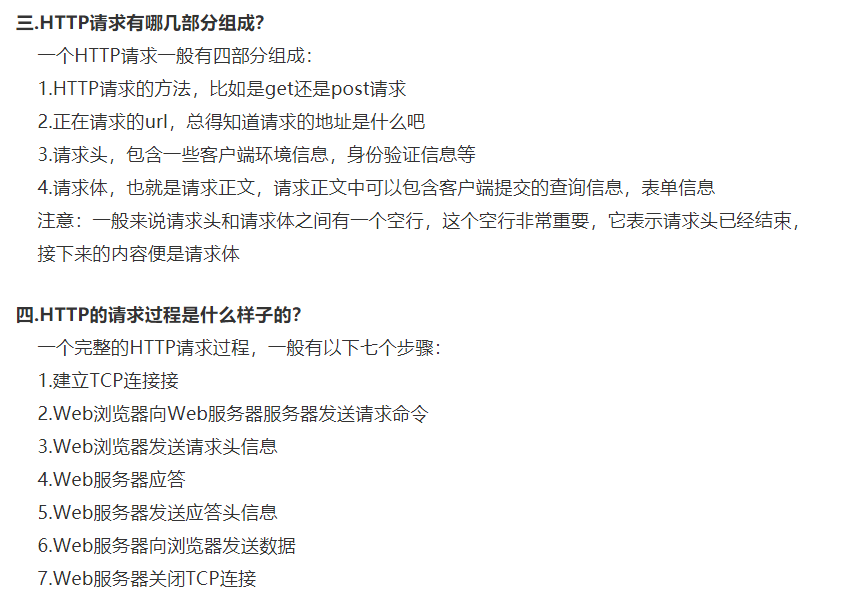
# 测试开发

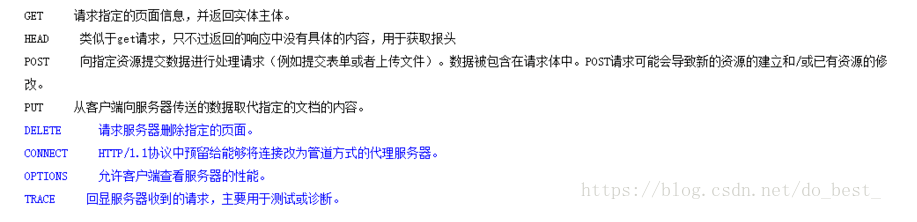
1. **计算机基础💬📝✏️📌🏃‍♀️💡🎉✍️ 📖 🌲 ✔️⌨🚩🔍 ✨**
   1. [熟悉的http的方法都有哪些？](https://www.cnblogs.com/chengzi-he/p/9443823.html)🔍

HyperText Transfer Protocol,超文本传输协议

* + 1. HTTP是一个基于TCP/IP通信协议来传递数据。是客户端浏览器或其他程序与Web服务器之间的应用层通信协议。
    2. 特点
       1. 工作在客户端-服务端架构上
       2. 浏览器作为HTTP客户端通过url向HTTP服务端即Web服务器发送所有请求
       3. HTTP默认端口号为80，但是你也可以改为8080或者其他端口
       4. HTTP是无连接的，即每次连接只处理一个请求，服务器处理完客户的请求，并收到客户的应答后，即断开连接，采用这种方式可以节省传输时间（用完即走，再用时必须重新发起请求，不存在占着茅坑的现象）
       5. HTTP是无状态的，即对于事务处理没有记忆能力，如果后续处理需要前面的信息，必须重传
       6. HTTP是媒体独立的：即只要是客户端和服务器知道如何处理的数据内容，任何类型的数据都可以通过HTTP发送
    3. http请求由哪几部分组成（请求行/请求头/请求体）
       1. 请求报文(请求行/请求头/请求数据/空行)
       2. 响应报文(状态行、消息报头、响应正文)



* + 1. http1.0新增了五种请求方法：



* + - 1. get：请求指定页面的信息，并返回实体主体
      2. post：向指定资源提交数据进行处理请求。数据被包含在请求体中。post请求可能会导致新的资源的建立和/或已有资源的修改
      3. put：从客户端向服务器传送的数据取代指定的文档的内容
      4. [Get和post区别](https://www.cnblogs.com/logsharing/p/8448446.html)🔍

最直观的区别就是GET把参数包含在URL中，POST通过request body传递参数。

* + - * 1. get请求一般用于获取服务器的信息

1.使用url传递参数，也就是参数是可见的

2.对发送信息的数量也有限制，一般在2000个字符左右

* + - * 1. post请求一般用于修改服务器上的资源

表单一般用post方式提交，post请求的参数一般都嵌入在请求体中，对用户不可见，可以使用工具拦截，如fiddler等

1.不使用url传递参数，即参数对用户是不可见的

2.对发送的信息数量是无限制的

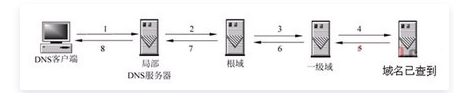
* + - * 1. GET/POST都是TCP链接，GET产生一个TCP数据包；POST产生两个TCP数据包。
    1. TCP和UDP的区别
       1. **TCP提供可靠的通信传输**
          1. 优点：**可靠，稳定。** TCP的可靠体现在TCP在传递数据之前，会有三次握手来建立连接，而且在数据传递时，有确认、窗口、重传、拥塞控制机制，在数据传完后，还会断开连接用来节约系统资源。
          2. 缺点： **慢，效率低，占用系统资源高**，易被攻击。 TCP在传递数据之前，要先建连接，这会消耗时间，而且在数据传递时，确认机制、重传机制、拥塞控制机制等都会消耗大量的时间，而且要在每台设备上维护所有的传输连接，事实上，每个连接都会占用系统的CPU、内存等硬件资源。 而且，因为TCP有确认机制、三次握手机制，这些也导致TCP容易被人利用，实现DOS、DDOS、CC等攻击。
       2. **UDP则常被用于广播和细节控制交给应用的通信传输**
          1. 优点： **快，比TCP稍安全** 。UDP没有TCP的握手、确认、窗口、重传、拥塞控制等机制，UDP是一个无状态的传输协议，所以它在传递数据时非常快。
          2. 缺点：**不可靠**，**不稳定** 。因为UDP没有TCP那些可靠的机制，在数据传递时，如果网络质量不好，就会很容易丢包。
       3. 二者区别
          1. 1.TCP面向连接（如打电话要先拨号建立连接）;UDP是无连接的，即发送数据之前不需要建立连接
          2. 2.TCP提供可靠的服务。也就是说，通过TCP连接传送的数据，无差错，不丢失，不重复，且按序到达;UDP尽最大努力交付，即不保证可靠交付
          3. 3.TCP面向字节流，实际上是TCP把数据看成一连串无结构的字节流;UDP是面向报文的
          4. UDP没有拥塞控制，因此网络出现拥塞不会使源主机的发送速率降低（对实时应用很有用，如IP电话，实时视频会议等）
          5. 4.每一条TCP连接只能是点到点的;UDP支持一对一，一对多，多对一和多对多的交互通信
          6. 5.TCP首部开销20字节;UDP的首部开销小，只有8个字节
          7. 6.TCP的逻辑通信信道是全双工的可靠信道，UDP则是不可靠信道
    2. [输入网址到获得页面的网络请求过程](https://blog.csdn.net/qq_36520235/article/details/82559847)🔍
       1. ①域名解析
       2. ②建立TCP连接（浏览器向服务器请求建立连接，发起三次握手）
       3. ③根据SpringMVC后台业务返回数据，并把数据填充到HTML页面上，然后返回给浏览器
       4. ④浏览器进行处理
       5. ⑤绘制网页
    3. http状态码1-5表示的含义
       1. 1xx:信息类，表示收到Web浏览器请求，正在进一步处理中
       2. 2xx:成功，表示用户请求被正确接收，理解和处理
       3. 3xx:重定向，表示请求没有成功，客户端必须采取进一步的动作
       4. 4xx:客户端错误，表示客户端提交的请求有错误，如404 NOT Found,请求的文件不存在
       5. 5xx:服务器错误，表示服务器不能完成对请求的处理
  1. 计算机网络体系结构分层



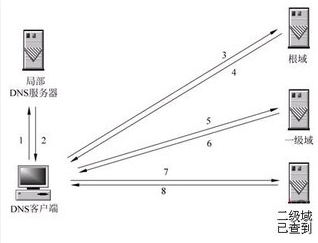
* 1. 传输层协议、应用层协议有哪些？
     1. 传输层协议：传输控制协议TCP；用户数据报协议UDP
     2. 网络层协议：地址解析协议，即ARP；网际互连协议，即IP；Internet控制报文协议，即ICMP；互联网组管理协议IGMP；路由信息协议，即RIP
     3. 应用层协议：超文本传输协议-HTTP；文件传输协议-FTP（文件传输协议有基于TCP的FTP和基于UDP的简单文件传输协议TFTP）；简单邮件传输协议-SMTP

HTTP默认的端口号为80，HTTPS的端口号为443；

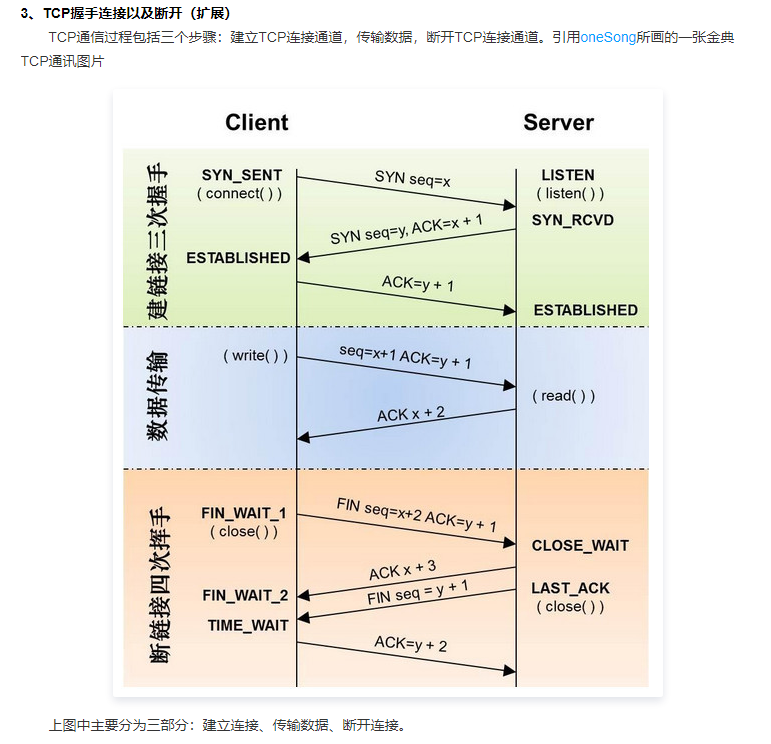
* 1. dns解析以及解析的过程，主机中的host
     1. DNS-域名系统。通过主机名，最终得到该主机名对应的IP地址的过程叫做域名解析（或主机名解析）
     2. **DNS查询的两种方式：递归查询和迭代查询**
        1. **递归解析 ：**
           1. 当局部DNS服务器自己不能回答客户机的DNS查询时，它就需要向其他DNS服务器进行查询。此时有两种方式，如图所示的是递归方式。局部DNS服务器自己负责向其他DNS服务器进行查询，一般是先向该域名的根域服务器查询，再由根域名服务器一级级向下查询。最后得到的查询结果返回给局部DNS服务器，再由局部DNS服务器返回给客户端。
           2. 简单来讲，就是参与此次寻找IP的所有服务器，最后都能够得到该域名对应的IP信息（将信息进行往返传送！）



* + - 1. **迭代解析**
         1. 当局部DNS服务器自己不能回答客户机的DNS查询时，也可以通过迭代查询的方式进行解析，如图所示。**局部DNS服务器不是自己向其他DNS服务器进行查询，而是把能解析该域名的其他DNS服务器的IP地址返回给客户端DNS程序，客户端DNS程序再继续向这些DNS服务器进行查询，直到得到查询结果为止。**也就是说，迭代解析只是帮你找到相关的服务器而已，而不会帮你去查。比如说：[baidu.com](http://baidu.com/)的服务器ip地址在192.168.4.5这里，你自己去查吧，本人比较忙，只能帮你到这里了。
         2. 简单的来讲，就是只有最后一台服务器与最初的服务器进行该域名/IP信息的传送！



* 1. TCP建立连接以及断开



* + 1. **三次握手：TCP连接的建立**。目的是“为了**防止已失效的连接请求报文段突然又传送到了服务端，因而产生错误**”。

请求建立连接-同意创建新连接-确认收到服务器端同意连接的信号

* + 1. **四次挥手：TCP连接的释放(解除)。**连接的释放必须是一方主动释放，另一方被动释放。

请求释放连接-接收到客户端发送的释放连接的请求-已经准备好释放连接了-接收到服务器准备好释放连接的信号

* + 1. 为什么“握手”是三次，“挥手”却要四次？
       1. 二次握手过程中：不仅有请求连接标志还有确认报文
       2. 挥手过程中：释放连接报文、确认接收报文是分开传输的
       3. 释放时候为何要分开？
          1. 建立连接时，被动方服务器不需要任何准备，可以直接返回SYN和ACK报文，开始建立连接
          2. 释放连接时，被动方服务器接收到释放连接的请求时，并不能立即释放连接，因为还有必要的数据需要处理，所以服务器先返回ACK确认报文，经过CLOSE-WAIT阶段准备好释放连接之后，才返回FIN释放连接报文。
  1. 页面刷新不出来是什么原因

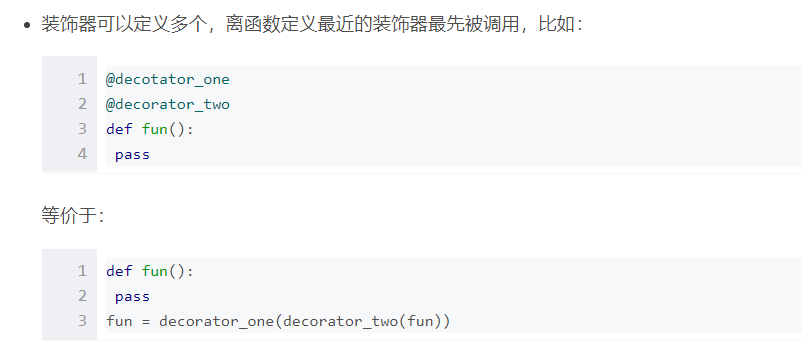
1. **python基础**
   1. [python中元组和列表的区别](https://blog.csdn.net/Star_SDK/article/details/80600673)🔍
      1. 列表是动态数组，它们可变且可以重设长度（改变其内部元素的个数）。
      2. 元组是静态数组，它们不可变，且其内部数据一旦创建便无法改变。

虽然它们不支持改变大小，但是我们可以将两个元组合并成一个新元组

* + 1. 元组缓存于Python运行时环境，这意味着我们每次使用元组时无须访问内核去分配内存。
  1. [python中的魔法方法有哪些](https://www.cnblogs.com/hz2lxt/p/13219223.html)🔍
     1. 1.[\_\_str\_](https://www.cnblogs.com/hz2lxt/p/13219223.html#一、__str__)
     2. 2.[\_\_repr\_](https://www.cnblogs.com/hz2lxt/p/13219223.html#二、__repr__)
     3. 3.[\_\_format\_](https://www.cnblogs.com/hz2lxt/p/13219223.html#三、__format__)
     4. 4.[\_\_del\_](https://www.cnblogs.com/hz2lxt/p/13219223.html#四、__del__)
     5. 5.[\_\_dict\_和\_slots\_](https://www.cnblogs.com/hz2lxt/p/13219223.html#五、__dict__和__slots__)
     6. 6.[\_\_item、attr\_系列](https://www.cnblogs.com/hz2lxt/p/13219223.html#六、__item__、__attr__系列)
     7. 7.[\_\_init\_](https://www.cnblogs.com/hz2lxt/p/13219223.html#七、__init__)
     8. 8.[\_\_new\_](https://www.cnblogs.com/hz2lxt/p/13219223.html#八、__new__)
     9. 9.[\_\_call\_](https://www.cnblogs.com/hz2lxt/p/13219223.html#九、__call__)
     10. 10.[\_\_doc\_](https://www.cnblogs.com/hz2lxt/p/13219223.html#十、__doc__)
     11. 11.[\_\_iter\_和\_next\_](https://www.cnblogs.com/hz2lxt/p/13219223.html#十一、__iter__和__next__)
     12. 12.[\_\_enter\_和\_exit\_](https://www.cnblogs.com/hz2lxt/p/13219223.html#十二、__enter__和__exit__)
     13. 13.[\_\_len\_](https://www.cnblogs.com/hz2lxt/p/13219223.html#十三、__len__)
     14. 14.[\_\_hash\_](https://www.cnblogs.com/hz2lxt/p/13219223.html#十四、__hash__)
     15. 15.[\_\_eq\_](https://www.cnblogs.com/hz2lxt/p/13219223.html#十五、__eq__)
  2. [python中的装饰器是什么？](https://blog.csdn.net/weixin_37972723/article/details/80666877?utm_medium=distribute.pc_relevant.none-task-blog-2%7Edefault%7EBlogCommendFromMachineLearnPai2%7Edefault-3.control&depth_1-utm_source=distribute.pc_relevant.none-task-blog-2%7Edefault%7EBlogCommendFromMachineLearnPai2%7Edefault-3.control)有什么用🔍
     1. 用一个新函数封装旧函数（是旧函数代码不变的情况下增加功能）然后会返回一个新函数，新函数就叫做装饰器。
     2. 装饰器可以动态的修改一个类或函数的功能，**本质上**就是一个返回函数的高阶函数

通过在原有的类或者函数上包裹一层修饰类或者修饰函数实现，不会直接去改变原有类或者函数的定义，这符合软件开发中的开闭原则（对拓展是开发的，对修改是封闭的）。

* + 1. 装饰器可以定义多个，**离函数定义最近的装饰器**最先被调用



* + 1. 使用装饰器的瑕疵：就是被修饰的函数，它的函数名称已经不是原来的名称了，因为最开始的名称指向了包装后的函数。
  1. [Python的迭代器、生成器了解有哪些？](https://www.zhihu.com/question/20829330)🔍
     1. 迭代器生成两个基本方法：**iter()** 和 **next()**。
        1. 实现了\_\_iter\_\_和\_\_next\_\_方法的对象都称为迭代器。迭代器是一个有状态的对象，在调用next() 的时候返回下一个值，如果容器中没有更多元素了，则抛出StopIteration异常。
     2. 生成器

①延迟计算，一次返回一个结果。它不会一次生成所有结果，对大数据量处理非常有用。②提高代码可读性。③但生成器只能遍历一次

* + - 1. **1.语法上和函数类似：**其他一样。def定义，区别在于生成器使用yield语句返回一个值；常规函数使用return返回一个值。
      2. **2.自动实现迭代器协议：**对于生成器，Python会自动实现迭代器协议，以便应用到迭代背景中（如for循环、sum函数）。由于生成器自动实现了迭代器协议，所以我们可以调用它的next方法，并且没有值返回的时候，生成器自动产生StopIteration异常
      3. **3.状态挂起：**生成器使用yield语句返回一个值。yield语句挂起该生成器函数的状态，保留足够的信息，以便之后从它离开的地方继续执行。
      4. 代码示例



* + - 1. [range 与 xrange 函数](https://www.cnblogs.com/molson/p/6613094.html)🔍
         1. range 函数说明：range([start,] stop[, step])，根据start与stop指定的范围以及step设定的步长，生成一个序列。
         2. xrange 函数：用法与range完全相同，所不同的是生成的不是一个数组，而是一个生成器。
         3. 区别：：要生成很大的数字序列的时候，用xrange会比range性能优很多，因为不需要一上来就开辟一块很大的内存空间，这两个基本上都是在循环的时候用：



* + 1. 除了节省内存空间外，也可以显著提升代码运行速度。
  1. \_\_new\_\_和\_\_init\_\_分别是干嘛的？

1. **数据结构与算法**
   1. 编程，字符串翻转
2. **测试基本理论**
   1. [黑盒和白盒](https://blog.csdn.net/ningmeng2010/article/details/8114530)🔍
      1. 黑盒的测试用例技术设计： 边界值分析、等价类划分、错误推测法。
      2. 白盒的测试用例技术：逻辑覆盖和基本路径测试。
   2. 测试用例的八大要素
      1. 用例编号、测试项目、测试标题、重要级别、预置条件、测试输入、测试步骤、预期结果。
   3. 测试用例：为测试设计的数据，由测试输入数据和与之对应的预期输出结构两部分组成。
   4. 软件测试步骤：
      1. 测试阶段：单元测试、集成测试、系统测试、验收测试
         1. 单元测试主要技术手段有：驱动代码、Stub代码、Mock代码。

GUI测试手段属于系统测试手段

* + 1. 其他测试：冒烟测试、α测试、β测试、回归测试
  1. 系统测试：
     1. **强度测试**：为了确定系统在**最差环境下**的工作能力，在非标准工作环境下，不断人为降低系统工作所需要的资源，以测试系统在**资源不足的情况**下的工作状态。
     2. **压力测试：高负荷下的负载测试**
     3. **负载测试**：**模拟实际软件系统所承受的系统负荷**，通过模拟增加用户量，观察响应时间，数据吞吐量，CPU占用，发现系统存在的性能瓶颈、内存泄漏、不能实时同步等问题。
     4. **容量测试**：是性能测试的一种，**测试系统的最大容量**，为系统扩容，为性能优化提供参考。
     5. **性能测试**—疲劳强度测试：通过**增加短时间的交易量**，而缩短测试时间来达到既定的测试目标，尽可能在短时间内完成规定的所有交易量。