

# 目标

---

- 能够说出什么是 HTTP 协议
- 能够知道 HTTP 请求消息的组成部分
- 能够知道 HTTP 响应消息的组成部分
- 能够说出常见的请求方法
- 能够说出常见的响应状态码

## HTTP协议简介

---

### 什么是通信

---

通信，就是**信息的传递和交换**。

通信三要素：

- 通信的主体
- 通信的内容
- 通信的方式

### 现实生活中的通信

案例：张三要把自己考上传智专修学院的好消息写信告诉自己的好朋友李四

其中：

通信的**主体**是张三和李四；

通信的**内容**是考上传智专修学院；

通信的**方式**是写信；

### 互联网中的通信

案例：服务器把传智专修学院的简介通过响应的方式发送给客户端浏览器。

其中，

通信的**主体**是服务器和客户端浏览器；

通信的**内容**是传智专修学院的简介；

通信的**方式**是响应；

### 什么是通信协议

---

**通信协议**（Communication Protocol）是指通信的双方完成通信所**必须遵守的规则和约定**。

**通俗的理解**：通信双方采用约定好的格式来发送和接收消息，这种**事先约定好的通信格式**，就叫做**通信协议**。

### 现实生活中的通信协议

张三与李四采用写信的方式进行通信，在填写信封时，写信的双方需要遵守固定的规则。**信封的填写规则**就是一种通信协议

150056

← 这里写收信人邮编

贴 邮  
票 处

黑龙江省哈尔滨市道外区

← 收信人地址, 地址要详细, 所以可能要占两行或者三行

王某

← 收信人姓名, 字体稍大一些

黑龙江省哈尔滨市道外区

← 寄信人地址、姓名。

邮政编码

100053

←

## 互联网中的通信协议

客户端与服务器之间要实现网页内容的传输，则通信的双方必须遵守网页内容的传输协议。

网页内容又叫做**超文本**，因此网页内容的传输协议又叫做**超文本传输协议**（HyperText Transfer Protocol），

简称 **HTTP 协议**。

## HTTP(☆☆☆)

### 什么是HTTP协议

**HTTP 协议**即超文本传送协议（HyperText Transfer Protocol），它规定了客户端与服务器之间进行网页内容传输时，所必须遵守的传输格式。

例如：

- 客户端要以HTTP协议要求的格式把数据提交到服务器
- 服务器要以HTTP协议要求的格式把内容响应给客户端

### HTTP协议的交互模型

HTTP 协议采用了 **请求/响应** 的交互模型。



# HTTP请求消息

## 什么是HTTP请求消息

由于 HTTP 协议属于客户端浏览器和服务端之间的通信协议。因此，客户端发起的请求叫做 HTTP 请求，客户端

端发送到服务器的消息，叫做 HTTP 请求消息。

**注意：** HTTP 请求消息又叫做 HTTP 请求报文

## HTTP请求消息的组成部分(☆☆☆)

HTTP 请求消息由请求行 (request line)、请求头部 (header)、空行 和 请求体 4 个部分组成。



## 请求行

请求行由请求方式、URL 和 HTTP 协议版本 3 个部分组成，他们之间使用空格隔开。

请求方式	空格	URL	空格	协议版本	回车符	换行符
------	----	-----	----	------	-----	-----



### 请求头部

**请求头部**用来描述客户端的基本信息，从而把客户端相关的信息告知服务器。比如：User-Agent 用来说明当前是什么类型的浏览器；

Content-Type 用来描述发送到服务器的数据格式；Accept 用来描述客户端能够接收什么类型的返回内容；Accept-Language 用来描述客户端期望接收哪种人类语言的文本内容。

请求头部由多行 **键/值对** 组成，每行的键和值之间用英文的冒号分隔

头部字段名称	:(冒号)	值	回车符	换行符
.....				
头部字段名称	:(冒号)	值	回车符	换行符

### 请求头部 – 常见的请求头字段

头部字段	说明
Host	要请求的服务器域名
Connection	客户端与服务器的连接方式(close 或 keepalive)
Content-Length	用来描述请求体的大小
Accept	客户端可识别的响应内容类型列表
User-Agent	产生请求的浏览器类型
Content-Type	客户端告诉服务器实际发送的数据类型
Accept-Encoding	客户端可接收的内容压缩编码形式
Accept-Language	用户期望获得的自然语言的优先顺序

▼ Request Headers view parsed

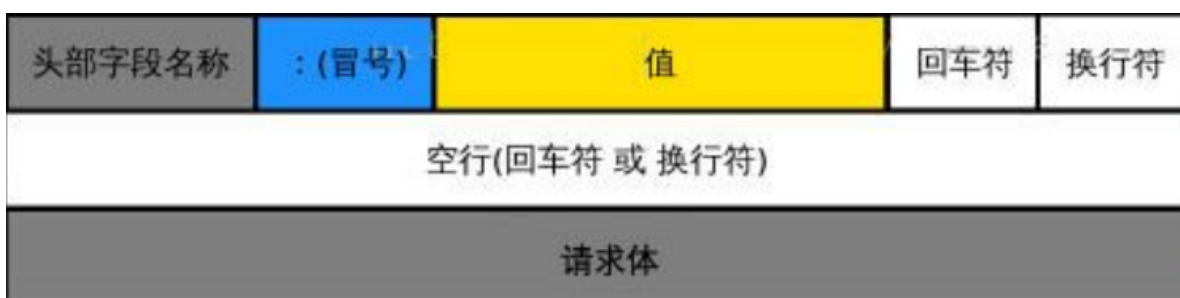
```
POST /api/post HTTP/1.1
Host: ajax.frontend.itheima.net:3006
Connection: keep-alive
Content-Length: 14
Accept: */*
Origin: null
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.108 Safari/537.36
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8
```

关于更多请求头字段的描述，可以查看 [MDN 官方文档](https://developer.mozilla.org/zh-CN/docs/Web/HTTP/Headers)：<https://developer.mozilla.org/zh-CN/docs/Web/HTTP/Headers>

## 空行

最后一个请求头字段的后面是一个**空行**，通知服务器**请求头部至此结束**。

请求消息中的**空行**，用来分隔请求头部与请求体



## 请求体

请求体中存放的，是要通过 `POST` 方式提交到服务器的数据。



**注意：**只有 `POST` 请求才有**请求体**，`GET` 请求**没有请求体**！

# HTTP响应消息

## 什么是HTTP响应消息

**响应消息**就是服务器响应给客户端的消息内容，也叫作**响应报文**。

## HTTP响应消息的组成部分(☆☆☆)

HTTP响应消息由**状态行**、**响应头部**、**空行** 和 **响应体** 4 个部分组成，如下图所示：



## 状态行

状态行由 HTTP 协议版本、状态码和状态码的描述文本 3 个部分组成，他们之间使用空格隔开

协议版本	空格	状态码	空格	状态码描述	回车符	换行符
------	----	-----	----	-------	-----	-----

```
▼ Response Headers view parsed
HTTP/1.1 200 OK
X-Powered-By: Express
Access-Control-Allow-Origin: *
Content-Type: application/json; charset=utf-8
Content-Length: 68
ETag: W/"44-nT/y6yOFj7H40EVW1DWB1MG+Pq0"
Date: Wed, 27 Nov 2019 01:48:57 GMT
Connection: keep-alive
```

## 响应头部

响应头部用来描述服务器的基本信息。响应头部由多行 键/值对 组成，每行的键和值之间用英文的冒号分隔。

头部字段名称	: (冒号)	值	回车符	换行符
.....				
头部字段名称	: (冒号)	值	回车符	换行符

## 响应头部 - 常见的响应头字段



关于更多响应头字段的描述，可以查看 MDN 官方文档：<https://developer.mozilla.org/zh-CN/docs/Web/HTTP/Headers>

## 空行

在最后一个响应头部字段结束之后，会紧跟一个**空行**，用来通知客户端**响应头部至此结束**。

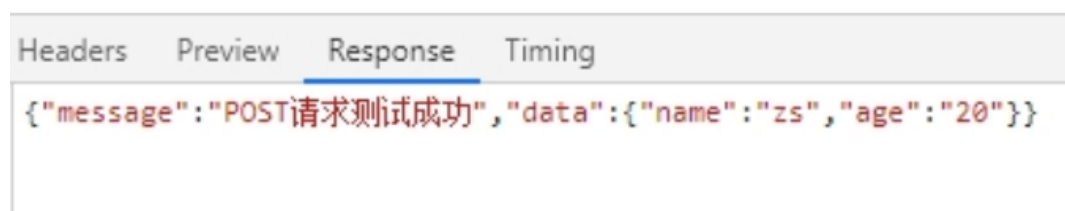
响应消息中的空行，用来分隔**响应头部**与**响应体**。

头部字段名称	:(冒号)	值	回车符	换行符
空行(回车符 或 换行符)				
响应体				

## 响应体

响应体中存放的，是服务器响应给客户端的资源内容。

头部字段名称	:(冒号)	值	回车符	换行符
空行(回车符 或 换行符)				
响应体				



## HTTP请求方法



# 什么是HTTP请求方法

HTTP 请求方法，属于 HTTP 协议中的一部分，请求方法的作用是：用来表明**要对服务器上的资源执行的操作**。最常用的请求方法是 GET 和 POST。

## HTTP的请求方法(☆☆☆)

序号	方法	描述
1	GET	(查询)发送请求来获得服务器上的资源，请求体中不会包含请求数据，请求数据放在协议头中。
2	POST	(新增)向服务器提交资源（例如提交表单或上传文件）。数据被包含在请求体中提交给服务器。
3	PUT	(修改)向服务器提交资源，并使用提交的新资源，替换掉服务器对应的旧资源。
4	DELETE	(删除)请求服务器删除指定的资源。
5	HEAD	HEAD 方法请求一个与 GET 请求的响应相同的响应，但没有响应体。
6	OPTIONS	获取http服务器支持的http请求方法，允许客户端查看服务器的性能，比如ajax跨域时的预检等。
7	CONNECT	建立一个到由目标资源标识的服务器的隧道。
8	TRACE	沿着到目标资源的路径执行一个消息环回测试，主要用于测试或诊断。
9	PATCH	是对 PUT 方法的补充，用来对已知资源进行局部更新。

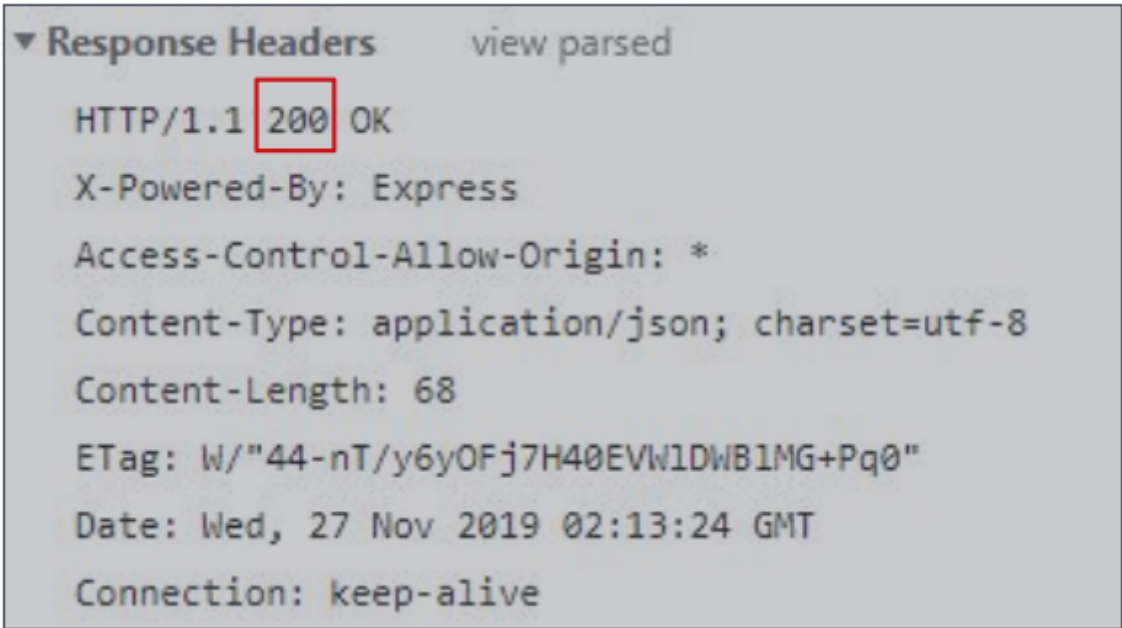
# HTTP响应状态码

## 什么是HTTP响应状态码

**HTTP 响应状态码**（HTTP Status Code），也属于 HTTP 协议的一部分，用来标识响应的状态。

响应状态码会随着响应消息一起被发送至客户端浏览器，浏览器根据服务器返回的响应状态码，就能知道这次

HTTP 请求的结果是成功还是失败了。



## HTTP响应状态码的组成及分类



HTTP 状态码由三个十进制数字组成，第一个十进制数字定义了状态码的类型，后两个数字用来对状态码进行细分。

HTTP 状态码共分为 5 种类型：

分类	分类描述
1**	信息，服务器收到请求，需要请求者继续执行操作（实际开发中很少遇到 1** 类型的状态码）
2**	成功，操作被成功接收并处理
3**	重定向，需要进一步的操作以完成请求
4**	客户端错误，请求包含语法错误或无法完成请求
5**	服务器错误，服务器在处理请求的过程中发生了错误

完整的 HTTP 响应状态码，可以参考 MDN 官方文档 <https://developer.mozilla.org/zh-CN/docs/Web/HTTP/Status>

## 常见的HTTP响应状态码(☆☆☆)

### 2\*\* 成功相关的响应状态码

2\*\* 范围的状态码，表示服务器已成功接收到请求并进行处理。常见的 2\*\* 类型的状态码如下：

状态码	状态码英文名称	中文描述
200	OK	请求成功。一般用于 GET 与 POST 请求
201	Created	已创建。成功请求并创建了新的资源，通常用于 POST 或 PUT 请求

### 3\*\* 重定向相关的响应状态码

3\*\* 范围的状态码，表示表示服务器要求客户端重定向，需要客户端进一步的操作以完成资源的请求。常见的 3\*\* 类型的状态码如下：

状态码	状态码英文名称	中文描述
301	Moved Permanently	永久移动。请求的资源已被永久的移动到新URI，返回信息会包括新的URI，浏览器会自动定向到新URI。今后任何新的请求都应使用新的URI代替
302	Found	临时移动。与301类似。但资源只是临时被移动。客户端应继续使用原有URI
304	Not Modified	未修改。所请求的资源未修改，服务器返回此状态码时，不会返回任何资源（响应消息中不包含响应体）。客户端通常会缓存访问过的资源。

### 4\*\* 客户端错误相关的响应状态码

4\*\* 范围的状态码，表示客户端的请求有非法内容，从而导致这次请求失败。常见的 4\*\* 类型的状态码如下：

状态码	状态码英文名称	中文描述
400	Bad Request	1、语义有误，当前请求无法被服务器理解。除非进行修改，否则客户端不应该重复提交这个请求。 2、请求参数有误。
401	Unauthorized	当前请求需要用户验证。
403	Forbidden	服务器已经理解请求，但是拒绝执行它。
404	Not Found	服务器无法根据客户端的请求找到资源（网页）。
408	Request Timeout	请求超时。服务器等待客户端发送的请求时间过长，超时。

## 5\*\* 服务端错误相关的响应状态码

5\*\* 范围的状态码，表示服务器未能正常处理客户端的请求而出现意外错误。常见的 5\*\* 类型的状态码如下

状态码	状态码英文名称	中文描述
500	Internal Server Error	服务器内部错误，无法完成请求。
501	Not Implemented	服务器不支持该请求方法，无法完成请求。只有 GET 和 HEAD 请求方法是要求每个服务器必须支持的，其它请求方法在不支持的服务器上会返回501
503	Service Unavailable	由于超载或系统维护，服务器暂时的无法处理客户端的请求。