



main 🡪 startup 🡪 accept\_request 🡪 execute\_cgi

startup初始化httpd服务，包括建立套接字，绑定端口，进行监听

accept\_request处理httpd请求

execute\_cgi运行cgi程序的处理

主进程监听新的套接字连接

每有一个新的已连接套接字就创建一个新的线程处理。

如果客户端是动态请求，创建子进程处理这个动态请求，并用管道进行父子进程之间的通信

### main()

1、定义server套接字描述符，调用**startup()**返回

2、定义client套接字已连接描述符，调用accept()返回

3、每次返回一个已连接套接字，调用pthread\_create()创建一个新的线程。然后再newthread线程中调用accept\_request处理连接请求。创建线程时，将clientsock（已连接描述符）传入新线程。

### startup

1、定义套接字描述符，并让bind指定IP地址和port，并通过getsockname获取绑定地址

2、listen并返回server套接字描述符。

有出错情况就调用error\_die

### accept\_request

1、调用get\_line()读取请求的第一行

2、解析出方法，如果不是post和get，调用unimplemented通知客户端

3、如果是post，设置cgi=1，执行动态服务。

4、解析出URL，从url中解析出地址。如果是GET方法并且url中有？，表明是动态服务，设置cgi=1

5、server目录为htdocs，在url前加上目录名。

6、如果是没有指定服务，则指定默认服务index.html

7、用stat获取文件信息，如果没有找到该文件信息，则将请求头部读取出来并丢弃。之后用not\_found向服务端发送错误信息

8、检查文件类型，如果是目录，默认执行index.html

9、检查文件权限，如果有执行权限，则cgi = 1;

10，检查cgi，不是动态服务，调用serve\_file()直接返回。如果是动态服务，调用execute\_cig执行cgi

11、关闭已连接套接字。

### serve\_file

1、读取并丢弃所有的头部信息

2、打开文件

3、如果文件不存在，调用not\_found

4、否则分别调用headers和cat返回响应头部和响应正文

### execute\_cgi

1、如果method是get，则读取请求头部并丢弃

2、如果method是post，则从请求头部解析出content-length，如果没有，调用bad\_request

3、返回状态行

4、创建两个管道用于父子进程之间的双向通信，无法创建则调用cannot\_execute

5、创建子进程，在子进程中执行cgi

6、从管道中读出cgi的输出内容。并返回给client。等待子进程退出后退出

重定向输入输出之后的图解见第一页

### 辅助函数

#### bad\_request

向客户端返回400错误，语法错误

#### cat

用于静态请求

读取resource文件，每次用fgets读取一行并返回

#### cannot\_execute

当无法创建管道时调用。通知客户端cgi程序无法执行

返回500错误

#### error\_die

调用perror打印错误信息，并退出

#### getline

每次从socket中读取一行。

使用recv每次从socket读取一个字符。遇到换行符\r\n即为一行。

读取的最后一行可能没有\n，则主动加上\n

#### headers

返回响应头部

#### not\_found

当请求的文件不存在时返回404错误

#### unimplemented

当前只实现了post和get，当出现其他请求方法时，返回501错误，方法没有实现