HTTP

**URI、URL、URN**

参考：<http://blog.csdn.net/zcc7up/article/details/73551250>

http://blog.csdn.net/readiay/article/details/52862379#1

URL：统一资源定位符。

URI：统一资源标识符。

URN：统一资源名称。

引用网上举出的生动的例子：大白话，就是URI是抽象的定义，不管用什么方法表示，只要能定位一个资源，就叫URI，本来设想的的使用两种方法定位：1，URL，用地址定位；2，URN 用名称定位。举个例子：去村子找个具体的人（URI），如果用地址：某村多少号房子第几间房的主人 就是URL， 如果用身份证号+名字 去找就是URN了。

* URI

URI是一个纯粹的语法结构，包含用来指定资源的路径字符串的各种组成部分。URI不包含任何用于访问资源的方法，它的唯一作用就是解析。URL和URN是URI的子集，任何一个URL和URN都可以叫URI。URI提供一个toURL()方法将URI转化成一个URL，如果转换不成功会抛出MalformedURLException或者在执行过程中可能会抛出：IllegalArgumentException

URI实现了Serializable和Comparable接口，由此看来URI是可以被序列化的，并且需要提供Comparable接口的compareTo方法。

一个URI有九个组成部分：



* **格式：**<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<fragment>
* **scheme：**获取资源使用的协议，例如http、ftp等，没有默认值
* **user:** password：用户名与密码，这个是一个特殊的存在，一般访问ftp时会用到，他显示的表明了访问资源的用户名与密码。但是这个可以不写，不写的话可能会让你输入用户名密码
* **host：**主机，访问那台主机，有时候可以是IP，有时候是主机名，例如www.baidu.com
* **port：**端口，访问主机时的端口，如果http访问默认80，可以省略。
* **path：**通过host:port我们能找到主机，但是主机上文件很多，通过path则可以定位具体文件。例如www.baidu.com/file/index.html。则path是/file/index.html，表示我们访问/file/index.html这个文件，他很像linux上的路径。
* **params：**这个很少见，主要作用就是像服务器提供额外的参数，用来表示本次请求的一些特性。例如ftp传输模式有两种，二进制和文本，你肯定不希望使用文本形式传输二进制图片，这样你的图片下载下来后可能没法看了。为了向应用程序提供更丰富的信息，URL中有个专门的部分来表示这种参数。例如ftp://file.qiandu.com/pub/guid.pdf;type=d其中的type=d就是params
* **query：**通过get方式请求的参数，例如：www.qiandu.com/index.html?username=dgh&passwd=123
* **fragment：**例如www.qiandu.com/index.html#1。当html页面比较长时，我们通常会将其分为好几段，#1就可以 快速定位到某一段。

上面的九个成员变量都是transient类型，意味着，序列化的时候，这些值是不会被序列化的。整个对象中唯一可以被序列化的就是

private volatile String string;

这个string是根据上面的成员按照URI的规则生成的一个字符串。并且可以调用URI.toURL()和URL.toURI()进行相互转换。

* URL

URL是URI的一个子集，**它是指向某个“资源”的一个地址**。URI 类不包含任何用于访问资源的方法，它的惟一作用就是解析。相反的是，URL类可以通过调用openConnection()可以打开一个到达资源的流，但是它需要指定一个具体的协议： http:、https:、ftp:、本地文件系统（file:）和 JAR 文件（jar:）。 下面是一些URL的例子：

http://java.sun.com/index.html#chapter1

http://www.socs.uts.edu.au:80/MosaicDocs-old/url-primer.html

ftp://www.xx.com/1.txt

**URLConnection**

参考：<https://www.kancloud.cn/kancloud/android-tutorial/87221>

**http请求流程**

在 Android上发送 HTTP请求的方式一般有两种，HttpURLConnection和 HttpClient。目前HttpClient已经被废。URLConnection是个抽象类，HttpURLConnection继承自URLConnection并且扩展了更多的方法。当操作一个 URLConnection 对象时，必须像下面这样非常小心地安排操作步骤：

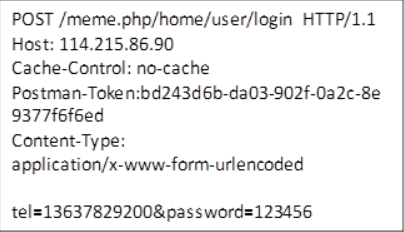




上面是一个标准的Get和Post方式请求的操作步骤，我们只是看到了它的请求步骤，但是这个请求到底向服务器发送了什么，还有服务器返回了什么我们都不清楚，看下面：

客户端——>服务端：请求方法 + 统一资源定位符(URL) + 协议版本号(一般是1.1)+MIME信息(多个消息头) + 一个空行 + 请求包体

客户端——>客户端：协议版本号 + 状态行(处理结果) + 多个信息头 + 空行 + 实体内容(比如返回的HTML)





**http请求方式**

在开发中我们用的最多的就是“GET”“POST”两种方式，但是还有其他的方式：

* **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信息报头
* **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

**GET和POST对比：**

**Get：**在请求的URL地址后以？隔开提交给服务器的参数，多个参数以&进行分隔，数据容量最大不超过2K。Get方式安全性低，但是执行效率高。如：:[http://xxx?username=…&pawd=…](http://xxx/?username=%E2%80%A6&pawd=%E2%80%A6%E8%BF%99%E7%A7%8D%E5%B0%B1%E6%98%AFGET)

**Post:**在请求的实体内容中添加提交各服务器的参数，传输没有大小限制。Post方式安全性高，这里所说的安全性高指的是提交数据，而不是查询数据。

**http请求头和响应头**

具体头信息可参考：<https://www.kancloud.cn/kancloud/android-tutorial/87222>

请求头可以通过setRequestProperty()方法来设置，具体可以设置哪些请求头和服务器返回的响应头包含什么可以参考上面链接。

**http请求Cookie问题的处理**

首先需要注意Cookie只是Session机制的一种常用形式，是客户端的一个唯一标识。所以Cookie和Session指的并不是一个玩意。另外，确定客户端唯一标识的另一个方法就是URL重写。

Cookie在http请求中起到的作用：一般登陆的时候，服务器会通过Set-Cookie响应头返回一个Cookie，客户端会保存这个Cookie，在客户端后续的访问中，会携带这个Cookie在服务器进行验证身份，当这个Cookie丢失或者过时，就会提示客户端登录超时，未登录等。而客户端通过HttpConnection请求登录成功后，调用getHeaderField("Set-Cookie");方法获取Cookie后保存到本地，并在下次请求中setRequestProperty("Cookie",cookie)带上这个Cookie。

另外，通过URL重写，也可以保证客户端的唯一：就是在请求链接上添加一个&sessionid=xxxxx这样的参数，然后由服务器来解析判断！

**http请求参数构建**

**Get请求的参数构成以及传递:**

**构成**：在URL的问号后面可以携带参数，每个参数都具有“名字 = 值”的形式，而这些参数之间用 & 字符分隔开。

**传递**：参数的值传递时将遵循下面的规则，使用 URL 编码模式进行编码：

● 保留字符 A-Z、a-z、0-9 以及 . - \* \_。

● 用 + 字符替换所有的空格。

● 将其他所有字符编码为 UTF-8，并将每个字节都编码为 % 后面紧跟一个两位的十六进制数字。

例如，若要发送街道名 New York，NY，可以使用 New+York%2C+NY，因为十六进制 数 2c（即十进制数 44）是“，”的 ASCII 码值。 这种编码方式使得在任何中间程序中都不会混入空格，并且也不需要对其他特殊字符进 行转换。

**Post请求的参数构成以及传递:**

**构成**：Post请求方式的参数也是“名字 = 值”的形式，而这些参数之间用 & 字符分隔开。

**传递**：首先调用setDoOutput(true);设置这个connection可以向服务器写数据，默认是不可以向服务器写数据的。然后getOutputStream();打开一个输出流，最后将拼接好的参数字符串写入到这个流中。如下：

