Intro to *nix and Shells

Level 0x00: The Shell

Quick Overview

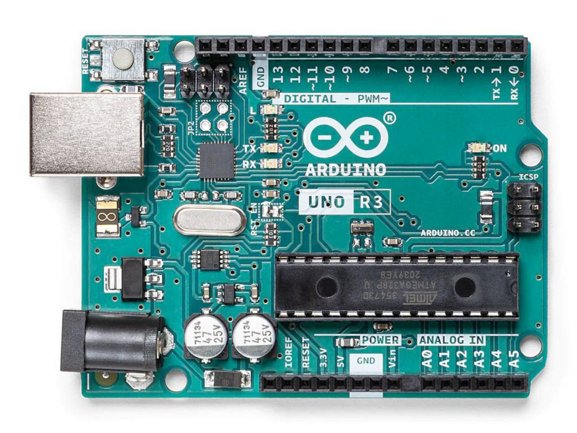
- About me
- Club Agenda
- Unix / Linux
- Virtual Machines

About me - Mike Wales

- Work at Nightwing for 9 years
- Worked at L3Harris for 14 years
- BS Computer Engineering - UCF 2001
- Software Development
  - Mostly embedded (VxWorks, Linux)
  - Mostly C/C++
- Cybersecurity Engineer
  - Hacking
  - Reverse Engineering
  - Vulnerability Research
  - Software Developer
- 4th year of helping with CS Club
- Retro-gaming

Bare-metal / Embedded

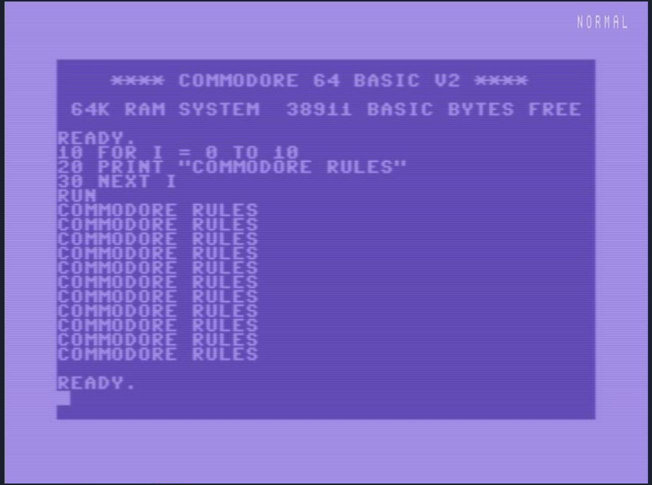
- ATMEGA 328P Processor (AVR)
  - 32KB Flash (programmable memory)
  - 1 KB EEPROM
  - 2 KB SRAM
  - 1 MHz Clock
- No operating system
- Runs 1 program at a time
  - Assembly
  - C
- Often faster than it seems



This picture is 160 KB

Commodore 64

- Commodore 64 $595
  - 1 MHz 6502 CPU
  - 16 colors
  - 64 KB RAM
  - 8 KB Kernal
  - 4 KB Character Graphics
  - 8 KB Basic
- 1541 Floppy Drive $400
  - 170 KB 5 ¼” floppy

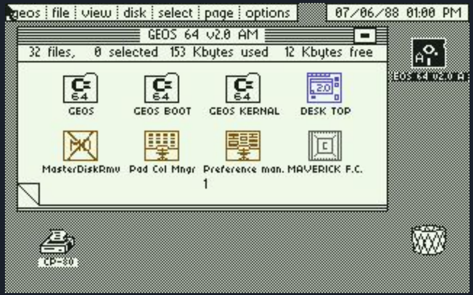


```
    **** COMMODORE 64 BASIC V2 ****

 64K RAM SYSTEM  38911 BASIC BYTES FREE

READY.
10 FOR I = 0 TO 10
20 PRINT "COMMODORE RULES"
30 NEXT I
RUN
COMMODORE RULES
COMMODORE RULES
COMMODORE RULES
COMMODORE RULES
COMMODORE RULES
COMMODORE RULES
COMMODORE RULES
COMMODORE RULES
COMMODORE RULES
COMMODORE RULES
COMMODORE RULES

READY.
```



32 files, 0 selected 153 Kbytes used 12 Kbytes free

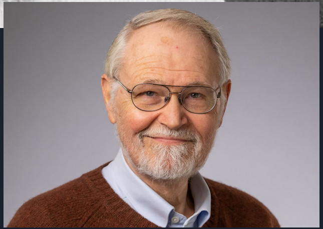
CS Club Topics

- Hacker History
- Linux
- Command line interface / Shell
- Computer Networking
- Cyber Security / Hacking
    - Encryption
    - Forensics
    - Reverse Engineering
    - Vulnerabilities and Exploits
    - Physical security / Lock-picking
- C programming

- Engineering / Embedded
    - Electronics
    - Digital Circuits
    - Microcontrollers (Arduino)
    - Raspberry Pi

Operating Systems - UNIX-like

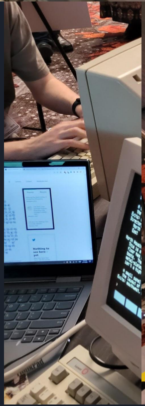
AT&T Bell Laboratories



- Unix
  - AT&T Bell Labs in 1970s by Ken Thompson, Dennis Ritchie, Brian Kernighan
  - Examples include: BSD, HP-UX, Solaris, SGI Irix
  - Multi-tasking, multi-user, programming tools included
  - Unix philosophy: “Write programs that do one thing and do it well”
- OpenBSD / FreeBSD
  - University of California Berkeley open sources their Unix - permissive license
  - Also used by Mac OS, iOS, Playstation 3 (and newer)
- Linux
  - Created by Linus Torvalds, first posted to Usenet in 1991
  - RedHat, Suse, Debian, Ubuntu, Android
  - “I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386(486) AT clones.”

Early Interfaces

- Serial terminal / Teletype
- No graphics, just characters
- No mouse

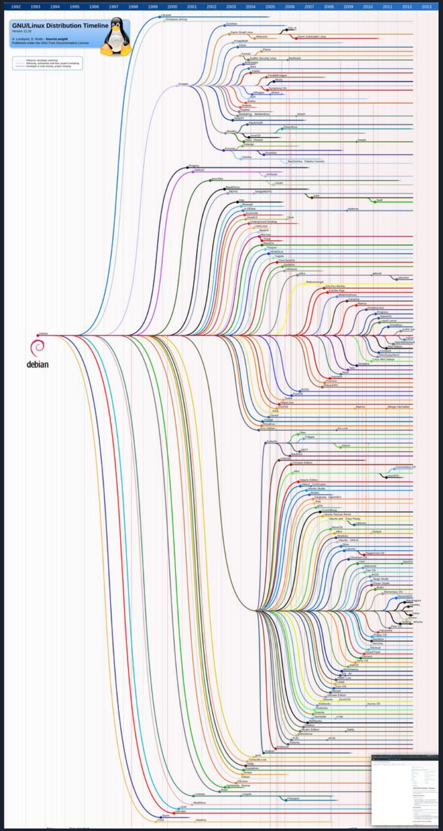


Linux Strengths

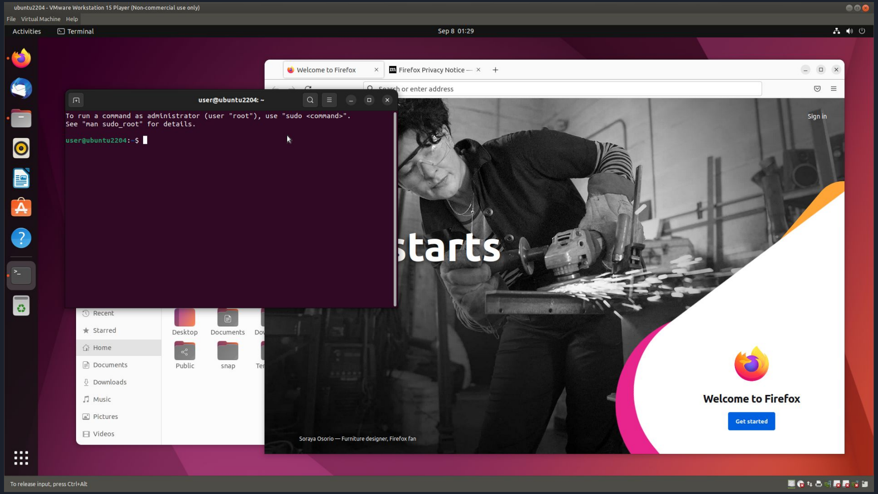
- Command Line First
  - Everything can be done via command line
  - Easy to automate
  - Remote access
- Live Versions
  - Knoppix / Tails
- Customization / Open Source
  - Kernel is fully transparent
  - Easy for anyone to add to kernel
  - User space is fully customizable (Steam Deck)
- Package Managers
  - Easy / fast to add other open source tools / development packages

Ubuntu

- Child / Fork of Debian Linux
- Ubuntu provides:
    - Software repository with > 20,000 packages (apps, libraries)
    - Apt package manager
        - Installs new packages
        - Updates packages
        - Removes packages
- Ubuntu has many flavors
    - Which desktop manager used by default
    - Which applications used by default
    - Custom theming
- Releases every 6 months
    - LTS every 2 years. 24.04 is current LTS

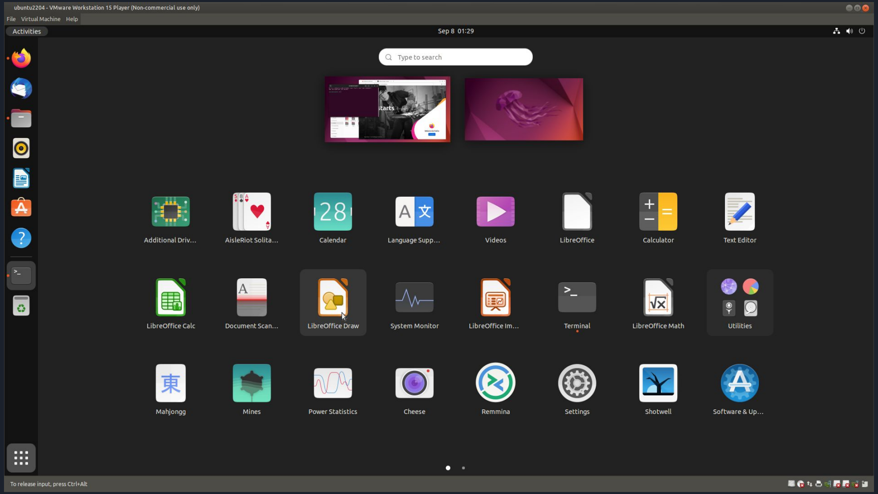


“Vanilla” Ubuntu

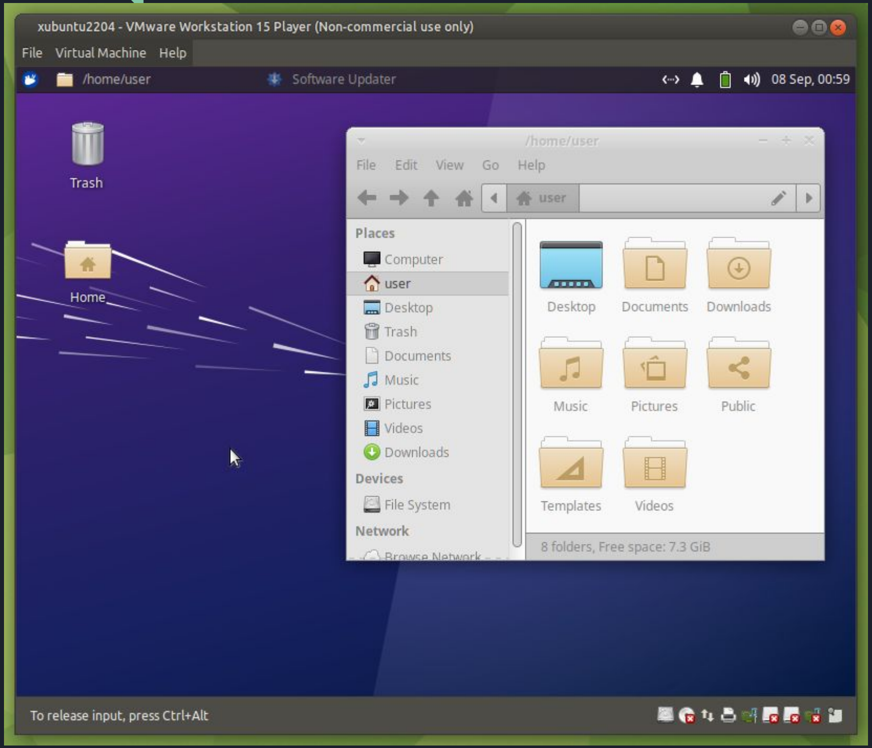


```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

user@ubuntu2204:~$ 
```

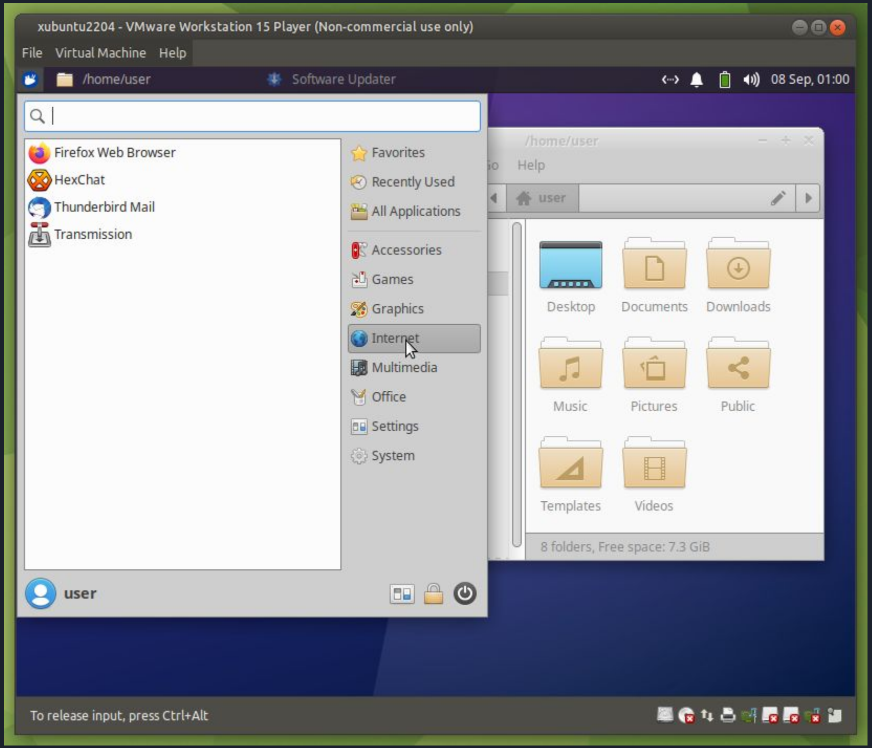


Xubuntu



8 folders, Free space: 7.3 GiB

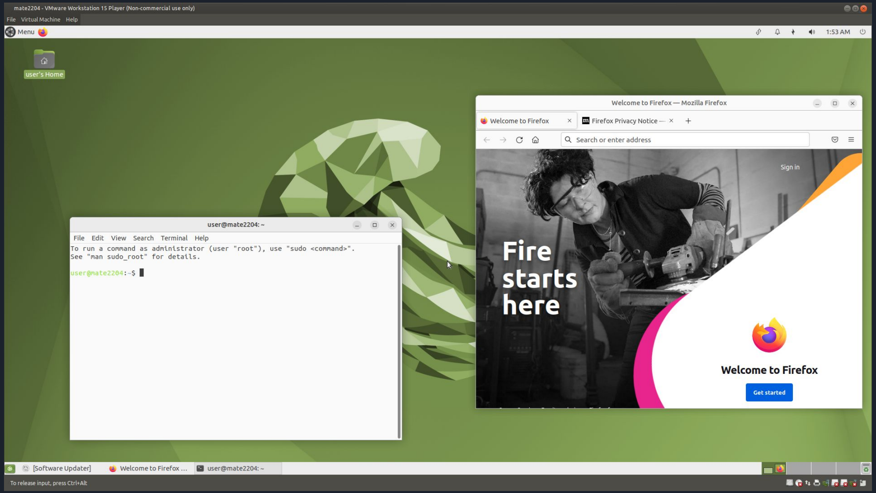
To release input, press Ctrl+Alt



8 folders, Free space: 7.3 GiB

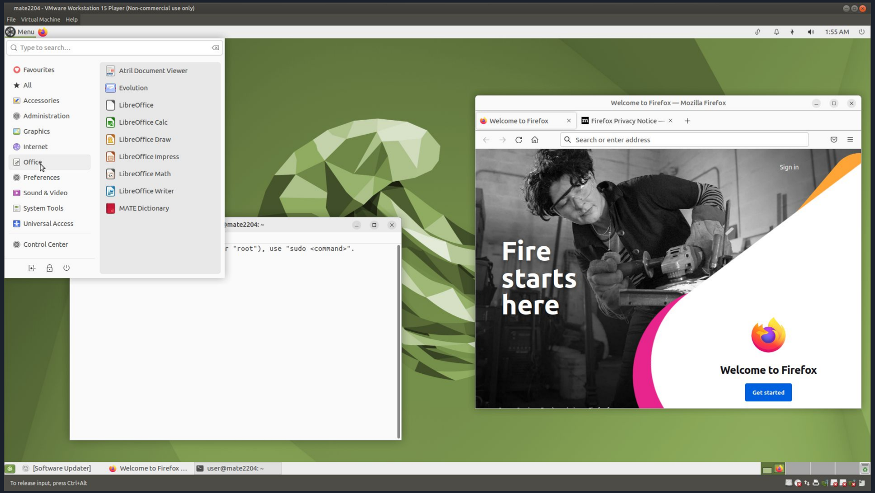
To release input, press Ctrl+Alt

Ubuntu MATE



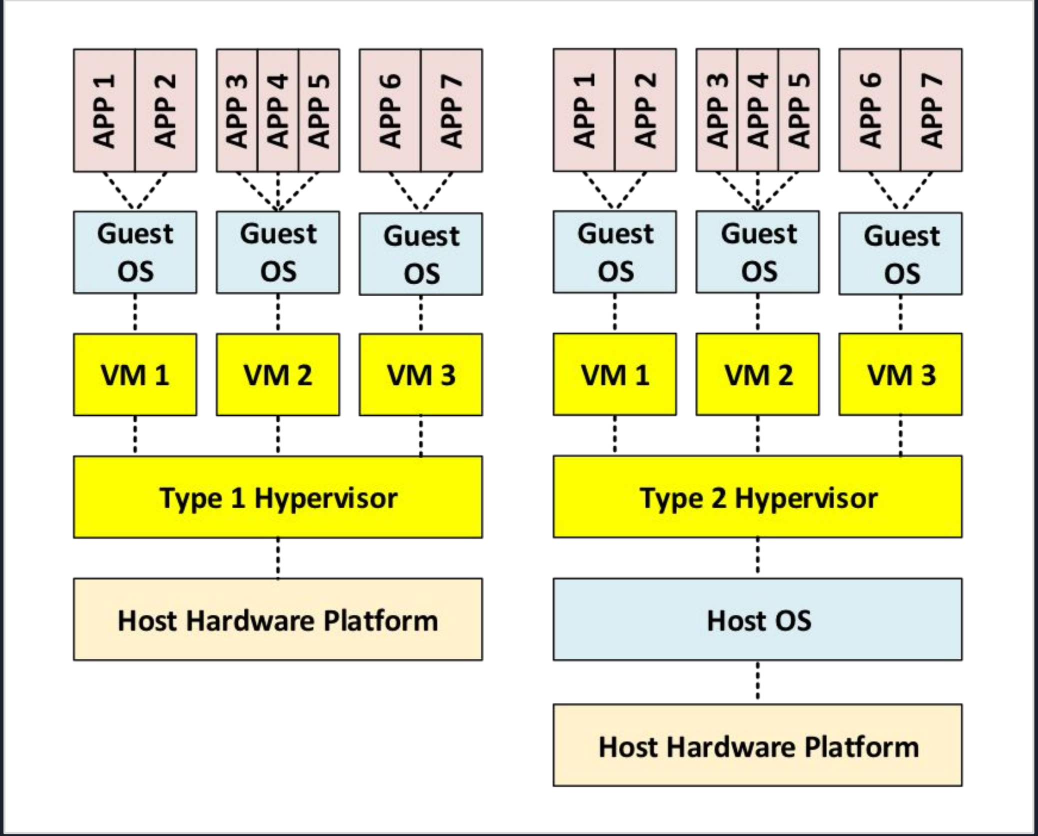
```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

user@mate2204:~$
```



```
r "root"), use "sudo <command>".
```

Virtualization



Shell / Linux apps

Ubuntu Linux 22.04

.vmdk files

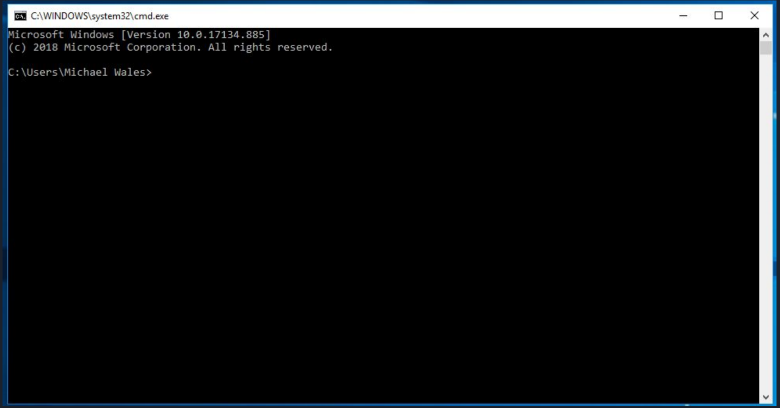
VMWare Player / WS

Windows 10/11

PC / Laptop

Windows Shell Basics

- Windows Basic Shell
  - Press Win+R to bring up Run dialog
  - Type cmd to open shell
  - Functional, but very basic
- Windows Alternative Shells
  - Powershell
  - WSL (Windows Subsystem for Linux)
  - WSL2

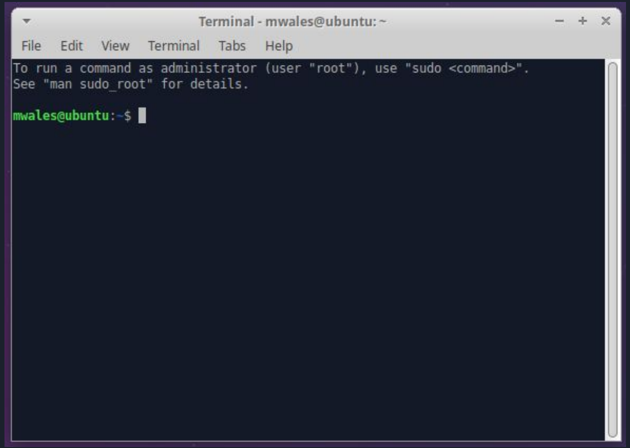


```
Microsoft Windows [Version 10.0.17134.885]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Michael Wales>
```

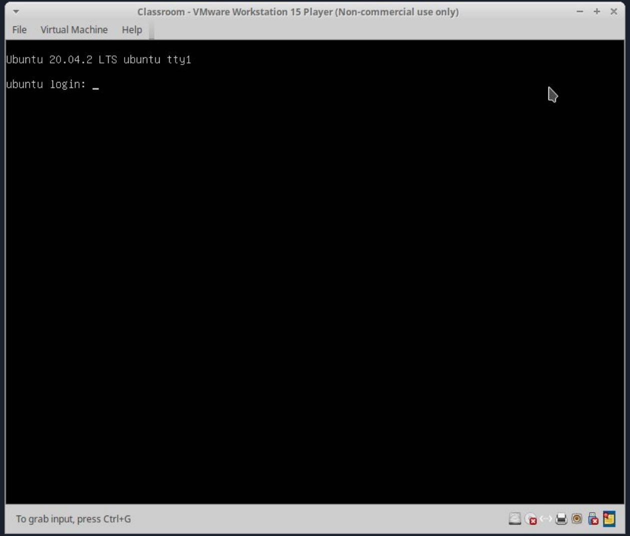
Linux Shell Basics

- Bourne Shell (sh) and Bash (Bourne Again Shell)
  - There are many many others
- Many ways to access the shell
  - GUI Shell Program (Terminal)
  - `/dev/tty1` text console
    - CTRL+ALT+F1 (through F6 typically)
    - CTRL+ALT+F7 restores GUI
  - Serial port
  - Remotely via SSH (or Telnet)



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

mwales@ubuntu:~$ 
```


Classroom - VMware Workstation 15 Player (Non-commercial use only)
File
Virtual Machine
Help
To grab input, press Ctrl+G

```
Ubuntu 20.04.2 LTS ubuntu tty1

ubuntu login: _
```

Filesystem

- Filesystem is usually a directory of files on your SSD / hard disk
  - Windows: C: D: (drive letters)
  - *nix: / /mnt /media/cdrom
- Each directory can have thousands of files and other directories

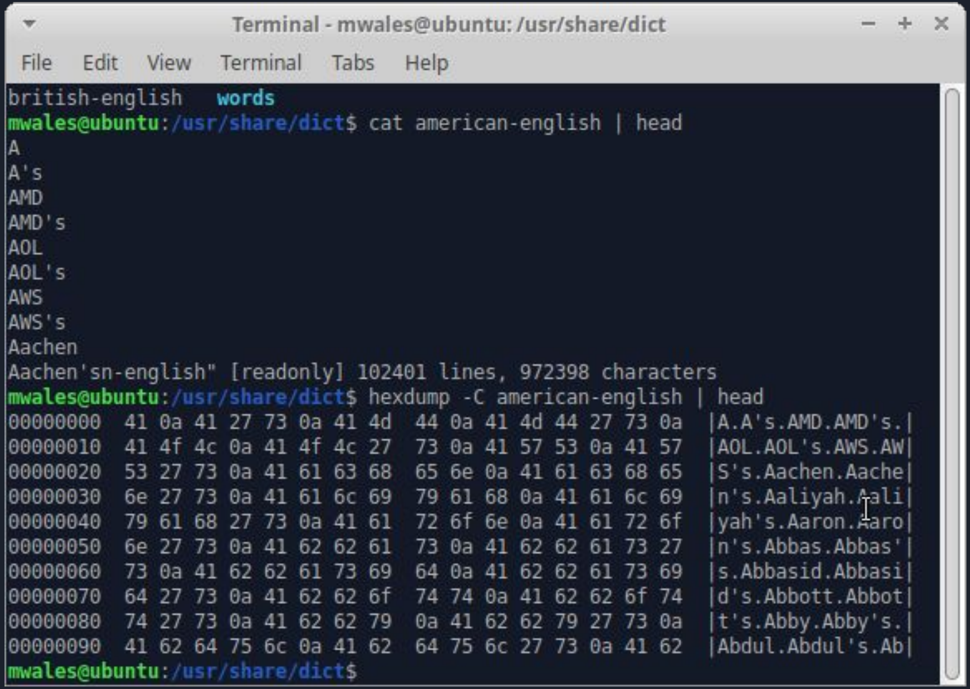
Linux Command	Windows/DOS Command	Explanation
`pwd`	`cwd`	Present working directory
`ls`	`dir`	List contents of a directory

Directory Commands

Linux	Windows / DOS	Explanation
`mkdir DIRECTORY`	`mkdir`	Makes a new directory
`cd DIRECTORY`	`cd`	Changes to a subdirectory
`cd ..`	`cd ..`	Changes to the parent directory
`rmdir DIRECTORY`	`rmdir`	Removes a directory (must be empty)
`tree`	`dirtree`	Lists all files / subdirectories

Files

- Common contents of a file
  - Text
  - Executable Programs
  - Databases (SQL)
  - Compressed Archive
  - Images
  - Word document
    - Compressed Archive
    - Text
    - Images



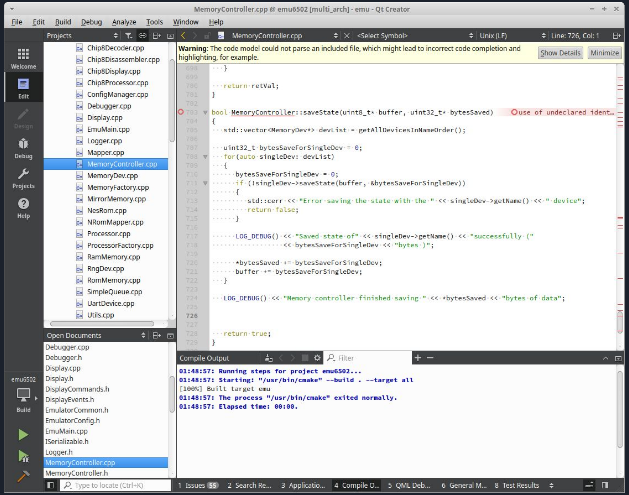
```
british-english   words
mwales@ubuntu:/usr/share/dict$ cat american-english | head
A
A's
AMD
AMD's
AOL
AOL's
AWS
AWS's
Aachen
Aachen'sn-english" [readonly] 102401 lines, 972398 characters
mwales@ubuntu:/usr/share/dict$ hexdump -C american-english | head
00000000  41 0a 41 27 73 0a 41 4d  44 0a 41 4d 44 27 73 0a  |A.A's.AMD.AMD's.|
00000010  41 4f 4c 0a 41 4f 4c 27  73 0a 41 57 53 0a 41 57  |AOL.AOL's.AWS.AW|
00000020  53 27 73 0a 41 61 63 68  65 6e 0a 41 61 63 68 65  |S's.Aachen.Aache|
00000030  6e 27 73 0a 41 61 6c 69  79 61 68 0a 41 61 6c 69  |n's.Aaliyah.Aali|
00000040  79 61 68 27 73 0a 41 61  72 6f 6e 0a 41 61 72 6f  |yah's.Aaron.Aaro|
00000050  6e 27 73 0a 41 62 62 61  73 0a 41 62 62 61 73 27  |n's.Abbas.Abbas'|
00000060  73 0a 41 62 62 61 73 69  64 0a 41 62 62 61 73 69  |s.Abbasid.Abbasi|
00000070  64 27 73 0a 41 62 62 6f  74 74 0a 41 62 62 6f 74  |d's.Abbott.Abbot|
00000080  74 27 73 0a 41 62 62 79  0a 41 62 62 79 27 73 0a  |t's.Abby.Abby's.|
00000090  41 62 64 75 6c 0a 41 62  64 75 6c 27 73 0a 41 62  |Abdul.Abdul's.Ab|
mwales@ubuntu:/usr/share/dict$
```

File Commands

Linux	Windows / DOS	Explanation
`touch FILE`	`copy con FILE`	Creates a blank file
`cat FILE`	`type FILE`	Displays contents of a file
`head FILE`		Displays beginning of a file
`tail FILE`		Displays ending of a file
`hexdump -C FILE`		Displays contents of a binary
`file FILE`		Tells you what type of a file

Editors

- GUI
  - Simple: write text, save to a file
    - Gedit, Mousepad, Notepad
  - Coding: automatic coloring, auto-complete
    - Geany
    - Sublime ($)
    - Atom
  - IDE: integrated development environment
    - Qt Creator
    - Visual Studio
    - CLion
- Command Line
  - nano, pico
  - vi / vim, emacs



Warning: The code model could not parse an included file, which might lead to incorrect code completion and highlighting, for example.

```
    }

    return retVal;
}

bool MemoryController::saveState(uint8_t* buffer, uint32_t* bytesSaved)   use of undeclared ident…
{
    std::vector<MemoryDev*> devList = getAllDevicesInNameOrder();

    uint32_t bytesSaveForSingleDev = 0;
    for(auto singleDev: devList)
    {
        bytesSaveForSingleDev = 0;
        if (!singleDev->saveState(buffer, &bytesSaveForSingleDev))
        {
            std::cerr << "Error saving the state with the " << singleDev->getName() << " device";
            return false;
        }

        LOG_DEBUG() << "Saved state of" << singleDev->getName() << "successfully ("
                    << bytesSaveForSingleDev << "bytes )";

        *bytesSaved += bytesSaveForSingleDev;
        buffer += bytesSaveForSingleDev;
    }

    LOG_DEBUG() << "Memory controller finished saving " << *bytesSaved << "bytes of data";



    return true;
}
```

```
01:48:57: Running steps for project emu6502...
01:48:57: Starting: "/usr/bin/cmake" --build . --target all
[100%] Built target emu
01:48:57: The process "/usr/bin/cmake" exited normally.
01:48:57: Elapsed time: 00:00.
```

File Commands

Command	Explanation
`strings FILE`	Prints out printable strings of a binary file
`sort [FILE]`	Prints lines in alphabetical order
`uniq [FILE]`	Removes redundant lines out output
`wc [FILE]`	Counts number of words in a file
`dos2unix / unix2dos [FILE]`	Converts file line endings
`more / less [FILE]`	Shows output 1 page at a time
`grep needle [FILEs]`	Searches for a string

Standard Input / Output

- 3 file descriptors open by CLI application
  - 0 = stdin (standard input)
  - 1 = stdout (standard output)
  - 2 = stderr (standard error)
- Pipes (|) can be used to connect output from one application to input of another application

```
strings somefile | grep -i password

cat logfile | sort | unique
```

I/O Redirection

- Using "`> file.txt`" after a command causes output from stdout to be redirected into a file
  - You won't be able to see it on screen
  - stderr will still be displayed
- Using "`2> file.txt`" after a command causes stderr to be redirected into file
- Using "`> file.txt 2>&1`" causes both to be redirected
  - Order matters!
- `tee` will write standard output to a file and also write it to the screen
  - Ex: `./myprogram arg1 arg2 | tee logfile.txt`
- >> will append to existing file, > overwrites it

Shell scripts

- A series of commands in a text file
  - Linux
    - Can start text file with #! (shebang) and make executable
    - Can call interpreter directly
  - Windows
    - .bat (batch) files
    - Windows Power Shell
- Can take arguments ($1, $2)
- Number of arguments ($#)
- Command Substitution (not just for scripts)
  - `echo “There are `ls *.txt | wc -l` files in this directory”`
  - `echo “There are $(ls *.txt | wc -l) files in this directory”`

Executable Files

- Linux - permissions bits
  - Permission bits for user, then group, then others
  - r = read, w = write, x = executable
  - `$ ls -l`

    ```
    -rwxrwxr-x 1 mwales mwales 16784 Feb  1  2023 a.out
    -rw-rw-r-- 1 mwales mwales    26 Feb  1  2023 flag.txt
    -rwxrwxr-x 1 mwales mwales  3969 Feb  3  2023 judge.py
    -rw-rw-r-- 1 mwales mwales   330 Feb  2  2023 solution.c
    ```
  - `chmod` can change file permissions
- Windows - file extension
    - .bat (batch) and .cmd (command) script files
    - .exe and .com binary files
    - Many others
-

Attributions

- Ken Thompson and Dennis Ritchie: from Wikipedia, public domain
- Linus Torvalds: Wikimedia Creative Commons Attribution-Share Alike 3.0
- Debian Family Tree: Andreas Lundqvist, Donjan Rodic from wikimedia.org
- Virtualization via Virtual Machines - VM architecture