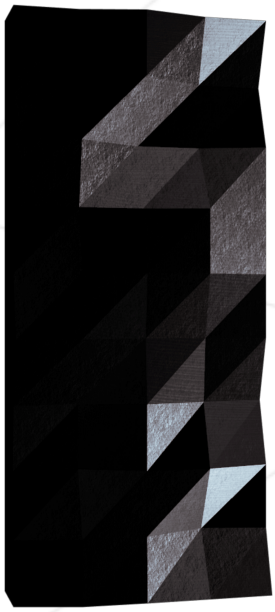

Servidor web

Aquí es cuando finalmente entiendes por qué comienza una URL con HTTP

Resumen:

Este proyecto trata sobre cómo escribir su propio servidor HTTP.
Podrás probarlo con un navegador real. HTTP es
uno de los protocolos más utilizados en Internet.
Conocer su misterio será útil, incluso si no vas a trabajar en un sitio web.

Versión: 21.2

Contenido

I Introducción **2**

II Reglas generales **3**

III Parte obligatoria **4**

III.1 Requisitos 6

III.2 Solo para MacOS . . . . . Archivo de 7

III.3 configuración 7

IV Parte extra **9**

V Presentación y evaluación por pares **10**

Capítulo I

Introducción

El**Protocolo de transferencia de hipertexto**(HTTP) es un protocolo de aplicación para sistemas de información hipermedia distribuidos y colaborativos.

HTTP es la base de la comunicación de datos en la World Wide Web, donde los documentos de hipertexto incluyen hipervínculos a otros recursos a los que el usuario puede acceder fácilmente, por ejemplo, con un clic del ratón o tocando la pantalla de un navegador web.

HTTP fue desarrollado para facilitar el hipertexto y la World Wide Web.

La función principal de un servidor web es almacenar, procesar y entregar páginas web a los clientes. La comunicación entre el cliente y el servidor se lleva a cabo mediante el Protocolo de transferencia de hipertexto (HTTP).

Las páginas entregadas suelen ser documentos HTML, que pueden incluir imágenes, hojas de estilo y scripts, además del contenido de texto.

Se pueden utilizar varios servidores web para un sitio web con mucho tráfico.

Un agente de usuario, normalmente un navegador web o un rastreador web, inicia la comunicación solicitando un recurso específico mediante HTTP y el servidor responde con el contenido de ese recurso o con un mensaje de error si no puede hacerlo. El recurso suele ser un archivo real en el almacenamiento secundario del servidor, pero no es necesariamente así y depende de cómo esté implementado el servidor web.

Si bien la función principal es entregar contenido, la implementación completa de HTTP también incluye formas de recibir contenido de los clientes. Esta función se utiliza para enviar formularios web, incluida la carga de archivos.

Capítulo II

Reglas generales

- Su programa no debería bloquearse bajo ninguna circunstancia (incluso cuando se quede sin memoria) y no debería cerrarse inesperadamente.
  Si esto sucede, su proyecto se considerará no funcional y su calificación será 0.

- Tienes que entregar unArchivo Makeque compilará sus archivos fuente. No debe volver a vincularse.

- SuArchivo MakeDebe contener al menos las reglas:
  $(NOMBRE), todo, limpio, fcleanyre.
- Compila tu código conC++y las banderas -Muro -Wextra -Werror
- Su código debe cumplir con las**Estándar C++ 98**. Entonces, aún debería compilarse si agrega la bandera -estándar=c++98.
- Intenta desarrollarte siempre utilizando lo máximoC++funciones que puede (por ejemplo, elegir <cadena>Más de <cadena.h>).Se le permite utilizardofunciones, pero siempre prefieren susC++versiones si es posible.
- Cualquier biblioteca externa yAumentarLas bibliotecas están prohibidas.

Capítulo III

Parte obligatoria

Nombre del programa	servidor web
Entregar archivos	Makefile, *.{h, hpp}, *.cpp, *.tpp, *.ipp, archivos de configuración
Archivo Make	NOMBRE, todo, limpio, limpio, re
Argumentos	[Un archivo de configuración]
Funciones externas.	Todo en C++ 98. execve, dup, dup2, pipe, strerror, gai_strerror, errno, dup, dup2, fork, socketpair, htons, htonl, ntohs, ntohl, select, poll, epoll (epoll_create, epoll_ctl, epoll_wait), kqueue (kqueue, kevent), socket, accept, listen, send, recv, chdir bind, connect, getaddrinfo, freeaddrinfo, setsockopt, getsockname, getprotobyname, fcntl, close, read, write, waitpid, kill, signal, access, stat, open, opendir, readdir y closedir.
Libft autorizado	n / A
Descripción	Un servidor HTTP en C++ 98

Debes escribir un servidor HTTP en C++ 98.

Su ejecutable se ejecutará de la siguiente manera:

. /webserv [archivo de configuración]

Incluso si se menciona poll() en el asunto y en la escala de evaluación, puedes usar cualquier equivalente como select(), kqueue() o epoll().

Lea el RFC y realice algunas pruebas con telnet y NGINX antes de comenzar este proyecto.
Incluso si no tiene que implementar todo el RFC, leerlo le ayudará a desarrollar las características necesarias.

III.1 Requisitos

- Su programa debe tomar un archivo de configuración como argumento o utilizar una ruta predeterminada.
- No puedesejecutivoOtro servidor web.
- Su servidor nunca debe bloquearse y el cliente puede rebotar adecuadamente si es necesario.
- Debe ser antibloqueante y usarse únicamente**1**encuesta() (o equivalente) para todas las operaciones de E/S entre el cliente y el servidor (escucha incluida).
- encuesta() (o equivalente) debe verificar la lectura y escritura al mismo tiempo.
- Nunca debes realizar una operación de lectura o escritura sin pasar porencuesta() (o equivalente).
- Comprobando el valor deerrorEstá estrictamente prohibido después de una operación de lectura o escritura.
- No es necesario utilizarencuesta() (o equivalente) antes de leer el archivo de configuración.

Debido a que debe utilizar descriptores de archivos no bloqueantes, es posible utilizar funciones de lectura/recepción o escritura/envío sin poll() (o equivalente), y su servidor no se bloquearía.
Pero consumiría más recursos del sistema.
Por lo tanto, si intenta leer/recibir o escribir/enviar cualquier descriptor de archivo sin utilizar poll() (o equivalente), su calificación será 0.

- Puedes usar cada macro y definir comoFD_SET, FD_CLR, FD_ISSET, FD_ZERO (Entender qué y cómo lo hacen es muy útil).
- Una solicitud a su servidor nunca debería quedarse bloqueada para siempre.
- Su servidor debe ser compatible con el**Navegador web**de tu elección.
- Consideraremos que NGINX es compatible con HTTP 1.1 y puede usarse para comparar encabezados y comportamientos de respuesta.
- Los códigos de estado de respuesta HTTP deben ser precisos.
- Su servidor debe tener**páginas de error predeterminadas**si no se proporciona ninguno.
- No puedes usar fork para nada más que CGI (como PHP o Python, etc.).
- Debes ser capaz de**Servir un sitio web completamente estático**.
- Los clientes deben poder**subir archivos**.
- Necesitas al menosOBTENER, PUBLICAR,yBORRARmétodos.
- Pon a prueba tu servidor. Debe permanecer disponible a toda costa.
- Su servidor debe poder escuchar múltiples puertos (consulte*Archivo de configuración*).

III.2 Solo para MacOS

Dado que MacOS no implementa write() de la misma manera que otros sistemas operativos Unix, se le permite usar fcntl().
Debes utilizar descriptores de archivos en modo sin bloqueo para obtener un comportamiento similar al de otros sistemas operativos Unix.

Sin embargo, solo se le permite utilizar fcntl() con los siguientes indicadores:
F_SETFL, O_NONBLOCK y FD_CLOEXEC.
Cualquier otra bandera está prohibida.

III.3 Archivo de configuración

Puedes inspirarte en la parte "servidor" del archivo de configuración NGINX.

En el archivo de configuración deberías poder:

- Elija el puerto y el host de cada 'servidor'.
- Configurar elnombres_de_servidorO no.
- El primer servidor para unhost:puertoSerá el valor predeterminado para esto.host:puerto (Esto significa que responderá a todas las solicitudes que no pertenezcan a otro servidor).
- Configurar páginas de error predeterminadas.
- Limite el tamaño del cuerpo del cliente.
- Configure rutas con una o varias de las siguientes reglas/configuraciones (las rutas no usarán expresiones regulares):
  - Define una lista de métodos HTTP aceptados para la ruta.
  - Definir una redirección HTTP.
  - Define un directorio o un archivo desde donde se debe buscar el archivo (por ejemplo, si url /capuchinotiene sus raíces en /tmp/www,URL /capo/pouic/toto/pouet es /tmp/www/pouic/toto/pouet).
  - Activar o desactivar el listado de directorios.

- Establezca un archivo predeterminado para responder si la solicitud es un directorio.
- Ejecutar CGI según cierta extensión de archivo (por ejemplo .php).
- Hazlo funcionar con los métodos POST y GET.
- Haga que la ruta pueda aceptar archivos cargados y configure dónde deben guardarse.
  - ¿Te preguntas qué es unCGI¿es?
  - Debido a que no llamará al CGI directamente, use la ruta completa comoINFORMACIÓN DE RUTA.
  - Recuerde que, para las solicitudes fragmentadas, su servidor debe desfragmentarlas. eso, el CGI esperaráfin de añocomo fin del cuerpo.
  - Lo mismo ocurre con la salida del CGI. Si no haylongitud del contenidose devuelve desde el CGI,fin de añomarcará el final de los datos devueltos.
  - Su programa debe llamar al CGI con el archivo solicitado como primer argumento. ∗ El CGI debe ejecutarse en el directorio correcto para acceder al archivo por ruta relativa. ∗ Su servidor debe funcionar con un CGI (php-CGI, Python, etc.).

Debe proporcionar algunos archivos de configuración y archivos básicos predeterminados para probar y demostrar que cada característica funciona durante la evaluación.

Si tiene alguna pregunta sobre un comportamiento, debe comparar el comportamiento de su programa con el de NGINX.
Por ejemplo, compruebe cómo funciona server_name.
Hemos compartido contigo una pequeña prueba. No es obligatorio pasarla si todo funciona bien con tu navegador y las pruebas, pero puede ayudarte a detectar algunos errores.

Lo importante es la resiliencia. Tu servidor nunca debería morir.

No realice pruebas con un solo programa. Escriba sus pruebas con un lenguaje más conveniente, como Python o Golang, etc. Incluso en C o C++ si lo deseas.

Capítulo IV

Parte extra

Estas son las características adicionales que puedes agregar:

- Admite cookies y gestión de sesiones (preparar ejemplos rápidos).
- Manejar múltiples CGI.

La parte adicional solo se evaluará si la parte obligatoria es PERFECTA. Perfecto significa que la parte obligatoria se ha realizado íntegramente y funciona sin fallas. Si no ha aprobado TODOS los requisitos obligatorios, su parte adicional no se evaluará en absoluto.

Capítulo V

Presentación y evaluación por pares

Entregue su tarea en suGitRepositorio como de costumbre. Solo el trabajo dentro de su repositorio será evaluado durante la defensa. No dude en volver a verificar los nombres de sus archivos para asegurarse de que sean correctos.

16D85ACC441674FBA2DF65190663F42A3832CEA21E024516795E1223BBA77916734D1
26120A16827E1B16612137E59ECD492E46EAB67D109B142D49054A7C2814 04901890F
619D682524F5