

44 | 先睹为快：HTTP/3实验版本长什么样子？

2021-03-01 罗剑锋

《透视HTTP协议》

课程介绍 >



讲述：罗剑锋

时长 09:32 大小 8.75M



你好，我是 Chrono。

不知不觉，《透视 HTTP 协议》这个专栏马上就要两周岁了。前几天我看了一下专栏的相关信息，订阅数刚好破万，非常感谢包括你在内的所有朋友们的关心、支持、鼓励和鞭策。

在专栏的结束语里我曾经说过，希望 HTTP/3 发布之时能够再相会。而如今虽然它还没有发布，但也为时不远了。

所以今天呢，我就来和你聊聊 HTTP/3 的一些事，就当是“尝尝鲜”吧。

HTTP/3 的现状

从 2019 到 2021 的这两年间，大家对 HTTP 协议的关注重点差不多全都是放在 HTTP/3 标准的制订上。

领资料



最初专栏开始的时候，HTTP/3 草案还是第 20 版，而现在则已经是第 34 版了，发展的速度可以说是非常快的，里面的内容也变动得非常多。很有可能最多再过一年，甚至是今年内，我们就可以看到正式的标准。

在标准文档的制订过程中，互联网业届也没有闲着，也在积极地为此做准备，以草案为基础做各种实验性质的开发。

这其中比较引人瞩目的要数 CDN 大厂 Cloudflare，还有 Web Server 领头羊 Nginx（而另一个 Web Server Apache 好像没什么动静）了。

Cloudflare 公司用 Rust 语言编写了一个 QUIC 支持库，名字叫“quiche”，然后在上面加了一层薄薄的封装，由此能够以一个 C 模块的形式加入进 Nginx 框架，为 Nginx 提供了 HTTP/3 的功能。（可以参考这篇文章：[🔗HTTP/3：过去，现在，还有未来](#)）

不过 Cloudflare 的这个 QUIC 支持库属于“民间行为”，没有得到 Nginx 的认可。Nginx 的官方 HTTP/3 模块其实一直在“秘密”开发中，在去年的 6 月份，这个模块终于正式公布了，名字就叫“http_v3_module”。（可以参考这篇文章：[🔗Introducing a Technology Preview of NGINX Support for QUIC and HTTP/3](#)）

目前，http_v3_module 已经度过了 Alpha 阶段，处于 Beta 状态，但支持的草案版本是 29，而不是最新的 34。

这当然也有情可原。相比于 HTTP/2，HTTP/3 的变化太大了，Nginx 团队的精力还是集中在核心功能实现上，选择一个稳定的版本更有利于开发，而且 29 后面的多个版本标准其实差异非常小（仅文字编辑变更）。

Nginx 也为测试 HTTP/3 专门搭建了一个网站：[🔗quic.nginx.org](#)，任何人都可以上去测试验证 HTTP/3 协议。

所以，接下来我们就用它来看看 HTTP/3 到底长什么样。

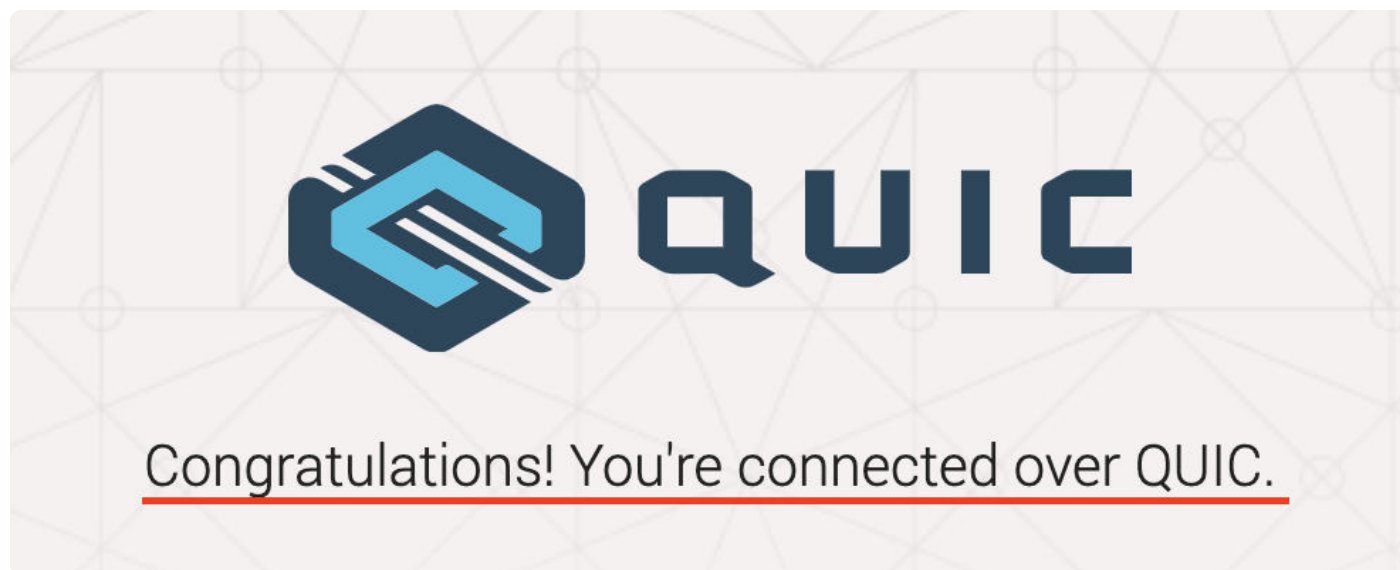
初识 HTTP/3

在体验之前，得先说一下浏览器，这是测试 QUIC 和 HTTP/3 的关键：最好使用最新版本的 Chrome 或者 Firefox，这里我用的是 Chrome88。

领资料

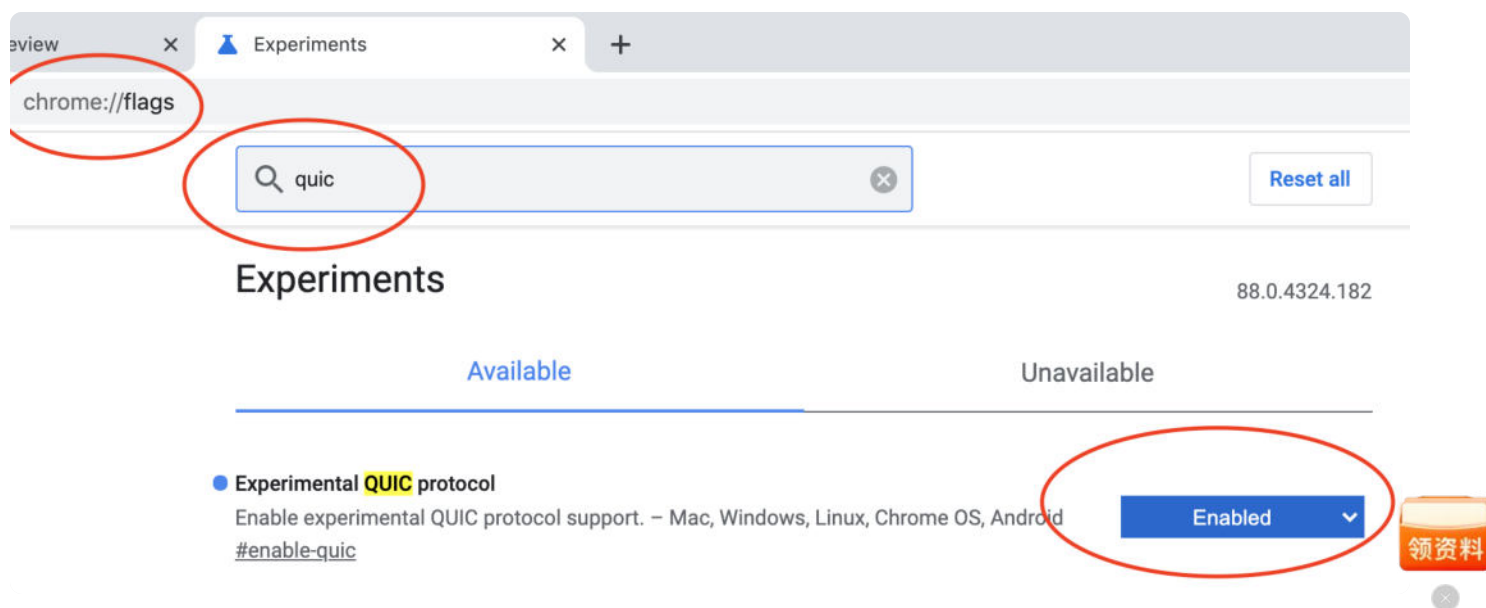


打开浏览器窗口，输入测试网站的 URI（<https://quic.nginx.org/>），如果“运气好”，刷新几次就能够在网页上看到大大的 QUIC 标志了。



不过你很可能“运气”没有这么好，在网页上看到的 QUIC 标志是灰色的。这意味着暂时没有应用 QUIC 和 HTTP/3，这就需要对 Chrome 做一点设置，开启 QUIC 的支持。

首先要在地址栏输入“chrome://flags”，打开设置页面，然后搜索“QUIC”，找到启用 QUIC 的选项，把它改为“Enabled”，具体可以参考下面的图片。



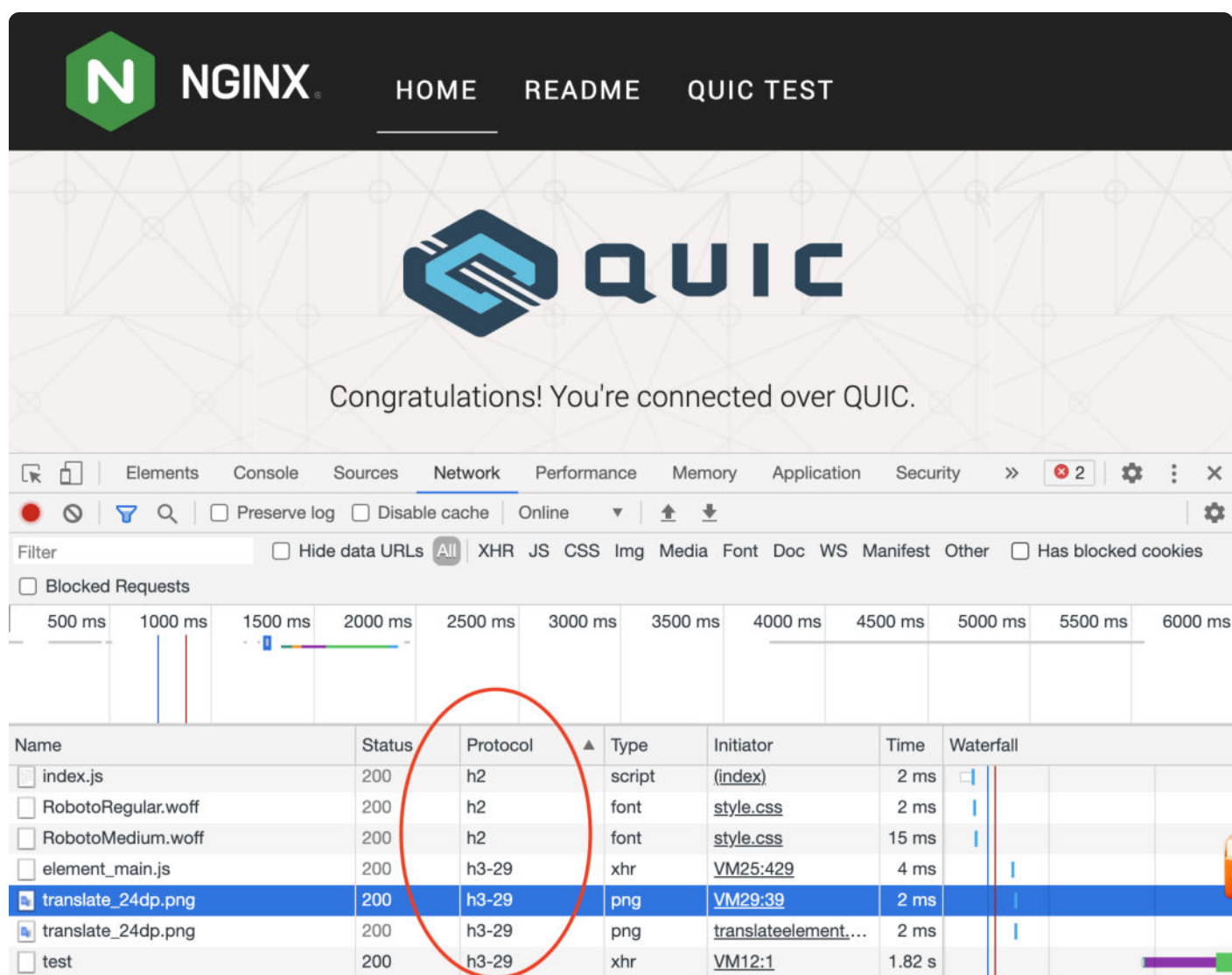
接下来，我们要在命令行里启动 Chrome 浏览器，在命令行里传递“enable-quic”“quic-version”等参数来要求 Chrome 使用特定的草案版本。

下面的示例就是我在 macOS 上运行 Chrome 的命令行。你也可以参考 Nginx 网站上的 README 文档，在 Windows 或者 Linux 上用类似的形式运行 Chrome 的命令行：

```
1 /Applications/Google\ Chrome.app/Contents/MacOS/Google\ Chrome \  
2 --enable-quic --quic-version=h3-29 \  
3 --origin-to-force-quic-on=quic.nginx.org:443
```

如果这样操作之后网页上仍然是显示灰色标志也不要紧，你还可以用“F12”打开 Chrome 的开发者工具面板，查看 protocol 一栏。

应该可以看到大部分资源显示的还是“h2”，表示使用的是 HTTP/2 协议，但有一小部分资源显示的是“h3-29”，这就表示它是使用 HTTP/3 协议传输的，而后面的“29”后缀，意思是基于第 29 版草案，也就是说启用了 QUIC+HTTP/3 协议。



Name	Status	Protocol	Type	Initiator	Time	Waterfall
index.js	200	h2	script	(index)	2 ms	
RobotoRegular.woff	200	h2	font	style.css	2 ms	
RobotoMedium.woff	200	h2	font	style.css	15 ms	
element_main.js	200	h3-29	xhr	VM25:429	4 ms	
translate_24dp.png	200	h3-29	png	VM29:39	2 ms	
translate_24dp.png	200	h3-29	png	translateelement...	2 ms	
test	200	h3-29	xhr	VM12:1	1.82 s	

Wireshark 抓包分析

好了，大概看了 HTTP/3 是什么样，有了感性认识，我们就可以进一步来抓包分析。

网络抓包工具 Wireshark 你一定已经比较熟悉了，这里同样要用最新的，不然可能识别不了 QUIC 和 HTTP/3 的数据包，比如我用的就是 3.4.3。

QUIC 的底层是 UDP，所以在抓包的时候过滤器要设置成“udp port 443”，然后启动就可以了。这次我抓的包也放到了 GitHub 的 [Wireshark 目录](#)，文件名是“44-1.pcapng”。

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.59.95.87	35.214.218.230	QUIC	1392	Initial, DCID=a1dba1db0077e2fd,
2	0.260861	35.214.218.230	10.59.95.87	QUIC	149	Retry, SCID=3ae893fa047246e55f9
3	1.070729	10.59.95.87	35.214.218.230	QUIC	1392	Initial, DCID=3ae893fa047246e55
4	1.338036	35.214.218.230	10.59.95.87	QUIC	1288	Handshake, SCID=5bd6b722142f863
5	1.338039	35.214.218.230	10.59.95.87	QUIC	1288	Handshake, SCID=5bd6b722142f863

▶ User Datagram Protocol, Src Port: 64738, Dst Port: 443

▼ QUIC IETF

- ▶ QUIC Connection information
[Packet Length: 1350]
1... = Header Form: Long Header (1)
.1.. = Fixed Bit: True
..00 = Packet Type: Initial (0)
.... 00.. = Reserved: 0
.... ..00 = Packet Number Length: 1 bytes (0)
Version: draft-29 (0xff00001d)
Destination Connection ID Length: 20
Destination Connection ID: 3ae893fa047246e55f963ea14fc5ecac3774f61e
Source Connection ID Length: 0
Token Length: 64
Token: 6f5335c14adf24b863f110bb5579f246ebef485ee47bb9b8b9cdd5a51229fc4824c2310e...
Length: 1255
Packet Number: 3
Payload: 16c4d24511f2043755c9464fa6f0fc0e1aac1c958fe5ffd70cae18fcfc21da3f2cdcf6a5...
- ▼ TLSv1.3 Record Layer: Handshake Protocol: Client Hello
Frame Type: CRYPTO (0x0000000000000006)
Offset: 0
Length: 309
Crypto Data
- ▶ Handshake Protocol: Client Hello

因为 HTTP/3 内置了 TLS 加密（可参考之前的 [第 32 讲](#)），所以用 Wireshark 抓包后看到的数据大部分都是乱码，想要解密看到真实数据就必须设置 SSLKEYLOG（参考 [第 26 讲](#)）。

不过非常遗憾，不知道是什么原因，虽然我导出了 SSLKEYLOG，但在 Wireshark 里还是无法解密 HTTP/3 的数据，显示解密错误。但同样的设置和操作步骤，抓包解密 HTTPS 和 HTTP/2 却是正常的，估计可能是目前 Wireshark 自身对 HTTP/3 的支持还不太完善吧。

所以今天我也就只能带你一起来看 QUIC 的握手阶段了。这个其实与 TLS1.3 非常接近，只不过是内嵌在了 QUIC 协议里，如果你学过了“安全篇”“飞翔篇”的课程，看 QUIC 应该是不会费什么力气。

首先我们来看 Header 数据：

领资料


```

1 [Packet Length: 1350]
2 1... .. = Header Form: Long Header (1)
3 .1... .. = Fixed Bit: True
4 ..00 .... = Packet Type: Initial (0)
5 .... 00.. = Reserved: 0
6 .... ..00 = Packet Number Length: 1 bytes (0)
7 Version: draft-29 (0xff00001d)
8 Destination Connection ID Length: 20
9 Destination Connection ID: 3ae893fa047246e55f963ea14fc5ecac3774f61e
10 Source Connection ID Length: 0

```

QUIC 包头的第一个字节是标志位，可以看到最开始建立连接会发一个长包（Long Header），包类型是初始化（Initial）。

标志位字节后面是 4 字节的版本号，因为目前还是草案，所以显示的是“draft-29”。再后面，是 QUIC 的特性之一“连接 ID”，长度为 20 字节的十六进制字符串。

这里我要特别提醒你注意，因为标准版本的演变，这个格式已经与当初 [第 32 讲](#) 的内容（draft-20）完全不一样了，在分析查看的时候一定要使用 [对应的 RFC 文档](#)。

往下再看，是 QUIC 的 CRYPTO 帧，用来传输握手消息，帧类型是 0x06：

```

1 TLSv1.3 Record Layer: Handshake Protocol: Client Hello
2   Frame Type: CRYPTO (0x0000000000000006)
3   Offset: 0
4   Length: 309
5   Crypto Data
6   Handshake Protocol: Client Hello

```

CRYPTO 帧里的数据，就是 QUIC 内置的 TLS “Client Hello”了，我把里面的一些重要信息摘了出来：

```

1 Handshake Protocol: Client Hello
2   Handshake Type: Client Hello (1)
3   Version: TLS 1.2 (0x0303)
4   Random: b4613d...
5   Cipher Suites (3 suites)

```


```

6      Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)
7      Cipher Suite: TLS_AES_256_GCM_SHA384 (0x1302)
8      Cipher Suite: TLS_CHACHA20_POLY1305_SHA256 (0x1303)
9      Extension: server_name (len=19)
10     Type: server_name (0)
11     Server Name Indication extension
12         Server Name: quic.nginx.org
13     Extension: application_layer_protocol_negotiation (len=8)
14     Type: application_layer_protocol_negotiation (16)
15     ALPN Protocol
16         ALPN Next Protocol: h3-29
17     Extension: key_share (len=38)
18     Key Share extension
19     Extension: supported_versions (len=3)
20     Type: supported_versions (43)
21     Supported Version: TLS 1.3 (0x0304)

```

你看，这个就是标准的 TLS1.3 数据（伪装成了 TLS1.2），支持 AES128、AES256、CHACHA20 三个密码套件，SNI 是“quic.nginx.org”，ALPN 是“h3-29”。

浏览器发送完 Initial 消息之后，服务器回复 Handshake，用一个 RTT 就完成了握手，包的格式基本一样，用了一个 CRYPTO 帧和 ACK 帧，我就不细分析了（可参考 [相应的 RFC](#)），只贴一下里面的“Server Hello”信息：

 复制代码

```

1 Handshake Protocol: Server Hello
2     Handshake Type: Server Hello (2)
3     Version: TLS 1.2 (0x0303)
4     Random: d6aede...
5     Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)
6     Extension: key_share (len=36)
7         Key Share extension
8     Extension: supported_versions (len=2)
9         Type: supported_versions (43)
10        Supported Version: TLS 1.3 (0x0304)

```

 领资料

这里服务器选择了“TLS_AES_128_GCM_SHA256”，然后带回了随机数和 key_share，完成了握手阶段的密钥交换。

小结

好了，QUIC 和 HTTP/3 的“抢鲜体验”就到这里吧，我简单小结一下今天的要点：

1. HTTP/3 的最新草案版本是 34，很快就会发布正式版本。
2. Nginx 提供了对 HTTP/3 的实验性质支持，目前是 Beta 状态，只能用于测试。
3. 最新版本的 Chrome 和 Firefox 都支持 QUIC 和 HTTP/3，但可能需要一些设置工作才能启用。
4. 访问专门的测试网站“quic.nginx.org”可以检查浏览器是否支持 QUIC 和 HTTP/3。
5. 抓包分析 QUIC 和 HTTP/3 需要用最新的 Wireshark，过滤器用 UDP，还要导出 SSLKEYLOG 才能解密。

希望你看完这一讲后自己实际动手操作一下，访问网站再抓包，如果能正确解密 HTTP/3 数据，就把资料发出来，和我们分享下。

如果你觉得有所收获，也欢迎把这一讲的内容分享给你的朋友。

领资料





== 课外小贴士 ==

- 01 大概在半年前，我在极客时间的“部落”里发了一些文字，介绍了Cloudflare和Nginx的HTTP/3开发进展，感兴趣的同学可以去翻一下“故纸堆”。
- 02 Chrome的不同版本对HTTP/3和QUIC的支持程度差异较大，像用“88.0.4324.150”这个版本，不需要什么设置就可以直接QUIC成功，而更新到稍后的“88.0.4324.182”反而就不行了，必须要做麻烦的设置。
- 03 为了防止中间盒（Middle Box）修改，QUIC协议要求握手时初始包也要做加密，不能直接在QUIC包里看到TLS1.3的记录，不过这个算法很简单，所以Wireshark是可以解密的。
- 04 Nginx的QUIC测试网站上还有一个专门的“QUIC TEST”链接，可以用来测试浏览器的QUIC传输性能。

分享给需要的人，Ta订阅超级会员，你将得 50 元

Ta单独购买本课程，你将得 20 元

生成海报并分享

赞 9 提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 43 | 如何进行Docker实验环境搭建？

下一篇 结束语 | 做兴趣使然的Hero

学习推荐

JVM + NIO + Spring

各大厂面试题及知识点详解

限时免费



精选留言 (6)



CC

2021-03-02

惊喜，刚刚好在学习第二遍，看到 HTTP/3 的新文章更新。谢谢老师。

作者回复: 学而时习之，不亦乐乎。

领资料

写留言



**水手辛伯达**

2021-05-17

第二遍学习Chrono老师的http协议。这门课分层清晰，环环相扣，由简入繁，对于初中级前端和运维人员去了解http协议帮助是比较大的！最难能可贵的是，老师前后两年时间一直坚持更新，又增加了docker试验环节和http3的发展updates，同时，老师十分注意和学员的互动，而且几乎和每个留言进行点评和分析，这些课后答疑又极大地丰富了大家的知识，增长了经验，十分庆幸在极客时间里面遇到这么优秀的课程！

祝老师健康，顺利！

作者回复: 感谢支持，大家共同进步。



👍 1

**Omooo**

2021-11-27

牛逼！



👍

**阿斯蒂芬**

2021-08-19

为老师对课程的持续关注和技术更新的科普点赞！

作者回复: Thanks



👍 1

**Luca**

2021-03-23

更新到了Chrome89.0.4389.90，无需进行额外设置就能够有QUIC支持了。

作者回复: Chrome的行为确实令人迷惑，有的版本就不行，不过相信以后会越来越简单方便。



领资料

**乘风破浪**

2021-03-02

实测chrome版本 88.0.4324.190（正式版本）无需设置可以支持quic

firefox 86.0需要简单设置一下，具体页面搜firefox，第一条就是

wireshark3.4.3抓包结果和大师一样，payload解不出来

请问大师，现在学习HTTP/3，现在如果要深入了解HTTP/3,需要看rfc吧？有没有其他好的资源？

作者回复: 我现在更新到88.0.4324.192，就不能直接显示出quic支持了。

目前HTTP/3的资料还只有rfc，不过也不用太着急，等正式发布后估计就会有其他很多的分析研究资料了。

共 2 条评论 >



领资料

