**部署描述符文件**

# 授课内容

授课科目：JavaEE

授课内容：部署描述符文件

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. 了解部署描述符文件的作用；
2. 掌握根据不同的需求配置部署描述符。

## 知识目标：

1. 部署描述符文件的作用；
2. 常用元素的声明与配置；
3. JavaEE元素；

# 课件分析

概 述：web.xml叫做部署描述符文件。它的作用是在Servlet规范中定义的，是web应用的配置文件，作用就是描述程序运行的配置信息。本节内容主要介绍web.xml文件的标签含义，其中包括：web-app标签、icon标签、display-name标签、description标签、filter标签、listener标签、servlet标签、welcome-file-list标签等。

教学重点：web.xml常见标签的配置方法及配置作用。

教学难点：web.xml常见标签的配置方法及配置作用。

# 教学方法

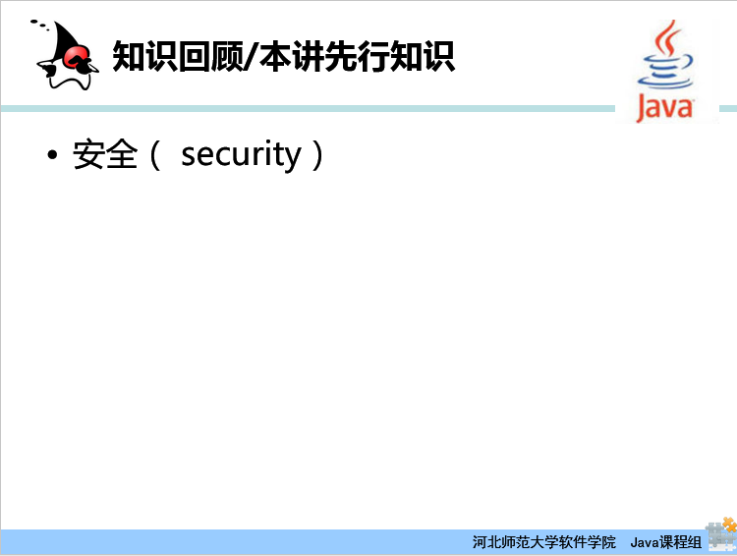
直观演示法、理论教学法等。

# 教学过程

## 课前补充

## 上节回顾

（2分钟）



上节课讲到在web应用中，最重要的一类问题就是安全问题，互联网作为商业交易的工具快速发展，越来越多的公司提供网络交易服务。很多商业活动在网上进行。这就需要web应用要有一定的健壮、安全的互联网保障机制。在上节课中讲解了很多种web应用的安全机制，包括：认证、授权、数据完整性、数据私密性、审核、恶意代码、网站攻击等。其中JSP授权方式就是对资源进行保护的一种方式，通过修改web.xml中的<security-constraint>结点配置信息，保证用户使用web应用的权限。

## 作业点评

（0分钟）

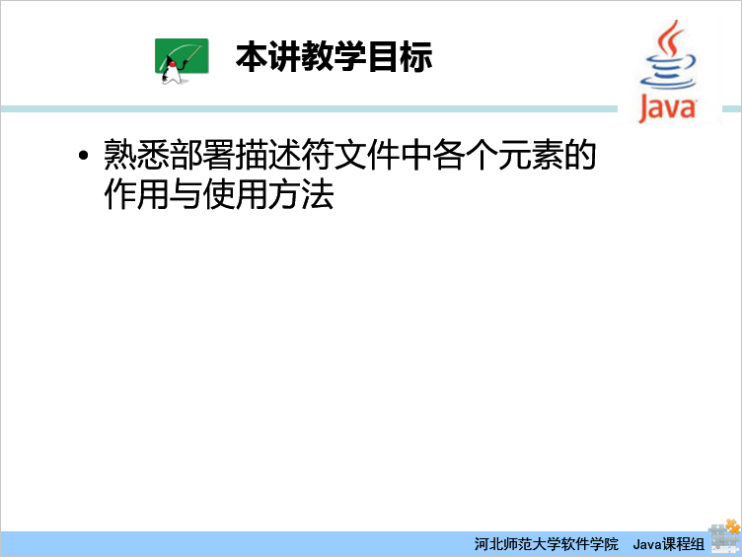
## 导入新课

（5分钟）

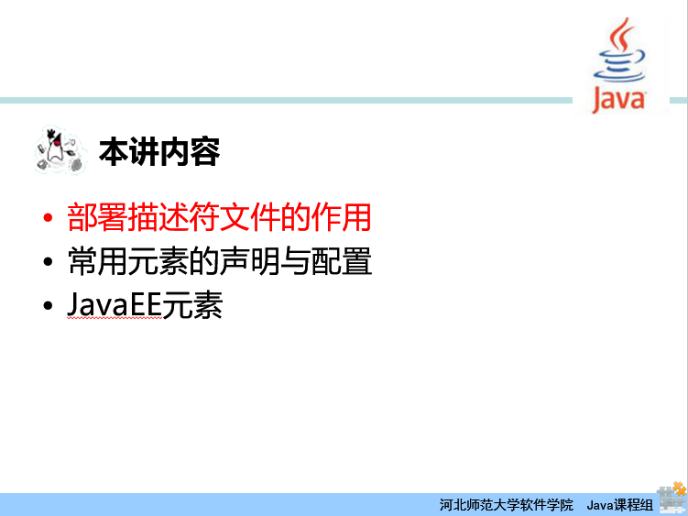
在Web应用配置文件中还可以进行更多的配置来约束Web应用的各项功能，其中有一些配置在之前的课程中已经用到，例如：servlet和servlet-mapping标签，定义了servlet元素，用来声明一个servlet，并且定义servlet与访问URL之间的对应关系。filter和filter-mapping标签，过滤器元素将一个名字与一个实现javax.servlet.Filter接口的类相关联，filter-mapping元素把对应的filter与一个或多个servlet或JSP页面相关联。listener标签定义事件监听程序类，监听某个事件的发生，并对其作出响应。本节课就来详细介绍web.xml的更多用法。

## 讲授新课

（120分钟）

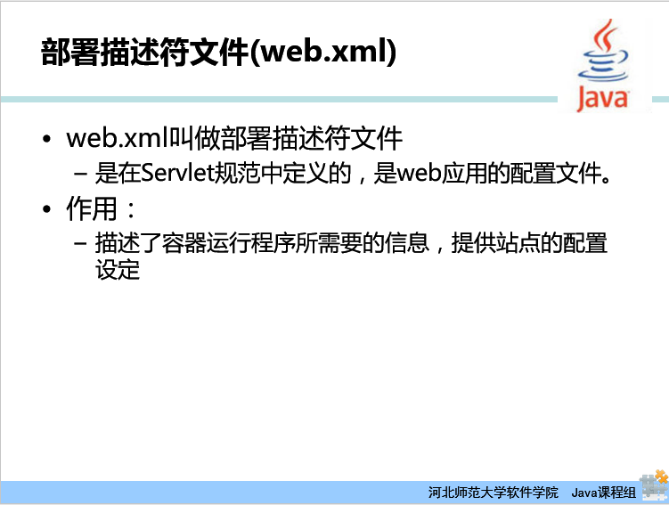


本章的教学目标：要求学生了解部署描述符文件中各元素的作用，熟练掌握部署描述符文件的配置方法。



本节内容分为三个部分，第一部分介绍部署描述符文件的作用，第二部分介绍部署描述符文件中常用元素的声明与配置，第三部分介绍部署描述符相关的JavaEE元素的作用。

首先介绍第一部分内容—部署描述符文件的作用。



web.xml文件，又叫做部署描述符文件，是在Servlet规范中定义的web应用的配置文件，其定义了Web应用的各项参数配置，描述了容器运行程序所需要的信息，提供站点的配置设定。

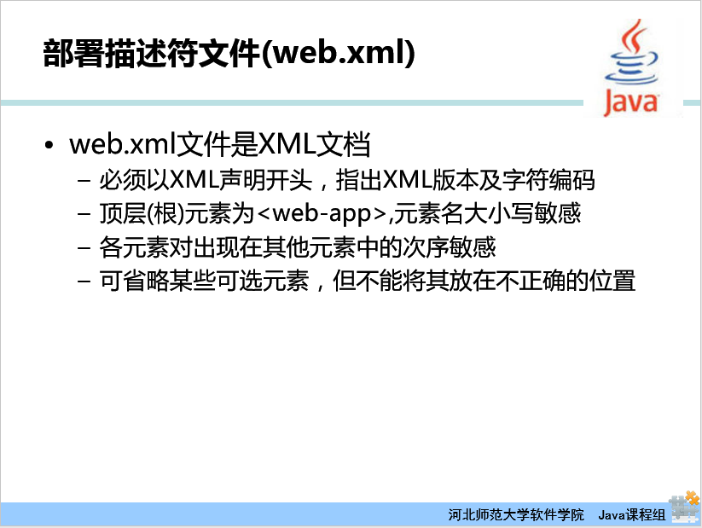
web.xml文件是一个XML文档，所以一定满足XML文档的要求，接下来看一下web.xml文件的构成。



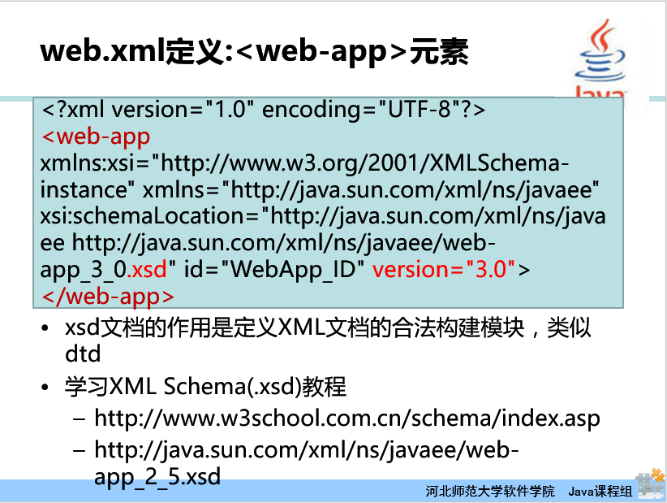
部署描述符文件要求满足XML文件标准，必须以一个XML头开始，代码如下：

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?> |

这个头声明可以定义XML版本并给出文件的字符编码。请注意，XML元素不像HTML，他们是大小写敏感的。因此，web-App和WEB-APP都是不合法的，web-app必须用小写。



web.xml中XML 元素不仅是大小写敏感的，而且要求元素中的次序必须按照指定的顺序出现。例如，XML头必须是文件中的第一行，而web-app元素必须是根元素。在web-app元素内，元素的次序也很重要。服务器不一定强制要求这种次序，但可以拒绝执行含有次序不正确的元素的Web应用。这表示使用非标准元素次序的web.xml文件是不可移植的。



接下来介绍web.xml中web-app元素的定义及具体功能。web-app元素是Web应用程序的根部署描述符。web-app元素有一个必需的属性version，来指定部署描述符符合哪个版本的要求。例如：

|  |
| --- |
| <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd" id="WebApp\_ID" version="3.0">  </web-app> |

version属性指定支持的Servlet版本。其中xsd文档的作用是定义XML文档的合法构建模块，具体xsd文档的配置可参考如下两个XML Schema(.xsd)教程。

http://www.w3school.com.cn/schema/index.asp

http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd

接下来介绍web.xml文件中的所有标签元素的含义。



icon：icon元素指出IDE和GUI工具用来表示Web应用的一个和两个图像文件的位置。

display-name：display-name元素提供GUI工具可能会用来标记这个特定的Web应用的一个名称。

description：description元素给出与此有关的说明性文本。

context-param：context-param元素声明应用范围内的初始化参数。

filter：过滤器元素将一个名字与一个实现javax.servlet.Filter接口的类相关联。

filter-mapping：一旦命名了一个过滤器，就要利用filter-mapping元素把它与一个或多个servlet或JSP页面相关联。

listener：servlet API的版本2.3增加了对事件监听程序的支持，事件监听程序在建立、修改和删除会话或servlet环境时得到通知。Listener元素指出事件监听程序类。

servlet：在向servlet或JSP页面制定初始化参数或定制URL时，必须首先命名servlet或JSP页面。Servlet元素就是用来完成此项任务的。

servlet-mapping：服务器一般为servlet提供一个缺省的URL：http://host/webAppPrefix/servlet/ServletName。但是，常常会更改这个URL，以便servlet可以访问初始化参数或更容易地处理相对URL。在更改缺省URL时，使用servlet-mapping元素。



session-config：如果某个会话在一定时间内未被访问，服务器可以抛弃它以节省内存。可通过使用HttpSession的setMaxInactiveInterval方法明确设置单个会话对象的超时值，或者可利用session-config元素制定缺省超时值。

mime-mapping：如果Web应用具有想到特殊的文件，希望能保证给他们分配特定的MIME类型，则mime-mapping元素提供这种保证。

welcom-file-list：welcome-file-list元素指示服务器在收到引用一个目录名而不是文件名的URL时，使用哪个文件。

error-page：error-page元素使得在返回特定HTTP状态代码时，或者特定类型的异常被抛出时，能够制定将要显示的页面。

jsp-config: 用来提供Web应用程序中的JSP文件的全局配置信息

security-constraint：security-constraint元素制定应该保护的URL。它与login-config元素联合使用

login-config 用login-config元素来指定服务器应该怎样给试图访问受保护页面的用户授权。它与sercurity-constraint元素联合使用。

security-role元素给出安全角色的一个列表，这些角色将出现在servlet元素内的security-role-ref元素的role-name子元素中。分别地声明角色可使高级IDE处理安全信息更为容易。。



resource-ref元素声明一个资源工厂使用的外部资源。

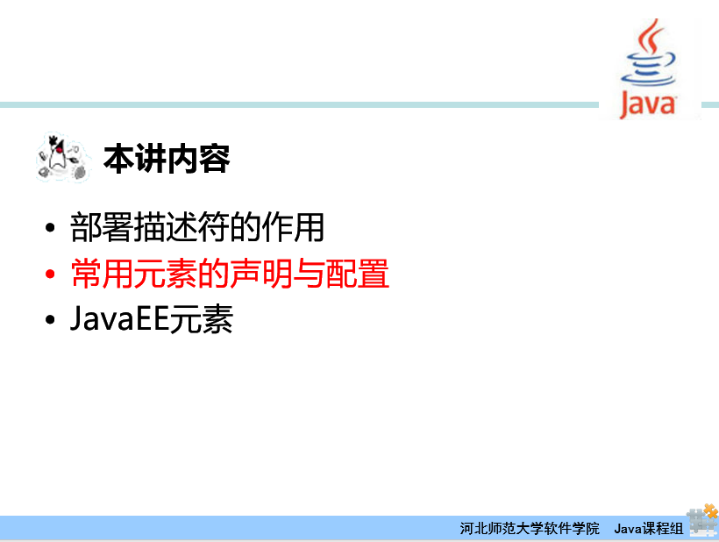
resource-env-ref元素声明与资源相关的一个管理对象。

distributable支持集群的服务器可安全的在多个服务器上分布Web应用.

env-entry声明了一个应用程序的环境入口

ejb-ref声明了一个对企业bean的home引用

ejb-local-ref声明了对企业bean的本地home引用



下一部分详细介绍常见标签元素的声明与配置。

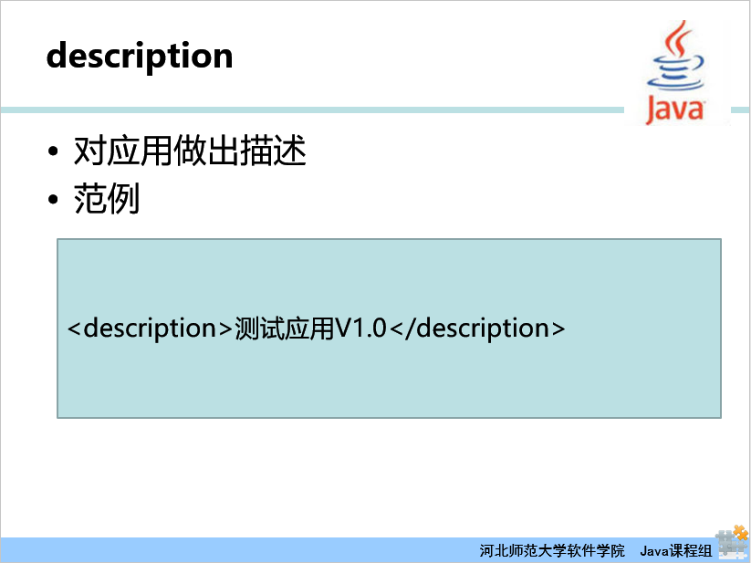


web.xml中包含small-icon和large-icon元素表示Web应用图标，指出IDE和GUI工具用来表示Web应用的大图标和小图标。

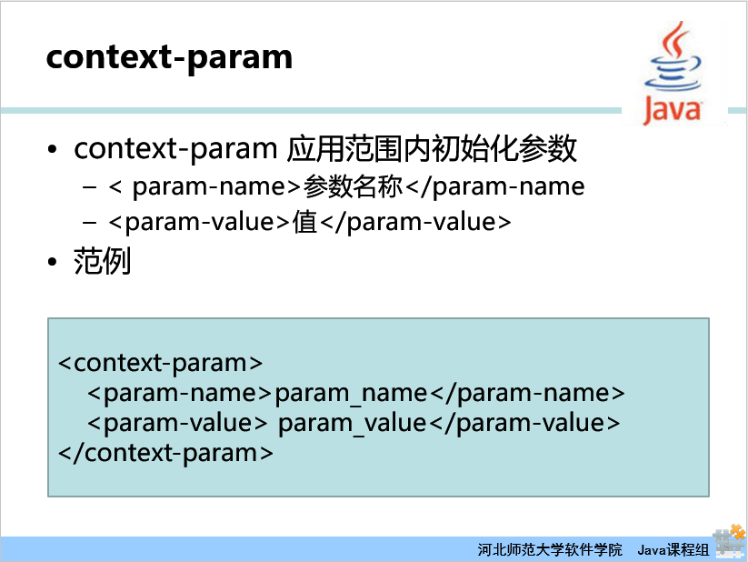
|  |
| --- |
| <icon>  <small-icon>/images/small.gif</small-icon>  <large-icon>/images/large.gif</large-icon>  </icon> |



Web 应用名称：提供GUI工具可能会用来标记这个特定的Web应用的名称。



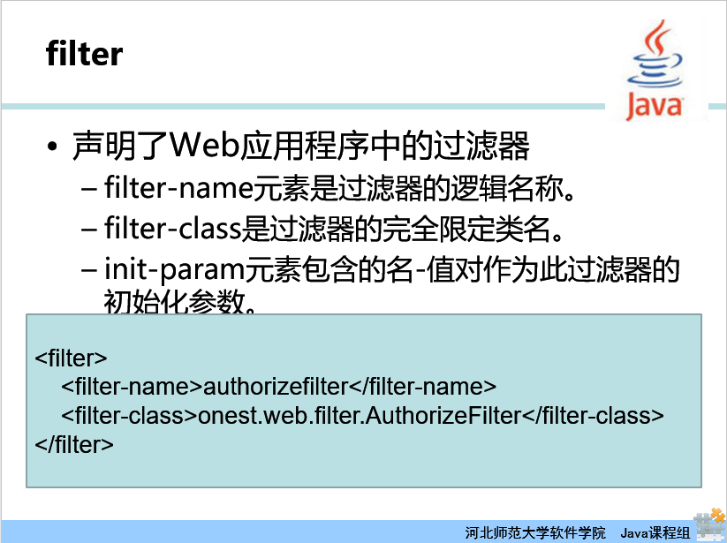
description元素提供了父元素的文本描述。此元素不仅出现在web-app元素中，其他很多元素中也有。



上下文参数：声明应用范围内的初始化参数：

|  |
| --- |
| <context-param>  <param-name>ContextParameter</para-name>  <param-value>test</param-value>  <description>It is a test parameter.</description>  </context-param> |

在servlet里面可以通过getServletContext().getInitParameter("ContextParameter")得到，在JSP网页中可以使用${initParam.param\_name}的方法来取得对应参数的值。



filter标签声明了Web应用程序中的过滤器，使用filter-name的值来引用。filter-name元素是过滤器的逻辑名称。它在Web应用程序中必须是唯一的。filter-name元素的内容不能为空。filter-class是过滤器的完全限定类名。init-param元素包含的名-值对作为此过滤器的初始化参数。当指定可选的async-supported元素时，表示该过滤器支持异步请求处理。

|  |
| --- |
| <filter>  <filter-name>authorizefilter</filter-name>  <filter-class>onest.web.filter.AuthorizeFilter</filter-class>  </filter> |

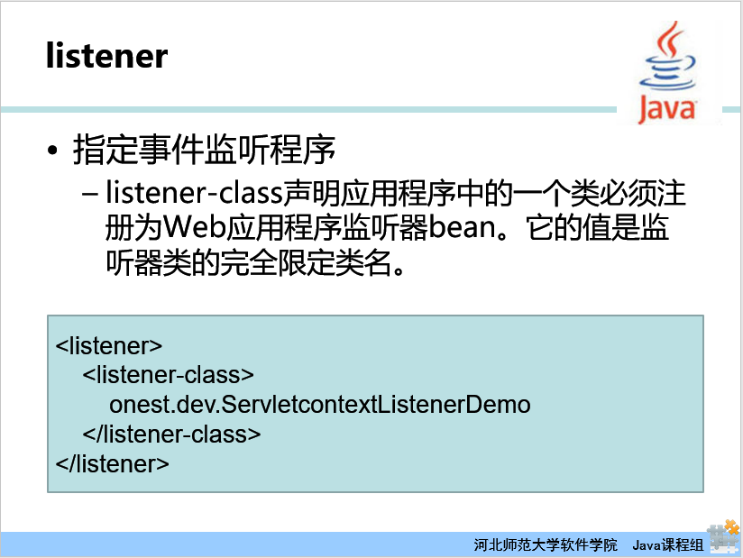
这段filter配置表示定义了一个过滤器，名字叫做authorizefilter。实际的处理类是onest.web.filter.AuthorizeFilter。



容器使用filter-mapping配置的url-pattern结点决定过滤到哪些url的请求。filter-name的值必须是部署描述符中声明的一个过滤器。匹配的请求被指定为url-pattern。

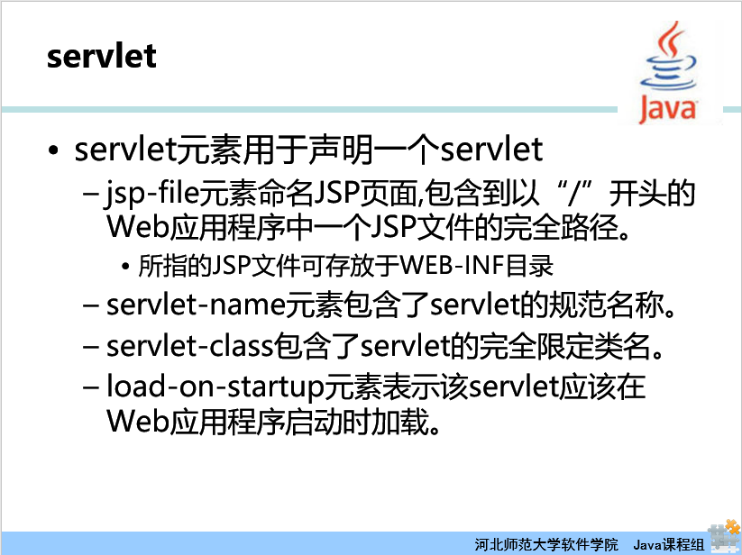
|  |
| --- |
| <filter>  <filter-name>authorizefilter</filter-name>  <filter-class>onest.web.filter.AuthorizeFilter</filter-class>  </filter>  <filter-mapping>  <filter-name>authorizefilter</filter-name>  <url-pattern>/admin/\*</url-pattern>  </filter-mapping> |

这段filter-mapping配置表示定义了一个过滤器的匹配方式，所有满足/admin/\*规则的url都要通过authorizefilter过滤器进行过滤。



listener结点指定事件监听程序：listener-class声明应用程序中的一个类注册为Web应用程序监听器bean。它的值是监听器类的完全限定类名。Servlet监听器用于监听一些重要事件的发生，监听器对象可以在事情发生前、发生后可以做一些必要的处理。

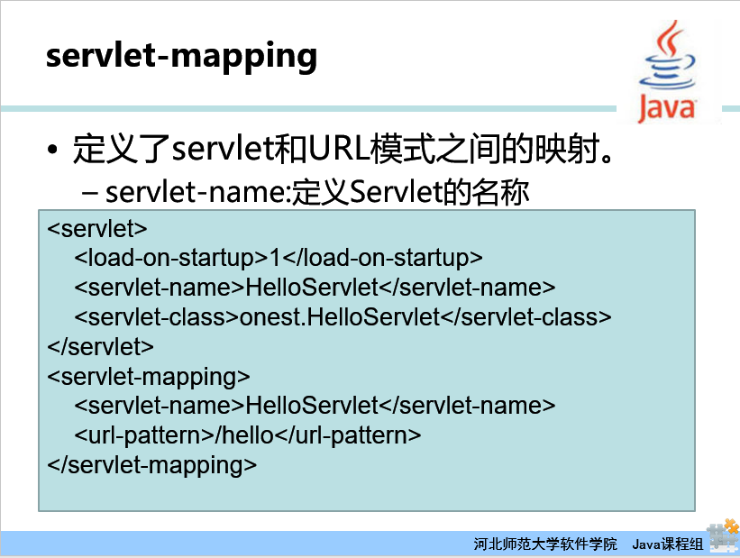
|  |
| --- |
| <listener>  <listener-class>  onest.dev.ServletcontextListenerDemo  </listener-class>  </listener> |



servlet元素用于声明一个Servlet,其中jsp-file元素命名JSP页面,包含到以“/”开头的Web应用程序中一个JSP文件的完全路径，其所指向的JSP文件存放于WEB-INF目录中。 servlet-name元素定义了容器中的一个Servlet名字。servlet-class包含了servlet对应的全包名类。load-on-startup元素表示该servlet应该在Web应用程序启动时加载。

|  |
| --- |
| <servlet>  <servlet-name>HelloServlet</servlet-name>  <jsp-file>/hello.jsp</jsp-file>  <load-on-startup>1</load-on-startup>  </servlet>  <servlet>  <servlet-name>HelloServlet</servlet-name>  <servlet-class>onest.HelloServlet</servlet-class>  <load-on-startup>1</load-on-startup>  </servlet> |

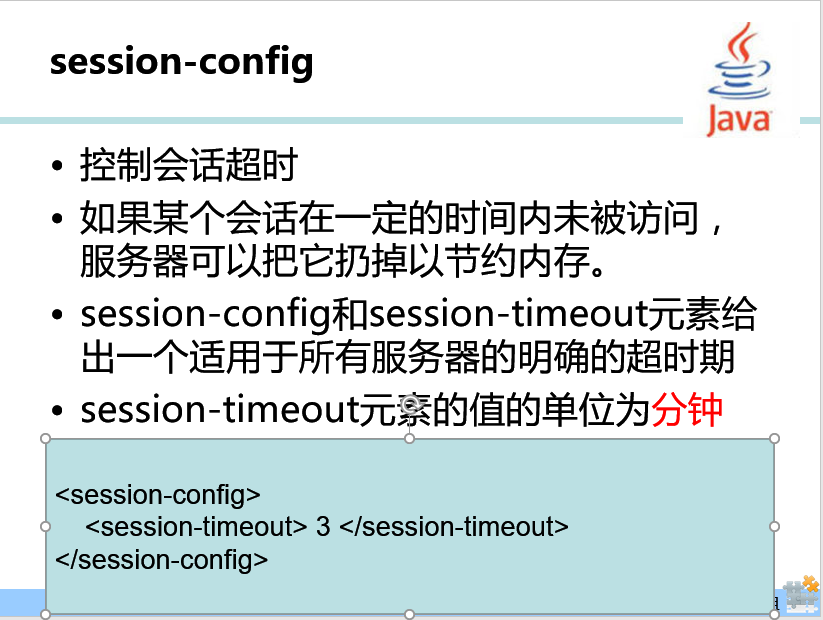
第一段配置代表定义名为HelloServlet的servlet元素，对应jspfile为/hello.jsp文件，并且启动工程后自动加载。第二段配置代表定义名为HelloServlet的servlet元素，对应onest.HelloServlet文件，并且启动工程后自动加载。



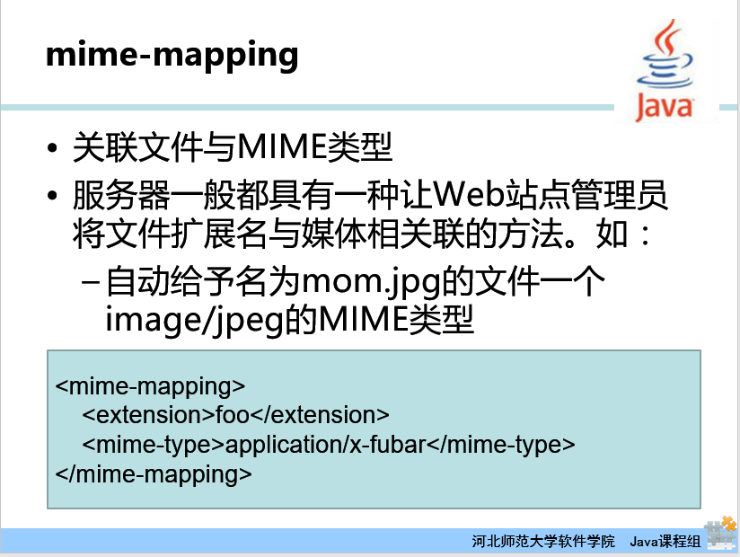
servlet-mapping结点同servlet结点共同使用，定义了Servlet和URL模式之间的映射。servlet-name:定义Servlet的名称，url-pattern:使用servlet-mapping元素将定制的URL与刚分配的名称相关联。例如：配置文件如下：

|  |
| --- |
| <servlet>  <load-on-startup>1</load-on-startup>  <servlet-name>HelloServlet</servlet-name>  <servlet-class>onest.HelloServlet</servlet-class>  </servlet>  <servlet-mapping>  <servlet-name>HelloServlet</servlet-name>  <url-pattern>/hello</url-pattern>  </servlet-mapping> |

定义了名为HelloServlet的servlet结点，其对应的处理类为onest.HelloServlet类，且自行加载。servlet-mapping结点定义了一组需要HelloServlet来处理的url，在url-pattern结点中配置了/hello的请求匹配模式，所以当请求/hello地址的请求，由onest.HelloServlet来处理。



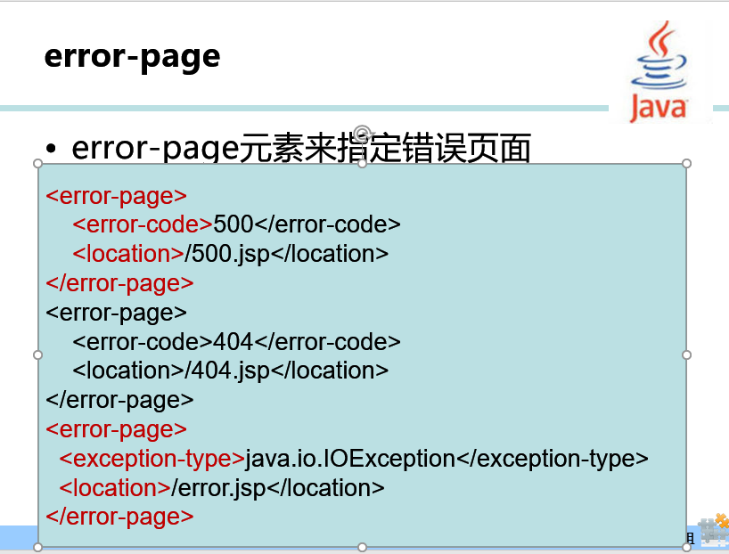
session-config配置会话超时时间。如果某个会话在一定的时间内未被访问，服务器可以把它扔掉以节约内存。其中session-config和session-timeout元素给出一个服务器会话的超时时间，session-timeout元素值的单位为分钟。配置效果为session超时时间为3分钟。



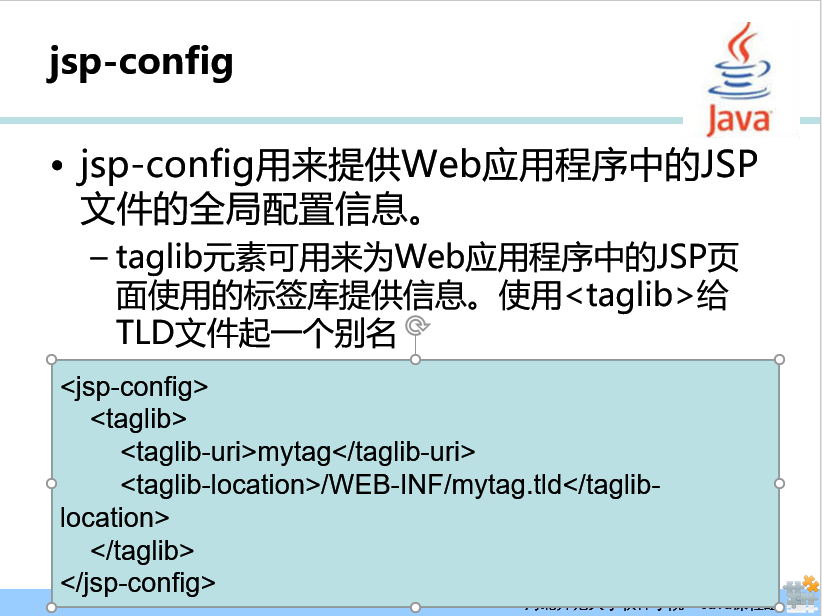
关联文件与MIME类型，服务器一般都具有一种让Web站点管理员将文件扩展名与媒体相关联的方法。如：自动给予名为mom.jpg的文件一个image/jpeg的MIME类型，如果希望将特殊的文件发送到客户端时分配为某种MIME类型，使用mime-mapping元素。



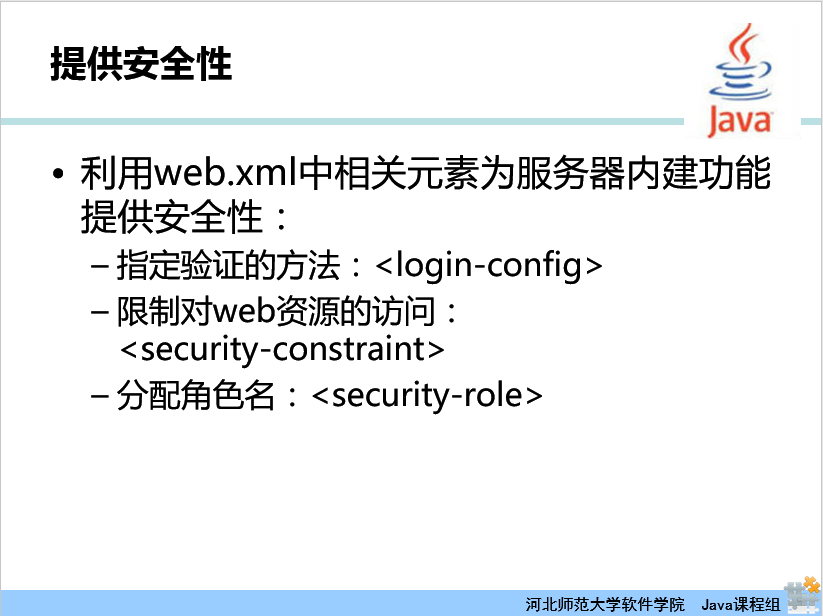
通过配置welcome-file-list元素可指定工程的欢迎页，如果一个URL给出了一个目录名但未给出文件名，服务器应该使用指定的欢迎页面。



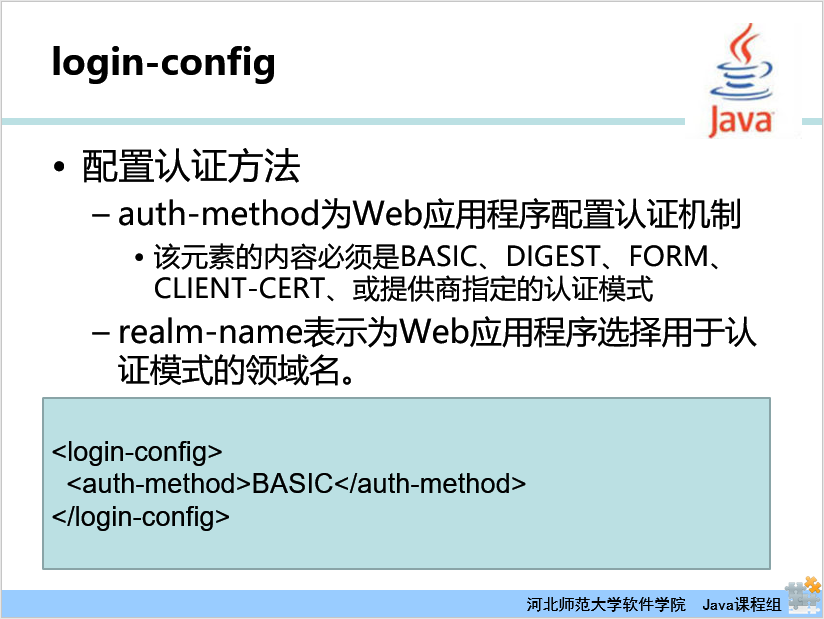
error-page元素来指定程序错误页面，包含一个错误代码或异常类型到Web应用程序中资源路径之间的映射，子节点error-code代表错误代码，location包含了web应用程序中相对于Web应用程序根目录的资源位置。location的值必须以'/'开头，exception-type包含了一个Java异常类型的完全限定类名称。



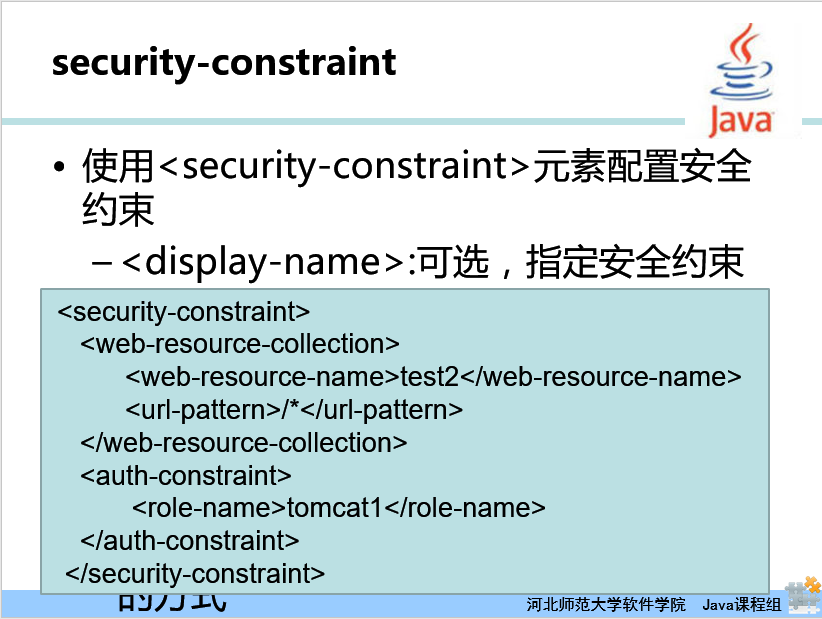
jsp-config结点用来提供Web应用程序中JSP文件的全局配置信息。taglib元素可用来为Web应用程序中JSP页面使用的标签库提供信息。使用<taglib>给TLD文件起一个别名，jsp-property-group:给一定范围类的JSP文件设置一些特性。课件中定义了一个mytag标签库，对应的描述文件为/WEB-INF/mytag.tld。



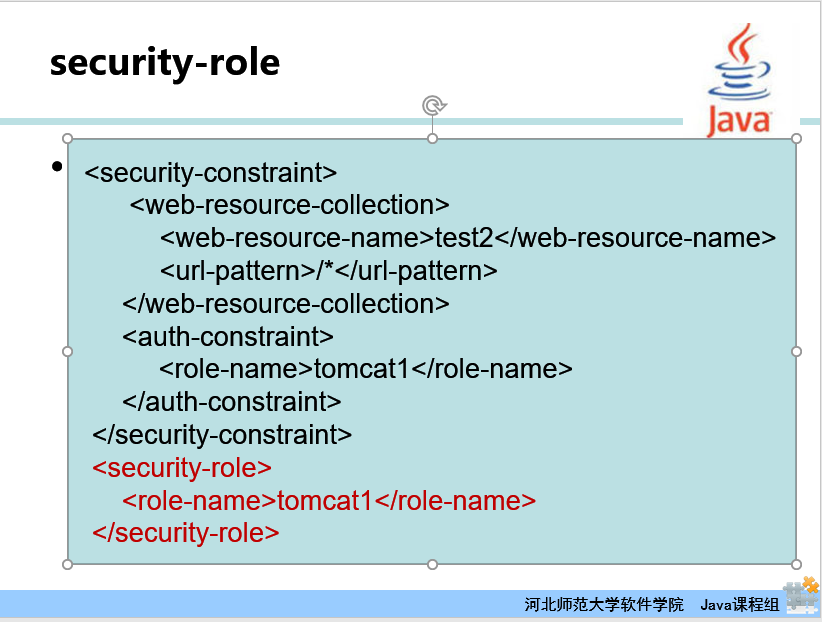
也可以使用web.xml中相关元素为服务器内建功能提供安全性保障，例如：指定验证的方法可定义<login-config>标签;限制对web资源的访问可使用<security-constraint>标签;分配角色名称，可使用<security-role>标签。



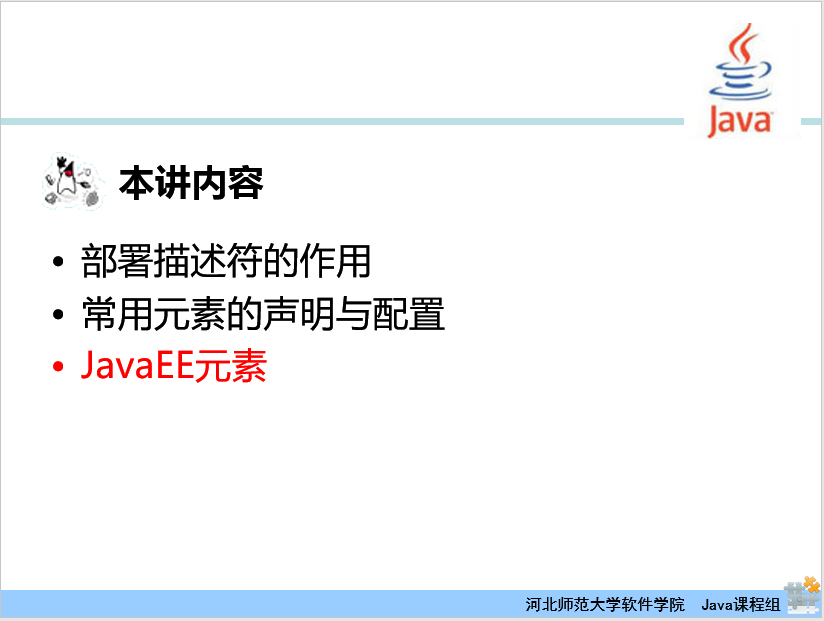
配置认证包括使用auth-method为Web应用程序配置认证机制，该元素的内容必须是BASIC、DIGEST、FORM、CLIENT-CERT、或提供商指定的认证模式，realm-name表示为Web应用程序选择用于认证模式的领域名。form-login-config指定应该用于基于表单登录的登录方式和错误页面。



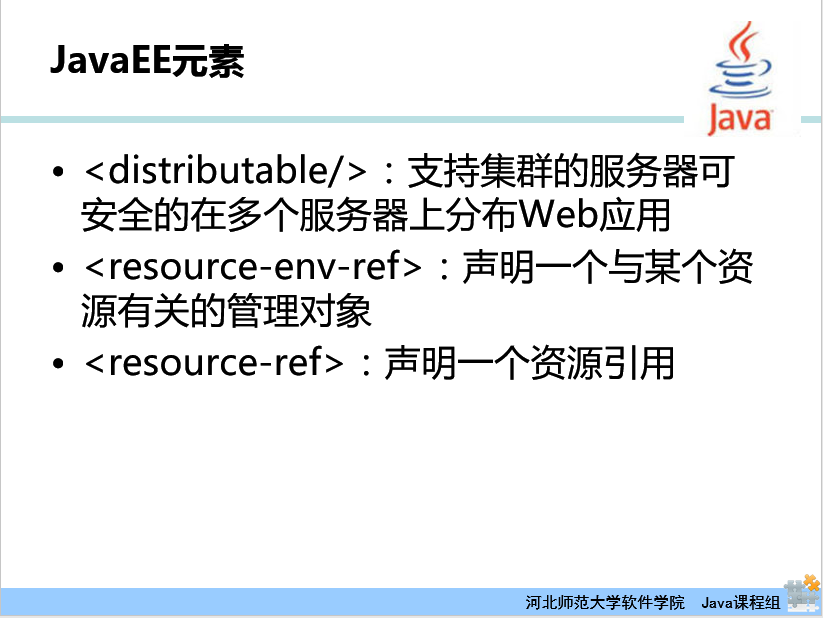
在开发中可使用<security-constraint>元素配置安全约束，其中<display-name>:可选，指定安全约束名称，<web-resource-collection>:指定安全约束应用的资源集合，<auth-constraint>:指定可访问受保护资源的角色，<user-data-constraint>:指定数据通讯的方式。



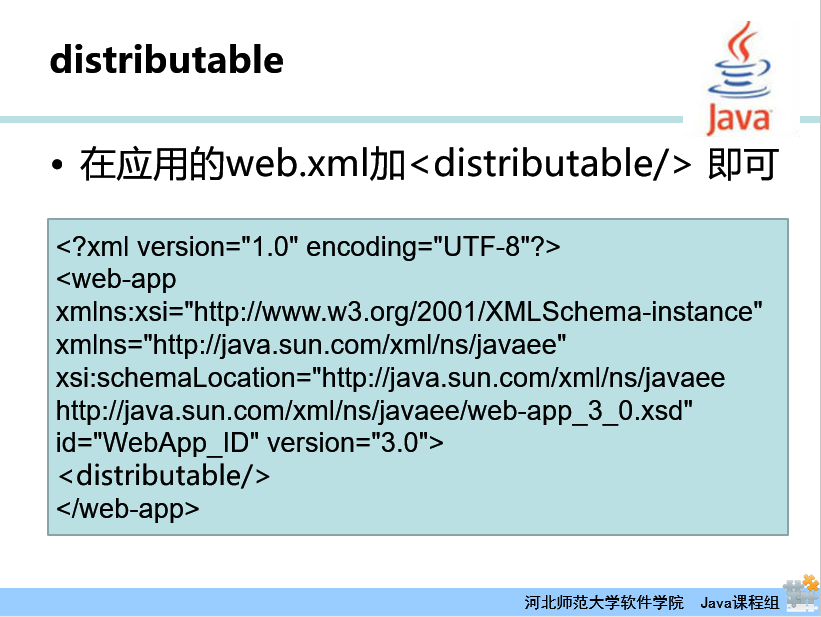
security-role元素定义了一个安全角色。子元素role-name指定安全角色的名称。常见的部署描述符的元素声明与配置就讲解完毕了，接下来介绍JavaEE元素。



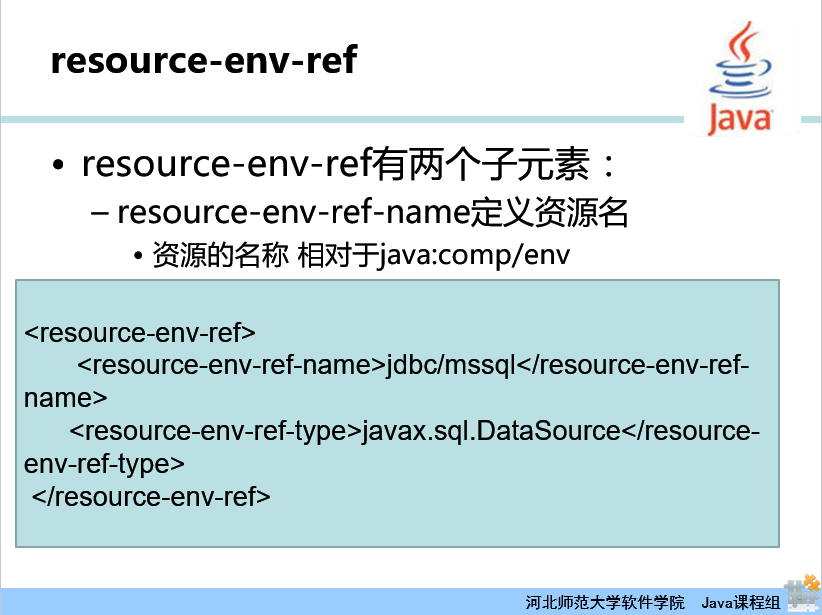
在部署描述符文件中也包含一系列的JavaEE元素的配置，接下来进行详细介绍。



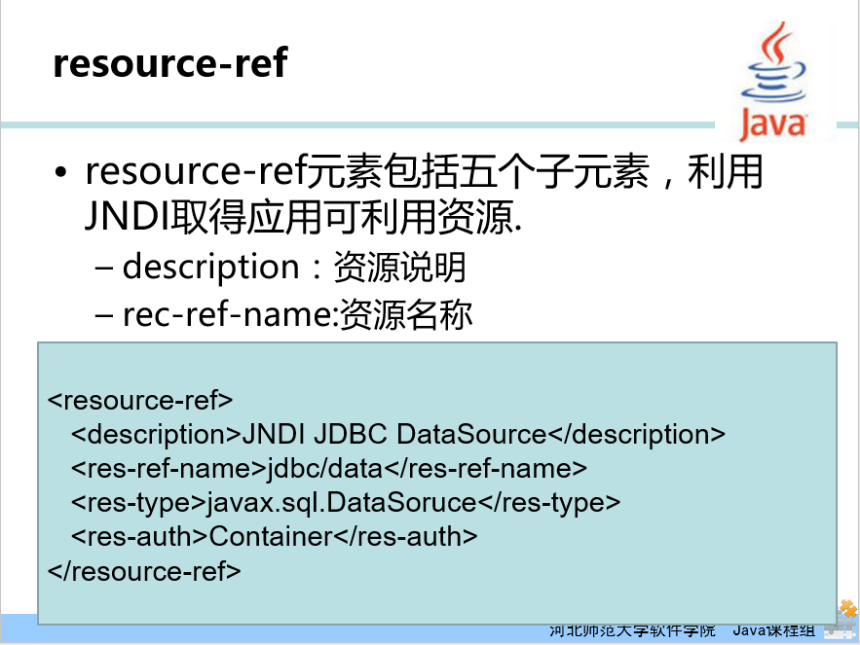
其中<distributable/>元素代表支持集群服务器，可安全的在多个服务器上分布Web应用，<resource-env-ref>元素声明一个与某个资源有关的管理对象，<resource-ref>元素声明一个资源引用。



如果希望该Web应用支持集群服务器。可安全的部署在多个服务器上，则加入<distributable/>结点。



resource-env-ref结点有两个子元素，分别是resource-env-ref-name定义资源名，资源的名称相对于java:comp/env，resource-env-ref-type查找资源时返回的资源类型，当web应用查找该资源的时候，返回的Java类名的全称。



resource-ref元素包括五个子元，利用JNDI取得应用可利用资源：

description：资源说明

rec-ref-name:资源名称

res-type:资源种类

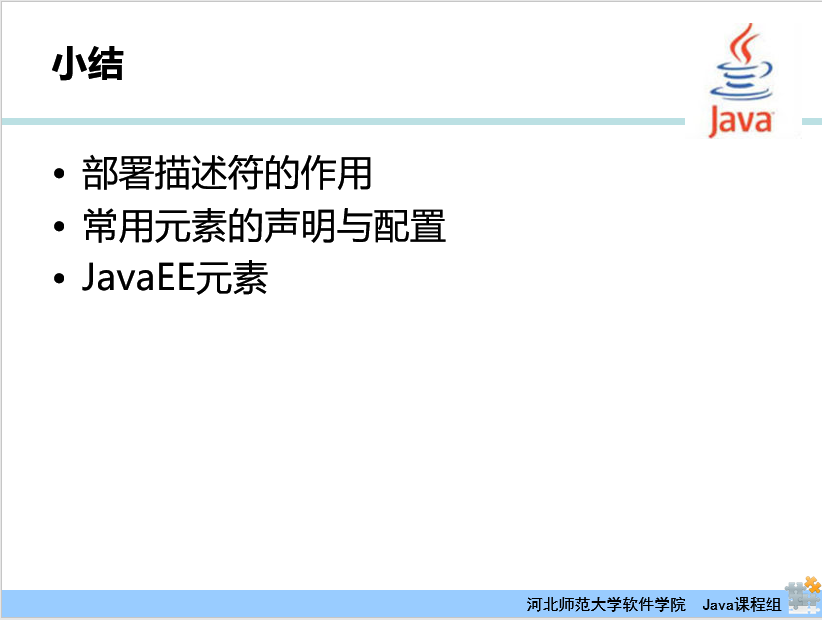
res-auth:资源由Application或Container来许可

res-sharing-scope:资源是否可以共享.默认值为 Shareable。

部署描述符部分的内容讲解完毕，接下来进行总结。

## 课时小结

（2分钟）



首先介绍了部署描述符的作用，然后详细介绍了常用部署描述符元素的声明与配置，最后介绍了部署描述符中JavaEE相关元素的配置。

# 布置作业

（）