# HTTP协议(极其重要)

### HTTP协议介绍:

HTTP协议:超文本传输协议 基于TCP/IP的应用层通信协议,这个协议详细规定了浏览 器和服务器之间互相通信的规则

\* 客户端向服务器发送数据:请求报文

\* 服务器向客户端返回数据:响应报文

$ HTTP协议规定了请求报文和响应报文之间的通信格式和内容

### **网页的加载过程:**

下列HTML文件(网页)加载过程:



(客户端和服务器先建立连接(三次握手))

1客户端浏览器向服务器发送请求,申请HTML

服务器接收请求,发送响应(响应体包含HTML页面代码);

2 客户端接收到HTML后解析到外链式index.css,再次向服务器发送请求,服务器接收请求发送响应传输index.css;

3 客户端浏览器解析HTML后解析到外链式index.js,再次向服务器发送请求,服务器接收请求后发送响应传输index.js

4 客户端接收到index.js后解析到img 再次向服务器发送请求,服务器接收请求发送响应传输img

总结:浏览器向服务器申请HTML,HTML中遇到外链式css js img等文件会向网站服务器发送请求



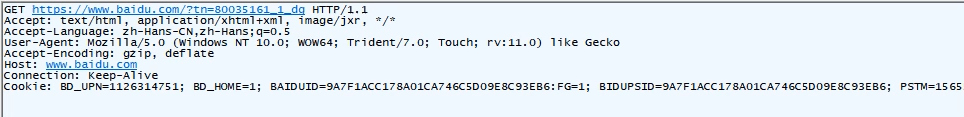
### Fiddle工具:

Fiddler是一个http协议调试代理工具,可抓取网页所有请求和响应,俗称抓包

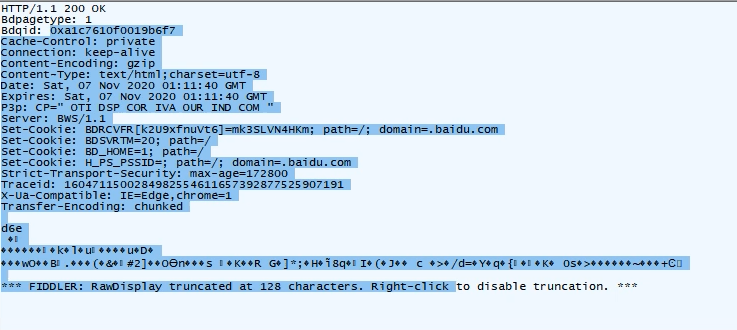
Fiddler实际上是代理(中间商),浏览器和服务器之间的所有报文都要经过代理

老师只使用此软件演示请求报文和响应报文(学习不用)

请求报文:



响应报文:



可以看出,响应报文和请求报文就是浏览器和服务器之间互相传递的字符串,

只不过报文内容(字符串)符合HTTP协议规定的格式罢了.

### 请求报文:

浏览器向服务器发送的请求报文如下:

GET <http://localhost:3000/index.html> HTTP/1.1

Host: localhost:3000

Connection: keep-alive

Pragma: no-cache

Cache-Control: no-cache

Upgrade-Insecure-Requests: 1

User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3282.140 Safari/537.36

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,\*/\*;q=0.8

Accept-Encoding: gzip, deflate, br

Accept-Language: zh-CN,zh;q=0.9

请求报文分成四大部分:

请求行 : GET/POST URL HTTP版本

请求头 ：{请求头名字:请求头的值}的集合

请求空行 ：真的空行

请求体 :报文的传输内容

请求行: GET  <http://localhost:3000/index.html> HTTP/1.1

请求头: Host: localhost:3000

Connection: keep-alive

Pragma: no-cache

Cache-Control: no-cache

Upgrade-Insecure-Requests: 1

User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64)

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,\*/\*;q=0.8

Accept-Encoding: gzip, deflate, br

Accept-Language: zh-CN,zh;q=0.9

?请求空行?

请求体:此处请求体为空

##### 请求行:

请求行: GET  <http://localhost:3000/index.html> HTTP/1.1

请求行分为三部分:

\* 请求方式: GET/POST (PUT DELETE等)；

\* 请求的URL: <http://localhost:3000/index.html>

\* HTTP协议版本: HTTP/1.1(一般固定为HTTP1.1)

URL介绍：

[https://www.baidu.com/ABC?wd=12&rsv\_spt=1#](http://localhost:3000/index.html)id

组成:

\* 传输协议:http/https/mongodb/ssh/FTP

\* 域名/ip地址: [www.baidu.com](http://www.baidu.com) / 128.10.23.2

\* 路径: /ABC

\* 查询字符串(query string): wd=12&rsv\_spt=1

\* 锚点: #id

**查询字符串: 用&连接并且每个&连接的都是一一映射的关系**

**(wd=12 rsv\_spt=1)**

##### 请求头:(请求头的键值对太多,介绍常见的几个)

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,\*/\*;q=0.8

Accept:表明客户端所能接受的数据类型

Accept-Language: zh-CN,zh;q=0.9

Accept-Language: 表明客户端所接受的语言，q为各种语言各自的权重

User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64)

User-Agent:客户端的字符串标志,服务器根据User-Agent分辨客户端类型(移动 PC/IE/IOS等)

Accept-Encoding: gzip, deflate, br

Accept-Encoding:客户端浏览器支持的压缩方式

Host: www.baidu.com

Host:客户端的主机名

Connection: keep-alive

Connetciton:连接配置 此次请求响应后,是否保留通信通道 Keep-alive close

Cookie:BD\_UPN=1126314751

Cookie: 用于会话控制,用于用户识别

Cookie:每次客户端发送请求,都会携带cookie,而服务器会获取cookie用于进行会话控制

Upgrade-Insecure-Requests: 1

Upgrade-Insecure-Requests: 告诉服务器,发请求的时候不用http,而用https

Upgrade-Insecure-Request通常与响应头中的Strict-Transport-Security使用

用以强制使用HTTPS协议(加密，有安全性);

##### 请求体:

GET请求的请求体是空的，但是POST请求的请求体一般不为空

[Login\_email=779498590@qq.com&login\_password=GREM9pus.fek-soos](mailto:Login_email=779498590@qq.com&login_password=GREM9pus.fek-soos)

以上就是表单验证当中请求体，也是一一映射的集合

login\_email和login\_password就是input标签name属性的值

[而779498590@qq.com和GREM9pus.fek-soos就是input标签value属性的值](mailto:而779498590@qq.com和GREM9pus.fek-soos就是input标签value属性的值)

所以input标签一定要加name属性，否则不会向浏览器提交input的内容(value);

此处就格式而言:请求体的格式和URL中查询字符串(查询参数)的格式一致

都是&映射的集合;、

但是实际上请求体的格式是非常灵活的,不限于为url的查询字符串形式,

{任意格式都可以,只要服务端有程序能正常识别即可}JSON与URL查询字符串用得较多

##### 表单验证:

form表单中的提交按钮就是表单提交按钮:

点击submit,表单内容会提交,页面会跳转,

实际就是将表单中用户输入的内容拼接到请求报文(请求体)当中.

然后浏览器将请求报文发送到服务器

### 响应报文:

响应报文如下:

HTTP/1.1 200 OK

X-Powered-By: Express

Accept-Ranges: bytes

Cache-Control: public, max-age=0

Last-Modified: Wed, 21 Mar 2018 13:13:13 GMT

ETag: W/"a9-16248b12b64"

Content-Type: text/html; charset=UTF-8

Content-Length: 169

Date: Thu, 22 Mar 2018 12:58:41 GMT

Connection: keep-alive

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <title>首页</title>

</head>

<body>

  <h1>网站首页</h1>

</body>

</html>

响应报文也分为 : 响应行 响应头 响应空行 响应体

##### 响应行(响应报文第一行):

响应行: HTTP/1.1  200  OK

\* 协议版本: HTTP/1.1

\* 响应状态码: 200 (404 320 403...)

\* 响应状态字符串: OK

常见的响应状态码:

\* 200 成功

\* 302 跳转

\* 404 找不到

\* 403 禁止的

\* 500 服务器内部错误

五类响应状态码:

1 信息响应 (100-199)

2 成功响应 (200-299)

3 重定向 (300-399)

4 客户端错误 (400-499)

5 服务器错误(500-599)

##### 响应头(与请求头格式一致):

X-Powered-By: Express

Accept-Ranges: bytes

Cache-Control: public, max-age=0

Last-Modified: Wed, 21 Mar 2018 13:13:13 GMT

ETag: W/"a9-16248b12b64"

Content-Type: text/html; charset=UTF-8

Content-Length: 169

Date: Thu, 22 Mar 2018 12:58:41 GMT

Expires: Thu, 22 Mar 2018 12:58:42 GMT

Connection: keep-alive

Cache-Control: public, max-age=0

Cache-Control: 缓存控制 private:只允客户端缓存数据 public:客户端服务器代理皆可;

Connection: keep-alive:

Conection: 连接设置 (是否保留通信通道)

\*\*\* Content-Type: text/html; charset=UTF-8

Content-Type:响应体的内容类型和响应体的字符集 [响应体]的内容类型

Content-Type: text/css text/html application/JavaScript img/png img/jpeg

Date: Thu, 22 Mar 2018 12:58:41 GMT

Date:响应时间

Expires: Thu, 22 Mar 2018 12:58:42 GMT

Expires:过期时间

Server: BWS/1.1

Server:服务器信息

Set-Cookie: BDKJRIJLNE=10 ; path = /

Set-Cookie:设置cookie

Strict-Transport-Security: max-age = 172800 单位为s

Strict-Transport-Security: 告诉客户端,在172899s之内所有请求报文都必须使用https协议进行加密;

Upgrade-Insecure-Request通常与响应头中的Strict-Transport-Security使用

用以强制使用HTTPS协议(加密，有安全性);

Traceid : 213975275384293483120984310751

Traceid: 跟踪id,形成唯一的标志

X-Ua-Compatible : IE = Edge , chrome = 1

强制IE浏览器使用最新的解析器解析网页,可以使用chrome的内核解析网页;

Content- Length : 231412321

Content-Length:响应体的长度(B)

##### 响应体:

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <title>首页</title>

</head>

<body>

  <h1>网站首页</h1>

</body>

</html>

响应体和请求体都比较灵活 可以容纳任意内容;

但是响应体内容 比较常见的有 html css js JSON PNG

访问百度: 只能看到网页将响应体内容样式渲染后的结果;

##### Chorme浏览器网络控制台

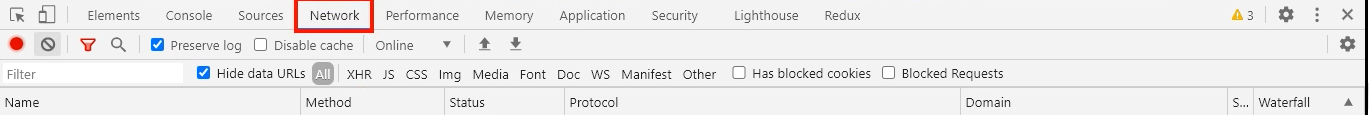
浏览器地址栏上输入URL,点击搜索后会发生什么?

浏览器会将URL收录,并以此封装HTTP请求报文,向服务器发送请求;

服务器接受请求后,根据请求体的内容获取服务器内容,服务器以此封装HTTP响应报文

客户端浏览器接受到响应报文后,根据响应报文的响应体渲染页面或获取数据更新页面

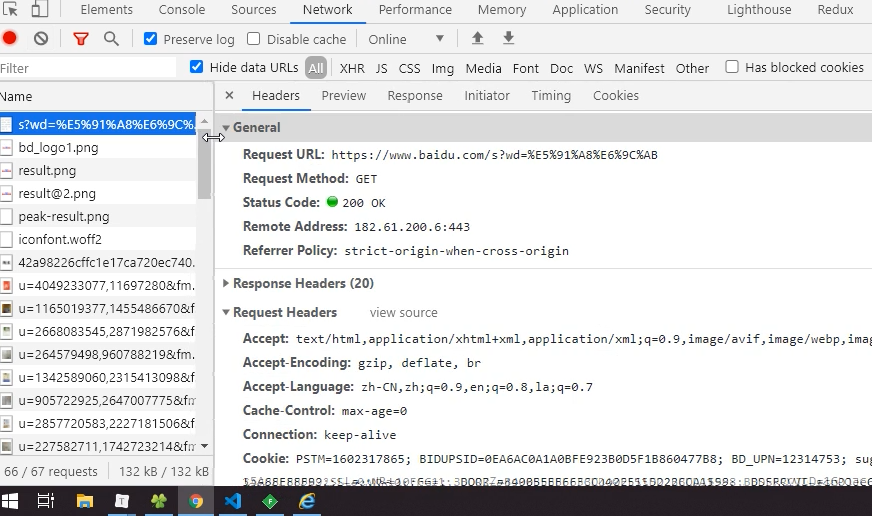
NetWork : 浏览器自带的网络控制台



Headers 头部栏

network网络控制台会列出,网页在加载过程中浏览器所有的请求报文(HTML/CSS/JS/AJAX)

并且第一个请求肯定是获取HTML原始页面的请求





General : 显示大体的请求信息

Response Headers : 响应头

Request Headers: 请求头

Query String Parameters: 搜索栏的原始输入内容(用户输入内容)

如何查看请求行和响应行怎么办,在各自的Headers 下点击 view source

Response 响应栏

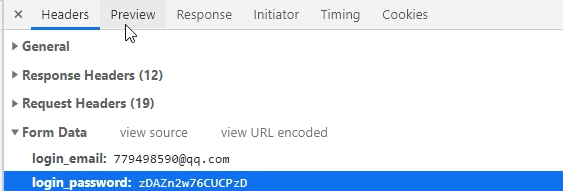


Response响应栏展示完整的响应体;

请求行请求头 响应头行体 都能看了 请求体呢?

首先请求体只有post方法的请求报文有;

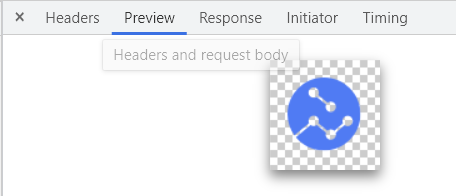
并且若有请求体,Headers栏下会出现Form Data



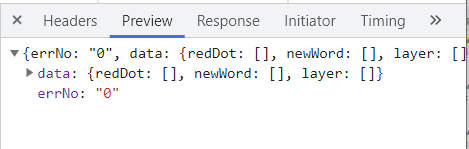
Preview 栏:

Preview栏负责将response栏中的响应体内容解析并展示出来(格式化预览)

(response栏中的响应体都是压缩过的可读性差或者图片的乱码)







\*\*\*\*\*\*十分重要-总结:

