# HTTP

## RESTful

REST 指的是一组架构约束条件和原则。满足这些约束条件和原则的应用程序或设计就是 RESTful。

* GET get方法在Rest中主要用于获取资源，能够发送参数，不过有限制，且参数都会以?开头的形 式附加在URL尾部。 规范的get方法处理器应该是幂等的，也就是说对一个资源不论发送多少次get请求都不会更改数据或造成破坏。
* POST post方法在Rest请求中主要用于添加资源，参数信息存放在请求报文的消息体中相对安全，且可发送较大信息
* PUT put方法在Rest中主要用于更新资源，因为大多数浏览器不支持put和delete，会自动将put和delete请求转化为get和post. 因此为了使用put和delete方法, 需要以post发送请求，在表单中使用隐藏域发送真正的请求。 put方法的参数是同post一样是存放在消息中的，同样具有安全性，可发送较大信息。 put方法是幂等的，对同一URL资源做出的同一数据的任意次put请求其对数据的改变都是一致的。
* DELETE Delete在Rest请求中主要用于删除资源，因为大多数浏览器不支持put和delete，会自动将put和delete请求转化为get和post。 因此为了使用put和delete方法,需要以post发送请求，在表单中使用隐藏域发送真正的请求。 Delete方法的参数同post一样存放在消息体中,具有安全性，可发送较大信息 Delete方法是幂等的，不论对同一个资源进行多少次delete请求都不会破坏数据

https://blog.csdn.net/jnshu\_it/article/details/80203696

## GET和POST的区别

* GET在浏览器回退时是无害的，而POST会再次提交请求。
* GET产生的URL地址可以被Bookmark，而POST不可以。
* GET请求会被浏览器主动cache，而POST不会，除非手动设置。
* GET请求只能进行url编码，而POST支持多种编码方式。
* GET请求参数会被完整保留在浏览器历史记录里，而POST中的参数不会被保留。
* GET请求在URL中传送的参数是有长度限制的，而POST么有。
* 对参数的数据类型，GET只接受ASCII字符，而POST没有限制。
* GET比POST更不安全，因为参数直接暴露在URL上，所以不能用来传递敏感信息。

## Accept和Content-Type

Accept 请求头用来告知客户端可以处理的内容类型，这种内容类型用MIME类型来表示。 服务器使用 Content-Type 应答头通知客户端它的选择。

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

1.Accept属于请求头， Content-Type属于实体头。 Http报头分为通用报头，请求报头，响应报头和实体报头。 请求方的http报头结构：通用报头|请求报头|实体报头 响应方的http报头结构：通用报头|响应报头|实体报头

2.Accept代表发送端（客户端）希望接受的数据类型。 比如：Accept：text/xml; 代表客户端希望接受的数据类型是xml类型

Content-Type代表发送端（客户端|服务器）发送的实体数据的数据类型。 比如：Content-Type：text/html; 代表发送端发送的数据格式是html。

二者合起来， Accept:text/xml； Content-Type:text/html 即代表希望接受的数据类型是xml格式，本次请求发送的数据的数据格式是html。

## 状态码

|  |  |  |
| --- | --- | --- |
| 状态码 | 类别 | 描述 |
| 1xx | Informational（信息状态码） | 接受请求正在处理 |
| 2xx | Success（成功状态码） | 请求正常处理完毕 |
| 3xx | Redirection（重定向状态码） | 需要附加操作已完成请求 |
| 4xx | Client Error（客户端错误状态码） | 服务器无法处理请求 |
| 5xx | Server Error（服务器错误状态码） | 服务器处理请求出错 |

[HTTP 响应代码](https://developer.mozilla.org/zh-CN/docs/Web/HTTP/Status)

## HTTP缓存

https://segmentfault.com/a/1190000010690320

## 如何处理不让别人盗用你的图片，访问你的服务器资源

* http header, 对refer做判断看来源是不是自己的网站，如果不是就拒绝
* 通过session校验，如果不通过特定服务生成cookie和session就不能请求得到资源

## Http与Https的区别

* HTTP 的URL 以http:// 开头，而HTTPS 的URL 以https:// 开头
* HTTP 是不安全的，而 HTTPS 是安全的
* HTTP 标准端口是80 ，而 HTTPS 的标准端口是443
* 在OSI 网络模型中，HTTP工作于应用层，而HTTPS 的安全传输机制工作在传输层
* HTTP 无法加密，而HTTPS 对传输的数据进行加密
* HTTP无需证书，而HTTPS 需要CA机构wosign的颁发的SSL证书

https://zhuanlan.zhihu.com/p/33778904

## 什么是Http协议无状态协议?怎么解决Http协议无状态协议?

无状态协议对于事务处理没有记忆能力。缺少状态意味着如果后续处理需要前面的信息也就是说， 当客户端一次HTTP请求完成以后，客户端再发送一次HTTP请求，HTTP并不知道当前客户端是一个”老用户“。

可以使用Cookie来解决无状态的问题，Cookie就相当于一个通行证，第一次访问的时候给客户端发送一个Cookie， 当客户端再次来的时候，拿着Cookie(通行证)，那么服务器就知道这个是”老用户“。

https://zhuanlan.zhihu.com/p/33778904

## 常用的HTTP方法有哪些

* GET：用于请求访问已经被URL（统一资源标识符）识别的资源，可以通过URL传参给服务器。
* POST：用于传输信息给服务器，主要功能与Get方法类似，但一般推荐POST方式。
* PUT：传输文件，报文主体包含文件内容，保存到对应URL位置。
* HEAD：获取报文首部，与GET方法类似，只是不返回报文主体，一般用于验证URL是否有效。
* DELET：删除文件，与PUT方法相反，删除对应URL位置的文件。
* OPTIONS：查询相应URL支持的HTTP方法。

## HTTPS 握手机制

* [HTTPs入门, 图解SSL从回车到握手](https://zhuanlan.zhihu.com/p/25587986)
* [SSL/TLS协议运行机制的概述](https://www.ruanyifeng.com/blog/2014/02/ssl_tls.html)
* [SSL/TLS 握手过程详解](https://www.jianshu.com/p/7158568e4867)

## options 预检请求

[预检请求](https://developer.mozilla.org/zh-CN/docs/Web/HTTP/CORS#%E9%A2%84%E6%A3%80%E8%AF%B7%E6%B1%82)

## dns查询过程，dns用什么协议发起dns查询的

[为什么 DNS 使用 UDP 协议](https://draveness.me/whys-the-design-dns-udp-tcp/)

## keep-alive 和多路复用的区别

keep-alive 是指 TCP 处理完一个 HTTP 连接后不关闭，可以继续处理下一个 HTTP 连接。多路复用是 HTTP2 的特性，指多个 HTTP 连接可以同时在一个 TCP 连接上处理。

## websocket握手过程

[socket 及 websocket的握手过程](https://blog.csdn.net/yournevermore/article/details/103067079)