

INUX

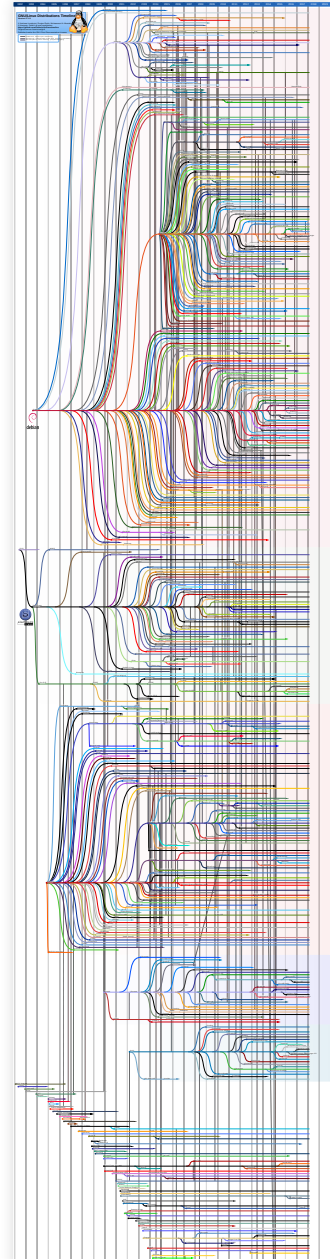
ener: Tomasz Dołbniak
dańsk, 8 lipca 2018

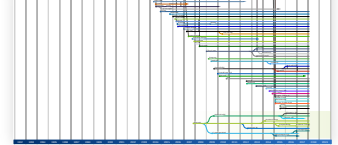
LINUX JAKO SYSTEM OPERACYJNY

- **Jądro linuxa** - kernel - najważniejszy element linuxa; zarządza procesorem, pamięcią, pozostałym sprzętem, procesami, usługami, siecią, ...
- **Linux** - rodzina systemów operacyjnych zbudowanych w oparciu o jądro linuxa
- **Dystrybucja Linuxa** = jądro + zestaw aplikacji + narzędzie do zarządzania paczkami
- Zwyczajowo słowem "Linux" określa się dowolną dystrybucję (**distro**)

DYSTRYBUCJE LINUKSA

https://en.wikipedia.org/wiki/List_of_Linux_distributions





NAJWAŻNIEJSZE(?) DYSTRYBUCJE

- Debian
- **Ubuntu**
- RHEL
- CentOS
- Fedora
- openSUSE
- Slackware
- Arch
- Mint
- Kali Linux
- Alpine Linux
- Android
- DD-WRT / OpenWRT

ZASTOSOWANIA

- Komputery desktopowe / laptopy
- Serwery
- TOP500 - 500 największych superkomputerów na świecie ⁽²⁰¹⁷⁾
- Telefony/tablety z Androidem
- Routery
- Raspbery PI
- Urządzenia IOT
- Systemy embedded - w tym branża "security"
- Branża automotive
- Urządzenia medyczne
- Przemysł - maszyny i ich sterowanie
- Telewizory, smartwatche i inne "wearables"
- W chmurze (Azure, AWS)

STRONY O LINUXSIE

- www.distrowatch.com - lista dystrybucji, na b aktualizowana tabela popularności, informac świata Linuksa, informacje o aktualizacjach p tutoriale, artykuły, ...
- wikipedia.org/wiki/Comparison_of_Linux_dist - tabelaryczne porównanie dystrybucji
- www.linuxquestions.org - baza pytań i odpow podziałem na dystrybucje

BUDOWA LINUXA

- **Kernel** - baza Linuksa
- **Bootloader** - odpowiada za uruchomienie OS; często ma kilka stopni ładujących kolejne aplikacje wymagane do działania systemu
- **Demony** - usługi działające w tle, zwykle uruchamiane automatycznie po starcie systemu
- **Powłoki** - interfejsy do obsługi systemu (tekstowe lub graficzne)
- **X** - X Window system - system do zarządzania aplikacjami graficznymi
- **Desktop environment** - graficzna powłoka zawierająca manager okien (GNOME, KDE, Unity, Cinnamon, Mate, LXDE, XFCE)

LINUX VS OPEN SOURCE

- Większość dystrybucji jest utrzymywana przez społeczność
- W wersjach społecznościowych mają swój udział duże firmy (np. HP)
- Istnieją również płatne dystrybucje (**RHEL, SUSE**)
- Mimo to dostępne są bazujące na nich wersje darmowe (**Fedora, openSUSE**)
- Płatne dystrybucje oferują pomoc/wsparcie oraz szkolenia i certyfikaty

POSIX

- Portable Operating System Interface
- Grupa standardów mająca na celu maksymalizację kompatybilności pomiędzy systemami operacyjnymi
- Pozwala to na pisanie przenośnych aplikacji i uruchamianie ich na wielu systemach
- Standardy obejmują m. in.: zarządzanie procesami oraz IPC, bibliotekę języka C, sposób komunikacji z jądrem systemu
- Linux jest bardzo związany z POSIXem i dąży do pełnej kompatybilności
- Istnieją implementacje POSIXa dla Windowsa (**Cygwin**, **MinGW**) - można pisać programy, które da się skompilować i uruchomić pod Linuxem i Windowsem

STRUKTURA KATALOGÓW W LINUXSIE

- Struktura drzewiasta jak w większości systemów
- Jest to jednak jedno duże drzewo ze wspólnym "korzeniem" (**root**)
- Katalog root skrótowo oznacza się symbolem **/**
- Podobny model działa w systemach Mac OSX (również bazują na UNIX)
- Windows ma osobne drzewa, po jednym na każdą partycję

DYSKI W WINDOWS

The screenshot displays the Windows Computer Management console, specifically the Disk Management section. The top table lists the volumes on the system, and the bottom section shows the graphical representation of the disks and their partitions. A diagram with red lines illustrates the mapping of logical hard drives to physical disks and their partitions.

Volume	Layout	Type	File System	Status	Capacity	Free Space	% Free	Fault Tolerance	Overhead
(C:)	Simple	Basic	NTFS	Healthy (Primary Partition)	120.72 GB	120.72 GB	100 %	No	0%
(H:)	Simple	Basic	FAT	Healthy (Primary Partition)	244 MB	244 MB	100 %	No	0%
System Reserved	Simple	Basic	NTFS	Healthy (System, Active, Primary Partition)	100 MB	32 MB	32 %	No	0%

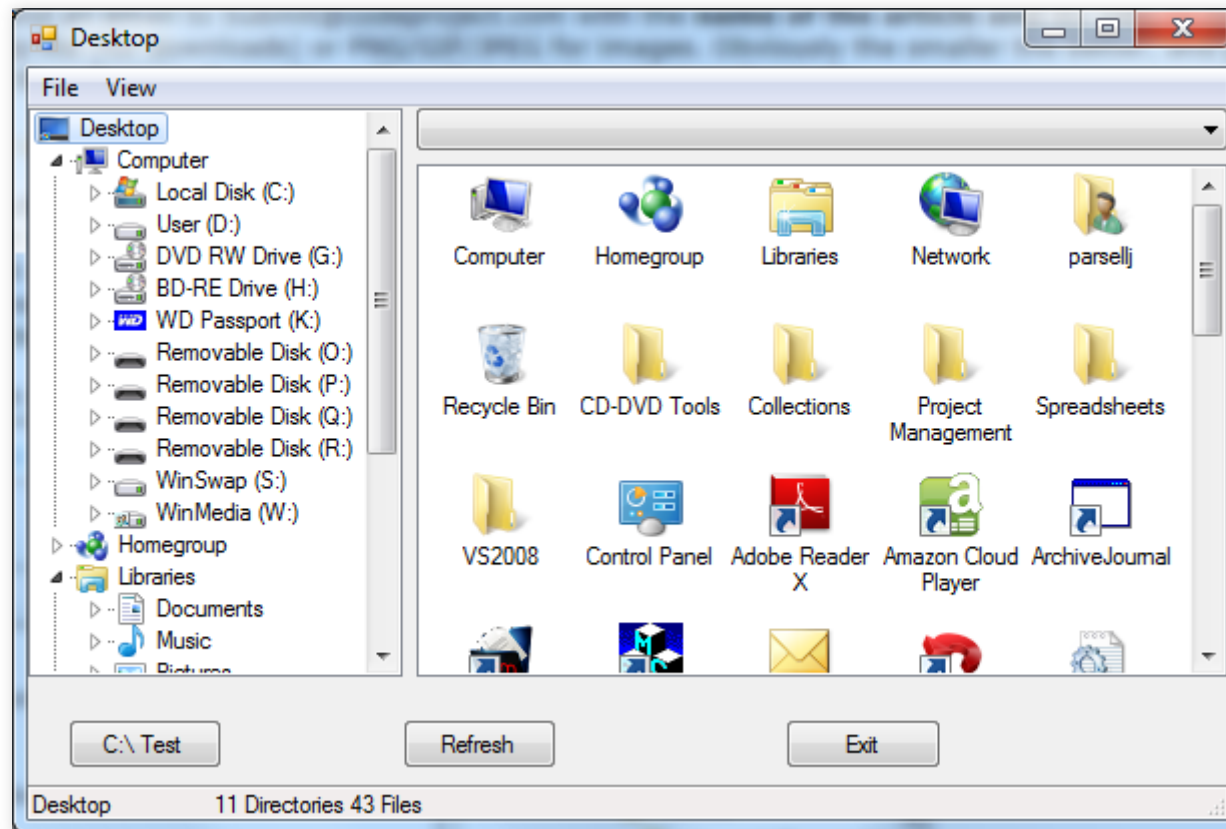
Disk	Type	Capacity	Status	Partitions
Disk 0	Basic	232.88 GB	Online	232.88 GB Unallocated
Disk 1	Basic	465.76 GB	Online	System Reserved (100 MB NTFS), (C:) (342.95 GB NTFS), 120.72 GB (Primary Partition), 2.00 GB (Primary Partition)
Disk 2	Removable	244 MB	Online	(H:) (244 MB FAT)

Diagram Labels:

- Physical Hard Drives:** Points to Disk 0, Disk 1, and Disk 2.
- Logical Hard Drives:** Points to the 232.88 GB Unallocated space on Disk 0, the (C:) partition on Disk 1, the 120.72 GB partition on Disk 1, and the 2.00 GB partition on Disk 1.

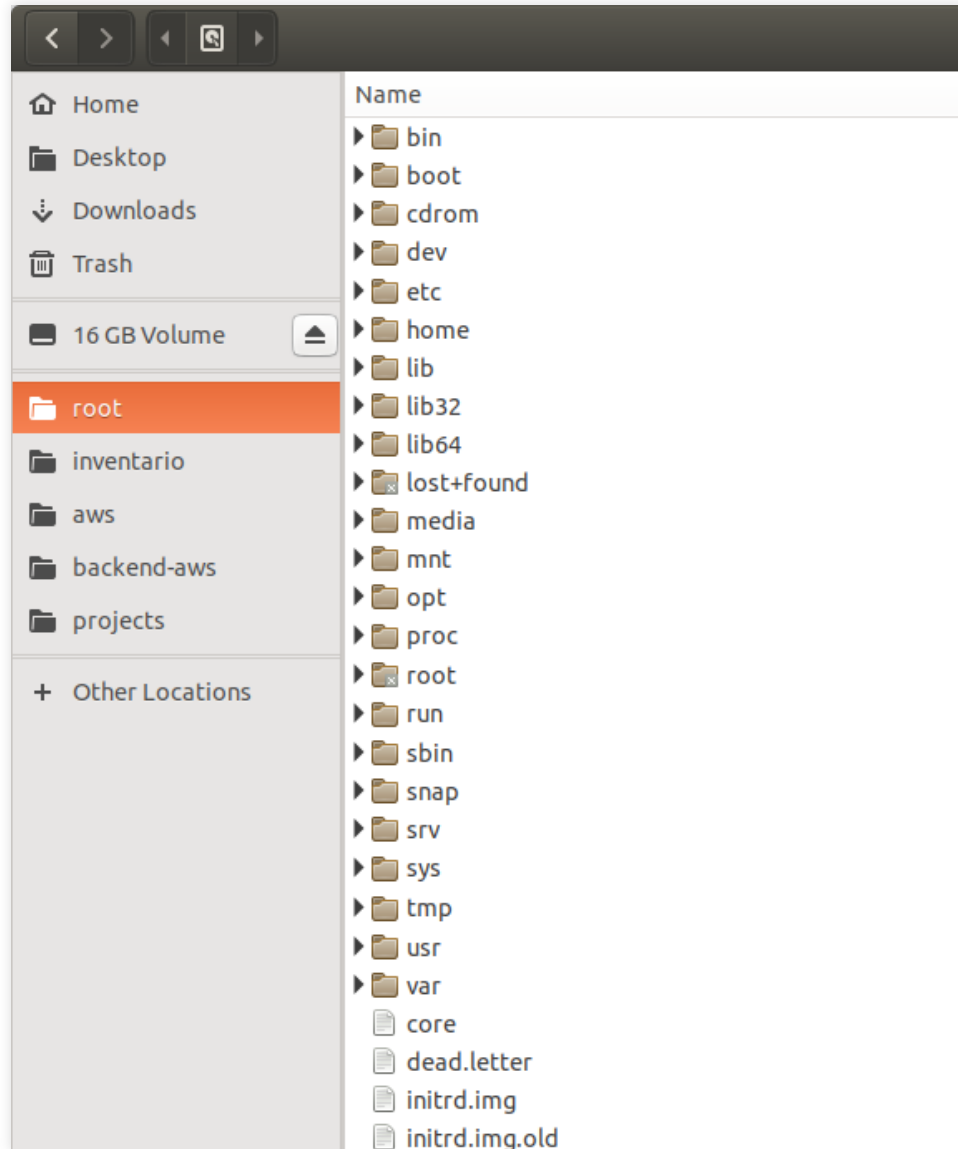
Legend: ■ Unallocated ■ Primary partition ■ Extended partition ■ Free space

DYSKI W WINDOWS

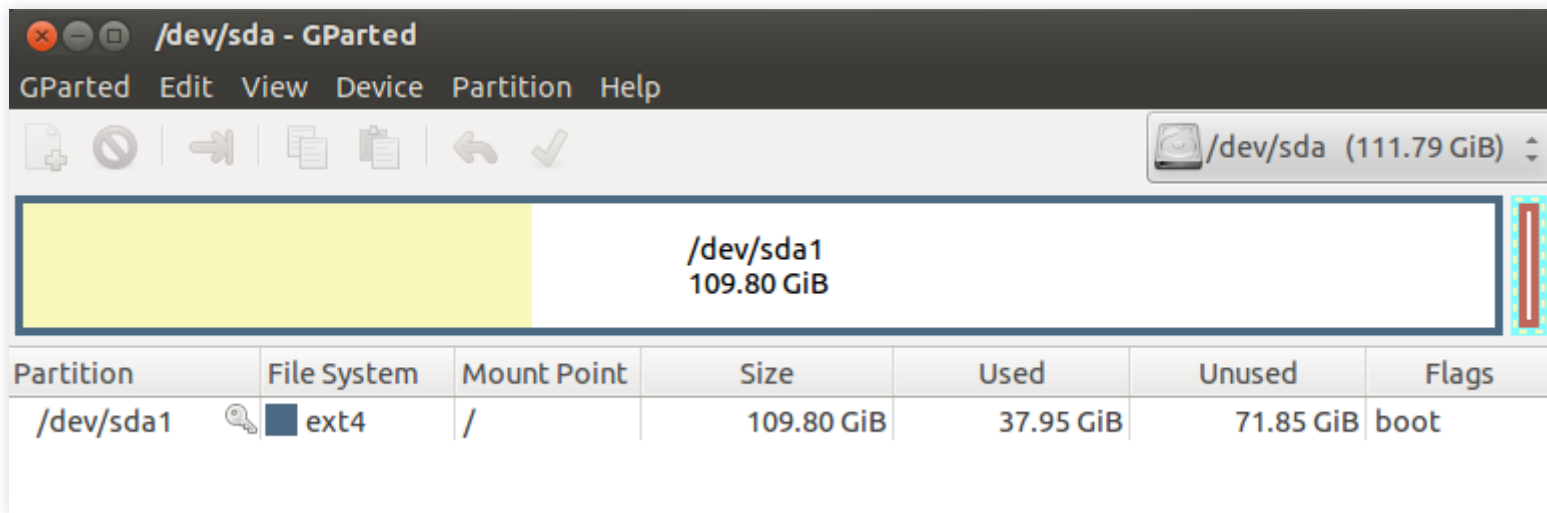


DRZEWO KATALOGÓW LINUX

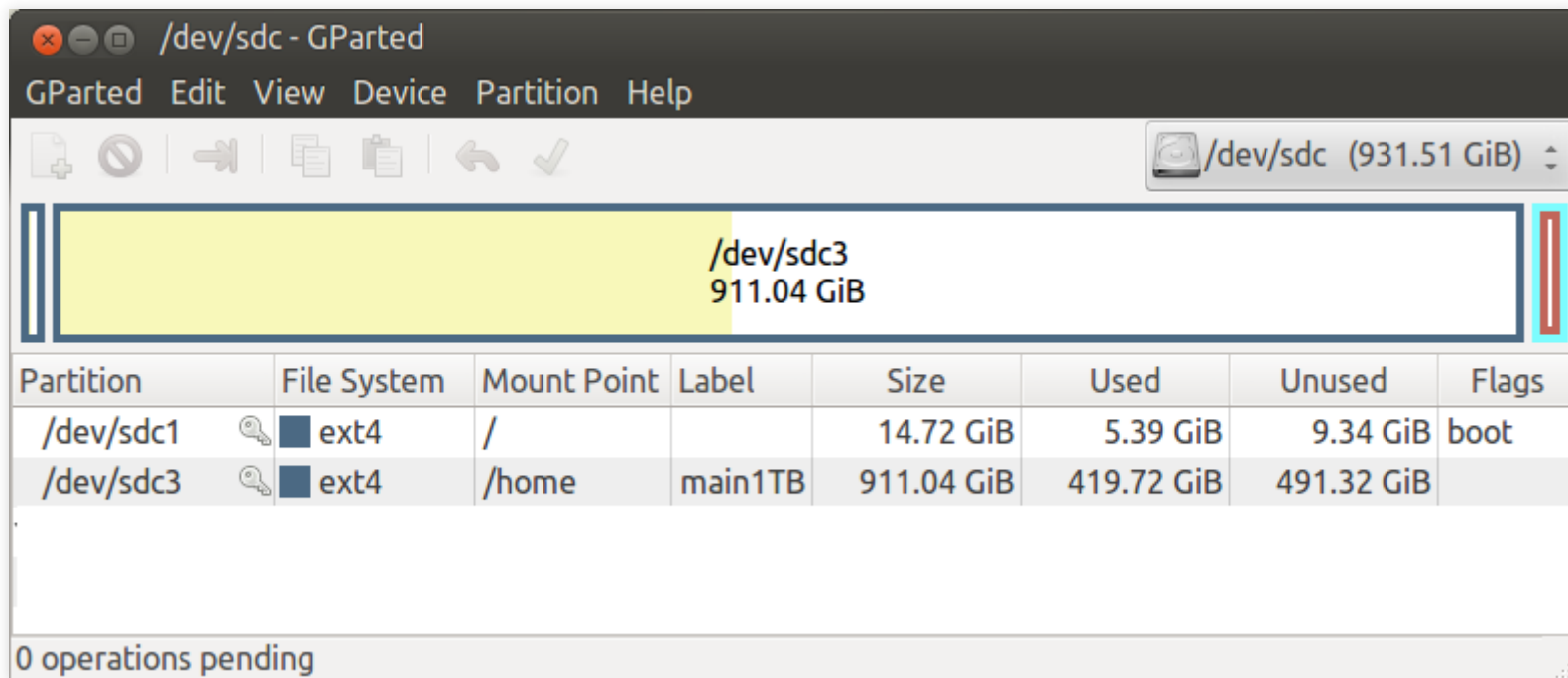
/bin
/boot
/dev
/etc
/home
/lib
/mnt
/opt
/proc
/root
/tmp
/usr
/var



DYSKI W LINUXSIE

The screenshot shows the GParted application window for the disk /dev/sda. The title bar reads "/dev/sda - GParted". The menu bar includes "GParted", "Edit", "View", "Device", "Partition", and "Help". The toolbar contains icons for adding, deleting, moving, copying, pasting, undo, and redo. A dropdown menu on the right shows the selected device as "/dev/sda (111.79 GiB)". The main area displays a single partition, /dev/sda1, with a size of 109.80 GiB. Below this, a table provides detailed information about the partition.

Partition	File System	Mount Point	Size	Used	Unused	Flags
/dev/sda1	ext4	/	109.80 GiB	37.95 GiB	71.85 GiB	boot

The screenshot shows the GParted application window for the disk /dev/sdc. The title bar reads "/dev/sdc - GParted". The menu bar includes "GParted", "Edit", "View", "Device", "Partition", and "Help". The toolbar contains icons for adding, deleting, moving, copying, pasting, undo, and redo. A dropdown menu on the right shows the selected device as "/dev/sdc (931.51 GiB)". The main area displays a single partition, /dev/sdc3, with a size of 911.04 GiB. Below this, a table provides detailed information about the partitions. At the bottom, a status bar indicates "0 operations pending".

Partition	File System	Mount Point	Label	Size	Used	Unused	Flags
/dev/sdc1	ext4	/		14.72 GiB	5.39 GiB	9.34 GiB	boot
/dev/sdc3	ext4	/home	main1TB	911.04 GiB	419.72 GiB	491.32 GiB	

0 operations pending

NAJWAŻNIEJSZE KATALOGI

- /bin** - pliki wykonywalne - narzędzia/polecenia/aplikacje
- /boot** - stąd startuje system
- /dev** - lista urządzeń (sprzęt) "widziany" przez system
- /etc** - pliki konfiguracyjne
- /home** - katalogi domowe użytkowników
- /mnt** - zamontowane systemy plików, np. pendrive
- /opt** - dodatkowe (opcjonalne) programy
- /root** - katalog domowy użytkownika **root**
- /tmp** - pliki tymczasowe

KATALOG DOMOWY

- Każdy użytkownik ma swój katalog domowy
- Katalogi domowe znajdują się w katalogu **/home**
- Domyślnie użytkownik A nie ma uprawnień do zapisu w katalogu użytkownika B
- Można jedynie przeglądać katalogi innych użytkowników
- Katalog domowy **aktualnego użytkownika** skrótowo oznacza się symbolem **~**

jeśli aktualnie zalogowany jest Tomek:

`/home/tomek` = `~`

`/home/tomek/projekty` = `~/projekty`

`/home/bartek/projekty` \neq `~/projekty`

ŚCIEŻKI W LINUXSIE

- Ścieżka jest zapisem lokalizacji pliku lub katalogu
- Pełna ścieżka zaczyna się od katalogu **root** czyli /
- W Windowsach lokalizacja zaczyna się od litery dysku
- Linux:

```
/bin/google/chrome
```

```
/home/tomek/projekty
```

- Windows:

```
C:\Programy\Firefox\firefox.exe
```

```
D:\projekty
```

ŚCIEŻKI WZGLĘDNE I BEZWZGLĘDNE

Ścieżka względna to taka, która podawana jest względem aktualnego katalogu



```
obrazki  
projekty  
projekty/projekt1  
projekty/projekt2  
projekty/projekt2/biblioteki
```

ŚCIEŻKI WZGLĘDNE I BEZWZGLĘDNE

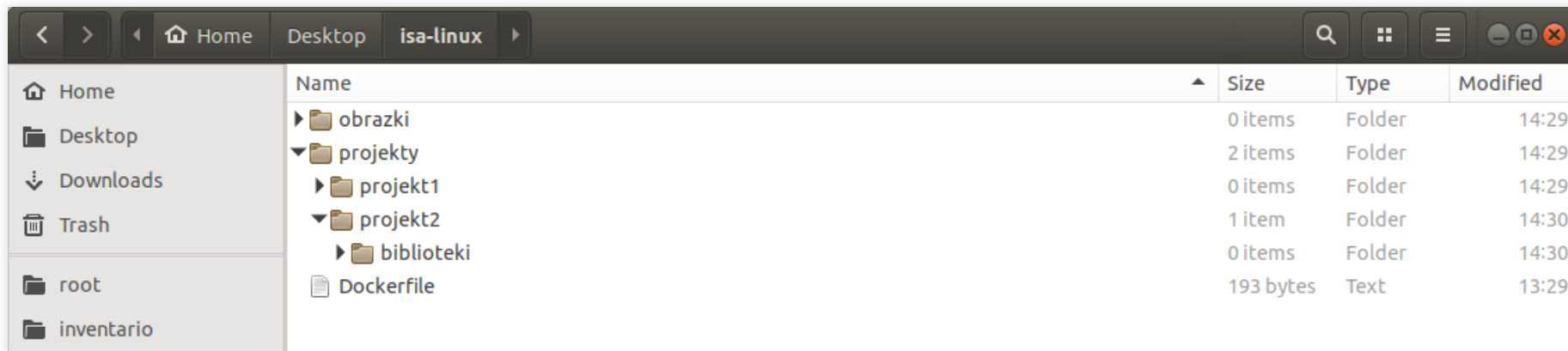
Ścieżka bezwzględna to pełna ścieżka do katalogu/pliku od katalogu root, niezależnie od aktualnego katalogu



```
/home/tomek/Desktop/isa-linux/obrazki  
/home/tomek/Desktop/isa-linux/projekty  
/home/tomek/Desktop/isa-linux/projekty/projekt1  
/home/tomek/Desktop/isa-linux/projekty/projekt2  
~/Desktop/isa-linux/projekty/projekt2/biblioteki
```

ŚCIEŻKI WZGLĘDNE C.D.

Jak zapisać ścieżkę względną do katalogu **projekt2** będąc w katalogu **projekt1**?



`../projekt2`

A do pliku **Dockerfile**?

`../../Dockerfile`

ROZSZERZENIA PLIKÓW

- Pliki w Linuksie często nie mają rozszerzenia
- Utrudnia to szybkie rozpoznanie rodzaju pliku
- Można też pomylić plik z katalogiem

```
/home/tomek/projekty
```

```
/home/tomek/webstorm
```

- Do sprawdzenia rodzaju pliku służy polecenie **file**
- Kolorowanie plików i katalogów w konsoli pozwala rozróżnić je od siebie

KONSOLA



- Urządzenie do obsługi komputera: wyświetlacz + "guziki"
- Zwykle składało się z klawiatury i terminala tekstowego
- Terminal służył do wprowadzania tekstu i wyświetlania rezultatów poleceń (ekran)
- Konsola była pojedynczym punktem komunikacji z komputerem (HMI)
- Zwykle zamiennie używa się określeń **konsola** i **terminal**

TERMINAL TEKSTOWY

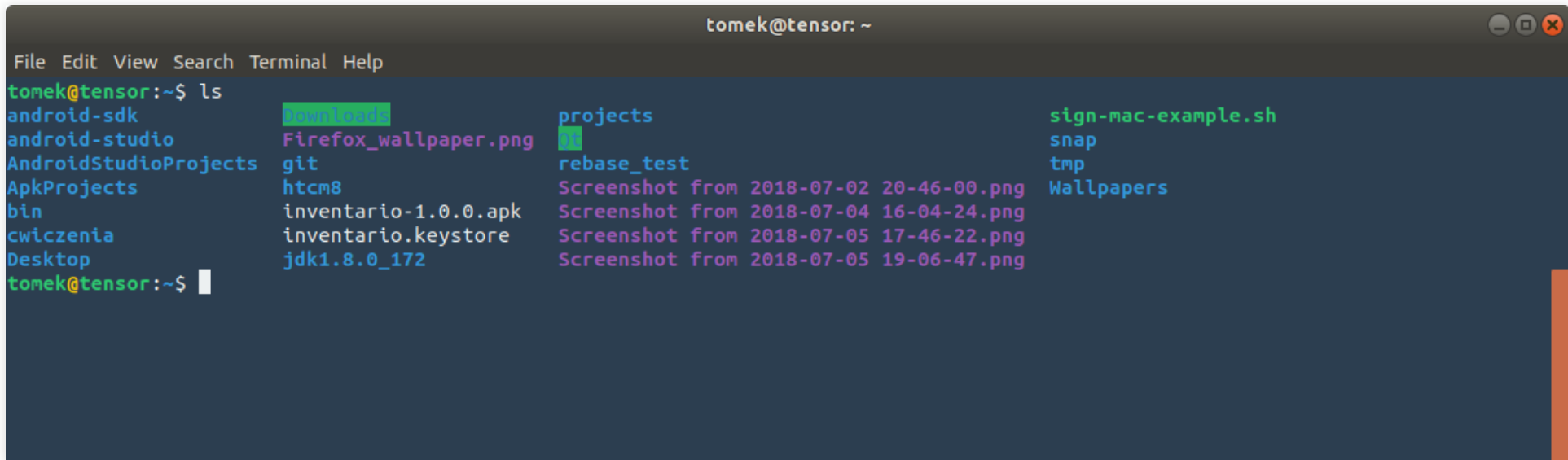
- W Linuksie jest to proces (uruchomiona aplikacja) który czeka na polecenia od użytkownika
- W reakcji na polecenia użytkownika wykonuje polecenia, uruchamia inne aplikacje i wyświetla jakiś rezultat
- Taki rodzaj aplikacji określa się jako CLI - command line interpreter
- Linux ma 2 rodzaje terminali **terminale wirtualne** i **emulatory terminala**

TERMINALE WIRTUALNE

- Są to procesy uruchamiane przez kernel w czasie startu systemu
- Jest ich kilka - umożliwiają uruchamianie różnych aplikacji za pomocą jednego komputera
- Można się między nimi przełączać:
Ctrl + Alt + F1-F6
- W instalacjach serwerowych (bez X) jest to jedyna metoda obsługi komputera

EMULATORY TERMINALA

- Uruchamiane przez użytkownika
- Zwykle uruchamiane w środowisku graficznym w formie okienek
- Najczęściej umożliwiają otwieranie nowych terminali w formie zakładek



The screenshot shows a terminal window titled "tomek@tensor: ~". The terminal has a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". The command "ls" has been executed, displaying a list of files and directories in a color-coded format. The files are arranged in four columns.

```
tomek@tensor: ~$ ls
android-sdk          download
android-studio       Firefox_wallpaper.png
AndroidStudioProjects  git
ApkProjects          htc8
bin                  inventario-1.0.0.apk
cwiczenia             inventario.keystore
Desktop              jdk1.8.0_172
tomek@tensor: ~$
```

Column 1	Column 2	Column 3	Column 4
android-sdk	download	projects	sign-mac-example.sh
android-studio	Firefox_wallpaper.png	rebase_test	snap
AndroidStudioProjects	git	Screenshot from 2018-07-02 20-46-00.png	tmp
ApkProjects	htc8	Screenshot from 2018-07-04 16-04-24.png	Wallpapers
bin	inventario-1.0.0.apk	Screenshot from 2018-07-05 17-46-22.png	
cwiczenia	inventario.keystore	Screenshot from 2018-07-05 19-06-47.png	
Desktop	jdk1.8.0_172		

POLECENIA W TERMINALU

- Polecenia to małe aplikacje które uruchamiane są jako procesy
- Polecenia mają nazwy: **ls, cd, reboot, git**
- Polecenia mogą przyjmować argumenty (parametry) lub być bezparametrowe
- Polecenia i ich argumenty rozdzielamy spacjami

```
ls
```

```
ls /etc
```

```
ls -l
```

FLAGI

- Argumenty zaczynające się od **-** lub **--** to tzw. flagi
- Flagi służą do sterowania/parametryzowania poleceń i wpływają na ich działanie
- Listę dostępnych flag można zwykle uzyskać przekazując do polecenia flagę **--help**
- Alternatywnie pomoc można uzyskać poleceniem **man [POLECENIE]**

```
ls
```

```
ls -l
```

```
ls -la --reverse
```

```
ls --help
```

```
man ls
```

FLAGI

- Po niektórych flagach należy podać dodatkową wartość
- Wartość podana po flagie "doprecyzowuje" działanie polecenia

Polecenie **head** wyświetla domyślnie pierwsze 10 linijek pliku

```
head example.log
```

Wyświetl pierwsze 3 linijki pliku example.log

```
head -n 3 example.log
```

Po flagie **-n** musi być podana liczba linii do wyświetlenia

```
head -n example.log - BŁĄD
```

FLAGI

- Flagi rozpoczynające się od **--** można parametryzować na dwa sposoby
- Wartość można podać po spacji albo po znaku równości
- Flagi z pojedynczym minusem obsługują tylko wersję ze spacją

```
head -n 3 example.log
```

```
head --lines 3 example.log
```

```
head --lines=3 example.log
```

```
head -n=3 example.log - BŁĄD
```

UŻYTKOWNICY, GRUPY I UPRAWNIENIA

- Uprawnienia w linuxsie: **odczyt, zapis, uruchamianie**
read, write, execute - rwx
- Uprawnienia mają zastosowanie zarówno dla katalogów jak i plików
 - **read** - możliwość otwarcia i zobaczenia zawartości pliku; dla katalogu - możliwość zobaczenia jego zawartości
 - **write** - możliwość modyfikacji(zapisu) pliku lub jego usunięcia; dla katalogu - dodawanie i usuwanie plików; brak uprawnień write dla katalogu wciąż pozwala go usunąć ale z dodatkowym ostrzeżeniem/pytaniem
 - **execute** - mając uprawnienia do wykonania pliku, użytkownik może próbować go uruchomić; uprawnienia execute dla katalogu pozwalają do niego wejść

GRUPY

- Grupy służą do zarządzania użytkownikami i ich organizowania
- Za pomocą grup można nadawać uprawnienia dla wielu użytkowników na raz
- Nie trzeba przy tym ustawiać uprawnień dla każdego użytkownika z osobna
- Wszystkie grupy są zebrane w pliku **/etc/group**


```
tomek@tensor:~/projects$ cat /etc/group
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:syslog,tomek
tty:x:5:
disk:x:6:
lp:x:7:
tomek:x:1000:
```

UŻYTKOWNICY VS GRUPY

- Użytkownicy należą zwykle do wielu grup
- Każdy użytkownik należy domyślnie do grupy o tej samej nazwie co użytkownik
- Jest to tzw. główna grupa tego użytkownika
- Żeby sprawdzić, do jakich grup należy użytkownik należy użyć polecenia **groups**
- Grupy innego użytkownika sprawdzamy za pomocą **groups [NAZWA_UŻYTKOWNIKA]**

SPRAWDZANIE UPRAWNIENÍ

- Uprawnienia najłatwiej jest sprawdzić w konsoli poleceniem **ls -l [PLIK/KATALOG]**
- Podając plik dostajemy informacje właśnie o tym pliku
- Podając katalog dostajemy informacje o **zawartości wewnątrz** tego katalogu
- Jeśli chcemy dostać informacje tylko o podanym katalogu, należy użyć **ls -ld [KATALOG]**



```
tomek@tensor:~$ ls -l Firefox_wallpaper.png
-rw-rw-r-- 1 tomek tomek 2902021 gru 16 2017 Firefox_wallpaper.png

tomek@tensor:~$ ls -ld projects
drwxrwxr-x 23 tomek tomek 4096 lip 8 10:01 projects
```

LS - OBJAŚNIENIA



```
tomek@tensor:~$ ls -ld projects
```

```
drwxrwxr-x 23 tomek tomek 4096 lip  8 10:01 projects
```

```
^
```

```
d => katalog
```

```
- => plik
```

```
l => link symboliczny (dowiązanie)
```

```
tomek@tensor:~$ ls -l Firefox_wallpaper.png
```

```
-rw-rw-r-- 1 tomek tomek 2902021 gru 16  2017 Firefox_wallpaper.png
```

```
^^^^^^^^
```

UPRAWNIENIA (właściciel, grupa, pozostali - user, group, others)

```
drwxrwxr-x 23 tomek tomek 4096    lip  8 10:01 projects
```

```
-rw-rw-r--  1 tomek tomek 2902021 gru 16  2017 Firefox_wallpaper.png
```

ZAPIS UPRAWNIENÍ


- **rwX** **rwX** **rwX**

d **rwX** **rwX** **rwX**

l **rwX** **rwX** **rwX**

- Uprawnienia składają się z 10 znaków
- Pierwszy znak od lewej mówi, czy mamy do czynienia z plikiem, katalogiem czy z linkiem symbolicznym
- Pozostałe 9 znaków to właściwe uprawnienia podzielone na trzy sekcje, po 3 znaki każda:
 - Uprawnienia **właściciela**
 - Uprawnienia **członków grupy**
 - Uprawnienia **pozostałych użytkowników**

UPRAWNIENIA - POLECENIE LS



```
drwxrwxr-x 23 tomek administratorzy 4096 lip 8 11:57 projects
```

- Właścicielem katalogu jest użytkownik **tomek**
- Katalog należy do grupy **administratorzy**
- Właściciel może odczytać zawartość katalogu, robić w nim dowolne zmiany i do niego wejść
- Dowolny użytkownik należący do grupy **administratorzy** może w tym katalogu zrobić dokładnie to samo co właściciel - **tomek**
- Pozostali użytkownicy mogą przeglądać katalog i do niego wejść - nie mogą natomiast niczego dodawać/usuwać

ZMIANA UPRAWNIENÍ

- Jeśli chcemy, żeby nawet administratorzy nie mogli modyfikować katalogu, zabieramy im uprawnienia do zapisu poleceniem

```
chmod g-w projects
```



```
drwxr-xr-x 23 torek administratorzy 4096 lip 8 11:57 projects
```

- W tym momencie tylko właściciel może tworzyć/usuwać zawartość

ZMIANA UPRAWNIENÍ

- Jeśli chcemy, dać wszystkim innym użytkownikom dodać możliwość robienia zmian:

```
chmod o+w projects
```



```
drwxr-xrwx 23 tomek administratorzy 4096 lip 8 11:57 projects
```

- Teraz katalog mogą modyfikować **tomek** i wszyscy użytkownicy za wyjątkiem tych, którzy są w grupie **administratorzy**



ZMIANA UPRAWNIENÍ

- W identyczny sposób możemy modyfikować inne uprawnienia: r, w, x

```
chmod u-r projects
```

```
chmod g+x projects
```

- Możemy też dodawać/usuwać uprawnienia ze wszystkich sekcji na raz

```
chmod a-x projects
```



```
$ ls -l projects
```

```
drwxrwxrwx 23 torek administratorzy 4096 lip 8 11:57 projects
```

```
$ chmod a-x projects
```

```
$ ls -l projects
```

```
drw-rw-rw- 23 torek administratorzy 4096 lip 8 11:57 projects
```

UŻYTKOWNIK ROOT

- Użytkownik **root** - superuser, administrator
- Ma uprawnienia do wszystkiego: tworzenie i kasowanie plików, instalacja systemu, wprowadzanie zmian w katalogach systemowych, "grzebanie" w katalogach wszystkich użytkowników
- Zwyczajny użytkownik domyślnie ma uprawnienia wyłącznie do własnego katalogu domowego
- Można przełączyć się na użytkownika **root** ale trzeba znać jego hasło
- Żeby zrobić coś poza katalogiem domowym lub np. zainstalować jakąś aplikację można przełączyć się w **tryb użytkownika root**
- Służy do tego polecenie **sudo**

SUDO

- sudo używa się poprzedzając polecenie tym słowem:
sudo [POLECENIE] [PARAMETRY_POLECENIA]
- Żeby użytkownik mógł używać sudo, musi należeć do pliku **/etc/sudoers**
- Alternatywnie użytkownik musi należeć do grupy **sudo** a w pliku sudoers musi istnieć linia

```
%sudo    ALL=(ALL:ALL) ALL
```

- Lepiej robić to przez grupy i nie modyfikować pliku sudoers - można stracić dostęp do systemu

INSTALACJA OPROGRAMOWANIA

- W Ubuntu oprogramowanie można instalować poleceniem **sudo apt-get install [NAZWA]**
- Instalacja oprogramowania wymaga uprawnień użytkownika root
- Użycie bez **sudo** zwraca błąd - brak uprawnień root'a
- Przed instalacją warto zaktualizować bazę oprogramowania **sudo apt-get update**
- Usuwanie oprogramowania: **sudo apt-get remove [NAZWA]**

```
sudo apt-get update
sudo apt-get install git
sudo apt-get install mc pinta vim
```

ZARZĄDZANIE UŻYTKOWNIKAMI

- Użytkowników w konsoli dodaje się poleceniem **useradd [NAZWA]**
- Usuwanie użytkowników: **userdel [NAZWA]**
- Wywołanie **useradd ania** tworzy jedynie użytkownika i jego domyślną grupę
- Jeśli chcemy stworzyć też katalog domowy trzeba użyć **useradd -m ania**
- Żeby dodać użytkownika do jakiejś grupy należy użyć **usermod -G [GRUPA] [UŻYTKOWNIK]**
- Użytkownik domyślnie nie ma hasła - trzeba je nadać poleceniem **passwd [UŻYTKOWNIK]**
- Zarządzanie użytkownikami wymaga uprawnień root'a (trzeba używać sudo)

ZARZĄDZANIE GRUPAMI

- Grupy dodaje się poleceniem **groupadd [NAZWA]**
- Usuwanie grup: **groupdel [NAZWA]**
- Pliki i katalogi mają właściciela i grupę do której należą (**ls -l**)



PLIKI I KATALOGI VS GRUPY

- Pliki i katalogi również należą do grup
- Pliki i katalogi tworzone przez użytkownika X trafiają do domyślnej grupy tego użytkownika
- Grupa w której znajduje się plik/katalog jest powiązana z zapisem uprawnień do tego pliku/katalogu
- duuu**ggg**ooo - uprawnienia dla grupy mówią, co z plikiem/katalogiem mogą zrobić użytkownicy należący do grupy, w której dany plik/katalog się znajduje



```
drwxr-xr-x 23 torek torek
drwxr-xr-x 23 torek sudo
```

```
4096 lip 8 11:57 projects
```

```
4096 lip 8 11:58 projects
```

ZMIANA WŁAŚCICIELA I GRUPY

- Zmiana właściciela pozwala "przepisać" plik z jednego użytkownika na drugiego

```
chown [UŻYTKOWNIK] [PLIK/KATALOG]
```

- Zmiana grupy dla pliku/katalogu

```
chgrp [GRUPA] [PLIK/KATALOG]
```

- Można jednym poleceniem zmienić zarówno właściciela jak i grupę

```
chown [UŻYTKOWNIK]:[GRUPA] [PLIK/KATALOG]
```



PRZEŁĄCZANIE SIĘ MIĘDZY UŻYTKOWNIKAMI

- Żeby przełączyć się na innego użytkownika w konsoli należy użyć polecenia **su**

```
su - ania
```

- Żeby wrócić do poprzedniego użytkownika używamy **Ctrl + D**



```
tomek@tensor:~$ su - ania
```

```
Password:
```

```
ania@tensor:~$ pwd
```

```
/home/ania
```

```
ania@tensor:~$
```

UBUNTU W KONTENERZE

- Będziemy grzebać w katalogach systemowych
- Możemy coś popsuć
- Bez reinstalacji systemu możemy uruchomić "czystą" instancję Ubuntu
- Możemy uruchomić kilka instancji na jednym komputerze

INSTALACJA DOCKERA

Instrukcja instalacji

```
curl -fsSL https://download.docker.com/linux/debian/gpg | sudo  
apt-key add -
```

```
sudo add-apt-repository "deb [arch=amd64]  
https://download.docker.com/linux/ubuntu $(lsb_release -cs)  
stable" sudo apt-get update
```

```
sudo apt-get install docker-ce
```

```
sudo groupadd docker
```

```
sudo usermod -aG docker $USER
```

LOGOUT => LOGIN

PODSTAWOWE POJĘCIA

- **Image** - gotowy do uruchomienia system
- **Container** - pojedyncza instancja systemu uruchomiona z wybranego obrazu
- Z pojedynczego obrazu można uruchomić wiele kontenerów - **instancji** systemu

ŚCIAĞANIE OBRAZU

```
docker pull [NAZWA_OBRAZU]
```

<https://hub.docker.com/explore/> - Repozytorium obrazów Dockera

```
tomek@tensor:~$ docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
9bb5a5d4561a: Pull complete
Digest: sha256:3e1764d0f546ceac4565547df2ac4907fe46f007ea229fd7ef27
Status: Downloaded newer image for hello-world:latest
```

Ściąganie Ubuntu 16.04

```
docker pull ubuntu:xenial
```

LISTOWANIE OBRAZÓW

docker images

```
tomek@tensor:~$ docker images
```

REPOSITORY	TAG	IMAGE ID
ubuntu	xenial	5e8b97a2a082
hello-world	latest	e38bc07ac18e
alpine	3.5	02674b9cb179

URUCHAMIANIE OBRAZU

```
docker run -it [NAZWA_OBRAZU]
```

Uruchamianie Ubuntu

```
docker run -it ubuntu:xenial  
root@4fddd3261cc7:/#
```

Wyjście: **Ctrl + D**

To polecenie powoduje uruchomienie instancji wybranego obrazu.

Można uruchomić wiele instancji tego samego obrazu.

PRZYGOTOWANIE OBRAZU

Ściągamy repozytorium

```
git clone https://github.com/infoshareacademy/jdqz2_linux
```

Wchodzimy do katalogu:

```
cd jqdz2_linux
```

Przełączamy się na branch **docker**:

```
git checkout docker
```

Budujemy spersonalizowaną kopię obrazu Ubuntu o nazwie **isa**

```
docker build -t isa .
```

URUCHOMIENIE OBRAZU "ISA"

```
docker run -it isa
```

```
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.
```

```
admin@6a607d9ec6d4:/$
```

Użytkownik: **admin**

Hasło: **admin**

UŻYTKOWNICY - ĆWICZENIE

- Dodać użytkowników **ania i tomek** wraz z katalogami domowymi
- Ustawić użytkownikom "bezpieczne" hasła: **ania i tomek**
- Dodać anię do grupy **sudo**
- Przełączyć się na użytkownika ania: **su - ania**
- Przejść do katalogu domowego tomka i stworzyć plik **touch ania.txt**
- Wylogować się przez **Ctrl + D**
- Zalogować się na użytkownika tomek i stworzyć plik **tomek.txt** w katalogu domowym ani
- Dodać anię do grupy **tomek**
- Jako ania stworzyć plik w katalogu domowym tomka **bez sudo**
- Jako tomek dodać uprawnienia do zapisu w katalogu domowym **dla grupy** i spróbować jeszcze raz jako ania
-

PORUSZANIE SIĘ W KONSOLI

Linux prompt

```
tomek@tensor:~$
```

```
użytkownik@host:aktualny_katalog
```

- Wypisanie ścieżki bezwzględnej do aktualnego katalogu:
pwd
- Wypisanie pełnej nazwy hosta:**hostname**
- Wypisanie aktualnego użytkownika**whoami**
- Żeby włączyć kolorowy prompt
 - Edytujemy plik **~/.bashrc** np. przez edytor nano
 - Szukamy linijki **#force_color_prompt=yes**
 - Usuwamy znak **#**
 - Jeżeli chcemy zmienić kolory, trzeba podmienić linijkę zawierającą **PS1=**
 - Generator kolorów do PS1 <http://ezprompt.net/>

LISTOWANIE ZAWARTOŚCI KATALOGU

- Wypisanie zawartości katalogu

```
ls          ls -l          ls -l [KATALOG]
```

- Wypisanie wszystkich - również ukrytych - plików/katalogów

```
ls -a       ls -la
```

- Wyświetlenie rozmiaru plików/katalogów w bardziej zrozumiałych jednostkach (KB, MB, GB...)

```
ls -lh
```

- Wyświetlenie informacji o katalogu a nie o jego zawartości

```
ls -ld [KATALOG]
```

ZMIANA KATALOGU

- Wejście do katalogu (ścieżka względna lub bezwzględna)

```
cd test          cd /etc/init
```

- Przejście do katalogu nadrzędnego

```
cd ..
```

- Przejście 3 katalogi wyżej

```
cd ../../..
```

- Przejście do swojego katalogu domowego

```
cd      lub      cd ~
```

- Przejście do poprzedniego katalogu

```
cd -
```

TWORZENIE KATALOGÓW

- Stworzenie katalogu **test** w aktualnym katalogu

```
mkdir test
```

- Stworzenie katalogu **test** w katalogu **/home/tomek/projects**

```
mkdir /home/tomek/projects/test
```

- Stworzenie ciągu podkatalogów **projects/project1/libraries**

```
mkdir -p projects/project1/libraries
```

Użycie powyższego polecenia bez flagi **-p** spowoduje błąd.

PLIKI

- Tworzenie pliku **test.txt** w aktualnym katalogu

```
touch test.txt
```

W przypadku istniejącego pliku zmieni się ostatnia data jego modyfikacji

- Wypisanie pliku na konsolę

```
cat test.txt
```

- Wypisanie pierwszych 10 linii pliku

```
head test.txt
```

Wypisanie pierwszych 3 linii pliku

```
head -n 3 test.txt
```

- Analogiczne polecenie, ale wypisuje ostatnie linijki z pliku

```
tail test.txt
```

```
tail -n 5 test.txt
```

USUWANIE

- Usuwanie plików

```
rm test.txt          rm /home/tomek/test.txt
```

- Usuwanie katalogów

```
rm -r test          rm -r /home/tomek/test
```

- Usuwanie tylko podkatalogu test1 w katalogu test2

```
rm -r test1/test2
```

Polecenie to szuka w katalogu test1 w aktualnym katalogu, wewnątrz szuka podkatalogu test2 i właśnie ten ostatni usunie

STDIN - STANDARDOWE WEJŚCIE

- Polecenia często odczytują dane z tzw. **standardowego wejścia**
- Dane na standardowe wejście możemy "podać" na kilka sposobów
- Jeśli uruchomimy polecenie **wc -l** będzie ono czekać wprowadzenie danych przez użytkownika

```
tomek@tensor:~$ wc -c  
abcdef  
7
```

- Po zakończeniu wprowadzania danych, polecenie wypisze ilość znaków wprowadzonych przez użytkownika
- (wc policzyło również znak nowej linii - klawisz enter - dlatego wypisało 7)

STDIN - STANDARDOWE WEJŚCIE

- Jeżeli dane mamy w pliku i chcemy zawartość tego pliku przekazać na **stdin** możemy użyć takiego zapisu

```
tomek@tensor:~$ wc -c < literki.txt  
7
```


STDOUT - STANDARDOWE WYJŚCIE

- Polecenie **wc** w poprzednim przykładzie "wyprodukowało" rezultat w formie liczby 7
- Polecenie pracowało na danych ze standardowego wejścia, a rezultat wypisało na **standardowe wyjście**
- Zawartość standardowego wyjścia zwykle trafia do terminala w formie tekstu
- Możemy jednak zapisać rezultat polecenia do pliku - przekierować standardowe wyjście

```
ls -l /home/tomek > domowy.txt
```

- W pliku **domowy.txt** dostajemy wówczas listę plików i katalogów z polecenia ls

STDOUT - PRZEKIEROWANIA

- Przekierowanie do pliku za pomocą operatora **>** kasuje całą zawartość tego pliku i dopiero wtedy zapisuje rezultat polecenia
- Musimy być ostrożni, bo możemy bezpowrotnie stracić dane
- Jeżeli chcemy **dopisać** coś do pliku, bez niszczenia zawartości należy użyć operatora

```
ls -l /home/tomek >> domowy.txt
```

- Najlepiej to widzieć przekierowując do tego samego pliku kilka razy polecenie **date**

```
date >> data.txt  
date >> data.txt
```

PIPES

- Polecenia można ze sobą łączyć - przekazywać wyjście jednego polecenia na wejście następnego
- Służy do tego operator pipe - |

```
cat plik.txt | wc -l
```

- Powyższe polecenie zrobi to samo co zapis

```
wc -l < plik.txt
```

- Pipe przydaje się najbardziej, kiedy chcemy wyjście z jednego polecenia przekazać bezpośrednio na wejście innego
- Możemy też tworzyć ciągi poleceń (możemy łączyć więcej niż 2 polecenia)

PIPES - PRZYKŁADY

- Chcemy wyświetlić 5 ostatnich plików/katalogów ze swojego katalogu domowego, posortowanych malejąco

```
ls ~ | sort --reverse | head -n 5
```

- Chcemy wylistować zawartość katalogu domowego i "wyciąć" z każdej linii godzinę modyfikacji (znaki od 40 do 45)

```
ls -l ~ | cut -c 40-45
```

PRZEKIEROWANIA - ĆWICZENIE

- Przejść do katalogu ze sklonowanym repozytorium
- Przełączyć się na branch **pipes**
- Wyświetlić datę na konsoli
- Wpisać aktualną datę do pliku **dates.txt**
- **Dopisać** na koniec pliku aktualną datę - w pliku mają teraz być **2 linijki**
- Ponownie wpisać aktualną datę do pliku, ale za pomocą przekierowania **>**
- Straciliśmy wszystkie poprzednie wpisy - warto o tym pamiętać robiąc przekierowania

PIPES - ĆWICZENIE

- Wypisać zawartość pliku **words.txt** na konsolę i zapoznać się z zawartością (**cat**)
- Wyświetlić statystyki pliku **words.txt** przekazując ten plik jako argument polecenia **wc**
- Policzyc ilość linii w pliku **words.txt**
- Policzyc ilość linii w pliku words ale tak, żeby nazwa pliku nie pojawiała się na ekranie
- (wypisać plik na konsolę i przekierować przez pipe na standardowe wejście polecenia wc)

PIPES - ĆWICZENIE

- Zostać w tym samym branch'u i wylistować pliki w katalogu
- Wypisać i posortować listę plików w tym katalogu (przyda się polecenie **sort**)
- Odwrócić kolejność sortowania (sprawdzić w dokumentacji, która flaga na to pozwala)
- Wypisać plik **numbers.txt** na konsolę
- Wypisać i posortować liczby z pliku **numbers.txt**
- Jak wymusić sortowanie numeryczne?
- Posortować liczby z pliku **numbers.txt** w trybie numerycznym
- Posortować jeszcze raz, ale bez duplikatów (**sort w trybie unique**)

GREP

- Polecenie **grep** służy do wyszukiwania w treści
- Grep'a można użyć na wiele sposobów - jest bardzo uniwersalny
- Można dzięki niemu wyszukiwać w pojedynczych plikach, we wszystkich plikach w danym katalogu lub we wszystkich podkatalogach jakiegoś katalogu
- Do grep'a możemy również przekazać wyjście z jakiegoś polecenia i "przefiltrować" je
- Grep jest bardzo wydajny dzięki czemu można go używać nawet do dużej ilości danych (nawet pliki o rozmiarze rzędu GB)
- Grep pracuje na wyrażeniach regularnych - zaawansowane wyszukiwanie za pomocą wzorców

WYSZUKIWANIE W PLIKACH

- Poniższe wywołanie szuka słowa "error" w pliku "apache.log"

```
grep error apache.log
```

- Możemy szukać w kilku plikach na raz

```
grep error apache1.log apache2.log apache3.log
```

- Żeby przeszukać wszystkie pliki w jakimś katalogu

```
grep error /home/tomek/projects/*
```

- Szukanie tylko w plikach z rozszerzeniem .txt

```
grep error /home/tomek/logs/*.txt
```

- Szukanie rekursywne (katalog + wszystkie podkatalogi)

```
grep -R error /home/tomek/logs
```

WYSZUKIWANIE W STDIN GREP'A

- Poniższe 2 wywołania mają identyczny skutek

```
grep error apache.log
```

```
cat apache.log | grep error
```

- Można zatem przekierowywać wyjście jakiegoś polecenia na wejście grep'a
- Nie trzeba wówczas zapisywać wyjścia polecenia, żeby coś w nim wyszukać
- Po przekierowaniu wyjścia polecenia do grep'a, będzie on filtrował linie
- Ten prosty skrypt generuje liczbę pseudolosową co sekundę

```
while true; do rand; sleep 1; done
```

- Możemy odfiltrować np. tylko te liczby, które zawierają cyfrę 5

```
while true; do rand; sleep 1; done | grep 5
```

(BARDZO) PRZYDATNE FLAGI

- Wyszukiwanie bez rozróżniania wielkości znaków

```
grep -i ToMeK imiona.txt
```

- Wyszukiwanie linii, które **nie zawierają** słowa error

```
grep -v error apache.log
```

- Żeby policzyć ilość linii zawierających słowo error

```
grep -c error apache.log
```

- Wypisanie numerów linii, w których występuje szukane słowo

```
grep -n error apache.log
```

WYRAŻENIA REGULARNE

- Za pomocą wyrażeń regularnych możemy wyszukiwać wzorców
- Można ich używać nie tylko z grepem, ale również praktycznie w każdym języku programowania
- W materiałach są linki - temat jest bardzo obszerny
- Zwykłe słowo np. "error" może być wyrażeniem regularnym
- Podobnie sama cyfra/liczba może nim być
- Tworząc wzorce można robić kombinacje liczb, liter i znaków specjalnych
- Wyrażenia regularne - regular expressions - **regexy**

WYRAŻENIA REGULARNE

- Do budowania wyrażeń regularnych używa się specjalnych znaków, każdy ma swoją funkcję
- **[a-z]** - mała litera
- **[A-Z]** - wielka litera
- **[a-zA-Z]** - litera dowolnej wielkości
- **[0-9]** - dowolna cyfra
- **.** - (kropka) - dowolny znak (litera, cyfra, znak specjalny)
- Z tych klocków można zacząć budować wyrażenia regularne

abc[a-z][0-9]

b[auy]k

b.k[1-5]

KWANTYFIKATORY

- Czasami potrzebujemy znaleźć dany znak, ale powtórzony ileś razy po sobie
- Używamy do tego kwantyfikatorów umieszczanych po "klocku", którego szukamy
- **a*** - pasuje do słów zawierających zero lub więcej liter "a" następujących po sobie

```
regex: aaab*ccc
```

```
aaaccc - pasuje (zero wystąpień litery "b")
```

```
aaabccc - pasuje
```

```
aaabbbbbbbbbbccc - pasuje
```

- Gwiazdka najbardziej przydaje się, kiedy w szukanej linii jest jakaś sekcja, która nas nie interesuje
- Np. wszystkie adresy na Alei Grunwaldzkiej możemy znaleźć tak

```
regex: Aleja Grunwaldzka [0-9]*
```

```
Aleja Grunwaldzka 1 - pasuje
```

```
Aleja Grunwaldzka 152 - pasuje
```

```
Aleja Grunwaldzka 0 - też pasuje
```

```
fix: Aleja Grunwaldzka [1-9]+[0-9]*
```

KWANTYFIKATORY

- **x+** - znajduje jedno lub więcej wystąpień litery "x"

```
regex: ab1+d
ab1d   - pasuje
ab111d - pasuje
abd    - nie pasuje
abd11  - nie pasuje
```

- **c?** - znajduje te wystąpienia, które zawierają zero lub jedno "c"

```
regex: abc?d
abcd   - pasuje
abd    - pasuje
abccd  - nie pasuje (2 wystąpienia po sobie)
```

GREP + REGEX

- Żeby użyć grep'a z regexem należy dodać flagę **-E**
- Szukanie błędów HTTP 4xx

```
cat apache.log | grep -E 4[0-9][0-9]
```

- Szukanie linii zawierających requesty do strony z "paginacją"

```
cat apache.log | grep -E page=[0-9] +
```

- Przetworzone wyniki, możemy ponownie "grepować"

```
cat apache.log | grep -E page=[0-9] + | grep -E blog
```

- Można też szukać wzorców w wielu plikach

```
grep -R -E [regex] /home/tomek/projects
```


GREP - ĆWICZENIE

- Przejść do katalogu ze sklonowanym repozytorium **jdqz2_linux**
- Przełączyć się na branch **logs**
- W pliku **adresy.txt** wyszukać linie zawierające adres email
- Następnie wyszukać liniami które **nie zawierają** adresów email
- Wywołać **chown u-r adresy.txt**
- Ponownie wyszukać linie z adresami email
- Przywrócić prawa odczytu do pliku właścicielowi
- Wyszukać tylko te linie, które zawierają zapytania **POST**


















GREP - ĆWICZENIE

- Przełączyć się na branch **pipes**
- Wygrepować we wszystkich plikach linijki zawierające liczby dwucyfrowe
- Użyć operatorów początku i końca linii, żeby wyłapać te linijki
- Wyszukać linie zawierające co najmniej dwie litery "b" i wyświetlić numery linii
- Wyświetlić dla każdego pliku ilość linijek, które nie zawierają żadnej litery
- Wyświetlić informacje tylko o tych plikach, dla których poprzedni punkt zwrócił ilość linijek większą od zera



ZARZĄDZANIE PROCESAMI


















- Uruchamiając aplikację system tworzy z niej proces
- Każdy proces dostaje **PID** - process identifier
- Listę procesów możemy zobaczyć w aplikacji **System Monitor**

Processes Resources File Systems					
Process Name	ID	User	% CPU ▲	Priority	Mem
 gnome-shell	7966	tomek		1 Normal	190
 gnome-system-monitor	15540	tomek		1 Normal	1
 Xorg	7816	tomek		1 Normal	20
 systemd	7802	tomek		0 Normal	
 (sd-pam)	7803	tomek		0 Normal	
 gnome-keyring-daemon	7810	tomek		0 Normal	
 gdm-x-session	7814	tomek		0 Normal	67
 dbus-daemon	7822	tomek		0 Normal	
 gnome-session-binary	7831	tomek		0 Normal	
 ssh-agent	7913	tomek		0 Normal	32
 at-spi-bus-launcher	7923	tomek		0 Normal	58
 dbus-daemon	7928	tomek		0 Normal	48
 at-spi2-registryd	7930	tomek		0 Normal	71
 gvfsd	7943	tomek		0 Normal	100
 gvfsd-fuse	7948	tomek		0 Normal	54
 pulseaudio	7975	tomek		0 Very High	
 gnome-shell-calendar-server	7996	tomek		0 Normal	



PROCESS MONITOR

- Kolumna **ID** zawiera PID
- Kolumna **User** mówi, który użytkownik jest właścicielem procesu
- Dodatkowo możemy sprawdzić który proces zużywa najwięcej CPU i pamięci - przydaje się do diagnostyki

Processes Resources File Systems					
Process Name	ID	User	% CPU ▲	Priority	Mem
 gnome-shell	7966	tomek	1	Normal	190
 gnome-system-monitor	15540	tomek	1	Normal	1
 Xorg	7816	tomek	1	Normal	28
 systemd	7802	tomek	0	Normal	
 (sd-pam)	7803	tomek	0	Normal	
 gnome-keyring-daemon	7810	tomek	0	Normal	
 gdm-x-session	7814	tomek	0	Normal	67
 dbus-daemon	7822	tomek	0	Normal	
 gnome-session-binary	7831	tomek	0	Normal	
 ssh-agent	7913	tomek	0	Normal	32
 at-spi-bus-launcher	7923	tomek	0	Normal	58
 dbus-daemon	7928	tomek	0	Normal	48
 at-spi2-registryd	7930	tomek	0	Normal	71
 gvfsd	7943	tomek	0	Normal	100
 gvfsd-fuse	7948	tomek	0	Normal	54
 pulseaudio	7975	tomek	0	Very High	
 gnome-shell-calendar-server	7996	tomek	0	Normal	

PROCESY W KONSOLI

- Czasami procesy chcemy wylistować w konsoli
- Służy do tego polecenie **ps**
- Do polecenia najczęściej stosuje się flagi:
Pokaż wszystkie procesy:

```
ps -e
```

Pokaż wszystkie procesy, z nazwą użytkownika i argumentami, które zostały przekazane do procesu:

```
ps -ef
```

Pokaż tylko procesy aktualnego użytkownika:

```
ps -u
```

Pokaż tylko kolumnę **pid**

```
ps -e -o pid
```

ZABIJANIE PROCESÓW

- Procesy trzeba czasami zakończyć - graficzne można zamknąć "krzyżykiem", te bez interfejsu - można z konsoli
- Zakańczanie procesu w Linuksie polega na wysłaniu do procesu jednego z dostępnych sygnałów a służy do tego polecenie **kill**
- Listę dostępnych sygnałów można dostać w konsoli:

```
kill -l
```

- Zakończenie procesu wykonuje się za pomocą:
kill -s [SYGNAŁ] [PID]

ZABIJANIE PROCESÓW

- Zakładając że proces ma pid równy 12345 i chcemy, żeby się on zakończył, wysyłamy mu sygnał **SIGTERM** który ma numer **15**:

```
kill -s SIGTERM 12345  
kill -s 15 12345
```

- Aplikacje mogą zignorować lub zablokować ten sygnał - nie zawsze da się w ten sposób zakończyć proces
- Jeżeli proces się zawiesił i nie odpowiada, możemy próbować go zabić "mocniejszym" sygnałem **SIGKILL**:

```
kill -s SIGKILL 12345  
kill -s 9 12345
```

- Można też używać skrótu:

```
kill -9 12345          kill -15 12345
```

PROCESY - ĆWICZENIE

- Sprawdzić czy w systemie jest zainstalowana aplikacja **xclock** - zainstalować w razie braku
- Otworzyć 2 konsole(zakładki) i w pierwszej z nich uruchomić **xclock**
- Użyć polecenia ps z odpowiednimi flagami, żeby wylistować procesy wraz z ich identyfikatorami
- Znaleźć na liście PID dla procesu xclock (przekierować ps do grep'a lub less)
- Mając PID, zakończyć proces sygnałem SIGTERM
- Powtórzyć procedurę ale wysłać na koniec sygnał SIGKILL

POLECENIE MC

File Edit View Search Terminal Help

Left	File	Command	Options	Right
•[^]•				
«= ~/projects/isa/isa-linux				
↓	Name	Size	Modify	time
UP--DIR				
./		UP--DIR	lip	2 20:29
./git		4096	lip	12 19:29
./idea		4096	lip	11 23:53
/img		4096	lip	11 18:18
/node_modules		4096	lip	2 20:32
.gitignore		27	lip	2 20:33
index.html		69708	lip	11 23:52
isa-theme.css		6821	lip	10 20:58
logo-big.png		28379	lip	2 22:13
logo-small.png		68577	lip	2 21:09
package-lock.json		384	lip	2 20:32
package.json		282	lip	2 20:32
UP--DIR				
32G/133G (23%)				
•[^]•				
↓	Name	Size	Modify	time
.bash_aliases		624	lut	18 19:00
.bash_history		37317	lip	11 23:53
.bash_logout		220	sty	19 20:17
.bashrc		4139	maj	8 17:26
.dmrc		25	sty	19 20:17
.emulator_console_auth_token		16	lip	20 20:17
.gitconfig		309	mar	25 13:34
.lessht		105	lip	11 20:06
.node_repl_history		539	maj	4 19:30
.npmrc		70	lis	22 20:17
.nvidia-settings-rc		477	kwi	10 20:01
.pam_environment		306	maj	21 22:11
.profile		655	paź	22 20:17
.recently-used		1724	sie	26 20:17
.rnd		1024	kwi	16 18:55
.selected_editor		66	lip	8 14:25
.sqlite_history		1278	lis	18 20:17
.sudo_as_admin_successful		0	sty	19 20:17
.v8flags.5.1.281.103~d74694de9fef1b0.json		9468	lip	17 20:17
.v8flags.5.1.281.89.tomek.json		9468	lut	5 20:17
.v8flags.5.1.281.93.~d74694de9fef1b0.json		9468	cze	9 20:17
.viminfo		12118	maj	13 16:35
.wget-hsts		204	cze	18 19:15
.xbindkeysrc		1780	sty	22 20:17
.xinputrc		131	maj	21 22:10
.xsession-errors		3638	lis	28 20:17
.xsession-errors.old		2334	lis	28 20:17
.yarnrc		116	sie	26 20:17
Firefox_wallpaper.png		2902021	gru	16 20:17
inventario.keystore		2239	maj	8 18:05
*sign-mac-example.sh		157	maj	8 18:15
inventario.keystore				
32G/133G (23%)				

Hint: Want your plain shell? Press C-o, and get back to MC with C-o again.

tomek@tensor:~\$

1Help 2Menu 3View 4Edit 5Copy 6RenMov 7Mkdir 8Delete 9PullDn 10Quit

[^]

POLECENIE MC

- Midnight Commander - bardzo użyteczny manager plików
- Pozwala na bardzo szybkie przeglądanie katalogów i ich tworzenie, kopiowanie, przenoszenie plików a także usuwanie
- Można w sposób "graficzny" uruchamiać polecenie **chown**
- MC pozwala porównywać pliki i katalogi zaznaczone w obu panelach
- Do przełączania się między panelami służy klawisz **TAB**
- Wyjście z aplikacji - klawisz **F10**
- Często używane polecenia są wylistowane na dole ekranu i odpowiadają klawiszom **F1-F10**
- Więcej funkcji/opcji jest ukrytych w menu na górze ekranu (Left, File, Command, Options, Right)

UŻYWANIE MC

- Sporo działań w mc można zrobić myszą przez klikanie, ale wygodniej i szybciej działa się samą klawiaturą
- Kopiowanie - F5
- Przenoszenie - F6
- Zmiana nazwy - Shift+F6
- Tworzenie katalogu - F7
- Usuwanie - F8
- Podgląd zaznaczonego pliku - F3
- Edycja pliku - F4
- Wybór menu z góry ekranu - F9
- Przejście do konsoli - Ctrl+O
- Zaznaczanie - Insert, *, +, -

- <https://www.bezkompilatora.pl/linux-kilka-rzeczy-ktore-powinienes-wiedziec/>
- <https://www.bezkompilatora.pl/jak-zainstalowac-linuxa-cztery-sposoby-na-start-czesc-2/>
- <https://www.bezkompilatora.pl/badz-jak-hacker-wprowadzenie-do-wiersza-polecen/>
- <https://www.bezkompilatora.pl/instaluj-jak-hacker-zarządzanie-programami-z-wiersza-polecen/>
- <https://www.bezkompilatora.pl/linuxowy-system-plikow-z-lotu-ptaka/>
- https://en.wikipedia.org/wiki/List_of_Unix_commands
- www.linuxportal.pl/.../wprowadzenie_do_systemu_linux.html
- www.linode.com/docs/tools-reference/linux-users-and-groups/
- <http://grabun.pl/wyrazenia-regularne/wstep/>
- <https://kobietydokodu.pl/4-wyrazenia-regularne/>
- <http://przepis-na-lo.pl/2013/07/wprowadzenie-do-wyrazen-regularnych/>