2、响应状态码

200：请求响应成功 200

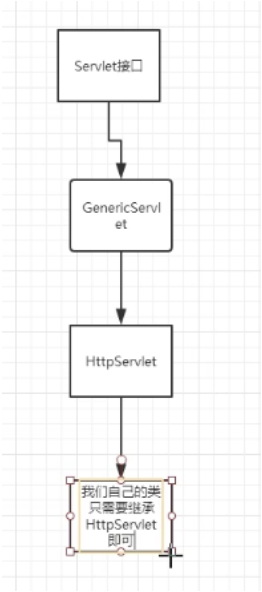
3xx：请求重定向

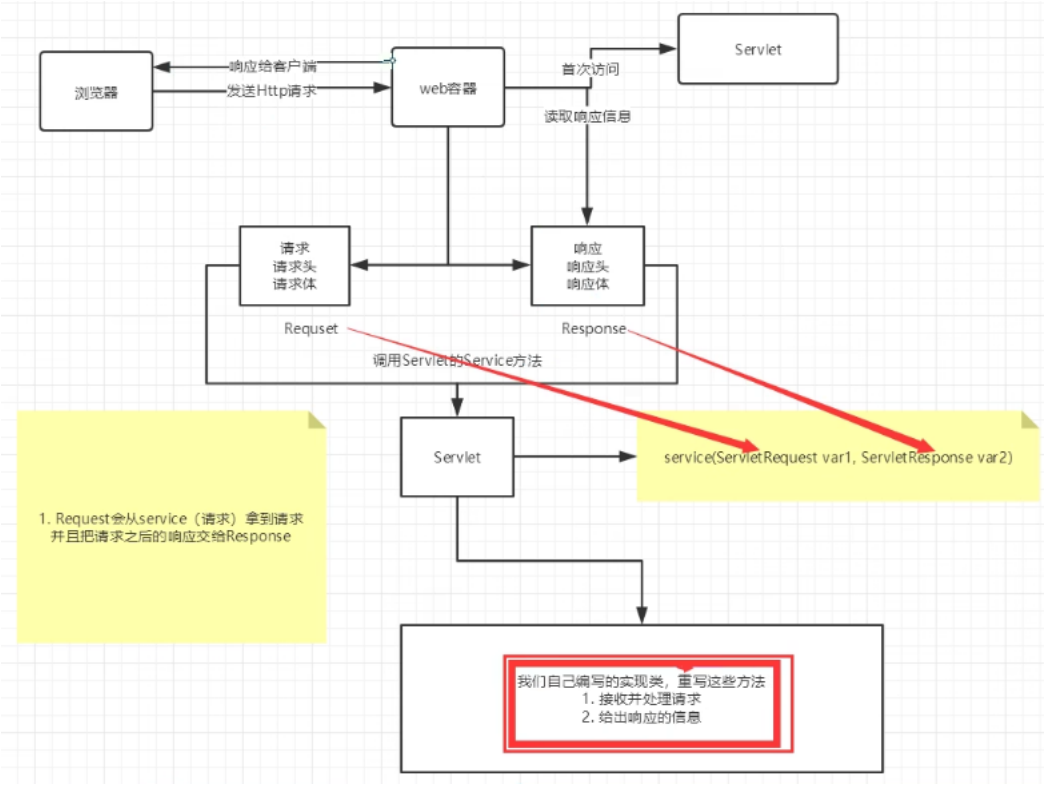
- 重定向：你重新到我给你新位置去；

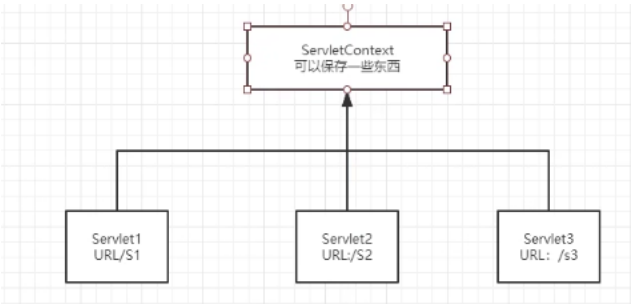
4xx：找不到资源 404

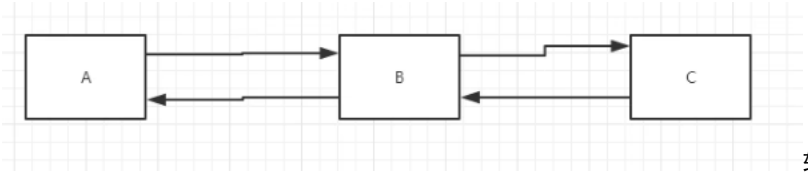
- 资源不存在；

5xx：服务器代码错误 500 502:网关错误









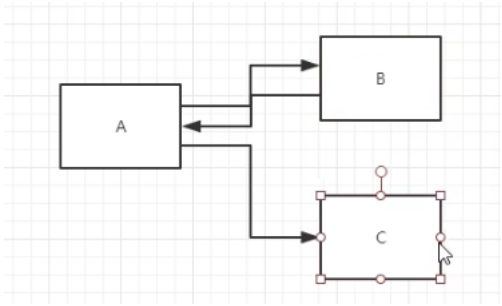
转发

6.6、HttpServletResponse

web服务器接收到客户端的http请求，针对这个请求，分别创建一个代表请求的HttpServletRequest对象，代表响应的一个HttpServletResponse；

- 如果要获取客户端请求过来的参数：找HttpServletRequest
- 如果要给客户端响应一些信息：找HttpServletResponse

2. 下载文件
    1. 要获取下载文件的路径
    2. 下载的文件名是啥?
    3. 设置想办法让浏览器能够支持下载我们需要的东西
    4. 获取下载文件的输入流
    5. 创建缓冲区
    6. 获取OutputStream对象
    7. 将FileOutputStream流写入到buffer缓冲区
    8. 使用OutputStream将缓冲区中的数据输出到客户端!



重　　定　　向

B一个web资源收到客户端A请求后，B他会通知A客户端去访问另外一个web资源C，这个过程叫重定向

面试题：请你聊聊重定向和转发的区别？

相同点

- 页面都会实现跳转

不同点

- 请求转发的时候，url不会产生变化
- 重定向时候，url地址栏会发生变化；

6.7、HttpServletRequest

HttpServletRequest代表客户端的请求，用户通过Http协议访问服务器，HTTP请求中的所有信息会被封装到HttpServletRequest，通过这个HttpServletRequest的方法，获得客户端的所有信息

会话：用户打开一个浏览器，点击了很多超链接，访问多个web资源，关闭浏览器，这个过程可以称之为会话

有状态会话：

你能怎么证明你是西开的学生？

你　　西开

1. 发票　　西开给你发票
2. 学校登记　　西开标记你来过了

一个网站，怎么证明你来过？

客户端　　服务端

1. 服务端给客户端一个 信件，客户端下次访问服务端带上信件就可以了； cookie
2. 服务器登记你来过了，下次你来的时候我来匹配你； seesion

7.2、保存会话的两种技术

cookie

- 客户端技术　（响应，请求）

session

- 服务器技术，利用这个技术，可以保存用户的会话信息？

7.3、Cookie

1. 从请求中拿到cookie信息
2. 服务器响应给客户端cookie

删除Cookie；

- 不设置有效期，关闭浏览器，自动失效；
- 设置有效期时间为 0,；

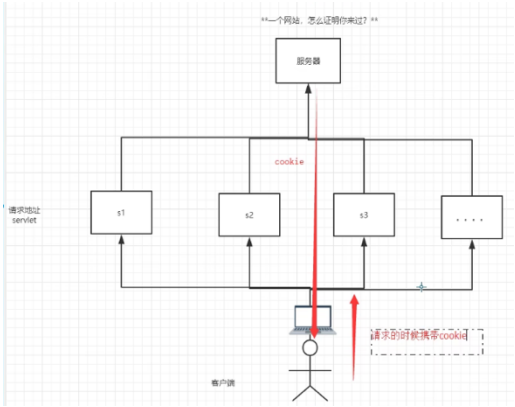
7.4、Session（重点）

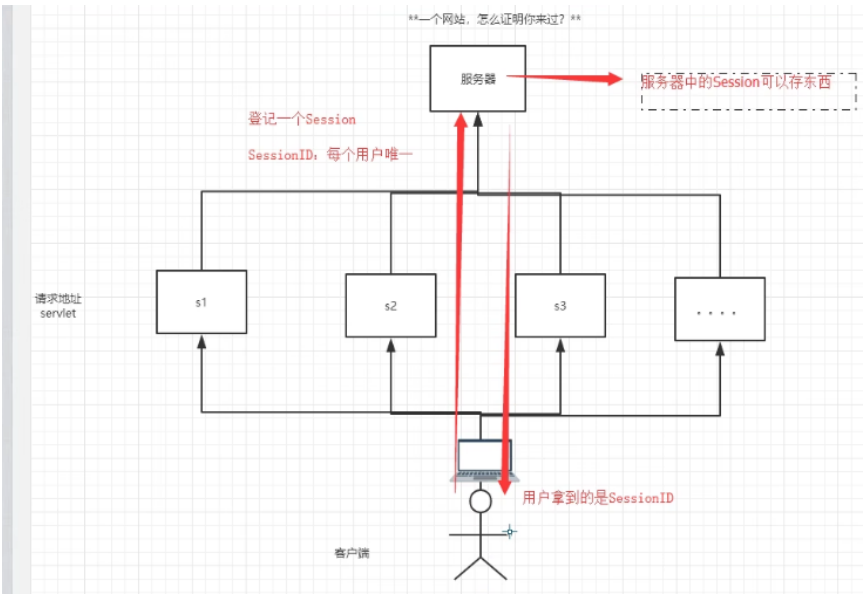
什么是Session：

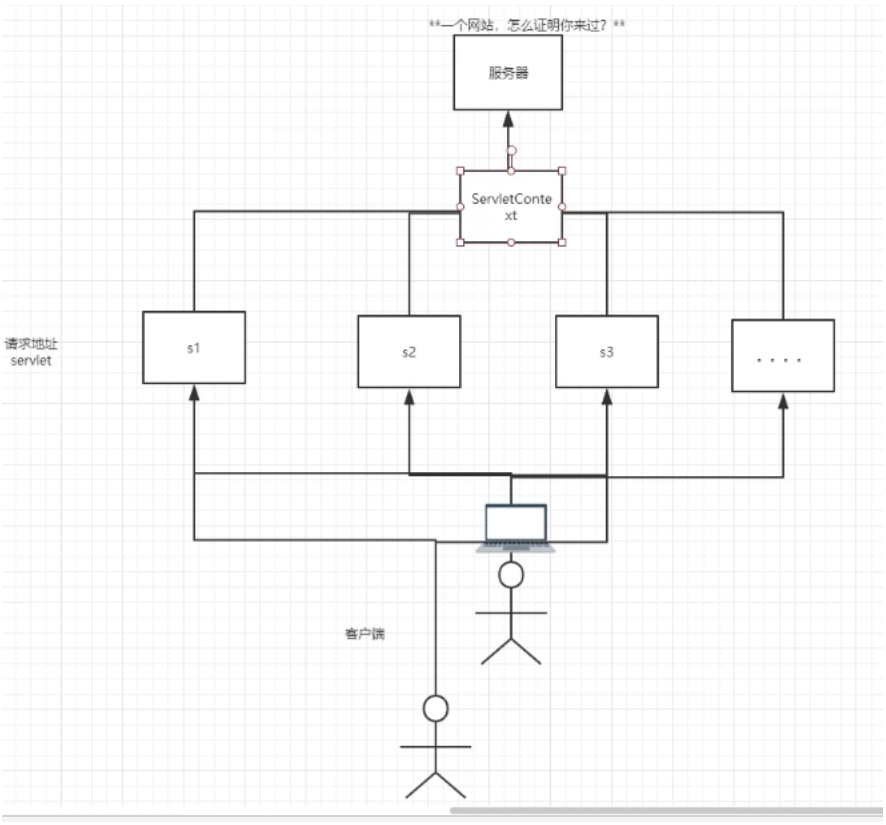
- 服务器会给每一个用户（浏览器）创建一个Seesion对象；
- 一个Seesion独占一个浏览器，只要浏览器没有关闭，这个Session就存在；
- 用户登录之后，整个网站它都可以访问！--> 保存用户的信息；保存购物车的信息.....

Session和cookie的区别：

- Cookie是把用户的数据写给用户的浏览器，浏览器保存 （可以保存多个）
- Session把用户的数据写到用户独占Session中，服务器端保存 （保存重要的信息，减少服务器资源的浪费）
- Session对象由服务创建；




一个网站，怎么证明你来过？
服务器
服务器中的Session可以存东西
登记一个Session
SessionID：每个用户唯一
请求地址
servlet
s1
s2
s3
. . . .
用户拿到的是SessionID
客户端


一个网站，怎么证明你来过？
服务器
ServletConte
xt
请求地址
servlet
s1
s2
s3
. . . .
客户端

8、JSP

8.1、什么是JSP

Java Server Pages： Java服务器端页面，也和Servlet一样，用于动态Web技术！

最大的特点：

- 写JSP就像在写HTML
- 区别：
  - HTML只给用户提供静态的数据
  - JSP页面中可以嵌入JAVA代码，为用户提供动态数据；

浏览器向服务器发送请求，不管访问什么资源，其实都是在访问Servlet！

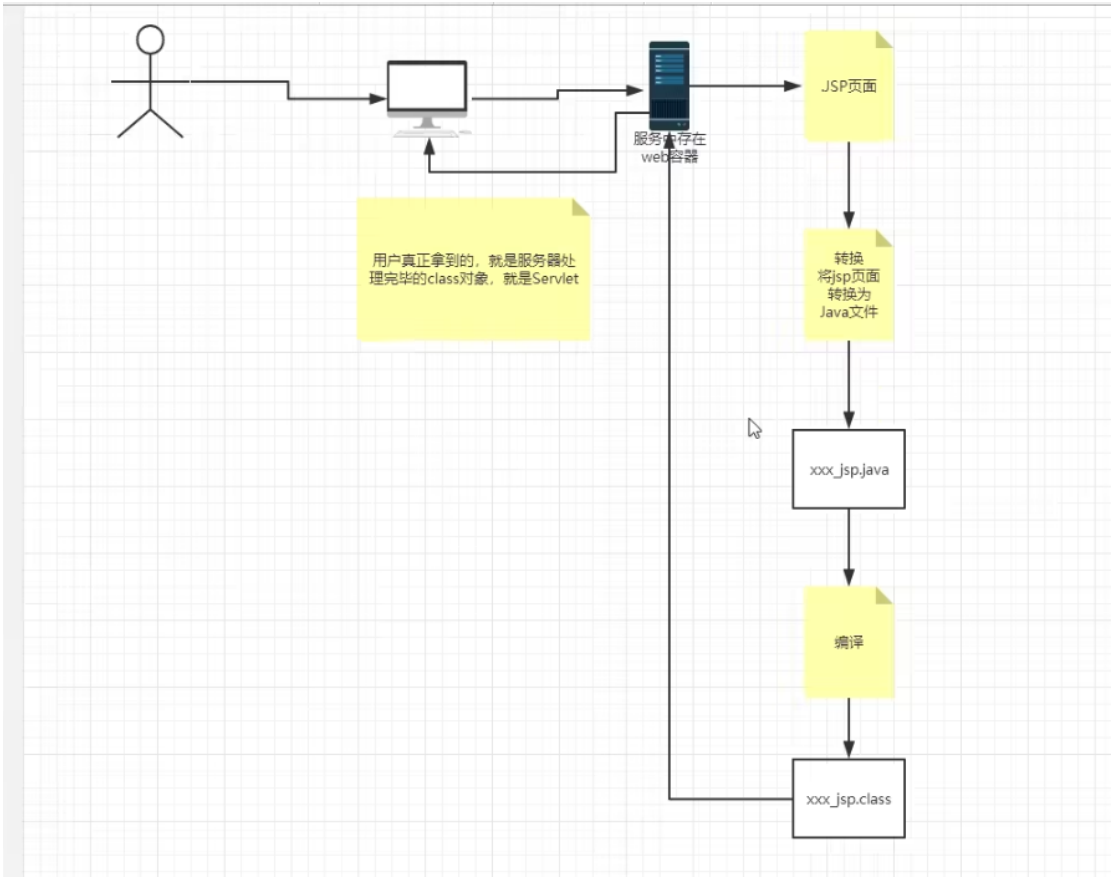
JSP最终也会被转换成为一个Java类！

JSP 本质上就是一个Servlet

```
10
```

1. 判断请求
2. 内置一些对象

```
final javax.servlet.jsp.PageContext pageContext;  //页面上下文
javax.servlet.http.HttpSession session = null;    //session
final javax.servlet.ServletContext application;   //applicationContext
final javax.servlet.ServletConfig config;         //config
javax.servlet.jsp.JspWriter out = null;           //out
final java.lang.Object page = this;               //page：当前
HttpServletRequest request                        //
HttpServletResponse response
```



在JSP页面中；

只要是 JAVA代码就会原封不动的输出；

如果是HTML代码，就会被转换为

```
out.write("<html>\r\n");
```

```
final javax.servlet.jsp.PageContext pageContext;  //页面上下文
javax.servlet.http.HttpSession session = null;    //session
final javax.servlet.ServletContext application;   //applicationContext
final javax.servlet.ServletConfig config;         //config
javax.servlet.jsp.JspWriter out = null;           //out
final java.lang.Object page = this;               //page：当前
HttpServletRequest request                        //请求
HttpServletResponse response                      //响应
```

request：客户端向服务器发送请求，产生的数据，用户看完就没用了，比如：新闻，用户看完没用的！

session：客户端向服务器发送请求，产生的数据，用户用完一会还有用，比如：购物车；

application：客户端向服务器发送请求，产生的数据，一个用户用完了，其他用户还可能使用，比如：聊天数据；

EL表达式： ${}

- **获取数据**
- **执行运算**
- **获取web开发的常用对象**

9、JavaBean

实体类

JavaBean有特定的写法：

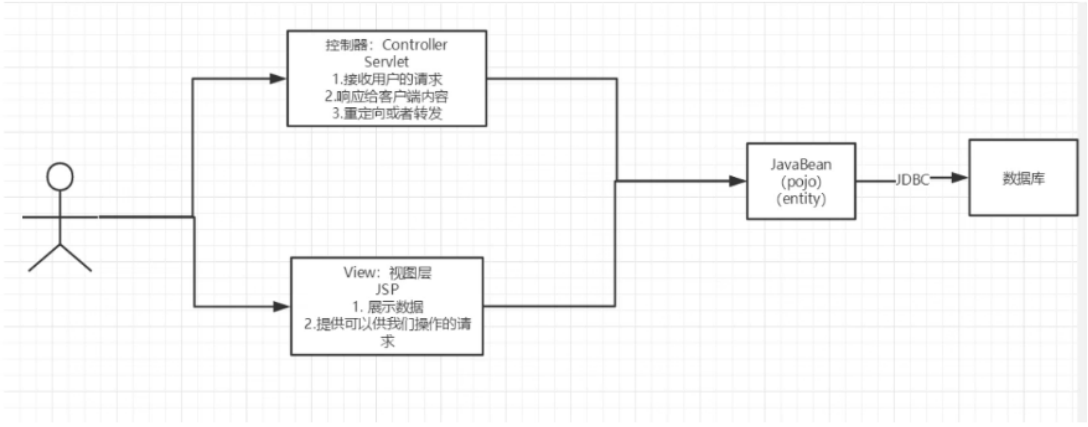
- 必须要有一个无参构造
- 属性必须私有化
- 必须有对应的get/set方法；

一般用来和数据库的字段做映射

> 一般用来和数据库的字段做映射 ORM；
>
> ORM ： 对象关系映射
>
> - 表--->类
> - 字段-->属性
> - 行记录---->对象

什么是MVC： Model view Controller 模型、视图、控制器

10.1、早些年

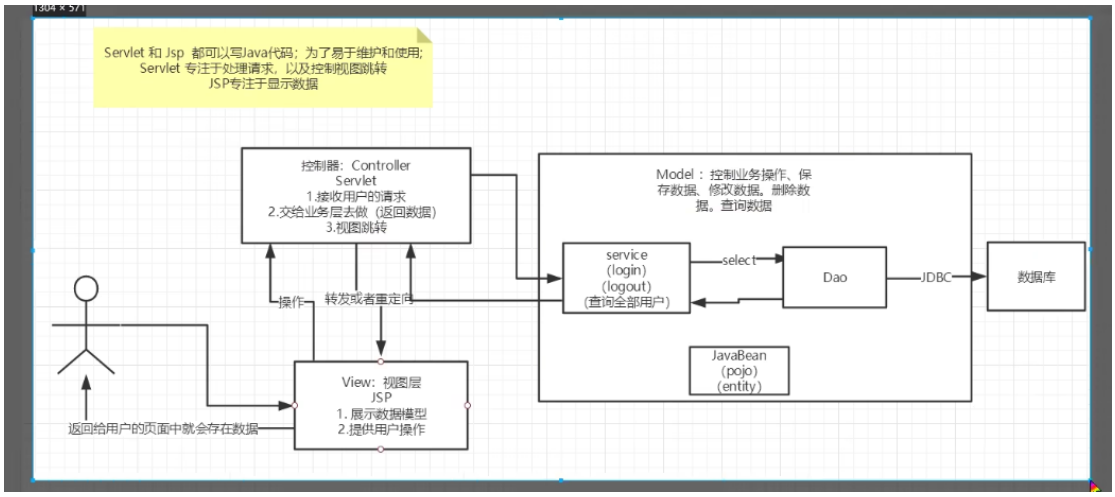


用户直接访问控制层，控制层就可以直接操作数据库；

```
servlet--CRUD-->数据库
弊端：程序十分臃肿，不利于维护
servlet的代码中：处理请求、响应、视图跳转、处理JDBC、处理业务代码、处理逻辑代码

架构：没有什么是加一层解决不了的！
```

MVC 三层架构




```
登录--->接收用户的登录请求--->处理用户的请求（获取用户登录的参数，username，password）---->交给业务层处理登录业务（判断用户名密码是否正确：事务）--->Dao层查询用户名和密码是否正确-->数据库
```

用户登录之后才能进入主页！用户注销后就不能进入主页了！

1. 用户登录之后，向Sesison中放入用户的数据
2. 进入主页的时候要判断用户是否已经登录；要求：在过滤器中实现！

```
HttpServletRequest request = (HttpServletRequest) req;
HttpServletResponse response = (HttpServletResponse) resp;

if (request.getSession().getAttribute(Constant.USER_SESSION)==null){
    response.sendRedirect("/error.jsp");
}

chain.doFilter(request,response);
```

JDBC 固定步骤：

1. 加载驱动
2. 连接数据库,代表数据库
3. 向数据库发送SQL的对象Statement : CRUD
4. 编写SQL （根据业务，不同的SQL）
5. 执行SQL
6. 关闭连接

Junit单元测试

依赖

```
<!--单元测试-->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.12</version>
</dependency>
```

简单使用

@Test注解只有在方法上有效，只要加了这个注解的方法，就可以直接运行！