1、Get 和 Post 方法的区别

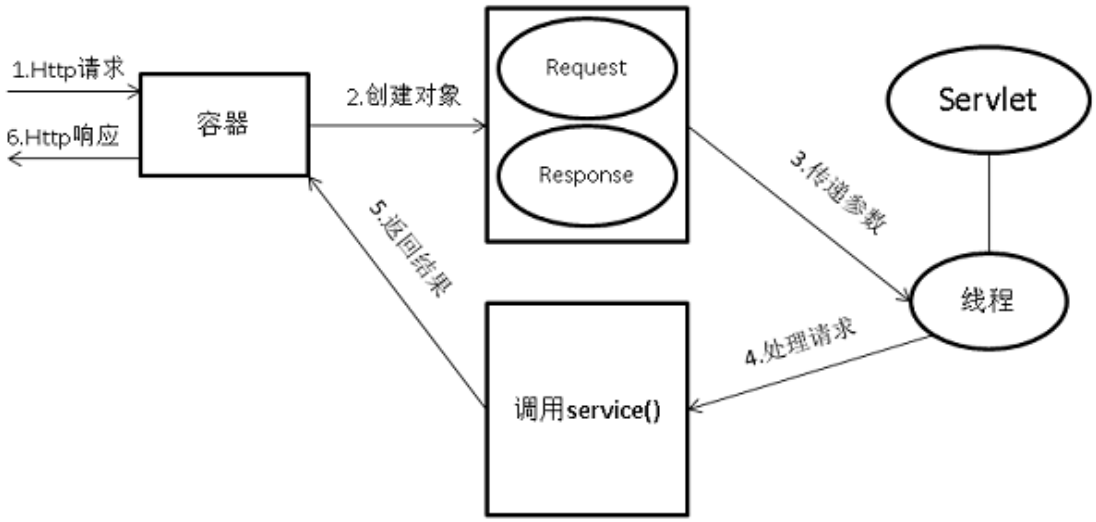
- Get 方法主要用于获取服务器资源，而 Post 方法除了能获取资源外，还可以向服务器上传数据。
- Get 方法会把请求的数据附在 URL 后面，而 Post 不会
- Get 方法传输小数据，而 Post 方法主要用来传递大数据或比较隐私的数据，因此 Post 方法的安全性高一点

2、Servlet 处理访问请求的流程

Servlet 是一种实现了 javax.servlet.Servlet 接口的类。编写 Servlet 时直接继承 HttpServlet，并覆盖 doGet()与 doPost()方法即可。

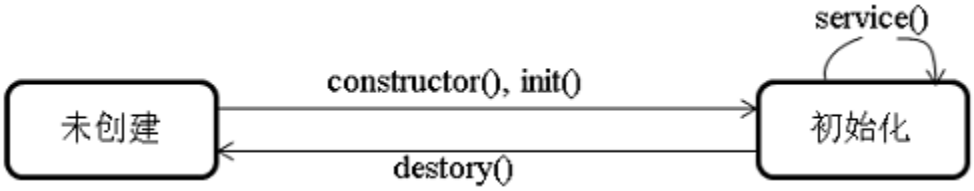
- Web 服务器接受到用户请求后，会把请求交给相应的容器来处理，此时容器会创建两个对象：HttpServletResponse 和 HttpServletRequest。
- 容器根据 URL 信息找到对应的 Servlet，然后会创建一个线程，把创建的两个对象以参数的形式传递到新建的线程中
- 容器调用 Servlet 的 service()方法来完成对用户请求的响应，service()方法会调用 doGet 或 doPost 方法来完成具体的响应任务，同时把生成的动态页面返回给容器
- 容器把响应信息组装成 HTTP 格式返回给客户端，同时线程结束，删除创建的两个对象：HttpServletResponse 和 HttpServletRequest

Servlet 不足：Servlet 生成网页的方法是在 java 类中嵌入 HTML 标签和表达式，开发麻烦，可读性差，由于业务逻辑与视图没有分离，系统的可扩展性和维护性都很差。



3、Servlet 的生命周期

Servlet 的生命周期由容器来控制，它只有两个状态：未创建状态与初始化状态。



Servlet 生命周期：init()、service()、destroy()。在生命周期中，init()和 destroy()只被服务器执行一次， 而 service()在每次客户端请求 Servlet 时都会被执行，或 doGet()或 doPost()。

4、Servlet 和 CGI（通用网关接口）的区别

Servlet 处于服务器进程中，它通过多线程方式运行其 service 方法，一个实例可以服务于多个请求，并且其实例一般不会销毁。而 CGI 对每个请求都产生新的进程，服务完成后就销毁，所以效率上低于 Servlet。

5、Servlet 中 forward 和 redirect 区别

forward（可定向同一个服务器的资源）：服务器内部的重定向，服务器直接访问目标地址的 URL，把那个 URL 响应的内容返回给客户端。而客户端并不知道该过程，因此地址栏中不会显示转向后的地址，还是原来的地址。整个定向过程使用同一个 Request，因此定向的 JSP 或 Servlet 可以使用该 Request 的信息。

redirect（可定向其他服务器的资源）：客户端的重定向，客户端会获取到重定向的地址，然后重新发送请求，因此地址栏会显示跳转后的地址。由于多一次请求，其效率低于 forword 方式。

6、Servlet 是否线程安全

Servlet 本身是多线程的，但不是线程安全的，多线程并发的读写会导致数据不同步的问题。三种解决方案：

- 实现 SingleThreadModel 接口，在这个 Servlet 中的 service 方法将不会有两个线程被同时执行
- 使用 synchronized 关键字， 能保证一次只有一个线程可以访问被保护的区段
- 避免使用实例变量，只要在 Servlet 里面的任何方法里面都不使用实例变量，那么该 Servlet 就是线程安全的

7、JSP 技术

JSP 是一种特殊的 Servlet，所有 Servlet 能完成的事情，JSP 都能完成，在实际中，常用 Servlet 来处理业务逻辑，采用 JSP 来生成动态网页。

JSP 与 Servlet 区别：

- 最大区别：Servlet 以 Java 程序为主，Java 中内嵌 HTML；而 JSP 则以 HTML 页面为主，HTML 中内嵌 Java
- 工作方式不同：Servlet 先编译后部署，即先编译成 class 文件后部署到服务器下；而 JSP 则是先部署后编译，JSP 会在首次请求时编译成 class 类，以后访问时用该类响应请求。

JSP 与 HTML 区别：

- HTML 超文本标记语言，是静态页面，和 JavaScript 一样解释性语言，只要有浏览器就可以正常显示，而不需要指定的编译工具。
- JSP 是是动态页面，我们都知道，JVM 执行的是 Java 文件编译后的 class 文件，而 JSP 要先转译成一个 Servlet 文件，然后在编译成 class 文件。

8、MVC 模型

MVC 的处理过程：对于用户的请求，先被控制器接收，并决定由哪个模型来处理，然后模型通过业务逻辑层处理用户的请求并返回数据，最后控制器用相应的视图模型来显示这些数据。

MVC 优点：

- 低耦合性
- 高重用性和可适用性
- 低开发周期
- 部署快速
- 易于维护

9、JSP 的内置对象有哪些

- request：客户端的请求信息被封装在 request 对象中，通过它才能了解到客户的需求， 然后做出响应。它是 HttpServletRequest 类的实例。
- response：表示服务端对客户端的响应，将服务器处理后的结果返回给客户端，它是 HttpServletResponse 类的实例。
- pageContext：提供了对 JSP 页面内所有的对象及名字空间的访问，也就是说他可以访问到本页所在的 SESSION，也可以取本页面所在的 application 的某一属性值，他相当于页面中所有功能的集大成者。
- session：session 对象指的是客户端与服务器的一次会话，从客户连到服务器的一个 WebApplication 开始，直到客户端与服务器断开连接为止。它是 HttpSession 类的实例。
- application：代表 JSP 所属的 Web 应用本身，application 对象可存放全局变量，以实现用户间的数据共享。它的生命周期和服务器的生命一致，它是 ServletContext 类的实例。
- out：用于输出信息，它是 JspWriter 的实例
- config：主要作用是取得服务器的配置信息，当 Servlet 初始化时，容器把某些信息通过 config 对象传递给这个 Servlet，Servlet 可使用该对象获取所需配置信息。
- page：表示当前 JSP 页面

- exception：表示异常，当运行过程发生了例外，就产生这个对象。如果一个 JSP 页面要应用此对象，就必须把 isErrorPage 设为 true，否则无法编译。他是 Java.lang.Throwable 的对象

10、JSP 指令有哪些

JSP 指令用来设置整个 JSP 页面相关属性，如网页编码方式和脚本语言等, JSP 指令的多个属性可以写在一个指令里, 也可以写在多个指令里，每个属性只能出现一次，import 例外。

- page 指令：指明网页依赖属性，语法格式：<%@ page attribute="value" %>
- include 指令：实现 JSP 页面的区块化，语法格式：<%@ include file="文件相对 url 地址" %>
- Taglib 指令：引入标签库，语法格式：<%@ taglib uri="uri" prefix="前缀名" %>

11、JSP 行为(动作)有哪些

- jsp:include：在页面被请求时插入一个文件，语法格式：<jsp:include page="相对 URL 地址" flush="true" />
- jsp:useBean：用于加载一个将在 JSP 页面中使用的 JavaBean，在类加载后，可以通过 jsp:setProperty 和 jsp:getProperty 动作来修改和检索 bean 的属性。语法格式：<jsp:useBean id="name" class="package.class" />
- jsp:forward：把请求转到另外的页面，只有 page 属性，语法格式：<jsp:forward page="相对 URL 地址" />

- jsp:plugin：用来根据浏览器的类型，插入通过 Java 插件 运行 Java Applet 所必需的 OBJECT 或 EMBED 元素。

12、include 指令与 include 动作的区别

include 指令（在编译时包含，包含的是源代码）：所包含的文件内容是在编译时插入到 JSP 文件中的，当文件内容有变化时就需要重新编译，因此适用于包含静态页面

include 动作(在运行时包含,包含的是运行结果)：在主页面被请求时，才将包含的页面输出包含进来，适用于包含动态页面

13、会话跟踪技术有哪些

HTTP 是无状态协议，无状态是指同一个会话的连续两个请求互相不了解，当浏览器发送请求给服务器的时候，服务器响应，但是同一个浏览器再次发送请求时，服务器同样会响应，但是它不知道你就是刚才那个浏览器，每一次请求和响应都是相对独立的。客户端与服务端的交互是需要承前启后的，简单的购物车程序也要知道用户到底在之前选择了什么商品，因此就有了会话跟踪技术。

- Cookie：通过客户端保持状态的解决方案。Cookie 是由服务器发给客户端的特殊信息，并存放在客户端浏览器上，然后客户端每次请求都会附上这些特殊的信息，服务器检查该 Cookie，以此来辨认用户状态。
- Session：通过服务器端保持状态的解决方案。浏览器访问服务器时，服务器把客户端信息存放在服务器，这就是 Session，客户端

每次访问都需要从该 Session 中确认该客户的状态，服务器维护并更新该 Session。

- **URL 重写**：是客户端不支持 Cookie 的解决方案。原理是将用户 Session 的 id 信息重写到 URL 地址中，服务器解析 URL 并获取 Session 的 id。这样即使客户端不支持 Cookie，也可以使用 Session 来记录用户状态。
- **隐藏表单域**：非常适合不需要大量数据储存的会话应用

14、Cookie 与 Session 的区别

- Cookie 机制采用客户端保持状态的解决方案，即数据存放在客户端浏览器上；Session 机制采用服务器端保持状态的解决方案，即数据存放在服务器上。
- Session 比 Cookie 的安全性高。敏感信息存放在浏览器上会有安全问题，最好是将 Cookie 信息加密，提交到服务器后再解密，而 Session 不存在该问题。
- cookie 性能更高一些。每个用户都会产生一个 Session 存放在服务器上，访问量增大时会降低服务器性能；而 Cookie 不占用服务器资源。
- Cookie 信息可永久保存在浏览器上，而保存在服务器的 Session 信息过了超时时间或关闭浏览器后就会失效。

15、什么是 Ajax 技术

Ajax 是一个结合了 Java 技术、XML 以及 JavaScript 的编程技术，其主要目的是在不刷新页面的情况下实现页面的局部刷新，通过与服务器进行

少量数据的交互来提高页面的交互性，减少响应的时间，从而改善用户的体验。

Ajax 是客户端技术，其核心是 JavaScript 对象 XmlHttpRequest，该对象是一种支持异步请求的技术，它使得开发人员可以使用 JavaScript 向服务器提出请求并处理响应，而并不阻塞用户。

16、过滤器 Filter

Filter 用于在 Servlet 之外对 request 和 response 进行修改。Filter 提出滤镜链的概念，一个 FilterChain 包括多个 Filter。客户端请求 request 在抵达 Servlet 之前会经过 FilterChain 里的所有 Filter，服务器响应 response 在从 Servlet 抵达客户端浏览器之前也会经过 FilterChain 里的所有 Filter。

Filter 实现案例

- 字符串编码 Filter
- 权限验证 Filter
- 异常捕捉 Filter
- 内容替换 Filter
- 图像水印 Filter

17、监听器 Listener

Listener 用于监听 web 应用程序中的 ServletContext, HttpSession 和 ServletRequest 等域对象的创建与销毁事件，以及监听这些域对象中的属性发生修改的事件。

18、拦截器(Intercepter)与 Filter 的区别

相同点：两者都是 AOP 编程思想的体现，都能实现权限检查、日志记

录等

不同点:

- 范围不同：Filter 是 Servlet 规定的，只能用于 web 程序中。而拦截器还能用于 application、swing 程序中
- 规范不同：Filter 是 Servlet 规范定义的，是 Servlet 容器支持的。而拦截器是在 Spring 容器里的，是 Spring 框架支持的。
- 拦截器是 Spring 的一个组件，归 Spring 管理，配置在 Spring 文件中，因此能使用 Spring 中的各种资源，对象，通过 IoC 注入即可；而 Filter 不能
- 深度不同：Filter 只能在 Servlet 前后起作用。而拦截器能够深入到方法前后、异常抛出前后等。拦截器具有更大的弹性

19、Struts1.x 工作流程

- 客户端发送请求，被控制器 ActionServlet 捕获
- ActionServlet 根据 struts-config.xml 配置文件里的映射关系找到对应的 Action 和 ActionForm
- 将提交的表单信息封装成 ActionForm 对象，并反射调用 Action 的 executr()方法，执行业务逻辑程序
- 执行完后，通过返回一个 ActionForward 对象跳转到对应的 JSP 页面

20、Struts2.x 工作流程

- 客户端发送请求，被控制器 FilterDispatcher 捕获

- FilterDispatcher 根据 struts.xml 配置文件找到对应的 Action 类，并通过 IoC 方式将表单信息注入到该 Action 类的属性值

- Action 调用相应的方法执行业务逻辑任务
- Action 执行完后返回 result 字符串，根据配置文件找到相应的跳转页面，并返回给客户端

21、Struts1.x 与 Struts2.x 的线程安全

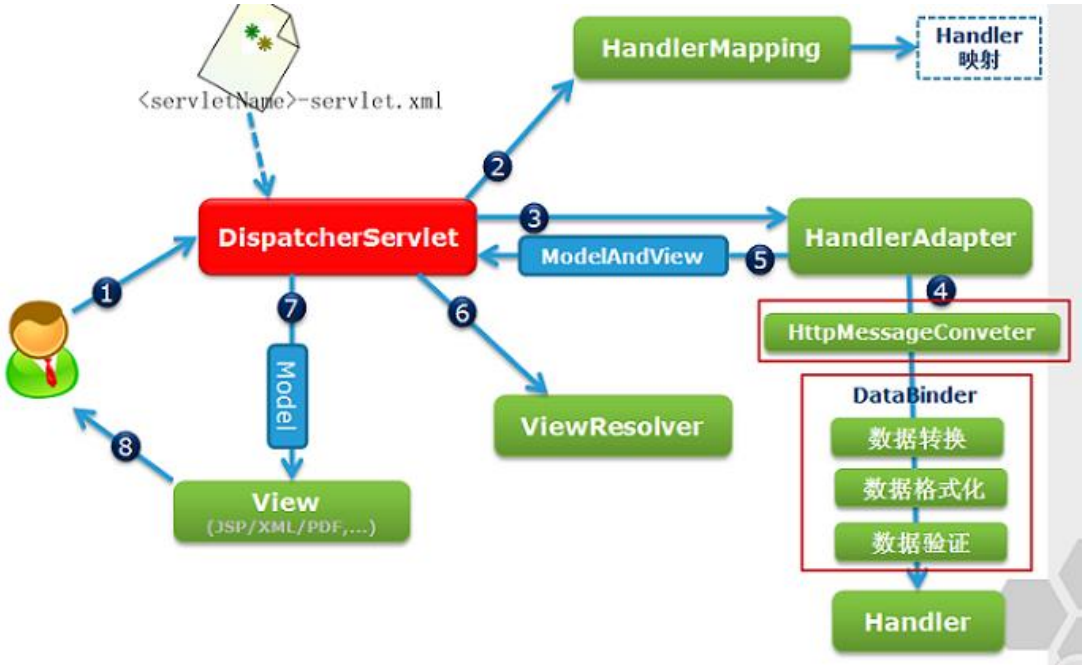
- **Struts1.x 的 Action 是非线程安全的**，Struts 1.x 在第一次请求某个 Action 时，会创建一个 Action 实例，而该 Action 实例会被反复使用，因此开发时尽量不要使用实例变量。
- **Struts2.x 的 Action 是线程安全的**，每次请求都会生成一个 Action 实例，这些实例彼此独立，处理完后立即销毁。

22、Spring

Spring 以 IoC、AOP 为主要思想，它能将 Struts、Hibernate 等众多框架部署到 Spring 中，统一维护管理它们，因此 Spring 也被称为轻量级的“容器”。

- **IoC（反向控制）**：IoC 是对传统控制流程的一种颠覆，在程序中既没有实例化对象，也没有设置依赖关系，而是交给 Spring，由 Spring 根据配置文件加载配置文件中的 Java Bean，并通过 Java 的反射机制调用 setter、getter 方法注入依赖的对象。
- **AOP（面向切面编程）**：AOP 使得在执行业务前后可以执行另外的程序，Filter 和拦截器就是 AOP 的实现

23、SpringMVC 工作流程



- 用户请求被 Spring 前端控制器 DispatcherServlet 捕获
- DispatcherServlet 对请求 URL 进行解析，得到请求 URI(资源标识符)，然后根据该 URI 以及 HandlerMapping（处理器映射器）的配置找到处理该请求的 Handler（处理器）

- DispatcherServlet 根据获得的 Handler，选择一个合适的 HandlerAdapter（处理器适配器）（如果成功获得将执行拦截器任务，否则可忽略该过程）

- Handler 执行完业务逻辑后，向 DispatcherServlet 返回一个 ModelAndView 对象
- DispatcherServlet 请求 ViewReaolver（视图解析器）来解析该 ModelAndView 对象，得到 View 视图，最后把渲染结果返回给客

户端

24、SpringMVC 的 Controller 是线程安全吗

不是，Spring 根据 IoC 来实例化对象，所生成的对象默认是单例的（如 Controller、Dao、Service），所以定义成员变量会有线程安全问题。如果一定要定义的话，有如下解决方法：

- 在 Controller 中使用 ThreadLocal 变量
- 在 Controller 类上声明 @Scope("prototype")，每次都创建新的 controller

25、Struts2 与 SpringMVC 比较

- SpringMVC 的入口是 Servlet，而 Struts2 是 Filter
- SpringMVC 稍微比 Struts2 快，SpringMVC 是基于方法设计，而 Sturts2 是基于类，每次发一次请求都会实例一个 Action
- SpringMVC 使用更加简洁，开发效率高，支持 JSR303（数据校验），处理 Ajax 的请求更方便
- Struts2 的 OGNL 表达式使页面的开发效率相比 Spring MVC 更高些.

26、可持久化对象的状态

可持久化对象就是可以保存进数据库的对象，可持久化对象有以下几种状态（Session 代表用户的一次数据库操作）

- **临时状态（Transient）**：对象在保存进数据库之前为临时状态。如果没有被持久化，程序退出时临时的对象信息将会丢失。
- **持久化状态（Persistent）**：对象保存进数据库后，并且没有脱离 Session 的为持久化状态。持久化状态的对象可以执行任何有关数据库的操作。
- **分离状态（Detached）**：分离状态是对象曾处于持久化状态，但现在已经离开 Session，此时不能执行数据库的操作。

27、JDBC、Hibernate 与 Mybatis 对比

- 从层次上看，JDBC 是较底层的持久层操作方式，而 Hibernate 和 MyBatis 都是在 JDBC 的基础上进行了封装使其更加方便程序员对持久层的操作。
- 从功能上看，JDBC 就是简单的建立数据库连接，然后操作 Connection 对象、Statement 对象和 ResultSet 对象去拿到数据；Hibernate 是将数据库中的数据表映射为持久层的 Java 对象，实现数据表的完整性控制；MyBatis 是将 sql 语句中的输入参数和输出参数映射为 java 对象，放弃了对数据表的完整性控制，但是获得了更灵活和响应性能更快的优势。
- 从使用上看，如果进行底层编程，而且对性能要求极高的话，应该采用 JDBC 的方式；如果要对数据库进行完整性控制，实现简单的增删改查，建议使用 Hibernate；如果要灵活使用 sql 语句，需要较多复杂的查询语句，建议采用 MyBatis。

- 从掌握程度来说，Hibernate 要精通门槛更高，而且设计 O/R 映射难，在性能和对象模型之间如何权衡取得平衡，以及怎样用好 Hibernate 方面需要你的经验和能力都很强才行。Mybatis 框架相对简单很容易上手。

28、Sql 中 delete 与 truncate 的区别

- delete：可选择性地删除数据，当删除整张表的数据时效率较低
- truncate：只能删除整张表的数据，但效率高于使用 delete 语句，当 truncate 执行删除之后，自动生成的主键值重新从默认值开始。

29、XML 的 java 解析有几种方式?

Dom 解析、Sax 解析、Dom4J 解析、JDOM 解析

下面比较官方提供的两种:

No	区 别	DOM解析	SAX解析
1	操作	将所有文件读取到内存中形成DOM树，如果文件量过大，则无法使用	顺序读入所需要的文件内容，不会一次性全部读取，不受文件大小的限制
2	访问限制	DOM树在内存中形成，可以随意存放或读取文件树的任何部分，没有次数限制	由于采用部分读取，只能对文件按顺序从头到尾读取XML文件内容，但不能修改
3	修改	可以任意修改文件树	只能读取XML文件内容，但不能修改
4	复杂度	易于理解，易于开发	开发上比较复杂，需要用户自定义事件处理器
5	对象模型	系统为使用者自动建立DOM树，XML对象模型由系统提供	对开发人员更加灵活，可以用SAX建立自己的XML对象模型

DOM 解析适合于对文件进行修改和随机存取的操作，但是不适合于大型文件的操作;

SAX 采用部分读取的方式，所以可以处理大型文件，而且只需要从文

件中读取特定内容，SAX 解析可以由用户自己建立自己的对象模型。

30、什么是事务？事务有那些特点？

事务：单个逻辑单元执行的一系列操作，要么全部执行，要么全部不执行。

- 原子性（Atomicity）：事务中各元素不可分割，要么执行所有操作，要么撤销所有操作
- 一致性（Consistency）：事务执行的结果必须是使数据库从一个一致性状态变到另一个一致性状态。比如，当数据库只包含成功事务提交的结果时，就说数据库处于一致性状态。如果数据库系统在运行中发生故障，有些事务尚未完成就被迫中断，这些未完成事务对数据库所做的修改有一部分已写入物理数据库，这时数据库就处于一种不正确的状态，或者说是不一致的状态。
- 隔离性（Isolation）：事务是相对独立的，对某数据进行修改时，其他事务不变
- 持久性（Durability）：事务完成后对系统的影响是永久性的。

31、JSP 由哪些内容组成？

- 指令：<%@ %>
- 脚本：<% %>
- 表达式：<%=%>
- 声明：<%! %>
- 注释：<% -- %>

- 动作：<jsp：动作名称 属性=””>
- 静态内容：html 内容

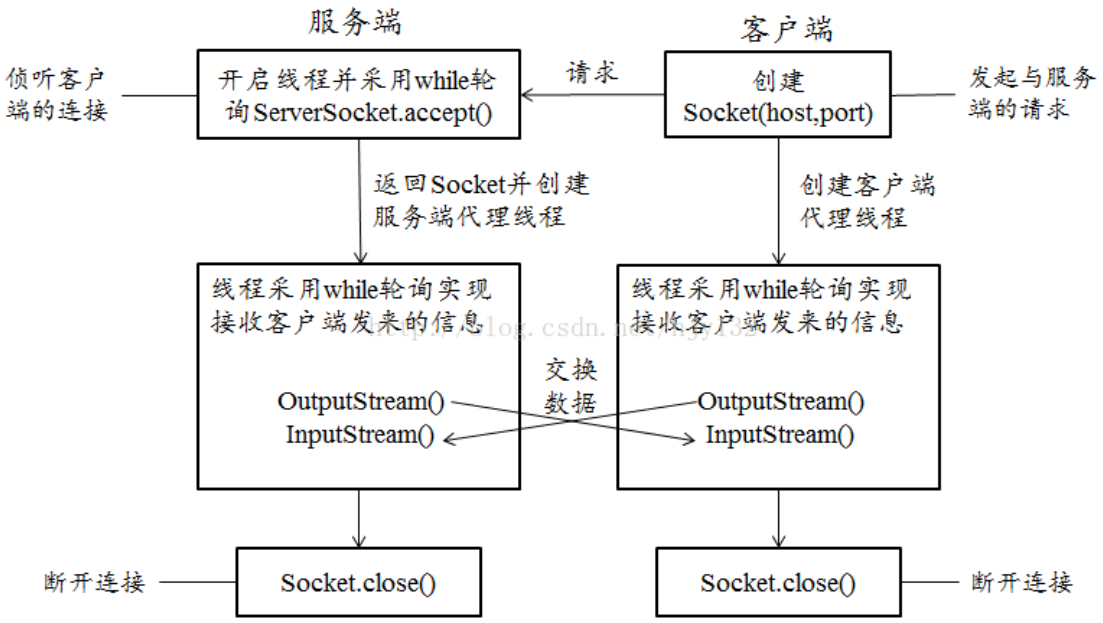
32、计算机网络有几层

- 应用层
- 表示层
- 会话层
- 传输层
- 网络层
- 数据链路层
- 物理层

33、TCP 与 UDP 区别

- TCP 是面向连接的，发送数据前必须建立可靠的连接；而 UDP 是无连接的，不需要建立连接
- TCP 传输可靠，能保证数据正确性；而 UDP 是不可靠的，传输时可能丢包
- TCP 用于传输大量数据（流模式）；UDP 用于少量数据（数据包模式）
- UDP 比 TCP 快

34、创建 Socket 通信的步骤



35、JSP、Servlet 中通信作用域有那些?

- PageContext
- Request
- Session
- Application

36、URL 与 URI，URL 由哪些部分组成

URI：统一资源标识符；URL：统一资源定位符

http://localhost:8080/myWeb/index.html：协议+主机地址+端口+项目名称+资源名称

37、HTTP 状态码

- 1**：信息，服务器收到请求，需要请求者继续执行操作

- 2**：成功，操作被成功接收并处理，如 200-请求成功
- 3**：重定向，需要进一步的操作以完成请求，如 301-资源被移到其他 URL
- 4**：客户端错误，如 404-请求资源不存在
- 5**：服务端错误，如 500-内部服务器错误

38、使用 sql 写出一个分页程序?

Select top 3 * from tb_name where id not in (select top 3 id from tb_name)

39、什么是 Web Service（Web 服务）

WebService 是一种跨编程语言和跨操作系统平台的远程调用技术。例如可以创建一个提供天气预报的 Web Service，那么无论你用哪种编程语言开发的应用都可以通过调用它的 API 并传入城市信息来获得该城市的天气预报。

40、TCP 三次握手

请求---应答---再次确认

- 一次握手：建立连接时，客户端发送 SYN 包到服务端，并进入 SYN_SEND 状态，等待服务端确认
- 二次握手：服务端收到 SYN 包后，必须确认客户的 SYN 包，同时自己也发送一个 SYN 包（即 SYN+ACK 包），此时服务端进入 SYN_RECV 状态

- 三次握手：客户端收到服务端的 SYN+ACK 包，向服务端发送确认包 ACK(ack=k+1)，到此客户端和服务端进入建立连接状态

41、TCP 四次撒手

- 客户端向服务端发送一个 FIN 报文，告诉服务端我没有数据要发给你了
- 服务端收到这个 FIN 报文后发回一个 ACK，告诉客户端已收到请求，但我目前还没准备好，请等待我的回复信息，此时客户端进入 FIN_WAIT 状态
- 服务端确定数据已传输完成后，向客户端发送一个 FIN 报文。告诉客户端我已准备好关闭连接
- 客户端收到 FIN 报文后返回一个 ACK（此时客户端进入 TIME_WAIT 状态，如果服务端没有收到 ACK 则可以重发），告诉服务器你可以关了，于是服务器就关闭了，客户端等待 30s 后没有收到回复也关闭了

软件设计的两种架构

B/S（Browse/Server）：指浏览器和服务器端，客户端只需有浏览器，就可以实现与服务器端通信的程序结构。（瘦客户端）

C/S（Client/Server）：指客户机和服务器，在客户机端必须装客户端软件及相应环境后，才能访问服务器（胖客户端）

B/S 特点：（客户端维护成本低，可跨平台，但服务器负担重，缺点是客户端功能较简单，用户体验不如 C/S。）

C/S 特点：（客户端功能强大，可以减轻服务器端压力，但是客户端维护开发成本高。）