lrz

Leibniz Supercomputing Centre

of the Bavarian Academy of Sciences and Humanities

Introduction to GNU/Linux and the Shell

October 2022

Course Information

- The aim of this course is to provide an introduction to GNU/Linux and the Unix Shell
- You will probably benefit the most if you're not yet familiar with GNU/Linux and the Unix Shell, but if you plan to work on the AI, HPC and/or Compute Cloud infrastructure provided by LRZ
  -> by the end of this course, you should have the basic skills to successfully interact with GNU/Linux-based systems
- This is the self-guided part of the course, it should get you started -> Follow the practical instructions. They can typically be found in blue text color, blue boxes and/or blue side columns. Try to answer all questions before you proceed to the next slide(s)

Do you recall using a GNU/Linux operating system?

If so, which one and for what purpose?

What is GNU/Linux?

- GNU/Linux is a free and open-source operating system
- As such, it is an alternative to popular operating systems like Microsoft Windows, Apple macOS, Google Android and others
- It powers the whole spectrum of computer systems like embedded and mobile devices as well as personal computers but also the majority of web servers and the world's largest supercomputers
- On a desktop system, it typically consists of the Linux kernel, libraries and tools, a graphical desktop environment and various applications like a web browser, an office suite and more. On servers, there is usually no graphical user interface provided ("headless" system)
- As these software projects are typically open source and freely available, there are several so-called distributions, which bundle these components to create a coherent system and user experience.
  Some of them are largely community-driven like Arch Linux or the Debian project, others are backed by commercial entities. These typically also provide community versions of their distributions.
  Examples are the Fedora and RHEL projects, backed by the company Red Hat/IBM, openSUSE and SLES, backed by SUSE and Ubuntu, backed by Canonical

Are you aware of any free/libre open source software (FLOSS) projects?

Meet Tux, the mascot of the Linux kernel.

https://www.gnu.org/licenses/quick-guide-gplv3

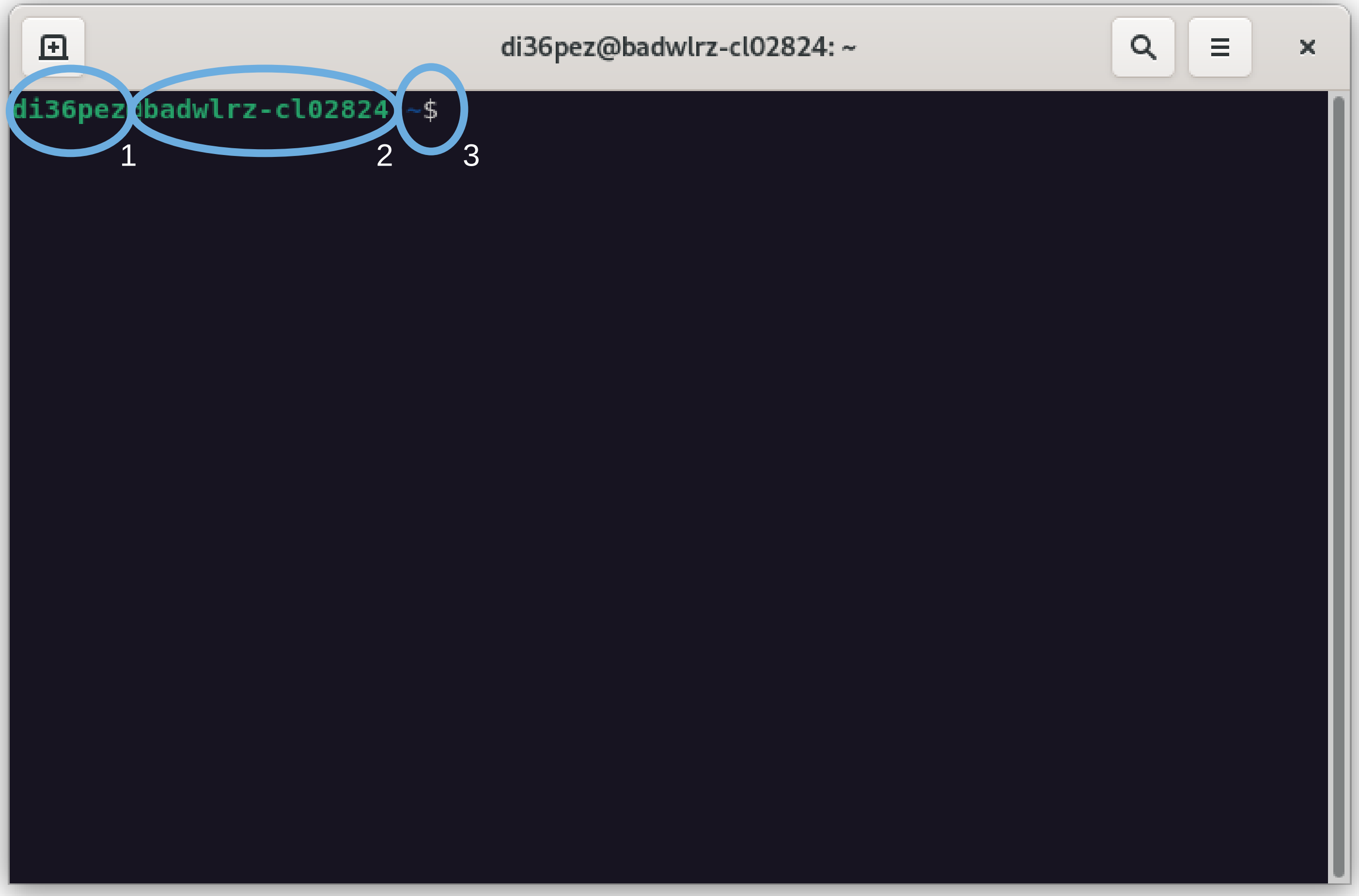
GNU General Public License (GPL)

The powerful versatility of free and open-source software is rooted in their licenses, such as the GNU General Public License (GPL). This license grants four essential freedoms or rights to the users of the software:

- The freedom to “**use** a program as they wish, for any purpose",
- the right to "**study** how the program works, and change it so it does the computing as they wish",
- the freedom to "**share** and redistribute copies so they can help their neighbor" and finally
- the right to "**improve** the software and to distribute copies of their modified versions to others".

These rights are key for e.g. tuning software on a one-of-a-kind supercomputer, but also, more generally, in an environment where the goal is to create reproducible research and open science.

A Shell in a Terminal Application



```
di36pez@badwlrz-cl02824 ~$
```

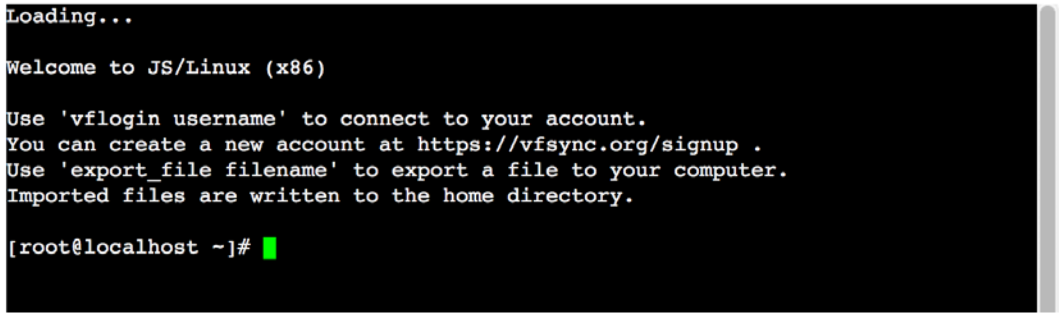
When working in the terminal of a GNU/Linux distribution or connecting to a headless server remotely, after login you will be greeted by a sight like this: a shell environment.
It typically displays your username (1), the hostname of the machine you're working on (2) as well as some representation of the current file system path. It awaits user input (3).

It’s time to get started: launch a Unix-like shell environment

- GNU/Linux: choose your favorite terminal application
- macOS: launch Terminal
- Windows:
  - Windows 10/11*: Install the Windows Subsystem for Linux (WSL)
    https://docs.microsoft.com/en-us/windows/wsl/install
  - Alternatively (especially on older versions of Windows):
    - Git BASH (as part of Git for Windows) or MSYS2
      https://gitforwindows.org/ https://www.msys2.org/
    - MobaXterm
      https://mobaxterm.mobatek.net/
  - Or, for a very basic introduction and hands-on,
    visit https://bellard.org/jslinux/ in your browser.
    NOTE: this will not be suitable for “real” work later on (e.g. working with SSH). You should preferably set up a proper shell environment on your local system now!

* If you intend to directly connect to a remote GNU/Linux system, you can also use built-in commands (i.e. ‘ssh’) in cmd.exe / PowerShell on recent versions of Windows (10+). See final slide.

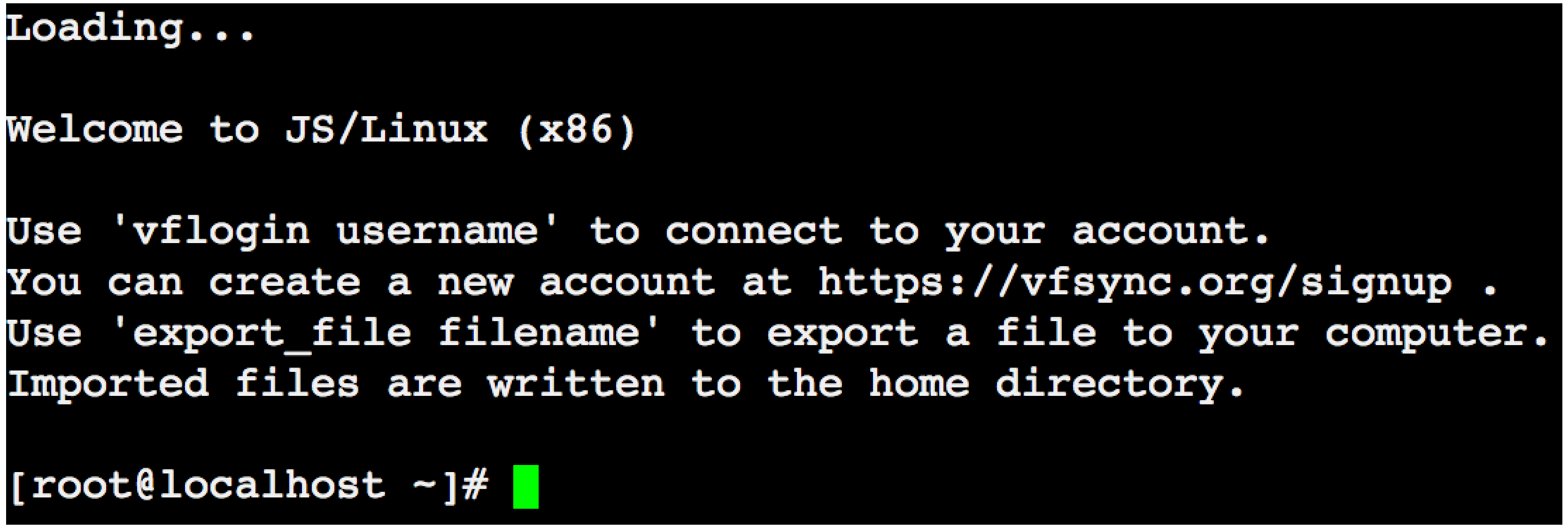
Get your bearings after launching the shell environment



```
Loading...

Welcome to JS/Linux (x86)

Use 'vflogin username' to connect to your account.
You can create a new account at https://vfsync.org/signup .
Use 'export_file filename' to export a file to your computer.
Imported files are written to the home directory.

[root@localhost ~]#
```

What is “your” username?

What is the hostname of the machine?



```
Loading...

Welcome to JS/Linux (x86)

Use 'vflogin username' to connect to your account.
You can create a new account at https://vfsync.org/signup .
Use 'export_file filename' to export a file to your computer.
Imported files are written to the home directory.

[root@localhost ~]#
```

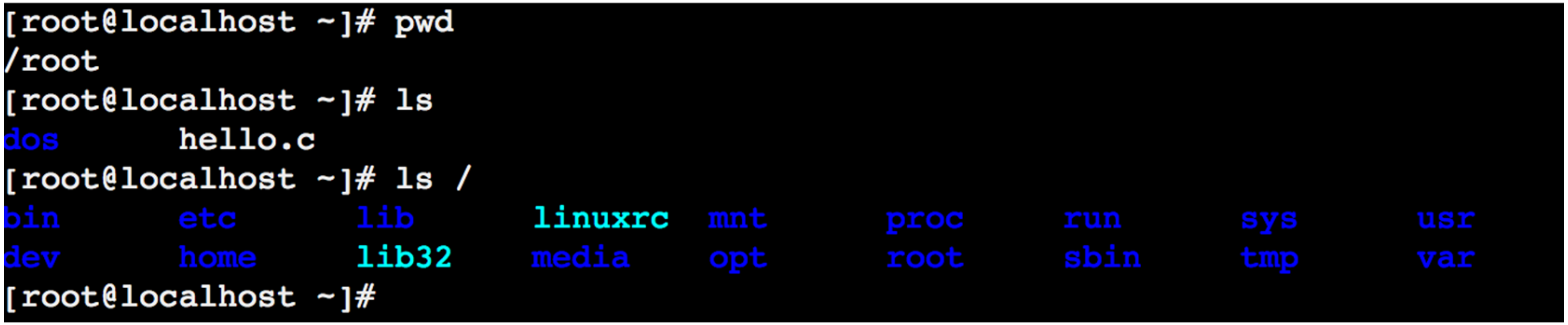
Let’s explore the file system…

File System Hierarchy Standard (FHS)

- On a Unix-like system (pretty much) everything is a file
- All files and directories appear (somewhere) under the root directory “/”, even if stored on different – possibly remote – devices. There are no drive letters like on other operating systems.
- Use `pwd` to get the name of the current working directory
- Use `ls` to list all files and directories in the current directory
- Use `ls /` to list all files and directories in the root directory
- Use `ls /any/other/dir` to list all files and directories in the specified directory

Try the commands introduced on the left.

Exploring the File System



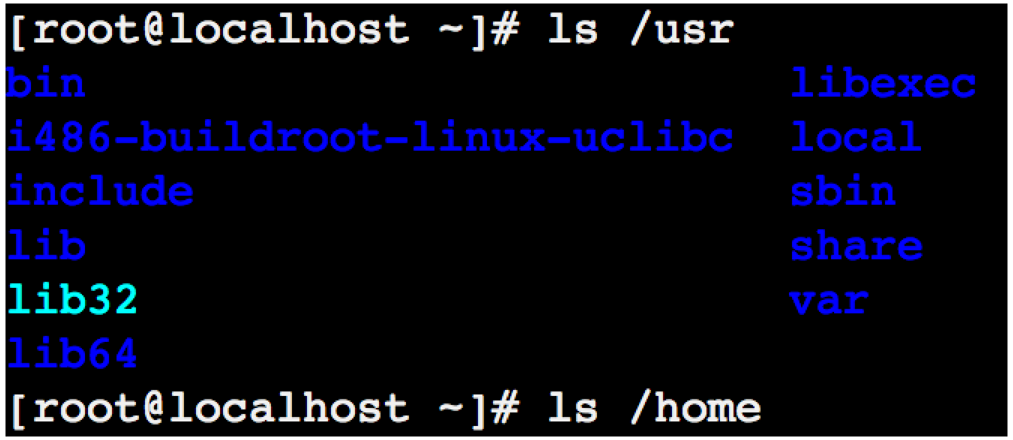
```
[root@localhost ~]# pwd
/root
[root@localhost ~]# ls
dos        hello.c
[root@localhost ~]# ls /
bin        etc        lib        linuxrc    mnt        proc       run        sys        usr
dev        home       lib32      media      opt        root       sbin       tmp        var
[root@localhost ~]#
```

/bin*: command binaries (e.g. ls)
/etc: configuration files
/home: (regular) users' home directories
/lib*: libraries (for binaries in /bin et al.)
/media: mount points for removable media
/mnt: mounted filesystems
/root: home directory of the root user
/sbin*: system binaries
/usr: secondary hierarchy for read-only user data
/var: variable, i.e. changing files

- As an exercise, take a look at the contents of / and /usr. Can you spot the similar directory structure?
- Explore other directories. Are there any (regular) user home directories on your system?

* On modern systems, these (and /libXX) are only symlinks/shortcuts. Their former contents have been merged into their respective /usr/… counterparts, which they then point to.

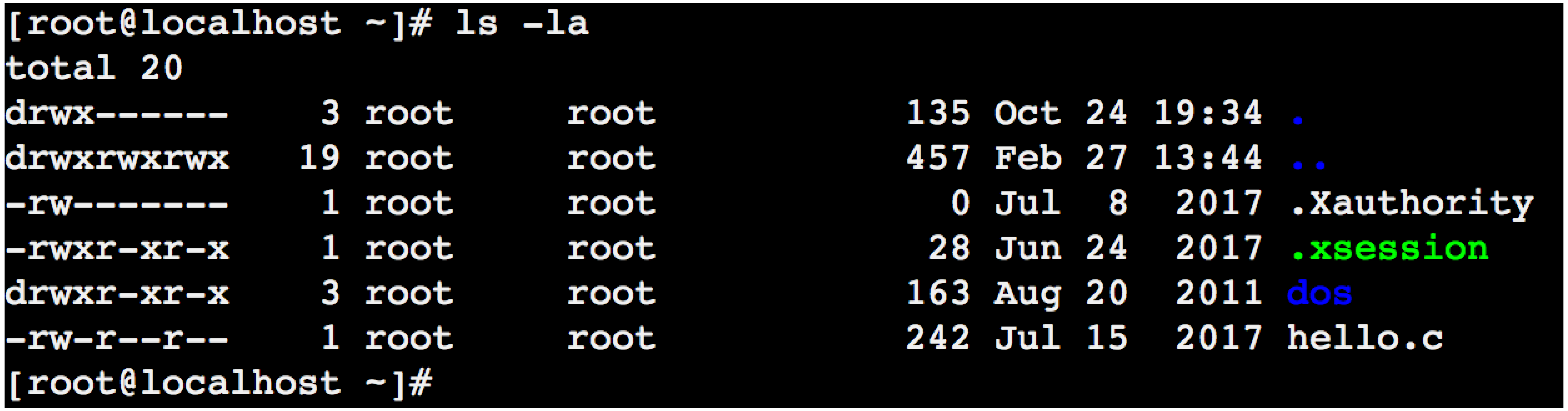
Detailed Listing of All Files



```
[root@localhost ~]# ls /usr
bin                           libexec
i486-buildroot-linux-uclibc   local
include                       sbin
lib                           share
lib32                         var
lib64
[root@localhost ~]# ls /home
```

Use the `l` and `a` options with `ls` (i.e. `ls -la`) to get a detailed listing of all files in your current (home) directory (we will cover most of this information later).

Can you spot the differences to the previous listing (using just `ls`)?



```
[root@localhost ~]# ls -la
total 20
drwx------    3 root     root           135 Oct 24 19:34 .
drwxrwxrwx   19 root     root           457 Feb 27 13:44 ..
-rw-------    1 root     root             0 Jul  8  2017 .Xauthority
-rwxr-xr-x    1 root     root            28 Jun 24  2017 .xsession
drwxr-xr-x    3 root     root           163 Aug 20  2011 dos
-rw-r--r--    1 root     root           242 Jul 15  2017 hello.c
[root@localhost ~]#
```

General Command Syntax

This command syntax can serve as general example to distinguish different components:

```
$ ls -la /home
```

`ls` is the **command**
with the **options** (also **switches** or **flags**) `-la` and
the **argument** `/home`

Options generally start with either a single dash - (as above) or two dashes --.

Getting Help

- There are two common, local ways to find out how a command works and which options it accepts… (you can find many additional resources online, recommendations are https://www.mankier.com/ and https://tldr.sh/)

1. Pass the `--help` option to the command:
   ```
   $ ls --help
   ```
2. Read a command's manual (man pages), using the `man` command:
   ```
   $ man ls
   ```
   (use the arrow keys to move up and down, press `q` to quit the man page)

- What effect does the `-h` option have on the `ls` command?
- Can you spot other interesting options?

At the outset…

- At this point, you should have an initial understanding of what a GNU/Linux operating system is, you should have access to a (Unix-like) shell environment on your *local* machine and made first steps exploring the system. You have encountered the very first commands to interact with the shell environment and you know how to get additional help for such commands.
- In fact, these are already the very basic skills that allow you to start working on *remote* systems using the Secure Shell (SSH).
  See the tutorial: https://doku.lrz.de/display/PUBLIC/SSH+Tutorial
- You will continue – and gain more experience – working with GNU/Linux systems in our on-site/remote course session (navigating the file system, file manipulation and ownership, characteristics of the shell environment, useful commands & concepts, …).
  See you, then!