# Introduction

Lecture 1

# Welcome

to the *Internal Strcuture of Operating* class

## You will learn, understand and experiment

- how operating systems work
- how basic POSIX (Linux) commands work
- how to use the Linux **command line**
- the history of Linux
- how use the development tools

## We expect

- to come to class
- ask a lot of questions

# Our team

## Lectures

- Alexandru Radovici (CD)
- Ioana Culic (AC)

## Labs

- Ana Gabriela Almăjanu (CD)
- Irina Bradu (AC)
- Andrei Dragtă (AC)
- Andreea Ocănoaia (CD)
- Irina Niță (CD)
- Cristian Paris (CD)
- Alexandra Văduva (CD)
- Alexandru Radovici (CD)

# Outline

## Lectures

- 12 lectures

## Labs

- 12 labs

# Homework

- 4 homework assignments

> *that will help you prepare for the hands on tests*

# Grading

| Part | Description | Points |
|---|---|---|
| Lecture tests | You will have a test at every class with subjects from the previous class. | 0.5p |
| Homework | You will have 4 homework assignments to exercise you for the hands-on tests | 1p |
| Lab activity | Your lab activity will be graded. | 0.5p |
| Hands-on midterm | You will have a practical test covering the first 9 weeks (a Saturday in December) | 4p |
| Theoretical exam | You will have a quiz on Moodle with French grading (in the exams period). | 2.5p |
| Hands-on exam | You will have a practical test covering all the subjects (in the exams period). | 2.5p |
| **Total** | *You will need at least **2.5p during the semester and 5 points in total** to pass* | **11p** |

# Notation

- *name* - this is a name
- **important** - this is important
- `command` - this is a command or a part of a source code

Code

```
$ command arguments
this is what the command wrote
$ command2 arguments
```

> `$` sigifies that the user can write a command, but is **not part of the command**.

*If you can’t explain it simply, you don’t understand it well enough*

Albert Einstein

# Bibliography


1. **Brian Ward**, *How LINUX Works*, 3rd Edition, No Starch Press, 2021
2. **Razvan Deaconescu, Razbvan Rughinis, Mihai Carabas, Alexandru Radovici**, *Utilizarea Sistemelor de Operare*, Printech 2021, download

# Operating System

the purpose of an OS

# Operating System

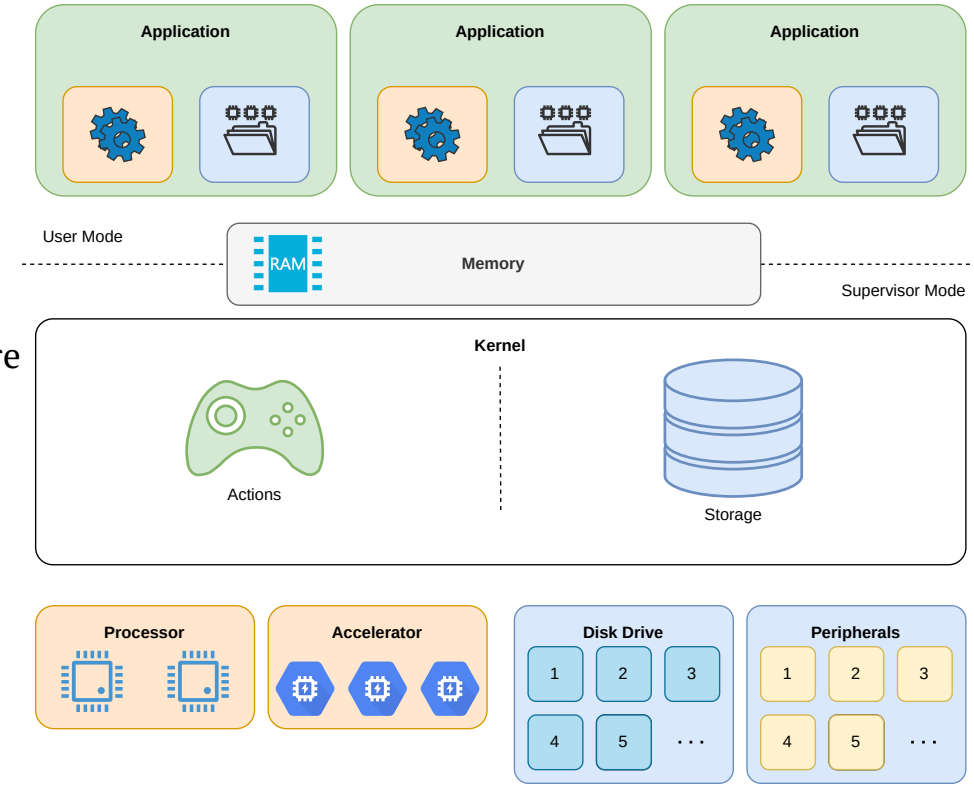
the main role

## Allow Portability

- provides a hardware independent API
- applications should run on any hardware

## Resources Management and Isolation

- allow applications to access resources
- prevent applications from accessing hardware directly
- isolate applications



# Desktop and Server Operating Systems

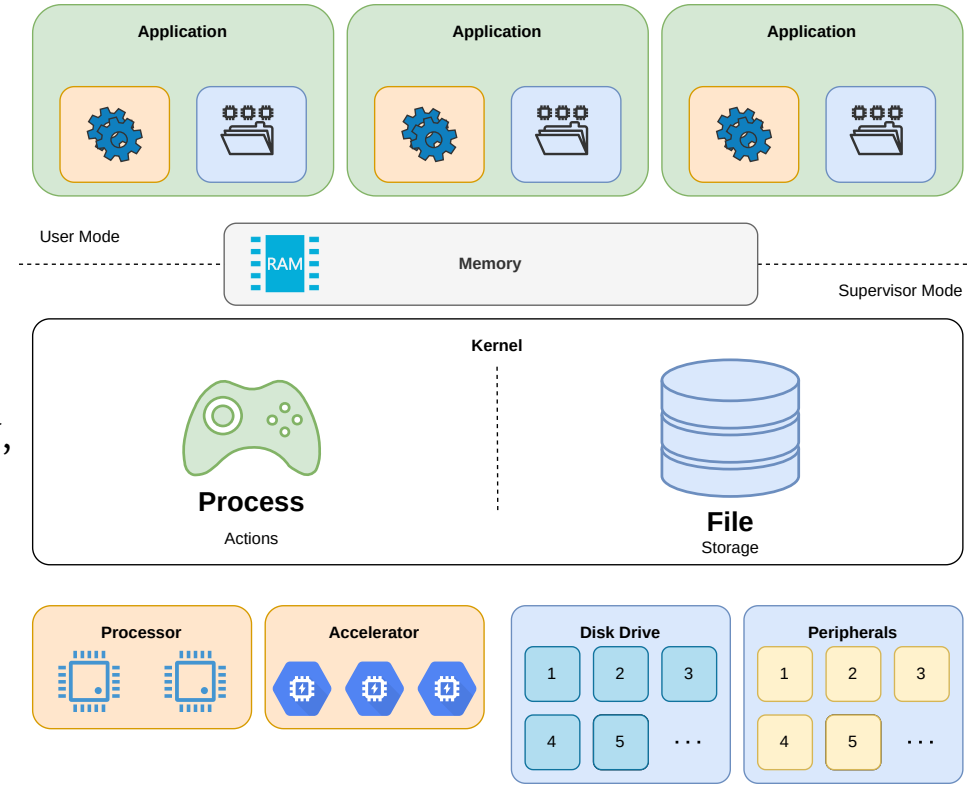
abstractions

## Actions

- *Applications*
- use the *Processor* and *Accelerators* (GPU, Neural Engine, etc)

## Data

- everything is a file
- peripherals are viewed as files (*POSIX*)
  - `/dev/input/keyboard` - keyboard
  - `/dev/fb` - screen (framebuffer)
  - `/dev/sda` - Disk Drive A (first)



# Embedded Operating Systems

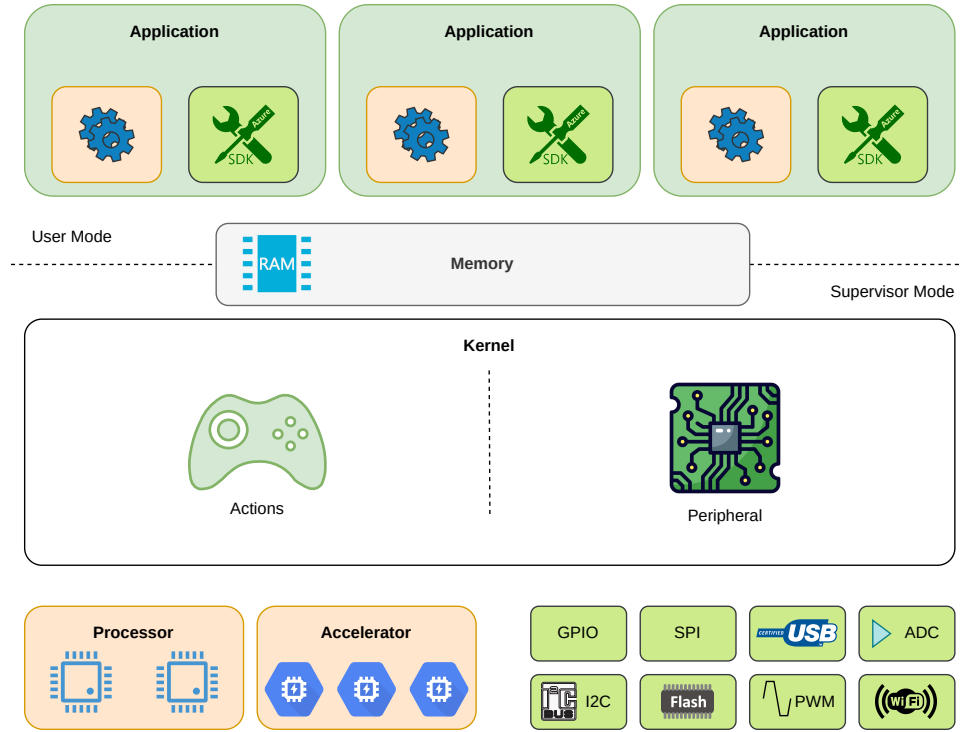
## Actions

- Simple *applications*
- use the *Processor* and *Accelerators* (Crypto Engines, Neural Engine, etc)

## Peripheral

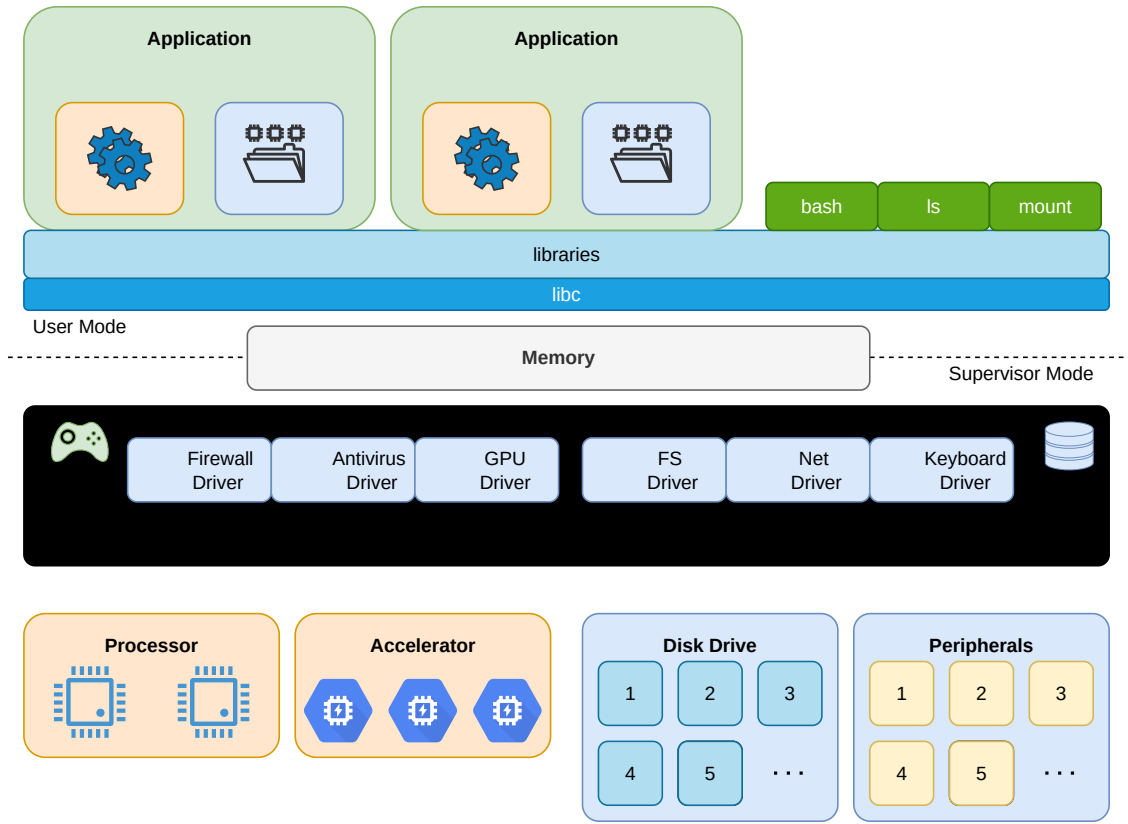
- provide a hardware independent API
- prevent processes from accessing the peripheral

*usually* the applications and the kernel are compiled together into a **single binary**



# The OS Stack

kernel / libc / libraries / basic tools (commands) / applications

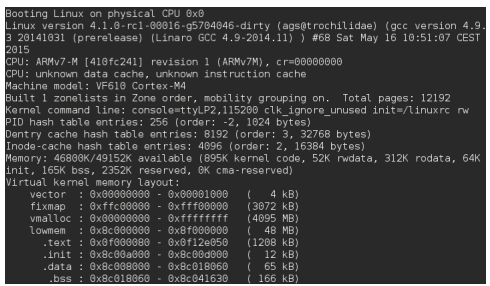


# Application Examples

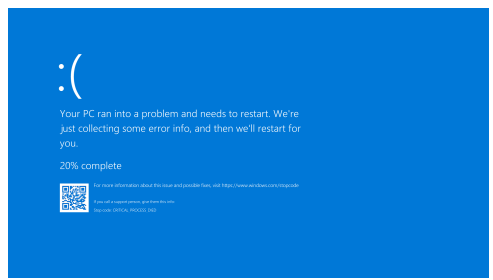
| What it does | Windows | Linux | macOS |
| --- | --- | --- | --- |
| Desktop | `explorer.exe` | `nautilus` | `Finder` |
| Applications | `taskbar.exe` | `gnome-shell` | `Dock` |
| Files | `explorer.exe` | `nautilus` | `Finder` |
| Settings | `start ms-settings:` | `gnome-control-center` | `System Settings` |
| Commands | `cmd.exe` or `powershell.exe` | `bash` or `zsh` | `bash` or `zsh` |
| Documents | `powerpoint.exe` | `libreoffice` | `Pages` |
| Edit Code | `code.exe` | `code` | `code` |

# Where can we see the OS

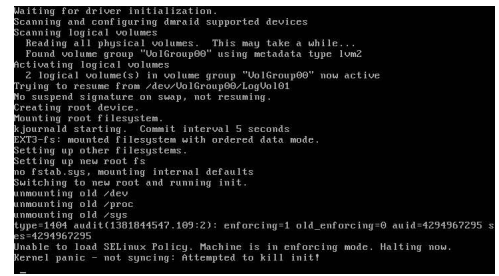
## During Boot




## Blue Screen (BSoD)

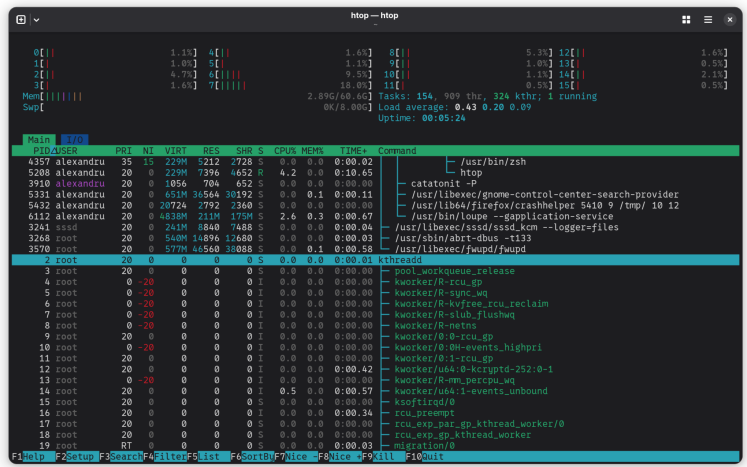



## Kernel Panic




# The Kernel

- is a collection of applications that run in priviliged mode
- one *main application* and several *utilities apps*




`htop`

# Virtual Machine

*run an OS within an application*

# The Idea

*run an operating system within an application*

Why?

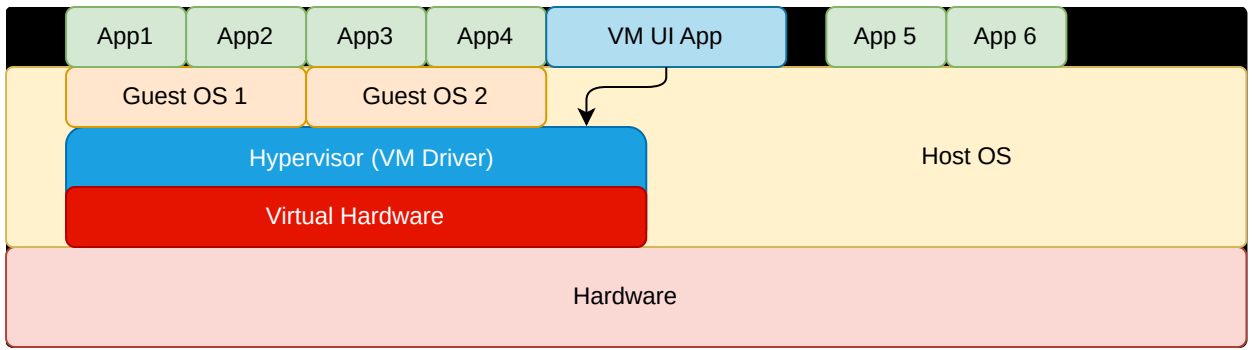
- Debug operating systems
- Run software that is not compatible with the OS your computer runs
- Securely share a server

Virtualization Software

- VirtualBox
- VMWare
- QEMU

# Simulation / Emulation

simulate all the hardware - very slow

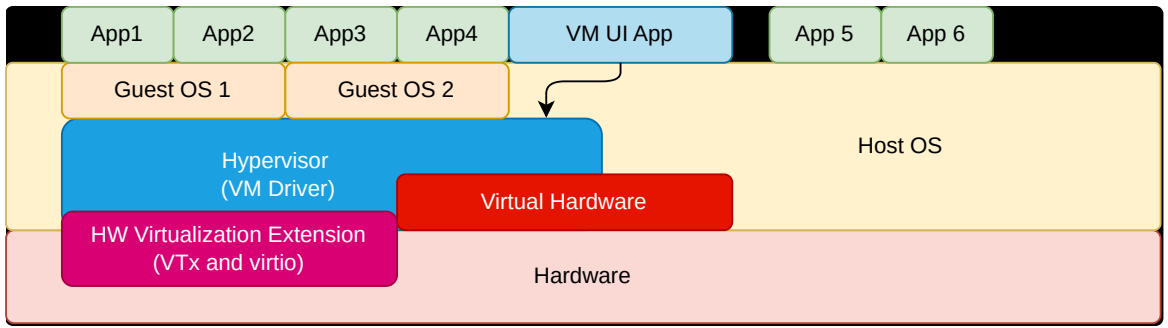



QEMU

- is able to simulate most of the available architectures
  - most used `x86`, `AMD64`, `arm`, `aarch64`, `powerpc`, `risc-v`

# Virtualization

use a part of the avalable hardware, emulates as if the system was running alone on the hardware




- Requires VT-x (Intel), … (AMD) and … (ARM) enabled
- The guest OS has to be built for the same CPU architecture as the host OS
  - *Apple Silicon* is ARM64 ( `aarch64` )

> Make sure you download a Linux version for `arm64` or `aarch64`

## Hypervisors

VMWare

VirtualBox

QEMU
via `hyper-v` , `KVM` or TODO
macOS

# Virtual Box

A Virtual Box Machine Settings Example

# Distribution

what poeple think an OS is

# UNIX

- allows several users
  - connect to the mainframe using several terminals
  - run several programs in paralel
- is open source up to version *UNIX System III*

Ken Thompson & Dennis Ritchie (Bell Labs, AT&T)


- mainframes provide a lot of hardware
  - needs to be shared by several users
  - each user has a terminal connected to the mainframe
- writes UNIX

# The POSIX Standard

Portable Operating System Interface

Defines:

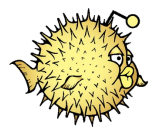
- how an OS should work (*everything is a file*)
- how the C API looks like (ex `stdio.h`, `unistd.h`)
- what commands (basic software) should be available

Allows:

- applications to be portable
  - *in C/C++ source code format only* ->

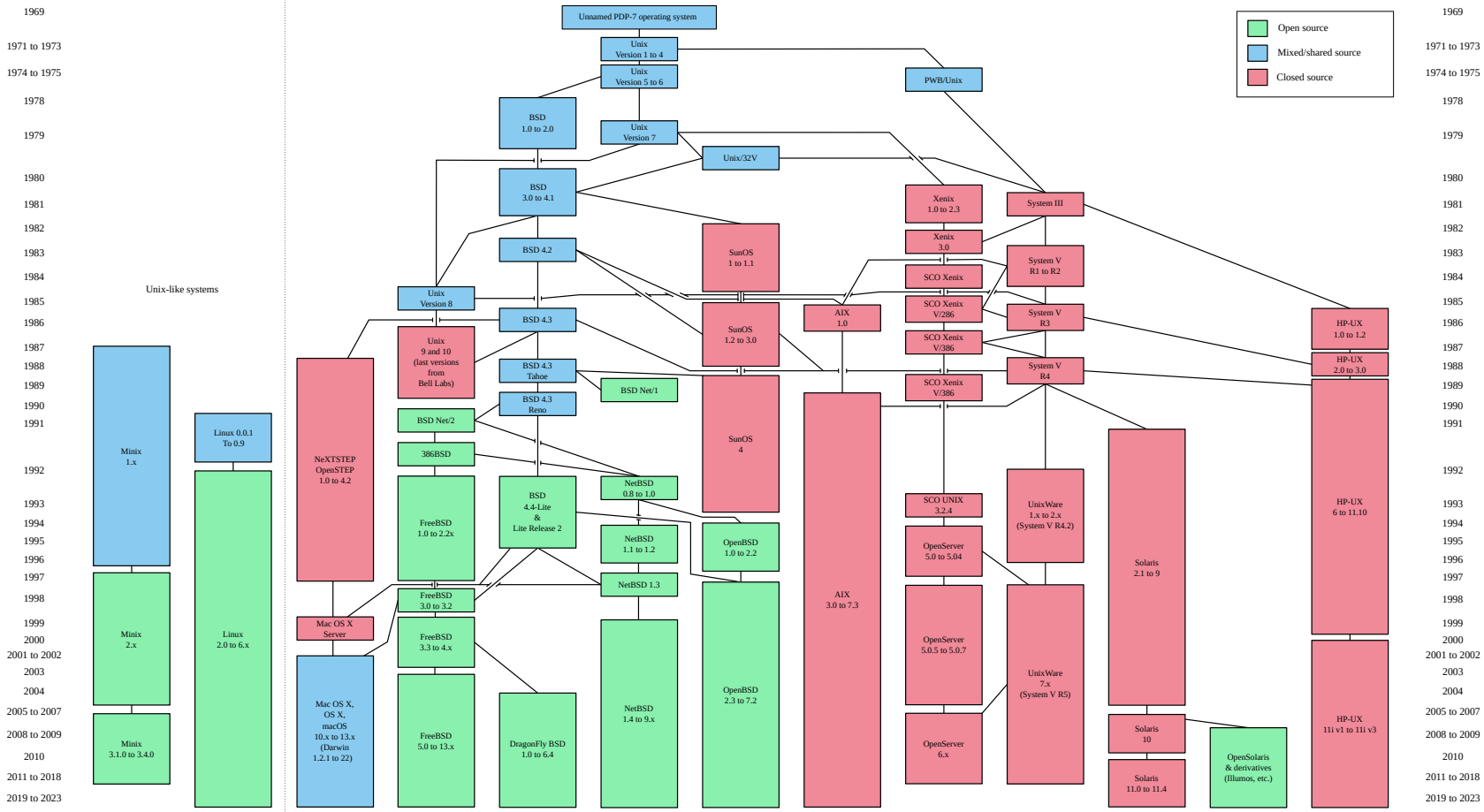
When UNIX became closed source ->

UC **Berkely Software Distribution (BSD)**

| FreeBSD | Darwin | OpenBSD | NetBSD |
| --- | --- | --- | --- |
|  |  |  |  |

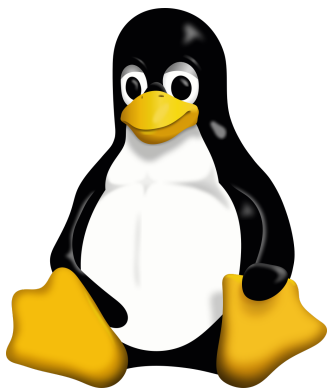
New kernels

| **Minix** | **Linux** |
| --- | --- |


1969
1971 to 1973
1974 to 1975
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001 to 2002
2003
2004
2005 to 2007
2008 to 2009
2010
2011 to 2018
2019 to 2023
Unix-like systems
Minix 1.x
Minix 2.x
Minix 3.1.0 to 3.4.0
Linux 0.0.1 To 0.9
Linux 2.0 to 6.x
Unnamed PDP-7 operating system
Unix Version 1 to 4
Unix Version 5 to 6
Unix Version 7
PWB/Unix
BSD 1.0 to 2.0
Unix/32V
BSD 3.0 to 4.1
BSD 4.2
Unix Version 8
BSD 4.3
BSD 4.3 Tahoe
BSD 4.3 Reno
Unix 9 and 10 (last versions from Bell Labs)
BSD Net/1
BSD Net/2
386BSD
FreeBSD 1.0 to 2.2x
FreeBSD 3.0 to 3.2
FreeBSD 3.3 to 4.x
FreeBSD 5.0 to 13.x
BSD 4.4-Lite & Lite Release 2
DragonFly BSD 1.0 to 6.4
NeXTSTEP OpenSTEP 1.0 to 4.2
Mac OS X Server
Mac OS X, OS X, macOS 10.x to 13.x (Darwin 1.2.1 to 22)
NetBSD 0.8 to 1.0
NetBSD 1.1 to 1.2
NetBSD 1.3
NetBSD 1.4 to 9.x
SunOS 1 to 1.1
SunOS 1.2 to 3.0
SunOS 4
OpenBSD 1.0 to 2.2
OpenBSD 2.3 to 7.2
AIX 1.0
AIX 3.0 to 7.3
Xenix 1.0 to 2.3
Xenix 3.0
SCO Xenix
SCO Xenix V/286
SCO Xenix V/386
SCO UNIX 3.2.4
OpenServer 5.0 to 5.04
OpenServer 5.0.5 to 5.0.7
OpenServer 6.x
System III
System V R1 to R2
System V R3
System V R4
UnixWare 1.x to 2.x (System V R4.2)
UnixWare 7.x (System V R5)
Solaris 2.1 to 9
Solaris 10
Solaris 11.0 to 11.4
OpenSolaris & derivatives (Illumos, etc.)
HP-UX 1.0 to 1.2
HP-UX 2.0 to 3.0
HP-UX 6 to 11.10
HP-UX 11i v1 to 11i v3
Open source
Mixed/shared source
Closed source

# The Linux Kernel

- is the actual operating system
- invisible to the user
  - except when it boots 🏁 and panics 😱
- 25 mil line of code (LOC)
- manages all the system resources



Meet *Tux*

*Linus Torvalds* (University of Helsinky)

- did not agree with A Tannenbaum about how Minix should work
- wrote this own 👨‍💻 *POSIX compliant* OS, called it *Linu**x***

# GNU Software

Basic *UNIX* commands and software **rewritten**

- startup software ( `init` )
- most of what we call today *coreutils*
- C Compiler -> GNU Compiler Collection ( `gcc` )
  - C and C++ standard libraries
  - `flex` / `yacc`
- X Windows Systsem ( `x11` )

**Modern Software**

- *coreutils* are being rewritten in Rust (*uutils*)
- `gcc` is being slowly replaced by LLVM
- `x11` is being replaced by Wayland

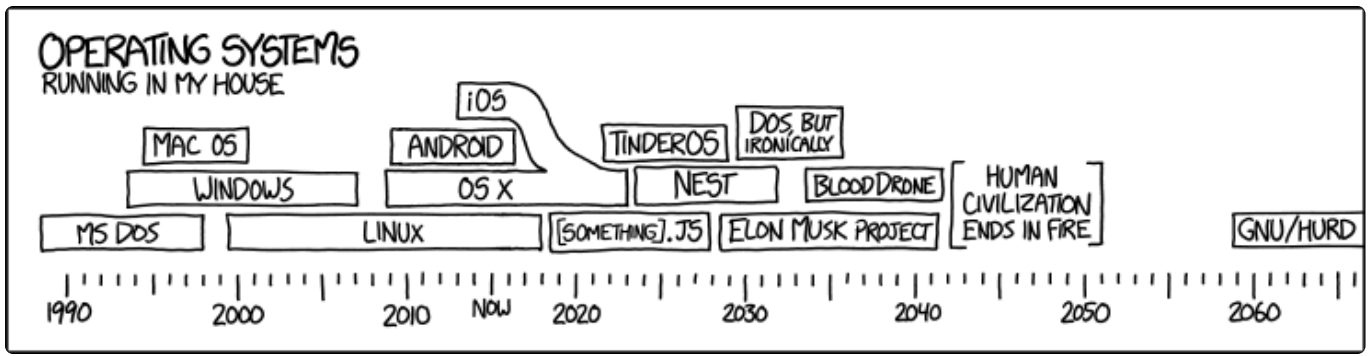
*Richard Stallman* (MIT)

His vision is that software should be:

- free to share and modify
- GPL and LGPL licenses
  - allows selling and modifying
  - modified code must be relicensed under GPL

# GNU/Linux = ❤

GNU Software on top of the Linux kernel




- GNU Hurd 👷🚧🏗️ (*work in progress for many many years*)
- The Linux kernel 🐧 still is the *temporary replacement*

# Base Distributions

The main Linux distributions that started it

*Took the Linux kernel, added the GNU libraries and tools and wrote a package manager to install software*

| | Name | Tagline | Package Format | Package Manager | Release Year |
|---|---|---|---|---|---|
|  | Slackware | *Oldest distribution* | `tgz` | `pkgtool` | 1993 |
|  | Debian | *Free Software* | `deb` | `apt` / `dpkg` | 1993 |
|  | Red Hat Linux | *Enterprise* | `rpm` | `dnf` / `rpm` | 1995 |

These are the base for most of the modern distributions that we have today.

# Modern Distributions

| | Name | Tagline | Parent | Package Format | Package Manager | Release Year |
|---|---|---|---|---|---|---|
|  | SUSE Linux | *Enterprise-grade Linux* | *Originally Slackware, later RPM-based* | `rpm` | `zypper`, `yast` | 1994 |
|  | ArchLinux | *Minimal Linux* | N/A | `pkg.tar.zst` | `pacman` | 2002 |
|  | Fedora | *Desktop Linux* | *Red Hat Linux renamed* | `rpm` | `dnf` / `rpm` | 2003 |
|  | Ubuntu | *Linux for Humans* | Debian | `deb` and `snap` | `apt` / `dpkg` and `snap` | 2004 |
|  | openSUSE | *Stable, usable Linux for everyone* | SUSE Linux | `rpm` | `zypper`, `rpm` | 2005 |

# Which is the most used Linux distribution?

# Non GNU Distributions

They use Linux, but most of the software is not from GNU

| | Distribution | Tagline | Parent | Package Format | Package Manager | Release Year |
|---|---|---|---|---|---|---|
|  | Android | *Mobile Linux platform* | Linux kernel (AOSP) | `.apk` (only Android apps) | AOSP tools (not typical package manager) | 2008 |
|  | ChromeOS | *The cloud-first OS* | Gentoo Linux | Custom ( `.crx` , `.apk` , others) | `portage` , `cros_sdk` , Flatpak (via Crostini) | 2011 |

# Conclusion

we talked about

- What is Operating System is
- The Operating System Stack
- Virtual Machines
- History of Linux
- Distributions