Links

- https://hub.xpub.nl/sandbox/ (XPUB 4 's server)
- https://hub.xpub.nl/sandbot/ (XPUB 3 's server)
- https://hub.xpub.nl/soupboat/ (XPUB 2 's server)
- https://pzwiki.wdka.nl/mediadesign/Sandbox (PZI wiki page about the idea of the Sandbox)
- https://www.bleu 2 5 5 .com/~aymeric/dump/aymeric_mansoux-sandbox_culture_phd_thesis- 2 0 1 7.pdf (Aymeric's PhD Thesis "Sandbox Culture, A Study of the Application of Free and Open Source – Software Licensing Ideas to Art and Cultural Production"
- https://vvvvvvaria.org/curriculum/In-the-Beginning-...-Was-the-Commandline/READER.html # how-to-work-with-text-commands (In the Beginning ... Was the Commandline, reader publishing by Varia (2 0 1 8))
- https://pzwiki.wdka.nl/mediadesign/Shell_Cheat_Sheet (shell cheat sheet)
- https://solarpunk.cool/zines/map-is-the-territory/ (The Map is the Territory, shell introduction zine by Zach Mandeville (https://coolguy.website/) and Angelica Blevins (https://angblev.com/) from https://solarpunk.cool/; sources of the zine: https://git.sr.ht/~zim/map-is-the-territory)

```
In [ ]:
```

Install ourselves into XPUB, sandboxes, servers, web servers, Hello World!

Sandbox

From: https://pzwiki.wdka.nl/mediadesign/Sandbox

***Living in a Sandbox* is an optional course that aims at exploring the culture of free and open source UNIX-like software and computer hardware from the viewpoint of a small device: the Raspberry Pi. During this course, students will be exposed to historical and technical elements of computing that are nowadays buried under an app centric culture grown in the names of user-friendliness, transparency and deceptive allegories such as the cloud.**

New technologies, like smart phones and web services, promise cutting edge technologies and software as a means to empower users with a seamingly endless progression of new digital possibilities. In fact, many of these new services are striking for the

append

`>>` appends the output of a command to a file, without overwriting the original file.

`echo 'also add this' >> df_output.txt` will add 'also add this' to the contents of df_output.txt

package managers

Package managers like `apt-get` and `aptitude` (on Debian/ Ubuntu Linux distributions) and Homebrew and MacPorts on Mac, allow more (command-line, but not only) programs, than the ones that come with the operating, to be installed on our system.

```
sudo apt search [app name]
sudo apt install [app name]
sudo apt remove [app name]
```

man pages

man pages are manuals of program. They tells you what the program is, what it can do and how.

`man df` show the manual for the program **df** that is used to display the free disk space

Can you find out how to display the output from df in a human readable format?

pipe

A pipes ("|") sends the output of one program to the input of another program.

`echo "my sentence"| wc` the echoed sentence "my sentence" is *pipped* into the program wc which counts the number of lines, words, and characters

write

`>` Writes the output of a command to a file, rather than to print on terminal.

`df > df_output.txt` redirect the content of `man dfM` to a file called df_output.txt

If the said file doesn't exit it will create it, if it already exists it will overwrite its contents/

many constraints they place (where can this be played, how many \"friends\" can connect, who decides a remix means and if it can be \"shared\").

Many of the platforms are themselves built on decades old technologies & software. Sandbox aims to deconstruct the digital black boxes, revealing the hidden (historical) layers of software and system, with the aim of: (1) empowering students through literacy of reading these systems, and (2) encouraging new assemblages to be (strategically) reconstructed.

When people hear the word sandbox, it is very likely that most of them will be thinking of the outdoor playset that consists of a container filled with sand. You probably have seen many already and have possibly played in one as a child. Using sand as medium and a couple of tools, the sandbox opens the door to a world where anything can be pretended and experimented with. For some others though, the sandbox is linked instead to the realm of software. Indeed, and similarly to its analogue counterpart, software sandboxes are used both to provide testing and prototyping environments, as well as to describe how users and processes can be isolated for security purposes. These two approaches play an important role in the development and execution of software. As a matter of fact, whether you are browsing a website, using an app, or working with your favourite digital tool, sandboxes have been and are currently used to enable and allow this action.

Truth is, digital sandboxes are everywhere and it is a bit ... problematic. Indeed, stepping out of an analogue sandbox is as easy as dusting off from your clothes the particules left from the imaginary world. The same cannot be said of the digital sandboxes which bits are tightly interleaved with our daily activities and digital diet. Seeing our increasing dependence on software and network infrastructure and in a post-PRISM age, it is becoming urgent to understand how these sandboxes operate and impact production, communication and more generally social dynamics.

The best way to explore these issues is to run your own sandbox! *Living in a Sandbox* aims to be a platform for:

- Critically (re)defining terms like Sharing, Network, Public/Private
- Understanding the history of networked computation, and an ability to trace to contemporary practices and to make strategic decisions in creating new work

Stopping & Starting

shutdown -h now -- Shutdown the system now and do not reboot
halt -- Stop all processes - same as above
shutdown -r `5` -- Shutdown the system in `5` minutes and reboot
shutdown -r now -- Shutdown the system now and reboot
reboot -- Stop all processes and then reboot - same as above
startx -- Start the X system

meta characters

Meta Characters are characters that have special meaning within the terminal

- `~` the tilde stands for the user's home. `cd ~/` change directory to home
- `.` dot stands for **this** directory. `ls .` list this directory
- `..` dot dot stands for **the parent directory** to this directory. `cp myfile.jpg ..` copy myfile.jpg to the parent directory
- `*` asterisk is a wildcards which represents zero or more characters `ls P*.jpg` will list all the files, in the current directory, that begin with P and end with .jpg
- `\` backslash it is a literal character. It escape the meta value of the meta-characters and display them only as literal characters. `echo Foo \*` will output `Foo *` If \ wasn't there it would output all the files in that directory.

wget *file* -- Download ***file***

wget -c ***file*** -- Continue a stopped download

SSH

ssh *user@host* -- Connect to ***host*** as ***user***

ssh -p *port user@host* -- Connect to ***host*** on port ***port*** as ***user***

ssh-copy-id *user@host* -- Add your key to ***host*** for ***user*** to enable a keyed or passwordless login

User Administration

adduser ***accountname*** -- Create a new user call ***accountname***

passwd *accountname* -- Give ***accountname*** a new password

su -- Log in as superuser from current login

exit -- Stop being superuser and revert to normal user

Process Management

ps -- Display your currently active processes

top -- Display all running processes

kill *pid* -- Kill process id ***pid***

killall *proc* -- Kill all processes named ***proc*** (use with extreme caution)

bg -- Lists stopped or background jobs; resume a stopped job in the background

fg -- Brings the most recent job to foreground

fg ***n*** -- Brings job ***n*** to the foreground

Command Line Interface (CLI)

Ghost in the Shell

From: https://vvvvvvaria.org/curriculum/In-the-Beginning-...-Was-the-Commandline/READER.html # how-to-work-with-text-commands

> Go ahead and start using the command line by opening a `terminal` application. You'll see a text interface with a blinking cursor. What happened when you opened the terminal is that it actually opened a so-called `shell` for you. The shell (`sh`) is a software which takes your keyboard input and gives it to the computer's operating system. There are various types of shells but the most common ones are `bash` (bourne again shell) or `zsh` .

Essential commands

From: https://pzwiki.wdka.nl/mediadesign/Shell_Cheat_Sheet

From: https://community.linuxmint.com/tutorial/view/ 2 4 4

System Info

date -- Show the current date and time

cal -- Show this month's calendar

uptime -- Show current uptime
w -- Display who is online
whoami -- Who you are logged in as
finger ***user*** -- Display information about ***user***
uname -a -- Show kernel information
cat /proc/cpuinfo -- CPU information
cat /proc/meminfo -- Memory information
df -h -- Show disk usage
du -- Show directory space usage
free -- Show memory and swap usage

Keyboard Shortcuts

Enter -- Run the command
Up Arrow -- Show the previous command
Ctrl + R -- Allows you to type a part of the command you're looking for and finds it

Ctrl + Z -- Stops the current command, resume with **fg** in the foreground or **bg** in the background
Ctrl + C -- Halts the current command, cancel the current operation and/or start with a fresh new line
Ctrl + L -- Clear the screen

command | **less** -- Allows the scrolling of the bash command window using **Shift + Up Arrow** and **Shift + Down Arrow**
!! -- Repeats the last command
!\$ -- Repeats the last argument of the previous command
Esc + . (**a period**) -- Insert the last argument of the previous

mv ***file*** 1 ***file*** 2 -- Rename or move ***file*** 1 to ***file*** 2 ; if ***file*** 2 is an existing directory, moves ***file*** 1 into directory ***file*** 2

ln -s ***file link*** -- Create symbolic link ***link*** to ***file***
touch ***file*** -- Create or update ***file***
cat > ***file*** -- Places standard input into ***file***
cat ***file*** -- Display the file called ***file***

more ***file*** -- Display the file called ***file*** one page at a time, proceed to next page using the spacebar
head ***file*** -- Output the first 10 lines of ***file***
head - 20 ***file*** -- Display the first 20 lines of the file called ***file***
tail ***file*** -- Output the last 10 lines of ***file***
tail - 20 ***file*** -- Display the last 20 lines of the file called ***file***
tail -f ***file*** -- Output the contents of ***file*** as it grows, starting with the last 10 lines

Network

ifconfig -- List IP addresses for all devices on the local machine
iwconfig -- Used to set the parameters of the network interface which are specific to the wireless operation (for example: the frequency)
iwlist -- used to display some additional information from a wireless network interface that is not displayed by **iwconfig**
ping ***host*** -- Ping ***host*** and output results
whois ***domain*** -- Get whois information for ***domain***
dig ***domain*** -- Get DNS information for ***domain***
dig -x ***host*** -- Reverse lookup ***host***

Check the current ownership of a file with: **ls -l**
Check which groups you are in with: **groups**

File Commands

ls -- Directory listing
ls -l -- List files in current directory using long format
ls -laC -- List all files in current directory in long format and display in columns
ls -F -- List files in current directory and indicate the file type
ls -al -- Formatted listing with hidden files

cd ***dir*** -- Change directory to ***dir***
cd -- Change to home
mkdir ***dir*** -- Create a directory ***dir***
pwd -- Show current directory

rm ***name*** -- Remove a file or directory called ***name***
rm ***-r dir*** -- Delete directory ***dir***
rm -f ***file*** -- Force remove ***file***
rm -rf ***dir*** -- Force remove an entire directory ***dir*** and all it's included files and subdirectories (use with extreme caution)

cp ***file*** 1 ***file*** 2 -- Copy ***file*** 1 to ***file*** 2
cp -r ***dir*** 1 ***dir*** 2 -- Copy ***dir*** 1 to ***dir*** 2 ; create ***dir*** 2 if it doesn't exist
cp ***file*** /**home**/***dirname*** -- Copy the filename called ***file*** to the /**home**/**dirname** directory

mv ***file*** /**home**/***dirname*** -- Move the ***file*** called filename to the /***home***/***dirname*** directory

command on the fly, which enables you to edit it before executing the command

Ctrl + A -- Return to the start of the command you're typing
Ctrl + E -- Go to the end of the command you're typing
Ctrl + U -- Cut everything before the cursor to a special clipboard, erases the whole line
Ctrl + K -- Cut everything after the cursor to a special clipboard
Ctrl + Y -- Paste from the special clipboard that **Ctrl + U** and **Ctrl + K** save their data to
Ctrl + T -- Swap the two characters before the cursor (you can actually use this to transport a character from the left to the right, try it!)
Ctrl + W -- Delete the word / argument left of the cursor in the current line **Ctrl + D** -- Log out of current session, similar to **exit**

Learn the Commands

apropos ***subject*** -- List manual pages for ***subject***
man -k ***keyword*** -- Display man pages containing ***keyword***
man ***command*** -- Show the manual for ***command***
man -t ***man*** | **ps** 2 **pdf - >** ***man.pdf*** -- Make a pdf of a manual page
which ***command*** -- Show full path name of ***command***
time ***command*** -- See how long a ***command*** takes

whereis ***app*** -- Show possible locations of ***app***
which ***app*** -- Show which ***app*** will be run by default; it shows the full path

Searching

grep *pattern* *files* -- Search for ***pattern*** in ***files***

grep -r *pattern* *dir* -- Search recursively for ***pattern*** in ***dir***

command | **grep *pattern*** -- Search for ***pattern*** in the output of ***command***

locate *file* -- Find all instances of ***file***

find / **-name *filename*** -- Starting with the root directory, look for the file called ***filename***

find / **-name "**\filename\" **-- Starting with the root directory, look for the file containing the string** *filename*

locate *filename** * -- Find a file called ***filename*** using the locate command; this assumes you have already used the command

updatedb (see next)

updatedb -- Create or update the database of files on all file systems attached to the Linux root directory

which *filename* -- Show the subdirectory containing the executable file called ***filename***

grep *TextStringToFind* /***dir*** -- Starting with the directory called ***dir***, look for and list all files containing ***TextStringToFind***

File Permissions

chmod *octal* *file* -- Change the permissions of ***file*** to **"octal**", which can be found separately for user, group, and world by adding:

`4` -- read (r), `2` -- write (w), `1` -- execute (x)

Examples:

chmod `7 7 7` **filename** -- read, write, execute for all

chmod `7 5 5` **filename** -- rwx for owner, rx for group and world

chmod *symbolic* *file* -- You can also change permissions in **symbolic** mode**.**

Examples:

chmod ugo+x filename -- to make a file executable

chmod g+w filename -- to grant write access to the group

chmod o-r filename -- to remove read access to others

```
u: user
g: group
o: others
```

```
r: read
w: write
x: executable
```

```
-R: recursively
```

For more options, see **man chmod.**

File Ownership

chown -- change ownership

chown *name_of_new_owner* "filename"

chown newuser:newgroup filename -- To change ownership of a file to **newuser** and the group **newgroup**

chown root:www-data /**var**/**www**/**html**/ -- To change ownership of a file to **root** and the group **www-data**