02 | HTTP协议必知必会

2019-05-14 李号双

深入拆解Tomcat & Jetty

进入课程 >

讲述：李号双
时长 10:21　大小 9.49M

在开始学习 Web 容器之前，我想先问你一个问题：HTTP 和 HTML 有什么区别？

为什么我会问这个问题？你可以把它当作一个入门测试，检测一下自己的对 HTTP 协议的理解。因为 Tomcat 和 Jetty 本身就是一个“HTTP 服务器 + Servlet 容器”，如果你想深入理解 Tomcat 和 Jetty 的工作原理，我认为理解 HTTP 协议的工作原理是学习的基础。

如果你对这个问题还稍有迟疑，那么请跟我一起来回顾一下 HTTP 协议吧。

HTTP 的本质

HTTP 协议是浏览器与服务器之间的数据传送协议。作为应用层协议，HTTP 是基于 TCP/IP 协议来传递数据的（HTML 文件、图片、查询结果等），HTTP 协议不涉及数据包

（Packet）传输，主要规定了客户端和服务器之间的通信格式。

下面我通过一个例子来告诉你 HTTP 的本质是什么。

假如浏览器需要从远程 HTTP 服务器获取一个 HTML 文本，在这个过程中，浏览器实际上要做两件事情。

与服务器建立 Socket 连接。

生成**请求数据**并通过 Socket 发送出去。

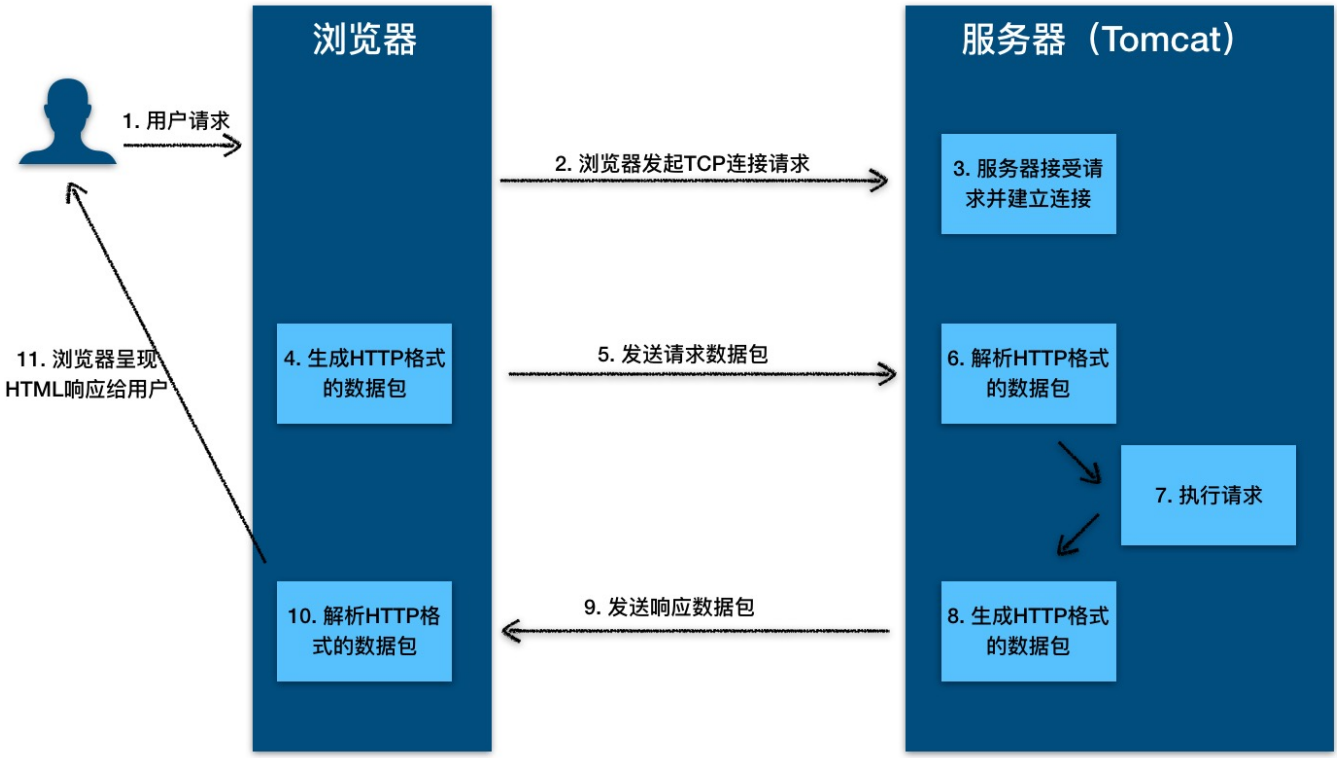
第一步比较容易理解，浏览器从地址栏获取用户输入的网址和端口，去连接远端的服务器，这样就能通信了。

我们重点来看第二步，这个请求数据到底长什么样呢？都请求些什么内容呢？或者换句话说，浏览器需要告诉服务端什么信息呢？

首先最基本的是，你要让服务端知道你的意图，你是想获取内容还是提交内容；其次你需要告诉服务端你想要哪个内容。那么要把这些信息以一种什么样的格式放到请求里去呢？这就是 HTTP 协议要解决的问题。也就是说，**HTTP 协议的本质就是一种浏览器与服务器之间约定好的通信格式。**那浏览器与服务器之间具体是怎么工作的呢？

HTTP 工作原理

请你来看下面这张图，我们过一遍一次 HTTP 的请求过程。



从图上你可以看到，这个过程是：

1. 用户通过浏览器进行了一个操作，比如输入网址并回车，或者是点击链接，接着浏览器获取了这个事件。

2. 浏览器向服务端发出 TCP 连接请求。

3. 服务程序接受浏览器的连接请求，并经过 TCP 三次握手建立连接。

4. 浏览器将请求数据打包成一个 HTTP 协议格式的数据包。

5. 浏览器将该数据包推入网络，数据包经过网络传输，最终达到端服务程序。

6. 服务端程序拿到这个数据包后，同样以 HTTP 协议格式解包，获取到客户端的意图。

7. 得知客户端意图后进行处理，比如提供静态文件或者调用服务端程序获得动态结果。

8. 服务器将响应结果（可能是 HTML 或者图片等）按照 HTTP 协议格式打包。

9. 服务器将响应数据包推入网络，数据包经过网络传输最终达到到浏览器。

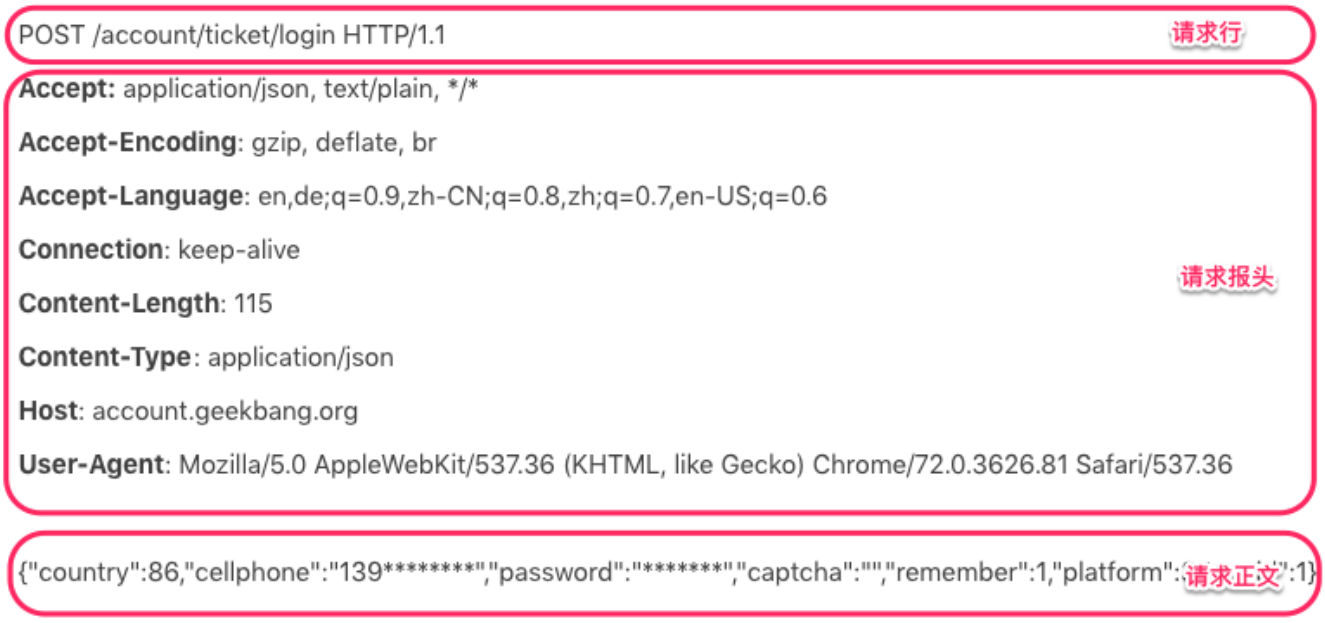
10. 浏览器拿到数据包后，以 HTTP 协议的格式解包，然后解析数据，假设这里的数据是 HTML。

11. 浏览器将 HTML 文件展示在页面上。

那我们想要探究的 Tomcat 和 Jetty 作为一个 HTTP 服务器，在这个过程中都做了些什么事情呢？主要是接受连接、解析请求数据、处理请求和发送响应这几个步骤。这里请你注意，可能有成千上万的浏览器同时请求同一个 HTTP 服务器，因此 Tomcat 和 Jetty 为了提高服务的能力和并发度，往往会将自己要做的几个事情并行化，具体来说就是使用多线程的技术。这也是专栏所关注的一个重点，我在后面会进行专门讲解。

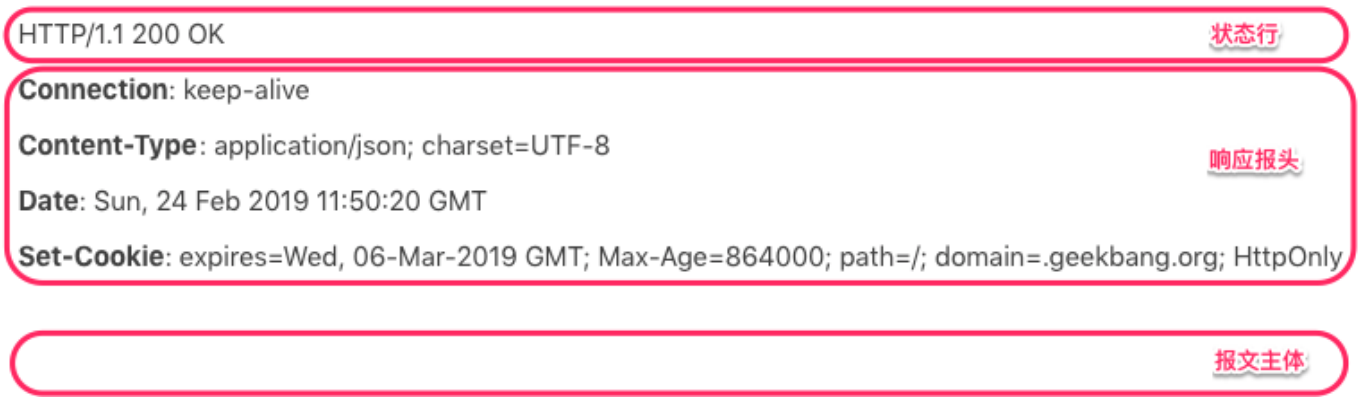
HTTP 请求响应实例

你有没有注意到，在浏览器和 HTTP 服务器之间通信的过程中，首先要将数据打包成 HTTP 协议的格式，那 HTTP 协议的数据包具体长什么样呢？这里我以极客时间的登陆请求为例，用户在登陆页面输入用户名和密码，点击登陆后，浏览器发出了这样的 HTTP 请求：



你可以看到，HTTP 请求数据由三部分组成，分别是**请求行、请求报头、请求正文**。当这个 HTTP 请求数据到达 Tomcat 后，Tomcat 会把 HTTP 请求数据字节流解析成一个 Request 对象，这个 Request 对象封装了 HTTP 所有的请求信息。接着 Tomcat 把这个 Request 对象交给 Web 应用去处理，处理完后得到一个 Response 对象，Tomcat 会把这个 Response 对象转成 HTTP 格式的响应数据并发送给浏览器。

我们再来看看 HTTP 响应的格式，HTTP 的响应也是由三部分组成，分别是**状态行、响应报头、报文主体**。同样，我还以极客时间登陆请求的响应为例。



具体的 HTTP 协议格式，你可以去网上搜索，我就不再赘述了。为了更好地帮助你理解 HTTP 服务器（比如 Tomcat）的工作原理，接下来我想谈一谈 Cookie 跟 Session 的原理。

Cookie 和 Session

我们知道，HTTP 协议有个特点是无状态，请求与请求之间是没有关系的。这样会出现一个很尴尬的问题：Web 应用不知道你是谁。比如你登陆淘宝后，在购物车中添加了三件商品，刷新一下网页，这时系统提示你仍然处于未登录的状态，购物车也空了，很显然这种情况是不可接受的。因此 HTTP 协议需要一种技术让请求与请求之间建立起联系，并且服务器需要知道这个请求来自哪个用户，于是 Cookie 技术出现了。

1. Cookie 技术

Cookie 是 HTTP 报文的一个请求头，Web 应用可以将用户的标识信息或者其他一些信息（用户名等）存储在 Cookie 中。用户经过验证之后，每次 HTTP 请求报文中都包含 Cookie，这样服务器读取这个 Cookie 请求头就知道用户是谁了。**Cookie 本质上就是一份存储在用户本地的文件，里面包含了每次请求中都需要传递的信息。**

2. Session 技术

由于 Cookie 以明文的方式存储在本地，而 Cookie 中往往带有用户信息，这样就造成了非常大的安全隐患。而 Session 的出现解决了这个问题，**Session 可以理解为服务器端开辟的存储空间，里面保存了用户的状态**，用户信息以 Session 的形式存储在服务端。当用户请求到来时，服务端可以把用户的请求和用户的 Session 对应起来。那么 Session 是怎么

和请求对应起来的呢？答案是通过 Cookie，浏览器在 Cookie 中填充了一个 Session ID 之类的字段用来标识请求。

具体工作过程是这样的：服务器在创建 Session 的同时，会为该 Session 生成唯一的 Session ID，当浏览器再次发送请求的时候，会将这个 Session ID 带上，服务器接受到请求之后就会依据 Session ID 找到相应的 Session，找到 Session 后，就可以在 Session 中获取或者添加内容了。而这些内容只会保存在服务器中，发到客户端的只有 Session ID，这样相对安全，也节省了网络流量，因为不需要在 Cookie 中存储大量用户信息。

3. Session 创建与存储

那么 Session 在何时何地创建呢？当然还是在服务器端程序运行的过程中创建的，不同语言实现的应用程序有不同的创建 Session 的方法。在 Java 中，是 Web 应用程序在调用 HttpServletRequest 的 getSession 方法时，由 Web 容器（比如 Tomcat）创建的。那 HttpServletRequest 又是什么呢？别着急，我们下一期再聊。

Tomcat 的 Session 管理器提供了多种持久化方案来存储 Session，通常会采用高性能的存储方式，比如 Redis，并且通过集群部署的方式，防止单点故障，从而提升高可用。同时，Session 有过期时间，因此 Tomcat 会开启后台线程定期的轮询，如果 Session 过期了就将 Session 失效。

本期精华

HTTP 协议和其他应用层协议一样，本质上是一种通信格式。回到文章开头我问你的问题，其实答案很简单：HTTP 是通信的方式，HTML 才是通信的目的，就好比 HTTP 是信封，信封里面的信（HTML）才是内容；但是没有信封，信也没办法寄出去。HTTP 协议就是浏览器与服务器之间的沟通语言，具体交互过程是请求、处理和响应。

由于 HTTP 是无状态的协议，为了识别请求是哪个用户发过来的，出现了 Cookie 和 Session 技术。Cookie 本质上就是一份存储在用户本地的文件，里面包含了每次请求中都需要传递的信息；Session 可以理解为服务器端开辟的存储空间，里面保存的信息用于保持状态。作为 Web 容器，Tomcat 负责创建和管理 Session，并提供了多种持久化方案来存储 Session。

课后思考

在 HTTP/1.0 时期，每次 HTTP 请求都会创建一个新的 TCP 连接，请求完成后之后这个 TCP 连接就会被关闭。这种通信模式的效率不高，所以在 HTTP/1.1 中，引入了 HTTP 长连接的概念，使用长连接的 HTTP 协议，会在响应头加入 Connection:keep-alive。这样当浏览器完成一次请求后，浏览器和服务器之间的 TCP 连接不会关闭，再次访问这个服务器上的网页时，浏览器会继续使用这一条已经建立的连接，也就是说两个请求可能共用一个 TCP 连接。

今天留给你的思考题是，我在上面提到 HTTP 的特点是无状态的，多个请求之间是没有关系的，这是不是矛盾了？

不知道今天的内容你消化得如何？如果还有疑问，请大胆的在留言区提问，也欢迎你把你的课后思考和心得记录下来，与我和其他同学一起讨论。如果你觉得今天有所收获，欢迎你把它分享给你的朋友。

上一篇　01 | Web容器学习路径

下一篇　03 | 你应该知道的Servlet规范和Servlet容器

精选留言 (61)

写留言

吃饭饭 置顶

2019-05-13

13

我一直不太理解什么是无状态，restful经常听说是无状态的，是一个概念吗？求解答

作者回复: 我的理解是REST是一种架构风格：将网络上的信息实体看作是资源，可以是图片、文件、一个服务...资源用URI统一标识，URI中没有动词哦，这是因为它是资源的标识，那怎么操作这些资源呢，于是定义一些动作：GET、POST、PUT和DELETE。通过URI+动作来操作一个资源。所谓的无状态说的是，为了完成一个操作，请求里包含了所有信息，你可以理解为服务端不需要保存请求的状态，也就是不需要保存session，没有session的好处是带来了服务端良好的可伸缩性，方便failover，请求被LB转到不同的server实例上没有差别。从这个角度看，正是有了REST架构风格的指导，才有了HTTP的无状态特性，顺便提一下，REST和HTTP1.1出自同一人之手。但是理想是丰满的，现实是骨感的，为了方便开发，大多数复杂的Web应用不得不在服务端保存Session。为了尽量减少Session带来的弊端，往往将Session集中存储到Redis上，而不是直接存储在server实例上..

阿斯蒂芬

2019-05-13

48

Http的无状态我理解是指不同请求间协议内容无相关性，即本次请求与上次请求没有内容的依赖关系，本次响应也只针对本次请求的数据，至于服务器应用程序为用户保存的状态是属于应用层，与协议是无关的。
keep-alive表示tcp的连接可以复用，指的是利用已有的传输通道进行http协议内容的传输，省去创建/关闭连接的开销达到提升性能的效果。应用程序其实一般不关心这次Http...
展开

作者回复: 说的很清楚

Zach_

2019-05-13

22

http1.0: 买一个信封只能传送一个来回的信。
http1.1: keep–alive:买一个信封可以重复使用，但前提是得等到服务端把这个信封送回来。

作者回复: 优秀👍

许童童 17
2019-05-13

Connection:keep-alive只是建立TCP层的状态，省去了下一次的TCP三次握手，而HTTP本身还是继续保持无状态的特点。

刘为红 14
2019-05-14

sessionid是服务端生成的，服务端通过set-cookie放在http的响应头里，然后浏览器写到cookie里，后续每次请求就会自动带上来了，这点感觉讲得不是很清楚

展开

作者回复: 谢谢指出~

微信小助手 11
2019-05-13

HTTP的无状态性与共用TCP连接发送多个请求之间没有冲突，
这些请求之间相对独立，唯一的关系可能只有发送的先后顺序关系。
此外，HTTP/1.1中的长连接依然没有解决 head of line blocking 的问题，
后面的连接必须等待前面的返回了才能够发送，
这个问题直到HTTP/2.0采取二进制分帧编码方式才彻底解决。

展开

作者回复: 👍Tomcat和Jetty都支持HTTP2.0了

李海明 10
2019-05-14

无状态的协议，使用cookie、session等机制实现有状态的的web。
无状态是指协议对于事务处理没有记忆功能，对同一个url请求没有上下文关系，每次的请

求都是独立的，服务器中没有保存客户端的状态。HTTP协议长连接、短连接实质上是TCP协议的长连接、短连接。长连接省去了较多的TCP建立、关闭操作，减少了浪费，节约时间；短连接对于服务器来说管理较为简单，存在的连接都是有用的连接，不需要额外的...

展开

作者回复: 说的很好很详细👍

二两豆腐

2019-05-14

5

建立tcp链接是可以理解为先修路，每次建立推出链接也就是先把路给修好，路修好之后才能走在这条路上送信（http），keep-alive指的就是需要送信的时候要不要修路，还是在走已经修好的路上去送信，信是一封一封的送，上一封信和下一封信没啥联系，这就是无状态

逍遥哥哥

2019-05-14

4

老师，您好，现在的web容器都支持将session存储在第三方中间件（如redis）中，为什么很多公司喜欢绕过容器，直接在应用中将会话数据存入中间件中？

展开

作者回复: 用Web容器的Session方案需要侵入特定的Web容器，用Spring Session可能比较简单，不需要跟特定的Servlet容器打交道。

这正是Spring喜欢做的事情，它使得程序员甚至感觉不到Servlet容器的存在，可以专心开发Web应用。但是Spring到底做了什么，Spring Session是如何实现的，我们还是有必要了解了解~

其实它是通过Servlet规范中的Filter机制拦截了所有Servlet请求，偷梁换柱，将标准的Servlet请求对象包装了一下，换成它自己的Request包装类对象，这样当程序员通过包装后的Request对象的getSession方法拿Session时，是通过Spring拿Session，没Web容器什么事了。

Joker

2019-05-18

2

老师，我这样进行比喻您看看合不合适，原来的http老师，我这样进行比喻您看看合不合适。原来的http/1.0的时期，如果把每次发送http数据包都看成一次送信的过程的话，那

么就是每次发送都会新叫一个送信员（也就是新建一个TCP连接）。
http/1.1的长链接就相当于给了你和服务器和客户端有了专属的随时待命的送信员，你就免去了以前每次都要寻找送信员的过程。...

展开

作者回复: Joker你好，把TCP连接当做送信员的比喻很形象也很贴切！无状态表示每次寄信都是新的信封，这也是对的，我再多说几句，在服务端看来这些信没有关系的，并且服务端通过阅读这封信就得到了它要的全部信息，不需要从其他地方（比如Session里）来获取这封信的更多上下文信息，服务端就知道怎么处理和回信。

今夜秋风和
2019-05-14

2

服务端怎么检测这个tcp链接什么时候可以销毁释放？如果一个连接里面处理一个长事物，其他的请求会不会排队等待

作者回复: 服务端会设置连接超时时间，如果TCP连接上超过一段时间没有请求数据，服务端会关闭这个连接。

在HTTP1.1中，请求是按顺序排队处理的，前面的HTTP请求处理会阻塞后面的HTTP请求，虽然HTTP pipelining对连接请求做了改善，但是复杂度太大，并没有普及，这个问题在HTTP2.0中得到了解决。

Royal
2019-05-13

2

您好！上面提到的引入session是因为cookie存在客户端，有安全隐患；但是session id也是通过cookie由客户端发送到服务端，同样有安全隐患啊？

作者回复: 是的，虽然敏感的用户信息没有在网络上传输了，但是攻击者拿到sessionid也可以冒充受害者发送请求，这就是为什么我们需要https，加密后攻击者就拿不到sessionid了，另外CSRF也是一种防止session劫持的方式。

暮色听雨声
2019-05-13

2

HTTP 的特点是无状态的，多个请求之间是没有关系的，这是不矛盾了么？老师我对这句话不理解呀！ 怎么就矛盾了呀。多个HTTP请求之间本来就没有关系呀，关系跟状态怎么就会矛盾呢？

展开

作者回复: 多个请求之间共用一个TCP连接，是不是会让人觉得它们有关系呢。

八百

2019-05-26

1

老师我有二个问题
1.如果没请求中没有jsessionid ,每次发起http请求是否都会生成session，如果每次请求都生成session，那么是不是可以作为一个攻击的手段啊，让服务器存在大量session，导致oom
2.如果我从别人的浏览器中拿到jsessionid，把它放在我自己的请求头中，是不是在服务...

展开

作者回复: 1. 会不会产生session取决于Web应用如何实现，如果在用户没有登录的情况下调用Request.getSession(true),会产生Session，也就有你说的那个问题。
2.对的

Geek_28b75...

2019-05-24

1

老师，我们经常说的cookie跨域问题中，跨域是什么概念呢

展开

作者回复: cookie有两个重要属性：
domain字段 ： 表示浏览器访问这个域名时才带上这个cookie
path字段：表示访问的URL是这个path或者子路径时才带上这个cookie

跨域说的是，我们访问两个不同的域名或路径时，希望带上同一个cookie，跨域的具体实现方式有很多..

业余草
2019-05-21

1

这里没有讲，为什么 Tomcat 等要采用 HTTP，为什么不采用其他的一些协议？HTTP 协议的好处是什么？以及扩展到 HTTPS 上！

唐木儿
2019-05-15

1

对本期内容，作以下总结：
1.HTTP是服务器与浏览器之间的数据传输协议，规定了数据传输的格式，与数据如何传输无关。统一的数据格式规范有利于各端交互，而具体的交互方式不是协议做的事！
2.Cookie是存储在浏览器的信息，Session是存储在服务器端的信息。
3.浏览器和客户端之间的交互，主要是两个过程：建立连接和数据传输！建立连接需要经...
展开

Monday
2019-05-15

1

1，tomcat配置server.xml 中Connector标签的maxThreads就是思考题中提到的tcp连接数的最大值吧?
2，spring boot的配置文件*.yml中的 server.tomcat.max-connections也是tcp连接的最大值吧?
3，tomcat维护一个tcp连接池响应所有客户端的请求。虽然一个tcp请求可以同时和多个...
展开

作者回复: Connector标签的maxThreads是线程池中线程的个数，不是TCP连接数~

凌霄
2019-05-14

1

http是无状态的，理解是没有记忆存储功能，每次的请求都是幂等的，而长连接只是共享了tcp连接，跟java中的池技术一样，避免了重复的开销，节省效率，降低响应时间。
展开

青莲
2019-05-14

1

http无状态指的是：每次请求是一个全新上下文，请求不会包含上次请求的上下文信息，

这是应用层面的。而长连接在响应头加上connection:keep-alive 是告诉传输层tcp连接 保持多久才释放，这样当浏览器与服务端发起http请求时，会先看已存在的tcp连接，有的话则复用，没有则新建一个。