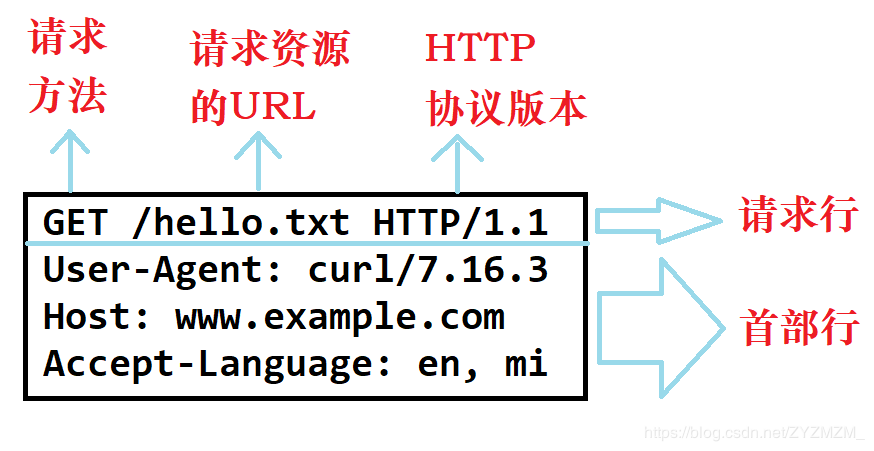
# 一，HTTP请求

http请求由三部分组成，分别是：请求行、消息报头、请求正文



1、起始行以一个方法符号开头，以空格分开，后面跟着请求的URI和协议的版本。

2、请求方法（所有方法全为大写）有多种，常见的为get和post。

 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

应用举例：   
GET方法：在浏览器的地址栏中输入网址的方式访问网页时，浏览器采用GET方法向服务器获取资源，eg:

* GET /form.html HTTP/1.1 (CRLF)

POST方法要求被请求服务器接受附在请求后面的数据，常用于提交表单。eg：

* POST /reg.jsp HTTP/ (CRLF)

Accept:image/gif,image/x-xbit,... (CRLF)

...

HOST:www.guet.edu.cn (CRLF)

Content-Length:22 (CRLF)

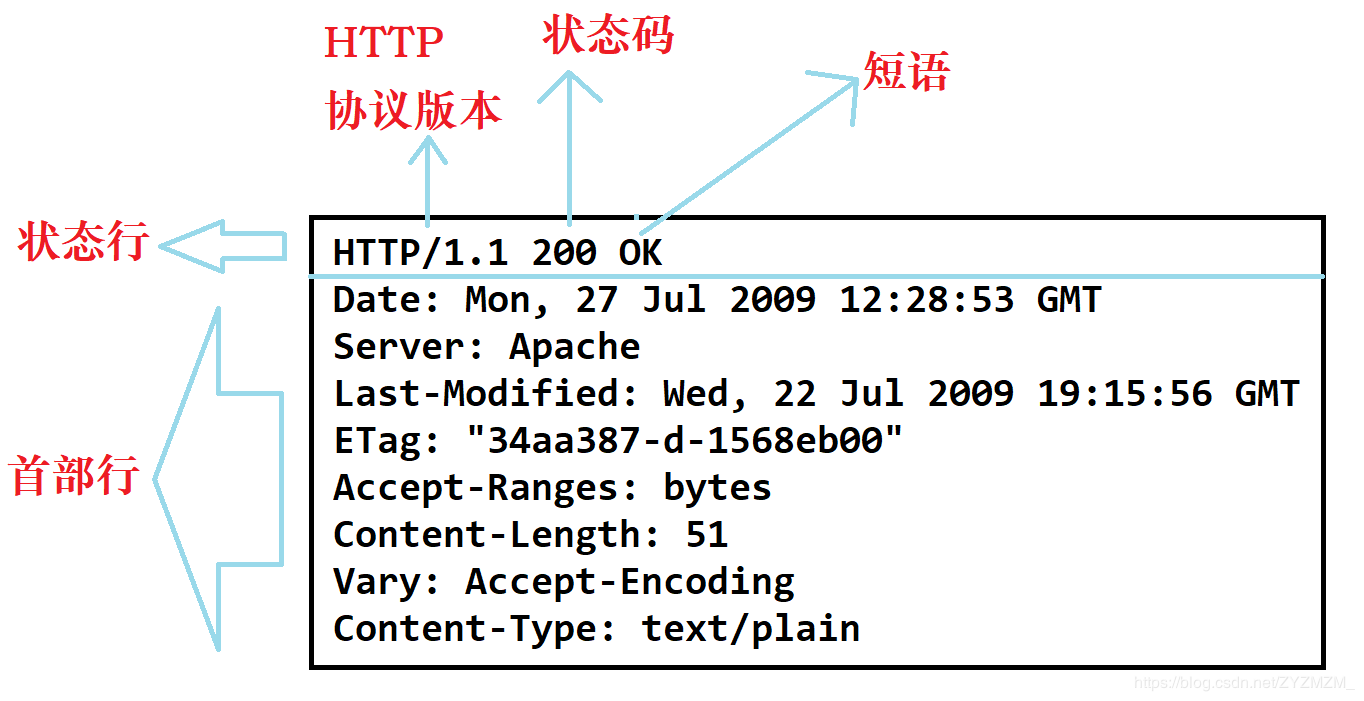
Connection:Keep-Alive (CRLF)

Cache-Control:no-cache (CRLF)

(CRLF)         //该CRLF表示消息报头已经结束，在此之前为消息报头

user=jeffrey&pwd=1234  //此行以下为提交的数据

# 二，HTTP响应



下面是常见的HTTP状态码：



1×× : 请求处理中，请求已被接受，正在处理

2×× : 请求成功，请求被成功处理

200 OK

3×× : 重定向，要完成请求必须进行进一步处理

301 : 永久性转移

302 ：暂时性转移

304 ： 已缓存

4×× : 客户端错误，请求不合法

400：Bad Request,请求有语法问题

403：拒绝请求

404：客户端所访问的页面不存在

5×× : 服务器端错误，服务器不能处理合法请求

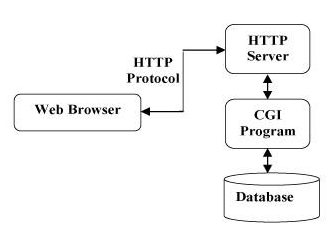
500 ：服务器内部错误

503 ： 服务不可用，稍等

# 三，CGI程序

https://www.jianshu.com/p/c4dc22699a42

CGI只是一个接口规范或协议，它的实现则与具体的编程语言相关。在2000年以前，CGI通用网关接口盛行，那个时候，Perl是编写CGI的主流语言，以至于一般的CGI程序都是Perl程序。

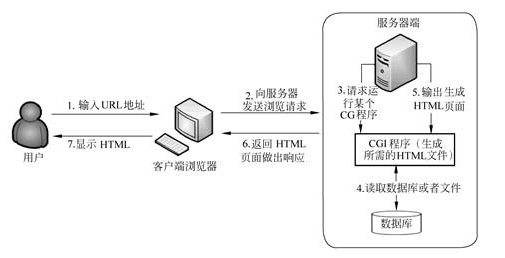


通过CGI接口，Web服务器就能够获取客户端传递的数据，并转交给服务器端的CGI程序处理，然后返回结果给客户端。简单来说，CGI实际上是一个接口标准。而通常所说的CGI指代其实是CGI程序，也就是实现了CGI接口标准的程序，只要编程语言具有标准输入、标准输出和环境变量，就可以用来编写CGI程序。

对于一个CGI程序，主要的工作是从环境变量和标准输入中读取数据，然后处理数据，最后向标准输出中输出数据。

* 环境变量  
  环境变量中存储的叫做Request Meta-Variables，也就是诸如QUERY\_STRING、PATH\_INFO之类的，这些都是由Web服务器通过环境变量传递给CGI程序的，CGI程序也是从环境变量中读取的。
* 标准输出  
  中存放的往往是用户通过PUTS或POST提交的数据，这些数据也是由Web服务器传递过来的。

为了处理动态请求，Web服务器会根据请求的内容，Fork创建一个新进程来运行外部C程序或Perl脚本等，这个进程会把处理完的数据返回给Web服务器，然后Web服务器把内容发送给用户，Fork创建出来的进程也会随之退出。如果下次用户请求为动态脚本，那么Web服务器会再次Fork创建一个新进程，如此周而复始的运行。



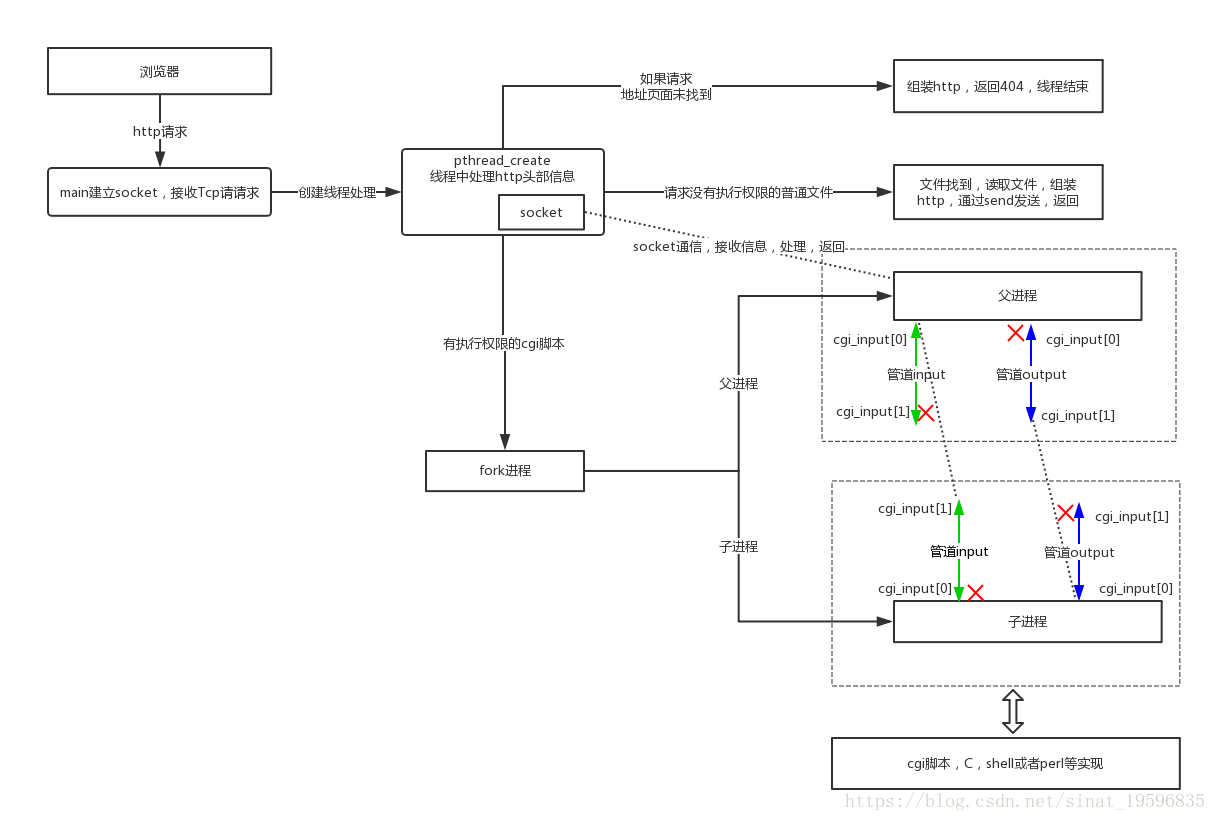
# 四，tinyhttpd源码分析

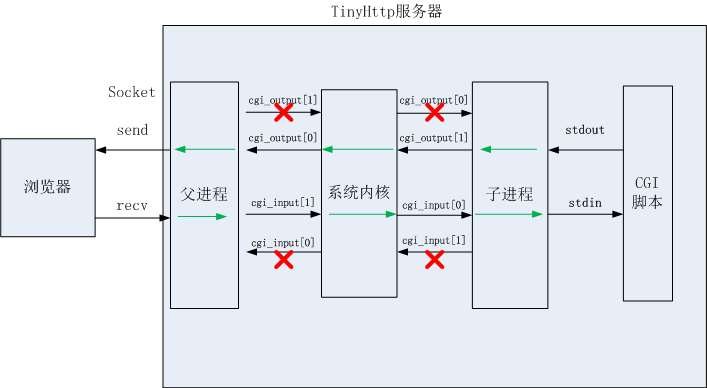
tinyhttpd总共包含以下函数：

* void accept\_request(int);//处理从套接字上监听到的一个 HTTP 请求
* void bad\_request(int);//返回给客户端这是个错误请求，400响应码
* void cat(int, FILE \*);//读取服务器上某个文件写到 socket 套接字
* void cannot\_execute(int);//处理发生在执行 cgi 程序时出现的错误
* void error\_die(const char \*);//把错误信息写到 perror
* void execute\_cgi(int, const char \*, const char \*, const char \*);//运行cgi脚本，这个非常重要，涉及动态解析
* int get\_line(int, char \*, int);//读取一行HTTP报文
* void headers(int, const char \*);//返回HTTP响应头
* void not\_found(int);//返回找不到请求文件
* void serve\_file(int, const char \*);//调用 cat 把服务器文件内容返回给浏览器。
* int startup(u\_short \*);//开启http服务，包括绑定端口，监听，开启线程处理链接
* void unimplemented(int);//返回给浏览器表明收到的 HTTP 请求所用的 method 不被支持。

源码主要执行顺序： **main -> startup -> accept\_request -> execute\_cgi**

按照以上顺序，看一下浏览器和tinyhttpd交互的整个流程：





# 五，执行结果

这个项目并不能直接在Linux上编译运行。它本来是在solaris上实现的，貌似在socket和pthread的实现上和一般的Linux还是不一样的，需要修改一部分内容。

可以参考：<https://www.cnblogs.com/qiyeboy/p/6296387.html>

编译执行后的结果：

