二.HTTP协议

1.URL

|  |
| --- |
| 结构  协议+主机名称+目录结构+文件名  http://cdn.tmooc.cn/bsfile/imgad/A.jpg |
| 完整url结构  <scheme>://<user>:<pwd>@<host>:<port>/<path>;<params>?<query>#<frag>  scheme:方案，协议。以哪种方式获取服务器资源  不区分大小写，常见的协议http/https/ftp  C:\Users\web\Desktop\web1811\web第二阶段练习效果图示\协议及端口号.png  DNS FTP HTTP/HTTPS |
| <user>用户名  <pwd>密码  host： 主机名localhost 、127.0.0.1/域名、IP  port:端口号  path:路径  params：参数，session/cookie 跟踪状态的参数  query:查询字符  frag:锚点 01.html#NO1 |

2.HTTP协议

|  |
| --- |
| 1.什么是HTTP协议  HyperText Transfer Protocol 超文本传输协议  规范了数据是如何打包传输的  2.HTTP历史  正在使用时HTTP/1.1  C:\Users\web\Desktop\web1811\web第二阶段练习效果图示\HTTP协议历史.png  3.详解  C:\Users\web\Desktop\web1811\web第二阶段练习效果图示\请求响应流程.png  把请求流程可以叙述出来  C:\Users\web\Desktop\web1811\web第二阶段练习效果图示\message.png |

三.HTTP的消息

|  |
| --- |
| Request请求消息  是客户端带给服务器的数据  由三部分组成，请求起始行，请求头，请求主体  Response响应消息  服务端发送给客户端的数据  由三部分组成，响应起始行，响应头，响应主体 |

1.Request请求消息

|  |
| --- |
| 1.请求起始行  1)请求的方法  a.get客户端向服务器要数据的时候使用  靠地址栏明文传输字符串，无请求主体(form data)  b.post 客户端向服务器提交数据的时候使用  隐式传输，有请求主体form data  c.delete 客户端想要删除服务器内容(一般禁用)  d.put 客户端想要放数据到服务器(一般禁用)  e.connect 测试连接  f.trace 追踪请求路径  g.option 选项，预请求  h.head表示客户端只获取响应头信息  2)请求的url  3)协议版本 HTTP/1.1  2.请求头信息  1)Host:告诉服务器请求的主机  2)Connection:keep-alive  告诉服务器，进行持久连接  3)User-agent 用户代理  告诉服务器，我自己(浏览器)的类型  4)Accept-Encoding:gzip  告诉服务器，自己能接收的压缩文件的类型  5)Accept-Language:zh-CN,zh  告诉服务器，自己能够接收的自然语言的类型  6)Referer 引用/推荐人  告诉服务器，请求来自于哪个网页  3.请求主体  form data |

2.Response响应消息

|  |
| --- |
| 1)响应起始行  1.http协议版本 http/1.1  2.响应状态码  3.原因短句，对状态码的简短的解释说明  2)响应头信息  1.Date:告诉浏览器，服务器的响应时间点  格林威治时间 北京+8小时  2.Connection:keep-alive;  告诉浏览器，已经启动了持久连接  3.Content-Type:响应主体类型是什么  取值  text/html 响应回来的数据是html文本  text/plain 响应回来的是普通文本(不包含特殊符号)  text/css 响应回来的是样式文件  application/javascript 响应回来的js脚本文件  application/xml 响应回来的是xml格式的字符串  application/json 响应回来的是json格式的字符串  images/jpg.png.... 响应回来的是图片  3)响应主体 |

响应状态码

|  |
| --- |
| 告诉浏览器，服务器的响应状态是什么  1XX:100-199 提示信息  2XX:成功响应 200 ok  301:永久重定向  302:临时重定向  304:请求未被修改，命中缓存  4XX:客户端请求错误  404 NOT Found 请求资源不存在  403 Forbidden 权限不够  405 Method not Allowed 请求方法不被允许  5XX:服务器运行错误  500 服务器内部错误 |

四.缓存

|  |
| --- |
| 客户端将服务器响应回来的数据进行自动保存  当再次访问的时候，直接使用保存的数据  缓存的优点和缺点  1.减少了冗余数据的传输，节省客户端流量  2.节省了服务器带宽  3.降低了对服务器资源的消耗和运行要求  4.降低了由于远程传输而造成的延时加载 |
| C:\Users\web\Desktop\web1811\web第二阶段练习效果图示\缓存流程.png  1.请求--无缓存--连服务器--存缓存--客户端得到  2.请求--有缓存--够新鲜--使用缓存--客户端得到  3.请求--有缓存--不新鲜--连接服务器查看是否过期--没过期--更新缓存的新鲜度--客户端得到  4. 请求--有缓存--不新鲜--连接服务器查看是否过期--已过期--连服务器--存缓存--客户端得到 |
| 与缓存相关的消息头  1.Cache-Control http/1.1的用法  从服务器将文档传到客户端之时器，  此文档处于新鲜的秒数，是一个相对时间  max-age=新鲜的秒数/0 不缓存  2.Expires http/1.0的用法  指定过去的确切时间点，是一个具体的时间点  Expires:Tue,31 Jul 2019 02:56:16 GMT; |

在网页中添加缓存

|  |
| --- |
| <meta http-equiv="消息头" content="值">  <meta http-equiv="Cache-Control" content="max-age=3600"> |

HTTP性能优化

|  |
| --- |
| 1.HTTP连接的过程  发送请求--->建立连接--->处理请求--->访问资源--->构建响应--->发送响应--->记录日志  2.HTTP连接性能的优化  减少连接的创建次数(开启持久连接)  减少请求次数  提高服务器端运行速度  尽可能减少响应数据的长度 |

安全的HTTP协议

|  |
| --- |
| HTTPS协议，安全版本的HTTP  SSL，为数据通信提供安全支持  1.客户端发送请求消息时，在ssl层加密  服务器接收加密文件，在ssl层解密  得到请求明文，对请求进行处理  2.服务器发送响应消息时，在SSL层进行加密  客户端接收加密文件，在SSL层进行解密  得到响应明文，解析相应内容 |