Introduction to Linux

Dr. George Magklaras
Research Computing Services

UiO : Universitetets senter for informasjonsteknologi

By way of Introduction

Department for Research Computing

Norwegian

Department for Research Computing offers advice regarding digital solutions in research and provides solutions, services and expertise to university scientists and research communities. The department contributes to the University's strategy for the area of IT in research and represents the University in national and international projects and programs that include e-infrastructure and scientific computing.

By way of Introduction (2)

- Abel supercomputer: Initially number 96 in the Top500 list
- 10000 + cores
- 258 Teraflops/sec max. Theoretical peak performance
- 40 TebiBytes of RAM
- 400 TebiBytes of FhGFS filesystem

Agenda

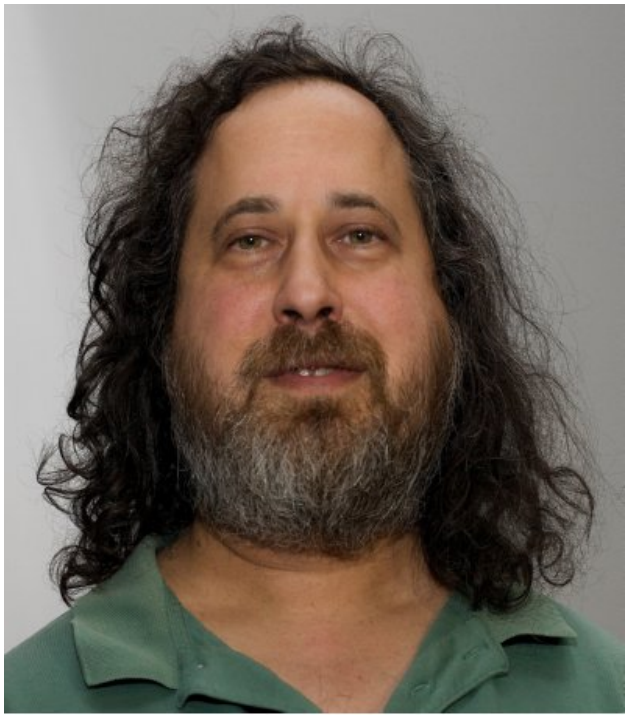
- History of Linux
- Why should I choose Linux?
- What is Linux made of (components, choices)
- How you can interact with/use a Linux system?
- The shell and command line interface
- Basic command line skills

History of Linux

Linus Torvalds

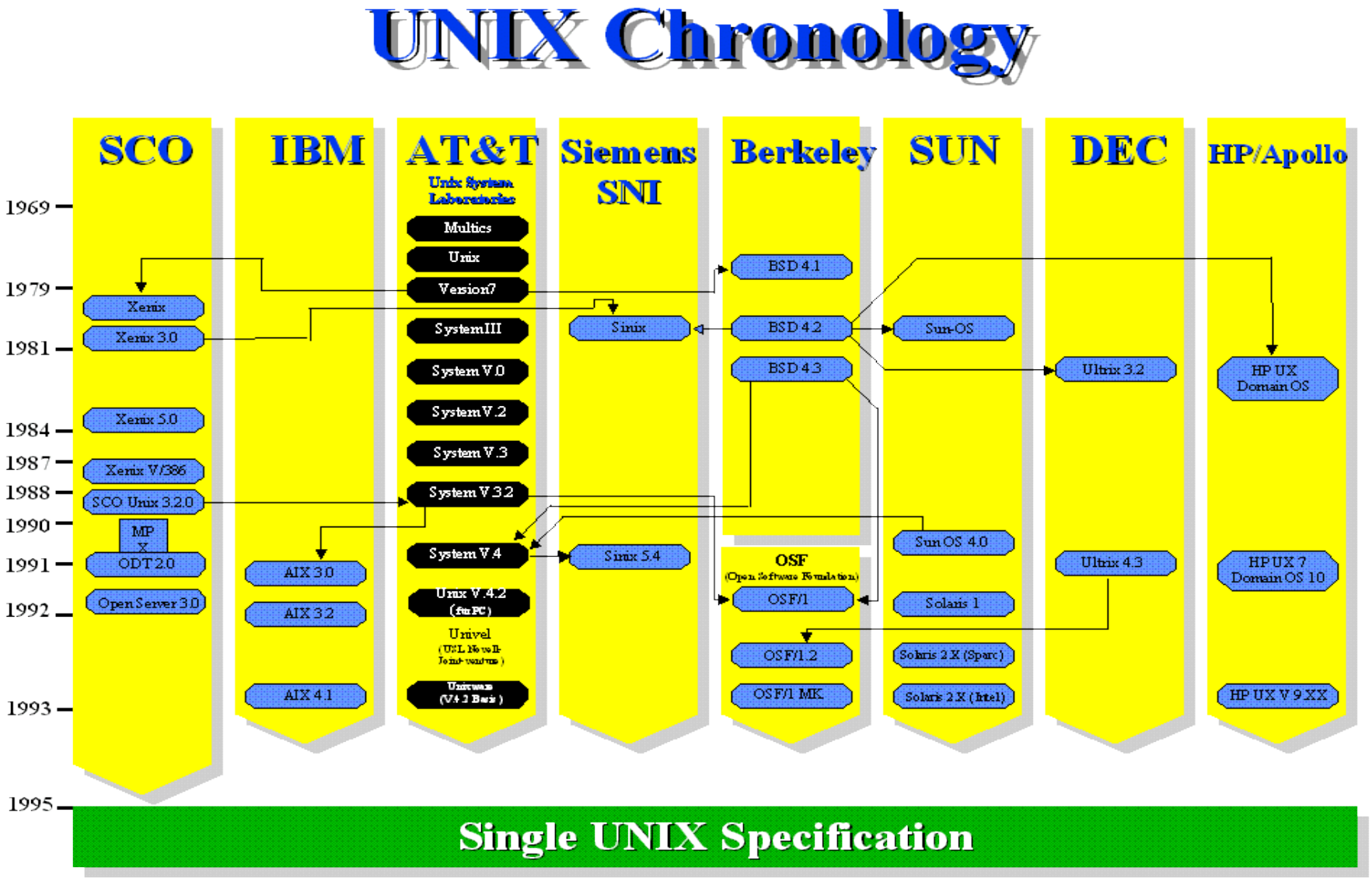
AT&T Bell Laboratories

FREE SOFTWARE FOUNDATION



Richard Stallman

History of Linux (2)



Courtesy of unix.org

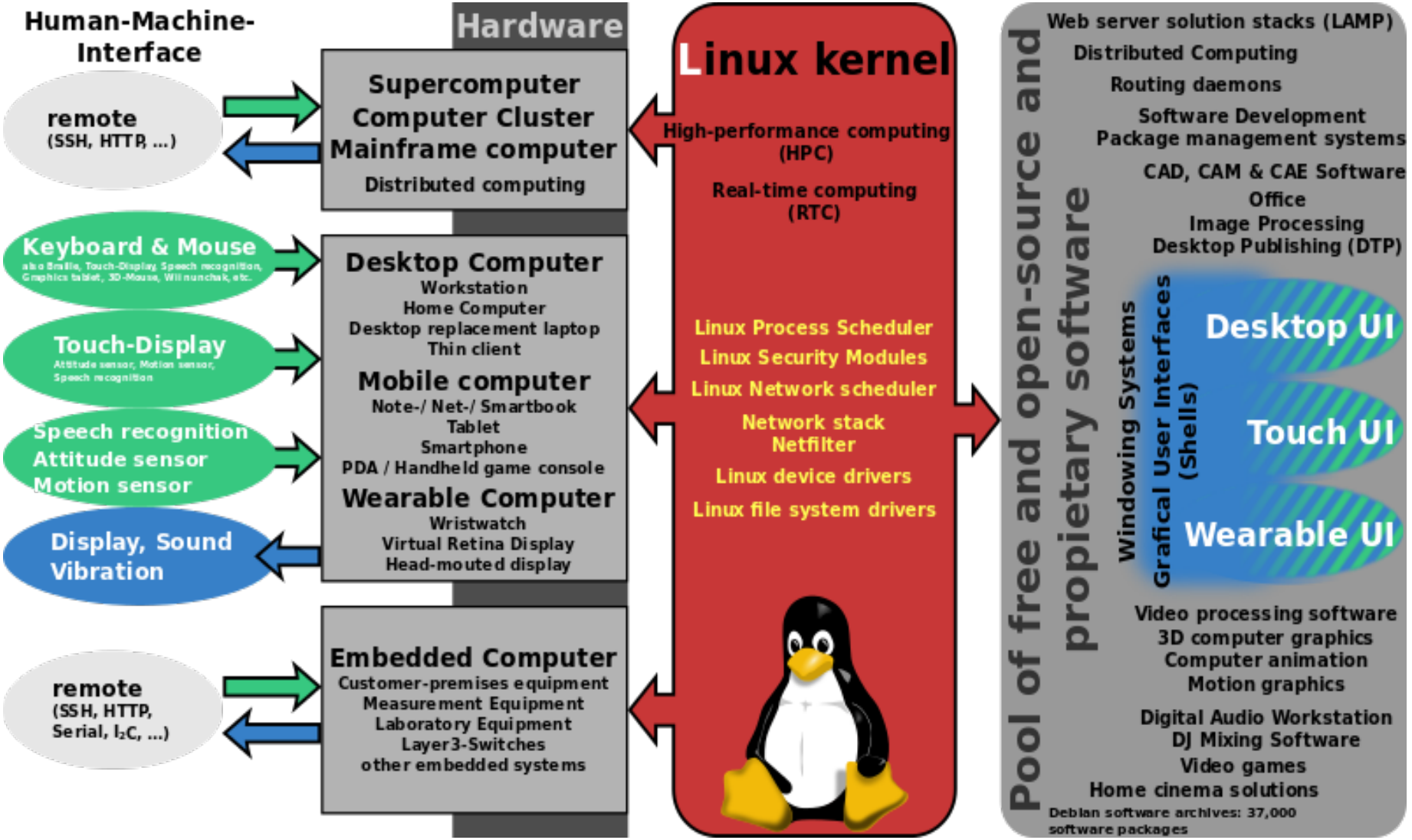
History of Linux (3)

- UNIX originated as a research project at AT&T Bell Labs in 1969 by Ken Thompson and Dennis Ritchie.
- The first multiuser and multitasking Operating System in the world.
- Developed in several different versions for various hardware platforms (Sun Sparc, Power PC, Motorola, HP RISC Processors).
- In 1991, a student at the University of Helsinki (Linus Torvalds) created a UNIX-like system to run on the Intel 386 processor. Intel had already started dominating the PC market, but UNIX was nearly absent from the initial processor Intel market.

Why should I choose Linux?

- Best price/performance ratio
- Reliable
- User friendly
- Ubiquitous (from your mobile phone to a supercomputer)
- Scientific software is developed mostly in Linux today.

What is Linux made of?



Linux distributions

- Often referred to as 'distros'.
- The Linux kernel with a set of programs/applications (text editors, compilers, office suites, web browsers, etc) that make the system usable.
- Slackware was one of the first Linux distributions.
- Debian, RedHat (Fedora, RHEL) and Canonical (Ubuntu) are some of the most popular ones today.

Linux distributions (2)

- There is a plethora of Linux distros out there, one of the strongest points of the Linux community.
- Which one to choose?
- General distributions: to replace your average desktop/server)
- Function specific distributions: They are tailored towards a specific audience (i.e. life science)

Linux distributions (3)

Generic distros:

- Redhat based: Fedora, RHEL, CentOS, Scientific Linux
- Debian based: Debian, Ubuntu

Or task-specific ones (tailored distributions):

- BioLinux
- BioKnoppix
- BioSLAX
- And many others

Package repositories

- Each Linux distro can connect to one or more package repositories
- They make it easy to search for/install/uninstall specific applications
- Package manager (yum, apt)
- “Find me all sequence analysis apps and install them”

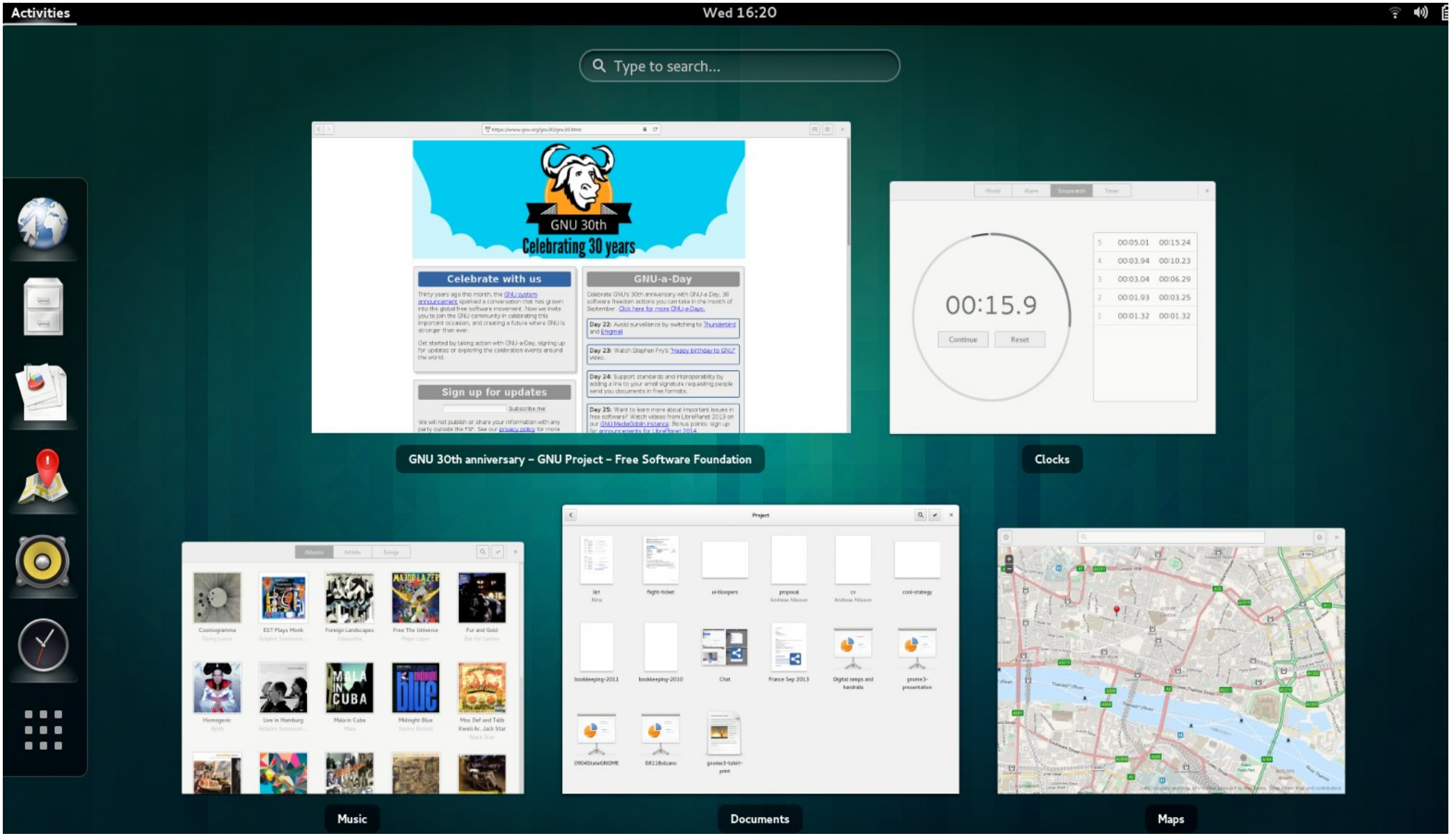
How to choose a Linux distro

- Try more than one to get a feeling.
- What do your colleagues/team members use?
- Do the package repositories have the applications you wish to use?
- How long the distro authors will keep maintaining it?
- Do you have a less common laptop/desktop that might have hardware compatibility problems with that distro? (rare but it happens)
- http://en.wikipedia.org/wiki/Linux_distribution

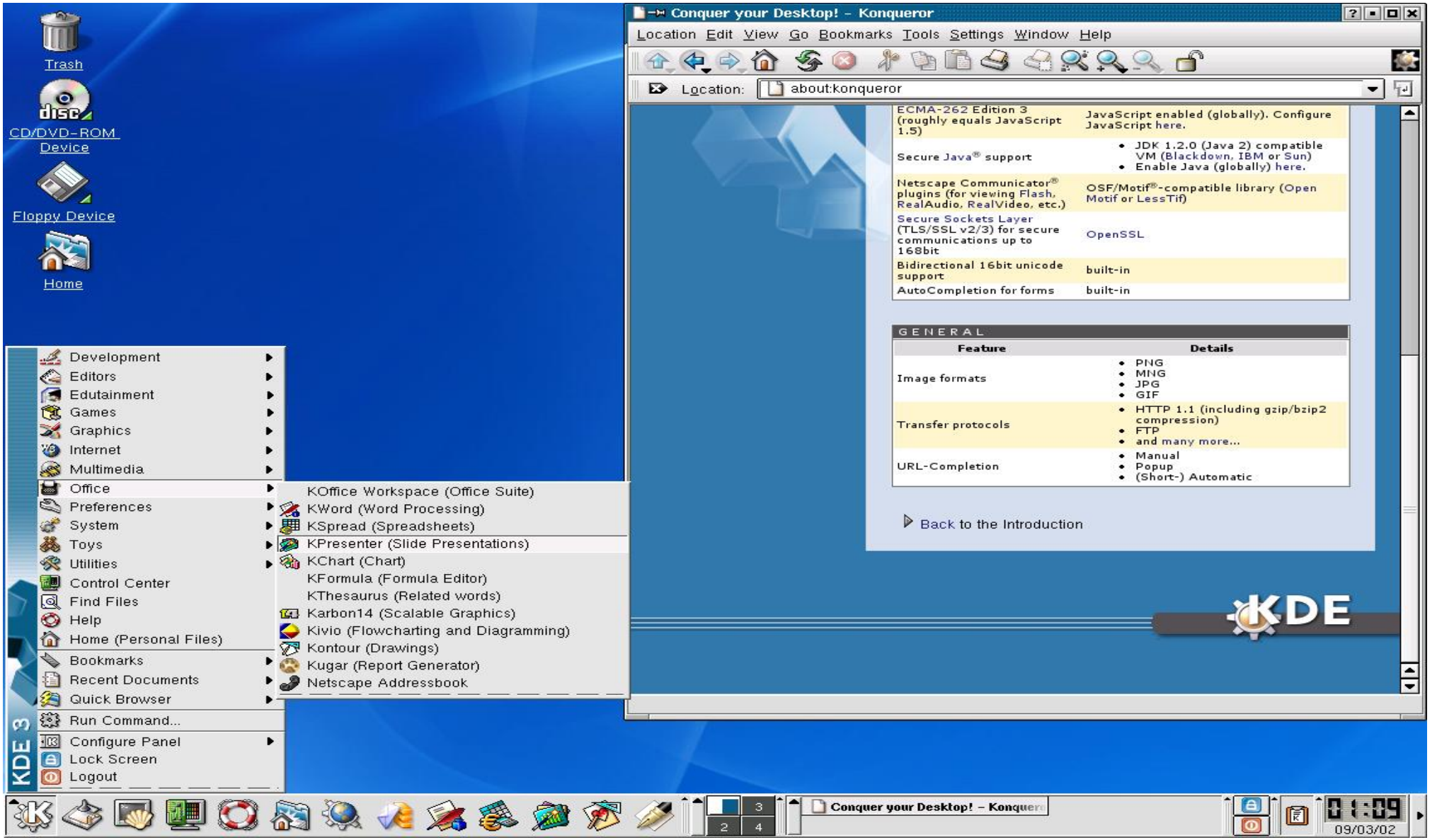
Interacting with Linux

- Using it via a Graphical User Interface (GUI) (aka Like Windows/Mac, your smartphone/tablet)
- Using it via the command line (like the PowerShell on Windows, or your Terminal window on your Mac)
- Pros and cons in each approach

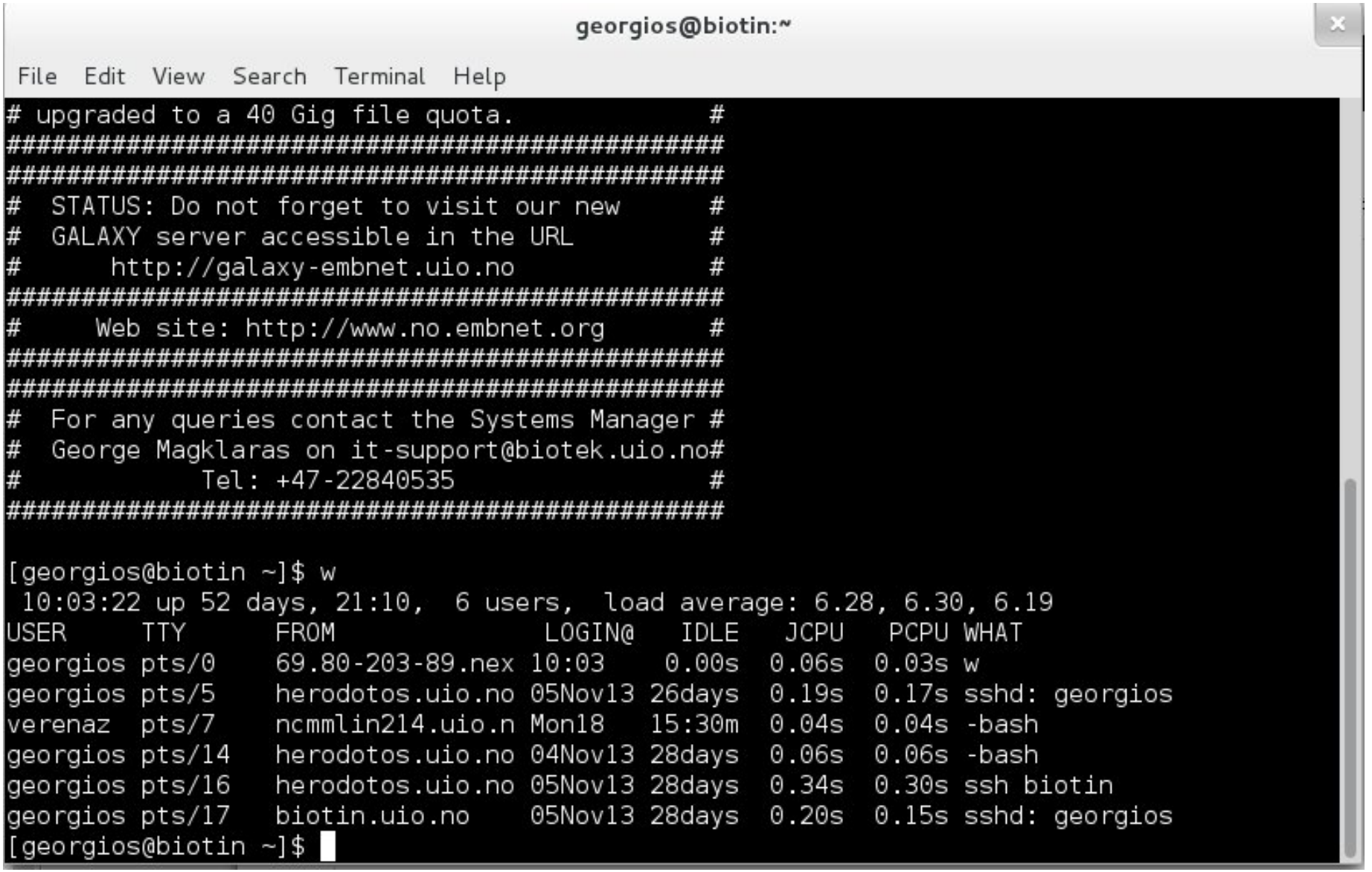
Linux GUI mode (GNOME)



Linux GUI mode (KDE)



Linux Command Line mode



So how to install/try Linux?

- Without affecting your current computer setup:
  - Use a Live CD (boot your computer from it)
  - Do a full installation of Linux on a virtual machine
- Links to distro Live CDs:
  - http://fedoraproject.org/wiki/FedoraLiveCD
  - http://www.debian.org/CD/live/
- Link to a video (install a Linux OS on Windows using VirtualBox):
  - https://www.youtube.com/watch?v=7jOnscRjaFs

Basic demo of a Linux system

- Objective: To demonstrate the GUI usage versus the command line/shell interface (5-10 min)

The shell and command line

- a powerful and productive tool: manipulates data and executes several applications under certain conditions.
- Comes under different flavours, but all of them do the same thing in slightly different ways.
- Essentially a program itself.
- In this course, we will be concerned with the 'Bash' shell. Other popular choices are the Tcsh,zsh and others.

Basic Shell Principles

- basic syntax for all commands executed at the shell:

command argument1 argument2 argument3...

'command' is the name of the actual shell command you wish to execute. Every command may take a certain number of arguments (or operands).

Example: **cd /storage/mydata**

Tip: Always make sure that you have a space between a shell command and its argument(s).

Basic Shell Principles (2):

- All UNIX shells are **case sensitive** with regards to both the commands and their arguments, in contrast to versions of Windows/DOS systems. This means that typing:

  cd /mydirectory/programs

  Is **not** the same as typing:

  CD /MYDIRECTORY/PROGRAMS

Tip: Usually, shell commands are lower case, unless otherwise stated.

The shell prompt

- The shell prompt is an indication that the system is ready to execute your commands, but it also gives you useful info:

georgios@biotin /usr/bin/virexp $

I am user 'georgios' at a server called 'biotin'. I am currently in a directory called 'virexp' that resides under a directory with name /usr/bin/. The $ sign says 'you can type now' and it should have a (sometimes blinking) cursor after it.

Shell ENVironment and execution PATH

- a collection of variables collectively known as the "**shell environment**" control a number of issues like the appearance of the shell prompt, what program might be your default text editor and many other issues.
- One of these variables is the "**execution path**": A list of directories that the shell remembers all the time, in order to automatically reference certain applications (without you remembering where they are). Type **echo $PATH** at the shell prompt to see this list of directories.

Filesystem basics

- Ever wondered how the computer keep tracks of your files?
- Imagine your dossier or file cabinet.
- You label your printed documents and you organize them in collections.
- Your computer does the same job with your electronic files using the 'filesystem'.

Filesystem basics (2)

- Files are named locations on the computer's storage device. Each filename points to a special filesystem record that contains information about:
  - The type of file (plain data, executable program, special device)
  - The user who created the file
  - Access permissions for the file
  - The beginning and end of the file record contents in the filesystem area, as well as its exact position in the filesystem.

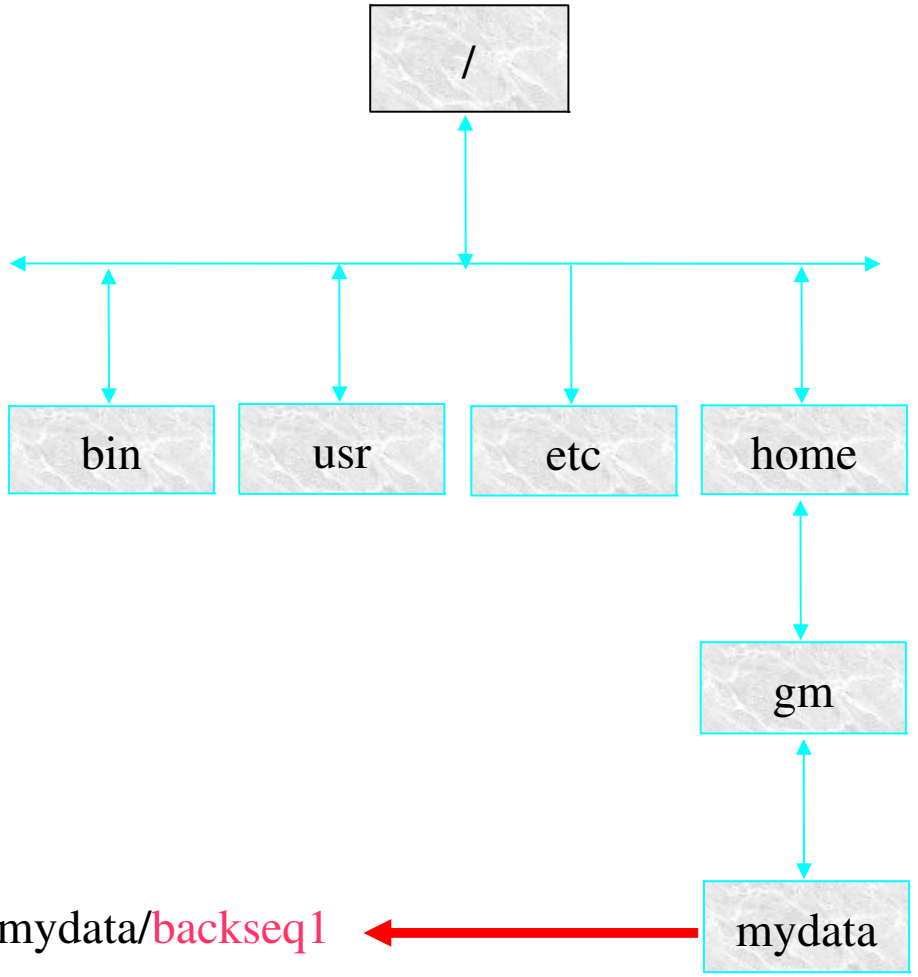
Filesystem Basics (3)

- Directories (or folders) are containers in which files can be grouped.
- They are arranged in hierarchical mode, starting from the top-level “root” directory (/). The root directory branches into several files and root subdirectories.
- The consequence of this hierarchy is that each file can be uniquely identified by a 'path'. A 'path' begins with a / (hint:root directory) and continues through a list of subdirectories, all the way down to the filename:

For example: **/home/gm/mydata/bac1.seq**

Tip: Remember not to confuse the term 'path' with the shell's execution path, as described in earlier slides.

Directory Hierarchy Diagram



Navigating the filesystem

- Use '**pwd**' to Print your Working Directory. For example, if I login to the host 'biotin' and I type pwd, I get the following:

```
georgios@biotin ~ $ pwd
/mn/biotroll/u1/georgios
georgios@biotin ~ $
```

This means that I am currently in a directory 'georgios', which is under a directory called 'u1'. This directory itself is under the 'biotroll' directory, which lives under the 'mn' directory. Finally the mn directory is under the root (toplevel) directory.

Navigating the filesystem (2)

- Your 'home' directory is the folder you are situated when you first login to the Linux system shell.
- Usually under **/home/username** (for example: **/home/georgios**)
- Your home directory is also symbolized by **~**
- Instead of typing /home/georgios, you could just type ~

Tip: Typing less by using well known symbols saves you time.

Navigating the filesystem (3)

- Your instructor says: “Under your home directory, you will find a directory called “mysequences. Could you go to that directory and tell me what kind of files exist under it?”

georgios@biotin ~ $ cd mysequences

georgios@biotin ~/mysequences $

Navigating the filesystem (4)

- The “cd” command (Change Directory) can be used for moving around the filesystem. It takes a path as its argument.
- The path can be “**absolute**”. For example:From your home directory, you can go to the /usr/bin directory by typing:

georgios@biotin ~ $ cd /usr/bin

georgios@biotin /usr/bin $

- The path can also be “relative”. For example: If you are already under the /usr directory, you could just type:

georgios@biotin /usr $ cd bin

georgios@biotin /usr/bin $

Navigating the filesystem (5)

- The command "**cd ..**" will get you one level up. For example, if we go back to slide 30 and we assume that you are under the 'mysequences' directory, if you want to go back to the toplevel of your home directory, you type:

georgios@biotin ~/mysequences $ cd ..

georgios@biotin ~ $

- "**..**" is a shorthand notation for the previous directory level and it can really save you from typing long directory names that you cannot remember. It always works in a relative path context.
- The alternative would be to give an "absolute" path to the cd command:

georgios@biotin ~/mysequences $ cd /mn/biotroll/u1/georgios

georgios@biotin ~ $

Listing files

- You are back at the mysequences directory under your home directory. Your instructor asked you to list the files in the directory:

```
georgios@biotin ~/mysequences $ ls

seqdocs  v2.3_admin.pdf  xlrhodop.fasta

georgios@biotin ~/mysequences $
```

- The **ls** command lists all the directory contents and is the equivalent of the dir command in DOS/Windows.

Listing Files (2)

- The instructor says: “That's not good enough. I want details (file size, permissions, etc). Why don't you use the -la options of the ls command?”

georgios@biotin ~/mysequences $ ls -la

total 340

drwx------ 3 georgios biotek 62 Mar 26 16:31 .

drwx--x--x 63 georgios biotek 8192 Mar 28 08:45 ..

drwx------ 2 georgios biotek 6 Mar 26 16:31 seqdocs

-rw------- 1 georgios biotek 325479 Mar 26 15:22 v2.3_admin.pdf

-rwxrw---- 1 georgios biotek 1777 Mar 26 15:22 xlrhodop.fasta

Locating files in the directory tree

- A colleague says: “Help! I have placed a file called xlrhodop.fast or xlrhodop.fasta (I can't remember the name) and now I can't find it. Can you help me locate it?”

find [starting point] -name filename -print

'starting point' indicates the directory tree position that we wish to start searching. 'Filename' could be an approximation of the file name (it doesn't have to be exact).

Listing Files (2)

- The instructor says: “That's not good enough. I want details (file size, permissions, etc). Why don't you use the -la options of the ls command?”

georgios@biotin ~/mysequences $ ls -la

total 340

drwx------ 3 georgios biotek 62 Mar 26 16:31 .

drwx--x--x 63 georgios biotek 8192 Mar 28 08:45 ..

drwx------ 2 georgios biotek 6 Mar 26 16:31 seqdocs

-rw------- 1 georgios biotek 325479 Mar 26 15:22 v2.3_admin.pdf

-rwxrw---- 1 georgios biotek 1777 Mar 26 15:22 xlrhodop.fasta

Locating filenames in the directory tree (2)

georgios@biotin ~ $ find ~/ -name xlrhodop.fas*

/mn/biotroll/u1/georgios/xlrhodop.fasta

/mn/biotroll/u1/georgios/mysequences/xlrhodop.fasta

- Note that the wildcard character (*) towards the end of the filename we are trying to search for. This says that we know that the name contains the string “xlrhodop.fas”. This would match all relevant filenames (reporting their exact location in the directory tree)

  /mn/biotroll/u1/georgios/xlrhodop.fasta

  /mn/biotroll/u1/georgios/mysequences/xlrhodop.fasta

File permissions (1)

- Every file in UNIX has a set of permission flags that define in a strict way, who is allowed to read, write (modify) or execute that file.For example, let's take one of the listed files of the ls -la output command:

-rwx------ 1 georgios biotek 325479 Mar 26 15:22 v2.3_admin.pdf

Starting from the left, this says: The file xlrhodop.fasta can be read (r)read, (w)modified,(x)executed by its owner (georgios). Ignore the rest of the flags for now.

File permissions (2)

- Directories are no exception to this rule and they also have permission flags. For example:

drwx------ 2 georgios biotek 6 Mar 26 16:31 seqdocs

- Note the leftmost flag (d). This indicates that 'seqdocs' is a directory and user georgios has full permissions (read, write and execute) for that directory. Hence, what we say about file permissions is true for directory permissions with a few exceptions (see special file permission consideration slides).

Changing File Permissions (1)

- Your colleague says “The file v2.3_admin.pdf is quite important and should not be modified. Can we have it as read only please? Use the chmod (change mode) command.”
- The generic syntax for the chmod command is:

chmod [u|g|o (+|-) (r,w,x)] [filename]

DON'T PANIC! We will explain this cryptic syntax with some examples!

Changing File Permissions (2):

- The file permissions were:

-rw------- 1 georgios biotek 325479 Mar 26 15:22 v2.3_admin.pdf

Thus, in order to make the file read only we need to remove the (w) flag. We type at the prompt:

georgios@biotin ~/mysequences $ chmod u-w v2.3_admin.pdf

Changing File Permissions (3):

- If we wanted to add back the write permission flag, we would type:

georgios@biotin ~/mysequences $ chmod u+w v2.3_admin.pdf

The + sign says add write permissions (w) for the user (u) that owns the file.

- You can also add/remove more than one flag at a time:

georgios@biotin ~/mysequences $ chmod u-wx v2.3_admin.pdf

This would remove write (w) and execute permissions (x).

The execute permission

- The execute permission is important when you are dealing with programs that you wish to run. In order to run those programs, you will always have to set the (x) permission flag

chmod u+x program_name

Tip: Remember this rule, before you try to run a program in the command line environment.

The execute permission on directories

- When changing permissions for directories, you will need to enable the x flag, in order to allow access to the directory. Read permission is not enough to allow access to the directory.

Try: **chmod u+rx dir_name**

Tip: This is often a confusing concept for beginners.

Deleting files:

- Given the right permissions, you can remove a file using the rm command. If, for example, you have a file named testfile.fasta and you want to remove it, you type:

  georgios@biotin ~/mysequences $ rm testfile.fasta

CAUTION: Take great care when you use the **rm** command. Whatever you delete, you WILL NOT BE ABLE TO UNDELETE. There is no “Recycle Bin/Wastebasket” in command line UNIX.

Viewing file contents

- Your colleague says: “How do I view the contents of a file? I want a simple shell command that will show the file contents.”
- The **cat** command is probably one of the most frequently used commands. It displays the contents of the file. For example:

cat xlrhodop.fasta

- will display the contents of the file xlrhodop.fasta on the screen
- An alternative way of viewing the file contents is to use a text editor. We are going to cover the basics of text-editors in the tutorial later in the course.

Viewing file contents (2)

- If you use the cat command and you see something like this:

000731 (Red H▒├ Li┼┤ │ 7.2 2.96-
10701.001.001.001.001.001.001.001.001.001.001.001.001.01.sy └┼▒
b.s├r ┼▒b.shs├r ┼▒b.i┼├erp.┼o ├e.ABI- ┼▒±.h▒sh.dy┼sy └.dy┼s ├r.±┼┤.┬ersio┼

You are looking at the contents of a binary file which contain special (non readable) characters. To filter these characters, you can also use the string command:

Try: **strings xlrhodop.fasta**

Viewing file contents (3)

- Your colleague says: “Ohh! I tried to use cat to view a file but the output is too long for my terminal screen. The text keeps scrolling and I loose the first lines of the text. Can I stop this somehow?”
- The **less** command can actually allow you to view a file, but it will stop the scrolling of the output, when your terminal window is filled.

```
less xlrhodop.fasta
```

- The **more** command would do exactly the same thing.

Viewing File Contents (4)

- Alternatively, if you suspect that the information you want to retrieve is towards the beginning or the end of the file, you can use head:

  head xlrhodop.fasta

  This displays the beginning of the file.
- On the other hand, tail can display the end of the file.

  tail xlrhodop.fasta
- Both of these commands can be tailored to display a certain number of lines from the beginning (head) or the end (tail of the file):

  head -3 xlrhodop.fasta -> displays the first 3 lines of the file

  tail -3 xlrhodop.fasta -> displays the last 3 lines of the file

Creating Directories

- BB says: “We need a new directory to store all the pdf documents. Could you create a new directory called pdfdoc under the mysequences directory?”

georgios@biotin ~/mysequences $ mkdir pdfdoc

georgios@biotin ~/mysequences $ ls -la

drwx------ 4 georgios biotek 75 Mar 28 15:15 .

drwx--x--x 63 georgios biotek 8192 Mar 28 14:53 ..

drwx------ 2 georgios biotek 6 Mar 28 15:15 pdfdoc

drwx------ 2 georgios biotek 6 Mar 26 16:31 seqdocs

-r-------- 1 georgios biotek 325479 Mar 26 15:22 v2.3_admin.pdf

Removing Directories

“What about the seqdocs directory? Delete it using the rmdir command”, the instructor replies.

georgios@biotin ~/mysequences $ rmdir seqdocs

So your directory structure should now look like this.

drwx------ 3 georgios biotek 61 Mar 28 15:25 .

drwx--x--x 63 georgios biotek 8192 Mar 28 14:53 ..

drwx------ 2 georgios biotek 6 Mar 28 15:15 pdfdoc

-r-------- 1 georgios biotek 325479 Mar 26 15:22 v2.3_admin.pdf

Removing Directories (2)

- The 'rmdir' command will promptly remove a directory if and only if it is empty. If the directory you are trying to remove (example:pfddoc) contains files, rmdir will fail with the following error message:

rmdir: `pdfdoc': File exists

You then have to delete all the files under the directory pdfdoc and then issue the rmdir command.

- The alternative would be to use the rm command. Remember, directories are 'special' files, so you could remove them with rm. The next slide shows you how.

Removing Directories (3)

rm -r -f [directory name]

The -r option says delete directories recursively. The -f option forces the command to go ahead, despite the fact that the target is a directory and has files under it. Both options are required. For example, in order to delete a directory pdfdoc under the ~/mysequences directory, you would type:

rm -r -f pdfdoc/

CAUTION: The usage of 'rm' in this way is even more dangerous, because it will delete EVERYTHING at a selected directory tree point, all the way down to the leaf nodes. Always check where you are with 'pwd' first. If you delete the files, they will be gone forever!

Copying Files

Your instructor says :”Under the ~/mysequences directory there is a file called v2.3_admin.pdf . Could you make another copy of that file with the name 23adminbeta.pdf ?”

You can now use the cp command. The command's general syntax is:

cp [sourcefilepath] [destfilepath]

sourcefilepath:absolute or relative path of the file we want to copy.

destfilepath:absolute or relative path of the new file. This might include a new filename. If you specify a different directory for the new destination file and NOT a filename, the source file's name is used by default.

Some examples to illustrate these points follow.

Copying Files (2)

cp v2.3_admin.pdf 23adminbeta.pdf

As a result, we should now have two files with exactly identical contents. Note that the size and the permission contents indicate that the files are identical.

-r-------- 1 georgios biotek 325479 Mar 28 17:01 23adminbeta.pdf

-r-------- 1 georgios biotek 325479 Mar 26 15:22 v2.3_admin.pdf

Also note that 'cp' was executed this time with relative paths for the source and destination files.

Copying Files (3)

- ”Could you make a copy of the v2.3_admin.pdf file into the pdfdoc directory with the name 23adminbeta.pdf“, you could then type:

cp v2.3_admin.pdf perldoc/23adminbeta.pdf

- By default, the 'cp' command preserves the permissions and ownership rights of files. If in doubt, use the -p flag. This situation can occur when performing a copy of the file from computer to computer using specialist Filesystems such as NFS..

Copying Directories:

You could copy entire directories recursively (including any files and their entire subdirectories) by using the 'cp' command

cp -p -r pdfdoc/ pdfcopy/

The -p flag preserves the permission and ownership properties and the -r instructs copy to copy all subdirectories under pdfdoc (recursive copy).

Moving Files

Sometimes we wish to move the file, in that we wish to copy the file to a new location without preserving the old one. This is when we can use the mv command, with the following syntax:

mv sourcefilepath destfilepath

sourcefilepath: absolute or relative path of the file we want to copy.

destfilepath: absolute or relative path of the new file. This might include a new filename. If you specify a different directory for the new destination file and NOT a filename, the source file's name is used by default.

Moving (or Renaming) Files

mv xlrhodop.fasta myxlr.fasta

This removes the xlrhodop.fasta file and re-generates it with the name myxlr.fasta, under the same directory.

-r-------- 1 georgios biotek 1777 Mar 26 15:22 myxlr.fasta

'mv' does not only preserve file permissions and ownership rights but it does also preserve timestamps, so it is an effective way to rename a file. The UNIX shell has a rename command, but mv could be used effectively to rename a file.

Tip: All the points we have made about mv for files are also true for directories.

Redirecting command output

- The **>** symbol is the output redirection operator and can be used to re-direct the output of any UNIX command that prints something on the screen.
- Lets suppose that you want to merge two fasta sequences into a single file:

cat myseq1.fasta myseq2.fasta

would print the contents of both files one-after the other on the screen (stdout). But what you really want is to place this output to a file. You can then type:

cat myseq1.fasta myseq2.fasta > mergedseq.fasta

to place the output in the file mergedseq.fasta .

Redirecting command input

- Suppose that you have a file with numbers and you wish to sort it from the smaller to the larger number

sort -g < numbers.txt

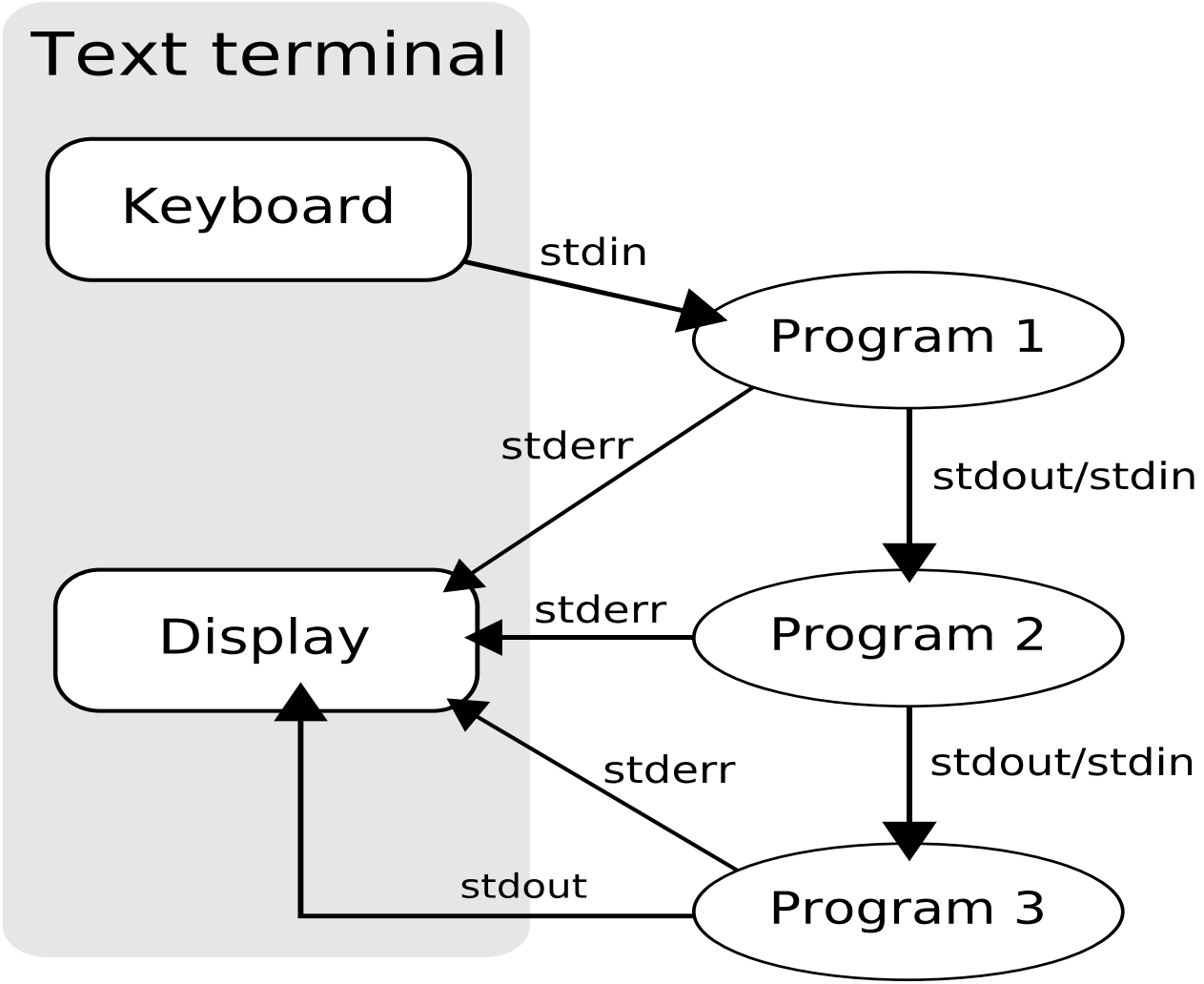
- Normally, 'sort' would take its input from the keyboard. However, because you use the input redirection symbol (<), it is like typing the contents of the file (numbers) in one step.
- Bottom line: You get your numbers sorted.
- Question: What do you think about this command?

sort -g < numbers.txt > sortednumbers.txt

The Shell Pipe

- Do you ever wonder how the term 'pipeline' was established in computing/bioinformatics context?
- One of the most powerful concepts of the command line environment.
- The more you learn to use it, the more you will appreciate its power.
- Mastering the shell pipe will allow you to build very powerful processing utilities to solve your problems.

The Shell Pipe (2)



The Shell Pipe (3)

- Quite often, we need to direct the standard output of one command to the standard input of another.
- The most commonly used operator to do that is the pipe oparator |
- Suppose for example that we need to count the number of lines of a text file to see how long it is.

cat mytext.txt | wc -l

The 'cat' command will print all the lines of the file. However, instead of doing that on the screen, it gives all the output to the 'wc -l' command. The result is an integer representing the number of lines of the mytext.txt file.

References

- UNIX has a built-in reference manual. The 'man' command should be you best friend, whenever you need help for a particular command. For example, type

```
man cat
```

Every UNIX system should have this facility.

References (2)

- What if you don't know which command to use? Let's say for example that I am looking for pattern matching commands. I would type

apropos pattern

at the shell prompt, and this would give me a list of relevant commands

References (3)

- University of Surrey Unix Tutorial for Beginners on the World Wide Web:

  http://www.ee.surrey.ac.uk/Teaching/Unix/

- “Developing Bioinformatics Computer Skills”, O'REILLY PRESS, ISBN: 1-56592-664-1, useful for Biologists and Bioinformaticians, especially for beginners.

http://www.amazon.co.uk/Developing-Bioinformatics-Computer-Skills-Cynthia/dp/1565926641

- The EMBnet Unix/Linux Quick Guide:

  http://www.embnet.org/sites/default/files/quickguides/guideUNIX.pdf