attribute伪指令中的fragment生效

jsp:doBody指令，用于处理标签内容体

自定义标签部分的内容讲解完毕，接下来进行总结。

## 课时小结

（2分钟）



首先介绍了自定义标签的应用场景，然后重点讲解使用简单标签和标准标签进行自定义标签的步骤。最后简单介绍JSP标签文件的基本用法。

# 布置作业

（2分钟）

请各位同学通过实验验证Tag接口的生命周期函数调用方式：

