B3 - C++ Pool

B-CPP-300

Rush 3

MyGKrellm

2.0

EPITECH.

Rush 3

binary name: MyGKrellm
language: C++
compilation: via Makefile, including re, clean and fclean rules

- The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.

The `*alloc`, `free`, `*printf`, `open` and `fopen` functions, as well as the `using namespace` keyword, are forbidden in C++.

Let's be clear: this rush does not aim to make you completely re-implement GKrellM.

Or does it?

However, you are expected to create a clone of it.
The subject is purposefully the least restrictive of the three rushes, so that you can feel free to enjoy yourselves and add whatever you want to your program.
However, **dat ain't no party**, and a few mandatory steps must be validated first.

This rush consists in three core steps and a bonus step.
Each step must be done entirely and perfectly before the next is started, as they are increasingly difficult.

Do not start the second step as long as the first isn't perfectly functional.

Your system monitor can be seen as a module container.
It is possible to activate and deactivate the available modules so that users can adapt the information to their needs, or even reorder them however they want.

GKrellM doesn't need privileged rights to run correctly, and neither should your monitor.

Once properly configured and with a sexy skin, **GKrellM** is a very nice system monitor. But before we get to that point, there's work to be done!

It must be possible to start your system monitor either in “text” or in “graphical” mode, with the same functionalities.

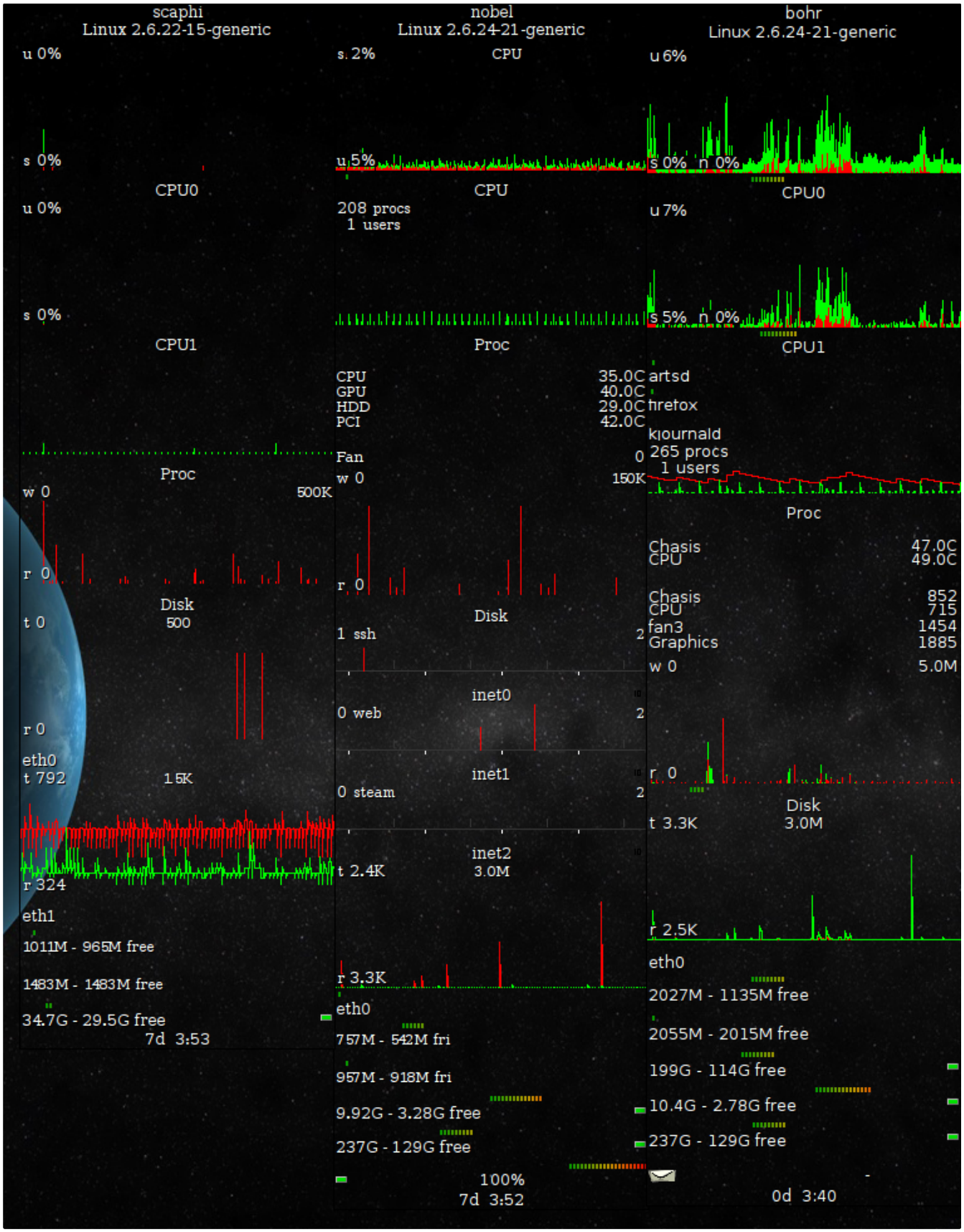
Keep this constraint in mind from the first line of code you write.

In “text” mode, your monitor will be displayed on your terminal.
You **MUST** use one for the following libraries: **nCurses**, **aa-lib** or **libcaca**.

In “graphical” mode, your monitor must be displayed in a graphical window.
You **MUST** use one of the following librairies: **SDL**, **SFML** or **Qt5**.

Whatever mode it is started in, the visual quality and ergonomy of your monitor will have a non-negligible impact on your grade.
For instance, some data are better represented by a numeric value or words, while a histogram or curve is more suited for others.
Be imaginative.

GKrellM has an awesome skin system. What about yours?


scaphi
Linux 2.6.22-15-generic
u 0%
s 0%
CPU0
u 0%
s 0%
CPU1
Proc
w 0
500K
r 0
Disk
t 0
500
r 0
eth0
t 792
15K
r 324
eth1
1011M - 965M free
1483M - 1483M free
34.7G - 29.5G free
7d 3:53
nobel
Linux 2.6.24-21-generic
s 2%
CPU
u 5%
CPU
208 procs
1 users
Proc
CPU 35.0C
GPU 40.0C
HDD 29.0C
PCI 42.0C
Fan 0
w 0
150K
r 0
Disk
1 ssh
inet0
0 web
inet1
0 steam
inet2
t 2.4K
3.0M
r 3.3K
eth0
757M - 542M fri
957M - 918M fri
9.92G - 3.28G free
237G - 129G free
100%
7d 3:52
bohr
Linux 2.6.24-21-generic
u 6%
s 0% n 0%
CPU0
u 7%
s 5% n 0%
CPU1
artsd
firefox
kjournald
265 procs
1 users
Proc
Chasis 47.0C
CPU 49.0C
Chasis 852
CPU 715
fan3 1454
Graphics 1885
w 0 5.0M
r 0
Disk
t 3.3K
3.0M
r 2.5K
eth0
2027M - 1135M free
2055M - 2015M free
199G - 114G free
10.4G - 2.78G free
237G - 129G free
0d 3:40

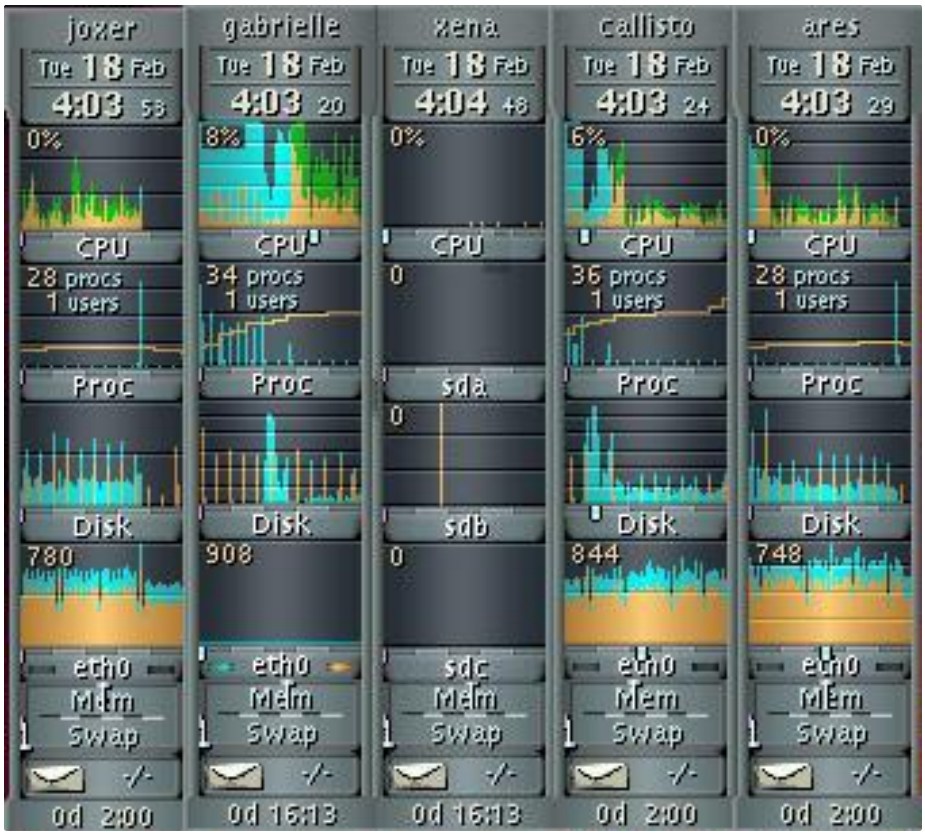
The design of your monitor is entirely up to you.
However, you must use at least the two following interfaces:

- `IMonitorModule`: describes the behavior of a module of your monitor
- `IMonitorDisplay`: describes a display mode of your monitor

The content of these interfaces is up to you.
You may want to take some time to wonder why we force you to use them.

- Level -1: modules are configurable at compilation time
- Level 0: modules are configurable at launch time
- Level 1: modules are configurable and can be added or removed at runtime

Be brave.
Reach level 1.



Step 1

- Monitor core
- Hostname and username module
- Operating system name and kernel version module
- Date and time module

Step 2

- CPU module (model, frequency, number of cores, activity…)
- RAM module

Step 3

- Network load module

Bonus

A pool rush is a bag filled with points for those who play by the rules to the end…
In this last step, create as many **useful** modules as possible.
Be creative and smart.

Don't forget that this step will not be taken into account if the mandatory part is not perfectly functional.