**Servlet程序结构与部署**

# 授课内容

授课科目：JavaEE

授课内容：Servlet程序结构与部署

授课类型：讲授+实践

授课时间：3+1学时

主讲教师：孟双英

# 教学目标要求

## 能力目标：

1. 培养学生部署JavaEE应用的能力；
2. 培养学生分析总结JavaEE应用结构的能力。

## 知识目标：

1. 总结JavaEE程序开发目录结构与部署目录结构；
2. 总结Servlet的配置和映射方式和技巧；
3. 总结Tomcat中部署JavaEE程序的三种方式。

# 课件分析

概 述： JavaEE规范的核心技术学习结束后，本章内容是一次总结和复习，重点复习JavaEE 程序的开发目录结构和部署目录结构，JavaEE应用的部署描述符以及Servlet的映射和配置的方式，总结Tomcat中部署JavaEE应用的三种方式。

教学重点：Tomcat中三种部署JavaEE应用程序的方式。

教学难点：JavaEE应用程序的开发和部署结构的区别以及开发和部署的环境区别。

# 教学方法

回顾法、问题教学法、案例法、讲授法、演示法等。

# 教学过程

## 课前补充

（5分钟）

学完JavaEE的内容后，很多同学只会在eclipse中写代码，运行代码，而写好的代码却不知道该如何拿出eclipse在外面运行，很显然是受了课程重视开发知识的讲解和传授的影响，首先，回顾本门课程的第一节课介绍的JavaEE开发环境的搭建过程（10分钟）。

请同学们思考，开发环境搭建的过程中哪些环节在部署环境中是不需要的？（可以通过提问的方式，比如：部署环境中是否需要写代码？不需要写代码是否eclipse就可以不安装，那么写好的程序该怎么运行呢）。

## 上节回顾

（0分钟）

## 作业点评

（0分钟）

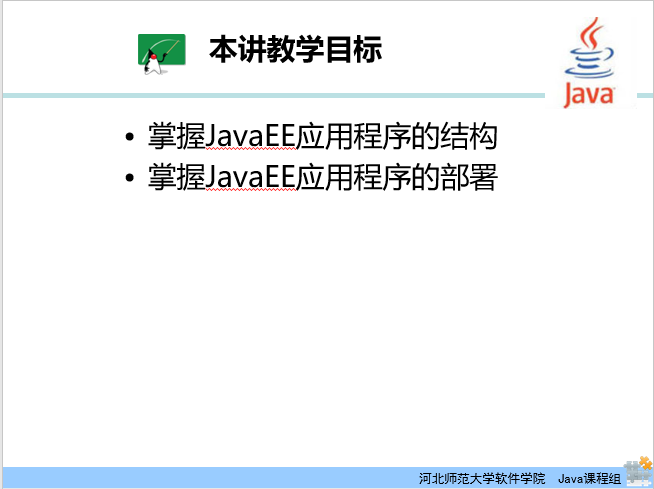
## 导入新课

（5分钟）

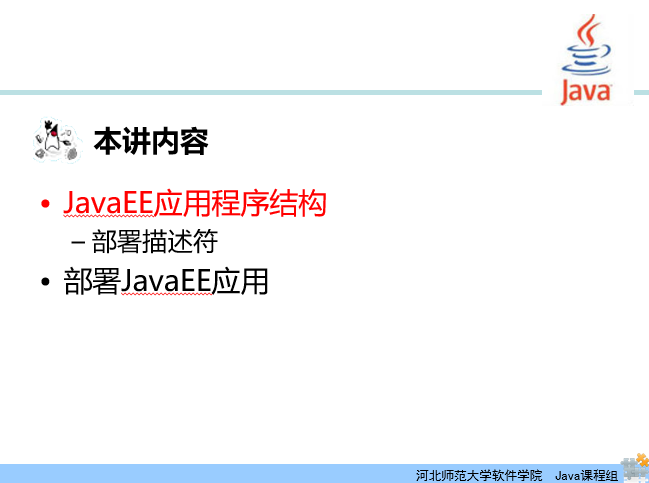
截止目前，JavaEE规范中的核心技术基本介绍全了，那开发的工作结束后，是不是所有的项目工作就结束了呢？开发好的程序如何部署到客户的环境里面去呢？难道要去客户的服务器上按照我们第一节课介绍的开发环境重新来一遍吗？显然不科学，也不可取，运行环境和开发环境还是不完全一样的，往往运行环境不需要开发IDE，本章内容将带大家回顾和总结JavaEE程序的部署。

## 讲授新课

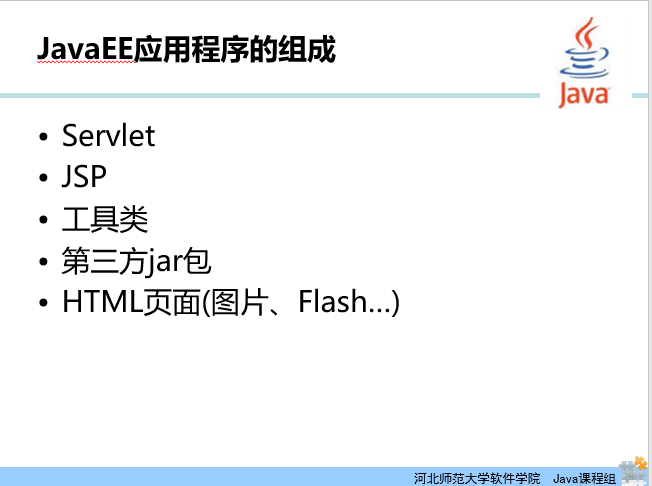
（130分钟）



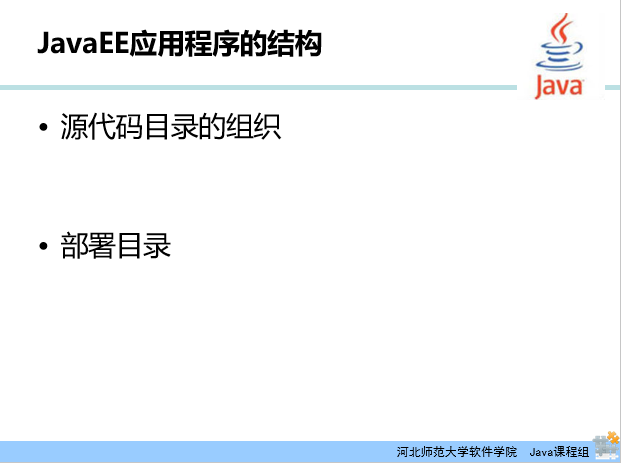
本章教学目标：掌握JavaEE应用程序的结构，包括开发的目录结构和部署的目录结构；掌握JavaEE应用程序的部署方式，为今后成为系统部署和运维师做准备。



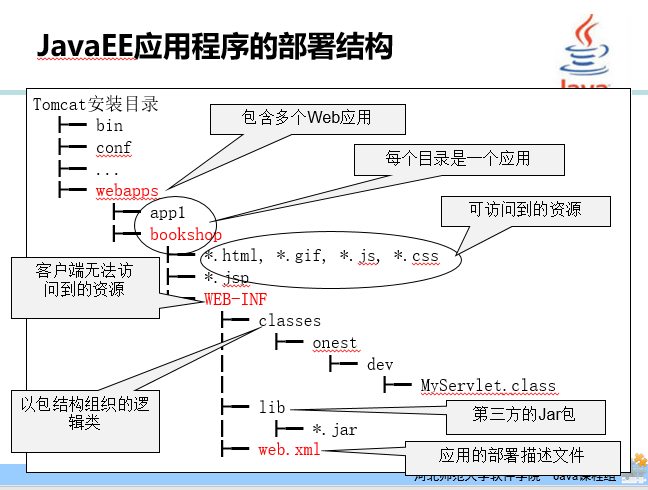
首先来回顾总结一下JavaEE开发的应用程序的组成和结构。



一般情况，JavaEE应用程序包括Servlet、HTML、JSP文件和一些工具类代码，除此之外还会包含一些第三方的函数库（jar），比如前面讲过的MySQL数据库驱动包mysql.jar，JSTL表达式jar等，那这么多的资源如何组织在一起呢？前面的课程介绍了开发中如何组织这些资源，今天再来重点介绍一下部署的时候这些资源应该如何组织在一起。

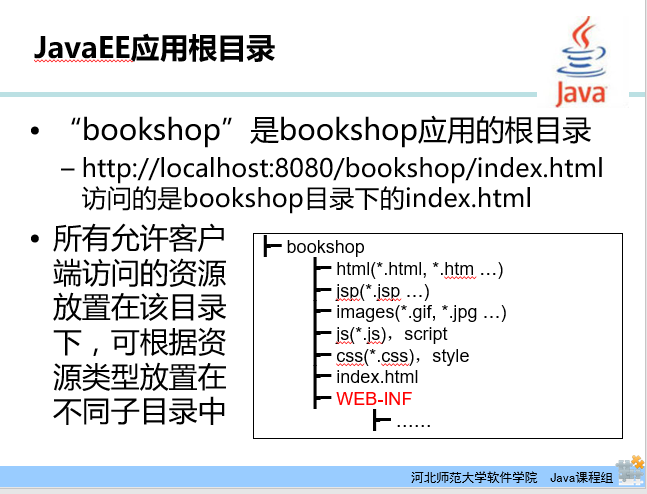


先回顾一下开发环境中的JavaEE程序源代码的目录组织，再展示部署后的JavaEE应用的目录结构，通过对比不难看出，部署后的目录结构与源代码中的WebContent目录结构基本一致。所以，JavaEE应用程序部署时部署的是源代码的WebContent目录，那源代码的src目录中的Java代码在部署的时候去哪了呢？



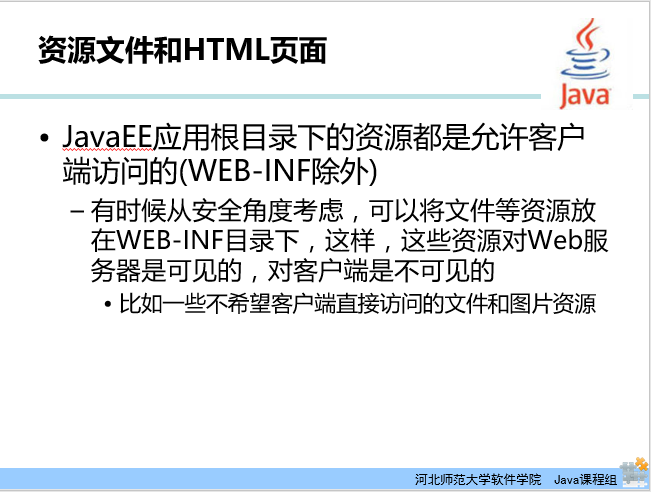
在JavaEE的程序部署结构图中，有一个WEB-INF目录，其中会包含classes目录，源代码的src目录中的.java文件经过编译生成的.class文件都会以包结构的形式被放于部署目录中的classes目录中。

下面来全面认识一下JavaEE应用程序的部署结构，JavaEE程序是部署在Servlet容器中的（Tomcat），Tomcat的webapps目录是其部署Web应用的目录，其中的每一个目录都是一个独立的Web应用程序，bookshop是部署在其中的一个JavaEE的应用程序，客户端能够直接访问到的资源放在根路径下，WEB-INF目录是客户端不能直接访问的资源文件，其中的classes存放以包结构组织Java的class文件，lib目录存放第三方的jar，web.xml文件是应用程序的部署描述文件。

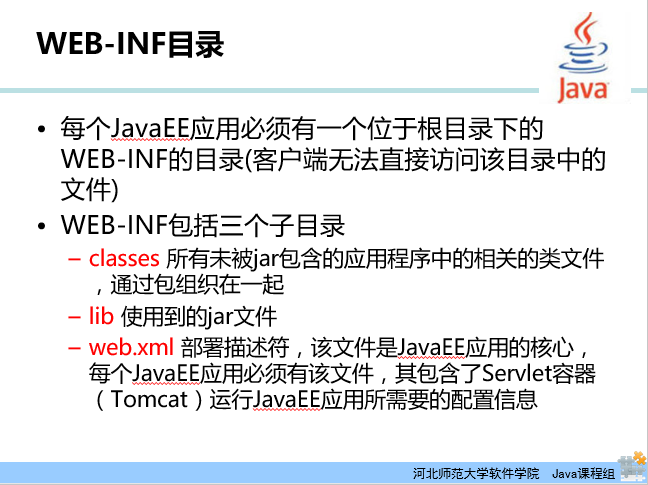


bookshop是此应用的根目录，访问http://localhost:8080/bookshop/index.html即是访问bookshop根路径下index.html页面。在web.xml中如果配置了首页，则访问<http://localhost:8080/bookshop/>会直接访问bookshop的首页。

允许客户端直接访问的资源和文件放于根路径下，可根据资源的类型进行再组织放于不同的子目录下，比如所有的图片资源放于images目录下，css资源放于css目录下，jsp文件放于jsp目录下等。

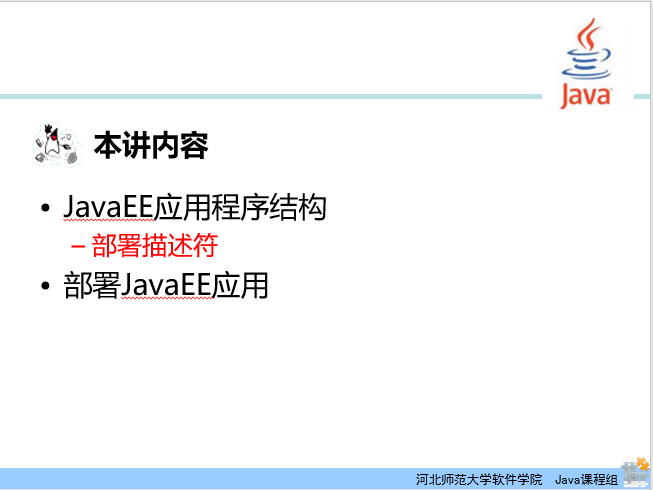


有时处于安全考虑，一些不能由客户端直接访问的文件或资源可放于WEB-INF目录下，比如一些需要通过服务器端下载的文件资源。

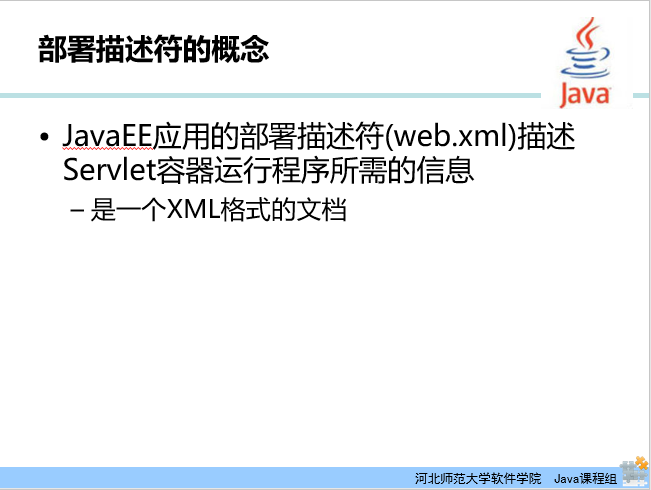


JavaEE应用程序的WEB-INF目录是必不可少的，WEB-INF目录中的内容是不能直接被客户端访问的（示例 ：在部署后的JavaEE应用的WEB-INF目录中放入一张图片，在客户端浏览器直接访问，将图片放入WEB-INF目录外，再次在浏览器访问，观察结果）。

WEB-INF包含三个子目录：classes、lib、web.xml，classes目录存放编译后的Java类，lib目录存放第三方的jar包，web.xml是应用的部署描述文件。



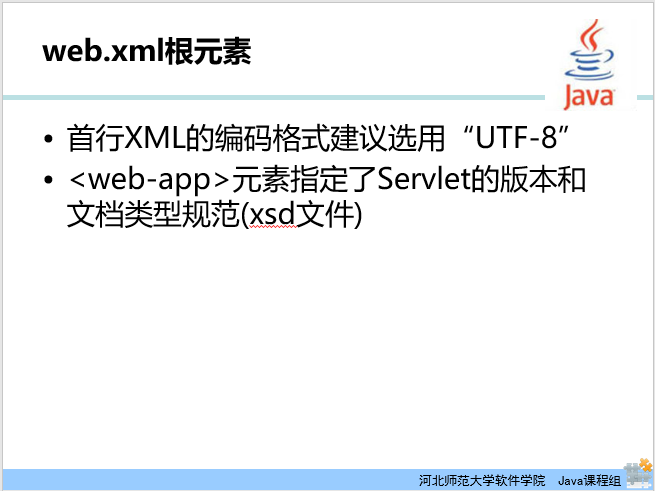
下面重点回顾一下部署描述符文件web.xml文件及其编写。



部署描述符文件中存放着Servlet容器运行JavaEE程序时需要读取的配置信息，是一个xml格式的文档，固定的名称是web.xml。



一个完整的web.xml文件示例如PPT所示，在Tomcat7.0以及之后版本中，JavaEE应用程序中可以以注解的形式取代web.xml文件，在Servlet规范中已介绍过，属于版本的更新。



web.xml文件的首行设置xml文档的编码格式，建议选择UTF-8，国际编码标准。<web-app>是JavaEE部署描述符的根节点，其中的属性用于指定Servlet版本和xml文档的类型规范xsd文件。



目前课程中接触到的部署描述符包括：Servlet配置相关的<servlet>和<servlet-mapping>标签，Filter配置相关的<filter>和<filter-mapping>，Listener配置相关的<listener>，Servlet初始化参数的<context-param>标签，应用的首页面设置<welcome-file-list>标签。

Servlet配置的代码如下：

|  |
| --- |
| <servlet>  <servlet-name>ServletName</servlet-name>  <servlet-class>xxx.xx.xx.xx.ServletName</servlet-class>  </servlet>  <servlet-mapping>  <servlet-name> ServletName </servlet-name>  <url-pattern>url</url-pattern>  </servlet-mapping> |

可以用@WebServlet注解取代。

Filter过滤器配置的代码如下：

|  |
| --- |
| <filter>  <filter-name>filtername</filter-name>  <filter-class>filter classname</filter-class>  </filter>  <filter-mapping>  <filter-name>filtername</filter-name>  <url-pattern>url</url-pattern>  </filter-mapping> |

可以用@WebFilter注解取代，但注解形式定制过滤器链的前后执行顺序。

Listener配置代码如下：

|  |
| --- |
| <listener>  <listener-class>listener classname</listener-class>  </listener> |

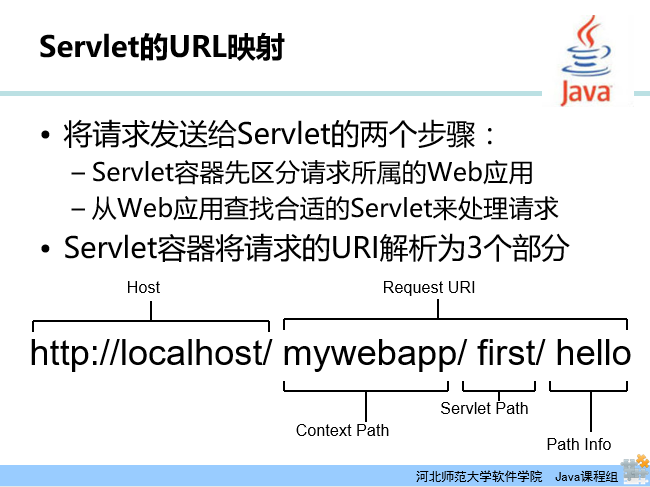
可以用@WebListener注解取代。

初始化参数配置代码如下：

|  |
| --- |
| <context-param>  <param-name>name</param-name>  <param-value>value</param-value>  </context-param> |

首页面配置代码如下：

|  |
| --- |
| <welcome-file-list>  <welcome-file>index.html</welcome-file>  <welcome-file>index.htm</welcome-file>  <welcome-file>index.jsp</welcome-file>  <welcome-file>default.html</welcome-file>  <welcome-file>default.htm</welcome-file>  <welcome-file>default.jsp</welcome-file>  </welcome-file-list> |

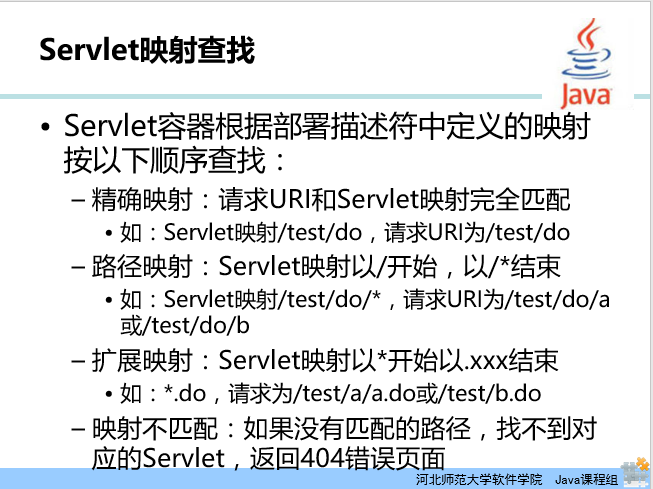


Servlet是JavaEE课程的重点内容，Servlet的映射配置必须重点掌握，浏览器提交一个请求给服务器端的Servlet，服务器端需要经过两个步骤的验证和查找，首先，Servlet容器会根据请求的URL找到所请求的Web应用，再从Web应用中查找合适的Servlet接收请求，URL的<http://localhost:8080/>是服务器的地址，mywebapp是具体的web应用名称，first/hello是应用的Servlet映射地址。Servlet容器把Request URI解析为3部分：ContextPath、ServletPath、PathInfo。



程序中URI的获取通过request.getRequestURI()，获得的URI去掉了协议和服务器地址。在程序中通过request.getContextPath()获得ContextPath地址，request.getServletPath()获得ServletPath地址，request.getPathInfo()获得PathInfo地址。

示例：演示获取ContextPath、ServletPath、PathInfo信息。

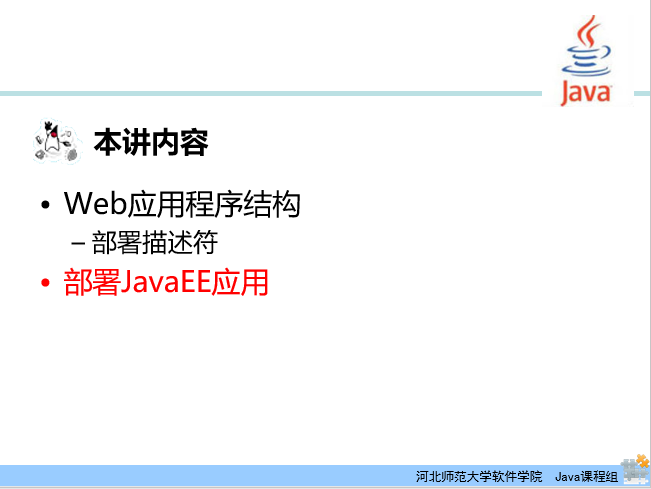


Servlet容器在查找Servlet时会根据web.xml中的映射（url-pattern）匹配Servlet，具体的查找顺序是：先查找精确映射，再查找路径映射，最后查找扩展映射。

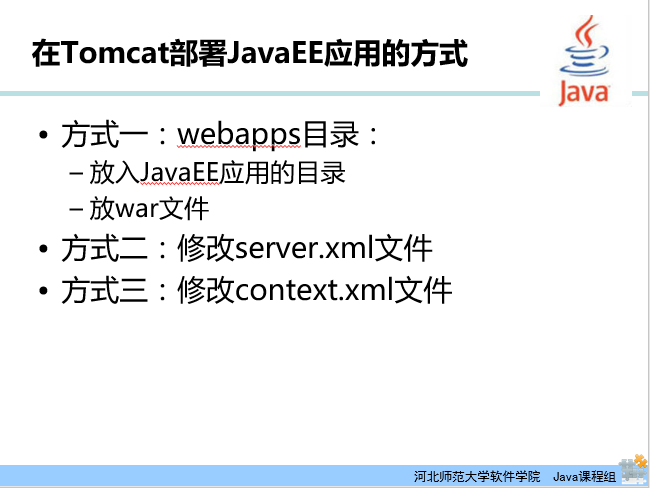
精确映射是指请求的URI与Servlet映射完全匹配，比如Servlet映射的地址是/test/do，请求的URI是/test/do。路径映射是指以/开始以/\*结束，比如Servlet映射的地址是/test/do/\*，请求的URI是/test/do/a或者/test/do/b等。扩展映射是指Servlet映射以\*开始以.xxx结束，比如Servlet映射地址是\*.do，请求的URI是test.do或者/test/a.do等。

如果通过以上三种方式都找不到与请求URI相匹配的Servlet路径则系统会返回404错误（找不到资源）。

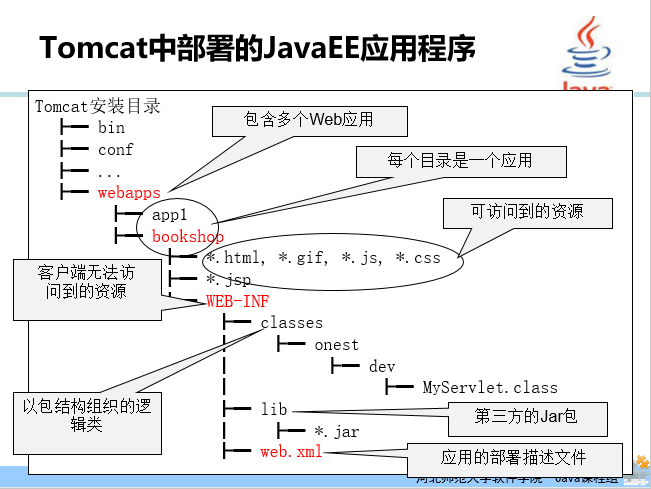
以上是关于JavaEE应用程序的结构和部署结构以及部署描述符的复习和总结。



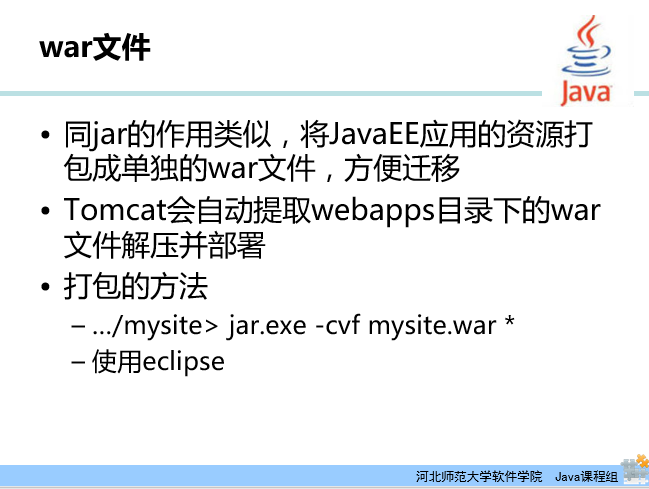
部署JavaEE应用是本章的重点内容，前面课程中也都提及了JavaEE应用的部署，接下来我们系统全面地总结和学习一下JavaEE应用程序的部署方式。



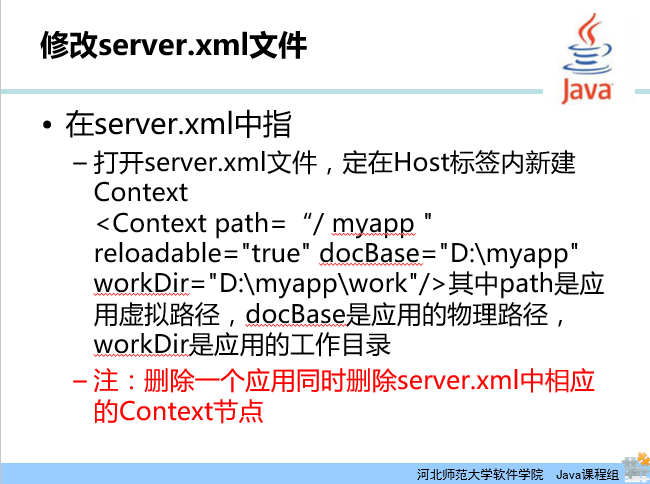
Tomcat中部署JavaEE应用程序的方式有三种：第一种方式，直接将组织好规则的JavaEE的应用程序放入Tomcat的webapps目录，或者将打包好的war文件放入webapps目录（演示）；第二种方式修改Tomcat中的server.xml文件；第三种方式，修改Tomcat中的context.xml文件。



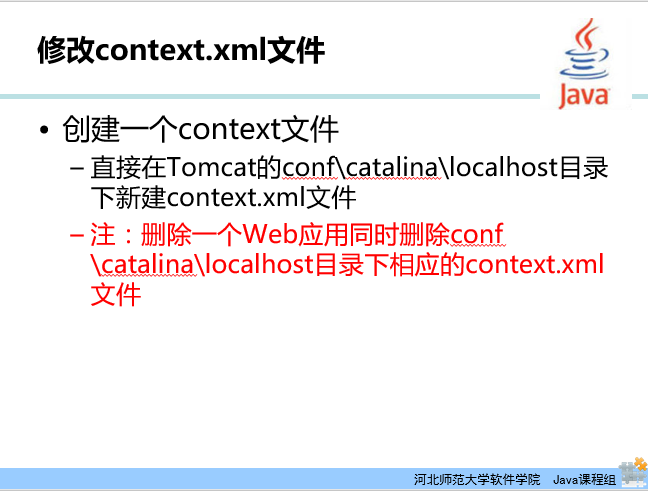
第一种方式JavaEE应用程序的目录结构必须满足图示的要求。



war文件同课程中用到的jar文件类似，是将JavaEE应用打包的一种方式，可以简单理解为压缩，就像使用压缩工具压缩一个文件夹一样，这种压缩打包方便JavaEE应用的迁移和拷贝。将war文件直接放在webapps目录下，Tomcat启动时会自动扫描webapps目录下的war文件解压并部署。在开发环境中打包JavaEE应用程序为war文件的方式有两种，一种是直接使用JDK中的jar工具，另一种借助eclipse工具打包（演示）。

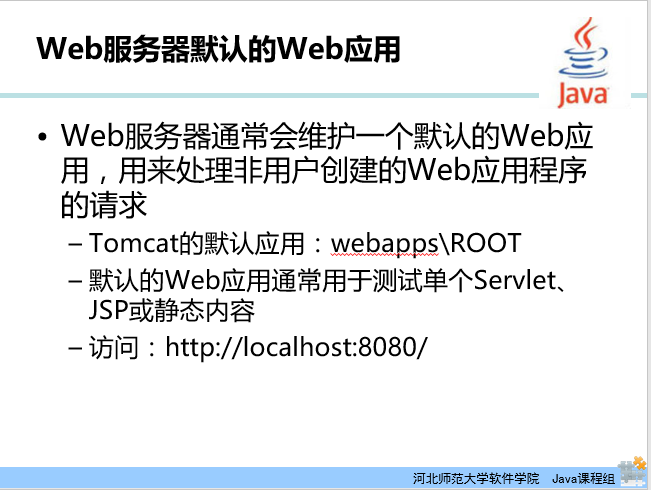


第二种部署方式，在Tomcat的conf目录下找到server.xml文件，打开server.xml文件，找到Host标签，在Host标签内新建<Context path="/myapp" reloadable="true" docBase="D:\myapp" workDir="D:\myapp\work" />标签，标签中的path属性是要部署的应用的虚拟路径，即浏览器访问时的应用名称，docBase是应用的物理路径，workDir是应用的工作目录。同理，从Tomcat中删除应用时需要删除server.xml文件中对应的<Context />标签。



第三种部署方式，在Tomcat的目录中找到conf\catalina\localhost目录，在其中新建context.xml文件，文件的内容与第二种方式的<context />标签内容一致，删除应用时，删除对应的context.xml文件即可。

对比三种部署方式，第一种操作简单，但要求应用的程序必须放在指定的目录中，第二种、第三种方式不限制应用的存放位置，需要额外增加配置信息，各有优劣。



细心的同学会发现，默认解压Tomcat后，在Tomcat的webapps目录下会有一个ROOT的应用，这是Tomcat服务器默认维护的一个应用程序，一般Web服务器都会维护类似的一个默认Web应用，用来处理非用户创建部署的Web应用请求，Tomcat的默认应用个就是ROOT，我们访问默认应用直接访问服务器的根路径即可，比如Tomcat的根路径默认是http://localhost:8080/，Tomcat的这一默认应用是用于测试单个Servlet或者JSP或者静态HTML内容的。

## 课时小结

（10分钟）

1. 复习JavaEE应用程序的结构和部署结构。
2. 复习JavaEE应用程序的部署描述符文件以及常用的配置标签。
3. 总结Tomcat中部署JavaEE应用程序的三种方式。
   1. 直接复制应用目录放于webapps目录。
   2. 修改server.xml文件部署。
   3. 修改context.xml文件部署。

# 布置作业

（2分钟）

完成之前开发应用程序的外部部署。