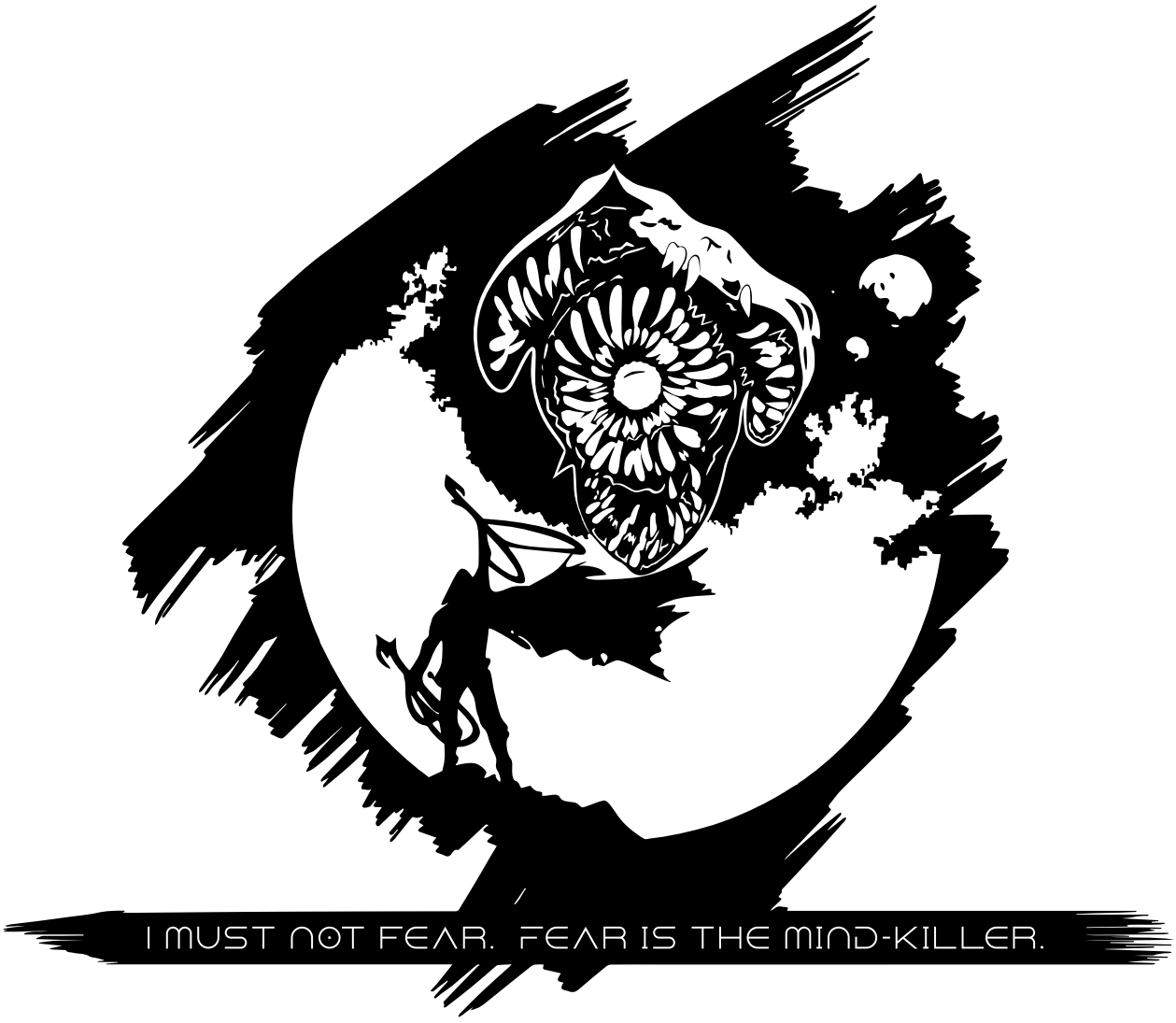
Exercises — HTTPd configuration

version **#dirty**



Assistants C/Unix 2024 <assistants@tickets.assistants.epita.fr>

Copyright

This document is for internal use at EPITA (website) only.

Contents

*https://intra.forge.epita.fr

File Tree

```
httpd/
└── src/
    ├── config/
    │   ├── config.c  (to submit)
    │   └── config.h  (to submit)
    └── utils/
        └── string/
            ├── string.c  (to submit)
            └── string.h  (to submit)
```

Main function : None

Compilation : Your code must compile with the following flags

- -std=c99 -Werror -Wall -Wextra -Wvla

Authorized headers : You are only allowed to use the functions defined in the following headers

- assert.h
- ctype.h
- stddef.h
- err.h
- errno.h
- stdio.h
- string.h
- stdlib.h
- fnmatch.h

1 Configuration

Since the purpose of this project is not to implement parsing algorithms, you need an easy and efficient way to provide parameters to your server. The first step in building your HTTP Server is to be able to read the configuration file.

The format is very simple as it consists of tag sections containing pairs of keys and values.

Here is an example of a configuration file:

```
[global]
log_file = server.log
log = true
pid_file = /tmp/HTTPd.pid

[[vhosts]]
server_name = images
port = 1312
ip = 127.0.0.1
root_dir = votai/test.
```

Tips

The first line of the file will always be the tag `[global]`.

Here are the keys to the *global* tag section:

Name	Description	Mandatory
pid_file	**Absolute** path to the file containing the PID of the daemon.	Yes
log_file	**Relative** path to the file where your logs must be written.	No
log	Whether or not you should write logs.	No

If no value is provided, `log` should be considered true. The default value for *log_file* should not be handled yet, this means **you can** be asked to log even if no *log_file* is specified. The behavior in this case will be explained in the HTTPd subject.

Tips

A tag section will always end with an empty line.

Be careful!

Make sure you understand the difference between relative and absolute paths.

The second tag you will have to parse is the tag `[[vhosts]]`.

Tips

The configuration **must** contain at least one `[[vhosts]]` tag section. For now, consider a single `[[vhosts]]` tag section to simplify your parser.

It will contain the following keys:

Name	Description	Mandatory
server_name	Hostname of the server.	Yes
port	Port of the server.	Yes
ip	IP of the VHost.	Yes
root_dir	Location on the host machine where the server can serve files.	Yes
default_file	Default file to serve when requesting a directory.	No

The configuration file will always be valid. The only error case you will encounter is a missing mandatory key. If that is the case your parsing function will return NULL.

Also, note that values cannot contain spaces, you will never encounter the following: “key = value1 value2 value3”

A complete configuration example can be found in the annex section.

Tips

You will understand the purpose of each key while reading the HTTPd subject later on.

Tips

You could take a look at the `strchr(3)` function (and more generally the `string.h` header), it should prove quite useful and should save you some time.

2 String

HTTPd being a networking project, regular C null-terminated strings are not very convenient. This is why we provide a minimalistic string interface that should make it easier to manipulate strings.

The particularity of this module is that strings are defined by a buffer of characters and a size. This means that a string is no longer defined by a simple array of characters that end with a null byte. This is important because it means that you can have null bytes in your strings.

This is particularly interesting when you want to send binary data that might contain null bytes over the network.

Tips

An empty *string* is defined by a *NULL* buffer and a size of 0.

Tips

We require you to at least implement the functions mentioned in the header files. You are very strongly encouraged to add your functions.

Now that you can read the parameters passed to your program, the real work can begin!

3 Annex

3.1 Complete Configuration Example

```
[global]
log_file = server.log
log = true
pid_file = /tmp/HTTPd.pid

[[vhosts]]
server_name = images
port = 4243
```

(continues on next page)

(continued from previous page)

```
ip = 127.0.0.1
default_file = club_des_frais.html
root_dir = tests/fake_images

[[vhosts]]
server_name = files
port = 6666
ip = 127.0.0.3
root_dir = tests/fake_files
```

I must not fear. Fear is the mind-killer.