# OKHttp3

交流群： 167481839

# 1.快速使用

1.OKHttp3 快速使用

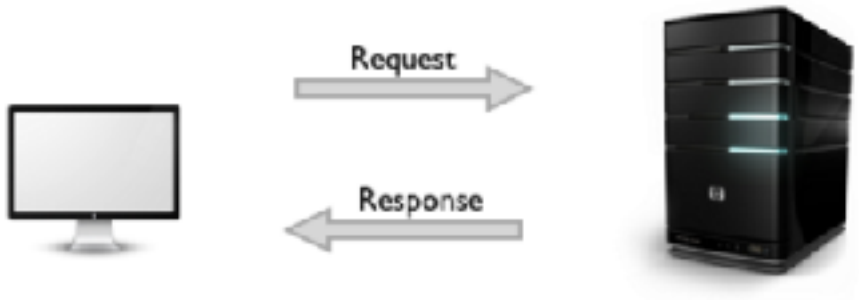
2.Http 协议简单介绍

1.官网地址：http://square.github.io/okhttp/

HTTP是一个属于应用层的面向对象的协议，由于其简捷、快速的方式，适用于分布式超媒体信息系统。

它于1990年提出，经过几年的使用与发展，得到不断地完善和扩展。目前在WWW中使用的是HTTP/1.0的第六版，HTTP/1.1的规范化工作正在进行之中，而且HTTP-NG(Next Generation of HTTP)的建议已经提出。

我们把Http协议中通信的两方称作**Client**和**Server**(或Host)，Client向Server端经过http协议发送一个Request，Server端收到Request后经过一些列的处理返回Client一个Response，整个过程如下图所示：



菜鸟窝
WWW.CNIAOE.COM

`http`?

- 全称：超文本传输协议`(HyperText Transfer Protocol)`
- 作用：设计之初是为了将超文本标记语言`(HTML)`文档从Web服务器传送到客户端的浏览器。现在`http`的作用已不局限于`HTML`的传输。
- 版本：`http/1.0` `http/1.1*` `http/2.0`

# URL详解

一个示例URL

```
http://www.mywebsite.com/sj/test;id=8079?name=sviergn&x=true#stuff

Schema: http
host: www.mywebsite.com
path: /sj/test
URL params: id=8079
Query String: name=sviergn&x=true
Anchor: stuff
```

- `scheme`：指定低层使用的协议(例如：`http`，`https`，`ftp`)
- `host`：`HTTP`服务器的`IP`地址或者域名
- `port#`：`HTTP`服务器的默认端口是`80`，这种情况下端口号可以省略。如果使用了别的端口，必须指明，例如`http://www.mywebsite.com:8080/`
- `path`：访问资源的路径
- `url-params`
- `query-string`：发送给`http`服务器的数据
- `anchor`：锚

## 无状态的协议

`http`协议是无状态的：

> **同一个客户端的这次请求和上次请求是没有对应关系，对http服务器来说，它并不知道这两个请求来自同一个客户端。**

解决方法：`Cookie`机制来维护状态

**既然Http协议是无状态的，那么`Connection:keep-alive`又是怎样一回事?**

> 无状态是指协议对于事务处理没有记忆能力，服务器不知道客户端是什么状态。从另一方面讲，打开一个服务器上的网页和你之前打开这个服务器上的网页之间没有任何联系。

1. Request
2. Response

菜鸟窝
WWW.CNIAO5.COM

1. `Request` 消息的结构：三部分

> 第一部分叫`Request line`（请求行）， 第二部分叫`http header`，第三部分是`body`
>
> - **请求行**：包括`http`请求的种类，请求资源的路径，`http`协议版本
> - `http header`：`http`头部信息
> - `body`：发送给服务器的`query`信息
>
> 当使用的是"`GET`"方法的时候，`body`是为空的（`GET`只能读取服务器上的信息，`post`能写入）

```
GET /hope/ HTTP/1.1   //---请求行
Host: ce.sysu.edu.cn
Accept: */*
Accept-Encoding: gzip, deflate, sdch
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.6
Cache-Control: max-age=0
Cookie:.........
Referer: http://ce.sysu.edu.cn/hope/
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like G
ecko) Chrome/44.0.2403.130 Safari/537.36

---分割线---

POST /hope/ HTTP/1.1   //---请求行
Host: ce.sysu.edu.cn
Accept: */*
Accept-Encoding: gzip, deflate, sdch
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.6
Cache-Control: max-age=0
Cookie:.........
Referer: http://ce.sysu.edu.cn/hope/
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like G
ecko) Chrome/44.0.2403.130 Safari/537.36

    body...
```

## 2. Response消息的结构

> 也分为三部分，第一部分叫request line, 第二部分叫request header，第三部分是body
>
> - `request line`：协议版本、状态码、`message`
> - `request header`：`request`头信息
> - `body`：返回的请求资源主体

```
HTTP/1.1 200 OK
Accept-Ranges: bytes
Content-Encoding: gzip
Content-Length: 4533
Content-Type: text/html
Date: Sun, 06 Sep 2015 07:56:07 GMT
ETag: "2788e6e716e7d01:0"
Last-Modified: Fri, 04 Sep 2015 13:37:55 GMT
Server: Microsoft-IIS/7.5
Vary: Accept-Encoding
X-Powered-By: ASP.NET

<!DOCTYPE html>
...
<>
...
```

GET　请求获取Request-URI所标识的资源

POST　在Request-URI所标识的资源后附加新的数据

HEAD　请求获取由Request-URI所标识的资源的响应消息报头

PUT　请求服务器存储一个资源，并用Request-URI作为其标识

DELETE 请求服务器删除Request-URI所标识的资源

TRACE 请求服务器回送收到的请求信息，主要用于测试或诊断

CONNECT 保留将来使用

OPTIONS 请求查询服务器的性能，或者查询与资源相关的选项和需求

## get 和 post 区别

http协议定义了很多与服务器交互的方法，最基本的有4种，分别是GET,POST,PUT,DELETE。一个URL地址用于描述一个网络上的资源，而HTTP中的GET，POST，PUT，DELETE就对应着对这个资源的查，改，增，删4个操作。我们最常见的就是GET和POST了。GET一般用于获取/查询资源信息，而POST一般用于更新资源信息.

1. GET提交的数据会放在URL之后，以?分割URL和传输数据，参数之间以&相连，
   如EditPosts.aspx?name=test1&id=123456。POST方法是把提交的数据放在HTTP包的Body中。
2. GET提交的数据大小有限制（因为浏览器对URL的长度有限制），而POST方法提交的数据没有限制.
3. GET方式需要使用Request.QueryString来取得变量的值，而POST方式通过Request.Form来获取变量的值。
4. GET方式提交数据，会带来安全问题，比如一个登录页面，通过GET方式提交数据时，用户名和密码将出现在URL上，如果页面可以被缓存或者其他人可以访问这台机器，就可以从历史记录获得该用户的账号和密码.

## 状态码

`Response` 消息中的第一行叫做**状态行**，由 `HTTP` 协议版本号， **状态码**， **状态消息** 三部分组成。

状态码用来告诉 `HTTP` 客户端， `HTTP` 服务器是否产生了预期的 `Response`.

`HTTP/1.1` 中定义了 5 类状态码。

状态码由三位数字组成，第一个数字定义了响应的类别

1xx：指示信息--表示请求已接收，继续处理

2xx：成功--表示请求已被成功接收、理解、接受

3xx：重定向--要完成请求必须进行更进一步的操作

4xx：客户端错误--请求有语法错误或请求无法实现

5xx：服务器端错误--服务器未能实现合法的请求

200 OK　　//客户端请求成功

400 Bad Request　//客户端请求有语法错误，不能被服务器所理解

401 Unauthorized //请求未经授权，这个状态代码必须和WWW-Authenticate报头域一起使用

403 Forbidden　//服务器收到请求，但是拒绝提供服务

404 Not Found　//请求资源不存在，eg：输入了错误的URL

500 Internal Server Error //服务器发生不可预期的错误

503 Server Unavailable　//服务器当前不能处理客户端的请求，一段时间后可能恢复正常

# Http Headers

1.GET

2.POST - FORM

3.POST- JSON

4.自动转换JSON为JavaBean