Operating Systems

OS Overview, Linux Shell, VM and Containers

SoftUni Team

Technical Trainers

Software University

SoftUni

Software University

https://about.softuni.bg/

Have a Question?

Software University

sli.do

#qa-fund

Table of Contents

Operating Systems (OS) – Overview

OS Purpose and Structure

What is Operating System (OS)?

- The **operating system (OS)** manages applications (processes), users, file system and resources in a device
- The OS is loaded into a device through a process called **booting**
- OS enables applications to **interact** with the device's **hardware** and software **resources**
- Applications make requests for services through a defined interface called an **application program interface (API)**
- At **least one OS** must be **installed** in a device to run basic programs, e. g. Web browser, file explorer, video player

OS Main Functions

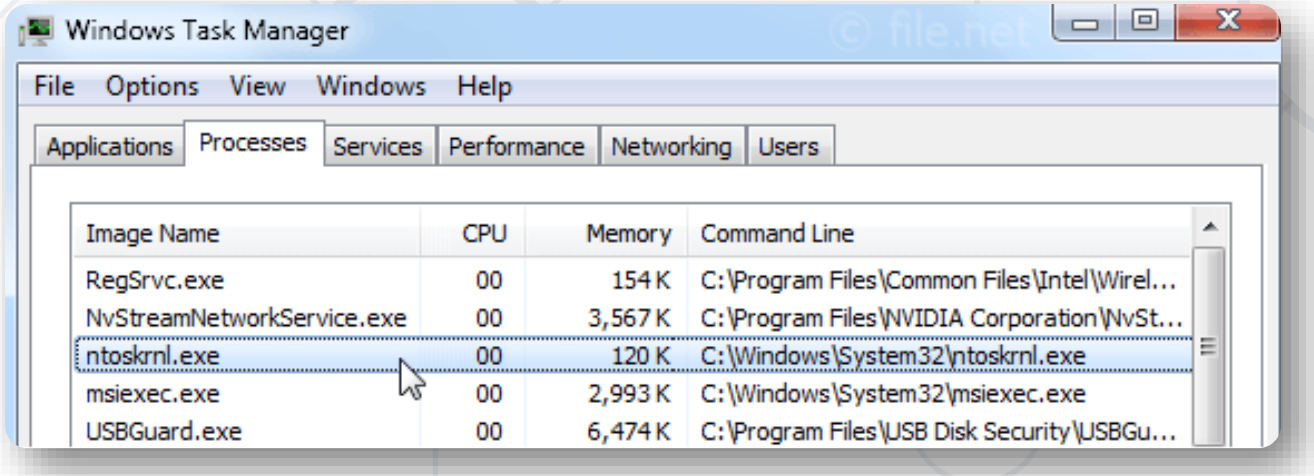
- **Booting** – turning on the device and loading the OS
- **App loading and execution** – load and run programs (processes, apps), start / view / pause / terminate apps
- **Process management** – allocates resources to OS processes, share data between processes, protects, and synchronizes them
- **Memory management** – controls and coordinates the memory allocation for the applications running in the OS
- **Disk management** – manages storage (hard drives, SSD disks, optical disk drives, flash drives) and file systems

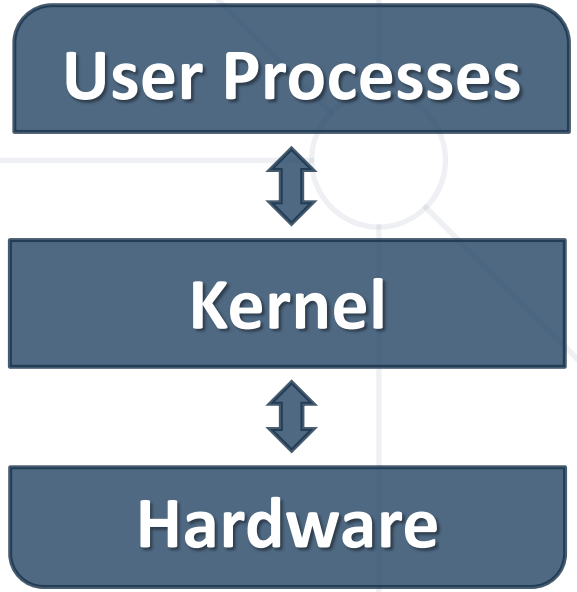
OS Main Functions

- **Device controlling** – controls the access to **physical devices** (like disk drives, CD/DVDs, USBs) and **virtual devices** (like random)
- **Networking** – communication over the network and Internet
- **Printing controlling** – takes control of **printers** connected and manages the printing process
- **User interface (UI)** – provides UI for the users to interact with the computer by commands or visual UI elements
- **Data security** – isolate apps, users and files to keep data secure (e. g. using file system / resource permissions)

Kernel

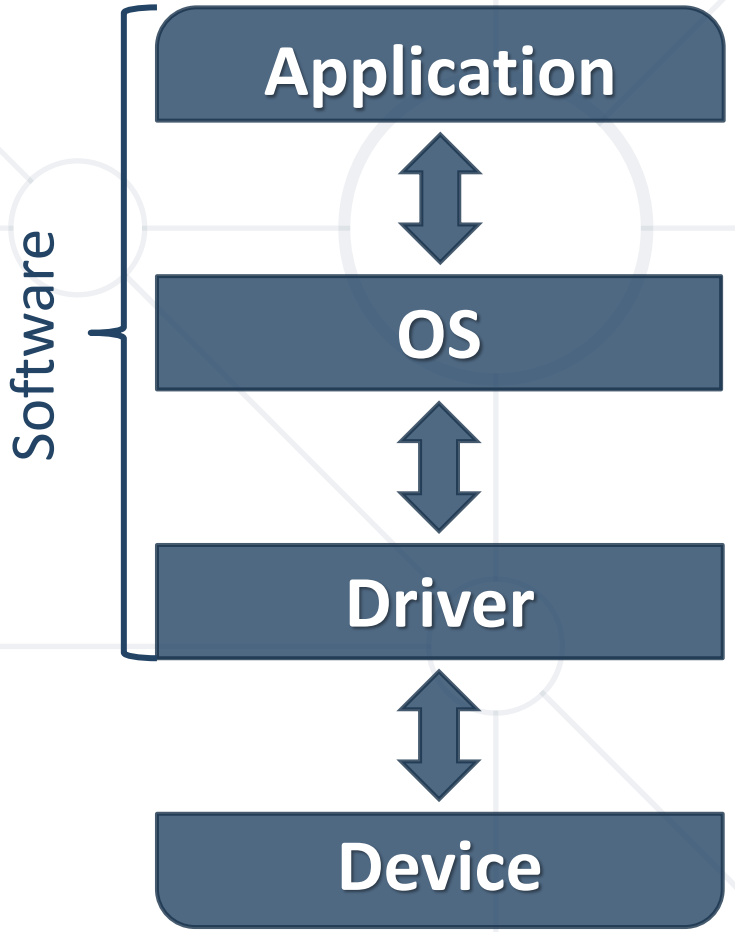
- **Kernel** - core component of the OS
- The OS "heart" – **bridges hardware** and **software** components
- **Facilitates communication** between different system components
- **Provides complete control** over the system
- **Always stays resident** in memory
- **Essential** for running any operating system



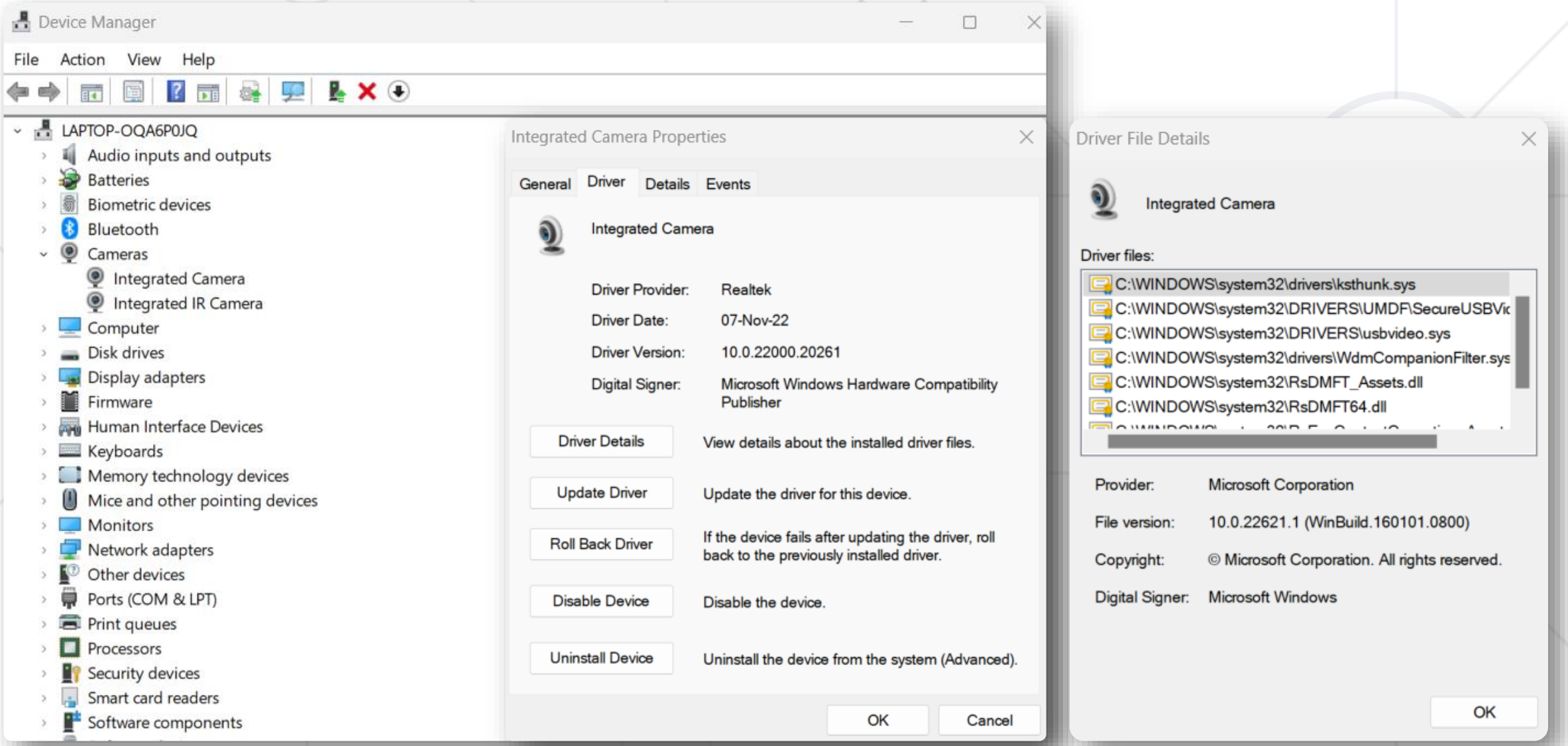


Drivers

- **Drivers** - set of **system programs** that enable hardware components to function
- Drivers **connect the OS and devices**
  - Enable **hardware components** or peripherals to operate properly
- Drivers are low-level software programs **without a user interface (UI)**
- All hardware components **require a driver** (e. g. disk drives, printers, keyboards)

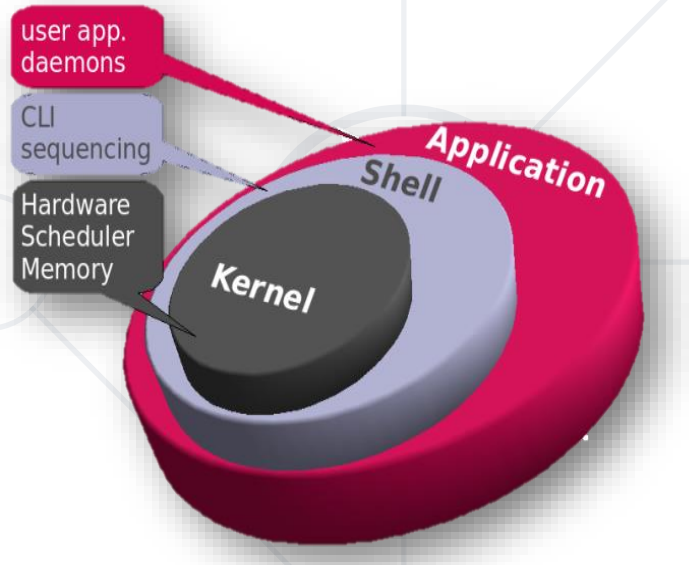


Drivers – Example



Shell

- **Shell** - user interface (UI) to the OS
  - Outermost layer of the operating system, located **between the kernel and the apps**
  - Provides a **UI and tools** to control processes, files, installed software, users, etc.



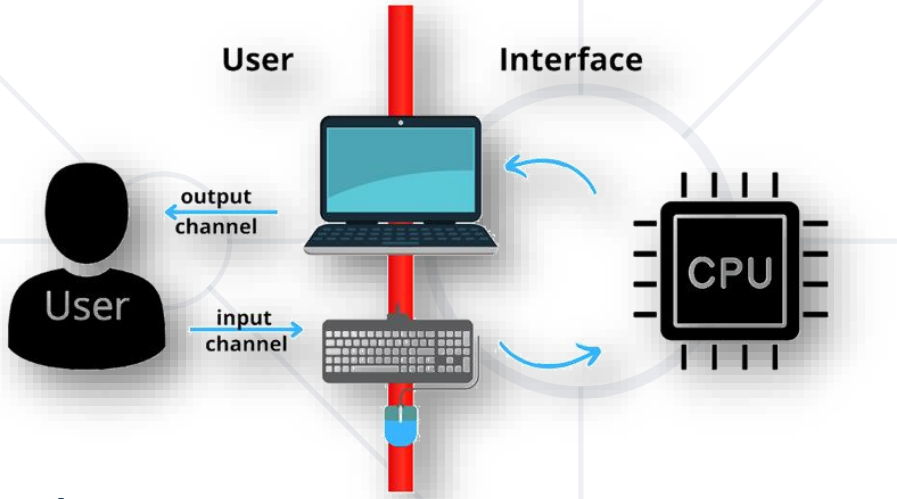
- Two types of shells:
  - **Command-line (CLI) shells** – require knowledge of commands, syntax, and concepts about the shell-specific scripting language (e. g. bash)
  - **Graphical (GUI) shells** – intuitive, easy to use (e. g. Windows Desktop)
- Most GUI-enabled OS provide also **CLI shells** for advanced users

Users in Operating Systems

- **Users** in the OS == individuals or entities who interact with the system by logging in and performing tasks
- A user often has a **user account** and is identified to the system by a **username**
- Users may have **privileges** over processes, folders and files, devices, services, network and other resources
  - Users are typically **isolated** from each other
- OS can be **single-user** (e. g. DOS) or **multi-user** (e. g. Linux, macOS, Windows)

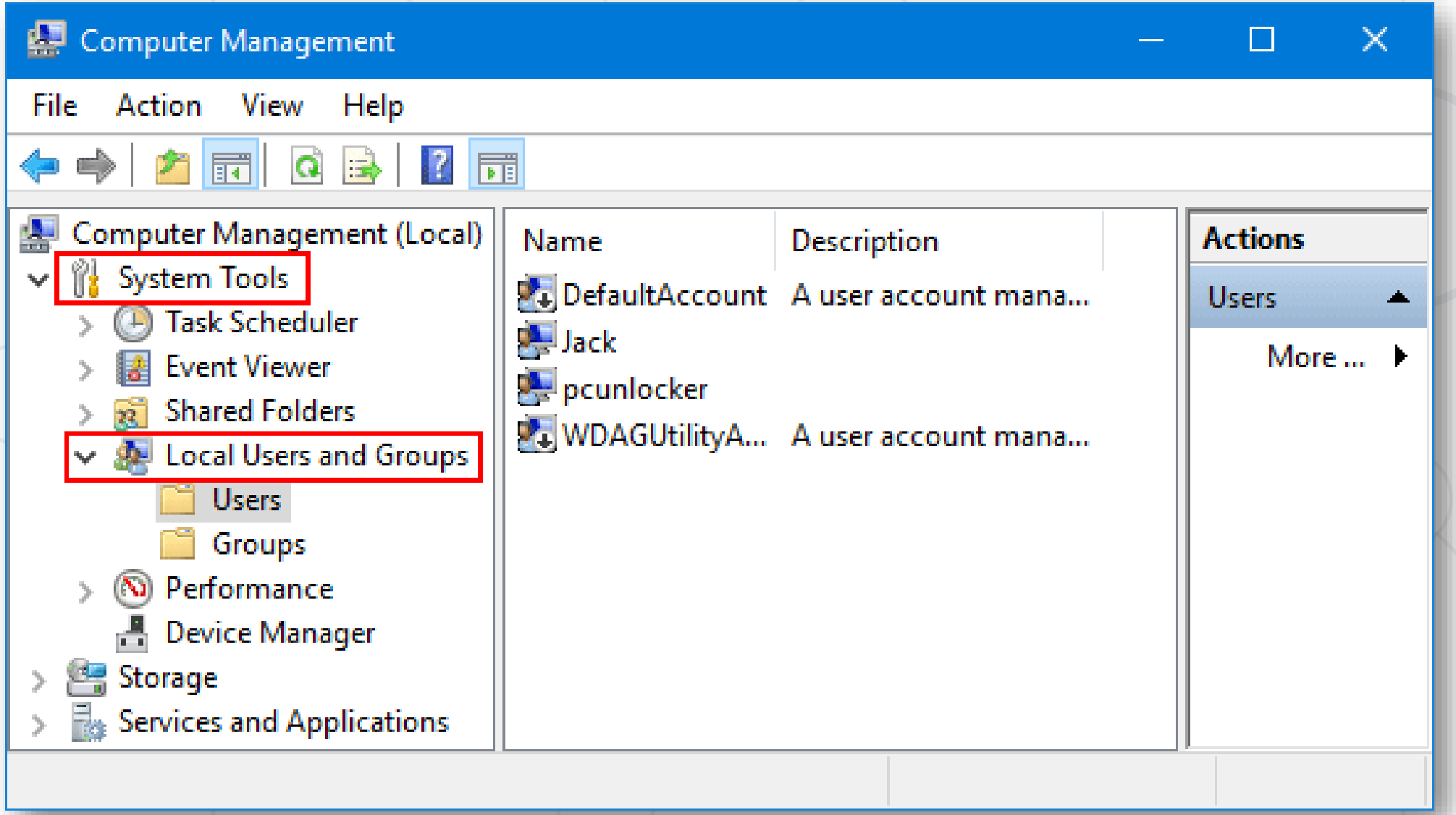
Users in OS

- **User accounts** allow access to a system's resources
- **Authentication** is the process of verifying a user's identity
  - Through **credentials** (like passwords / keys)



- **Authorization** determines what resources a user can access based on their authenticated identity
- User accounts in the OS are **important** for **accounting**, **security**, **logging**, and **resource management**

Users in the OS – Examples



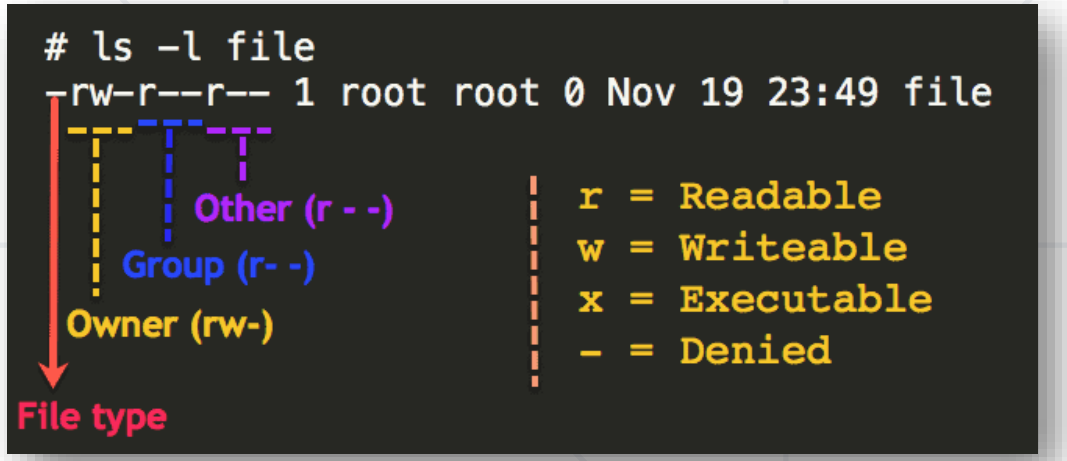
Authentication vs. Authorization

- Authentication **verifies the identity** of a user or service
- Authentication answers the question:
  - **Who are you?**

- Authorization determines the **user's access rights**
- Authorization answers the question:
  - **What are you allowed to do?**

User Permissions

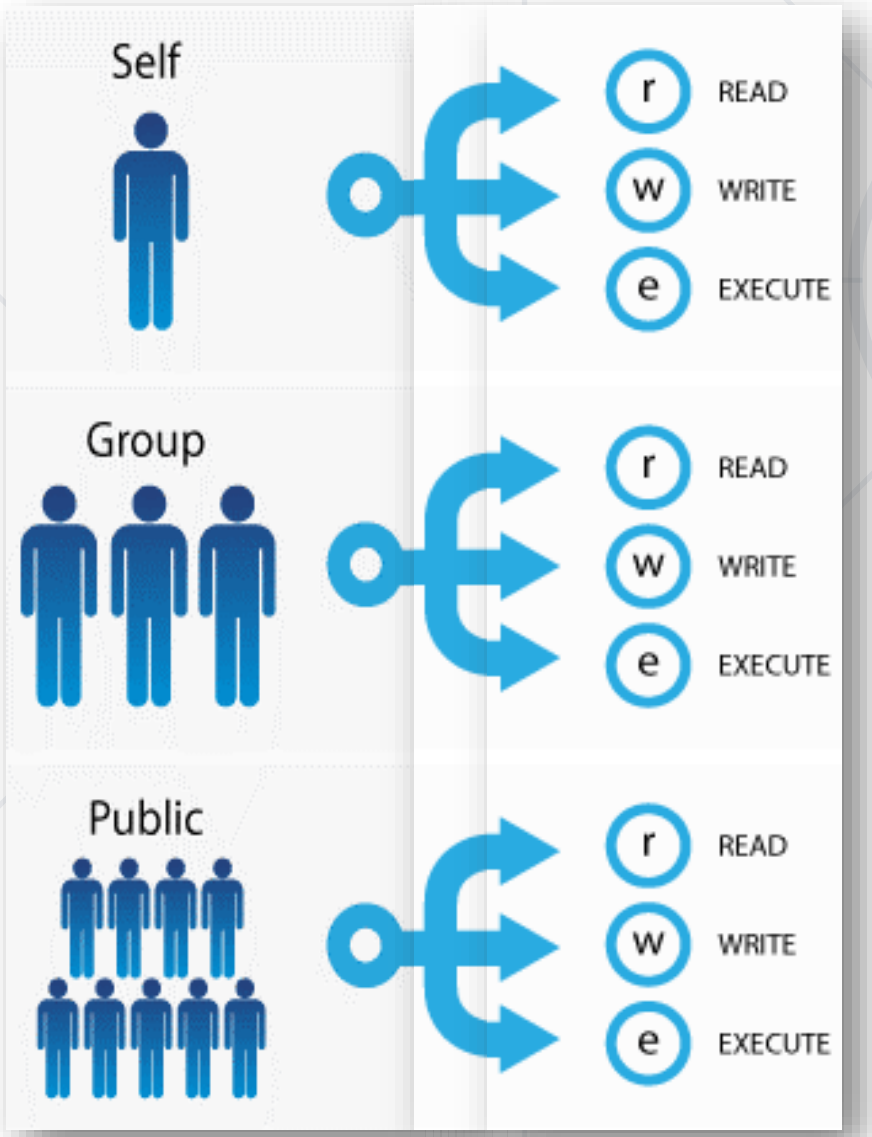
- OS **controls the use of system and network resources**
  - Through **authentication** and **authorization**
  - Based on user **permissions** over resources (e. g. file permissions)
- The OS **determines** if an **authenticated user** has the **correct permissions** to access a resource
  - Using built-in authorization and access control technologies



User Roles (Groups)

- **User roles** (**groups**) are **permission sets** that control access to resources (files, folders, processes, services)
  - Simplify permission assignments, e. g. in a hosting company, all customers may use the group "**web**"
- Each user account may have multiple **roles**
- Examples of user roles in MS Windows: **Administrator**, **User**, **Power User**, **Guest**
- Examples of user groups in Linux: **root**, **user**, **nobody**

Access Permissions in OS



- **Access permissions** determine a user's ability to perform a specific action, or access a feature or object
- Set access permissions to specify which **users**, **groups**, or **roles** can **access** your **content**
- The most common permissions are **read**, **write** and **execute**

Processes in OS

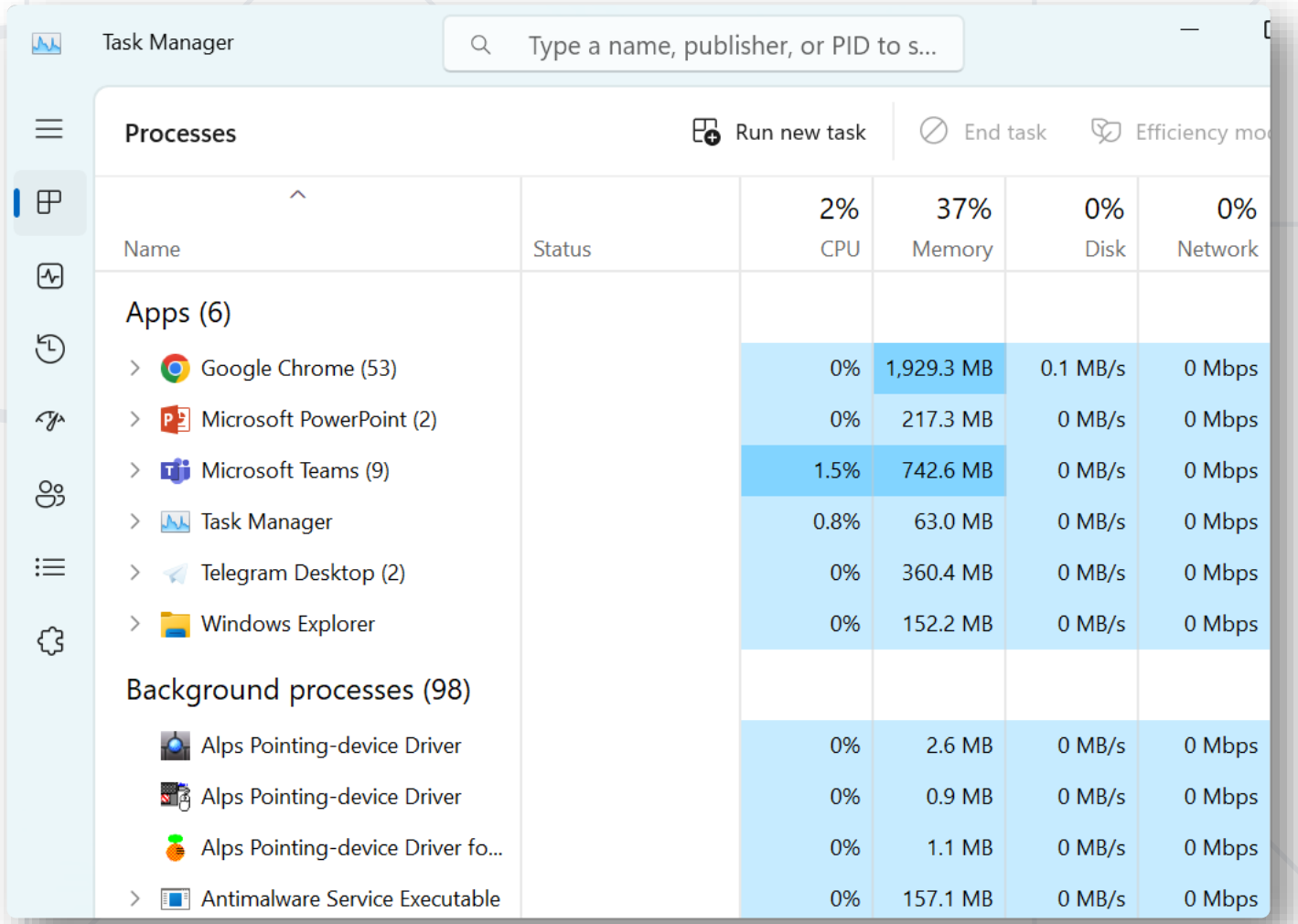
- A **process** is a **program in action** (a running app)
  - Consume CPU time, RAM memory, file handles and other OS resources
- It's the basic unit of work in the operating system
- Unlike files, which are **passive**, processes are an **active entity**
- For example, when you open a browser to search the web, that's a process

Task Managers

- In OS, a **task manager** is a **system monitoring app**
  - View **processes**, **users**, consumed resources, etc.
  - View RAM, CPU, GPU, disk, network load
  - Start / terminate (kill) processes
- Examples:
  - **Windows Task Manager** in MS Windows
  - **top** and **htop** in Linux
  - **Activity Monitor** in macOS

Windows Task Manager

- Open the **Task Manager** in MS Windows:
  - [**Ctrl** + **Alt** + **Delete**] → select [**Task Manager**] from the menu
  - Right click on the task bar → [**Task Manager**]



Popular Operating Systems

Windows, Linux, macOS, Android, iOS

Most Popular Operating Systems

- **Five major** operating system
  - Microsoft Windows
  - Apple macOS
  - Google's Android OS
  - Apple iOS
  - Linux (open source)

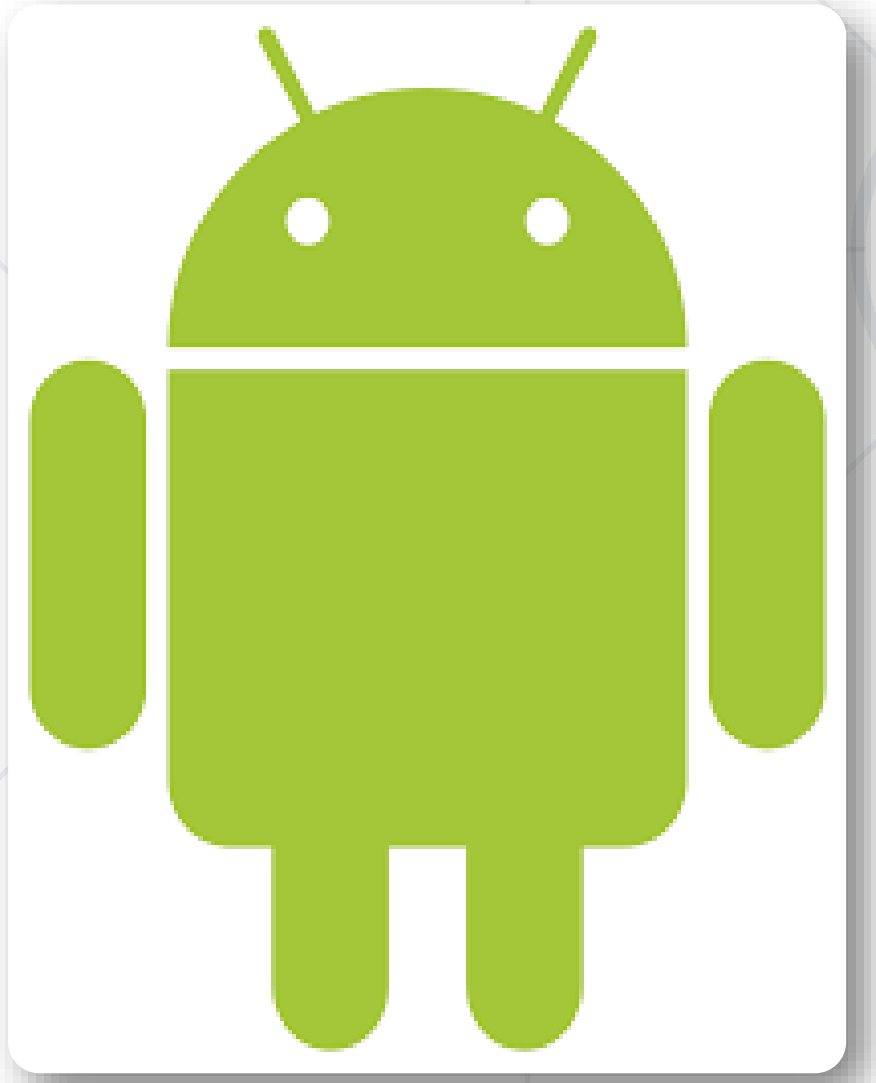
Microsoft Windows

- Proprietary OS, developed by **Microsoft**
- One of the **most popular** OS
- Typically **preinstalled** on new PC
- Several versions: Windows 95 / 98 / Vista, Windows 7 / 8 / 10 / 11
- Has been around since the 1980s
- Easy-to-use, intuitive GUI shell
- Many apps and games

Apple macOS

- **Apple** and **Macintosh** computers run on **macOS** and **OS X**
- Proprietary OS developed by Apple
- **macOS** is a **Unix-based** OS
- Released over 20 years ago
- In 2020, Apple began **transitioning** to its own 64-bit ARM-based Apple M CPU
- Apple M1 / M2 CPU: powerful and silent

Android OS

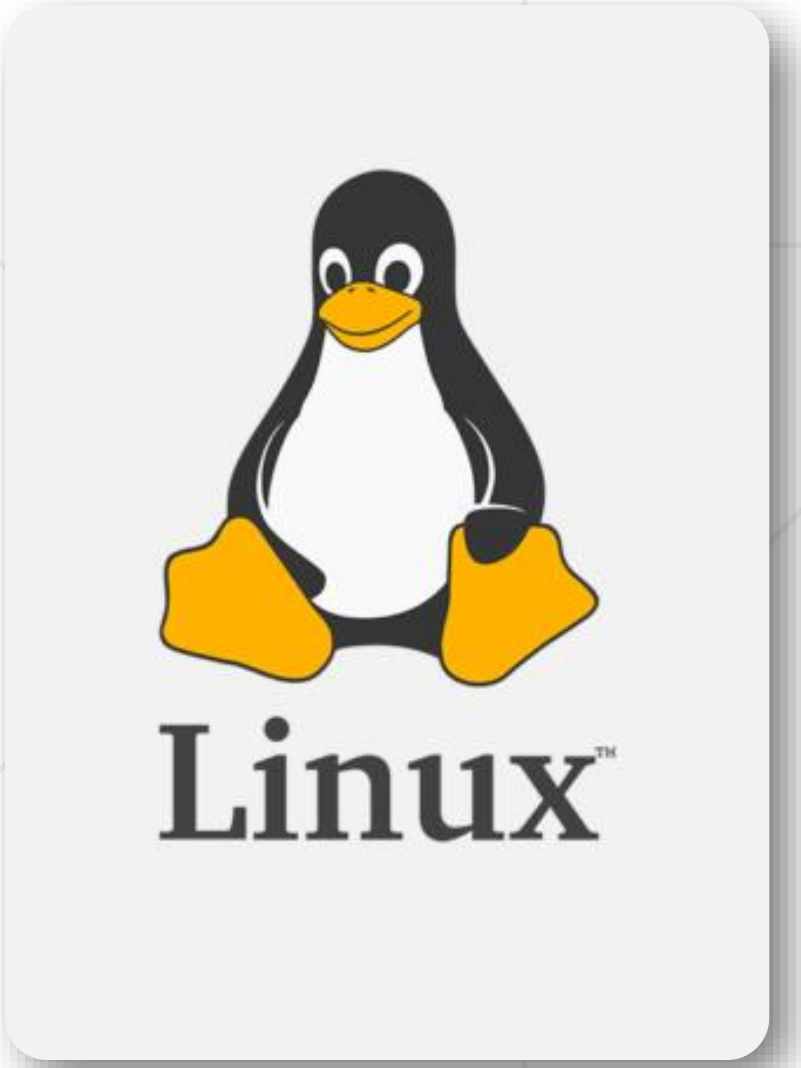


- Mobile OS, **designed** for **touchscreen** mobile devices
- **Based** on a **modified version** of the **Linux kernel** and other open-source software
- Core OS is called **Android Open-Source Project (AOSP)**
- Free and open-source software
- **Developed** and **maintained** by **Google**
- Many distributions (by Samsung, Xiaomi)

Apple iOS

- **Mobile OS**, developed by Apple
- Exclusively for its hardware devices: **iPhone**, **iPad** and **iPod Touch**
- Closed ecosystem, dominated by Apple
- iOS UI uses multi-touch gestures: **swipe**, **tap**, **pinch**, and **reverse pinch**
- iOS runs on **Apple hardware only**
- Might run on PC emulators (it's illegal!)

Linux



- **Linux** is Free **and open-source family** of operating systems
- Linux's **popularity** comes from its ease of customization and open license
- Offers **CLI shell** and many **GUI desktops**
- Many **distributions**: Ubuntu, CentOS, Debian, Mint, openSUSE, Alpine, …
- It offers a **variety of options** for those who understand how to use it

Virtual Machines & Containers

Remote Instances & Emulators

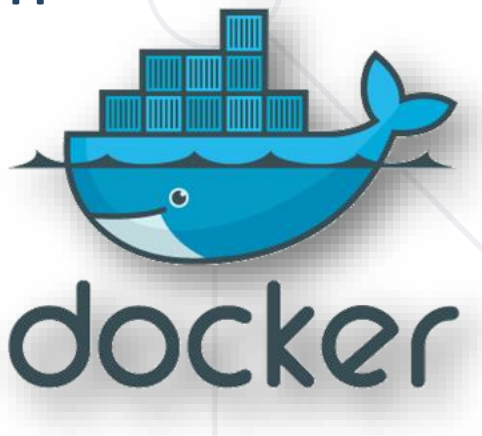
Virtual Machines (VM)

- A **virtual machine (VM)** is a software-based computer resource, used to run an OS inside another OS
  - Digital version of a physical computer that **can run programs** and **OS**, **store data**, **connect to networks**, and other computing functions

- **Virtualization** == running a **virtual machine** (VM) / virtual environment inside a physical hardware system
  - E. g. run Android VM or Linux inside a Windows host
  - Storage, networking, desktops can also be virtual

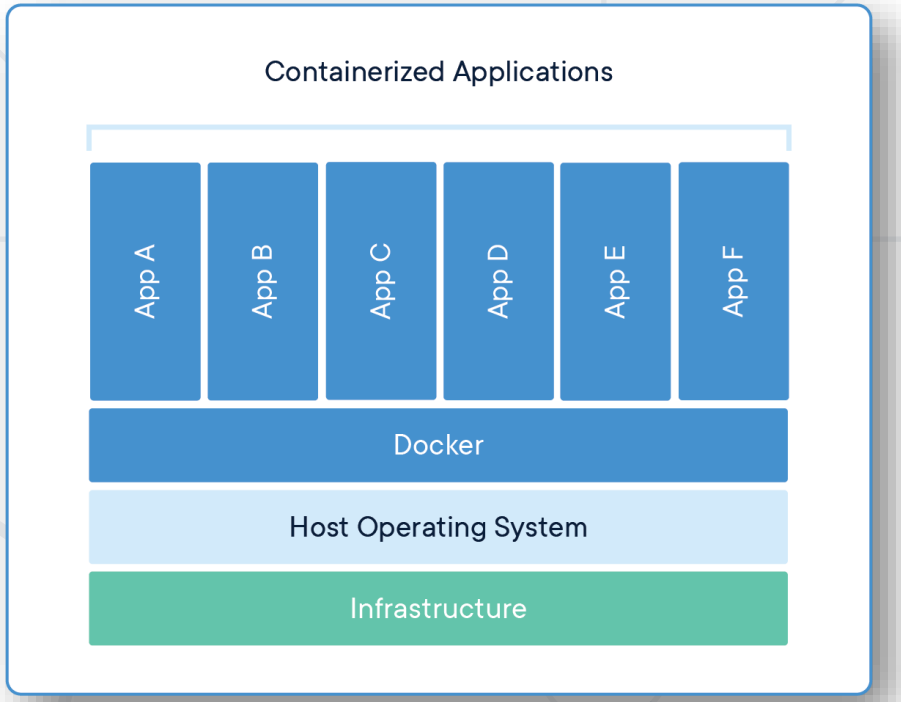
Containers and Docker

- **Container image** == software, packaged with its dependencies, designed to run in a virtual environment (like Docker)
  - E. g. WordPress instance (Linux + PHP + Apache + WordPress)
  - Simplified installation, configuration and deployment
  - **Lightweight** – containers use shared OS kernel with the host
- **Docker** is the most popular containerization platform
  - Runs **containers** from local **image** or downloaded from the **Docker Hub** online repository
  - Open-source, runs on Linux, Windows, Mac



Docker Containers

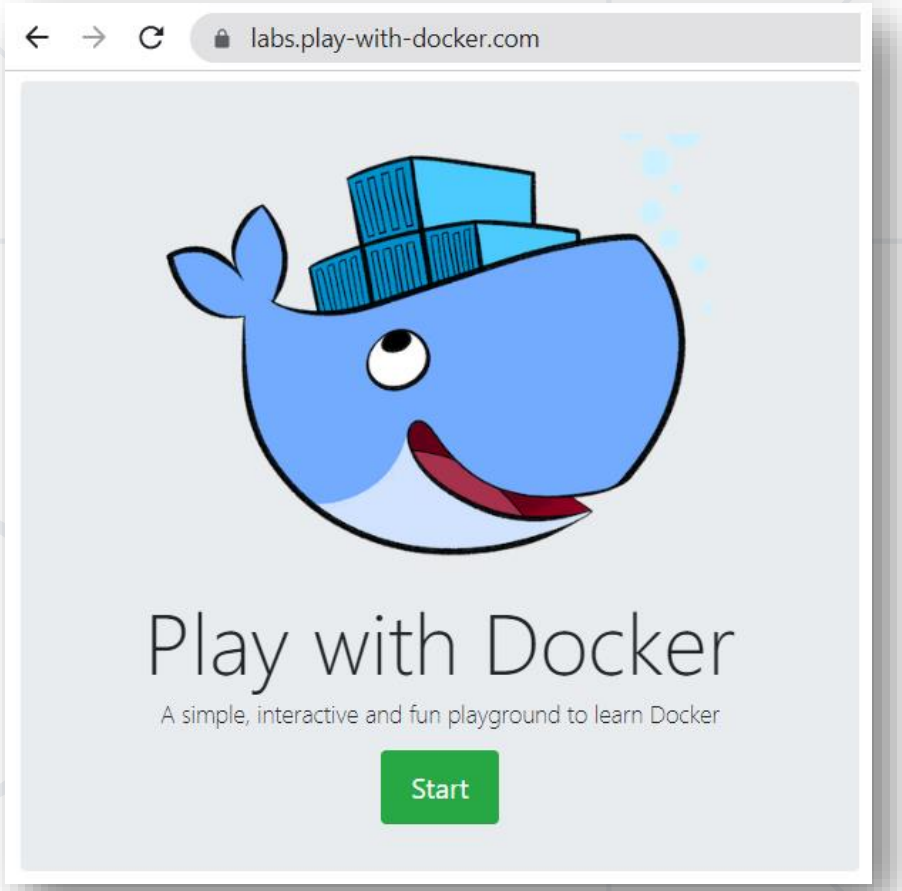
- A Docker **container image** is a lightweight, standalone executable package of software
  - Contains everything needed to run an app: code, runtime, libraries, tools, and settings
- **Container** == running Docker image
  - App, running inside the Docker Engine
- Containers provide fast and simple way to **run apps, without installing them** on the host OS
- Containers are **isolated** from the host and other containers → **security**



Remote VM Instances and Docker Playground

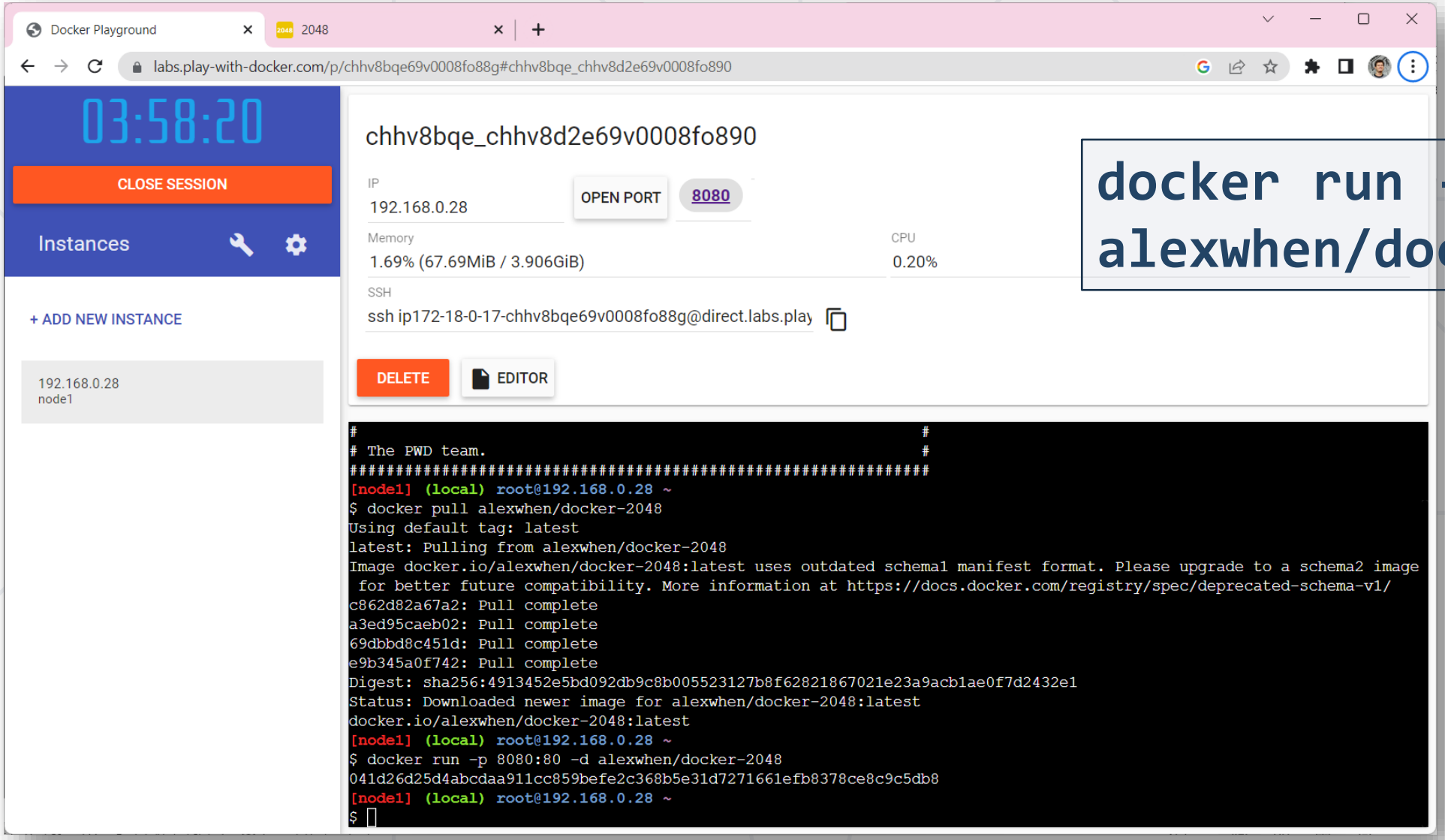
Software University

- Containers allow for **customizable** and **replicable instances** of an application
  - Without interfering with anything else on a user's system (no conflicts)
- **Docker Playground** is an interactive and fun way to learn Docker
  - Provides free Linux + Docker VMs
  - Accessible for 4 hours, for learning
  - https://labs.play-with-docker.com



Docker Playground – Live Demo

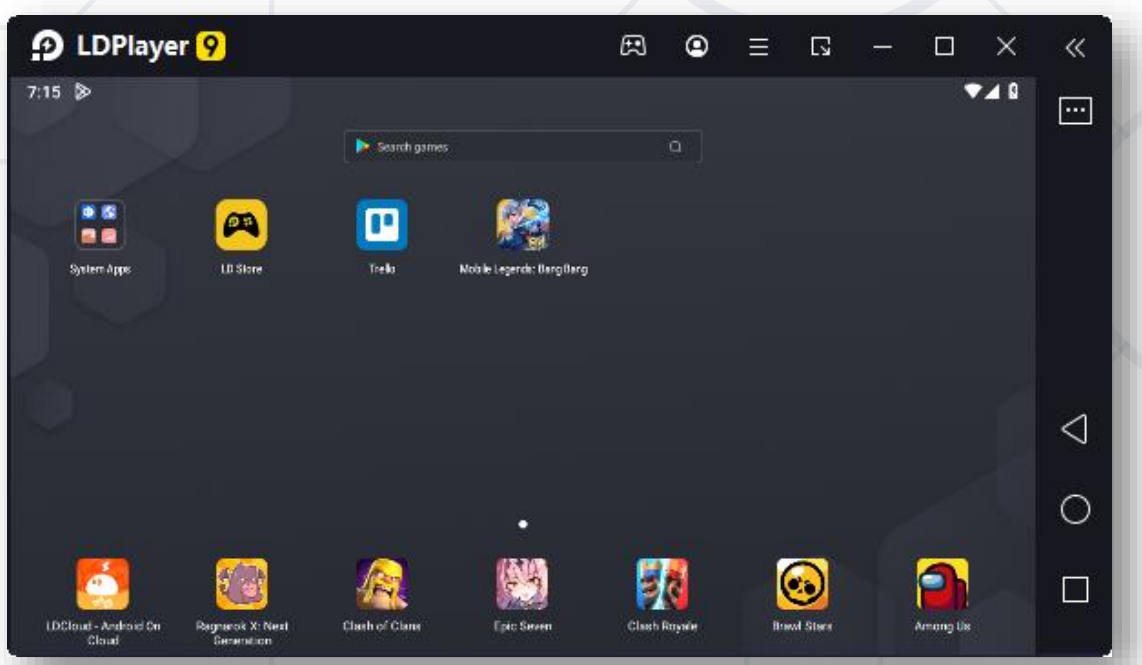
Software University

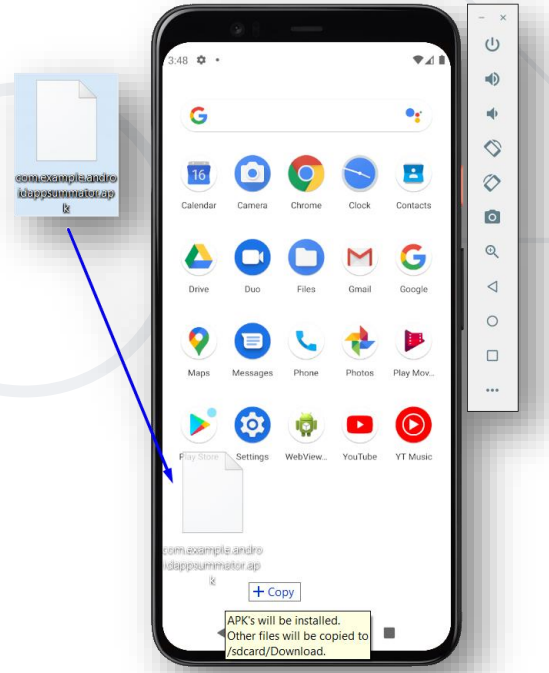



```
docker run -p 8080:80
alexwhen/docker-2048
```

Device Emulators

- **Device emulators** run Android / iOS / other OS in a virtual machines (VM) and simulate device functions (e. g. rotation)
  - **BlueStacks**, **LDPlayer**, **Android Emulator** – run Android apps in Windows and simulate mobile devices





BrowserStack – App & Browser Testing

- BrowserStack – **manual** and **automated** online mobile testing for Web sites and mobile apps
  - Test on **remote physical devices**: iPhone, iPad, Samsung, Xiaomi, Google smartphones / tablets
  - Modern devices, modern Web browsers
  - Android, iOS, Windows, macOS
- **BrowserStack Live** offers **3000+ device-browser-OS** combinations for testing

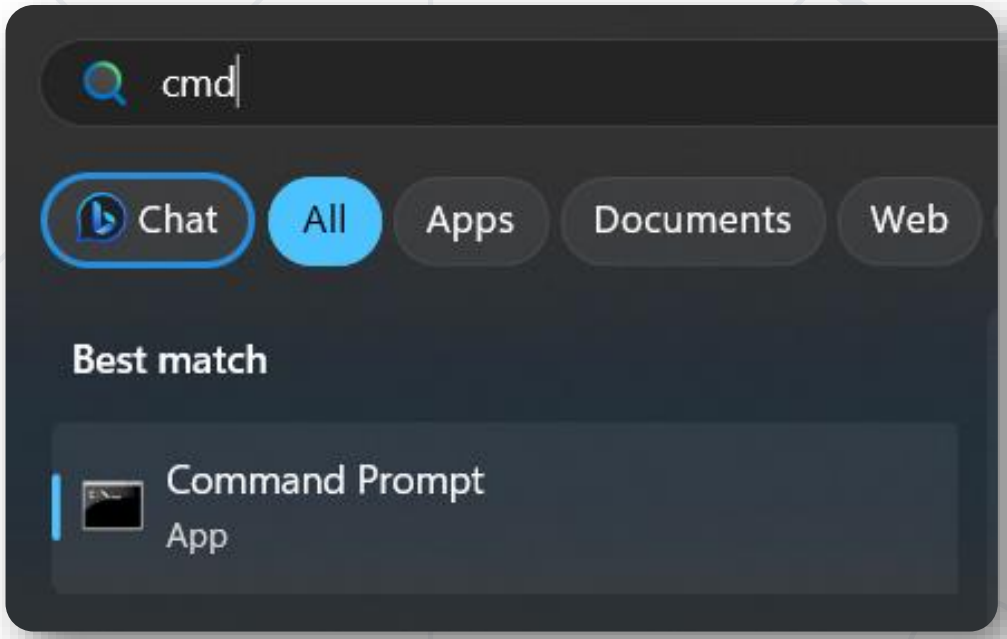
Shell & Shell Commands

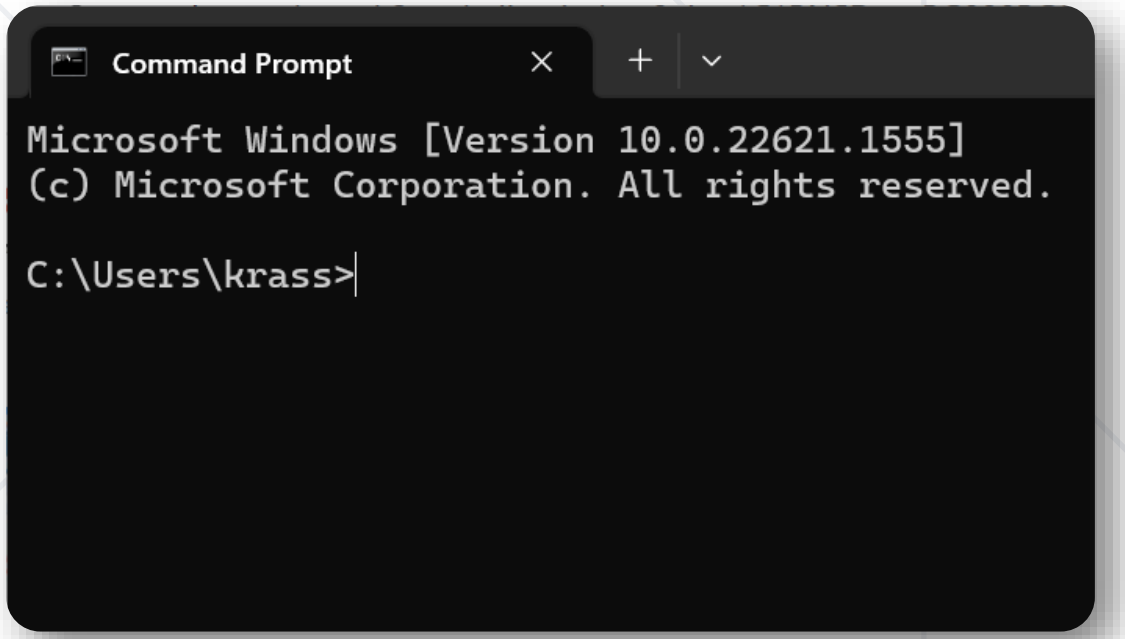
Shell Command Execution on Linux and Windows

Opening the CLI Shell in MS Windows

Software University

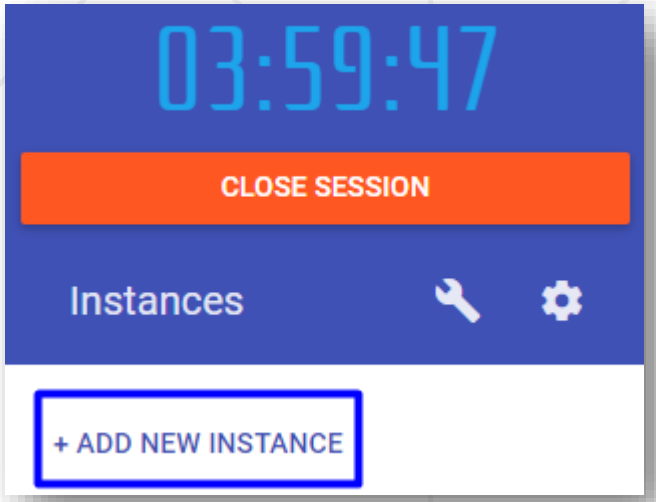
1. Click [**Start**] -> [**Run**] or press [**Windows + R**] key
2. Type "**`cmd`**"
3. Click on [**Command Prompt**]





Linux Shell in Docker Playground

- Starting a **Docker Playground** session
  - Open **Docker Playground**, register and log in
  - Press **[Start]** and add a **new instance**
  - Now you have a **Linux VM + Docker environment** to experiment with




```
################################################################################
#                              WARNING!!!!                                     #
# This is a sandbox environment. Using personal credentials                    #
# is HIGHLY! discouraged. Any consequences of doing so are                     #
# completely the user's responsibilites.                                       #
#                                                                              #
# The PWD team.                                                                #
################################################################################
[node1] (local) root@192.168.0.13 ~
$
```

Commands: ls & dir

- **ls** list files and directories in Linux / UNIX / macOS

```
user@host:~$ ls
```

```
user@host:~$ ls -al
```

```
nakov@Nakov-Laptop-HP:~$ ls -al
total 64
drwxr-xr-x 9 nakov nakov 4096 May 16 19:38 .
drwxr-xr-x 3 root  root  4096 Dec 11  2021 ..
-rw------- 1 nakov nakov 2520 May 17 01:04 .bash_history
-rw-r--r-- 1 nakov nakov  220 Dec 11  2021 .bash_logout
-rw-r--r-- 1 nakov nakov 3771 Dec 11  2021 .bashrc
drwx------ 3 nakov nakov 4096 Mar 27 12:56 .cache
drwx------ 5 nakov nakov 4096 Mar 27 12:56 .config
```

- **dir** lists the files and folders in Windows

```
C:\Users\nakov> dir
```

```
C:\Users\nakov>dir
 Volume in drive C is Nakov's SSD
 Volume Serial Number is B295-4B6D

 Directory of C:\Users\nakov

09-May-23  14:32    <DIR>          .
29-Sep-22  18:44    <DIR>          ..
11-May-23  19:23    <DIR>          .android
28-Apr-23  14:58    <DIR>          .azure
16-May-23  21:35             1 112 .bash_history
07-Mar-23  21:55    <DIR>          .cache
```

Commands: cd

- **cd** changes the current working directory in Linux

```
user@host:~$ cd /home
user@host:~/home$ ls -al
```

```
nakov@Nakov-Laptop-HP:~$ cd /home
nakov@Nakov-Laptop-HP:/home$ ls -al
total 12
drwxr-xr-x  3 root  root  4096 Dec 11  2021 .
drwxr-xr-x 19 root  root  4096 May 17 11:16 ..
drwxr-xr-x  9 nakov nakov 4096 May 16 19:38 nakov
```

```
user@host:~/home$ cd ..
user@host:~/$ ls -al
```

- **cd** works the same way in Windows

```
C:\Users\nakov> cd ..
C:\Users> dir
```

```
C:\Users>dir
 Volume in drive C is Nakov's SSD
 Volume Serial Number is B295-4B6D

 Directory of C:\Users

29-Sep-22  18:44    <DIR>          .
29-Sep-22  18:47    <DIR>          defaultuser100000
09-May-23  14:32    <DIR>          nakov
29-Sep-22  21:43    <DIR>          Public
18-Jan-22  15:13    <DIR>          svetl
```

Commands: pwd / cd

- **pwd** prints the current working directory in Linux

```
user@host:~$ pwd
```

```
nakov@Nakov-Laptop-HP:~$ pwd
/home/nakov
```

- **cd** works the same way in Windows

```
C:\Users\nakov> cd
```

```
C:\Users\nakov>cd
C:\Users\nakov
```

Commands: echo and cat / echo and type

- **echo '...' > filename** prints a text to a file in Linux
- **cat** displays the content of given file

```
echo 'Hi Linux' > hi.txt
cat hi.txt
```

```
nakov@Nakov-Laptop-HP:~$ echo 'Hi Linux' > hi.txt
nakov@Nakov-Laptop-HP:~$ cat hi.txt
Hi Linux
```

- **echo ... > filename** prints a text to a file in Windows
- **type** displays the content of given file

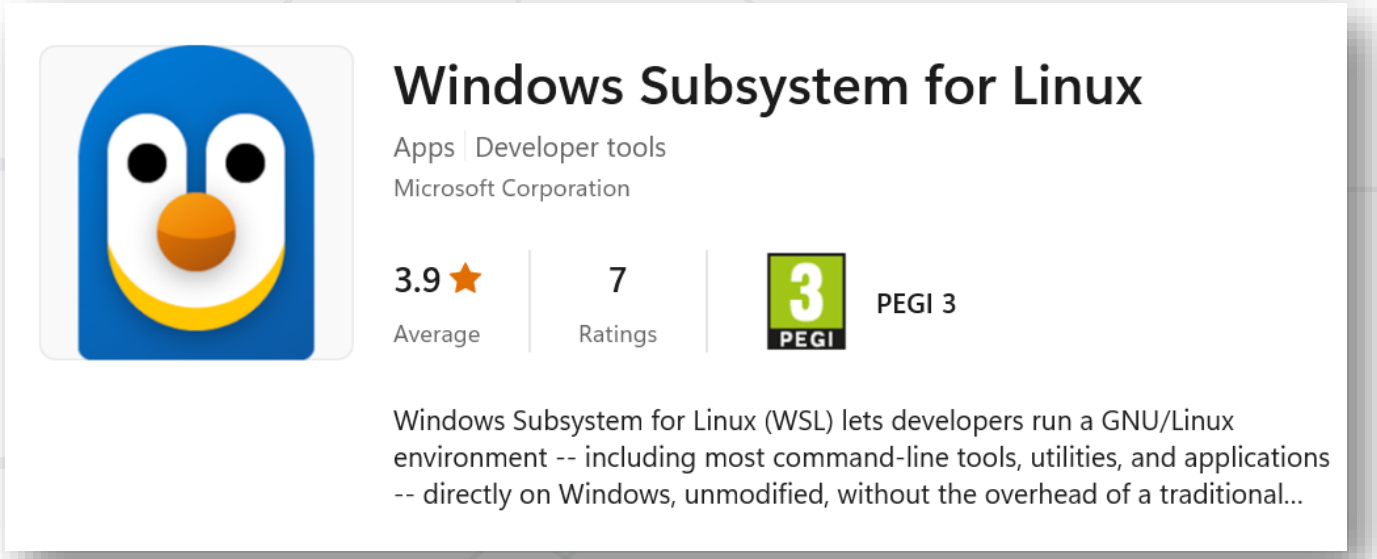
```
echo Hi Windows > hi.txt
type hi.txt
```

```
C:\Users\nakov>echo Hi Windows > hi.txt
C:\Users\nakov>type hi.txt
Hi Windows
```

Can I Run Linux Commands on Windows PC?

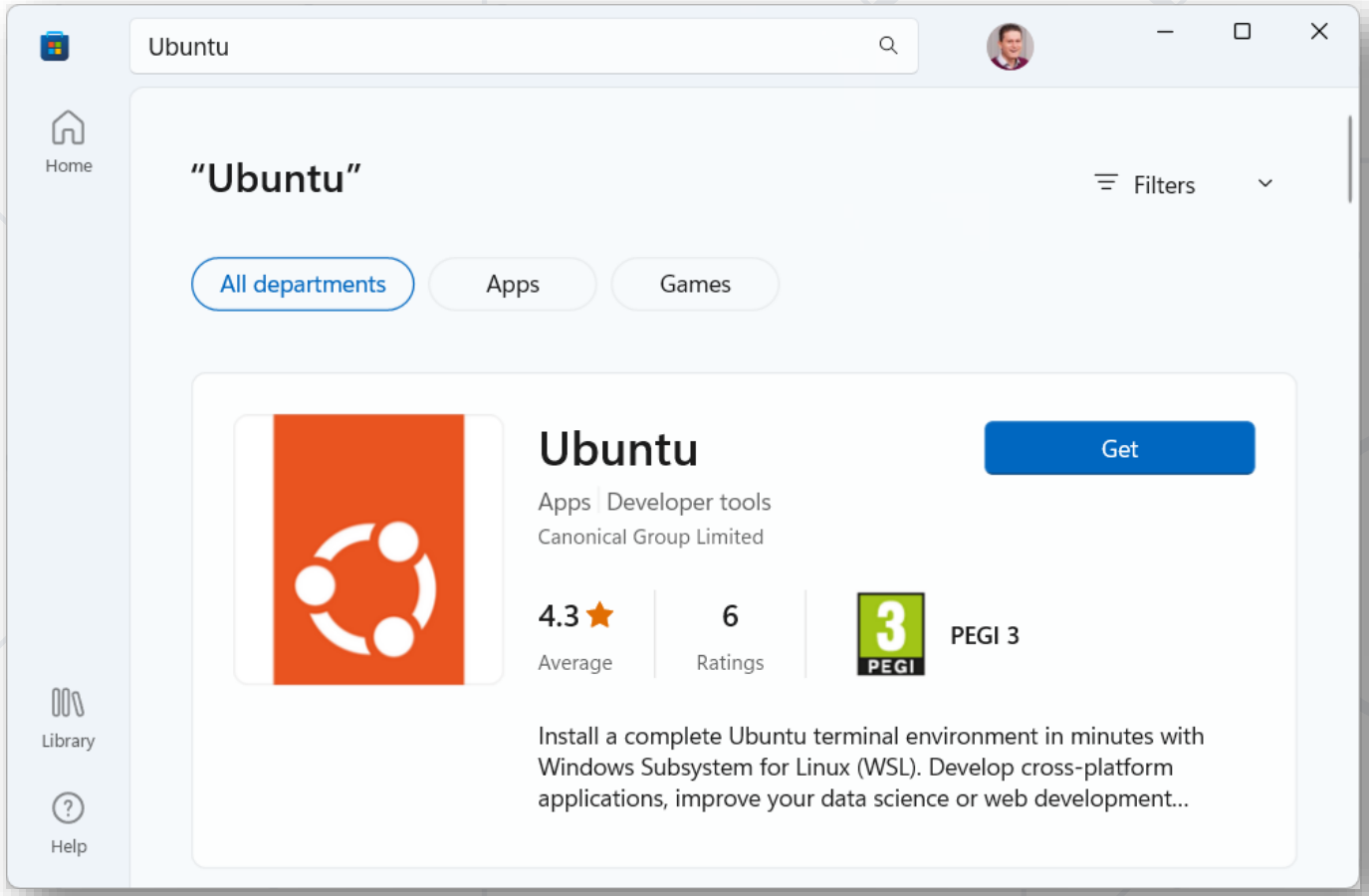
Software University

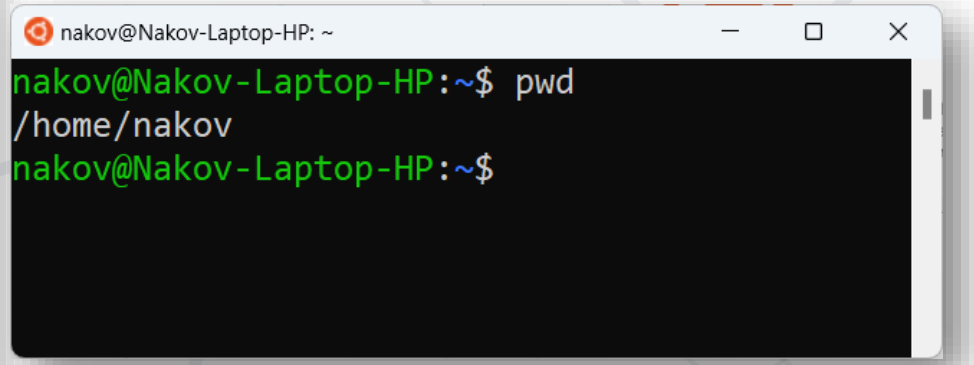
- You can **run Linux in Windows** through a **virtual machine**
  - E. g. Ubuntu Linux in Virtual Box
- **Windows Subsystem for Linux** (WSL)



Install WSL and Ubuntu Linux in Windows 11

Software University

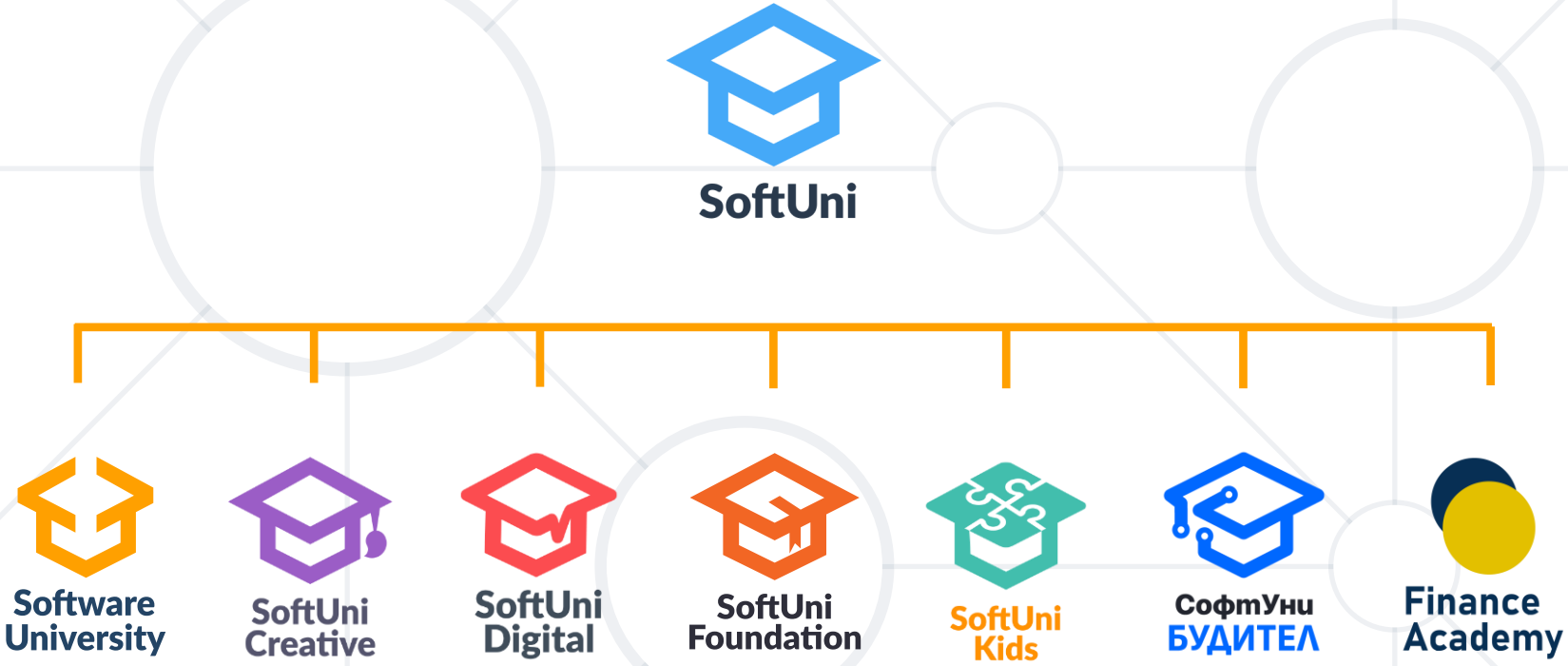




Summary

- **Operating Systems (OS)** manage processes, users, files and other resources
- **OS Examples**: Windows, macOS, Linux, Android, iOS
- **Virtual machine** (VM) == OS inside another OS
- **Container** == app image, running in Docker
- **Shell commands** == execute commands from the console (Linux / Windows shell)

Questions?



SoftUni Diamond Partners

Software University

SmartIT

Coca-Cola HBC Bulgaria

SUPER HOSTING .BG

createX

Postbank

Решения за твоето утре

DRAFT KINGS

VIVACOM

AMBITIONED

INDEAVR

Serving the high achievers

SOFTWARE GROUP

PHAR VISION

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg, about.softuni.bg
- Software University Foundation
  - softuni.foundation
- Software University @ Facebook
  - facebook.com/SoftwareUniversity

License

- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © SoftUni – https://about.softuni.bg/
- © Software University – https://softuni.bg