

15 | 海纳百川：HTTP的实体数据

2019-07-01 Chrono

《透视HTTP协议》

课程介绍 >



讲述：Chrono

时长 11:53 大小 13.61M



你好，我是 Chrono。

今天我要与你分享的话题是“海纳百川：HTTP 的实体数据”。

这一讲是“进阶篇”的第一讲，从今天开始，我会用连续的 8 讲的篇幅来详细解析 HTTP 协议里的各种头字段，包括定义、功能、使用方式、注意事项等等。学完了这些课程，你就可以完全掌握 HTTP 协议。

在前面的“基础篇”里我们了解了 HTTP 报文的结构，知道一个 HTTP 报文是由“header+body”组成的。但那时我们主要研究的是 header，没有涉及到 body。所以，“进阶篇”的第一讲就从 HTTP 的 body 谈起。

数据类型与编码

领资料



在 TCP/IP 协议栈里，传输数据基本上都是“header+body”的格式。但 TCP、UDP 因为是传输层的协议，它们不会关心 body 数据是什么，只要把数据发送到对方就算是完成了任务。

而 HTTP 协议则不同，它是应用层的协议，数据到达之后工作只能说是完成了一半，还必须告诉上层应用这是什么数据才行，否则上层应用就会“不知所措”。

你可以设想一下，假如 HTTP 没有告知数据类型功能，服务器把“一大坨”数据发给了浏览器，浏览器看到的是一个“黑盒子”，这时候该怎么办呢？

当然，它可以“猜”。因为很多数据都是有固定格式的，所以通过检查数据的前几个字节也许就能知道这是个 GIF 图片、或者是个 MP3 音乐文件，但这种方式无疑十分低效，而且有很大几率会检查不出来文件类型。

幸运的是，早在 HTTP 协议诞生之前就已经有了针对这种问题的解决方案，不过它是用在电子邮件系统里的，让电子邮件可以发送 ASCII 码以外的任意数据，方案的名字叫做“**多用途互联网邮件扩展**”（Multipurpose Internet Mail Extensions），简称为 MIME。

MIME 是一个很大的标准规范，但 HTTP 只“顺手牵羊”取了其中的一部分，用来标记 body 的数据类型，这就是我们平常总能听到的“**MIME type**”。

MIME 把数据分成了八大类，每个大类下再细分出多个子类，形式是“type/subtype”的字符串，巧得很，刚好也符合了 HTTP 明文的特点，所以能够很容易地纳入 HTTP 头字段里。

这里简单列举一下在 HTTP 里经常遇到的几个类别：

1. text：即文本格式的可读数据，我们最熟悉的应该就是 text/html 了，表示超文本文档，此外还有纯文本 text/plain、样式表 text/css 等。
2. image：即图像文件，有 image/gif、image/jpeg、image/png 等。
3. audio/video：音频和视频数据，例如 audio/mpeg、video/mp4 等。
4. application：数据格式不固定，可能是文本也可能是二进制，必须由上层应用程序来解释。常见的有 application/json、application/javascript、application/pdf 等，另外，如果实在是不知道数据是什么类型，像刚才说的“黑盒”，就会是 application/octet-stream，即不透明的二进制数据。



但仅有 MIME type 还不够，因为 HTTP 在传输时为了节约带宽，有时候还会压缩数据，为了不要让浏览器继续“猜”，还需要有一个“Encoding type”，告诉数据是用的什么编码格式，这样对方才能正确解压缩，还原出原始的数据。

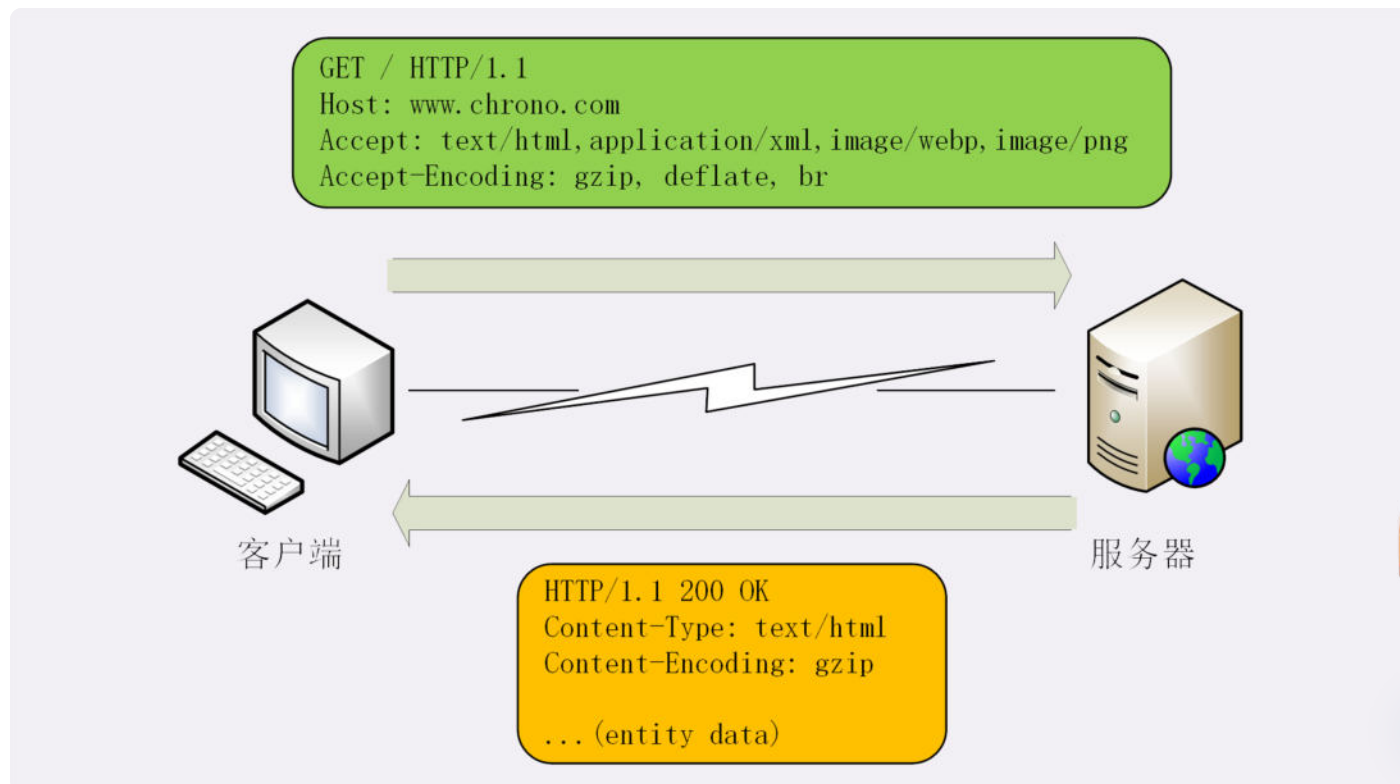
比起 MIME type 来说，Encoding type 就少了很多，常用的只有下面三种：

1. gzip: GNU zip 压缩格式，也是互联网上最流行的压缩格式；
2. deflate: zlib (deflate) 压缩格式，流行程度仅次于 gzip；
3. br: 一种专门为 HTTP 优化的新压缩算法 (Brotli) 。

数据类型使用的头字段

有了 MIME type 和 Encoding type，无论是浏览器还是服务器就都可以轻松识别出 body 的类型，也就能够正确处理数据了。

HTTP 协议为此定义了两个 Accept 请求头字段和两个 Content 实体头字段，用于客户端和服务端进行“内容协商”。也就是说，客户端用 Accept 头告诉服务器希望接收什么样的数据，而服务器用 Content 头告诉客户端实际发送了什么样的数据。



Accept 字段标记的是客户端可理解的 MIME type，可以用“,”做分隔符列出多个类型，让服务器有更多的选择余地，例如下面的这个头：

 复制代码

```
1 Accept: text/html,application/xml,image/webp,image/png
```

这就是告诉服务器：“我能够看懂 HTML、XML 的文本，还有 webp 和 png 的图片，请给我这四类格式的数据”。

相应的，服务器会在响应报文里用头字段 **Content-Type** 告诉实体数据的真实类型：

 复制代码

```
1 Content-Type: text/html
2 Content-Type: image/png
```

这样浏览器看到报文里的类型是“text/html”就知道是 HTML 文件，会调用排版引擎渲染出页面，看到“image/png”就知道是一个 PNG 文件，就会在页面上显示出图像。

Accept-Encoding 字段标记的是客户端支持的压缩格式，例如上面说的 gzip、deflate 等，同样也可以用“,”列出多个，服务器可以选择其中一种来压缩数据，实际使用的压缩格式放在响应头字段 **Content-Encoding** 里。

 复制代码

```
1 Accept-Encoding: gzip, deflate, br
2 Content-Encoding: gzip
```

不过这两个字段是可以省略的，如果请求报文里没有 Accept-Encoding 字段，就表示客户端不支持压缩数据；如果响应报文里没有 Content-Encoding 字段，就表示响应数据没有被压缩。

 领资料

语言类型与编码

MIME type 和 Encoding type 解决了计算机理解 body 数据的问题，但互联网遍布全球，不同国家不同地区的人使用了很多不同的语言，虽然都是 text/html，但如何让浏览器显示出每



个人都可理解可阅读的语言文字呢？

这实际上就是“国际化”的问题。HTTP 采用了与数据类型相似的解决方案，又引入了两个概念：语言类型与字符集。

所谓的“**语言类型**”就是人类使用的自然语言，例如英语、汉语、日语等，而这些自然语言可能还有下属的地区性方言，所以在需要明确区分的时候也要使用“type-subtype”的形式，不过这里的格式与数据类型不同，**分隔符不是“/”，而是“-”**。

举几个例子：en 表示任意的英语，en-US 表示美式英语，en-GB 表示英式英语，而 zh-CN 就表示我们最常使用的汉语。

关于自然语言的计算机处理还有一个更麻烦的东西叫做“字符集”。

在计算机发展的早期，各个国家和地区的人们“各自为政”，发明了许多字符编码方式来处理文字，比如英语世界用的 ASCII、汉语世界用的 GBK、BIG5，日语世界用的 Shift_JIS 等。同样的一段文字，用一种编码显示正常，换另一种编码后可能就会变得一团糟。

所以后来就出现了 Unicode 和 UTF-8，把世界上所有的语言都容纳在一种编码方案里，遵循 UTF-8 字符编码方式的 Unicode 字符集也成为了互联网上的标准字符集。

语言类型使用的头字段

同样的，HTTP 协议也使用 Accept 请求头字段和 Content 实体头字段，用于客户端和服务器的语言与编码进行“**内容协商**”。

Accept-Language 字段标记了客户端可理解的自然语言，也允许用“,”做分隔符列出多个类型，例如：

```
1 Accept-Language: zh-CN, zh, en
```

 复制代码

领资料

这个请求头会告诉服务器：“最好给我 zh-CN 的汉语文字，如果没有就用其他的汉语方言，如果还没有就给英文”。

相应的，服务器应该在响应报文里用头字段 **Content-Language** 告诉客户端实体数据使用的实际语言类型：

复制代码

```
1 Content-Language: zh-CN
```

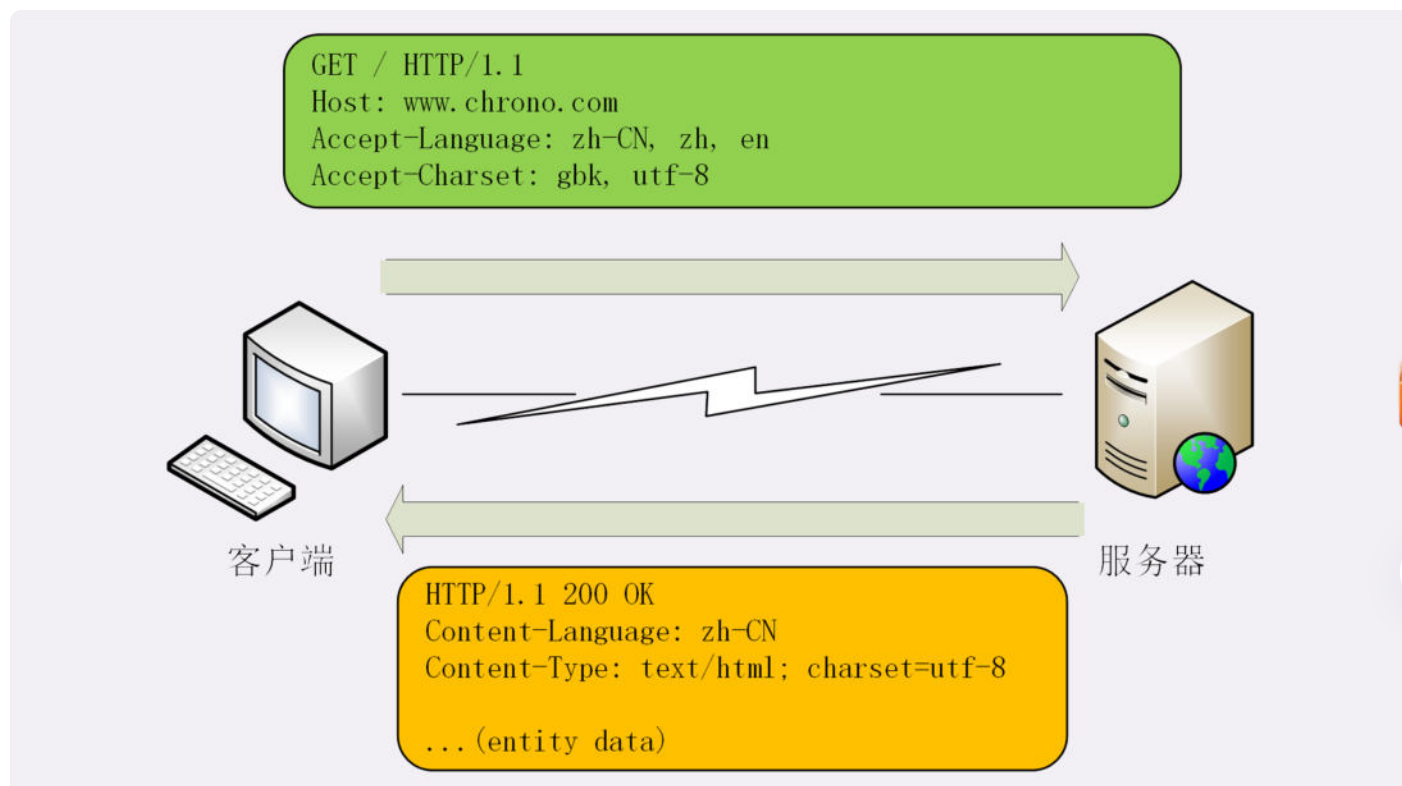
字符集在 HTTP 里使用的请求头字段是 **Accept-Charset**，但响应头里却没有对应的 **Content-Charset**，而是在 **Content-Type** 字段的数据类型后面用“charset=xxx”来表示，这点需要特别注意。

例如，浏览器请求 GBK 或 UTF-8 的字符集，然后服务器返回的是 UTF-8 编码，就是下面这样：

复制代码

```
1 Accept-Charset: gbk, utf-8
2 Content-Type: text/html; charset=utf-8
```

不过现在的浏览器都支持多种字符集，通常不会发送 **Accept-Charset**，而服务器也不会发送 **Content-Language**，因为使用的语言完全可以由字符集推断出来，所以在请求头里一般只有 **Accept-Language** 字段，响应头里只有 **Content-Type** 字段。



领资料



内容协商的质量值

在 HTTP 协议里用 Accept、Accept-Encoding、Accept-Language 等请求头字段进行内容协商的时候，还可以用一种特殊的“q”参数表示权重来设定优先级，这里的“q”是“quality factor”的意思。

权重的最大值是 1，最小值是 0.01，默认值是 1，如果值是 0 就表示拒绝。具体的形式是在数据类型或语言代码后面加一个“;”，然后是“q=value”。

这里要提醒的是“;”的用法，在大多数编程语言里“;”的断句语气要强于“，”，而在 HTTP 的内容协商里却恰好反了过来，“;”的意义是小于“，”的。

例如下面的 Accept 字段：

```
1 Accept: text/html,application/xml;q=0.9,*/*;q=0.8
```

 复制代码

它表示浏览器最希望使用的是 HTML 文件，权重是 1，其次是 XML 文件，权重是 0.9，最后是任意数据类型，权重是 0.8。服务器收到请求头后，就会计算权重，再根据自己的实际情况优先输出 HTML 或者 XML。

内容协商的结果

内容协商的过程是不透明的，每个 Web 服务器使用的算法都不一样。但有的时候，服务器会在响应头里多加一个 **Vary** 字段，记录服务器在内容协商时参考的请求头字段，给出一点信息，例如：

```
1 Vary: Accept-Encoding,User-Agent,Accept
```

 复制代码

这个 Vary 字段表示服务器依据了 Accept-Encoding、User-Agent 和 Accept 这三个头字段，然后决定了发回的响应报文。

领资料



Vary 字段可以认为是响应报文的一个特殊的“版本标记”。每当 Accept 等请求头变化时，Vary 也会随着响应报文一起变化。也就是说，同一个 URI 可能会有多个不同的“版本”，主要用在传输链路中间的代理服务器实现缓存服务，这个之后讲“HTTP 缓存”时还会再提到。

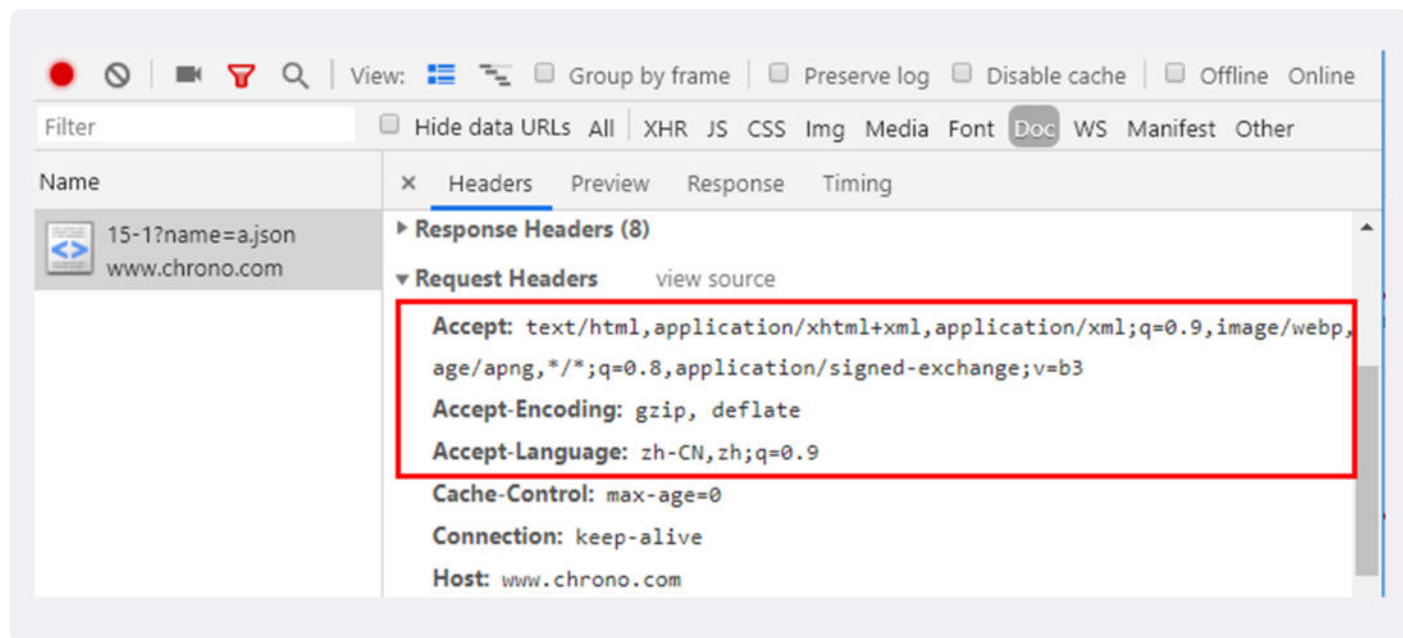
动手实验

上面讲完了理论部分，接下来就是实际动手操作了。可以用我们的实验环境，在 www 目录下有一个 mime 目录，里面预先存放了几个文件，可以用 URI“/15-1?name=file”的形式访问，例如：

复制代码

```
1 http://www.chrono.com/15-1?name=a.json
2 http://www.chrono.com/15-1?name=a.xml
```

在 Chrome 里打开开发者工具，就能够看到 Accept 和 Content 头：



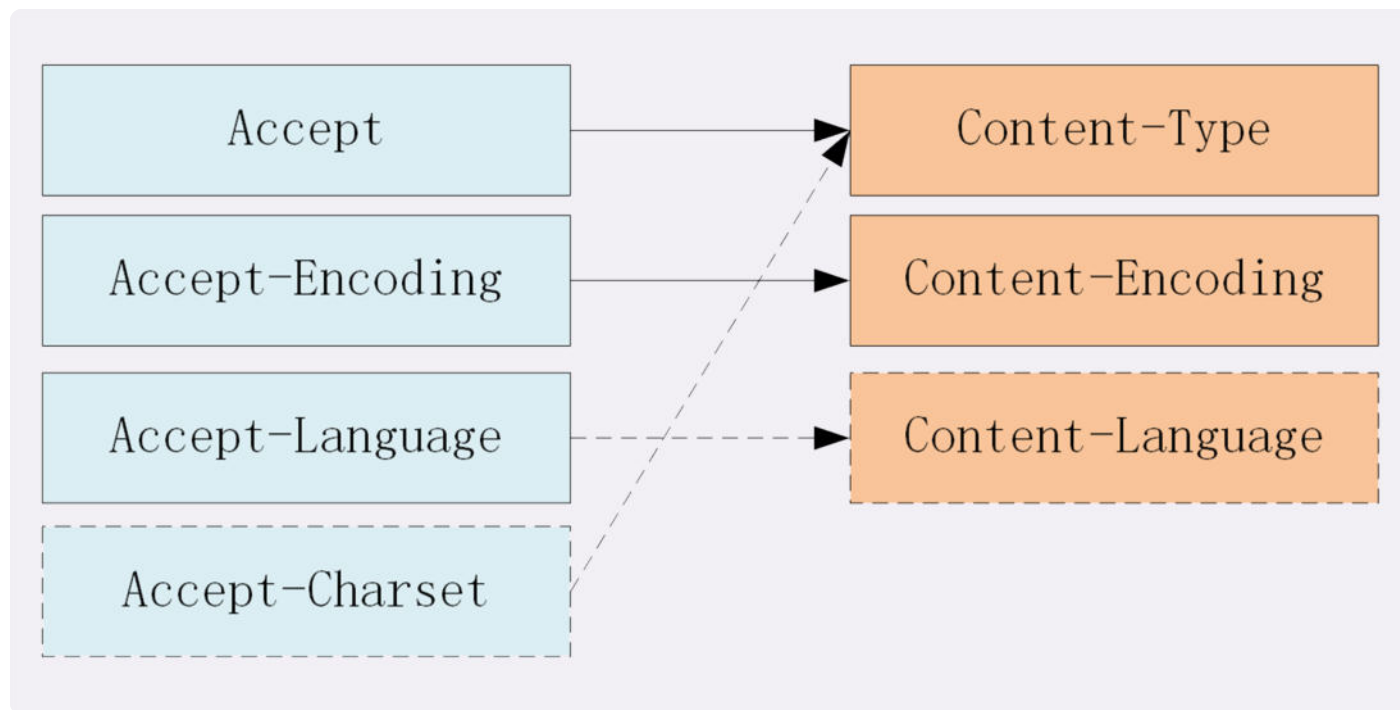
你也可以把任意的文件拷贝到 mime 目录下，比如压缩包、MP3、图片、视频等，再用 Chrome 访问，观察更多的 MIME type。

领资料

有了这些经验后，你还可以离开实验环境，直接访问各大门户网站，看看真实网络世界里的 HTTP 报文是什么样子的。

小结

今天我们学习了 HTTP 里的数据类型和语言类型，在这里为今天的内容做个小结。



1. 数据类型表示实体数据的内容是什么，使用的是 MIME type，相关的头字段是 Accept 和 Content-Type；
2. 数据编码表示实体数据的压缩方式，相关的头字段是 Accept-Encoding 和 Content-Encoding；
3. 语言类型表示实体数据的自然语言，相关的头字段是 Accept-Language 和 Content-Language；
4. 字符集表示实体数据的编码方式，相关的头字段是 Accept-Charset 和 Content-Type；
5. 客户端需要在请求头里使用 Accept 等头字段与服务器进行“内容协商”，要求服务器返回最合适的数据；
6. Accept 等头字段可以用“,”顺序列出多个可能的选项，还可以用“;q=”参数来精确指定权重。

课下作业

1. 试着解释一下这个请求头“Accept-Encoding: gzip, deflate;q=1.0, *;q=0.5, br;q=0”，再模拟一下服务器的响应头。
2. 假设你要使用 POST 方法向服务器提交一些 JSON 格式的数据，里面包含有中文，请求头应该是什么样子的呢？

领资料




3. 试着用快递发货收货比喻一下 MIME、Encoding 等概念。

欢迎你把自己的答案写在留言区，与我和其他同学一起讨论。如果你觉得有所收获，欢迎你把文章分享给你的朋友。

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 24  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 14 | HTTP有哪些优点？又有哪些缺点？

下一篇 16 | 把大象装进冰箱：HTTP传输大文件的方法

学习推荐

JVM + NIO + Spring

各大厂面试题及知识点详解

限时免费 



领资料





或豪

2019-07-01

上周五和服务端做上传图片的时候遇到过这个content-type的问题，上传图片时候我这边需要设置content-type:"image/jpg"，然后传完了，我在预览的时候获取图片的地址，此时比如通过a标签的方式打开新标签预览该图片时才能成功预览，不然如果使用上传的js-sdk设置的默认类型：content-type:"octet-stream"，那么浏览器就不认识这个图片了，转而会下载这个文件(图片)，所以我是不是可以理解为content-type这字段在请求头，和响应头里都能使用？或者上传文件这个业务又不同于一般的请求操作呢？

作者回复: 是的，看来是我没说清楚，导致有的同学误会了。

content-type是实体字段，所以请求和响应里都可以用，作用是指明body数据的类型。

正文里为了叙述方便，只在服务器的响应报文里出现了content-type，实际上它是个通用字段，如果要发post请求，就需要带上它。



77



走马

2019-07-04

accept 表达的是你想要的

而你发送 post请求时，你发送的数据是给服务器的，这时候就需要像 服务器会用 content-type 标明它给你的数据类型一样，你也需要用 content- 来表明你给别人的数据的一些属性

作者回复: √

共 4 条评论 >

34



亚洲舞王.尼古拉斯赵...

2019-07-01

1.含义是：我这个请求最希望服务器给我返回的编码方式是gzip和deflate，他们俩在我这是最优的地位，我不接受br的编码方式，如果还有其他的编码方式的话对我来说权重0.5。服务器可能的响应头是

HTTP/1.1 200 OK

Content-Encoding: gzip

2.请求头可能是

POST /serv/v1/user/auth HTTP/1.1

Content-Type: application/json

Accept-Language: zh-CN, zh

Accept-Charset: gbk, utf-8

领资料



3.MIME类比快递的话就是你要快递的物品（衣服，食物等），Encoding就是快递你这个物品的包装方式，如果是衣服可能就随意一点一个袋子，如果是食物担心腐烂给你放个冰袋进去不知道回答的对不对，请老师指正

作者回复: 回答的不错。

第二个问题要修改一下，这也怪我没有说清楚。

content-*字段也可以用在请求报文里，说明请求体的数据类型。在这里不能用accept字段，因为是post，所以要用content-language来指明body的语言类型，在content-type里用charset指明编码类型。

可以参考其他同学的回答。

共 5 条评论 >

👍 27



苦行僧

2019-07-02

现在很多小文件 比如图片 都往云存上放了 千万指定正确content-type 一旦指定错 批量修改太麻烦 而且会影响终端的解析

作者回复: 经验之谈!

共 2 条评论 >

👍 22



隰有荷

2019-10-02

用post方法请求接口时，在客户端语言的设置上不能使用Accept-Language吗？为什么一定是Content-Language呢？是不是Accept-Language只用于get方式时，表明客户端需要的语言呢？

作者回复: post和get时都可以使用Accept-Language，表示客户端可以理解的语言，要求服务器按照指示返回数据。

Content-Language表示的是body数据的语言，因为post带有body，所以要用Content-Language来告诉服务器，报文的body是什么。

如果get报文也有body，那么它也可以使用Content-Language。

Accept-Language是请求头字段，只要是发请求就可以带。

领资料



Content-Language是实体头字段，只要有body就可以带。



20



MJ

2019-07-05

老师，每一个并发连接，都需要新建tcp三次握手吗？还是一次tcp握手后，可以并发多个连接？

作者回复: 每建立一个连接就需要tcp握手，对同一个ip地址+端口，浏览器通常最多建立6个并发连接。



19



BellenHsin

2019-07-01

这篇写的不错

作者回复: thanks。

共 2 条评论 >

11



1900

2019-07-01

“所以后来就出现了 Unicode 和 UTF-8，把世界上所有的语言都容纳在一种编码方案里，UTF-8 也成为了互联网上的标准字符集。”

这句话最后有点问题吧？Unicode才是字符集，应该是“遵循UTF-8字符编码方式的Unicode字符集也成为了互联网上的标准字符集”，是吗？

作者回复: 嗯，我说的时候不太准确。utf-8只是编码方案，Unicode是字符集。



10

领资料



-W.LI-

2019-07-01

老师好!有个问题,之前遇到过一个发送ajax请求。前端忘记在content-type里面指定, application/json。后端接受数据失败。具体表现不太记得了好像都是null。后来前端加了content-type就好了。accept比较好理解就是发起请求放想要接受的内容。content-type是服务器,是响应类型的话。客户端在发送请求时压根就不知道啊,也不应该由客户端来设置。



所以我想问的是，accept相关的都是请求头里面的数据

content-type相关的都是响应头里的数据么？

至于我前面正确的写法应该是在accept里面设置json类型。错写了content-type。框架做了兼容处理(在服务端看起来content-type起作用了)？

谢谢老师

作者回复: 客户端在发送请求的时候也有义务设置content-type，也应该是知道数据是什么类型的，你设置成json，服务器看到了就好处理。

content-type是实体字段，请求响应里都可以出现。

accept是告诉服务器，客户端支持什么类型，防止服务器发过来的数据不认识。



👍 5



do it

2020-05-20

1、试着解释一下这个请求头“Accept-Encoding: gzip, deflate;q=1.0, *,q=0.5, br;q=0”，再模拟一下服务器的响应头。

：最好使用gzip,deflate压缩格式，我不接受br压缩，如果都没有的话就选择其他格式

2、假设你要使用 POST 方法向服务器提交一些 JSON 格式的数据，里面包含有中文，请求头应该是什么样子的呢？

content-type: application/json, charset=gbk

content-language:zh-cn, zh

3、试着用快递发货收货比喻一下 MIME、Encoding 等概念。

物品的种类（水果、衣服）就是MIME，包装方式就是Encoding

作者回复: 回答的很好，继续努力。



👍 4



-W.LI-

2019-07-01

老师好!那accept是不是有两个语意

1.客户端希望接受(支持)的数据类型

2.我发送的数据就是这个类型的。请用这些方式解析？

问题:accept指定text。实际传的数据是一个json这样的后台会用text解析。然后拿不到数据是么?在请求头里加content-type这些字段会起作用么？

领资料



作者回复: 1.accept是你说的第一个意思, 没有第二个意思。

2.第二个意思应该用Content-Type

3.看后台逻辑如何处理, 数据是肯定可以拿到的, 而且json也属于text。

4.在请求头里可以加content-type字段, 表示请求体的数据类型。



👍 4



Jinlee

2020-05-07

1. 含义: 服务器, 你返回的数据编码方式最好是gzip或者deflate, 实在不行返回其他的编码方式也行, 但是, 我不接受br类型的数据编码方式

模拟服务器响应头:

HTTP/1.1 200 OK

Content-Encoding: gzip

2. 请求头:

POST / HTTP/1.1

Content-type:text/json; charset=GBK, utf-8

3. 收发快递比喻:

MIME type就是要发收的具体物品, 如文件、生鲜、衣物等。Encoding就是快递的包装方式, 如果是文件呢, 那我就用专用文件袋给你寄过去; 如果是生鲜, 那我就给你套个保鲜泡沫寄过去; 如果是衣物, 那我就给你个一般的快递包装寄过去。

参考其他优秀同学的答案😁

作者回复: 回答的很好。

第二个问题, 少了个content-language, 而且charset不能同时使用两种编码。

共 2 条评论 >

👍 3

领资料



钱

2020-03-29

做web开发有两个问题, 估计许多人都遇到过, 一个是乱码问题, 这个是字符集设置不一致导

致的具体哪里设置的不一致通常需要观察所有需要数据转换的地方，一般是客户端和服务端不一致了。另外，就是文件上传，这个格式一定要设置对，否则就会感到莫名其妙。

我想请教老师两个问题：

1：看到说浏览器最多会有六个连接并发执行，为什么是六个，不会是因为六六大顺吧？

2：文件上传的速度和文件的大小密切相关，文件上传的大小限制都是有哪些限制或控制？之前，这是老早了，发现有些框架默认只能是2g

作者回复：

1.这个是rfc的规定，其实以前还要小。

2.http协议对大小没有限制，这些应该都是网站服务器做的限制，比如Nginx就可以设置client body的大小。



👍 3



啦啦啦

2019-07-02

不错不错，靠谱这篇，天天看这些参数一直不知道具体意思，今天老师讲了以后理解了

作者回复：继续努力。



👍 3



蓝配鸡

2019-10-29

试着解释一下这个请求头“Accept-Encoding: gzip, deflate;q=1.0, *;q=0.5, br;q=0”，再模拟一下服务器的响应头。

我希望用GZIP或者deflate压缩算法，实在不行给我其他的算法也行，但千万别给我br压缩过的数据，我这边可不会这个算法！

假设你要使用 POST 方法向服务器提交一些 JSON 格式的数据，里面包含有中文，请求头应该是什么样子的呢？

content-type: zh-CN, zh, charset=utf-8, application/json

试着用快递发货收货比喻一下 MIME、Encoding 等概念。

比如说我想在某宝上买乐高，

MIME 确定了我购买的乐高种类，是成品呢，还是零散的需要自己拼装。

Encoding好比这个乐高的包装方式，某宝可以选择把所有的乐高零件全都放在一个包装里，也可以分模块把零件放在不同的包装里。我也可以主动告诉某宝我想要哪一种包装方式。

作者回复：good，很勤奋啊。

领资料





2



aNught

2019-07-09

老师，您好，如果我accept-encoding填写了gzip，那服务端发来的报文是是gzip压缩过的吗，我需要解压才行是吗？

作者回复: accept-encoding:gzip表示客户端支持gzip，但服务器发过来的是否经过压缩需要看content-encoding，也有可能不压缩。



2



djfhchdh

2019-07-01

1.服务器优先按照gzip和deflate压缩，否则用其他压缩算法，但是不用brotli算法

作者回复: √



2



兔嘟嘟

2021-07-20

罗老师，我不是很能理解accept-language这类字段的意义，因为我们开发前后端肯定是有有一套接口文档的，里面约定了各种开发细节，前后端应该使用什么编码什么语言，为何还要在报文里去提accept-language？

总不能前端写好了，发送一个报文给后端，后端程序员慢慢看里面的字段，再开始后端的编程吧？

作者回复: http协议当初设计的时候可没有考虑过前后端开发的问题，它的目标是给互联网上的各种人群去阅读，所以就需要有语言编码。

在你说的开发领域，前后端都约定好了，自然就不需要用accept-language来交换信息了，但对于普通的浏览网页来说，还是要用这个字段的。

领资料

共 2 条评论 >

1



Celine

2020-03-27

1，接收的编码格式最好是GUN zip 和deflate ,不接受br,如果还有其他格式。我勉为其难也接受一下吧，权重是0.5；http响应头应该是 Http 1.1 200 Ok Content-Encoding: gzip



2,Content-Type:Application/json

Content-Language: cn-zh

Content-charset: gbk, utf-8

作者回复: 有小错误, 注意没有Content-charset这样的字段。

共 2 条评论 >



1



芒果

2020-01-19

给老师点赞, 大牛就是大牛

作者回复: 诚惶诚恐。



1

领资料

