谈谈HTTP协议中的短轮询、长轮询、长连接和短连接

2016/04/18 • 基础技术 • 10 评论 • HTTP

分享到：
本文作者： 伯乐在线 - 左潇龙 。未经作者许可，禁止转载！
欢迎加入伯乐在线 专栏作者。

引言

最近刚到公司不到一个月，正处于熟悉项目和源码的阶段，因此最近经常会看一些源码。在研究一个项目的时候，源码里面用到了HTTP的长轮询。由于之前没太接触过，因此LZ便趁着这个机会，好好了解了一下HTTP的长长短短。

了解的方式主要都是LZ在网络上获取的，这里只是谈一下LZ对于这四种叫法最直观的理解。如果你之前不懂的话，可以帮你普及一下，如果你之前就懂得话，可以互相对照一下。

以前的误解

很久之前LZ就听说过长连接的说法，而且还知道HTTP1.0协议不支持长连接，从HTTP1.1协议以后，连接默认都是长连接。但LZ终究觉得对于长连接一直懵懵懂懂的，有种抓不到关键点的感觉。

今天LZ通过一番研究，终于明白了这其中的奥秘。而之前，LZ也看过长连接相关的内容，但一直都是云里雾里的。这次之所以能在这么短的时间里搞清楚，和LZ自己技术的沉淀密不可分。因此，这里LZ借着这个机会，再次强调一下，千万不要试图去研究你研究了很久都整不明白的东西，或许是你的层次不到，也或许是你从未在实际的应用场景接触过，这种情况下你去研究，只会事倍功半，徒劳一番罢了。

回到正题，既然说是误解，那么LZ的误解到底是什么？

那就是LZ一直认为，HTTP连接分为长连接和短连接，而我们现在常用的都是HTTP1.1，因此我们用的都是长连接。

这句话其实只对了一半，我们现如今的HTTP协议，大部分都是1.1的，因此我们平时用的基本上都是长连接。但是前半句是不对的，HTTP协议根本没有长短连接这一说，也正因为误解了这个，导致LZ对于长连接一直不明不白，始终不得其要领，具体下面一段会说到。

网络上很多文章都是误人子弟，根本没有说明白这个概念。这里LZ要强调一下，HTTP协议是基于请求/响应模式的，因此只要服务端给了响应，本次HTTP连接就结束了，或者更准确的说，是本次HTTP请求就结束了，根本没有长连接这一说。那么自然也就没有短连接这一说了。

之所以网络上说HTTP分为长连接和短连接，其实本质上是说的TCP连接。TCP连接是一个双向的通道，它是可以保持一段时间不关闭的，因此TCP连接才有真正的长连接和短连接这一说。

其实知道了以后，会觉得这很好理解。HTTP协议说到底是应用层的协议，而TCP才是真正的传输层协议，只有负责传输的这一层才需要建立连接。

一个形象的例子就是，拿你在网上购物来说，HTTP协议是指的那个快递单，你寄件的时候填的单子就像是发了一个HTTP请求，等货物运到地方了，快递员会根据你发的请求把货物送给相应的收货人。而TCP协议就是中间运货的那个大货车，也可能是火车或者飞机，但不管是什么，它是负责运输的，因此必须要有路，不管是地上还是天上。那么这个路就是所谓的TCP连接，也就是一个双向的数据通道。

因此，LZ现在甚至觉得，“HTTP连接”这个词就不应该出现，它只是一个应用层的协议，根本就没有所谓的连接这一说，就像FTP也是应用层的协议，但是你有听说过FTP连接吗？（恩，好像是听过，-_-，但你现在知道了，其实所谓的FTP连接，严格来说，依旧是TCP连接）

实际上，说HTTP请求和HTTP响应会更准确一些，而HTTP请求和HTTP响应，都是通过TCP连接这个通道来回传输的。

不管怎么说，一定要务必记住，长连接是指的TCP连接，而不是HTTP连接。

一个疑问

之前LZ一直对一件事有些模糊不清，首先是怎么样就算是把HTTP变成长连接了，是不是只要设置Connection为keep-alive就算是了？

如果是的话，那都说HTTP1.1默认是长连接，而观察我们平时开发的Web应用的HTTP头部，Connection也确实是keep-alive，那就是说我们大部分都是用的长连接，但是长连接不是一般用于交互比较频繁的应用吗？像我们这种普通的Web应用，比如博客园这种，或者我的个人博客这种，长连接有什么用？

如果有用那用处到底是什么，我们又不是客户端与服务器交互频繁的那种应用（毕竟你打开网页肯定要半天才打开另外一个吧），如果没用的话，那到底应不应该把Connection为keep-alive这个header值给改掉，从而改成短连接？

这个疑问，在LZ明白了长连接其实是指的TCP连接之后，基本上就明白了。而这个疑问，也正是LZ在“以前的误解”那一段所提到的，那个因为误解导致LZ一直搞不明白的问题。

为什么解决了上面那个误解之后，前面所说的这些疑问LZ都明白了？

因为长连接意味着连接会被复用，毕竟一直保持着连接不就是为了重复使用嘛。但如果长连接是指的HTTP的话，那就是说HTTP连接可以被重复利用，这个话听起来就感觉很别扭。之所以觉得别扭，其实就是LZ的一种直觉，没什么理论依据。而这种别扭的根源就在于，之前一直没有融会贯通的感觉，所以总感觉缺少点什么。不过这点疑惑，并没有影响LZ的工作，因此也就没深究过。

但现在好了，明白了长连接实际上是指的TCP连接，LZ瞬间自己就想明白了上面的那些问题。

第一个问题是，是不是只要设置Connection为keep-alive就算是长连接了？

当然是的，但要服务器和客户端都设置。

第二个问题是，我们平时用的是不是长连接？

这个也毫无疑问，当然是的。（现在用的基本上都是HTTP1.1协议，你观察一下就会发现，基本上Connection都是keep-alive。而且HTTP协议文档上也提到了，HTTP1.1默认是长连接，也就是默认Connection的值就是keep-alive）

第三个问题，也是LZ之前最想不明白的问题，那就是我们这种普通的Web应用（比如博客园，我的个人博客这种）用长连接有啥好处？需不需要关掉长连接而使用短连接？

这个问题LZ现在终于明白了，问题的答案是好处还是有的。

好处是什么？

首先，刚才已经说了，长连接是为了复用，这个在之前LZ就明白。那既然长连接是指的TCP连接，也就是说复用的是TCP连接。那这就很好解释了，也就是说，长连接情况下，多个HTTP请求可以复用同一个TCP连接，这就节省了很多TCP连接建立和断开的消耗。

比如你请求了博客园的一个网页，这个网页里肯定还包含了CSS、JS等等一系列资源，如果你是短连接（也就是每次都要重新建立TCP连接）的话，那你每打开一个网页，基本要建立几个甚至几十个TCP连接，这浪费了多少资源就不用LZ去说了吧。

但如果是长连接的话，那么这么多次HTTP请求（这些请求包括请求网页内容，CSS文件，JS文件，图片等等），其实使用的都是一个TCP连接，很显然是可以节省很多消耗的。

这样一解释，就很明白了，不知道大家看了这些解释感觉如何，反正LZ在自己想明白以后，有种豁然开朗的感觉。

另外，最后关于长连接还要多提一句，那就是，长连接并不是永久连接的。如果一段时间内（具体的时间长短，是可以在header当中进行设置的，也就是所谓的超时时间），这个连接没有HTTP请求发出的话，那么这个长连接就会被断掉。

这一点其实很容易理解，否则的话，TCP连接将会越来越多，直到把服务器的TCP连接数量撑爆到上限为止。现在想想，对于服务器来说，服务器里的这些个长连接其实很有数据库连接池的味道，大家都是为了节省连接重复利用嘛，对不对？

长轮询和短轮询

前面基本上LZ已经把长短连接说的差不多了，接下来说说长短轮询，今天也正是为了研究长短轮询，LZ才顺便研究了下长短连接这回事。

短轮询相信大家都不难理解，比如你现在要做一个电商中商品详情的页面，这个详情界面中有一个字段是库存量（相信这个大家都不陌生，随便打开淘宝或者京东都能找到这种页面）。而这个库存量需

要实时的变化，保持和服务器里实际的库存一致。

这个时候，你会怎么做？

最简单的一种方式，就是你用JS写个死循环，不停的去请求服务器中的库存量是多少，然后刷新到这个页面当中，这其实就是所谓的短轮询。

这种方式有明显的坏处，那就是你很浪费服务器和客户端的资源。客户端还好点，现在PC机配置高了，你不停的请求还不至于把用户的电脑整死，但是服务器就很蛋疼了。如果有1000个人停留在某个商品详情页面，那就是说会有1000个客户端不停的去请求服务器获取库存量，这显然是不合理的。

那怎么办呢？

长轮询这个时候就出现了，其实长轮询和短轮询最大的区别是，短轮询去服务端查询的时候，不管库存量有没有变化，服务器就立即返回结果了。而长轮询则不是，在长轮询中，服务器如果检测到库存量没有变化的话，将会把当前请求挂起一段时间（这个时间也叫作超时时间，一般是几十秒）。在这个时间里，服务器会去检测库存量有没有变化，检测到变化就立即返回，否则就一直等到超时为止。

而对于客户端来说，不管是长轮询还是短轮询，客户端的动作都是一样的，就是不停的去请求，不同的是服务端，短轮询情况下服务端每次请求不管有没有变化都会立即返回结果，而长轮询情况下，如果有变化才会立即返回结果，而没有变化的话，则不会再立即给客户端返回结果，直到超时为止。

这样一来，客户端的请求次数将会大量减少（这也就意味着节省了网络流量，毕竟每次发请求，都会占用客户端的上传流量和服务端的下载流量），而且也解决了服务端一直疲于接受请求的窘境。

但是长轮询也是有坏处的，因为把请求挂起同样会导致资源的浪费，假设还是1000个人停留在某个商品详情页面，那就很有可能服务器这边挂着1000个线程，在不停检测库存量，这依然是有问题的。

因此，从这里可以看出，不管是长轮询还是短轮询，都不太适用于客户端数量太多的情况，因为每个服务器所能承载的TCP连接数是有上限的，这种轮询很容易把连接数顶满。之所以举这个例子，只是因为大家肯定都会网购，所以这个例子比较通俗一点。

哪怕轮询解决不了获取库存这个问题，但只要大家明白了长短轮询的区别，这就足够了。实际上，据LZ自己平日里购物的观察，那个库存量应该是不会变的，这个例子纯属LZ个人的意淫，-_-。

长短轮询和长短连接的区别

这里简单说一下它们的区别，LZ这里只说最根本的区别。

第一个区别是决定的方式，一个TCP连接是否为长连接，是通过设置HTTP的Connection Header来决定的，而且是需要两边都设置才有效。而一种轮询方式是否为长轮询，是根据服务端的处理方式来决定的，与客户端没有关系。

第二个区别就是实现的方式，连接的长短是通过协议来规定和实现的。而轮询的长短，是服务器通过编程的方式手动挂起请求来实现的。

结语

好了，本文就到此为止吧。LZ写这篇文章，主要也是为了避免自己遗忘。说实话，写到最后了，LZ感觉对于它们的理解又进了一步，这就是写博客的好处吧。

> 打赏支持我写出更多好文章，谢谢！
>
> 打赏作者

关于作者：左潇龙

一个坚信技术改变世界的屌丝程序猿。交流一群【已满】：300638185 -----交流二群【已满】：475044650 ------交流三群：367659782【新开】

个人主页 • 我的文章 • 119 •

相关文章

- HTTP 缓存
- 99%的人都理解错了HTTP中GET与POST的区别 • 11
- HTTP 协议入门 • 1
- 刨根问底HTTP和WebSocket协议 • 1
- 选择一个 HTTP 状态码不再是一件难事 • 1

可能感兴趣的话题

- iOS10版本兼容性测试有模拟器吗
- 程序员们,来谈谈你工作中使用到数据结构与算法知识的经历吧!
- 在Mysql中查询数据库中每个月的最新的数据，并按月份分组. • 1
- Android官网交互八大要点
- PHP的Yii框架中移除组件所绑定的行为的方法
- 大家感觉技术工程师转销售工程师或者应用或者运营怎么样? • 2

登录后评论 新用户注册

直接登录

最新评论

梦依旧 （ 1 ） 04/19

赞。。。写得不错。。以前也对这些概念比较模糊。。瞬间启迪了我，谢谢楼主

赞 回复

sleep （ 1 ） 04/20

误人子弟

赞 回复

伯小乐 （ 215 ） 04/20
小编

如果发现不对的地方，请直接指出吧

赞 回复

左潇龙 （ 119 • ） 04/21
Java程序猿

估计你所说的和下面这位张书艾是指的一件事，我就不重复解释了。

赞 回复

张书艾 （ 3 ）
游戏服务器程序
04/21

不太赞同，http最初就是网站用的协议。而网站因为浏览次数会非常大，单客户端（浏览器）可能只浏览一次，所以没有必要保持一个连接，来节省网站主机被频繁申请socket连接所产生的性能消耗。为什么会有keep-alive，那是因为往往浏览一个网站时，可能多次发送tcp协议包，一方面可以让客户端（浏览器）上的程序设计更加灵活，一方面可以减少httpd服务的压力。http keep-alive与tcp keep-alive，不是同一回事，意图不一样。http keep-alive是为了让tcp活得更久一点，以便在同一个连接上传送多个http，提高socket的效率。而tcp keep-alive是TCP的一种检测TCP连接状况的保鲜机制。tcp keep-alive对于及时性比较高的游戏服务器来说不够用（过长的保活无法及时探知客户端是否依然alive），也因此有了心跳这个特有的逻辑保活包。

1 赞 回复

左潇龙 （ 119 • ）
Java程序猿
04/21

你说的没错，我在这个地址找到了和你一模一样的话，http://blog.csdn.net/lys86_1205/article/details/21234867。
但是我这篇文章，归根结底成一句话其实就是HTTP的keep-alive并不是保持了“HTTP连接alive”，而是保持的“TCP连接alive”。个人觉得这与你的并不冲突。换句话说，我这里只解释了HTTP的keep-alive，也就是把connection这个header设置成keep-alive会发生什么。
至于你的说TCP的keep-alive，我看那个文章里写的是要更改/proc/sys/net/ipv4这个下面的东西，和我文章所说的不是一回事。

1 赞 回复

张书艾 （ 3 ）
游戏服务器程序
04/21

大半夜看到的这文章，我实在懒的解释keep-alive在http的作用了，我就搜了一个解释的比较清晰的说法（给你点个赞）。我想解释的，http是短连接并不是错误的说法。它设计的方式就是短连接的应用层协议，至于为什么有keep-alive，只是连接优化并且有性能提升，是http协议发展过程中的一个更新。我感觉这篇文章有些误导人，并且理解上有些错误，怕其他人把http keep-alive与TCP的混淆。

赞 回复

左潇龙 （ 119 • ）
Java程序猿
04/21

说到底，你就是怕别人把我解释的HTTP的keep-alive，当成TCP的keep-alive。
但是我想说的是，你全文上下有看到我提到TCP的keep-alive这回事儿吗。。。。0-0。我一直在说的就是HTTP长连接，http的connection设置成keep-alive这些词汇，真不知道你是如何联想到TCP的keep-alive的。。。
不过也谢谢你把，也是好心一片。但说实话，你解释了这么多压根儿跟文章内容没关系的东西，一定会把一群人整的更懵逼。我好不容易写出来一个扫盲贴，就这么被破坏了，0-0。

赞 回复

张书艾 （ 3 ）
游戏服务器程序
04/21

下次我会直接回复他：你搜一下TCP keep-alive 与HTTP keep-alive的区别，一定要分清楚。在我做过的服务器里，如果HTTP想做推送，用到的一些方法，我认为还是不太能称为一个长连接应该有的样子。所以说，说HTTP是短连接协议，没问题，如果你想让HTTP支持长连接可以server推送东西到client，搜一下HTTP 推送，应该也都是退化到keep-alive这个新增特性得出的方式。

赞 回复

人非草木 （ 1 ）
程序猿
05/02

谢谢博主，豁然开朗！

赞 回复

hai^O^ 发起 ◆6◆1 115 回复

项目中有大量的Ajax回调时如何避免...
chaoxiaoliunian 发起 ◆6◆1 1 回复

UI设计,周末想报个前端开发培训班
小狼阿亮 发起 ◆6◆1 1 回复

想找一份web前端的工作需要准备什么...
。 发起 ◆6◆1 12 回复

前端自学真的很难找工作吗啊?
一抹茶 发起 ◆6◆1 26 回复

- 本周热门前端文章
- 本月热门
- 热门标签

0 Web 前端从入门菜鸟到实践老司机所...

1 JavaScript 时间与日期处理实战:...

2 2016:编写高性能的 JavaScript

3 如何写出漂亮的 React 组件

4 Chrome开发者工具详解(3):Timeli...

5 前后端分离开发模式的 mock 平台预研

6 正则表达式理论篇

7 Chrome开发者工具详解(4):Profil...

8 HTML也可以静态编译?

9 Node.js + Web Socket 打造...

0 前端面试题整理汇总

1 Web 前端从入门菜鸟到实践老司机所...

2 一道 JS 面试题引发的思考

3 手把手教你打造一个纯CSS图标库

4 前端跨域知识总结

5 ES6 的 12 个核心功能一览

6 专治前端焦虑的学习方案

7 你不知道的 Javascript

8 JavaScript在浏览器上的调试技巧

9 聊聊响应式图片

3D Ajax angular AngularJS apply async Babel BFC bind box-shadow Canvas Chrome Chrome开发者工具 clip-path Console CSS CSS3 CSS Reset DOM ECMAScript 6 es6 facebook featuredpost float gulp HTML HTML5 HTTP HTTP/2 HTTPS Javascript jQuery JSON MVC MVVM node node.js NodeJS npm Promise React ReactJS rem Service Worker settimeout SVG this underscore URL Virtual DOM WEB WebGL webpack 书籍 事件 作用域 函数式编程 前端 前端工程师 前端框架 动画 原型 变量 响应式 图片 字体 对象 层叠上下文 居中 工具 异步 性能 性能优化 插件 数组 框架 模块 模块化 模板引擎 正则表达式 测试 浏览器 移动端 算法 组件 组件化 继承 缓存 职场 自适应 表单 调试 跨域 运算符 选择器 重构 闭包 面试 预加载 高性能

业界热点资讯 **更多 »**

知名黑客 George Hotz 开源了自动驾驶软件
19 小时前 • 6

俄罗斯政府将采用 Sailfish 智能手机操作系统
19 小时前 • 3

微软希望所有 Linux 开发人员迁移到 Windows 10
3 天前 • 23 • 6

5 分钟回顾 Linux 25 年的发展历程与变迁
6 天前 • 13

微软警告用户不要去随意修改 Linux 文件
11/25 • 28

PyCharm 2016.3 发布
11/25 • 14

前端工具资源 **更多资源 »**

Electron：用Html、CSS和JavaScript构建跨平台的客...
MVC框架和库

Velocity.js：加速JavaScript动画
动画

three.js：JavaScript 3D 库
JavaScript, Web 数据可视化工具

jquery.transit：提供流畅CSS3变换和过渡效果的jQuer...
JavaScript, 动画

bounce.js：创建有趣的CSS3动画
JavaScript, 动画

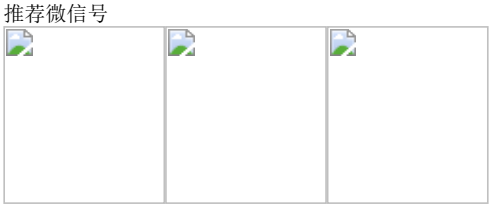
关于伯乐前端

伯乐前端分享Web前端开发，包括JavaScript，CSS和HTML5开发技术，前端相关的行业动态。

快速链接
网站使用指南 ◆0◆3
加入我们 ◆0◆3
问题反馈与求助 ◆0◆3
网站积分规则 ◆0◆3
网站声望规则 ◆0◆3

关注我们

新浪微博：@前端大全
RSS：订阅地址
推荐微信号



合作联系
Email：bd@Jobbole.com
QQ： 2302462408 （加好友请注明来意）

更多频道

小组 - 好的话题、有启发的回复、值得信赖的圈子
头条 - 分享和发现有价值的内容与观点
相亲 - 为IT单身男女服务的征婚传播平台
资源 - 优秀的工具资源导航
翻译 - 翻译传播优秀的外文文章
文章 - 国内外的精选文章
设计 - UI,网页，交互和用户体验
iOS - 专注iOS技术分享
安卓 - 专注Android技术分享
前端 - JavaScript, HTML5, CSS
Java - 专注Java技术分享
Python - 专注Python技术分享