# Servlet_JSP

@M了个J
李明杰

https://github.com/CoderMJLee

https://space.bilibili.com/325538782

小码哥教育 SEEMYGO

实力IT教育 www.520it.com

# Servlet

- Servlet是Server Applet的简称，译为"小型的服务程序"，用于响应客户端的请求

- 一般的使用要素
  - 继承javax.servlet.http.HttpServlet，实现doGet、doPost或service方法
  - 通过request对象获取客户端的请求数据
    - ✓ request.getParameter()
  - 通过response对象给客户端返回响应
    - ✓ response.getWriter().write()
  - 通过注解@WebServlet设置Servlet对应的请求路径

# 乱码问题解决

- 客户端的请求数据乱码
  - `request.setCharacterEncoding("UTF-8");`

- 服务器的响应数据乱码
  - `response.setContentType("text/plain;charset=UTF-8");`
  - 其中"text/plain"是数据的MIMEType，根据实际情况而定
  - 更多的MIMEType可以参考TOMCAT_HOME/conf/web.xml

# 一些细节

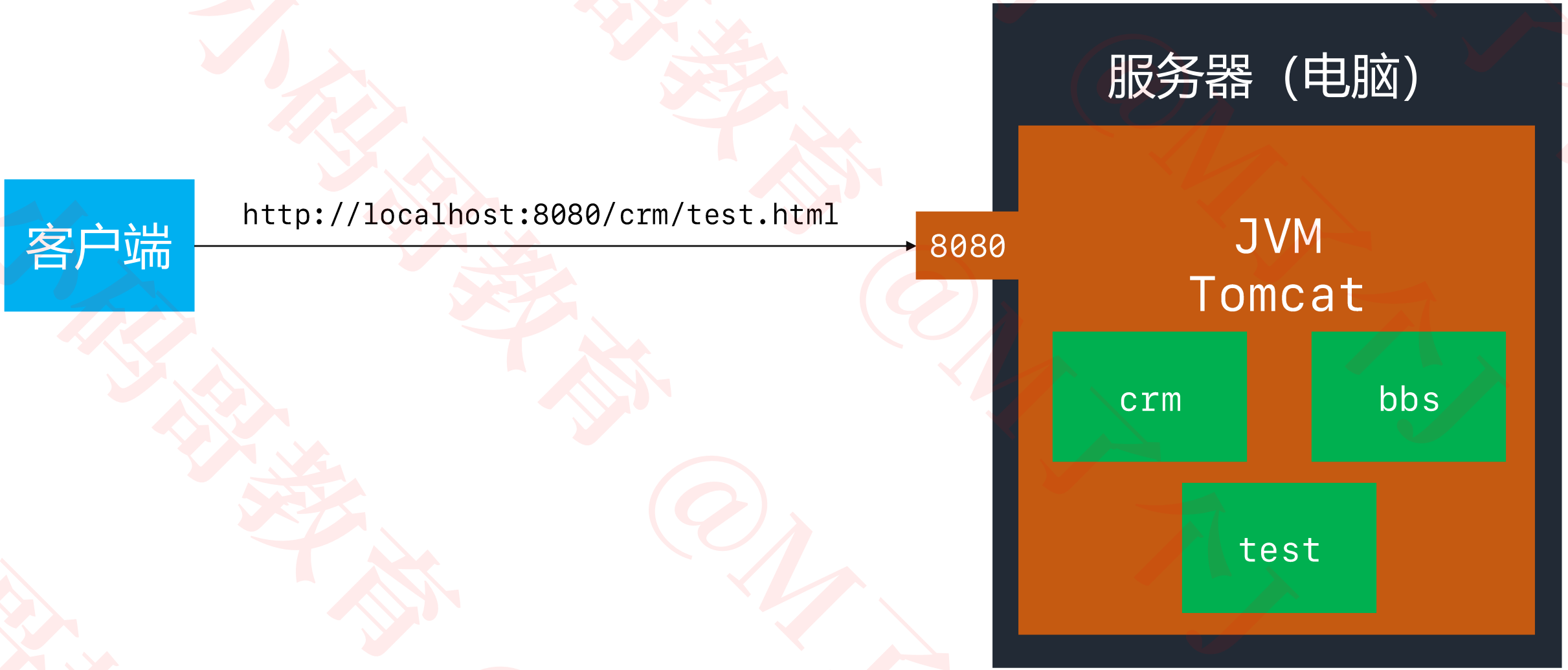
- 通过response拿到的输出流对象（比如getWriter），不需要程序员调用close去关闭

- 默认情况下，一个Servlet类，只会被服务器创建一个实例对象，而且是在第一次处理客户端请求才创建实例
  - 注意：Servlet并没有设计成单例模式！！！
  - 建议：不要在Servlet中定义可写（writable）的成员变量，会引发线程安全问题

- HTTP请求的默认端口号是80，所以，如果Tomcat服务器的端口号设置为80，那么
  - localhost/path等价于localhost:80/path

# Servlet处理请求的常见过程


- 如果要返回一个网页给客户端，会存在大量可读性差、难以维护的字符串拼接代码



# JSP

■ JSP是JavaServer Pages的简称，是一种动态网页技术标准

■ 指令

□ <%@ page %>：配置当前页面信息

□ <%@ include %>：包含其他页面

□ <%@ taglib %>：导入标签库

■ 嵌入Java代码

□ <% Java代码 %>

■ 声明

□ <%! 声明成员变量、方法 %>

■ 输出

□ <%= 需要输出的内容 %>

□ 等价于out.print(需要输出的内容)

■ 注释

□ <%-- 注释内容 --%>

□ HTML、CSS、JS注释照常使用

# EL表达式、JSTL标签库

- EL是Expression Language的简称
  - ${obj.property}、${obj["property"]}、${obj[propertyVar]}
  - empty、not empty

- JSTL是JSP Standard Tag Library的简称，译为"JSP标准标签库"，由Apache的Jakarta小组维护

- EL表达式、JSTL标签库都可以简化JSP代码

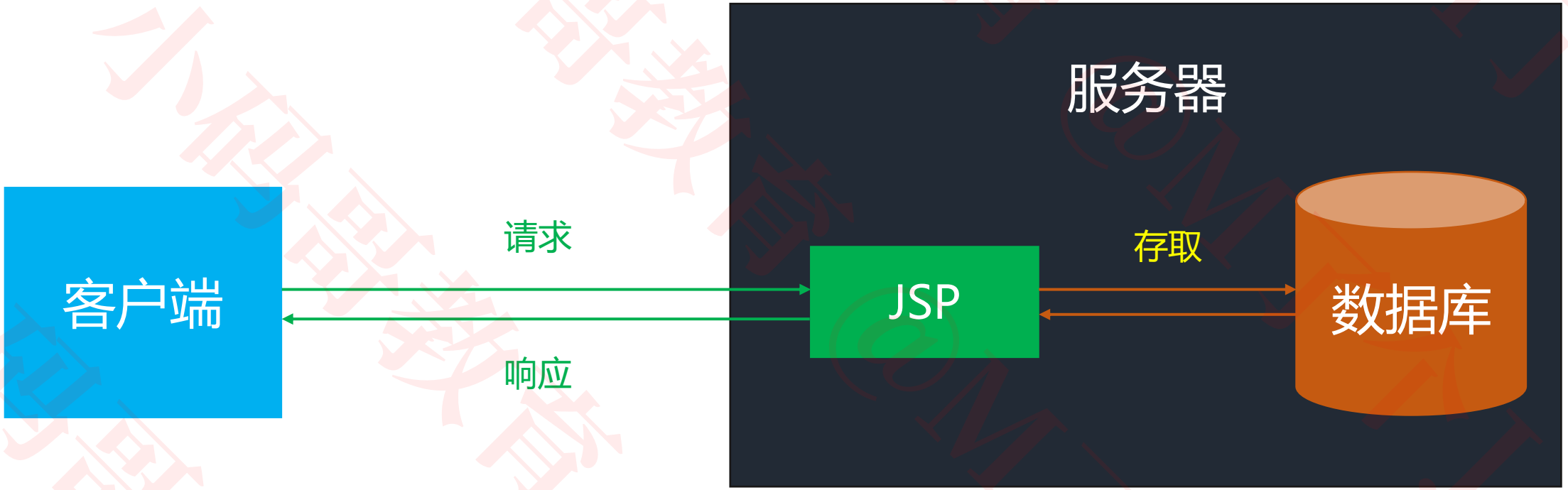
# 下载的使用JSTL核心标签库

- 下载地址：http://tomcat.apache.org/download-taglibs.cgi
  - 如果使用其核心标签库，只需要下载2个jar包即可
    - ✓ taglibs-standard-impl-1.2.5.jar、taglibs-standard-spec-1.2.5.jar

- 将jar包加入到WEB-INF/lib中，并且右击选择"Add as Library"
- 使用taglib指令导入JSTL核心标签库
  - <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

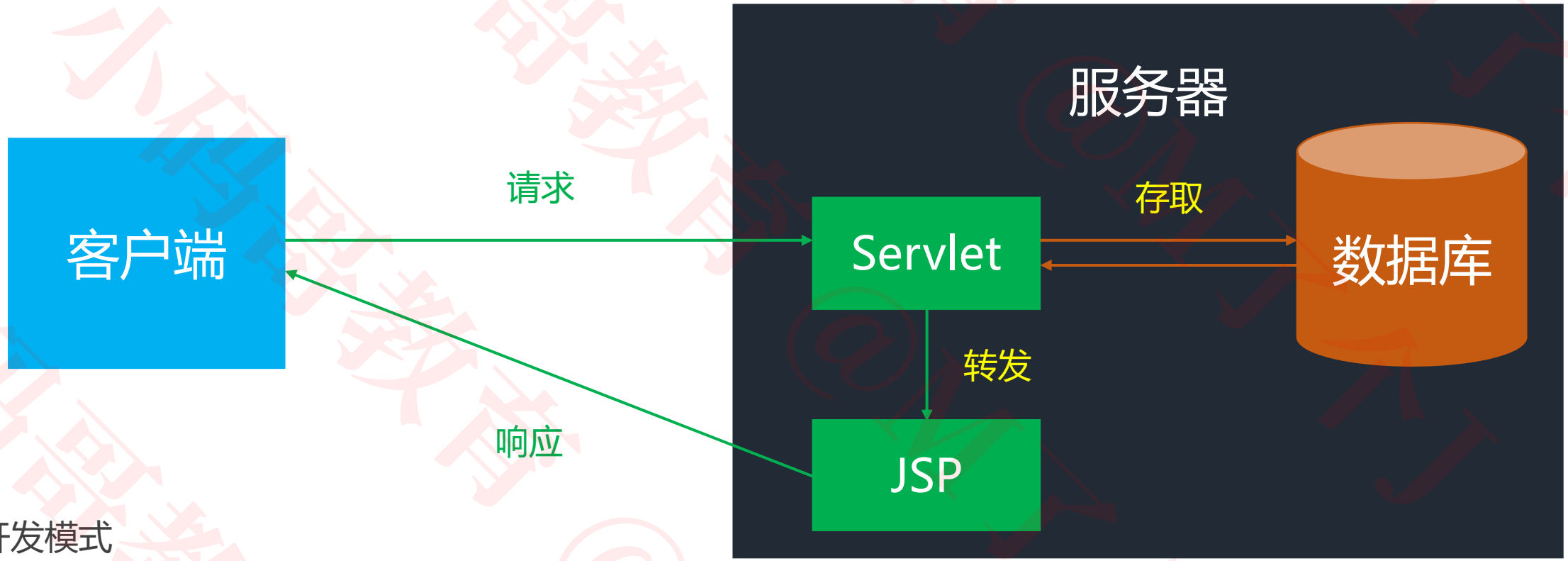
- 常用标签
  - <c:if test="条件">
  - <c:forEach items="集合" var="元素" varStatus="循环相关的信息">
  - <c:choose>、<c:when test="条件">、<c:otherwise>

# JSP处理请求的常见过程




- 明显的缺点
  - Java代码和HTML代码混合在一起，导致可读性差、难以维护
  - 整个JSP文件基本得由Java后台工程师来完成，前端工程师根本无法维护
- ✓ 在很久以前，有些公司是需要前端后台一起开发JSP。现在比较流行前后端分离

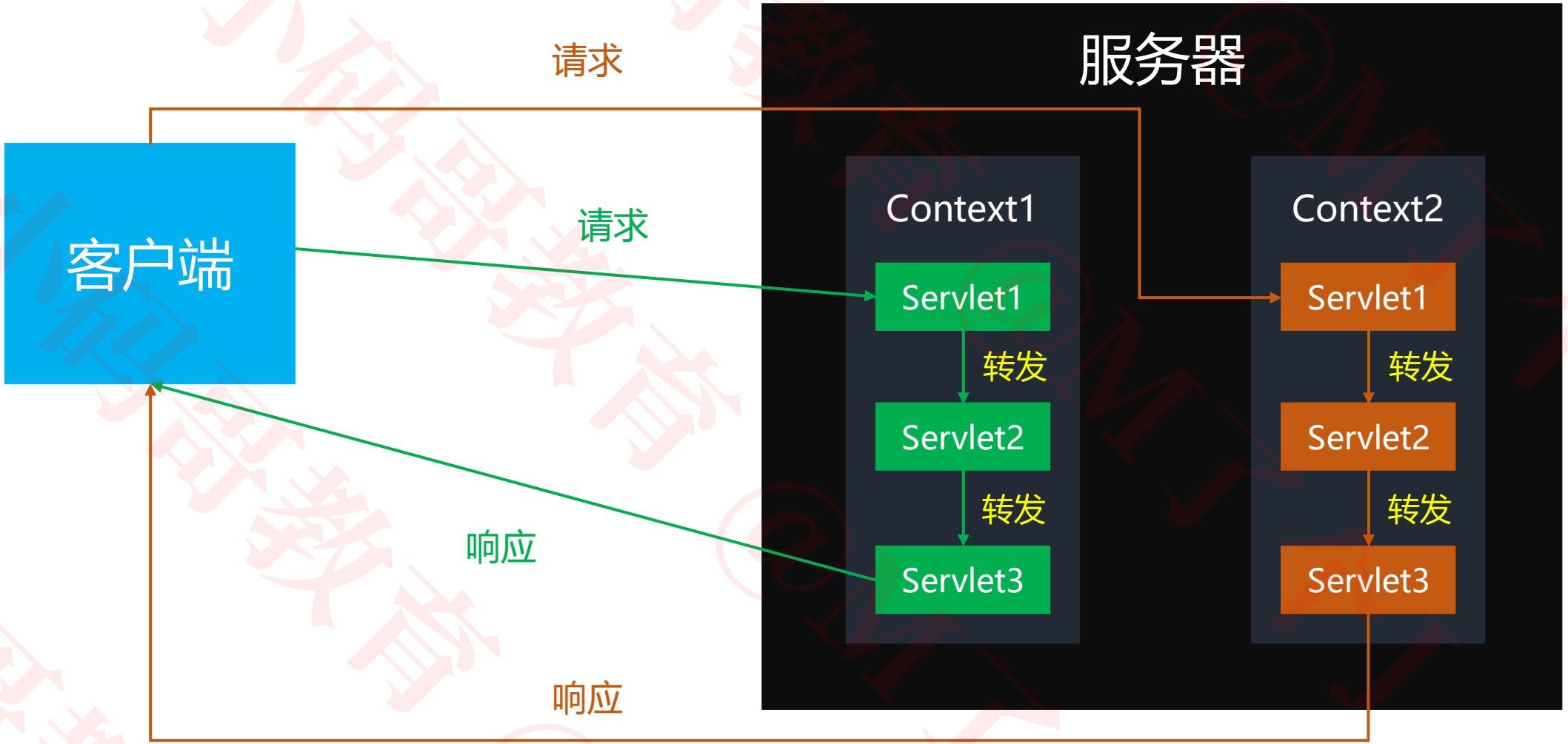
# Servlet + JSP处理请求的常见过程




- MVC开发模式
  - C (Controller) : 控制器 (Servlet)
  - M (Model) : 数据
  - V (View) : 页面展示 (JSP)

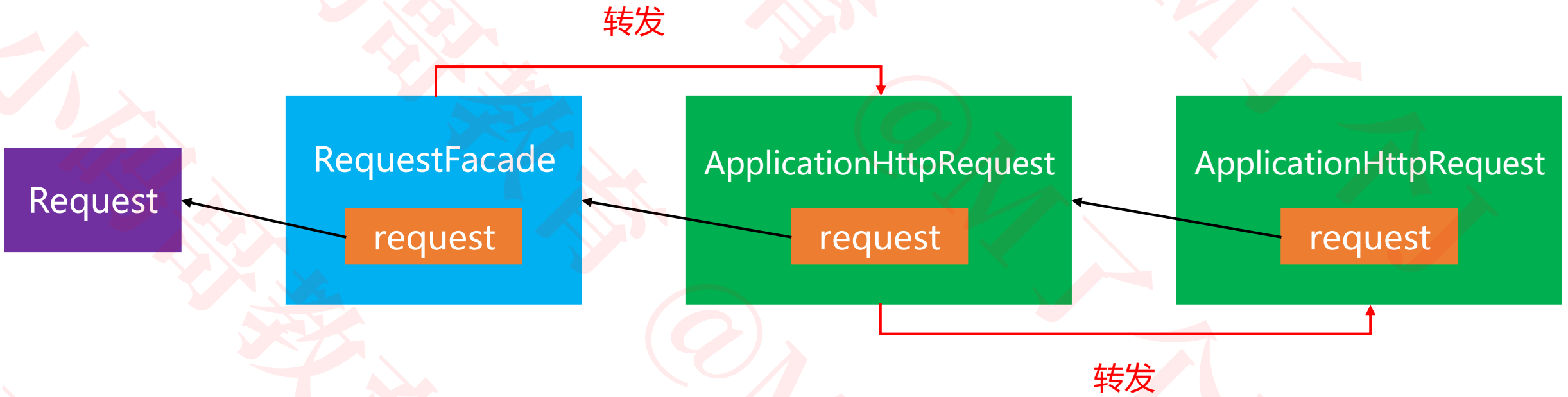
# 转发（forward）

- 在同一个Context中进行请求转发



# 转发链条

- 在同一次请求中，可以转发多次，形成一个转发链条。在一个转发链条上
  - 可以通过request.setAttribute、request.getArrribute来共享数据
  - 每一次转发都会创建一个新的request对象，用成员变量request指向前一个request对象




- 在转发链条上，所有的attribute都存储在头部的Request对象中

# 重定向（redirect）

- 重定向：服务器通知客户端重新发送请求到新的任意URL地址

# 转发 vs 重定向

- 转发代码：`request.getRequestDispatcher("/路径").forward(request, response)`
  - 只能转发到同一个Context下，路径中不用包含ContextPath
  - 客户端只发了一次请求
  - 浏览器地址栏的URL不会发生变化
  - 转发的操作只由服务器完成

- 重定向代码：`response.sendRedirect("/路径")`
  - 可以重定向到任意URL，如果重定向到同一个Context下，路径中需要包含ContextPath
  - 客户端发了两次请求
  - 浏览器地址栏的URL会发生变化
  - 重定向的操作由服务器+客户端配合完成