웹서버

URL이 HTTP로 시작하는 이유를 마침내 이해하게 된 때입니다.

요약: 이 프로
젝트는 자신의 HTTP 서버를 작성하는 것에 관한 것입니다.
실제 브라우저로 테스트해 볼 수 있습니다.
HTTP는 인터넷에서 가장 많이 사용되는 프로토콜 중 하나입니다.
웹사이트에서 작업하지 않더라도 그 비밀을 아는 것은 유용할 것입니다.

버전: 21

내용물

제1장

소개

HTTP(Hypertext Transfer Protocol)는 분산, 협업, 하이퍼미디어 정보 시스템을 위한 애플리케이션 프로토콜입니다.

HTTP는 하이퍼텍스트 문서에 사용자가 쉽게 액세스할 수 있는 다른 리소스에 대한 하이퍼링크가 포함되어 있는 World Wide Web용 데이터 통신의 기초입니다.
예를 들어 마우스를 클릭하거나 웹 브라우저에서 화면을 탭하면 됩니다.

HTTP는 하이퍼텍스트와 World Wide Web을 용이하게 하기 위해 개발되었습니다.

웹 서버의 주요 기능은 웹 페이지를 저장, 처리 및 클라이언트에 전달하는 것입니다. 클라이언트와 서버 간의 통신은 HTTP(Hypertext Transfer Protocol)를 사용하여 이루어집니다.

전달되는 페이지는 가장 자주 HTML 문서로, 여기에는 이미지,
텍스트 콘텐츠 외에 스타일 시트 및 스크립트도 포함됩니다.

트래픽이 많은 웹사이트에는 여러 웹 서버를 사용할 수 있습니다.

일반적으로 웹 브라우저 또는 웹 크롤러인 사용자 에이전트는 HTTP를 사용하여 특정 리소스를 요청하여 통신을 시작하고 서버는 해당 리소스의 콘텐츠로 응답하거나 그렇게 할 수 없는 경우 오류 메시지로 응답합니다. 리소스는 일반적으로 서버의 보조 저장소에 있는 실제 파일이지만 반드시 그런 것은 아니며 웹 서버가 구현되는 방식에 따라 다릅니다.

주요 기능은 콘텐츠를 제공하는 것이지만 HTTP의 전체 구현에는 클라이언트로부터 콘텐츠를 수신하는 방법도 포함됩니다. 이 기능은 파일 업로드를 포함하여 웹 양식을 제출하는 데 사용됩니다.

제2장

일반 규칙

- 어떤 상황에서도 프로그램이 중단되어서는 안 됩니다.
  메모리), 예기치 않게 종료되어서는 안 됩니다.
  이러한 일이 발생하면 귀하의 프로젝트는 작동하지 않는 것으로 간주되며 귀하의 성적은 0이 됩니다.

- 소스 파일을 컴파일할 Makefile을 제출해야 합니다. 그러면 안 된다
  다시 연결합니다.

- Makefile에는 최소한 다음 규칙이 포함되어야 합니다.
  $(NAME), 모두, 깨끗하고, fclean 및 re.

- C++ 및 -Wall -Wextra -Werror 플래그를 사용하여 코드를 컴파일합니다.

- 코드는 C++ 98 표준을 준수해야 합니다 . 그런 다음 여전히 컴파일되어야 합니다.
  -std=c++98 플래그를 추가하면.

- 항상 최대한 많은 C++ 기능을 사용하여 개발하십시오(예: <string.h> 대신 <cstring> 선택). C 함수를 사용할 수 있지만
  가능하면 항상 C++ 버전을 선호합니다.

- 모든 외부 라이브러리 및 Boost 라이브러리는 금지됩니다.

제3장

필수부분

프로그램 이름	웹서버
파일 제출	메이크파일, *.{h, hpp}, *.cpp, *.tpp, *.ipp, 구성 파일
메이크파일	이름, 모두, 깨끗한, fclean, 다시
인수 외부 함수.	[구성 파일]
	C++ 98의 모든 것. execve, dup, dup2, 파이프, strerror, gai_strerror, errno, dup, dup2, 포크, 소켓 쌍, htons, htonl, ntohs, ntohl, 선택, 설문 조사, epoll(epoll_create, epoll_ctl, epoll_wait), kqueue(kqueue, kevent), 소켓, 수락, 듣기, 보내기, recv, chdir 바인딩, 연결, getaddrinfo, freeaddrinfo, setockopt, getsockname, getprotobyname, fcntl, 닫기, 읽기, 쓰기, waitpid, kill, 신호, 액세스, stat, opendir, readdir 및 closedir.
Lift 승인됨	해당사항 없음
설명	C++ 98의 HTTP 서버

C++ 98로 HTTP 서버를 작성해야 합니다.

실행 파일은 다음과 같이 실행됩니다.

./webserv [구성 파일]

주제와 평가척도에 poll()이 언급되더라도,
select(), kqueue() 또는 epoll()과 같은 동등한 것을 사용할 수 있습니다.

RFC를 읽고 텔넷 및 NGINX를 사용하여 몇 가지 테스트를 수행한 후 수행하십시오.
이 프로젝트를 시작합니다.
RFC를 모두 구현하지 않아도 읽어보면 필요한 기능을 개발하는 데 도움이 됩니다.

III.1 요구사항

- 프로그램은 구성 파일을 인수로 사용하거나 기본 경로를 사용해야 합니다.
- 다른 웹 서버를 실행할 수 없습니다.
- 서버는 절대 차단해서는 안 되며 필요한 경우 클라이언트가 적절하게 반송될 수 있습니다.
- 비차단이어야 하며 클라이언트와 서버 사이의 모든 I/O 작업(수신 포함)에 대해 1개의 poll()(또는 이와 동등한 것)만 사용해야 합니다.
- poll()(또는 이에 상응하는 것)은 읽기와 쓰기를 동시에 확인해야 합니다.
- poll()을 거치지 않고 읽기나 쓰기 작업을 수행해서는 안 됩니다.
  동등한).
- 읽기 또는 쓰기 작업 후에 errno 값을 확인하는 것은 엄격히 금지됩니다.
- 구성 파일을 읽기 전에 poll()(또는 이와 동등한 기능)을 사용할 필요가 없습니다.

비차단 파일 설명자를 사용해야 하기 때문에 poll()(또는 이와 동등한 기능) 없이 읽기/recv 또는 쓰기/전송 기능을 사용할 수 있으며 서버가 차단되지 않습니다.

그러나 더 많은 시스템 리소스를 소비하게 됩니다.
따라서 poll()(또는 이에 상응하는 것)을 사용하지 않고 파일 설명자에서 읽기/수신 또는 쓰기/보내기를 시도하면 성적은 0이 됩니다.

- 모든 매크로를 사용하고 FD_SET, FD_CLR, FD_ISSET, FD_ZERO와 같이 정의할 수 있습니다(그들이 수행하는 작업과 방법을 이해하는 것은 매우 유용합니다).
- 서버에 대한 요청이 영원히 중단되어서는 안 됩니다.
- 귀하의 서버는 귀하가 선택한 웹 브라우저 와 호환되어야 합니다 .
- NGINX는 HTTP 1.1과 호환되며 이를 비교하는 데 사용될 수 있다고 생각합니다.
  헤더 및 답변 동작.
- HTTP 응답 상태 코드는 정확해야 합니다.
- 아무것도 제공되지 않은 경우 서버에 기본 오류 페이지가 있어야 합니다.
- CGI 이외의 것(PHP, Python 등)에는 포크를 사용할 수 없습니다.
- 완전히 정적인 웹사이트를 제공 할 수 있어야 합니다 .
- 클라이언트는 파일을 업로드할 수 있어야 합니다 .
- 최소한 GET, POST 및 DELETE 메소드가 필요합니다.
- 서버를 스트레스 테스트합니다. 어떤 희생을 치르더라도 계속 사용할 수 있어야 합니다.
- 서버는 여러 포트를 수신할 수 있어야 합니다(구성 파일 참조).

III.2 MacOS에만 해당

MacOS는 다른 Unix OS와 동일한 방식으로 write()를 구현하지 않으므로 fcntl()을 사용할 수 있습니다.
파일 디스크립터를 얻으려면 비차단 모드에서 파일 설명자를 사용해야 합니다.
다른 Unix OS 중 하나와 유사한 동작.

그러나 fcntl()은 다음과 같은 방법으로만 사용할 수 있습니다: fcntl(fd, F_SETFL, O_NONBLOCK, FD_CLOEXEC);
다른 플래그는 금지됩니다.

III.3 구성 파일

NGINX 구성 파일의 '서버' 부분에서 영감을 얻을 수 있습니다.

구성 파일에서 다음을 수행할 수 있어야 합니다.

- 각 '서버'의 포트와 호스트를 선택합니다.
- server_names 설정 여부.
- 호스트:포트의 첫 번째 서버는 이 호스트:포트의 기본값이 됩니다. 즉, 다른 서버에 속하지 않는 모든 요청에 응답한다는 의미입니다.
- 기본 오류 페이지를 설정합니다.
- 클라이언트 본체 크기를 제한합니다.
- 다음 규칙/구성 중 하나 이상을 사용하여 경로를 설정합니다(경로는 그렇지 않음).
  정규 표현식을 사용하세요):
  - 경로에 대해 허용되는 HTTP 메서드 목록을 정의합니다.
  - HTTP 리디렉션을 정의합니다.
  - 파일을 검색해야 하는 디렉터리 또는 파일을 정의합니다. 예를 들어 url /kapouet이 /tmp/www에 루트된 경우 url /kapouet/pouic/toto/pouet는 /tmp/www/pouic/toto/pouet입니다.).
  - 디렉토리 목록을 켜거나 끕니다.

◦ 요청이 디렉터리인 경우 응답할 기본 파일을 설정합니다.

◦ 특정 파일 확장자(예: .php)를 기반으로 CGI를 실행합니다.

◦ POST 및 GET 방법으로 작동하도록 만듭니다.

◦ 경로가 업로드된 파일을 수락할 수 있도록 만들고 파일이 업로드되어야 하는 위치를 구성합니다.
저장됩니다.

＊ CGI가 무엇인지 궁금하시죠? 이다? ＊ CGI

를 직접 호출하지 않으므로 전체 경로를 PATH_INFO로 사용하세요.

＊ 청크된 요청의 경우 서버가 요청을 청크 해제해야 하며 CGI는 EOF를 본문의 끝으로 예상한다는 점을 기억하세요. ¡CGI의 출력도 마찬가지입니다. content_length가 반

환되지 않은 경우
CGI에서 EOF는 반환된 데이터의 끝을 표시합니다.

¡ 귀하의 프로그램은 요청된 파일을 첫 번째 인수로 사용하여 CGI를 호출해야 합니다.

＊ 상대 경로 파일 접근을 위해서는 CGI가 올바른 디렉터리에서 실행되어야 합니다. ¡ 귀하의 서버는 하나의

CGI(php-CGI, Python 등)와 작동해야 합니다.

평가 중에 모든 기능이 작동하는지 테스트하고 시연하려면 일부 구성 파일과 기본 기본 파일을 제공해야 합니다.

한 가지 동작에 대한 질문이 있는 경우 프로그램 동작을 NGINX의 동작과 비교해야 합니다.

예를 들어, server_name이 어떻게 작동하는지 확인하세요.
우리는 작은 테스터를 여러분과 공유했습니다. 브라우저와 테스트에서 모든 것이 제대로 작동한다면 통과하는 것이 필수는 아니지만 일부 버그를 찾는 데 도움이 될 수 있습니다.

중요한 것은 회복탄력성이다. 서버는 절대 죽어서는 안 됩니다.

하나의 프로그램만으로 테스트하지 마십시오. Python이나 Golang 등과 같은 보다 편리한 언어로 테스트를 작성하세요. 원한다면 C나 C++에서도 가능합니다.

제4장

보너스 부분

추가할 수 있는 추가 기능은 다음과 같습니다.

- 쿠키 및 세션 관리를 지원합니다(빠른 예시 준비).
- 여러 CGI를 처리합니다.

보너스 부분은 필수 부분이 PERFECT인 경우에만 평가됩니다. 완벽하다는 것은 필수 부분이 완벽하게 수행되어 오작동 없이 작동한다는 것을 의미합니다. 모든 필수 요구 사항을 통과하지 못한 경우 보너스 부분은 전혀 평가되지 않습니다.

제5장

제출 및 동료 평가

평소처럼 Git 저장소에 과제를 제출하세요. 방어 중에는 저장소 내부 작업만 평가됩니다. 주저하지 말고 파일 이름이 올바른지 다시 확인하세요.

16D85ACC441674FBA2DF65190663F42A3832CEA21E024516795E1223BBA77916734D1
26120A16827E1B16612137E59ECD492E46EAB67D109B142D49054A7C281404901890F
619D682524F5