



Java程序设计

第05-03讲 Servlet程序结构与部署

Java课程组



本讲教学目标



- 掌握JavaEE应用程序的结构
- 掌握JavaEE应用程序的部署





本讲内容

- JavaEE应用程序结构
 - 部署描述符
- 部署JavaEE应用



JavaEE应用程序的组成



- Servlet
- JSP
- 工具类
- 第三方jar包
- HTML页面(图片、Flash...)



JavaEE应用程序的结构



Project Explorer

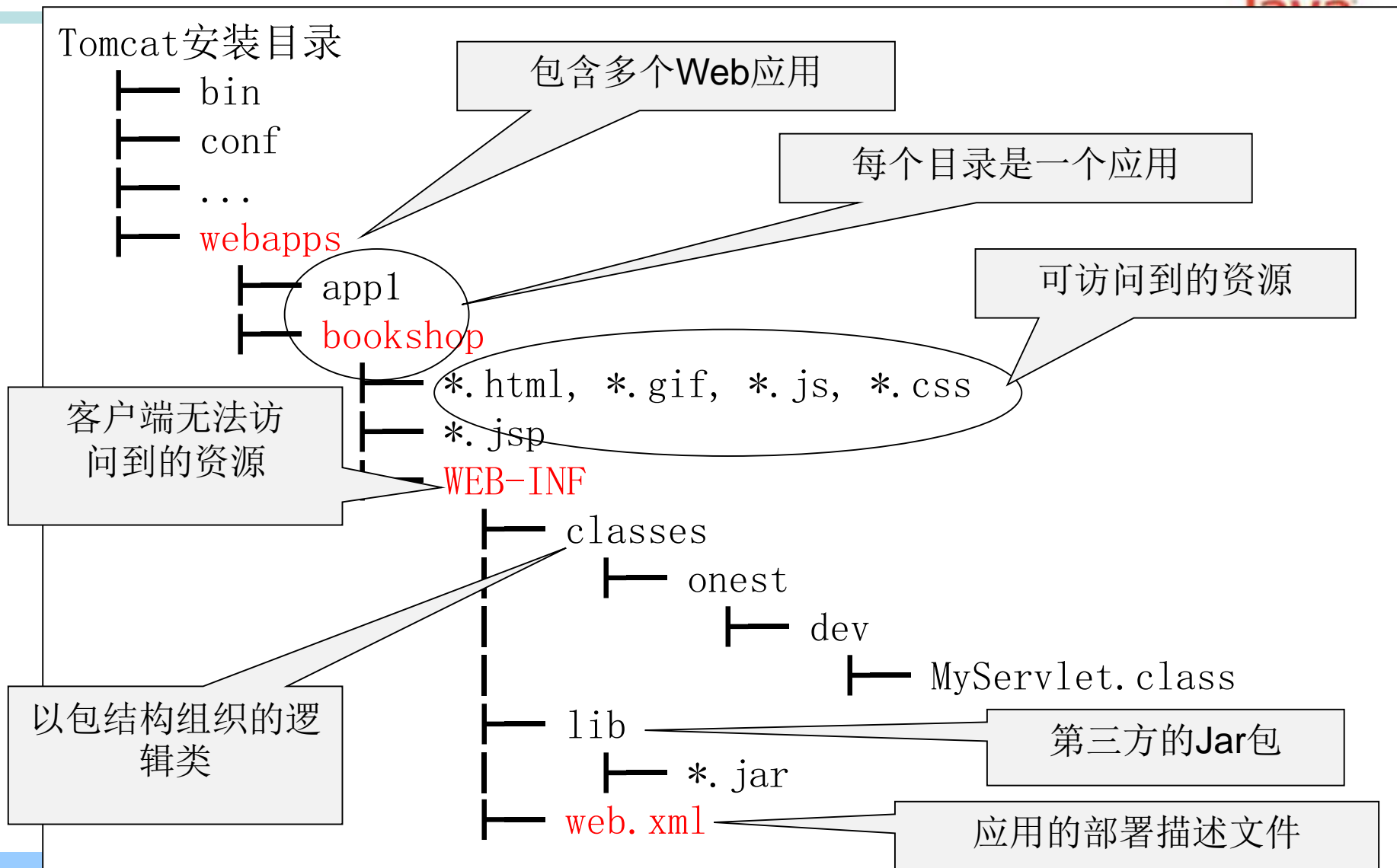
- bookshop
 - Deployment Descriptor: bookshop
 - JAX-WS Web Services
 - Java Resources
 - JavaScript Resources
 - build
 - doc
 - WebContent
 - admin
 - commons
 - images
 - inc
 - META-INF
 - script
 - style
 - temp
 - user
 - WEB-INF
 - lib
 - web.xml
 - deleteFiled.jsp
 - Error.jsp
 - index.html
 - index.jsp
 - OrderConfirm.jsp
 - productCart.jsp

的组织

demo_Hibernate > .metadata > .plugins > org.eclipse.wst.server.core > tmp0 > wtpwebapps > bookshop >

称	修改日期	类型	大小
admin	2017/7/13 16:22	文件夹	
commons	2017/7/13 16:22	文件夹	
images	2017/7/13 16:22	文件夹	
inc	2017/7/13 16:22	文件夹	
META-INF	2017/7/13 16:22	文件夹	
script	2017/7/13 16:22	文件夹	
style	2017/7/13 16:22	文件夹	
temp	2017/7/13 16:22	文件夹	
user	2017/7/13 16:22	文件夹	
WEB-INF	2017/7/13 16:22	文件夹	
deleteFiled.jsp	2009/10/15 14:15	JSP 文件	1 KB
Error.jsp	2009/10/15 14:11	JSP 文件	1 KB
index.html	2009/10/15 14:11	360 se HTML Do...	1 KB
index.jsp	2009/11/2 16:39	JSP 文件	5 KB
OrderConfirm.jsp	2009/10/30 14:31	JSP 文件	3 KB
productCart.jsp	2009/11/5 15:00	JSP 文件	4 KB

JavaEE应用程序的部署结构



JavaEE应用根目录



- “bookshop” 是bookshop应用的根目录
 - <http://localhost:8080/bookshop/index.html>访问的是bookshop目录下的index.html
- 所有允许客户端访问的资源放置在该目录下，可根据资源类型放置在不同子目录中

```
└─ bookshop
    ├── html(*.html, *.htm ...)
    ├── jsp(*.jsp ...)
    ├── images(*.gif, *.jpg ...)
    ├── js(*.js), script
    ├── css(*.css), style
    ├── index.html
    └── WEB-INF
        └── .....
```



资源文件和HTML页面



- JavaEE应用根目录下的资源都是允许客户端访问的(WEB-INF除外)
 - 有时候从安全角度考虑，可以将文件等资源放在WEB-INF目录下，这样，这些资源对Web服务器是可见的，对客户端是不可见的
 - 比如一些不希望客户端直接访问的文件和图片资源



WEB-INF目录



- 每个JavaEE应用必须有一个位于根目录下的WEB-INF的目录(客户端无法直接访问该目录中的文件)
- WEB-INF包括三个子目录
 - **classes** 所有未被jar包含的应用程序中的相关的类文件，通过包组织在一起
 - **lib** 使用到的jar文件
 - **web.xml** 部署描述符，该文件是JavaEE应用的核心，每个JavaEE应用必须有该文件，其包含了Servlet容器(Tomcat) 运行JavaEE应用所需要的配置信息





本讲内容

- JavaEE应用程序结构
 - 部署描述符
- 部署JavaEE应用



部署描述符的概念



- JavaEE应用的部署描述符(web.xml)描述Servlet容器运行程序所需的信息
 - 是一个XML格式的文档



示例



```
<?xml version="1.0" encoding="UTF-8"?>
<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">
  <display-name>06-03_001</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>
  <context-param>
    <param-name>context.text</param-name>
    <param-value>Hi, ContextParam</param-value>
  </context-param>
</web-app>
```



web.xml根元素



- 首行XML的编码格式建议选用 “UTF-8”
- `<web-app>` 元素指定了Servlet的版本和文档类型规范(xsd文件)



已接触到的描述符



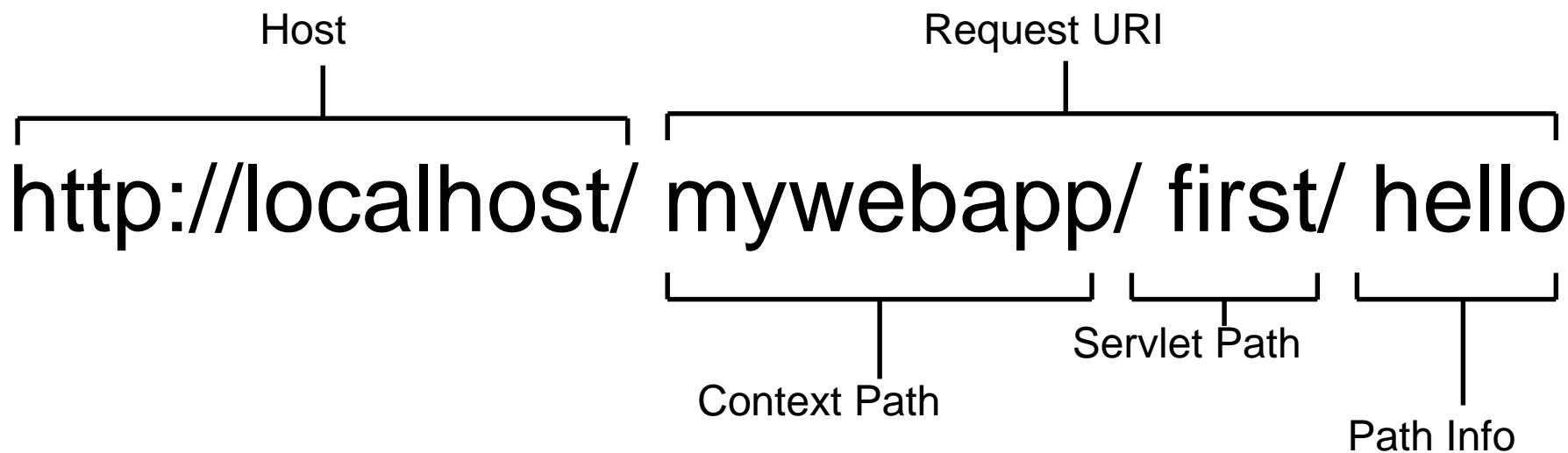
- Servlet Declarations:<servlet>
- Servlet Mappings :<servlet-mapping>
- Filter Declarations:<filter>
- Filter Mapping:<filter-mapping>
- Listener Declarations:<listener>
- ServletContext Init parameters :<context-param>
- Welcome File list:<welcome-file-list>



Servlet映射配置



- 将请求发送给Servlet的两个步骤：
 - Servlet容器先区分请求所属的Web应用
 - 从Web应用查找合适的Servlet来处理请求
- Servlet容器将请求的URI解析为3个部分



Request URI



- Request
URI=ContextPath+ServletPath+PathInfo
 - 使用request.**getRequestURI()**获得
- 上下文路径(ContextPath): Web应用的根目录
- Servlet路径(ServletPath): 由部署描述符中Servlet映射指定的路径
- 路径信息(PathInfo): Servlet之后的字符串
- HttpServletRequest对象提供:
 - **getContextPath()**,**getServletPath()**和**getPathInfo()**方法, 分别获取上下文路径、Servlet路径和路径信息



Servlet映射查找



- Servlet容器根据部署描述符中定义的映射按以下顺序查找：
 - 精确映射：请求URI和Servlet映射完全匹配
 - 如：Servlet映射/test/do，请求URI为/test/do
 - 路径映射：Servlet映射以/开始，以/*结束
 - 如：Servlet映射/test/do/*，请求URI为/test/do/a或/test/do/b
 - 扩展映射：Servlet映射以*开始以.xxx结束
 - 如：*.do，请求为/test/a/a.do或/test/b.do
 - 映射不匹配：如果没有匹配的路径，找不到对应的Servlet，返回404错误页面





本讲内容

- Web应用程序结构
 - 部署描述符
- 部署JavaEE应用



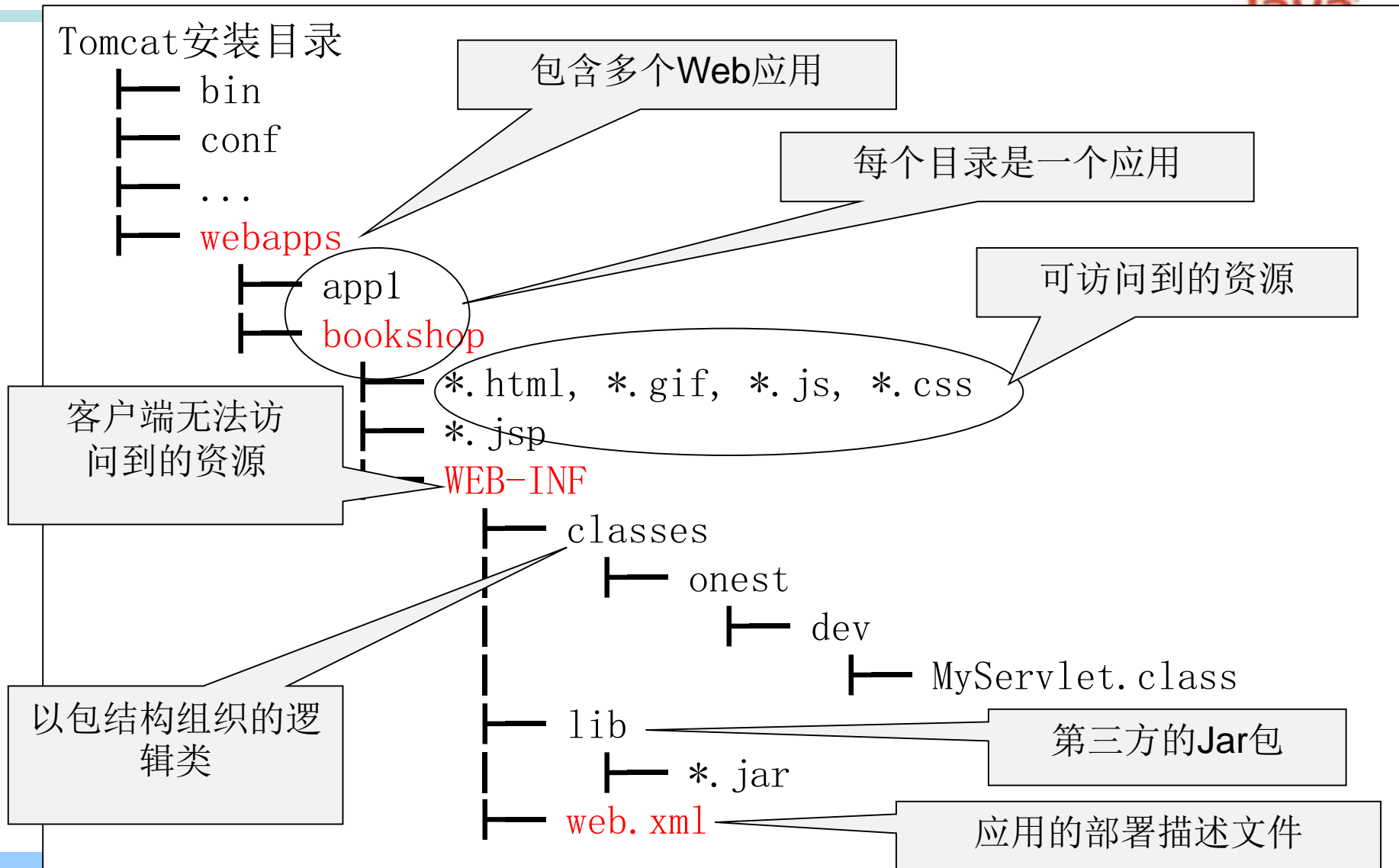
在Tomcat部署JavaEE应用的方式



- 方式一：webapps目录：
 - 放入JavaEE应用的目录
 - 放war文件
- 方式二：修改server.xml文件
- 方式三：修改context.xml文件



Tomcat中部署的JavaEE应用程序



war文件



- 同jar的作用类似，将JavaEE应用的资源打包成单独的war文件，方便迁移
- Tomcat会自动提取webapps目录下的war文件解压并部署
- 打包的方法
 - `.../mysite> jar.exe -cvf mysite.war *`
 - 使用eclipse



修改server.xml文件



- 在server.xml中指
 - 打开server.xml文件，定在Host标签内新建Context
`<Context path= "/ myapp "
reloadable="true" docBase="D:\myapp"
workDir="D:\myapp\work"/>`其中path是应用虚拟路径，docBase是应用的物理路径，workDir是应用的工作目录
 - 注：删除一个应用同时删除server.xml中相应的Context节点



修改context.xml文件



- 创建一个context文件
 - 直接在Tomcat的conf\catalina\localhost目录下新建context.xml文件
 - 注：删除一个Web应用同时删除conf\catalina\localhost目录下相应的context.xml文件



Web服务器默认的Web应用



- Web服务器通常会维护一个默认的Web应用，用来处理非用户创建的Web应用程序的请求
 - Tomcat的默认应用：webapps\ROOT
 - 默认的Web应用通常用于测试单个Servlet、JSP或静态内容
 - 访问：http://localhost:8080/



小结



- JavaEE应用程序结构
 - 部署描述符
- 部署JavaEE应用





实验



- 完成实验



本讲结束

- 谢谢大家

