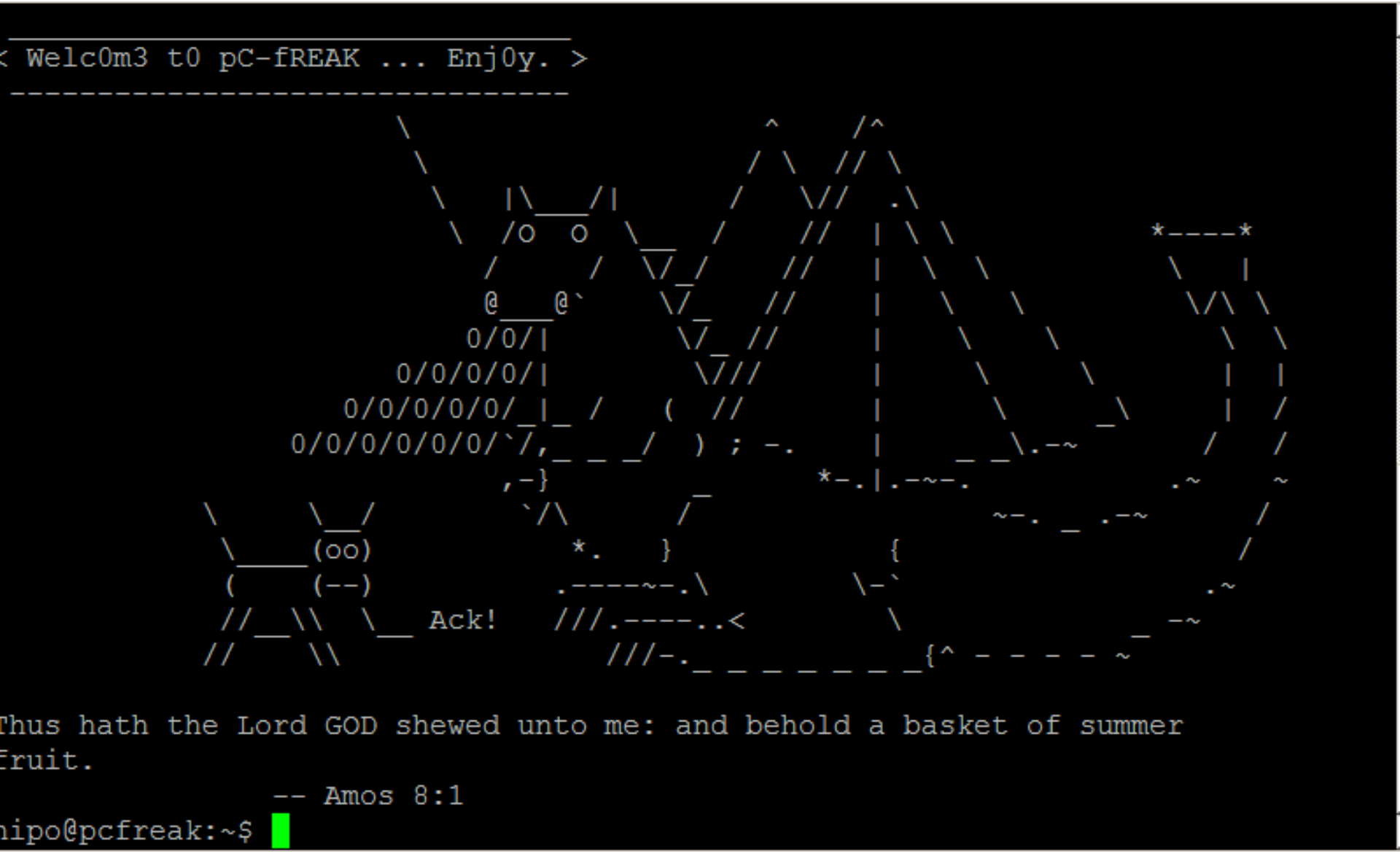
Introduction to Unix:
Getting to Grips with the Command Line

Dr. James Emmanuel San
CERI-KRISP, UKZN
sanemmanueljames@gmail.com

The terminal is your best ally….



…use it to your best advantage!

What We're Going To Do

THIS MORNING

- Introduction to Unix
- The terminal
- Software Installation
- Package management
- Linux environments

Good Workshop Practice

- PowerPoint interspersed with Challenges
- Ask lots of questions!
- Work together
- Take breaks
- Cheat!

Unix/Linux Command Reference

FOSSwire.com

File Commands

ls – directory listing
ls -al – formatted listing with hidden files
cd *dir* - change directory to *dir*
cd – change to home
pwd – show current directory
mkdir *dir* – create a directory *dir*
rm *file* – delete *file*
rm -r *dir* – delete directory *dir*
rm -f *file* – force remove *file*
rm -rf *dir* – force remove directory *dir* *
cp *file1 file2* – copy *file1* to *file2*
cp -r *dir1 dir2* – copy *dir1* to *dir2*; create *dir2* if it doesn't exist
mv *file1 file2* – rename or move *file1* to *file2* if *file2* is an existing directory, moves *file1* into directory *file2*
ln -s *file link* – create symbolic link *link* to *file*
touch *file* – create or update *file*
cat > *file* – places standard input into *file*
more *file* – output the contents of *file*
head *file* – output the first 10 lines of *file*
tail *file* – output the last 10 lines of *file*
tail -f *file* – output the contents of *file* as it grows, starting with the last 10 lines

Process Management

ps – display your currently active processes
top – display all running processes
kill *pid* – kill process id *pid*
killall *proc* – kill all processes named *proc* *
bg – lists stopped or background jobs; resume a stopped job in the background
fg – brings the most recent job to foreground
fg *n* – brings job *n* to the foreground

File Permissions

chmod *octal file* – change the permissions of *file* to *octal*, which can be found separately for user, group, and world by adding:

- 4 – read (r)
- 2 – write (w)
- 1 – execute (x)

Examples:
chmod 777 – read, write, execute for all
chmod 755 – rwx for owner, rx for group and world
For more options, see **man chmod**.

SSH

ssh *user@host* – connect to *host* as *user*
ssh -p *port user@host* – connect to *host* on port *port* as *user*
ssh-copy-id *user@host* – add your key to *host* for *user* to enable a keyed or passwordless login

Searching

grep *pattern files* – search for *pattern* in *files*
grep -r *pattern dir* – search recursively for *pattern* in *dir*
command* | grep *pattern – search for *pattern* in the output of *command*
locate *file* – find all instances of *file*

System Info

date – show the current date and time
cal – show this month's calendar
uptime – show current uptime
w – display who is online
whoami – who you are logged in as
finger *user* – display information about *user*
uname -a – show kernel information
cat /proc/cpuinfo – cpu information
cat /proc/meminfo – memory information
man *command* – show the manual for *command*
df – show disk usage
du – show directory space usage
free – show memory and swap usage
whereis *app* – show possible locations of *app*
which *app* – show which *app* will be run by default

Compression

tar cf *file.tar files* – create a tar named *file.tar* containing *files*
tar xf *file.tar* – extract the files from *file.tar*
tar czf *file.tar.gz files* – create a tar with Gzip compression
tar xzf *file.tar.gz* – extract a tar using Gzip
tar cjf *file.tar.bz2* – create a tar with Bzip2 compression
tar xjf *file.tar.bz2* – extract a tar using Bzip2
gzip *file* – compresses *file* and renames it to *file.gz*
gzip -d *file.gz* – decompresses *file.gz* back to *file*

Network

ping *host* – ping *host* and output results
whois *domain* – get whois information for *domain*
dig *domain* – get DNS information for *domain*
dig -x *host* – reverse lookup *host*
wget *file* – download *file*
wget -c *file* – continue a stopped download

Installation

Install from source:
./configure
make
make install
dpkg -i *pkg.deb* – install a package (Debian)
rpm -Uvh *pkg.rpm* – install a package (RPM)

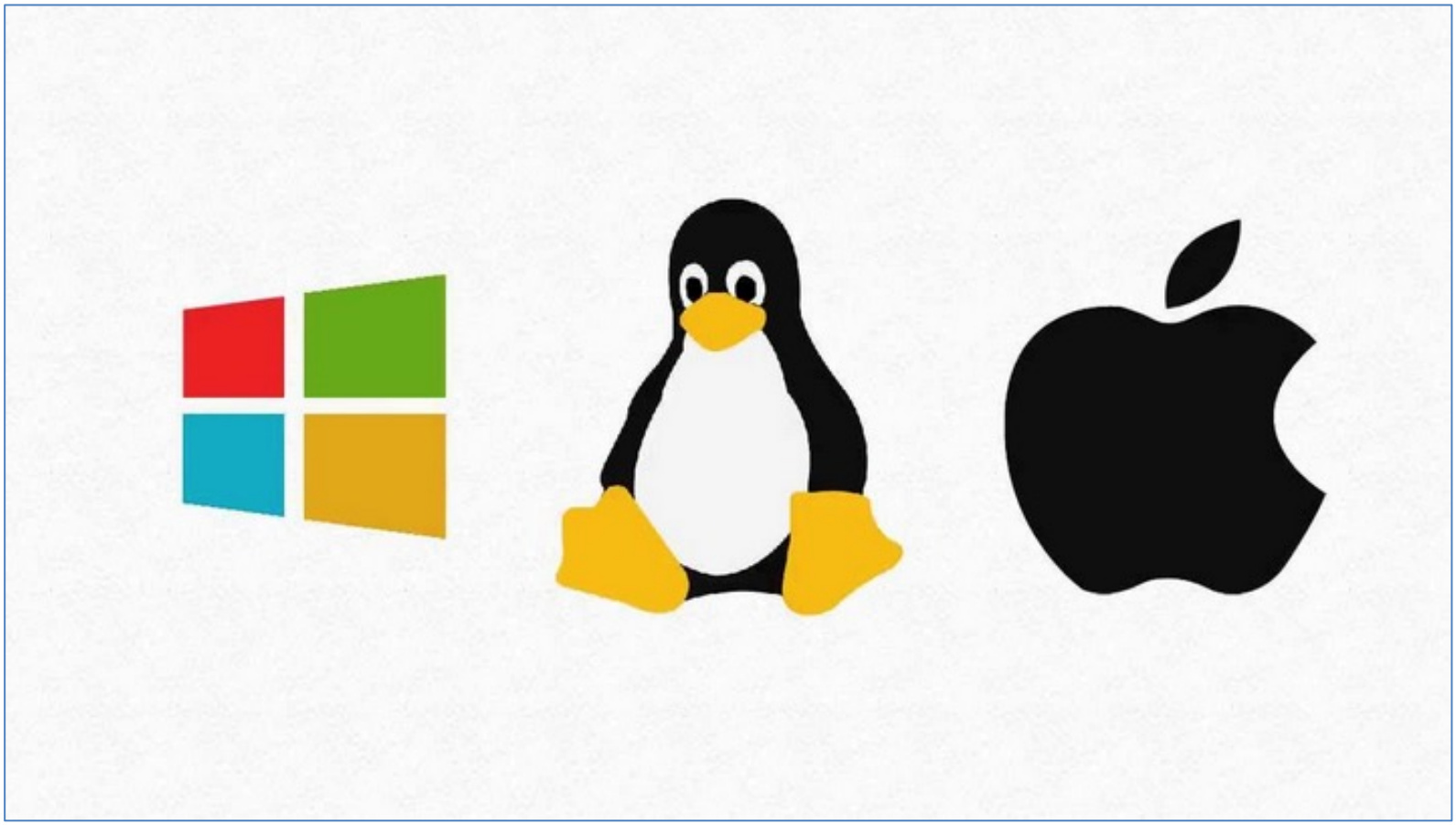
Shortcuts

Ctrl+C – halts the current command
Ctrl+Z – stops the current command, resume with **fg** in the foreground or **bg** in the background
Ctrl+D – log out of current session, similar to **exit**
Ctrl+W – erases one word in the current line
Ctrl+U – erases the whole line
Ctrl+R – type to bring up a recent command
!! - repeats the last command
exit – log out of current session

* use with extreme caution.

What is Unix?

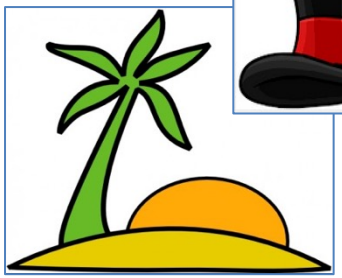
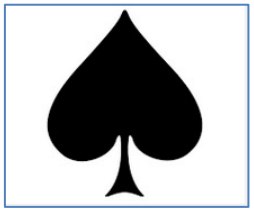
- An operating System



Why Unix?

- Most bioinformatics software designed to run on Unix platforms.
- Large amounts of data.
- Much faster than your Windows PC.

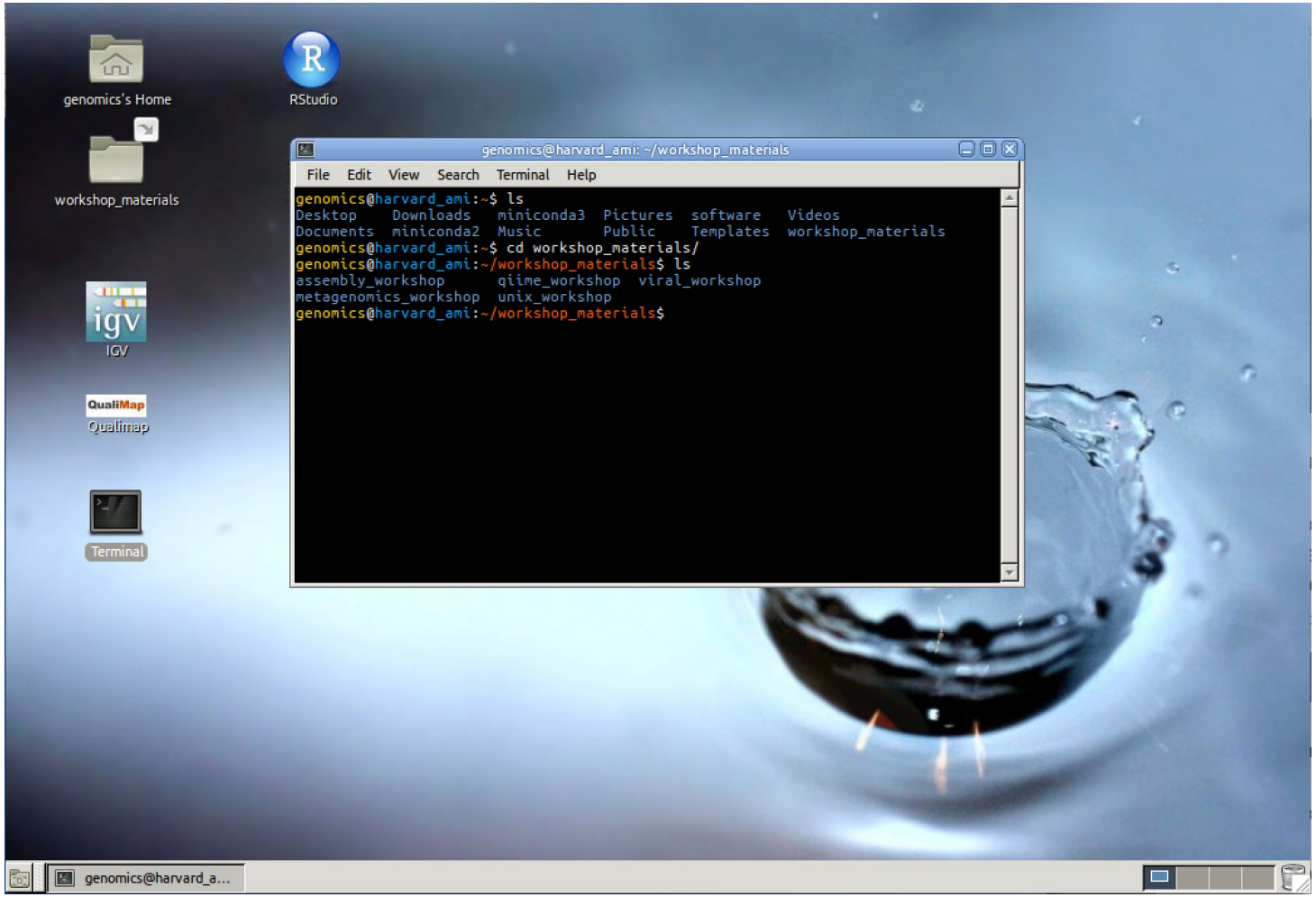





The Terminal

- Text interface the the computer
- Can also be used to connect to a remote computer



The Terminal

- Most IDEs today integrate a Linux terminal

app.terra.bio/#workspaces/workshop-south-africa/workshop_testing/applications/RStudio

Terra BETA WORKSPACES

Workspaces › workshop-south-africa/workshop_testing › RStudio

COVID-19 Data & Tools

DASHBOARD DATA ANALYSES WORKFLOWS JOB HISTORY

File Edit Code View Plots Session Build Debug Profile Tools Help

rstudio

Go to file/function Addins Project: (None)

testing.Rmd data-wrangling-1.Rmd

Run Document Run

Source Visual Outline

```
1  ---
2  ##############################################################
3  #                                                            #
4  # Click on "Run Document" in RStudio to run this worksheet. #
5  #                                                            #
6  ##############################################################
7  title: "Data wrangling 1"
8  author: "Claus O. Wilke"
9  output: learnr::tutorial
10 runtime: shiny_prerendered
11 ---
12
```

1:1 Data wrangling 1 R Markdown

Console Terminal Jobs

Terminal 1 /home/rstudio

```
rstudio@650a7121b06a:~$ ls
data-wrangling-1.Rmd  lost+found  R  testing.Rmd
rstudio@650a7121b06a:~$
```

Environment History Connections Tutorial

Import Dataset 194 MiB List

R Global Environment

Environment is empty

Files Plots Packages Help Viewer

New Folder New Blank File Upload Delete Rename More

Home

Name	Size	Modified
data-wrangling-1.Rmd	11.6 KB	Aug 19, 2022, 5:12 PM
lost+found		
R		
testing.Rmd	108 B	Aug 19, 2022, 5:12 PM

Rate: $0.07 per hour

For the rest of this workshop, we will access the terminal from Rstudio in Terra!

The Terminal

Console | Terminal | Jobs

Terminal 1 | /home/rstudio

```
rstudio@650a7121b06a:~$ ls
data-wrangling-1.Rmd  lost+found  R  testing.Rmd
rstudio@650a7121b06a:~$
```

Is also refered to by several names including; the Command Line, the Shell, the Command Prompt

Where you see this “$” followed by text, I want you to type the text on your command line

Useful commands to get you going

Location – Where am I?

```
$ pwd
/home/rstudio
```

This is your “present working directory”.

/ = ROOT

home

bin

lib

...

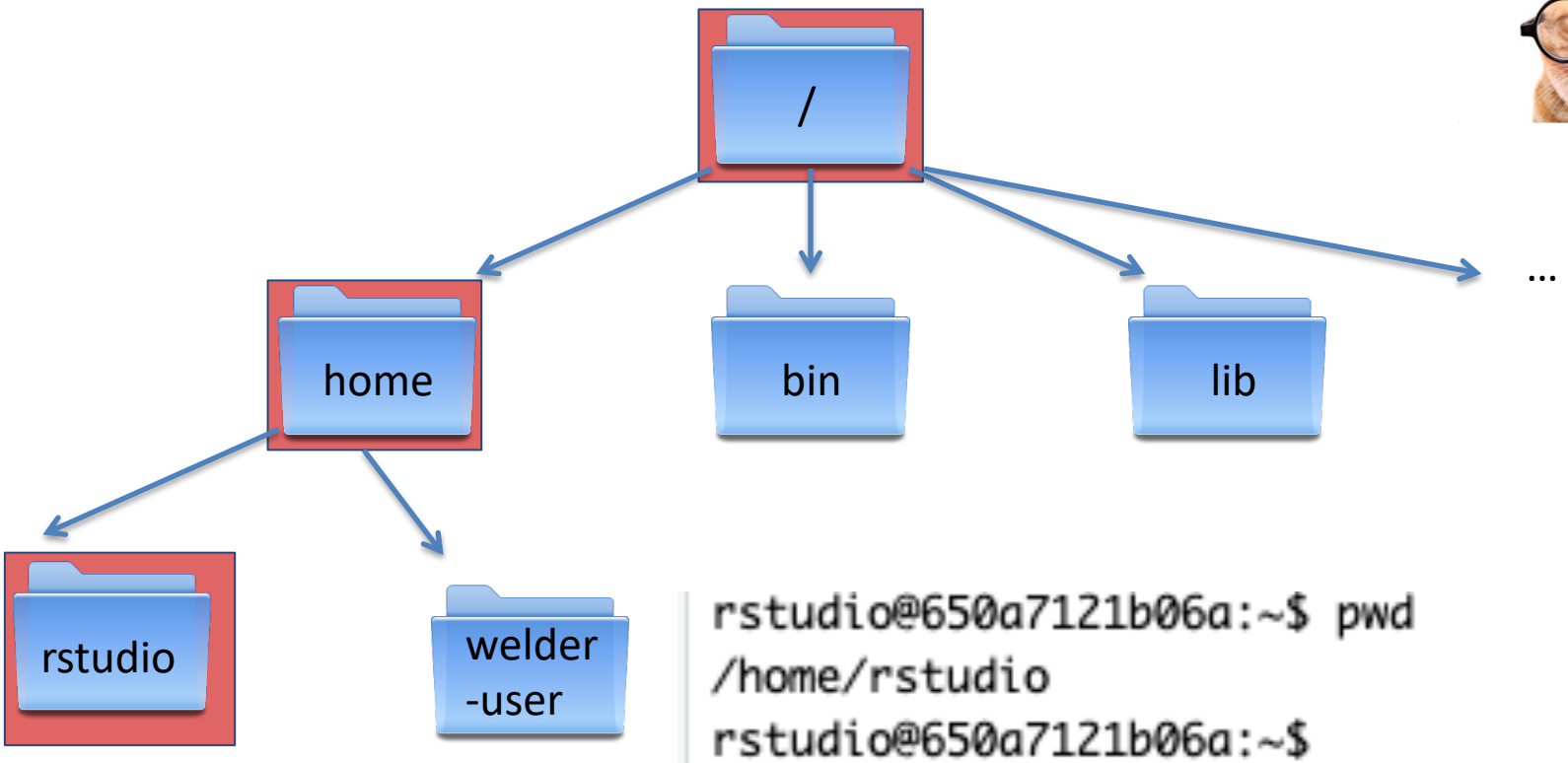
rstudio

welder
-user

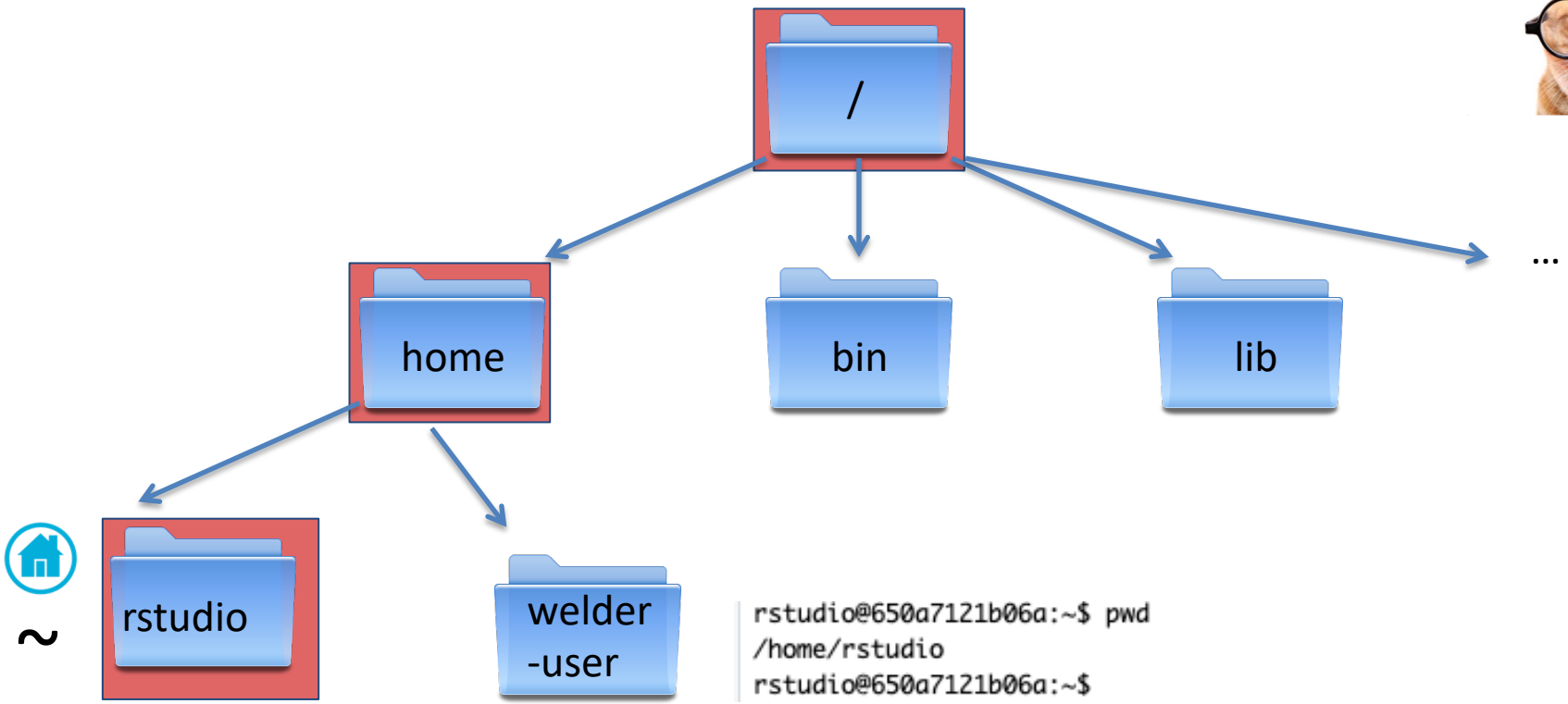
Some of the directories you will likely interact with

/

/bin	User Binaries
/sbin	System Binaries
/etc	Configuration Files
/dev	Device Files
/proc	Process Information
/var	Variable Files
/tmp	Temporary Files
/usr	User Programs
/home	Home Directories
/boot	Boot Loader Files
/lib	System Libraries
/opt	Optional add-on App
/mnt	Mount Directory
/media	Removable Devices
/srv	Service Data




```
rstudio@650a7121b06a:~$ pwd
/home/rstudio
rstudio@650a7121b06a:~$
```



This location is also known as your Home Directory

Tilde is shorthand for Home ‘~’

Now let's create some directories and files

Make a directory

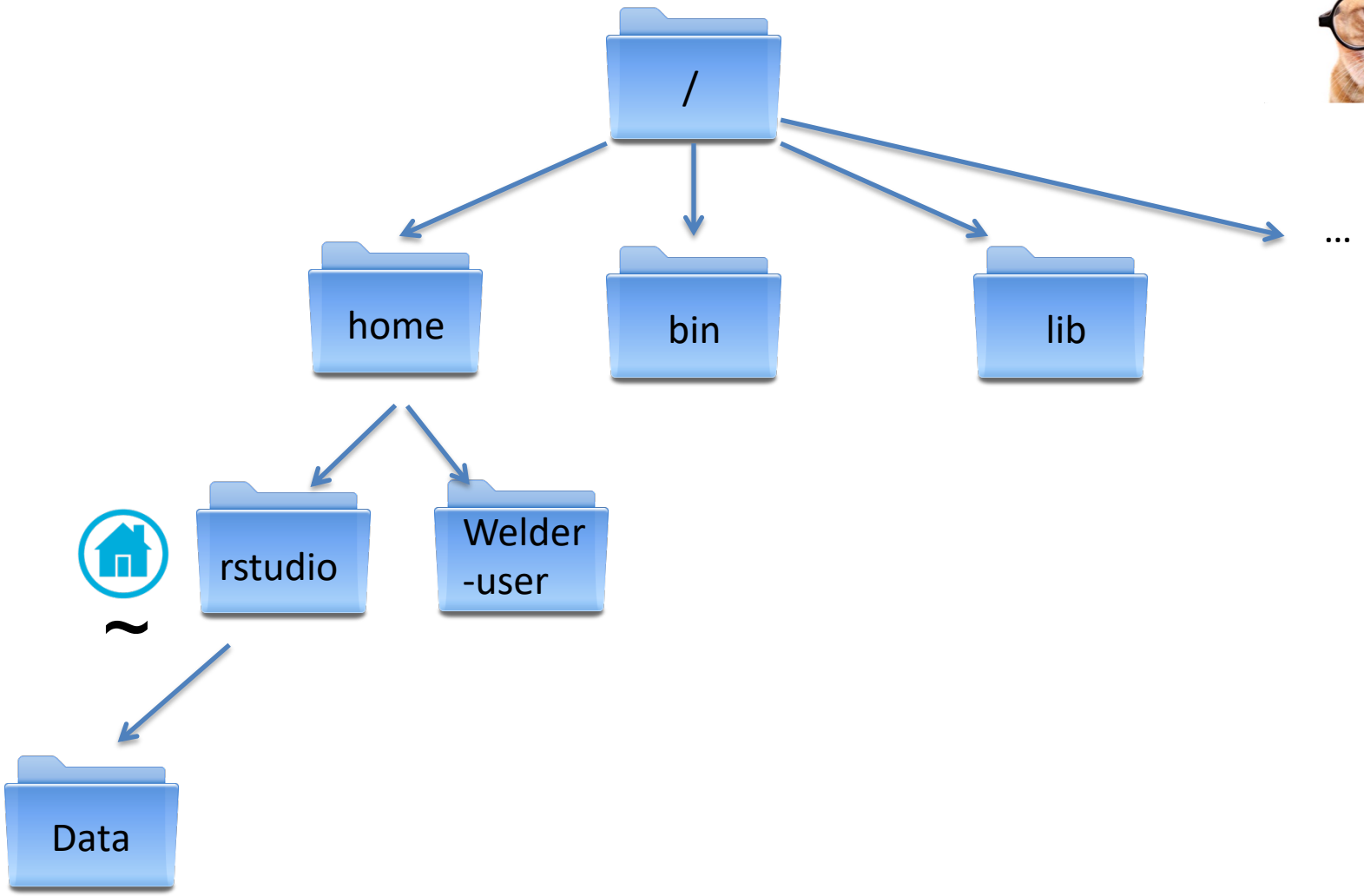
```
$ mkdir Data
```

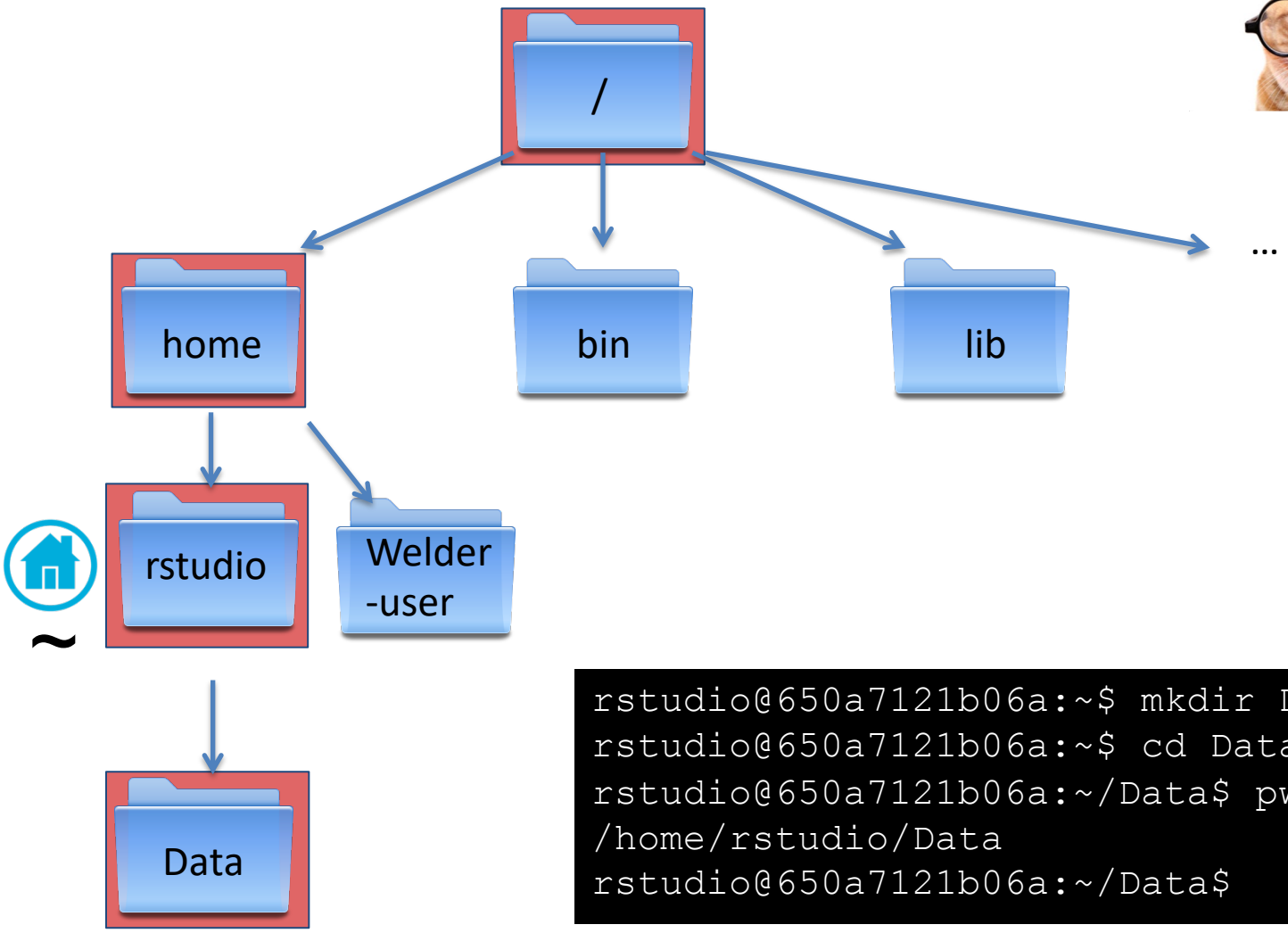
Change into this directory

```
$ cd Data
```

Now what is your present working directory?

NOTE! Directory names (and file names for the matter) can not contain spaces. Underscores are often used instead if you want to separate words.


/
home
bin
lib
...
rstudio
Welder
-user
~
Data




```
rstudio@650a7121b06a:~$ mkdir Data
rstudio@650a7121b06a:~$ cd Data/
rstudio@650a7121b06a:~/Data$ pwd
/home/rstudio/Data
rstudio@650a7121b06a:~/Data$
```

Now let's create some directories and files

Make an empty file

```
$ touch rags
```

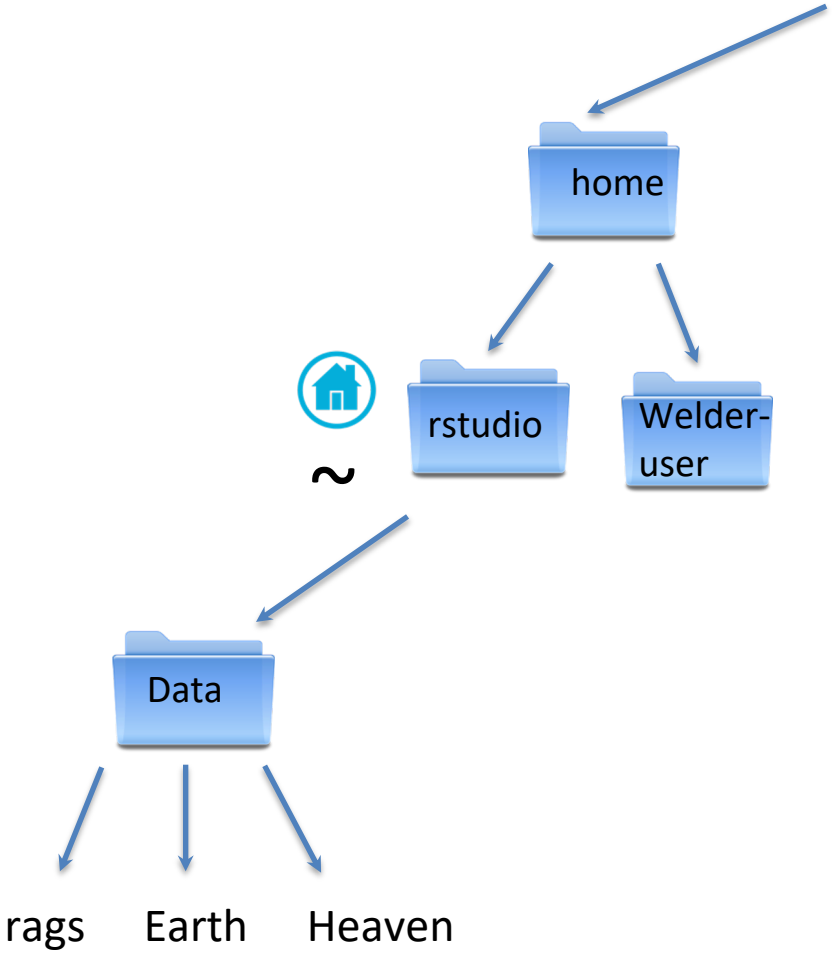
And another two

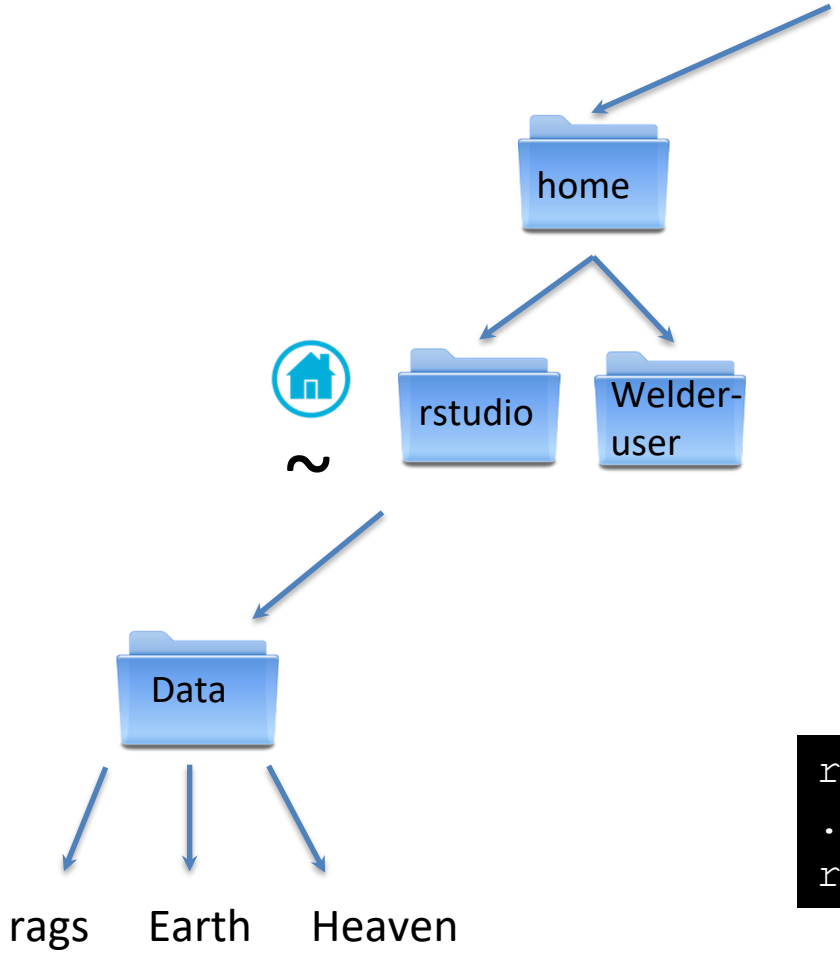
```
$ touch Earth Heaven
```

Now let's list the contents of the current directory (Data)

```
$ ls
```

```
rstudio@650a7121b06a:~/Data$ touch rags
rstudio@650a7121b06a:~/Data$ touch Earth Heaven
rstudio@650a7121b06a:~/Data$ ls
Earth  Heaven  rags
rstudio@650a7121b06a:~/Data$
```

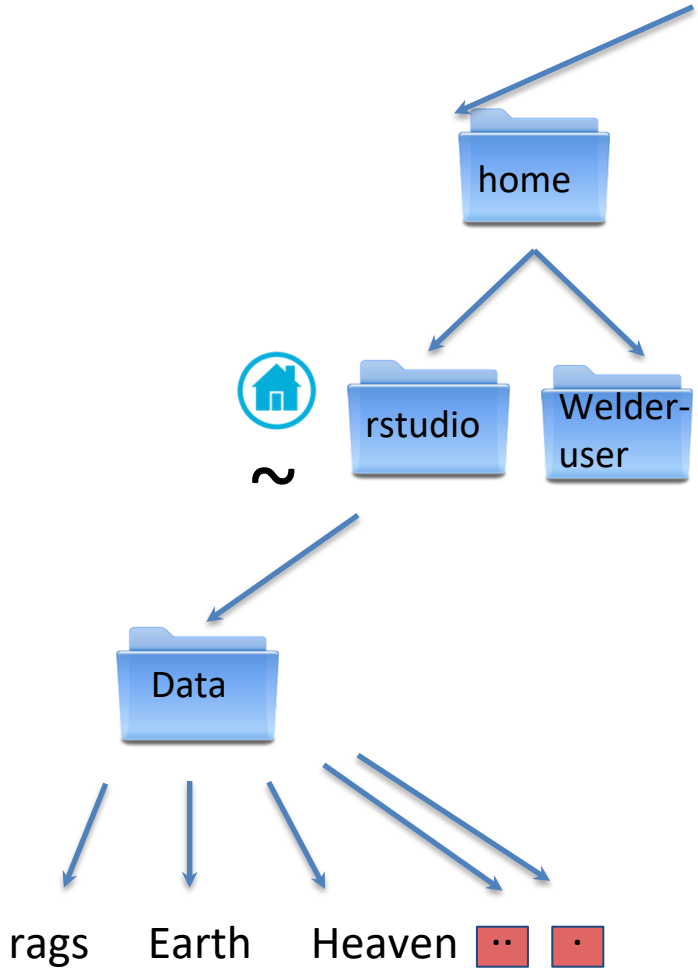

home
rstudio
Welder-user
~
Data
rags
Earth
Heaven



Now list ALL of the files

```
rstudio@650a7121b06a:~/Data$ ls -a
.   ..  Earth  Heaven  rags
rstudio@650a7121b06a:~/Data$
```

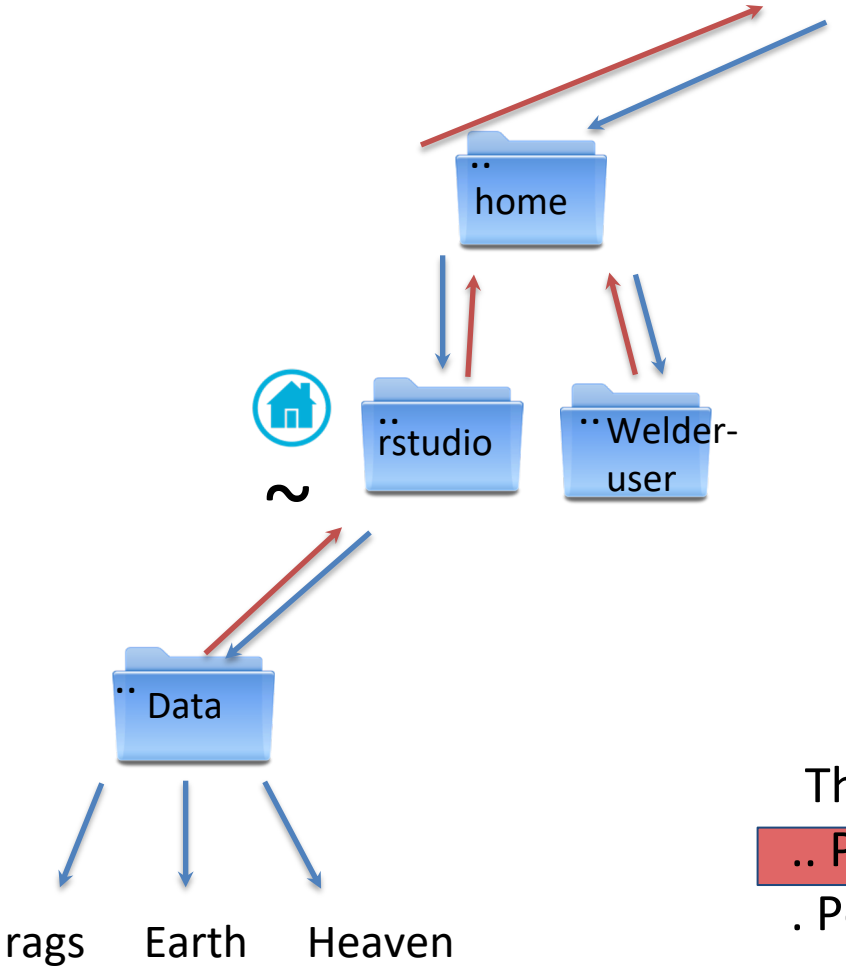
Two special files!



Now list ALL of the files

```
rstudio@650a7121b06a:~/Data$ ls -a
.   ..  Earth  Heaven  rags
rstudio@650a7121b06a:~/Data$
```

Two special files!

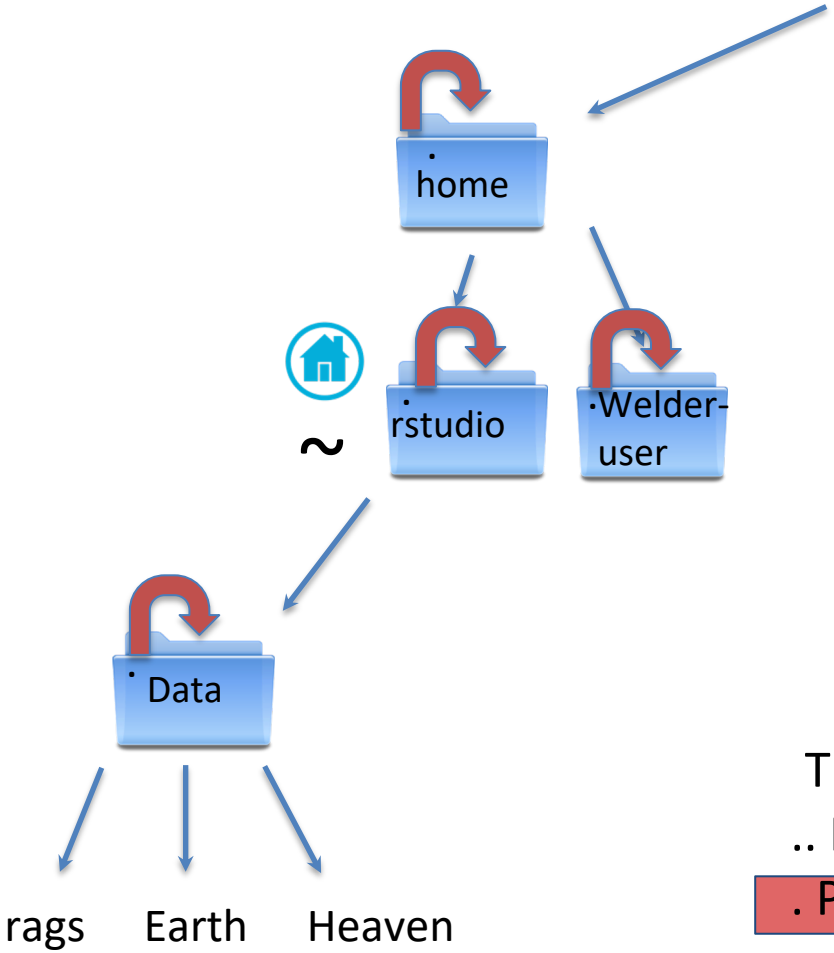


These special files are in every directory

.. Points to one directory above

. Points to the current directory

Two special files!



These special files are in every directory

.. Points to one directory above

. Points to the current directory

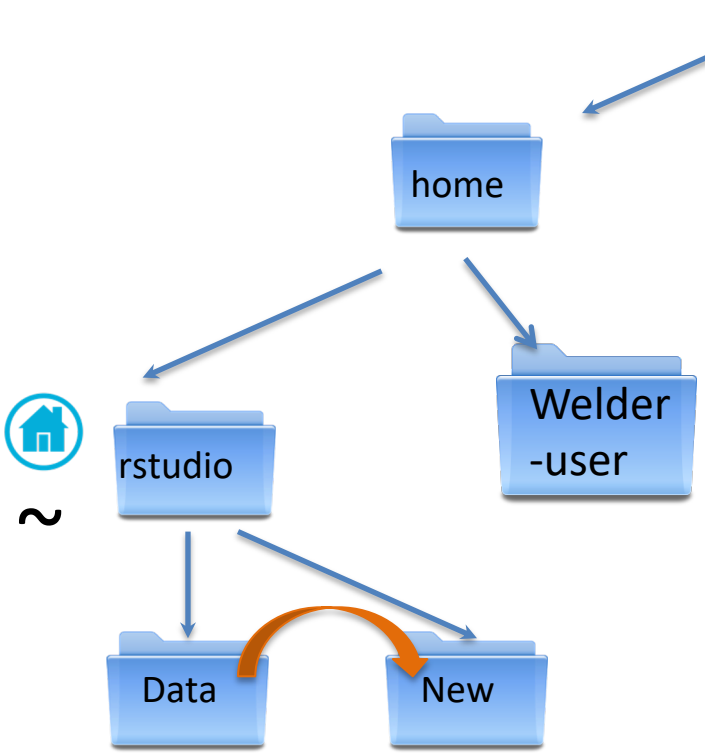
Two special files!

. and .. are used for specifying location

Whenever you do anything on Unix
(move around, move a file, rename a file etc…)
You have to tell the system where that thing is
using a path

. and .. are part of RELATIVE paths

Two special files!

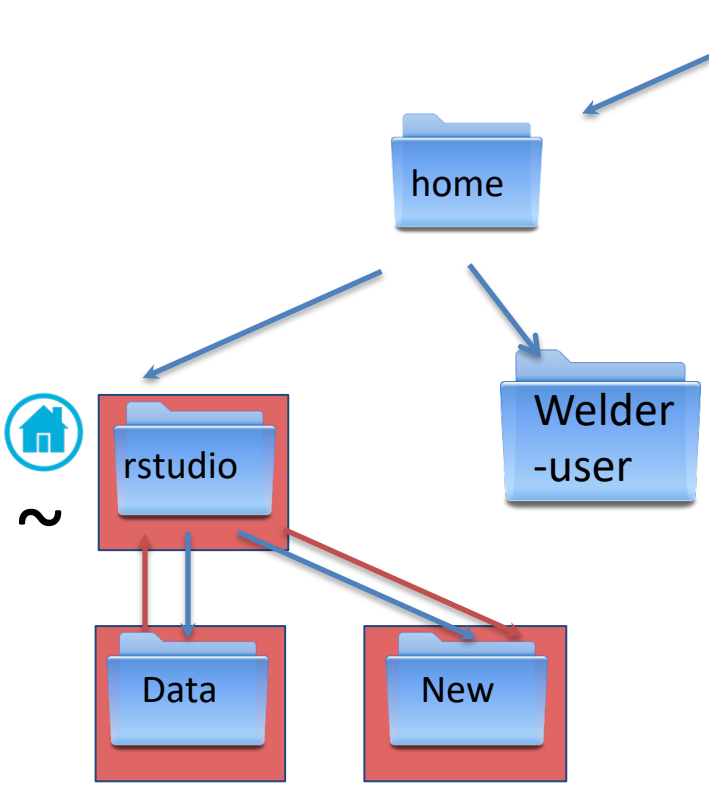


Moving from Data to New

RELATIVE PATH

```
$ cd ../New
```

Two special files!

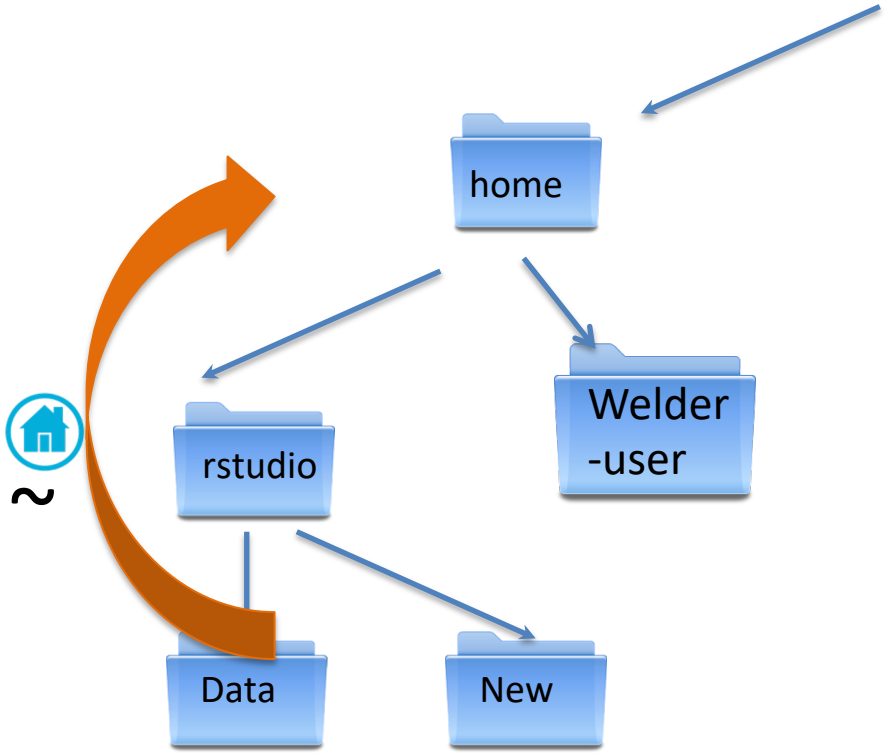


Moving from Data to New

RELATIVE PATH

```
$ cd ../New
```

Shortcuts to the home dir!

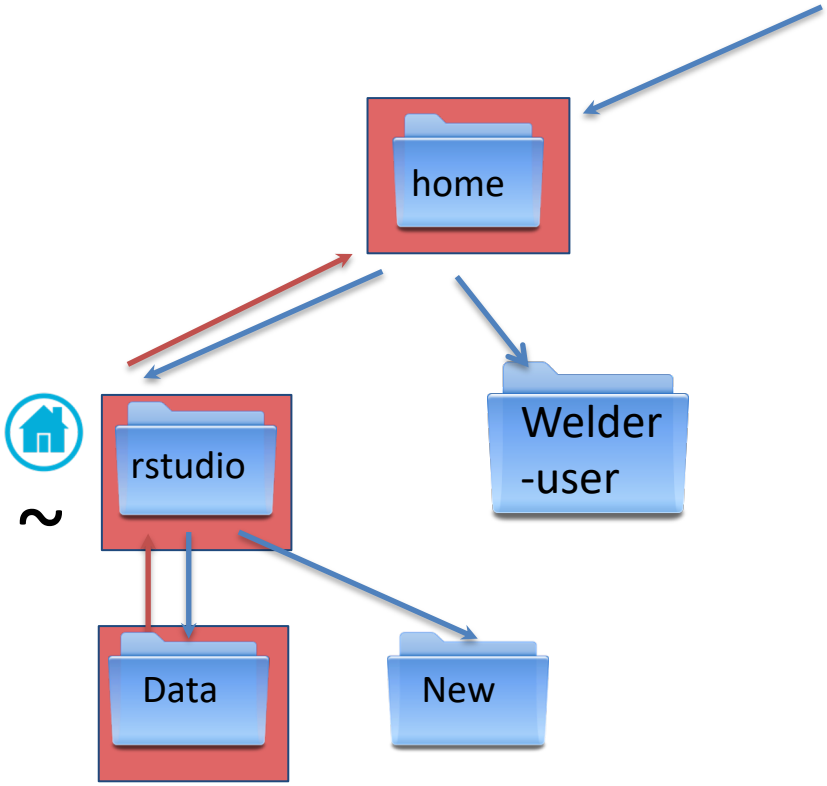


Moving from Data to home

RELATIVE PATH

```
$ cd ../../
```

Shortcuts to the home dir!



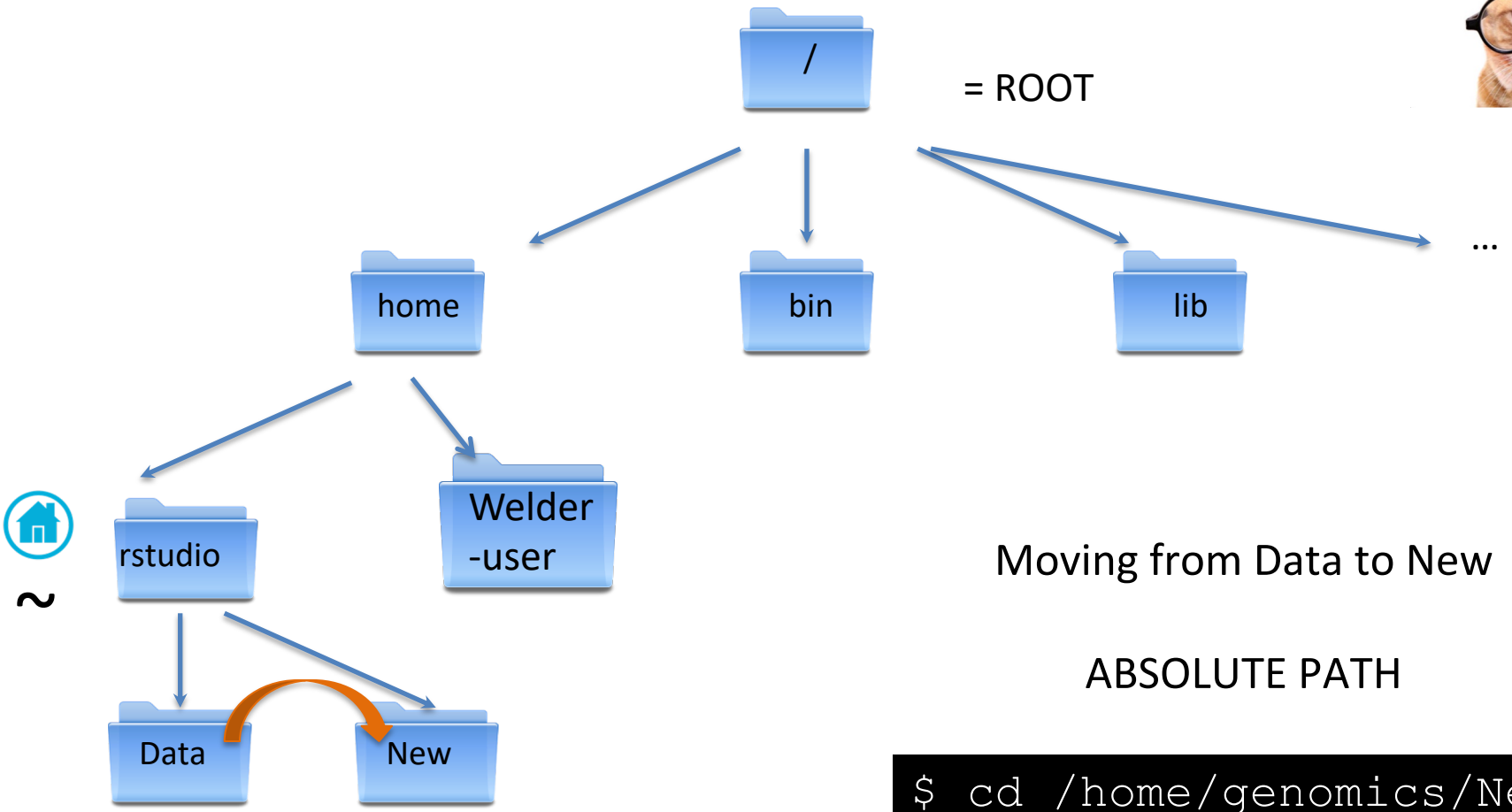
Moving from Data to home

RELATIVE PATH

```
$ cd ../../
```

Relative paths will always change depending on your location.

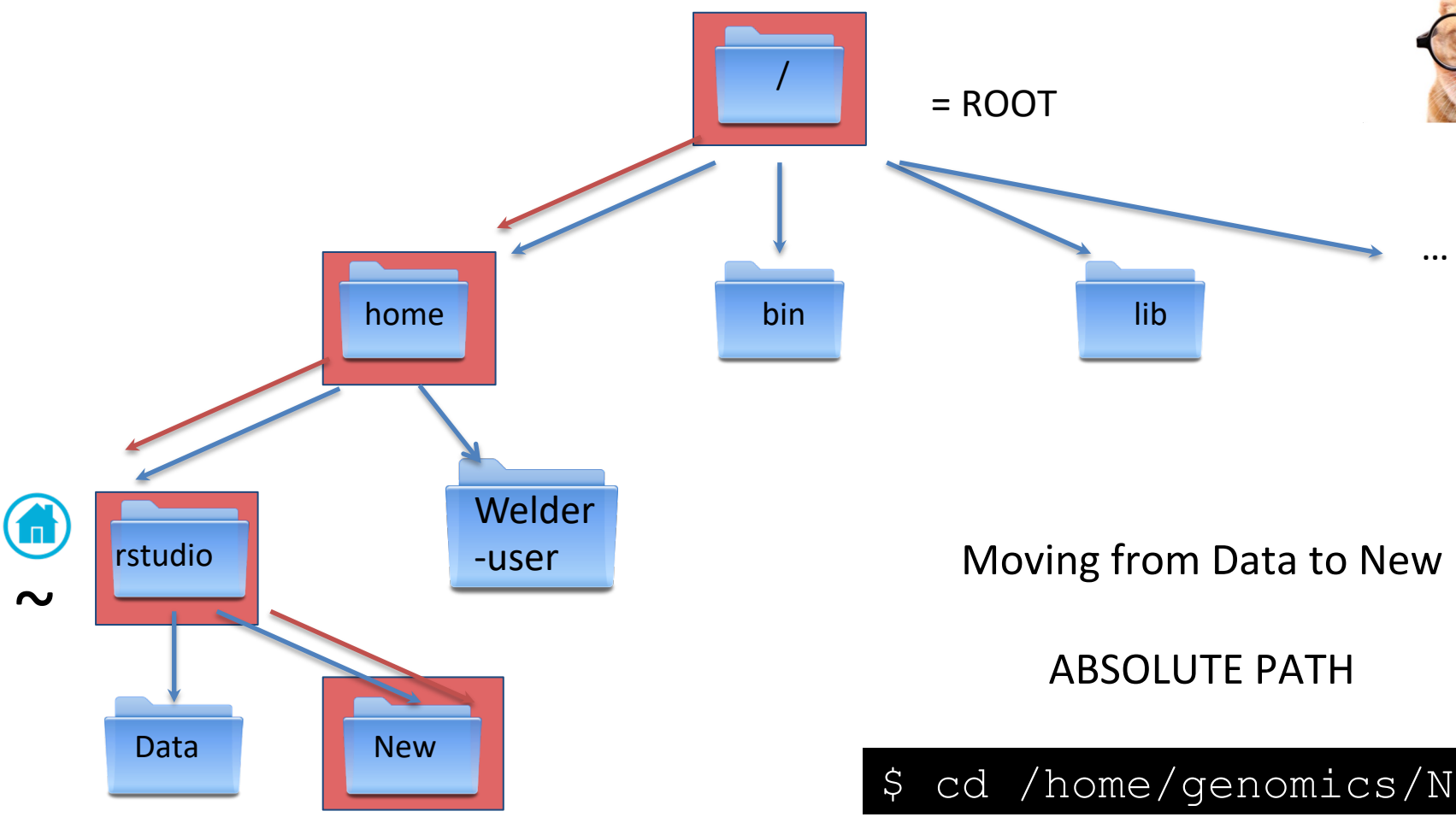
The alternative is ABSOLUTE paths. These always start from root.



Moving from Data to New

ABSOLUTE PATH

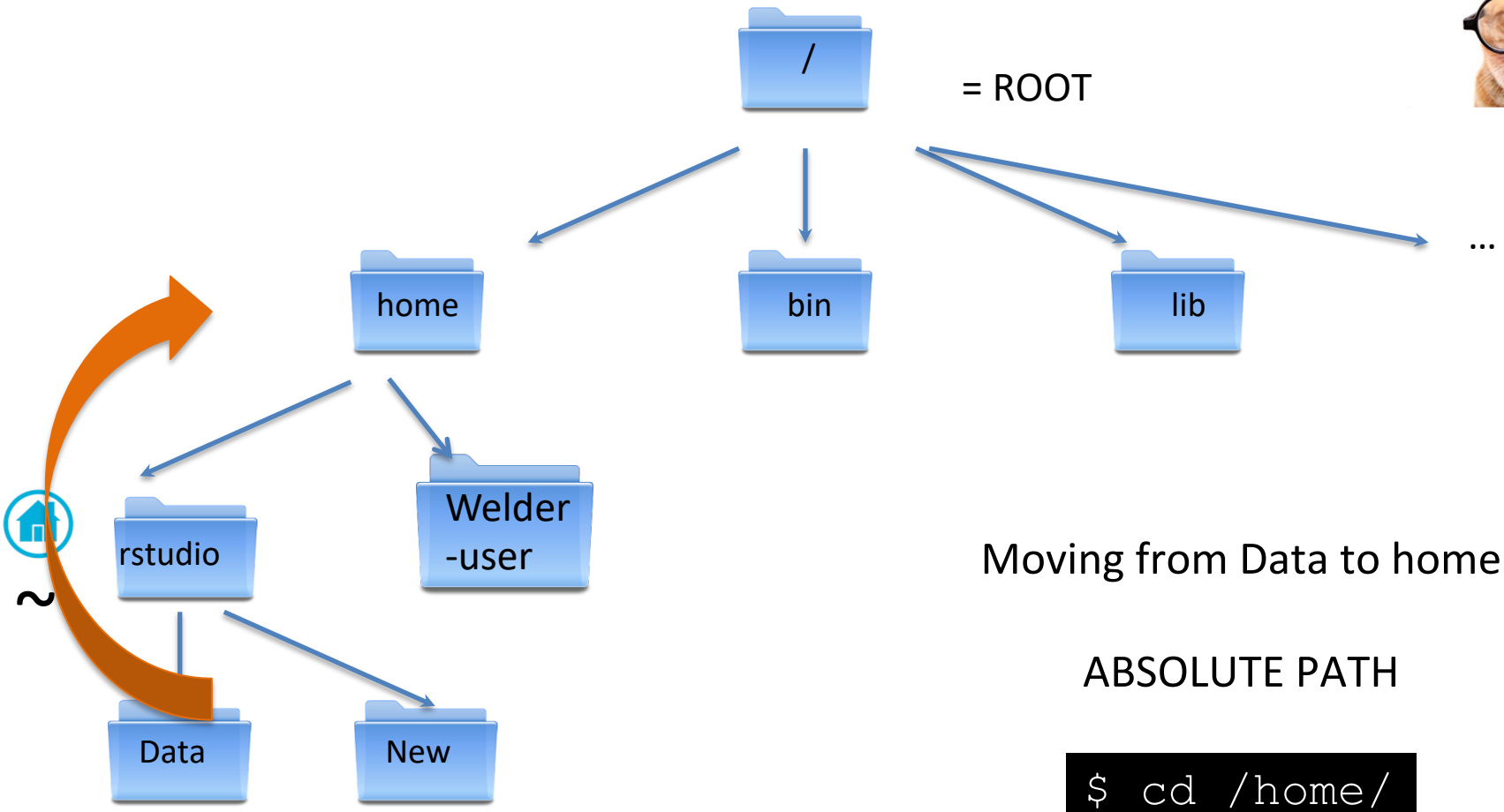
```
$ cd /home/genomics/New
```



Moving from Data to New

ABSOLUTE PATH

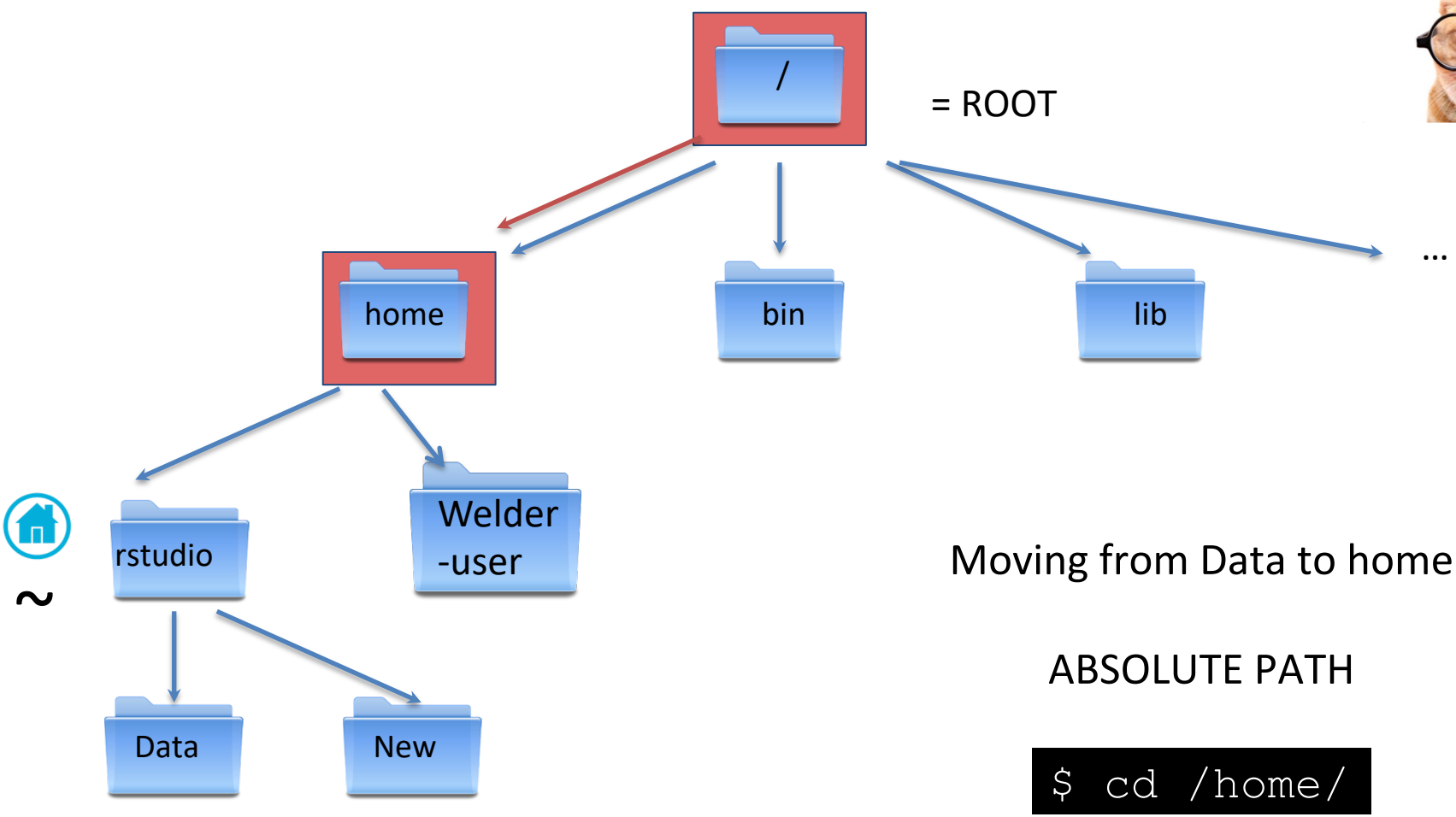
```
$ cd /home/genomics/New
```



Moving from Data to home

ABSOLUTE PATH

```
$ cd /home/
```



Moving from Data to home

ABSOLUTE PATH

```
$ cd /home/
```

Let's put this to practice

Where am I right now? (Should be the Data directory)

```
$ pwd
```

Change to the directory above

```
$ cd ../
```

Let's list the contents of the Data directory

```
$ ls ./Data/
```

CHALLENGE 1!

1. Move into the Data directory and list the contents of your home directory
2. In Data, make a new directory and move into this location
3. From this new directory, move into your home directory IN ONE COMMAND and check your location

Challenge 1!

1. Move into the Data directory and list the contents of your home directory

```
$ cd Data
$ ls ..
```

OR `$ ls /home/rstudio/` OR `$ ls ~`

2. In Data, make a new directory and move into this location

```
$ mkdir new
$ cd new
```

3. From this new directory, move into your home directory IN ONE COMMAND and check your location

```
$ cd ../..
$ pwd
```

OR `$ cd /home/rstudio/` OR `$ cd ~` OR `$ cd`

Tricks for less typing!

Tab complete is a nice trick to save you typing paths

For this example we are going to list everything in the directory /home/genomics/workshop_materials/

Start by typing:

```
$ ls /
```

Followed by tab twice quickly

```
genomics@genomics:~$ ls /
bin/            initrd.img      media/          run/            tmp/
boot/           initrd.img.old  mnt/            sbin/           usr/
dev/            lib/            opt/            snap/           var/
etc/            lib64/          proc/           srv/            vmlinuz
home/           lost+found/     root/           sys/            vmlinuz.old
genomics@genomics:~$ ls /
```

This shows the contents of the root directory

Tricks for less typing!

Now type:

```
$ ls /h
```

Followed by tab once. The path to the /home/ directory has filled in.

```
$ ls /home/
```

Now type:

```
$ ls /home/r
```

Followed by tab once. The path to the /home/genomics/ directory has filled in.

```
$ ls /home/rstudio
```

Finally type:

```
$ ls /home/rstudio/w
```

Followed by tab once to finish the path, and then enter.
You've now listed that directory contents.

Tab complete will fill in paths, save you time in typing and prevent typos!

Tricks for less typing!

* Represents a special character
For example:

```
$ ls /home/rstudio/*.txt
```

Will list everything in my home directory ending .txt

The up arrow can be used to re-run commands

Press your up arrow and see

If you want all of these commands listed, simply type

```
$ history
```

Binary programs

These are all programs installed on the Unix machine.

They can be found in /bin

```
$ ls /bin
```

```
genomics@genomics:~$ ls /bin/
bash                bzmore          fsck.btrfs      ls              ntfscluster     setfacl                         true
btrfs               cat             fuser           lsblk           ntfscmp         setfont                         udevadm
btrfs-calc-size     chacl           fusermount      lsmod           ntfsfallocate   setupcon                        ulockmgr_server
btrfsck             chgrp           getfacl         mkdir           ntfsfix         sh                              umount
btrfs-convert       chmod           grep            mkfs.btrfs      ntfsinfo        sh.distrib                      uname
btrfs-debug-tree    chown           gunzip          mknod           ntfsls          sleep                           uncompress
btrfs-find-root     chvt            gzexe           mktemp          ntfsmove        ss                              unicode_start
btrfs-image         cp              gzip            more            ntfstruncate    static-sh                       vdir
btrfs-map-logical   cpio            hciconfig       mount           ntfswipe        stty                            vmmouse_detect
btrfs-select-super  dash            hostname        mountpoint      open            su                              wdctl
btrfs-show-super    date            ip              mt              openvt          sync                            which
btrfstune           dd              journalctl      mt-gnu          pidof           systemctl                       whiptail
btrfs-zero-log      df              kbd_mode        mv              ping            systemd                         ypdomainname
bunzip2             dir             kill            nano            ping6           systemd-ask-password            zcat
busybox             dmesg           kmod            nc              plymouth        systemd-escape                  zcmp
bzcat               dnsdomainname   less            nc.openbsd      ps              systemd-hwdb                    zdiff
bzcmp               domainname      lessecho        netcat          pwd             systemd-inhibit                 zegrep
bzdiff              dumpkeys        lessfile        netstat         rbash           systemd-machine-id-setup        zfgrep
bzegrep             echo            lesskey         networkctl      readlink        systemd-notify                  zforce
bzexe               ed              lesspipe        nisdomainname   red             systemd-tmpfiles                zgrep
bzfgrep             egrep           ln              ntfs-3g         rm              systemd-tty-ask-password-agent  zless
bzgrep              false           loadkeys        ntfs-3g.probe   rmdir           tailf                           zmore
bzip2               fgconsole       login           ntfs-3g.secaudit rnano          tar                             znew
bzip2recover        fgrep           loginctl        ntfs-3g.usermap run-parts       tempfile
bzless              findmnt         lowntfs-3g      ntfscat         sed             touch
genomics@genomics:~$
```

These include pwd, mkdir, ls …

Every binary program has a manual

To view the manual page, type man followed by the name of the program

```
$ man <PROGRAMME>
```

Open the manual page for ls

```
$ man ls
```

Scroll through (enter) and find the options for:
long listing format (-l), human-readable sizes (-h) and sort by modification time (-t)

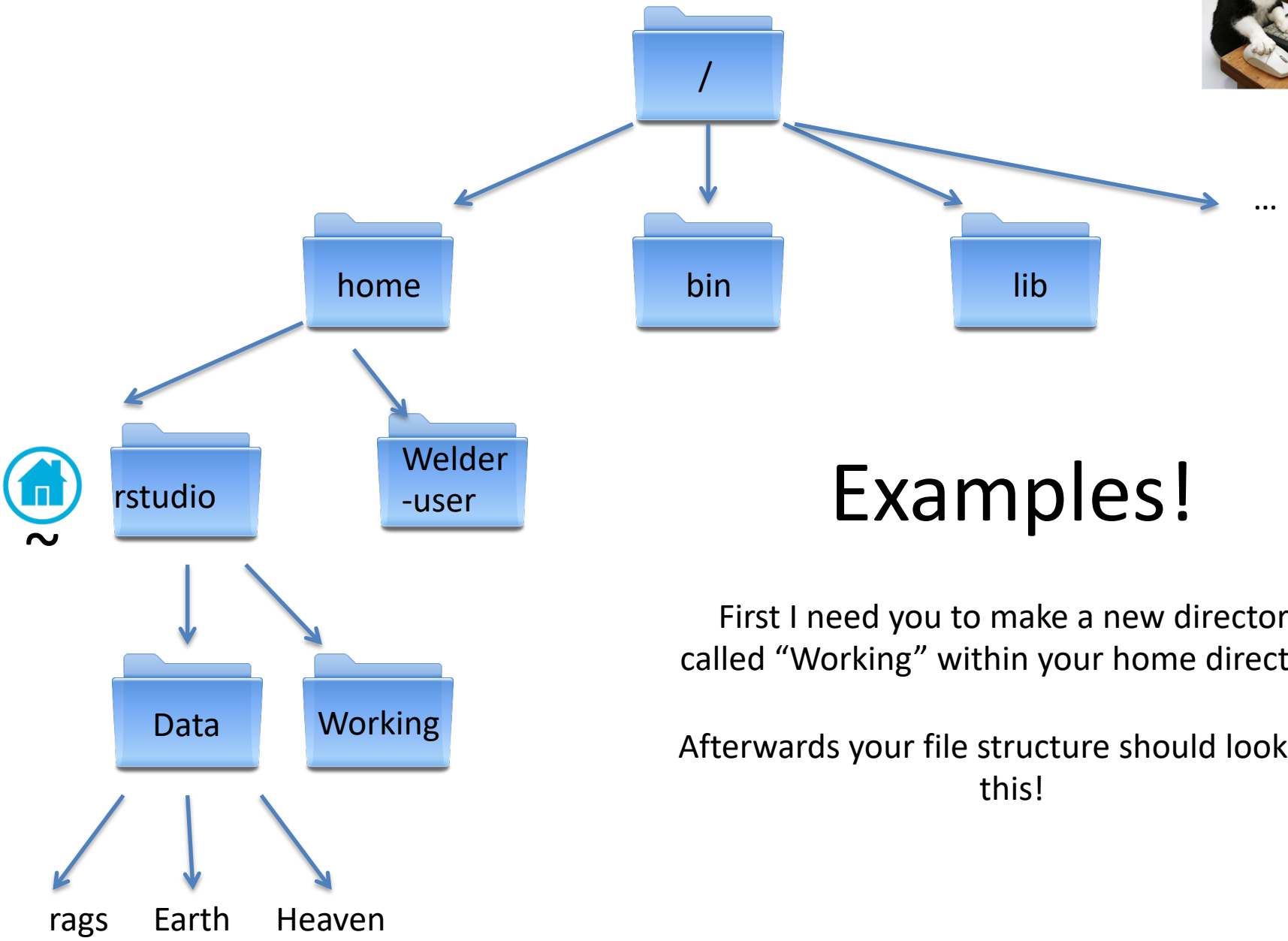
Exit the manual page (type q) and give these ls options a go in your Data directory

```
$ ls -l ./Data
```
OR
```
$ ls -t ./Data
```

```
$ ls -l -h ./Data
```
OR
```
$ ls -lh ./Data
```

```
$ ls -l -h -t./Data
```
OR
```
$ ls -lht ./Data
```

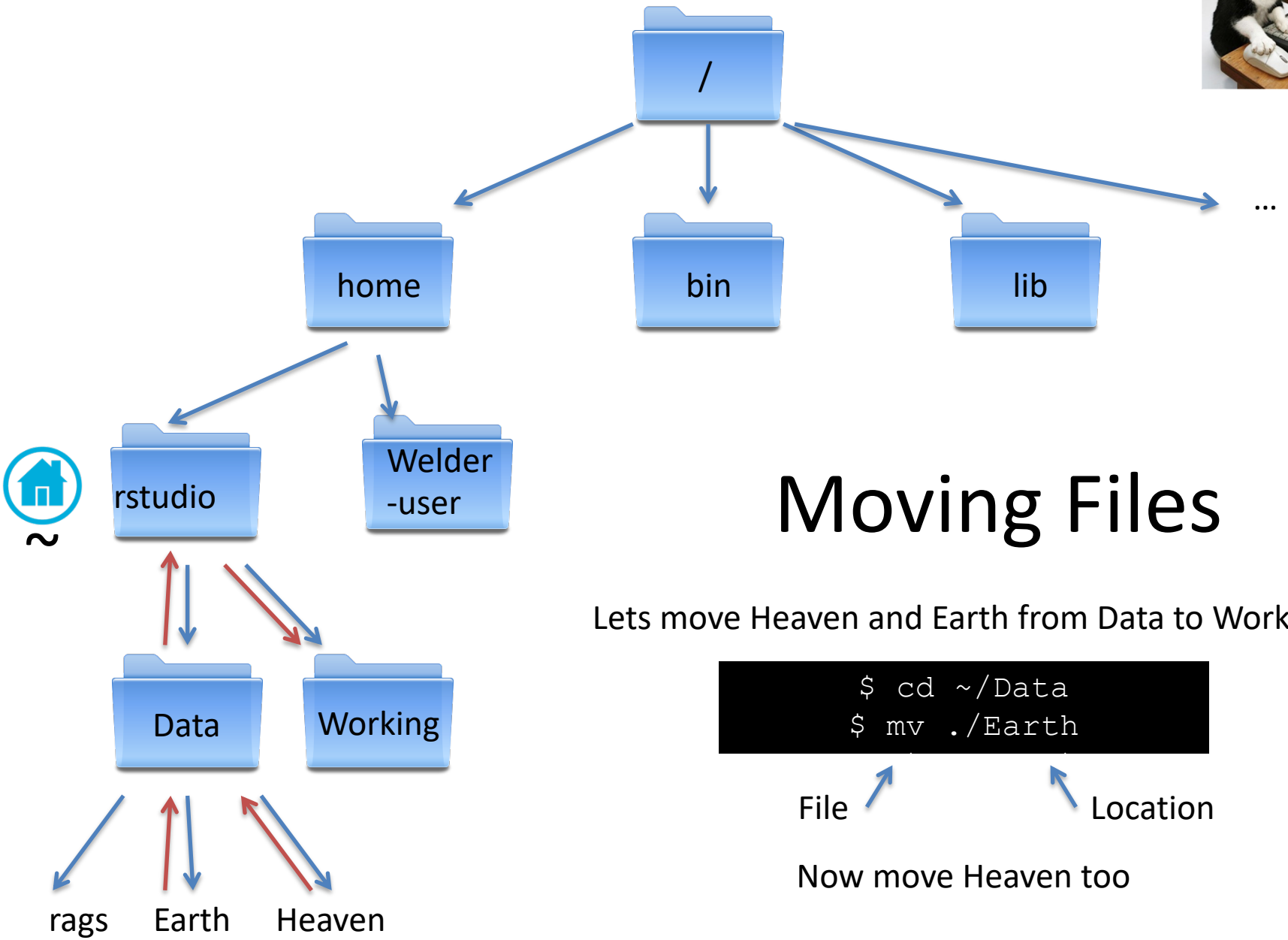
```
$ ls -l -h -t -a ./Data
```
OR
```
$ ls -lhta ./Data
```



Examples!

First I need you to make a new directory called “Working” within your home directory.

Afterwards your file structure should look like this!



Moving Files

Lets move Heaven and Earth from Data to Working

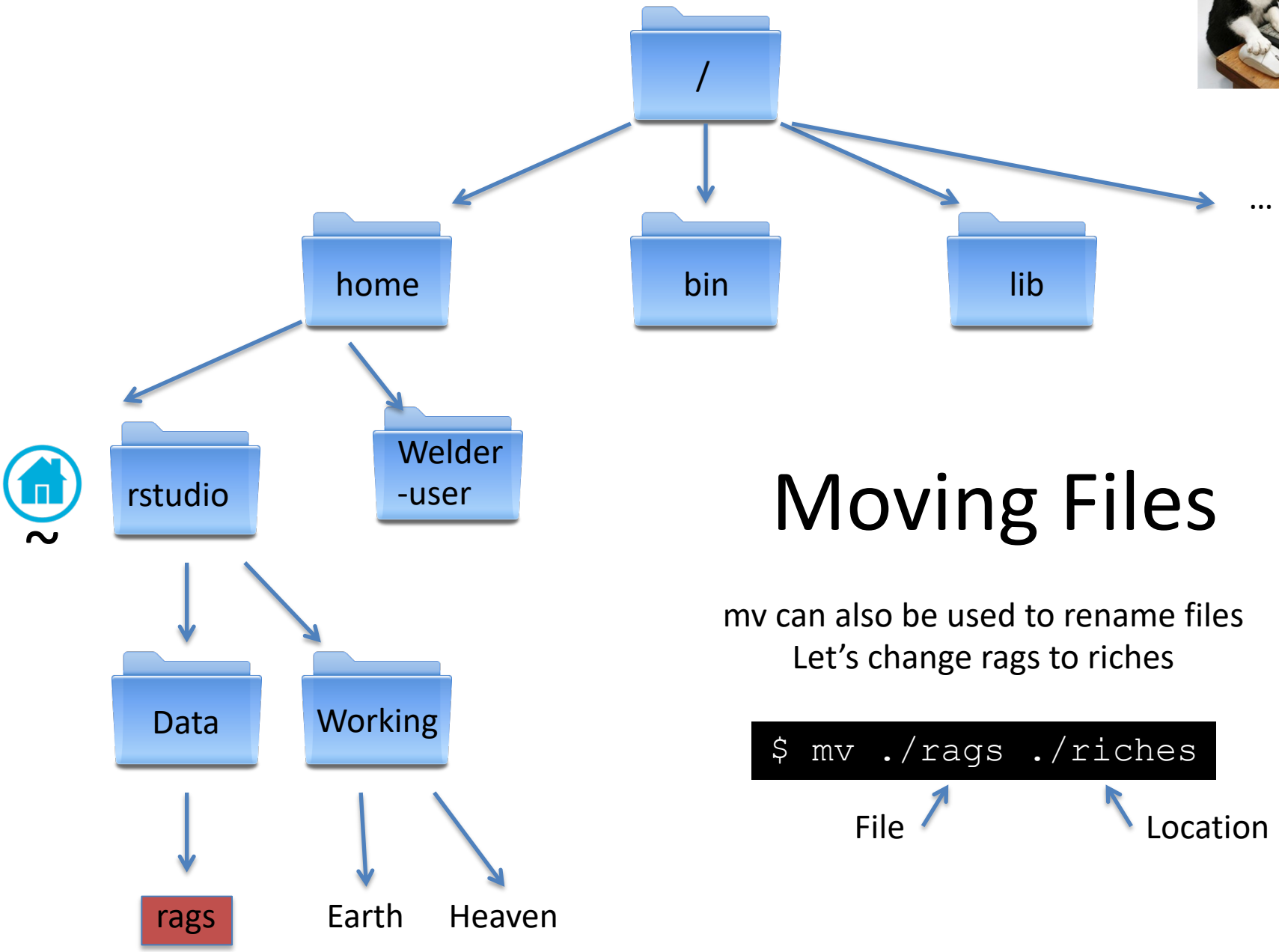
```
$ cd ~/Data
$ mv ./Earth
```

File

Location

Now move Heaven too

REMEMBER TAB COMPLETE!



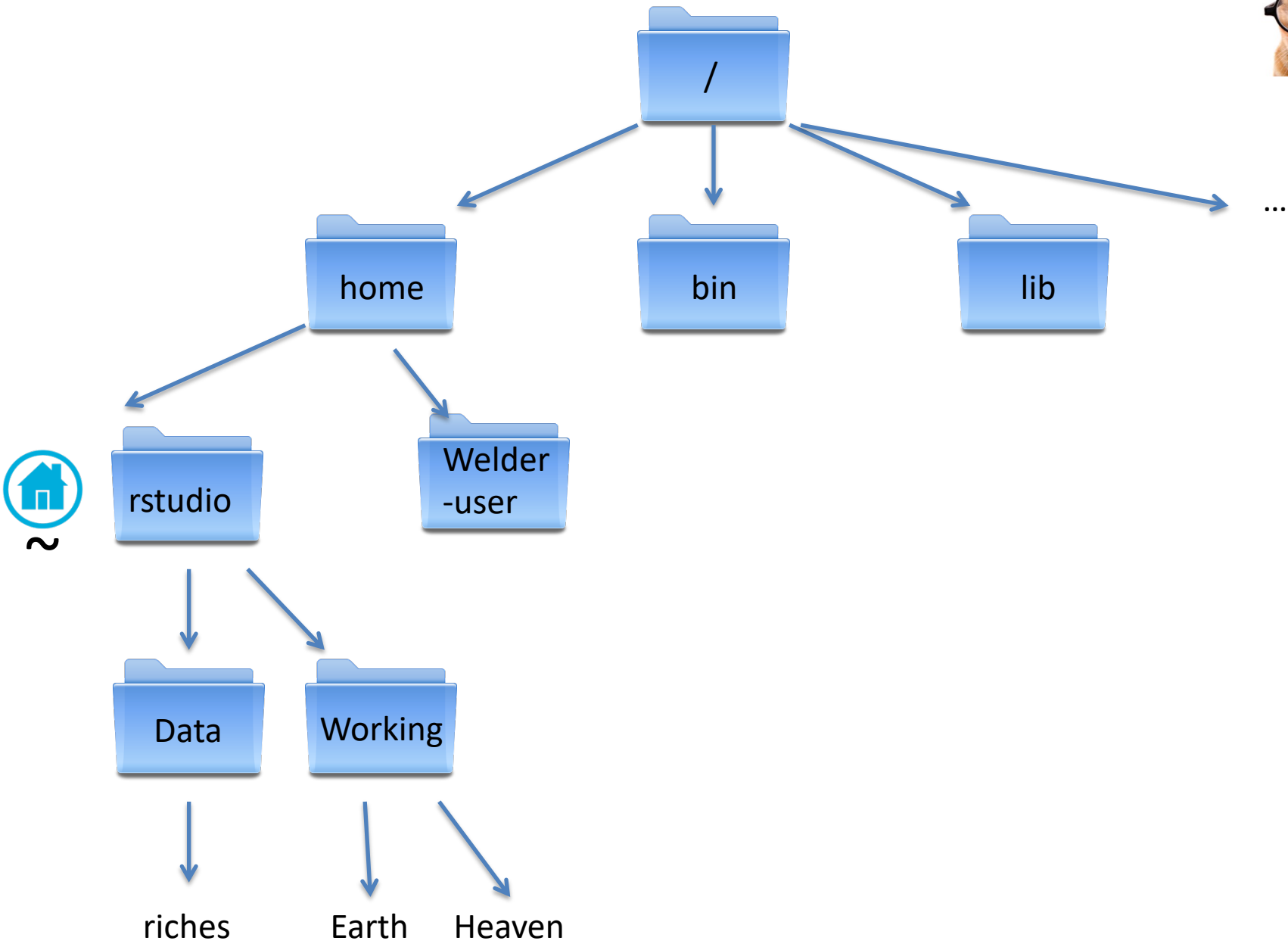
Moving Files

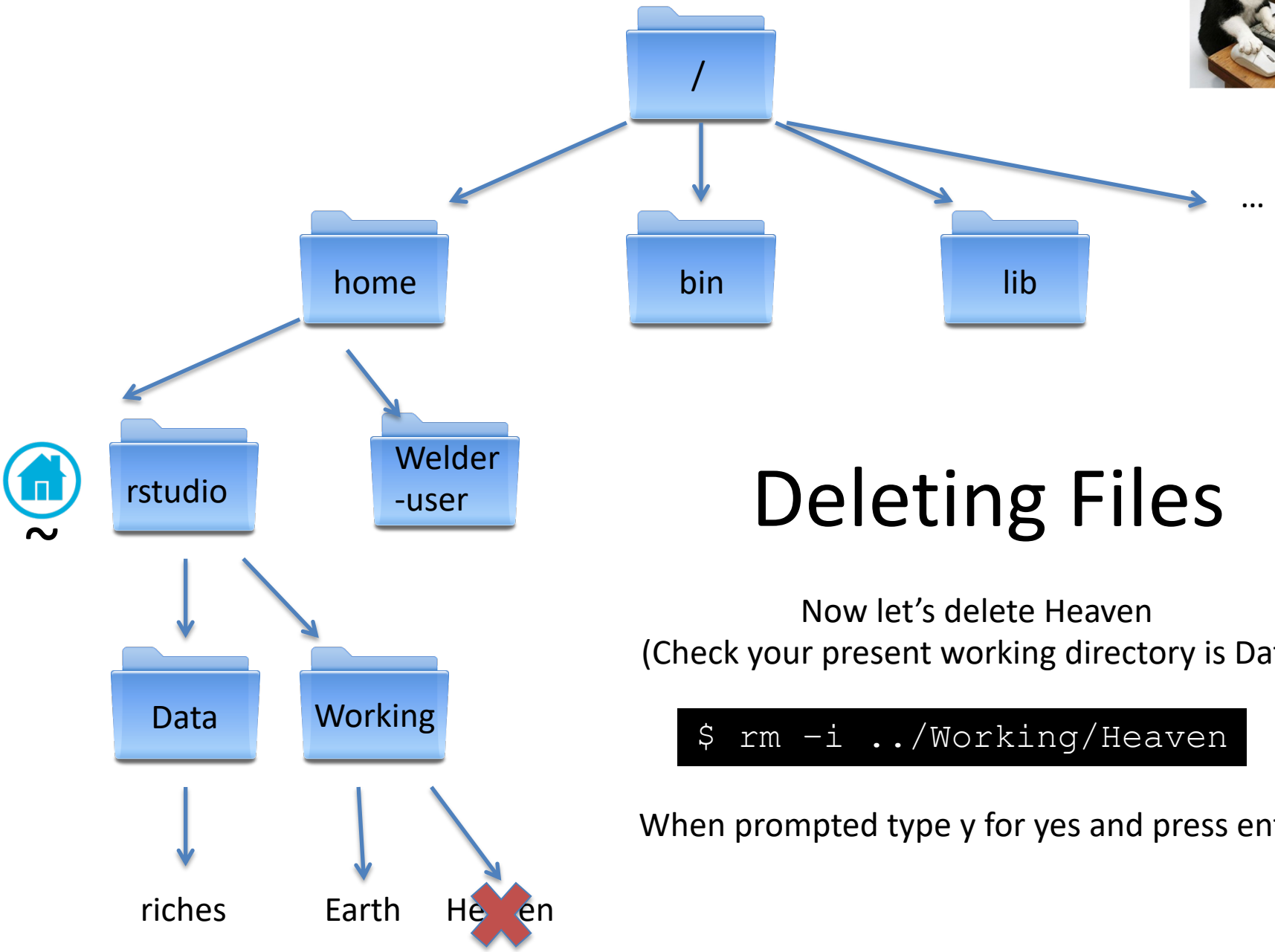
mv can also be used to rename files
Let's change rags to riches

```
$ mv ./rags ./riches
```

File

Location


/
home
bin
lib
...
rstudio
Welder
-user
~
Data
Working
riches
Earth
Heaven

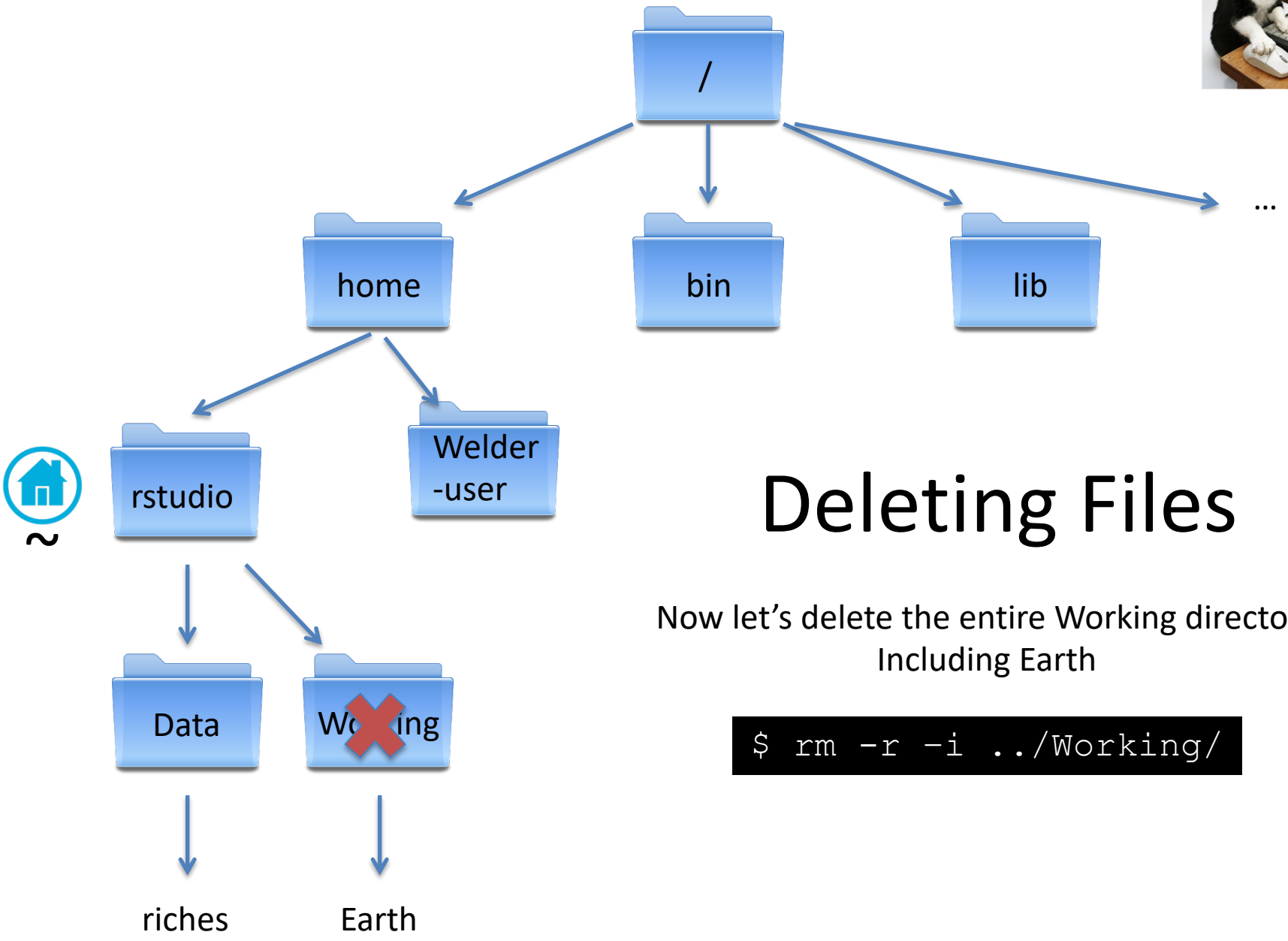


Deleting Files

Now let's delete Heaven
(Check your present working directory is Data)

```
$ rm -i ../Working/Heaven
```

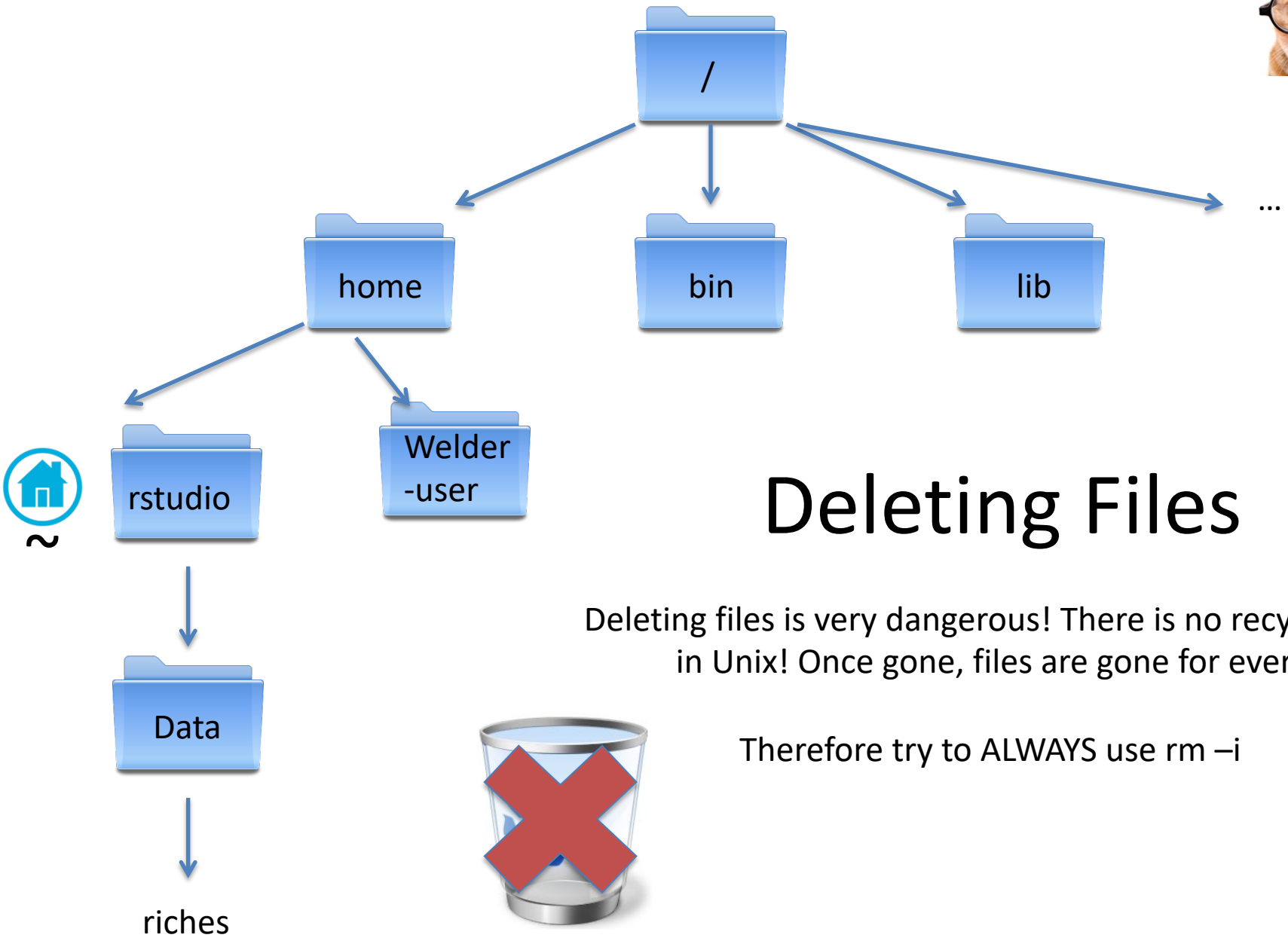
When prompted type y for yes and press enter



Deleting Files

Now let's delete the entire Working directory
Including Earth

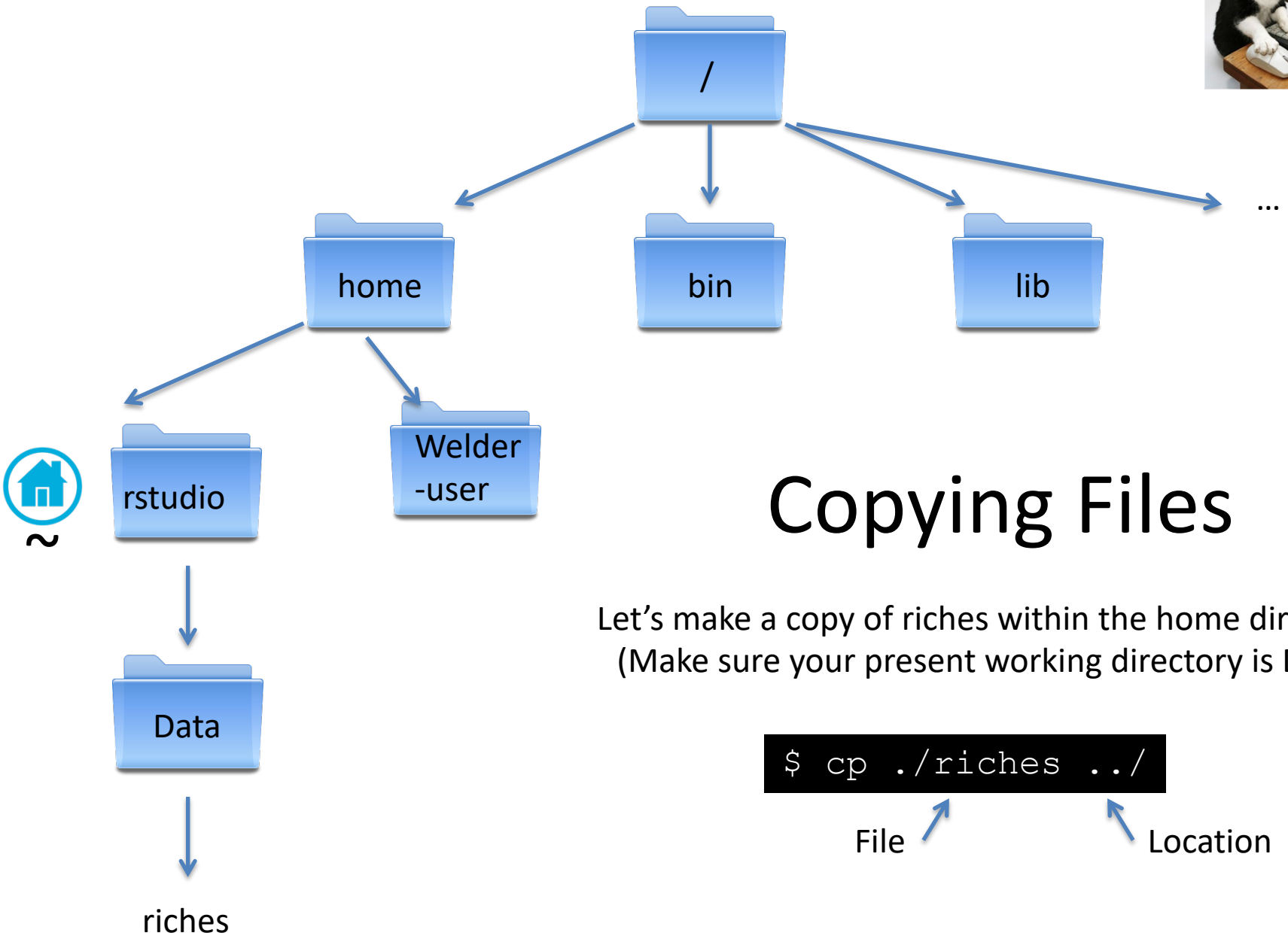
```
$ rm -r -i ../Working/
```



Deleting Files

Deleting files is very dangerous! There is no recycle bin in Unix! Once gone, files are gone for ever!

Therefore try to ALWAYS use rm –i



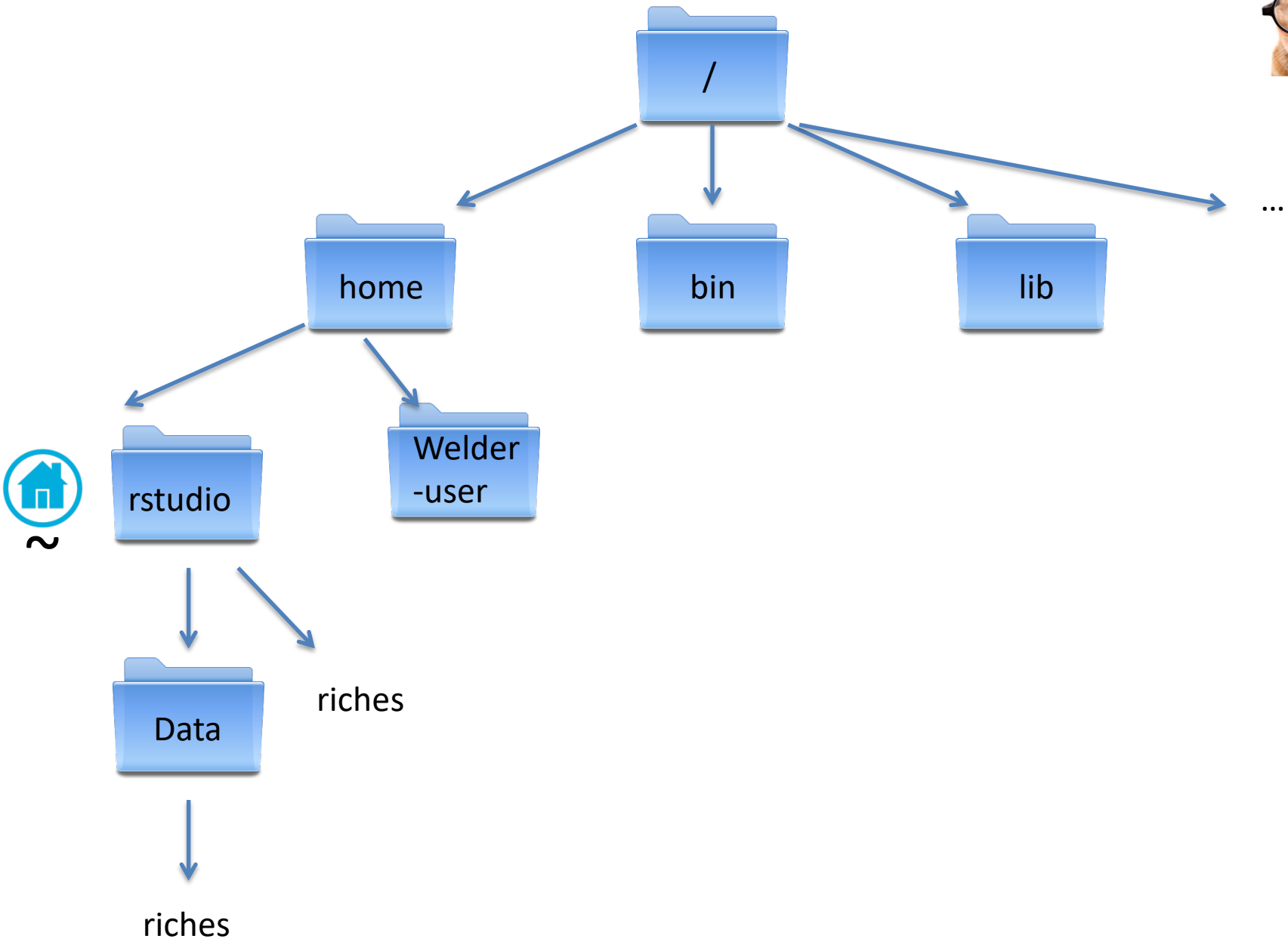
Copying Files

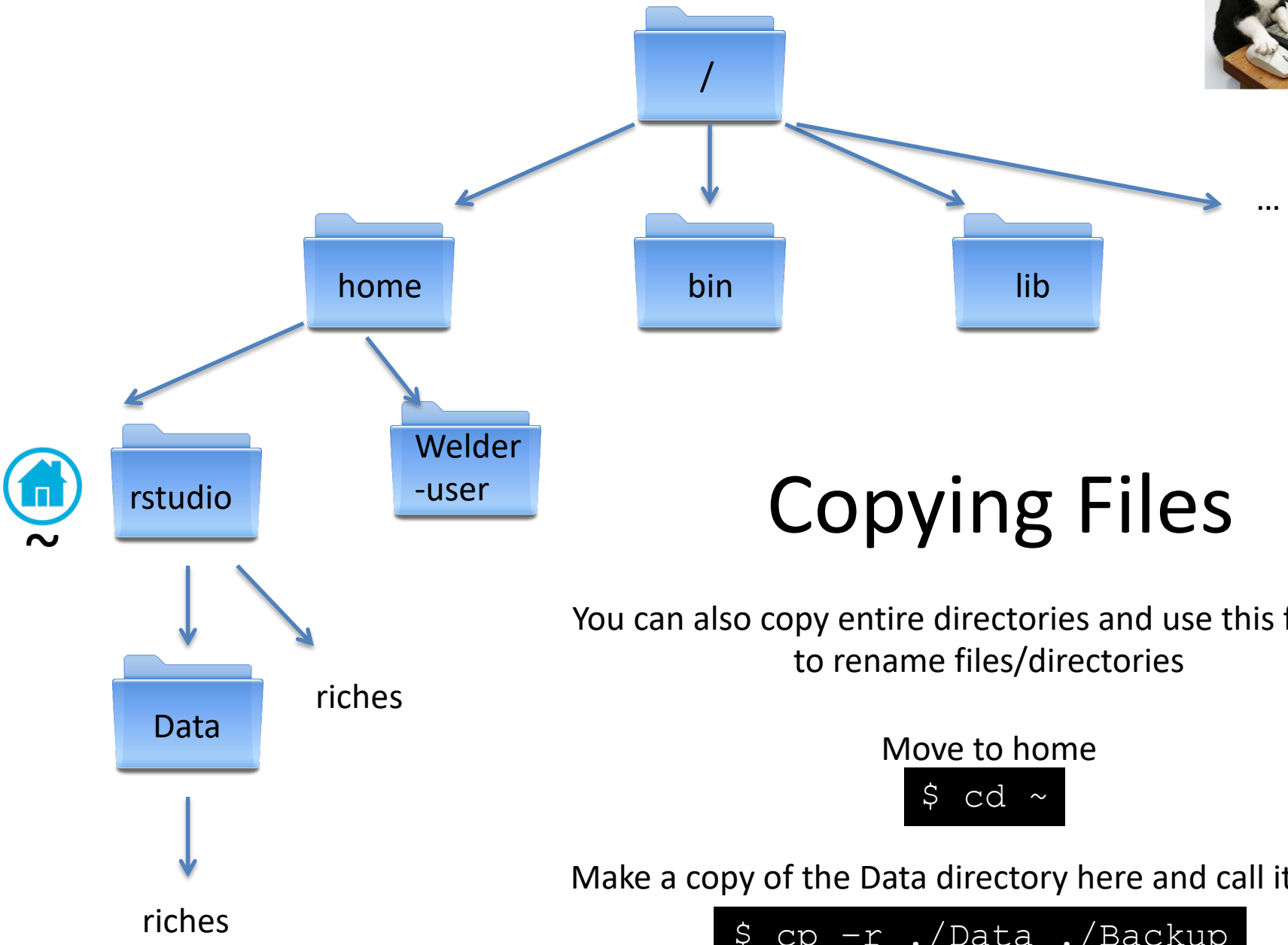
Let's make a copy of riches within the home directory
(Make sure your present working directory is Data)

```
$ cp ./riches ../
```

File

Location


/
home
bin
lib
...
Welder
-user
rstudio
~
Data
riches
riches



Copying Files

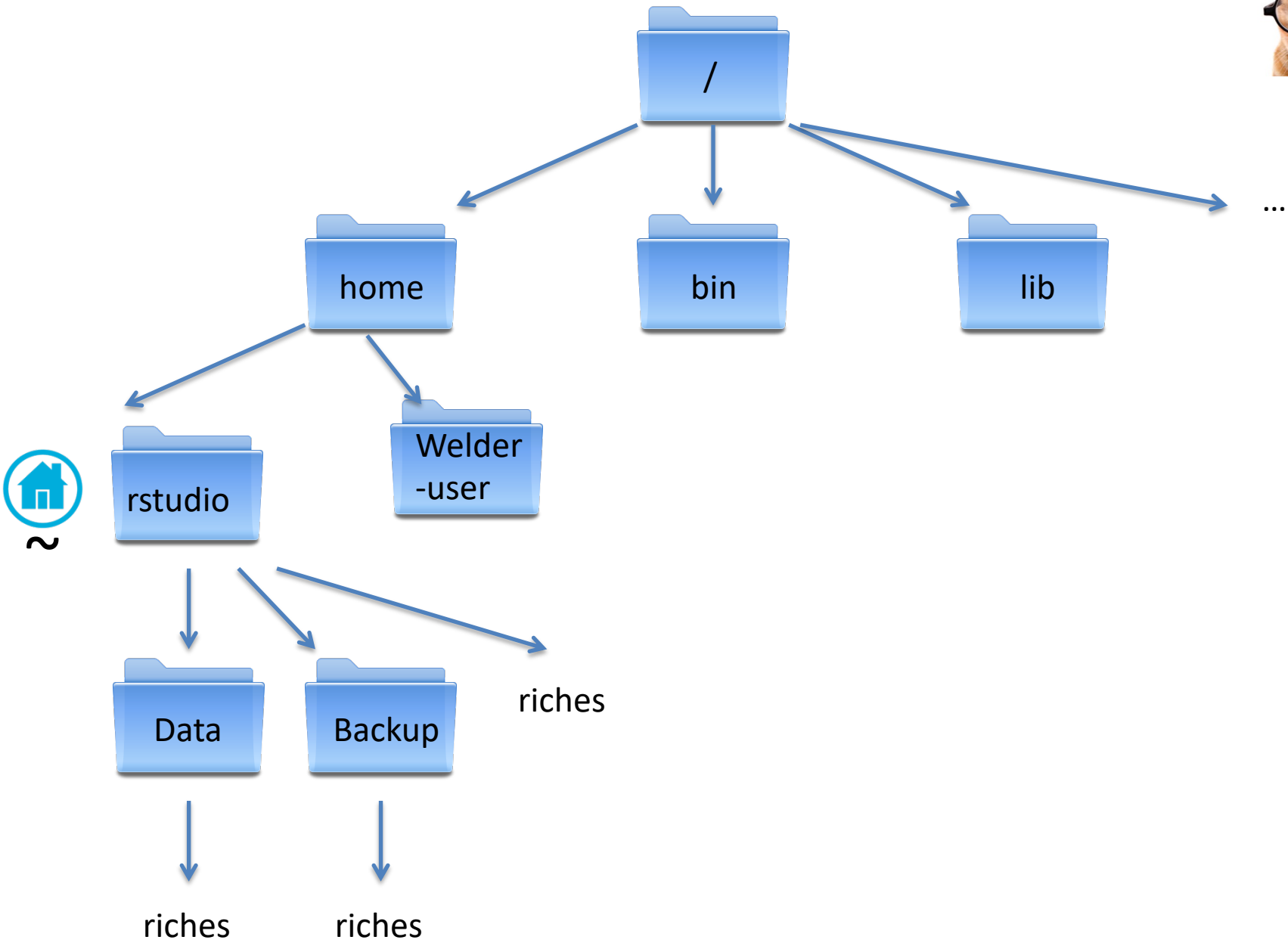
You can also copy entire directories and use this function to rename files/directories

Move to home

```
$ cd ~
```

Make a copy of the Data directory here and call it Backup

```
$ cp -r ./Data ./Backup
```


/
home
bin
lib
...
Welder
-user
rstudio
~
Data
Backup
riches
riches
riches

Typical File Sizes

One Sequencing Sample on the Illumina NextSeq
3,000,000 reads = 1 Gb

But typically you will sequence more than one sample
You may have different patients, different locations, replicates etc…

The size of the sequencing data file can easily become 100s of Gb

(or even bigger depending on the sequencer used)

Archived/Compressed Files

Commonly, people will compress large files so that they are easier to store or share
Here's an example:
sequences.tar.gz

.tar – means that it is a tape archive
.gz – means that it is gzipped

These can be used alone or in combination

To uncompress
A Tar Archive

```
$ tar -xvf <filename>
```

(x = extract, v = verbose, f = all files)

A Gzipped file

```
$ gunzip <filename>
```

A Gzipped Tar archive

```
$ tar -xzvf <filename>
```

Challenge 2!

1. Change to the workshop_materials directory at the following path:

```
~/workshop_materials/Unix
```

You should find a compressed directory:

```
Sequences.tar
```

2. Make a copy of this file in the Backup directory you created earlier

3. Un archive the original directory

4. Unzip the read files

5. Rename the unarchived files – sequence_1.fq and sequence_2.fq

6. Delete the original .tar file

tar	gunzip
cp	mv
rm –i	mkdir
cd	

Challenge 2!

1. Get the File!

```
cd ~/workshop_materials/Unix
```

2. Make a copy of this file in the Backup directory you created earlier

```
$ cp Sequences.tar /home/genomics/Backup/
```

3. Un archive the directory

```
$ tar –xvf Sequences.tar
```

4. Unzip the read files

```
$ gunzip Sequences/E_coli_Sequence1.fq.gz
$ gunzip Sequences/E_coli_Sequence2.fq.gz
```

OR

```
$ gunzip Sequences/E_coli_Sequence*
```

5. Rename the unarchived files – sequence_1.fq and sequence_2.fq

```
$ mv Sequences/E_coli_Sequence1.fq Sequences/sequence_1.fq
$ mv Sequences/E_coli_Sequence2.fq Sequences/sequence_2.fq
```

6. Delete the original .tar file

```
$ rm –i Sequences.tar
```

Any Questions So Far?

How Do You Install Software?

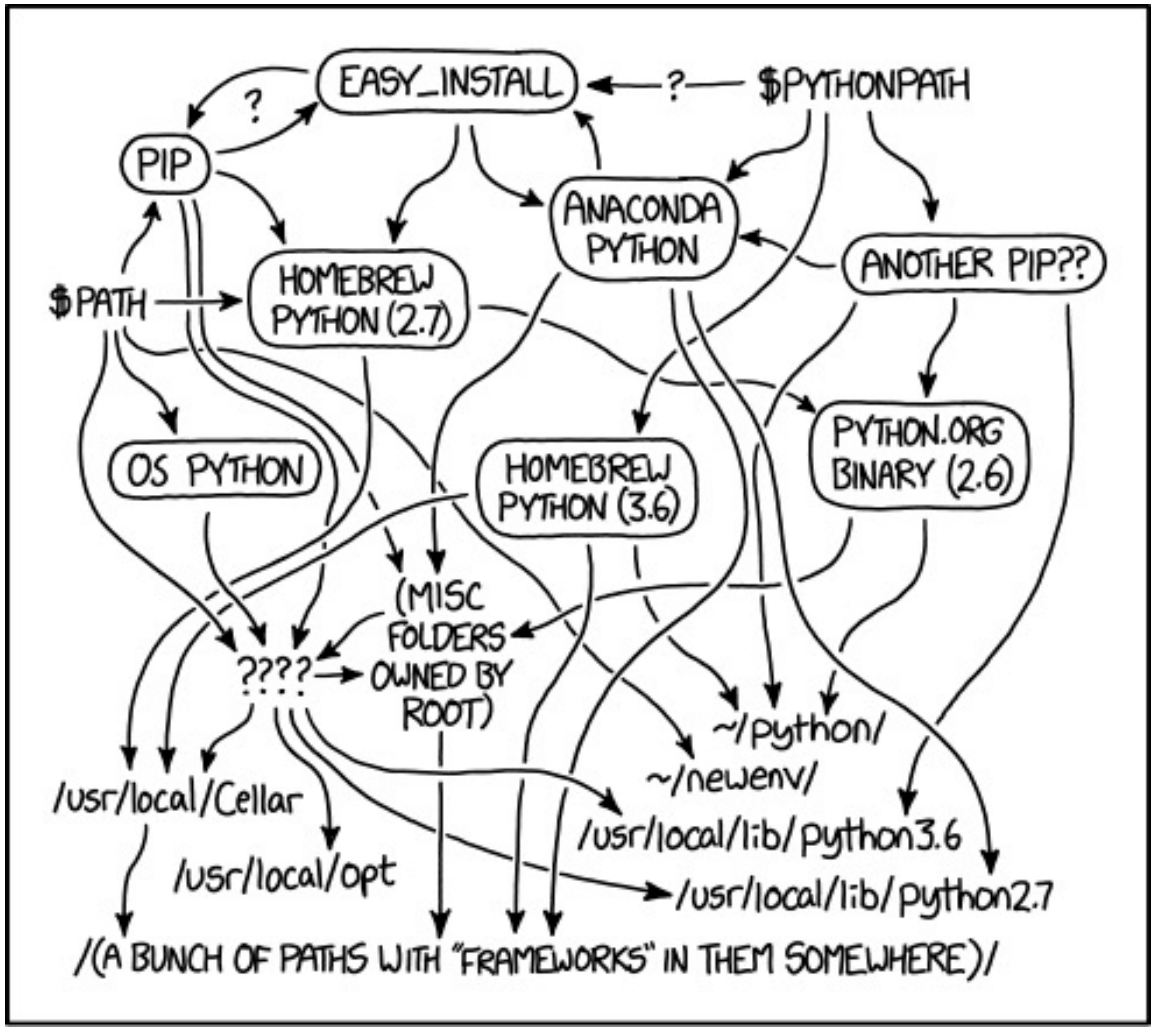
Some binary programs come with Ubuntu e.g. mkdir, ls etc…

However most software for bioinformatics needs to be installed.

For this week, we will learn how to install some of these.

If you're using a corporate or shared compute resource, you'll likely need to ask
your system administrator to do this.

IT IS NOT AS EASY AS IT SOUNDS



https://xkcd.com/1987/

Software Installation

The “get and compile” method!

Get the software from somewhere and then install in manually.

GitHub

GitHub is a fantastic repository containing a lot of bioinformatics software! It works using git so it’s easy to download.

There are others… sourceforge, personal webpages etc...
Commonly wget is used with these!

Challenge 3!

I want you to download and compile the software unicycler.

Here is the github page - https://github.com/rrwick/Unicycler

Make sure you are in your home directory.

Follow the instructions under “Build and run without installation”.

Take a read/glance of all of the information under “Requirements” and “Installation” – I just want you to read here!

What Have We Just Done?

```
git clone https://github.com/rrwick/Unicycler.git
```

This has downloaded the software from github

```
cd Unicycler
```

This has changed the location to this new directory

```
make
```

This has COMPILED the software

But what happens if you type unicycler on the command line now?

```
genomics@genomics:~$ unicycler
unicycler: command not found
genomics@genomics:~$
```

Why!? Compiling ≠ Installing!

Installation

For software to be truly installed, it needs to be added to the directory /usr/bin or /usr/local/bin

And this requires admin privileges which you don't have!

However you can still run the newly compiled software directly by using the path:

```
genomics@genomics:~/Unicycler$ ./unicycler-runner.py
usage: unicycler-runner.py [-h] [--help_all] [--version] [-1 SHORT1] [-2 SHORT2] [-s UNPAIRED] [-l LONG] -o OUT
                           [--verbosity VERBOSITY] [--min_fasta_length MIN_FASTA_LENGTH] [--keep KEEP] [-t THREADS]
                           [--mode {conservative,normal,bold}] [--linear_seqs LINEAR_SEQS] [--vcf]

Unicycler: an assembly pipeline for bacterial genomes
```

Dependencies

However! What if it still won't work…

What! I hear you cry! Why!?

Requirements

- Linux or macOS
- Python 3.4 or later
- C++ compiler with C++14 support:
  - GCC 4.9.1 or later
  - Clang 3.5 or later
  - ICC also works (though I don't know the minimum required version number)
- setuptools (only required for installation of Unicycler)
- For short-read or hybrid assembly:
  - SPAdes v3.6.2 or later (`spades.py`)
- For long-read or hybrid assembly:
  - Racon (`racon`)
- For polishing
  - Pilon (`pilon1.xx.jar`)
  - Java (`java`)
  - Bowtie2 (`bowtie2-build` and `bowtie2`)
  - Samtools v1.0 or later (`samtools`)
- For rotating circular contigs:
  - BLAST+ (`makeblastdb` and `tblastn`)

Dependencies

A lot of software needs other software to work…
Which in turn needs other software..
Which also needs other software...

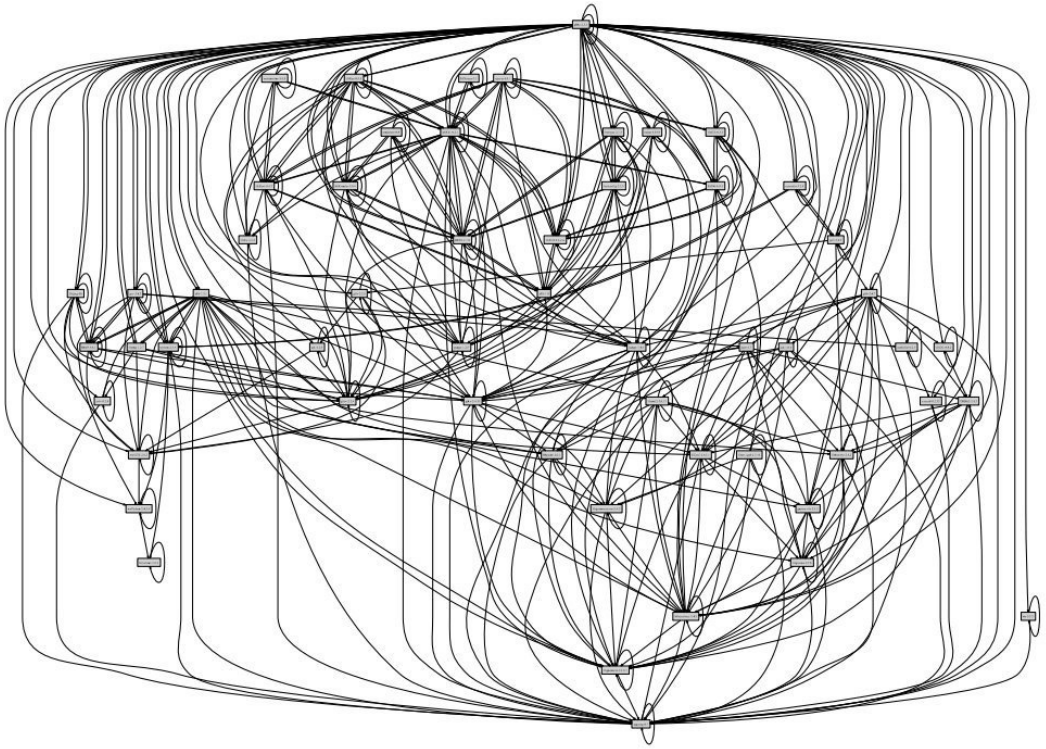


Figure 1.5.: The *dependency hell*: the runtime dependency graph of Mozilla Firefox

https://nixos.org/~eelco/pubs/phd-thesis.pdf

Software Managers

`apt-get install` OR `yum`

CONDA

`pip install`

Mamba

Linuxbrew

These are all *trying* to make your life easier...

Don’t forget cpanm for perl and install.packages() for R

Environments

Environments allow you to control which software (and dependencies) are available and when.

This is particularly useful during conflicts or when you want to use multiple versions of the same software.

Other systems such as Docker and module systems on clusters aim to do the same thing.

Environments

CONDA

This is your compute system.

You want to run qiime2. It's not installed.

You've installed conda.

Pipenv

python virtualenv

Further reading:
https://realpython.com/python-virtual-environments-a-primer/#the-conda-package-and-environment-manager

Environments

CONDA

To use conda, first we need to install it.

1. Install Miniconda:

```
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
bash Miniconda3-latest-Linux-x86_64.sh
# follow through installation prompts
rm Miniconda3-latest-Linux-x86_64.sh
```

2. Install Mamba on the base Conda environment:

```
conda install -n base -c conda-forge mamba --yes
conda activate base
```

Environments

CONDA

You use conda to create an environment for qiime2.

```
wget https://data.qiime2.org/distro/core/qiime2-2022.2-py38-linux-conda.yml
conda env create -n qiime2-2022.2 --file qiime2-2022.2-py38-linux-conda.yml

# OPTIONAL CLEANUP rm qiime2-2022.2-py38-linux-conda.yml
```

You then activate this environment.

```
conda activate qiime2-2022.2
```

Environments

CONDA

You then deactivate this environment.

```
source deactivate qiime2-2018.8
```

qiime2 is no longer available.

Environments

CONDA

You can now create your own environments.

```
conda env create -n microbiome
```

OR:

```
conda env create --name microbiome python=3.9  #Specify python version
```

OR:

```
conda env create --name microbiome python=3.9 scipy=0.17.3  #Specify software to be installed
```

OR:

```
conda env create -n microbiome
conda install -n microbiome scipy
```

You then activate this environment.

```
conda activate microbiome
```

Can we try to install dada2 in our newly created environment?

Tip: Look at the anaconda page

anaconda.org/bioconda/bioconductor-dada2

bioconda / packages / bioconductor-dada2 1.22.0

0

Accurate, high-resolution sample inference from amplicon sequencing data

Conda | Files | Labels | Badges

License: LGPL-2
Home: https://bioconductor.org/packages/3.14/bioc/html/dada2.html
370379 total downloads
Last upload: 5 months and 22 days ago

Installers

Info: This package contains files in non-standard labels.

conda install

linux-64 v1.22.0
osx-64 v1.22.0

To install this package with conda run one of the following:

```
conda install -c bioconda bioconductor-dada2
conda install -c bioconda/label/gcc7 bioconductor-dada2
conda install -c bioconda/label/cf201901 bioconductor-dada2
```

Confused!?
Don’t worry, you
aren’t the only one!

This section is just to make you aware of software installation techniques so you aren’t surprised when you get home and can’t find SAMtools on your system!

What We're Going To Do

THIS MORNING

- Getting Connected ✓
- Introduction to Unix ✓
- Software Installation ✓

THIS AFTERNOON

- Introduction to Sequencing
- Checking the Quality of Sequencing Data

For after the workshop…

A Quick Guide to Organizing Computational Biology Projects.
Noble (2009) PLoS Computational Biology
http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1000424

Best Practices for Scientific Computing.
Wilson *et al.* (2014) PLoS Biology
http://journals.plos.org/plosbiology/article?id=10.1371/journal.pbio.1001745

Codecademy
https://www.codecademy.com/learn/learn-the-command-line

Hackerrank
https://www.hackerrank.com/domains/shell

Lots of text books

Google!!