Webserv

This is when you finally understand why a URL starts with HTTP

Summary:
This project is about writing your own HTTP server.
You will be able to test it with an actual browser.
HTTP is one of the most used protocols on the internet.
Knowing its arcane will be useful, even if you won't be working on a website.

Version: 21.2

Contents

Chapter I

Introduction

The **Hypertext Transfer Protocol** (HTTP) is an application protocol for distributed, collaborative, hypermedia information systems.

HTTP is the foundation of data communication for the World Wide Web, where hypertext documents include hyperlinks to other resources that the user can easily access. For example, by a mouse click or by tapping the screen in a web browser.

HTTP was developed to facilitate hypertext and the World Wide Web.

The primary function of a web server is to store, process, and deliver web pages to clients. The communication between client and server takes place using the Hypertext Transfer Protocol (HTTP).

Pages delivered are most frequently HTML documents, which may include images, style sheets, and scripts in addition to the text content.

Multiple web servers may be used for a high-traffic website.

A user agent, commonly a web browser or web crawler, initiates communication by requesting a specific resource using HTTP and the server responds with the content of that resource or an error message if unable to do so. The resource is typically a real file on the server's secondary storage, but this is not necessarily the case and depends on how the webserver is implemented.

While the primary function is to serve content, full implementation of HTTP also includes ways of receiving content from clients. This feature is used for submitting web forms, including the uploading of files.

Chapter II

General rules

- Your program should not crash in any circumstances (even when it runs out of memory), and should not quit unexpectedly.
  If it happens, your project will be considered non-functional and your grade will be `0`.
- You have to turn in a `Makefile` which will compile your source files. It must not relink.
- Your `Makefile` must at least contain the rules:
  `$(NAME)`, `all`, `clean`, `fclean` and `re`.
- Compile your code with `c++` and the flags `-Wall -Wextra -Werror`
- Your code must comply with the **C++ 98 standard**. Then, it should still compile if you add the flag `-std=c++98`.
- Try to always develop using the most `C++` features you can (for example, choose `<cstring>` over `<string.h>`). You are allowed to use `C` functions, but always prefer their `C++` versions if possible.
- Any external library and `Boost` libraries are forbidden.

Chapter III

Mandatory part

Program name	webserv
Turn in files	Makefile, *.{h, hpp}, *.cpp, *.tpp, *.ipp, configuration files
Makefile	NAME, all, clean, fclean, re
Arguments	[A configuration file]
External functs.	Everything in 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 and closedir.
Libft authorized	n/a
Description	A HTTP server in C++ 98

You must write a HTTP server in C++ 98.

Your executable will be run as follows:

```
./webserv [configuration file]
```

Even if poll() is mentionned in the subject and the evaluation scale, you can use any equivalent such as select(), kqueue(), or epoll().