java.net

Java程序设计

第01-02讲 Servlet模型（一）

Java课程组

知识回顾/本讲先行知识

- JavaWeb开发环境搭建
  - Web应用范围以及执行流程
  - Tomcat的安装，配置和部署
  - Eclipse搭建JavaEE开发环境

本讲教学目标

- 理解Servlet的编写和访问过程
- 掌握Servlet父子类之间的继承关系
- 理解Servlet处理、解析请求以及返回响应的过程
- Servlet页面跳转

本讲内容

- Servlet的编写、访问过程
- Servlet中的请求与响应
- Servlet中请求数据的传递、请求转发与请求重定向

什么是Servlet?

- Servlet（Server Applet）是Java Servlet的简称，是小服务程序或服务连接器，是用Java编写的服务器端程序，主要功能在于交互式地浏览和修改数据，生成动态Web内容。

编写部署Servlet程序

编写源文件

- 编写类实现(implements) javax.servlet.Servlet接口，在类中实现 javax.servlet.Servlet 接口中的方法

编译类文件

- 通过javac命令编译写好的源文件
- 注意：-cp参数的使用

部署程序

- 将编译生成字节码(.class)文件复制到“%站点%/WEB-INF/classes/”目录，需要注意目录结构
- 配置web.xml中的<servlet>、<servlet-mapping>节点

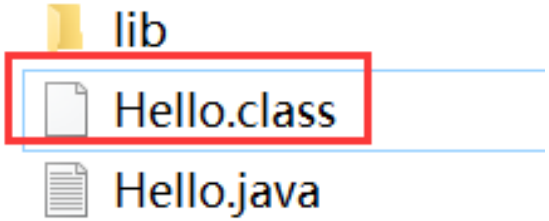
编写部署Servlet程序

- 1. 编写Servlet程序（实现Servlet接口）

```
public class Hello implements Servlet{

    public void destroy() {

    public ServletConfig getServletConfig() {

    public String getServletInfo() {

    public void init(ServletConfig arg0) throws ServletException {

    @Override
    public void service(ServletRequest request, ServletResponse response)
        response.getWriter().println("Hello Servlet");
    }

}
```


编写部署Servlet程序

- 2. 编译Servlet源文件，生成字节码文件

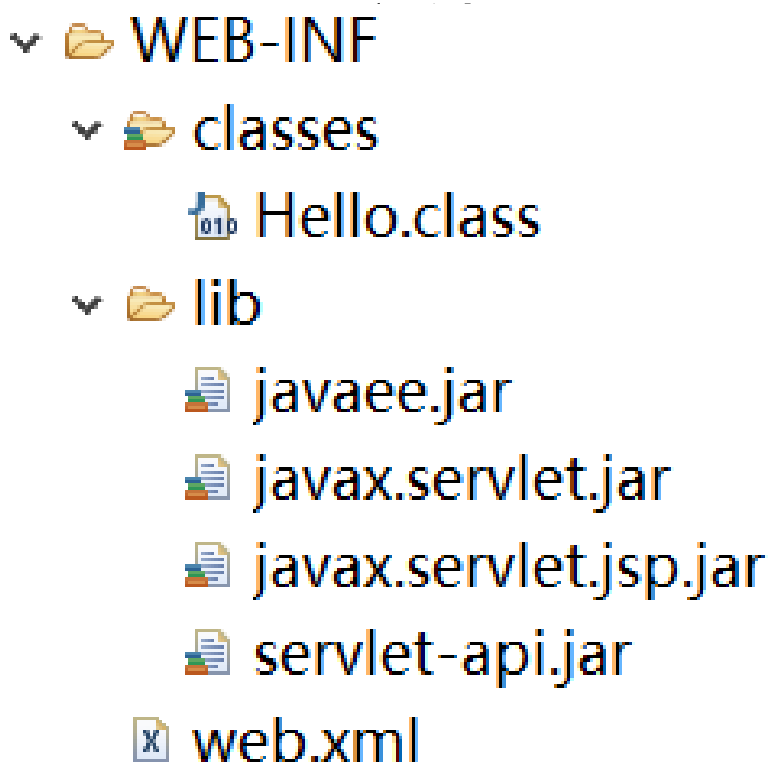


- 注意：
  ① DOS窗口中进入Servlet文件所在目录
  ② -cp参数 导入需要的jar包

```
javac -cp .;lib\* Hello.java
```

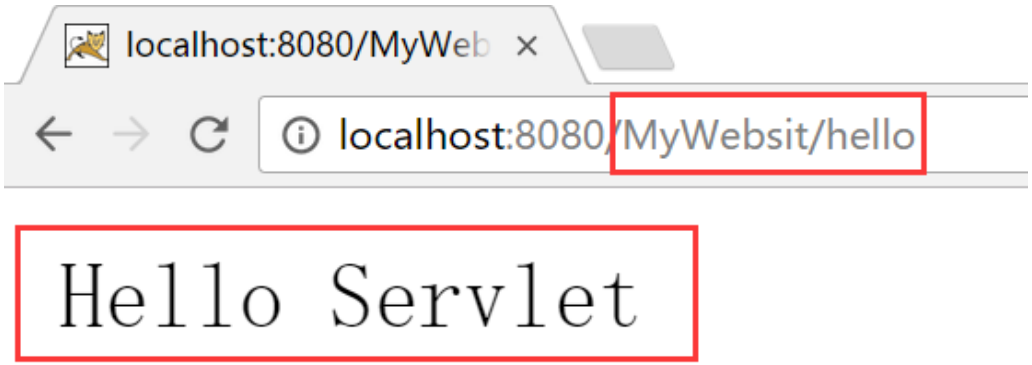
编写部署Servlet程序

- 3. 部署
  - 在Tomcat的webapps目录下创建项目目录
  - 在项目目录下创建WEB-INF目录
  - 在WEB-INF目录下放置w
    classes目录和lib目录
    - classes目录中是步骤2中生
    - lib目录中是用到的jar包



编写部署Servlet程序

- 4. 运行
  - 启动Tomcat
  - 访问Servlet




```
<servlet>
  <servlet-name>Hello</servlet-name>
  <servlet-class>Hello</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>Hello</servlet-name>
  <url-pattern>/hello</url-pattern>
</servlet-mapping>
```


编写部署Servlet程序

- Servlet处理请求，返回响应

http://localhost:8080/MyWebsite/hello

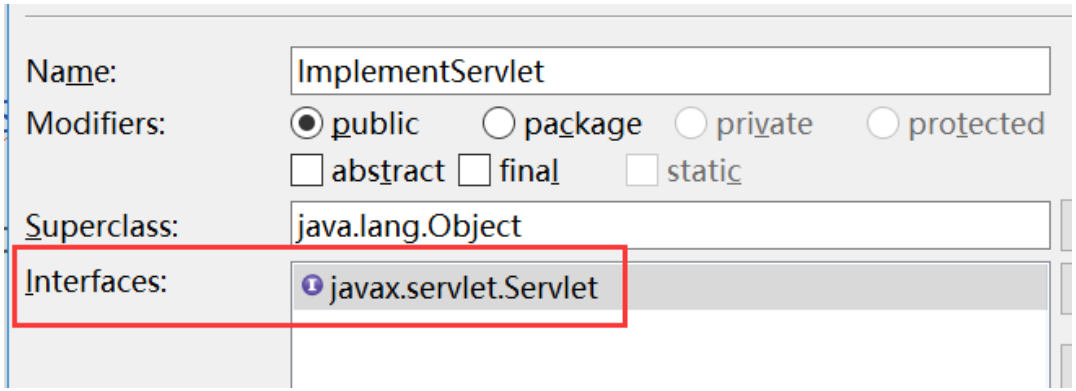
WEB-INF目录下web.xml

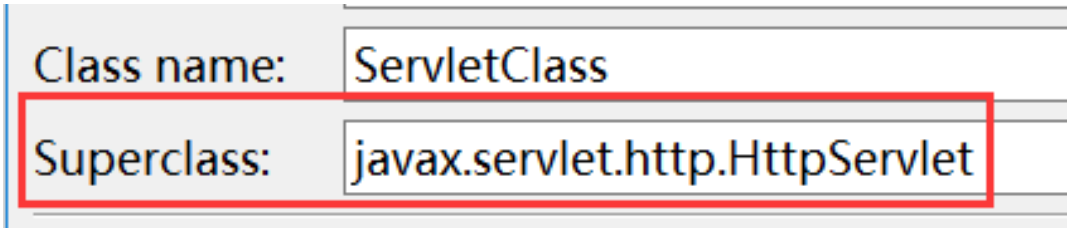
```
<servlet>
    <servlet-name>helloWorld</servlet-name>
    <servlet-class>xx.xx.HelloServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>helloWorld</servlet-name>
    <url-pattern>/hello</url-pattern>
</servlet-mapping>
```

HelloServlet.class → service方法 → response → 客户端的浏览器

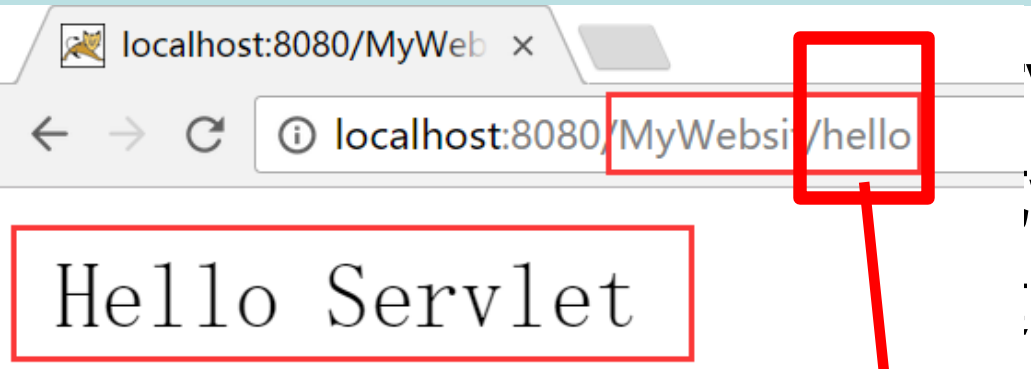
Eclipse开发Servlet

- 新建类，实现Servlet接口
- 新建Servlet（继承HttpServlet）





Eclipse开发Servlet



vlet

web.xml配置文件

Servlet

– 3. 在配置文件中配置Servlet

WEB-INF目录下web.xml

```
<servlet>
    <servlet-name>helloWorld</servlet-name>
    <servlet-class>xx.xx.HelloServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>helloWorld</servlet-name>
    <url-pattern>/hello</url-pattern>
</servlet-mapping>
```

Servlet别名

Servlet类文件

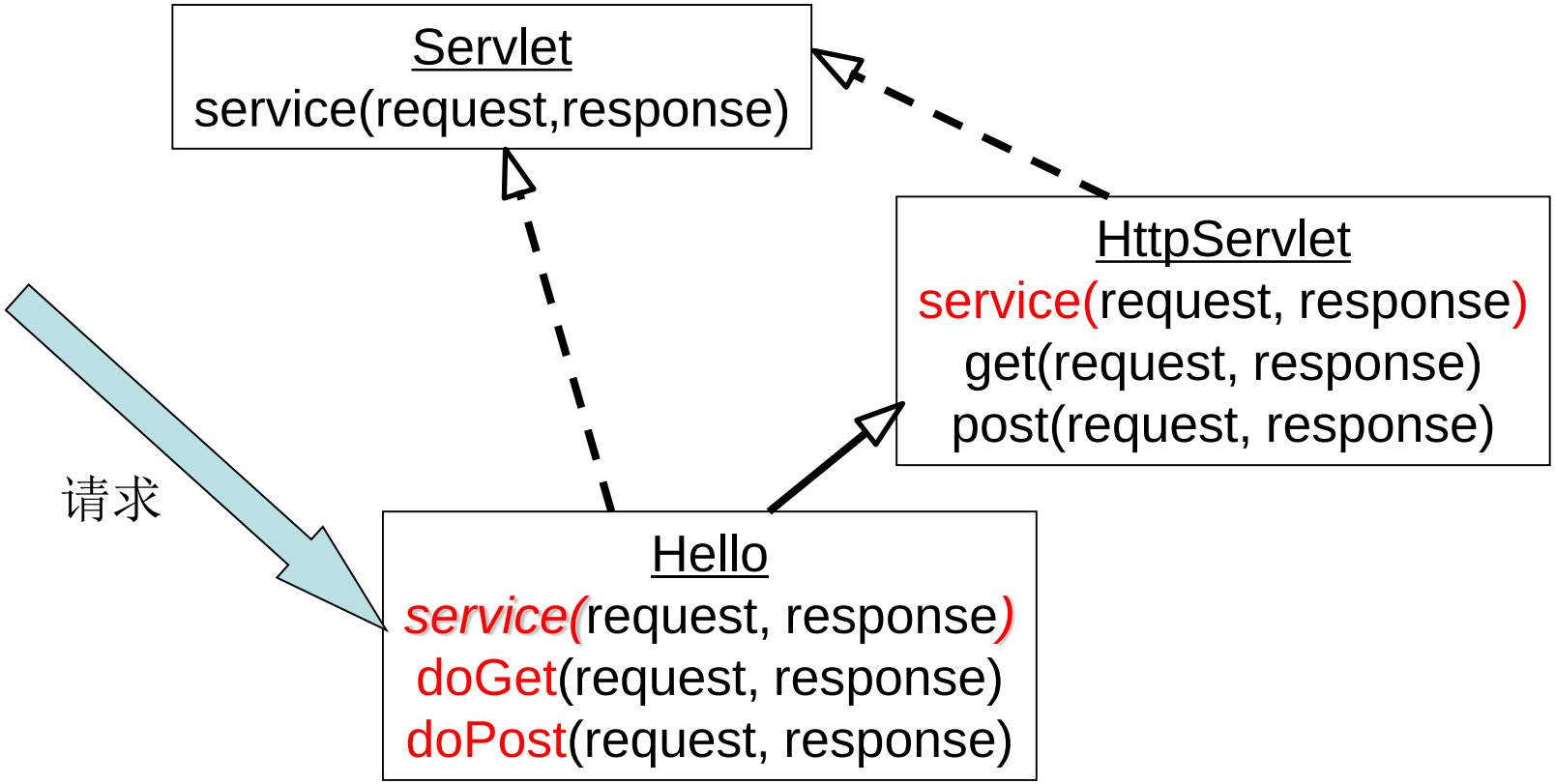
虚拟路径

URL到Servlet的映射

- <url-pattern>标签的匹配方式
  - “完整URL”：如“/HelloServlet”
  - “/目录/*”方式：以“/目录”做为开头的URL
  - “*.ext”：以.ext结尾的地址

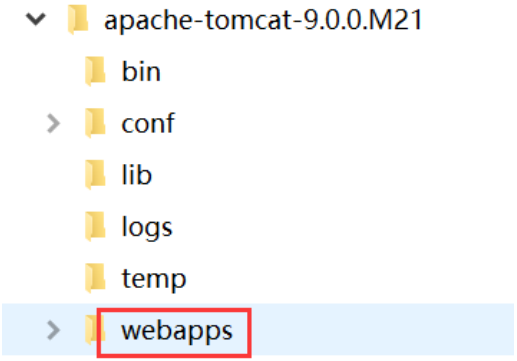
Eclipse开发Servlet

- 实现Servlet接口与继承HttpServlet区别?

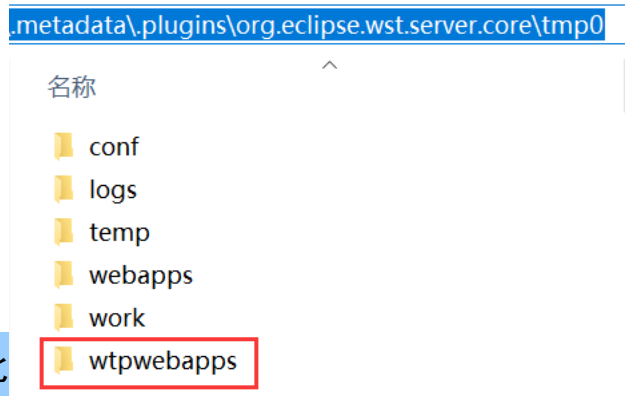


Eclipse开发Servlet

- 部署位置：
  - Tomcat 9.0\apache-tomcat-9.0.0.M21\webapps



- Eclipse编写的工程默认部署位置：
  - Eclipse工作目录\.metadata\.plugins\org.eclipse.wst.server.core\tmp0\wtpwebapps



本讲内容

- Servlet编写、访问的过程
- Servlet中的请求与响应
- Servlet中请求数据的传递、请求转发与请求重定向

接收请求

- Tomcat启动后Web容器首先做的工作

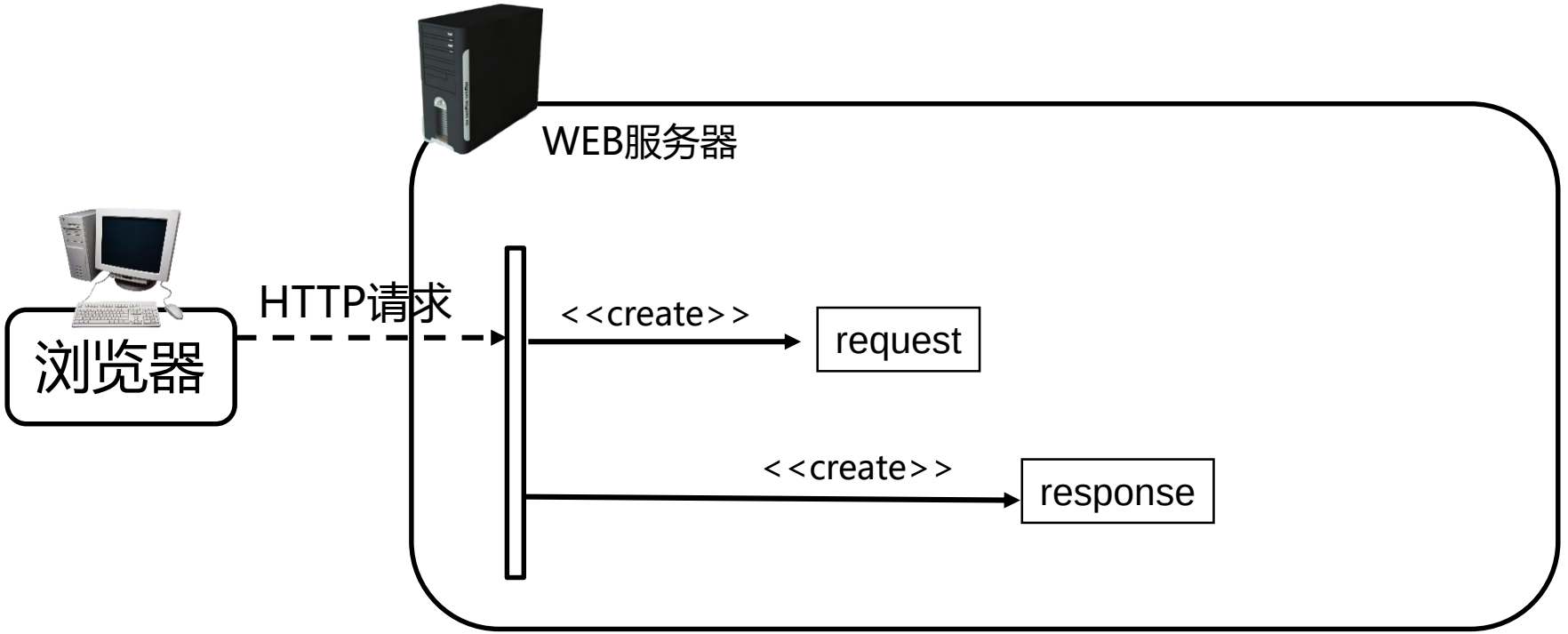
Servlet与URL对应

完成Servlet名称注册

创建Servlet实例

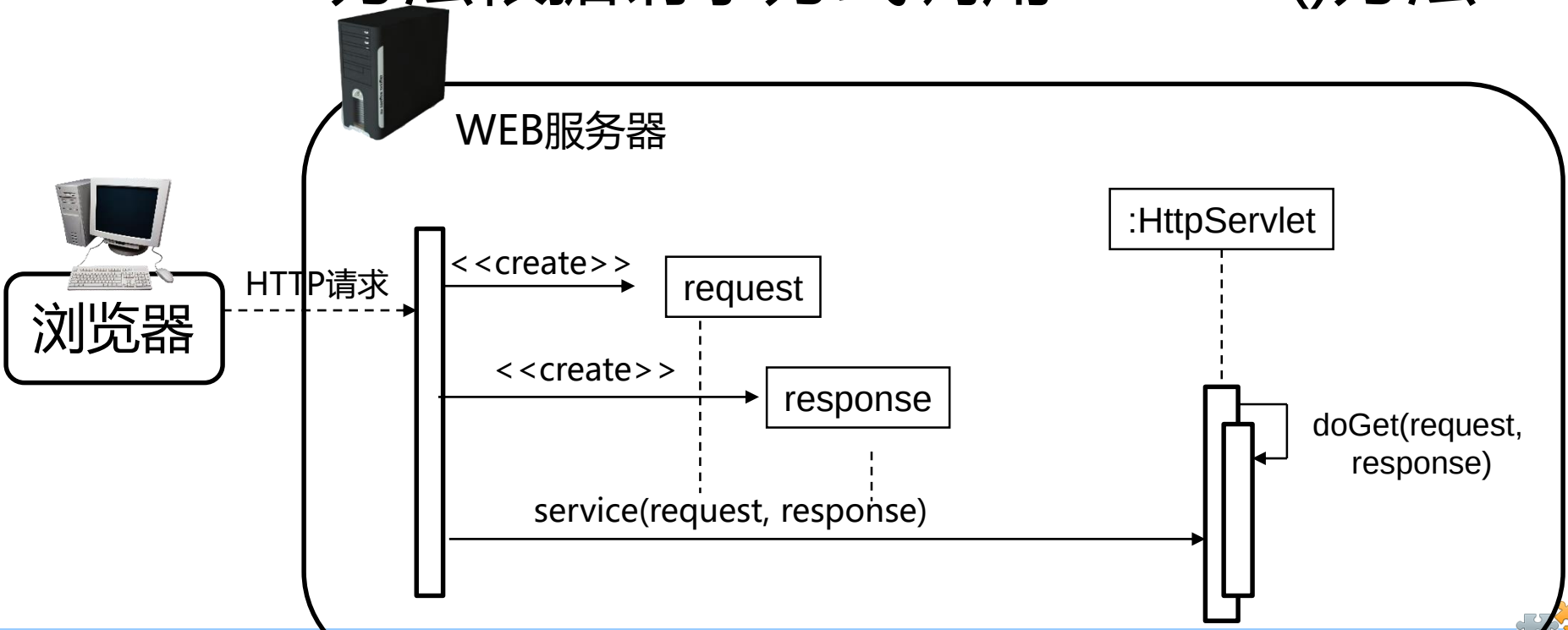
接收请求

- 接收到请求后服务器转交请求给Web容器
- Web容器产生请求对象和响应对象



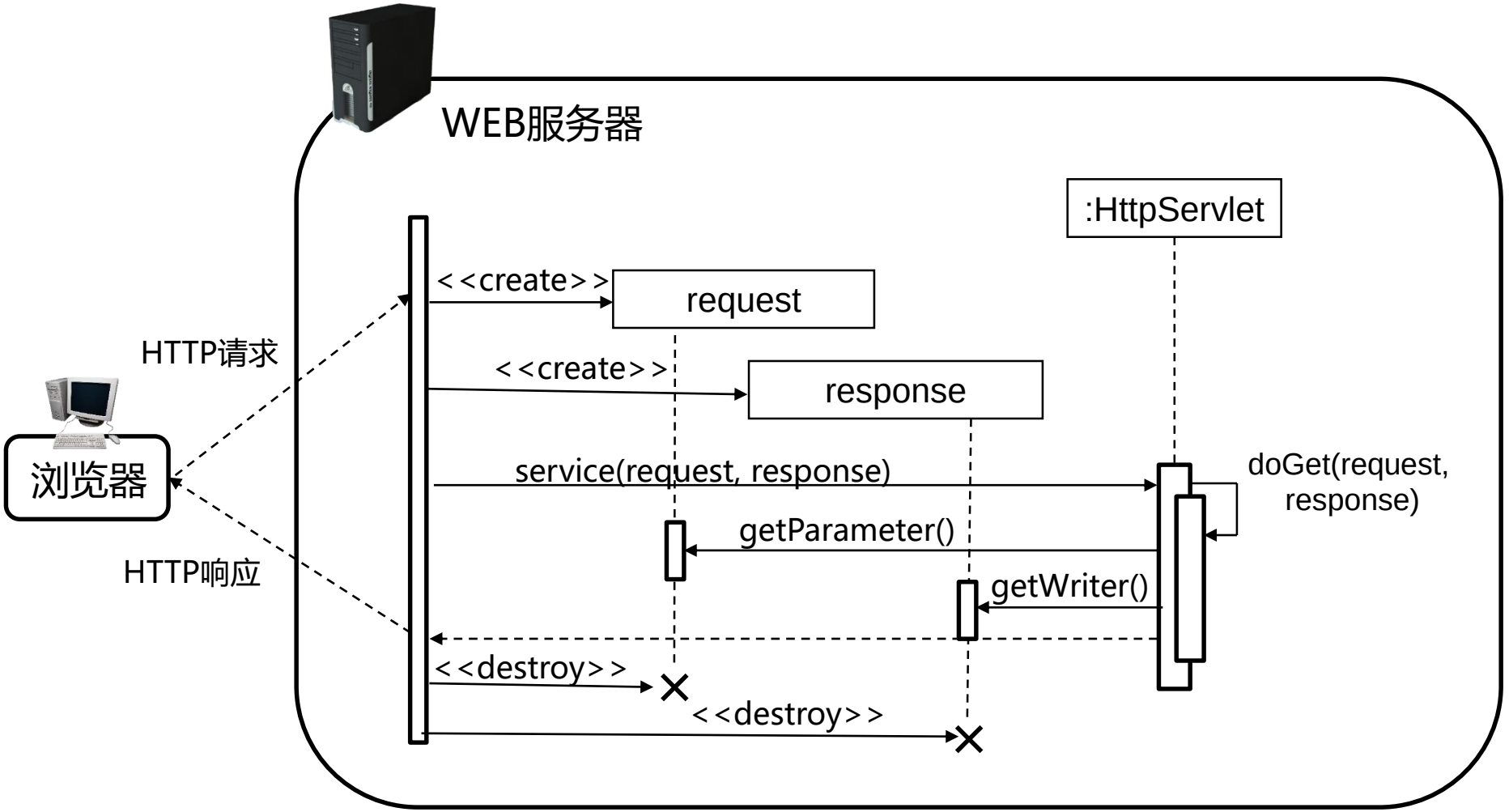
接收请求

- 调用Servlet的service()方法，并将请求和响应对象作为参数传入该方法
- Service方法根据请求方式调用doXXX()方法



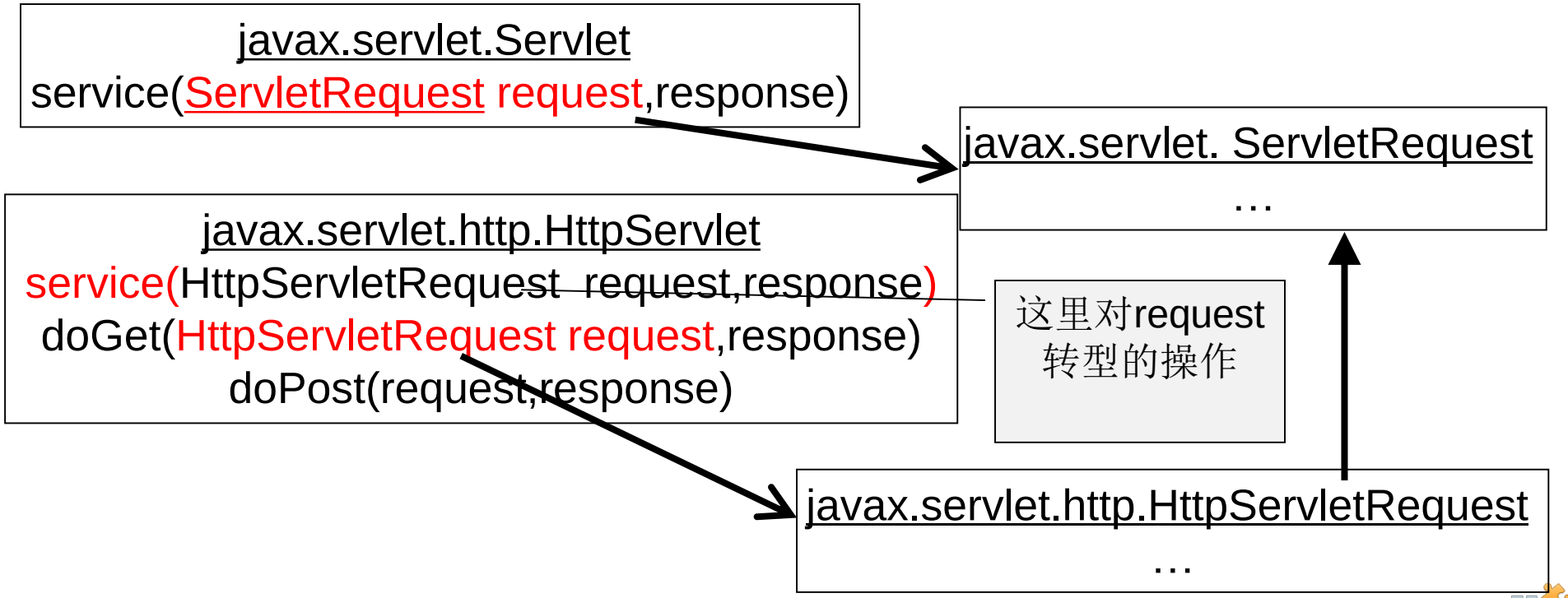
返回响应

- 在doXXX()方法中使用请求对象获取请求信息，响应对象返回响应结果（PrintWriter对象）
- 销毁请求、响应对象


WEB服务器
:HttpServlet
<<create>>
request
<<create>>
response
HTTP请求
浏览器
service(request, response)
doGet(request, response)
getParameter()
getWriter()
HTTP响应
<<destroy>>
<<destroy>>

接收请求

- HttpServletRequest接口以参数的形式传递到service方法里
- protected void doGet(HttpServletRequest request, HttpServletResponse response)



接收请求

- Servlet接口的service方法

```
public void service(ServletRequest req, ServletResponse res)
    throws ServletException, IOException {
    try {
        HttpServletRequest request = (HttpServletRequest) req;
        response = (HttpServletResponse) res;
    } catch (ClassCastException e) {
        HttpServletResponse response;
        throw new ServletException("non-HTTP request or response");
    }
    HttpServletResponse response;
    HttpServletRequest request;
    service(request, response);
}
```

调用重载的service()方法
参数类型是HttpServletRequest
HttpServletResponse

- service()方法根据请求类型调用doXXX()方法
- 所以必须重写doXXX()方法

解析请求

- getParameter()方法
  - href="RegistServlet?userName=zs"
  - 表单中的name属性值的获取

```
<body>
    <a href="hello?userName=lww">link</a>
</body>
```

```
<form action="hello">
    <input type="text" name="text" value="information">
    <input type="submit" value="submit">
</form>
```

```
request.getParameter("userName")
request.getParameter("text")
```


解析请求

- getParameter()方法
  - href="RegistServlet?userName=zs"
  - 表单中的name属性值的获取
- Attribute参数
  - setAttribute()： 设置参数
  - getAttribute()： 获取某个参数的值

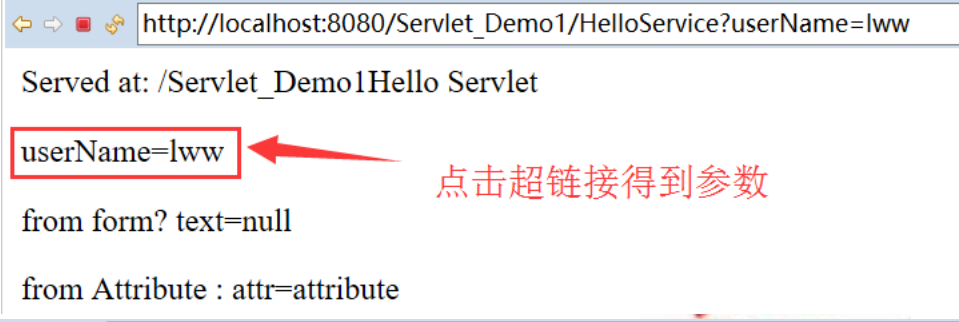
```
request.setAttribute("attr", "attribute");
request.getAttribute("attr")
```

返回响应

- HttpServletResponse接口
- protected void doGet(HttpServletRequest request, HttpServletResponse response)
  - getWriter()方法，返回的是一个输出流
  - response.setContentType("text/html");
  - 编码
    - response.setCharacterEncoding("UTF-8");
  - response.setContentType ("text/html;charset=GB2312");

例子

```
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <a href="HelloService?userName=lw
    <form action="HelloService">
        <input type="text" name="text
        <input type="submit" value="submit">
    </form>
</body>
</html>
```




```
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
    response.getWriter().append("Served at: ").append(request.getContextPath());
    response.setContentType ("text/html;charset=GB2312");

    request.setAttribute("attr", "attribute");
    PrintWriter out = response.getWriter();
    out.println("Hello Servlet");
    out.print("<p>userName=" + request.getParameter("userName") + "</p>");
    out.print("<p>from form:  text=" + request.getParameter("text") + "</p>");
    out.print("<p>from Attribute : attr=" + request.getAttribute("attr") + "</p>");
}
```

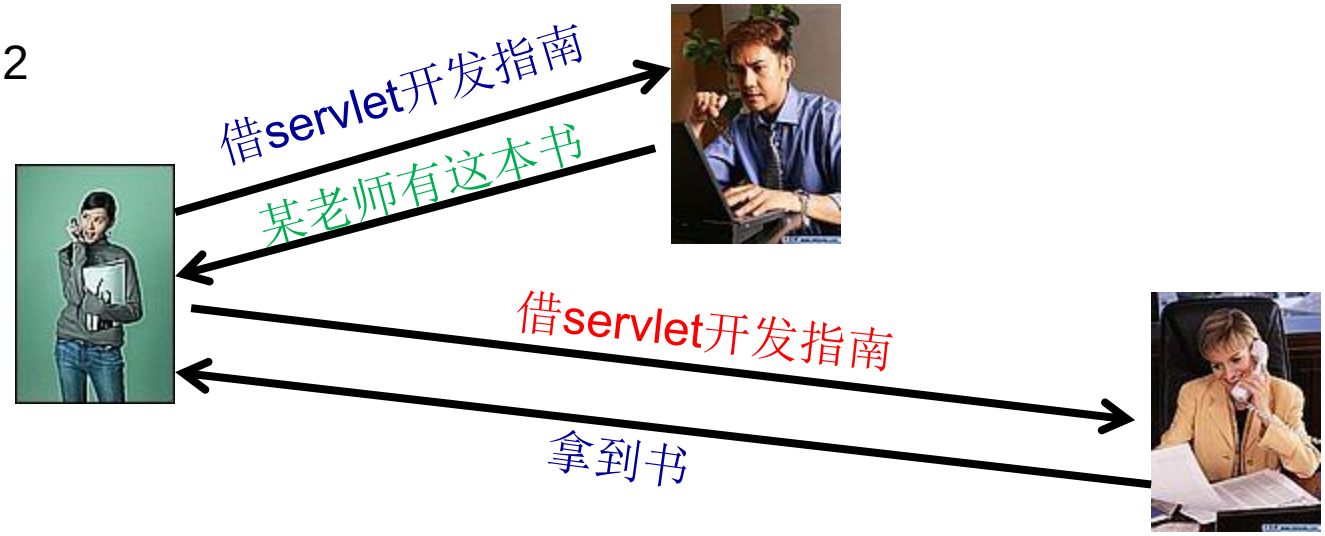
本讲内容

- Servlet编写、访问的过程
- Servlet中的请求与响应
- Servlet中请求数据的传递、请求转发与请求重定向

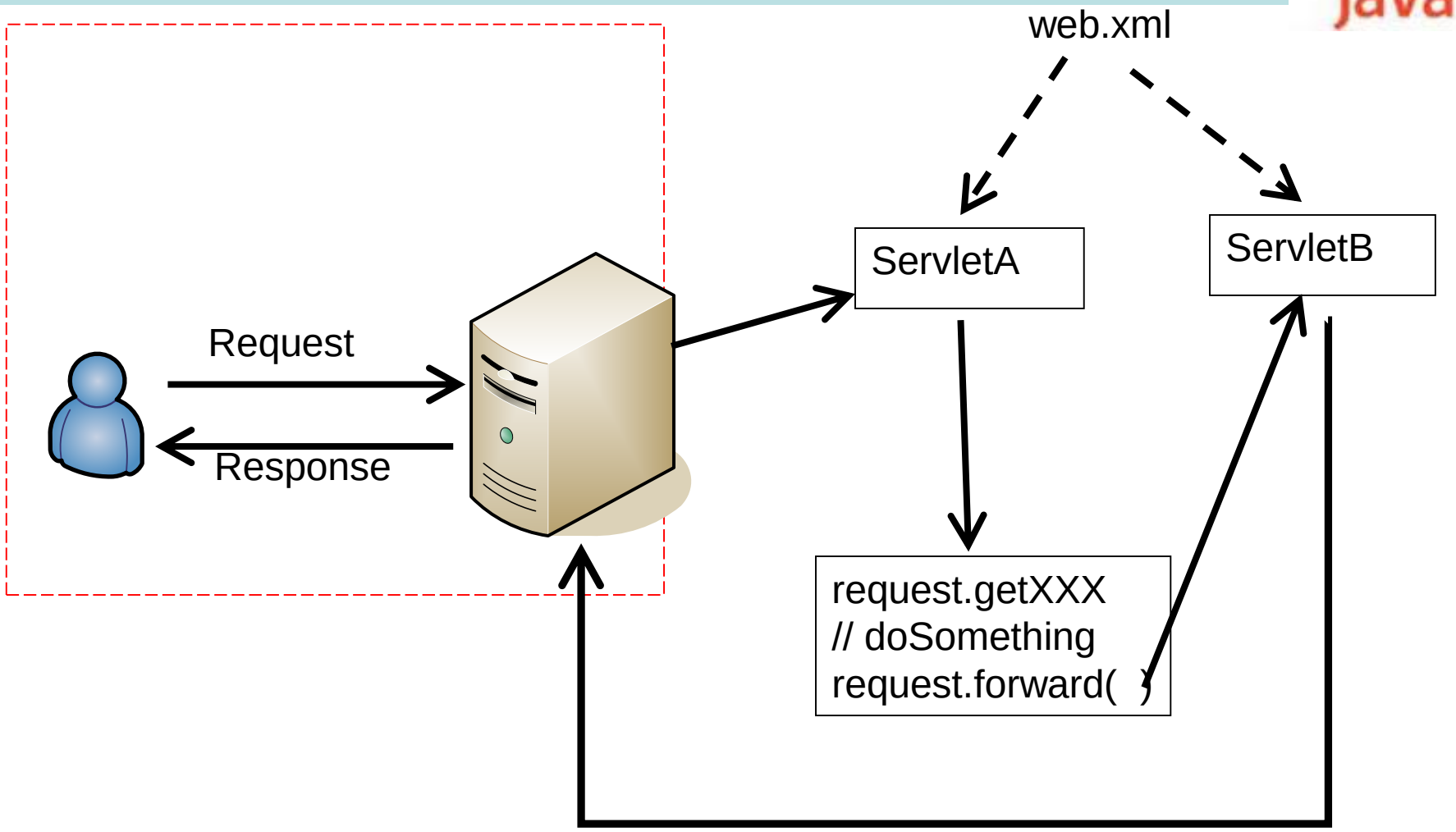
借书的故事

场景1

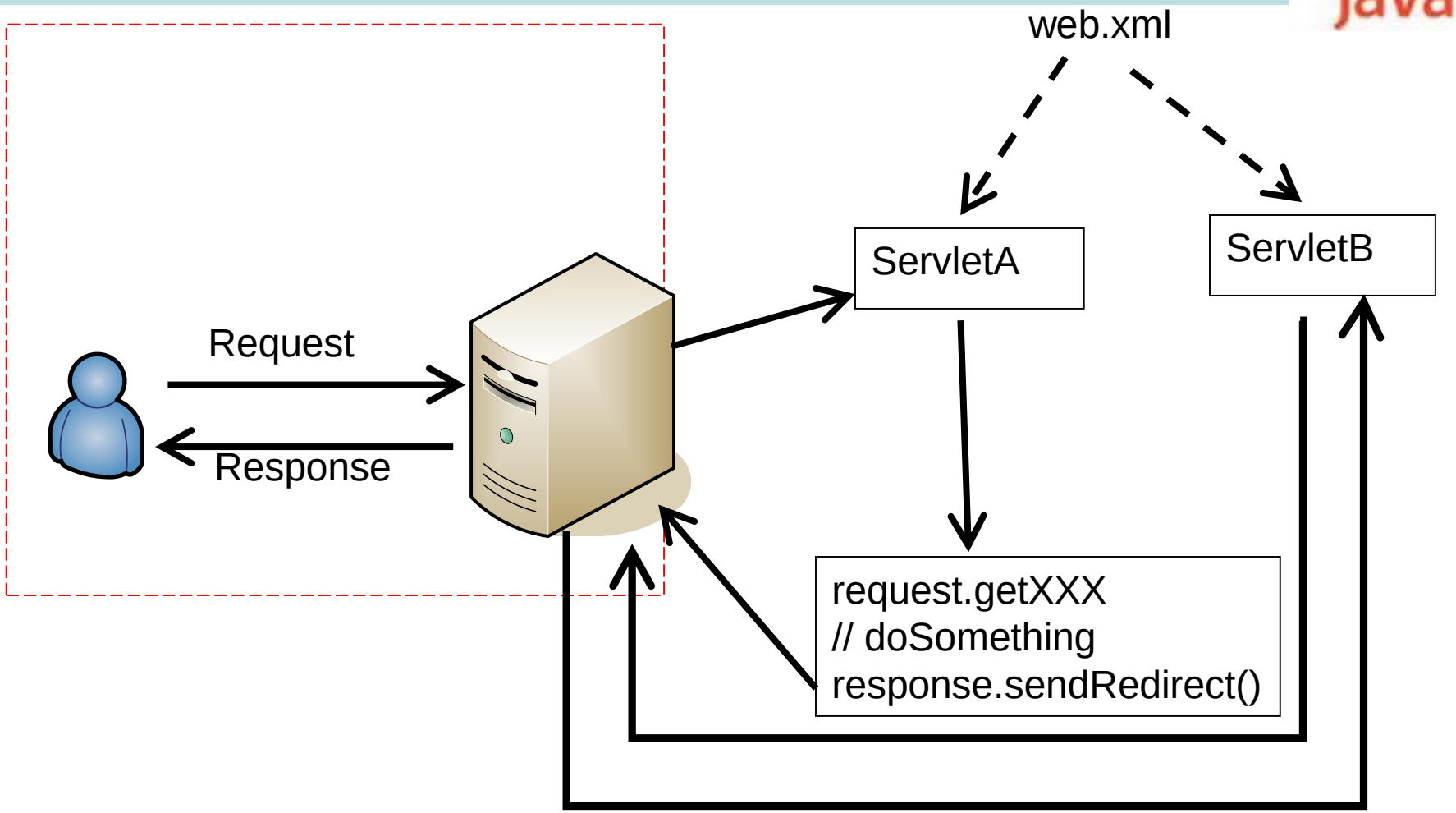
场景2



Servlet应用执行过程——转发请求



Servlet应用执行过程——重定向



请求重定向、请求转发

- 请求转发
  - request.getRequestDispatcher("*targetURL*").forward(request, response);
- 请求重定向
  - response.sendRedirect("*targetURL*")

请求数据的传递

- 请求数据的传递
  - request.setAttribute(String,Object)
  - request.getAttribute(String),返回Object
- 注意：
  - 请求转发可以共享参数
  - 请求重定向无法共享参数

请求转发、请求重定向区别

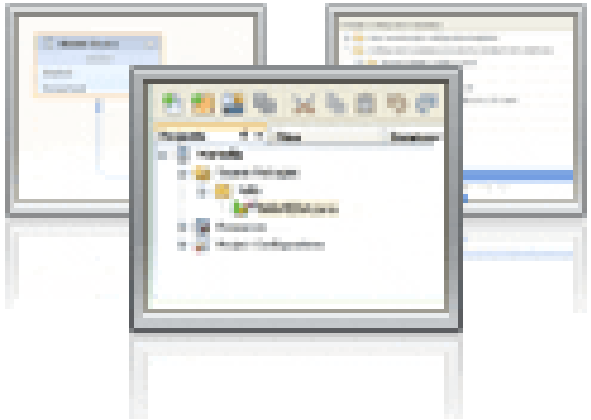
- 本质
  - 请求转发：一次请求
  - 请求重定向：两次请求
- 速度
  - 请求转发 > 页面跳转
- Attribute参数
  - 请求转发可共享参数
  - 请求重定向不可共享参数

小结

- Servlet编写、访问的过程
- Servlet中的请求与响应
- Servlet中请求数据的传递、请求转发与请求重定向

实验

- 将数据库中的数据展示在页面上
  - 如：将poem数据库中dynasty表中的所有记录展示在页面上
- 实现简单的页面登录的功能
  - 参照：
    - 《Servlet模型(一)_实验手册.doc》实验五



本讲结束

- 谢谢大家