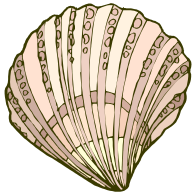
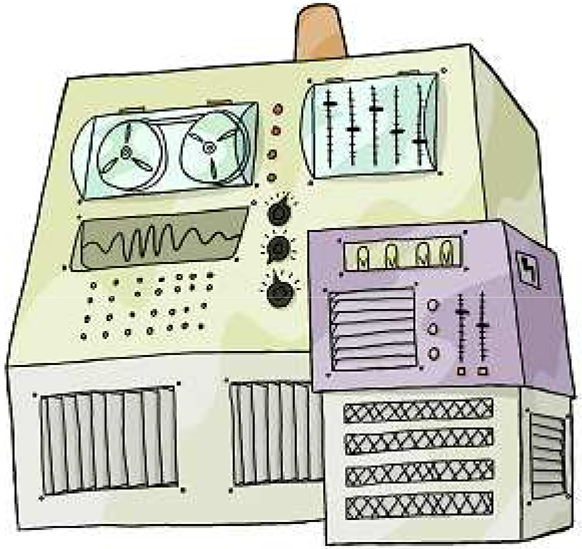
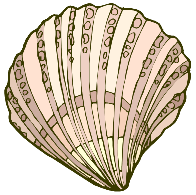
The Unix Shell

Permissions



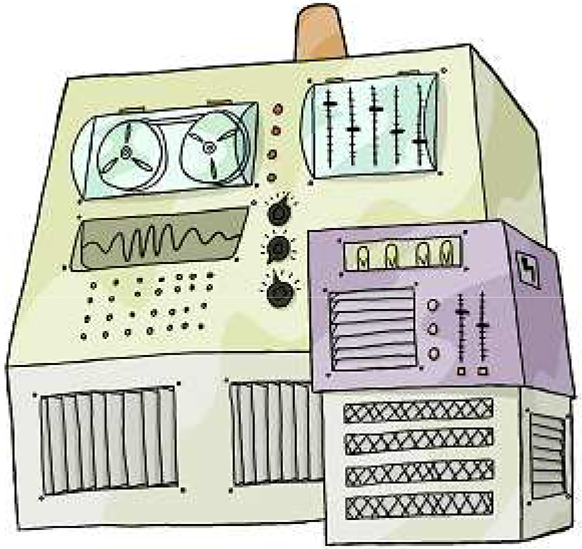
shell

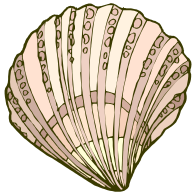




shell

`pwd, mkdir, cp, ...`

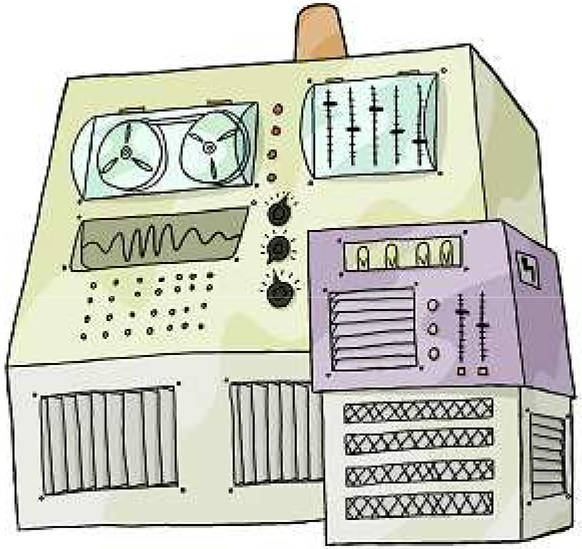


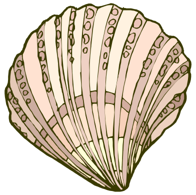


shell

```
pwd, mkdir, cp, ...
```

*



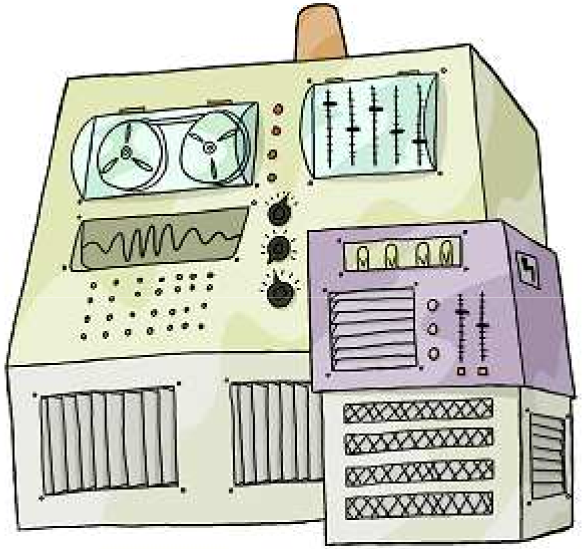


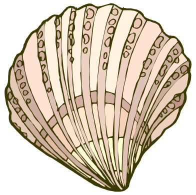
shell

pwd, mkdir, cp, ...

*

>, |



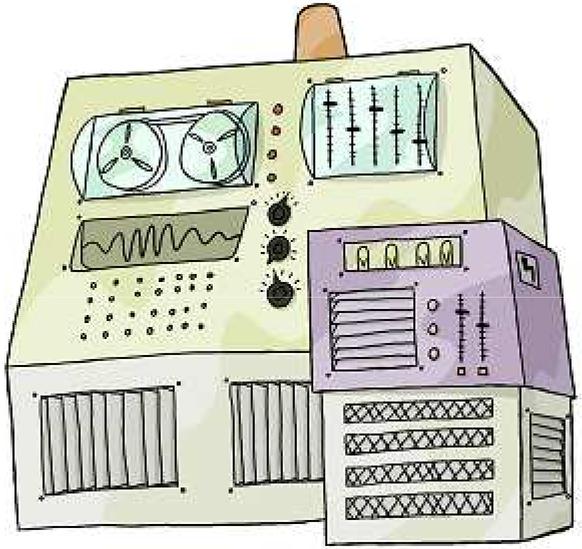


shell

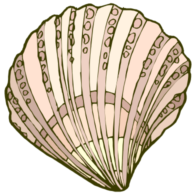
`pwd, mkdir, cp, ...`

`*`

`>, |`

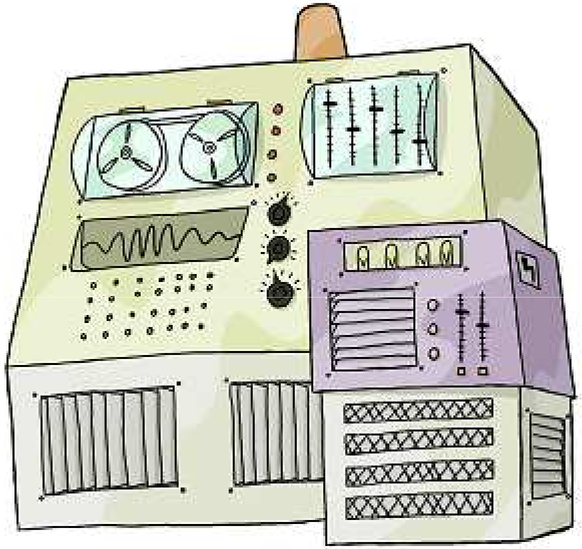


Who can see what?



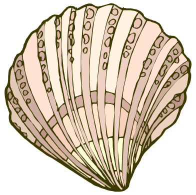
shell

```
pwd, mkdir, cp, ...

*

>, |
```



Who can see what?

change

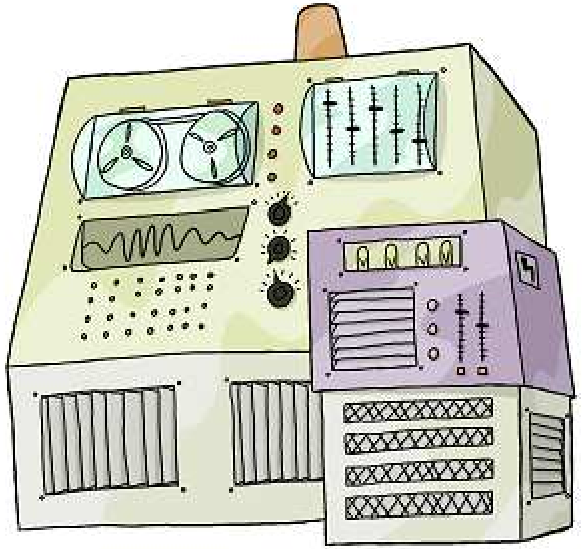


shell

`pwd, mkdir, cp, ...`

`*`

`>, |`



Who can see what?

change

run

Simplified version of Unix permissions

Simplified version of Unix permissions

Windows uses similar concepts...

Simplified version of Unix permissions

Windows uses similar concepts...

...but there is no exact translation between the two

user

user

Has unique *user name* and *user ID*

user

Has unique *user name* and *user ID*

User name is text: "imhotep", "larry", "vlad", ...

user

Has unique *user name* and *user ID*

User name is text: "imhotep", "larry", "vlad", ...

User ID is numeric (easier for computer to store)

user

group

user

group

Has unique *group name* and *group ID*

Has unique *group name* and *group ID*

User can belongs to zero or more groups

user

group

Has unique *group name* and *group ID*

User can belongs to zero or more groups

List is usually stored in `/etc/group`

user

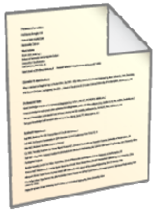
group

all

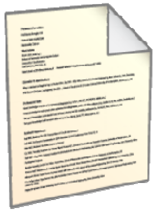
user | group | all

Everyone else

user | group | all

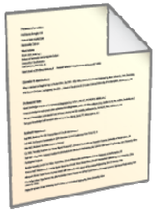


Has user and group IDs

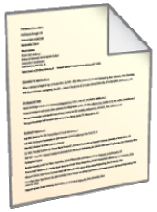


	user	group	all
read			

	user	group	all
read			
write			

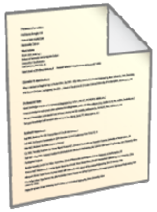


	user	group	all
read			
write			
execute			



	user	group	all
read	✓	✓	✗
write	✓	✗	✗
execute	✗	✗	✗

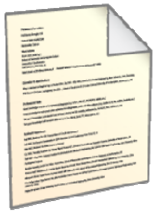
File's owner can read and write it



	user	group	all
read	✓	✓	✗
write	✓	✗	✗
execute	✗	✗	✗

File's owner can read and write it

Others in group can read



	user	group	all
read	✓	✓	✗
write	✓	✗	✗
execute	✗	✗	✗

File's can read and write it

Others in group can read

That's all

	user	group	all
read	✓	✓	✗
write	✓	✗	✗
execute	✗	✗	✗

```
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$
```

```
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$
```

```
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$
```

means "executable"

```
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$
```

```
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$
```

name

```
$ cd labs
$ ls
safety.txt    setup       waiver.txt
$ ls -F
safety.txt    setup*      waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$
```

last modified

```
$ cd labs
$ ls
safety.txt    setup     waiver.txt
$ ls -F
safety.txt    setup*    waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio  1158 2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988 2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312 2010-07-11 08:23 waiver.txt
$
```

size (in bytes)

```
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$
```

group owner

```
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$
```

user owner

```
$ cd labs
$ ls
safety.txt    setup     waiver.txt
$ ls -F
safety.txt    setup*    waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$
```

don't care (for now)

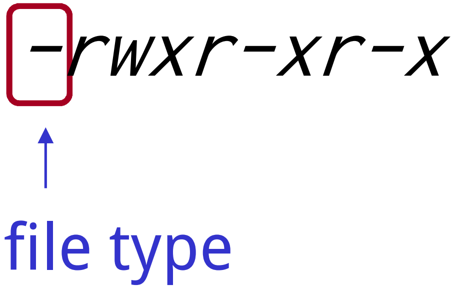
```
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$
```

permissions

```
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$
```

-rwxr-xr-x

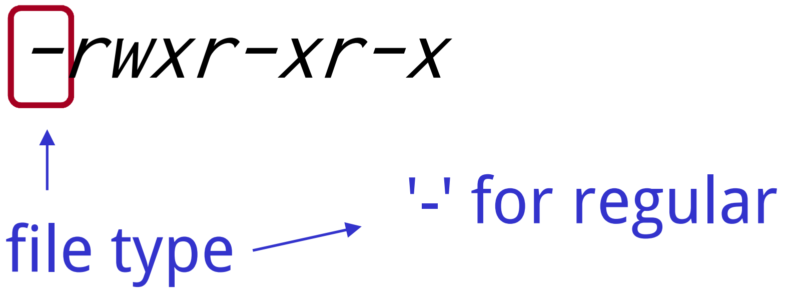
```
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$
```



-rwxr-xr-x

file type

```
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$
```

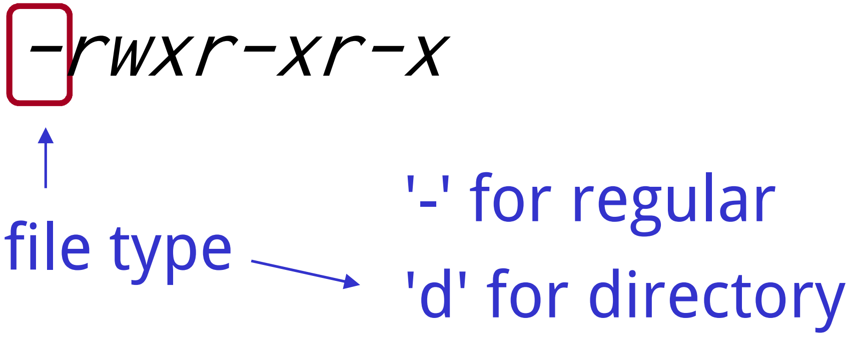


-rwxr-xr-x

file type

'-' for regular

```
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$
```



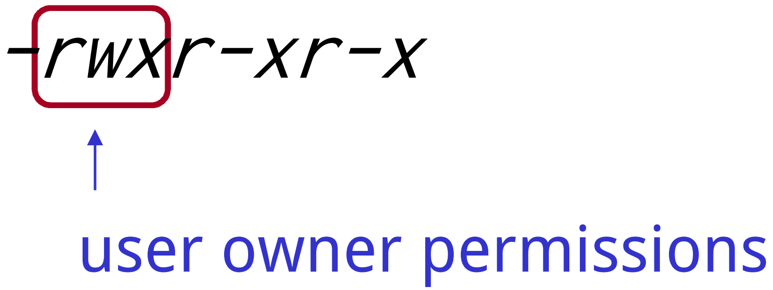
-rwxr-xr-x

file type

'-' for regular

'd' for directory

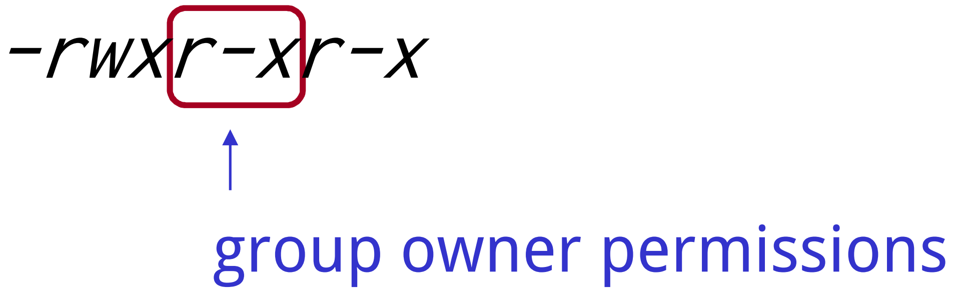
```
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$
```



-rwxr-xr-x

user owner permissions

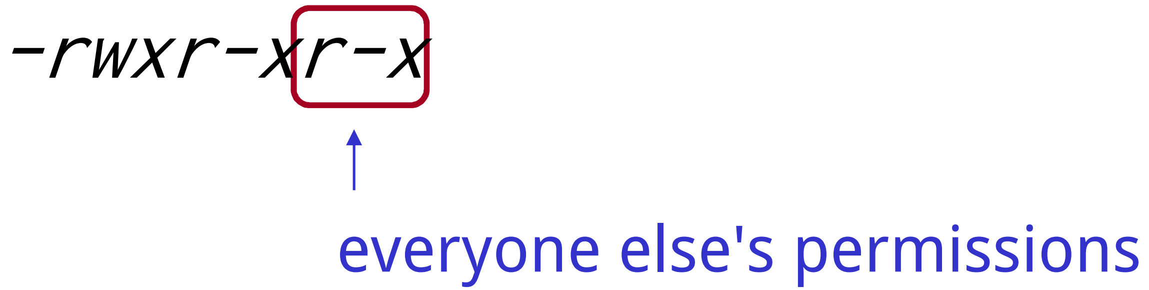
```
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$
```



-rwxr-xr-x

group owner permissions

```
$ cd labs
$ ls
safety.txt    setup      waiver.txt
$ ls -F
safety.txt    setup*     waiver.txt
$ ls -l
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$
```



```
$ ls -a -l
drwxr-xr-x 1 vlad bio     0  2010-08-14 09:55 .
drwxr-xr-x 1 vlad bio  8192  2010-08-27 23:11 ..
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$
```

```
$ ls -a -l
drwxr-xr-x 1 vlad bio     0  2010-08-14 09:55 .
drwxr-xr-x 1 vlad bio  8192  2010-08-27 23:11 ..
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$
```

```
$ ls -a -l
drwxr-xr-x 1 vlad bio     0  2010-08-14 09:55 .
drwxr-xr-x 1 vlad bio  8192  2010-08-27 23:11 ..
-rw-rw-r-- 1 vlad bio  1158  2010-07-11 08:22 safety.txt
-rwxr-xr-x 1 vlad bio 31988  2010-07-23 20:04 setup
-rw-rw-r-- 1 vlad bio  2312  2010-07-11 08:23 waiver.txt
$
```

What does "execute" mean for directories?

What does "execute" mean for directories?

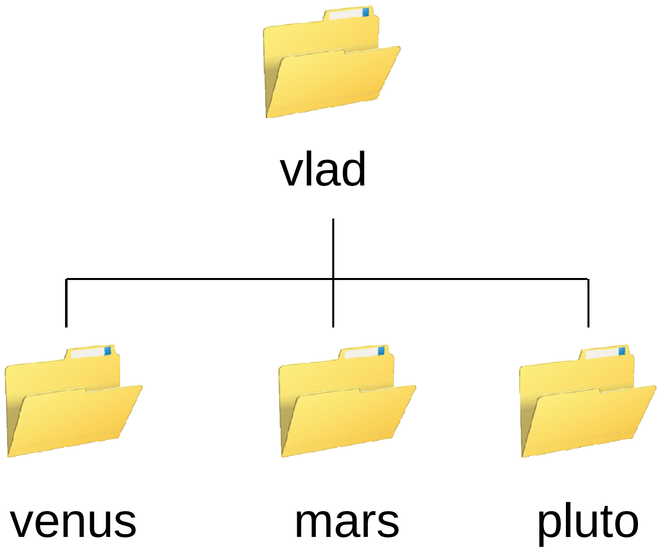
Gives the right to *traverse*

the directory

What does "execute" mean for directories?

Gives the right to *traverse*

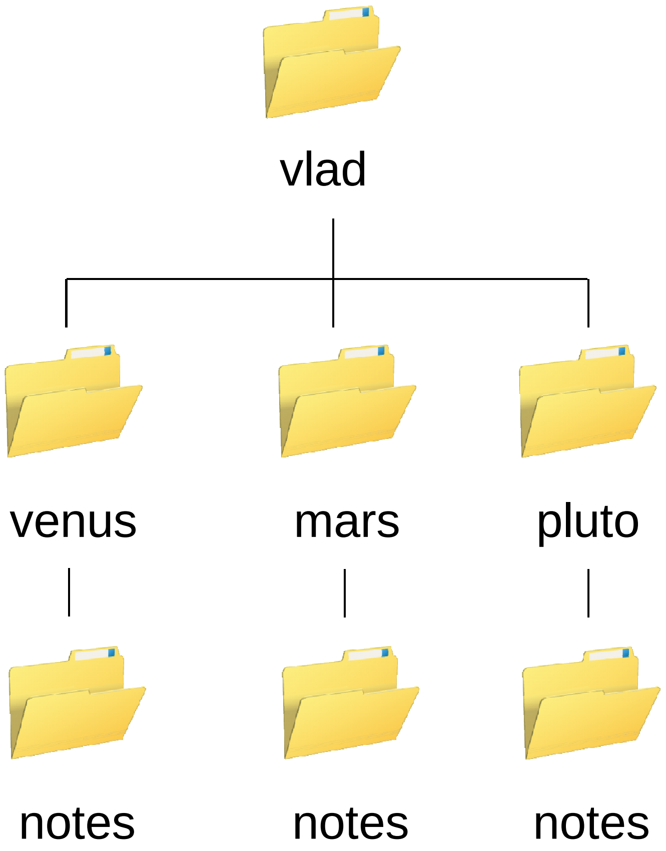
the directory



What does "execute" mean for directories?

Gives the right to *traverse*

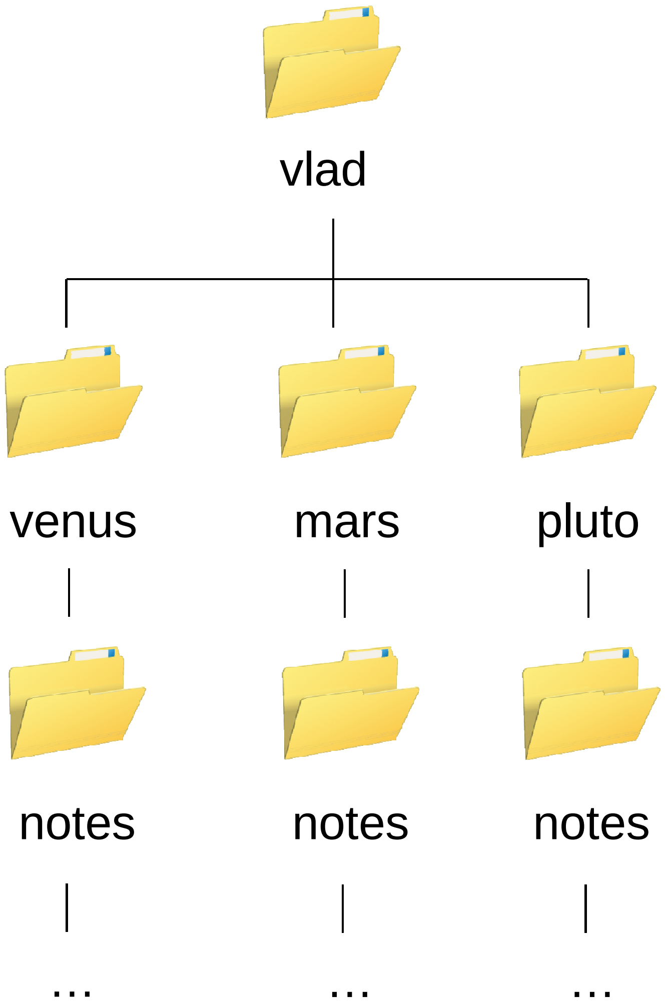
the directory



What does "execute" mean for directories?

Gives the right to *traverse*

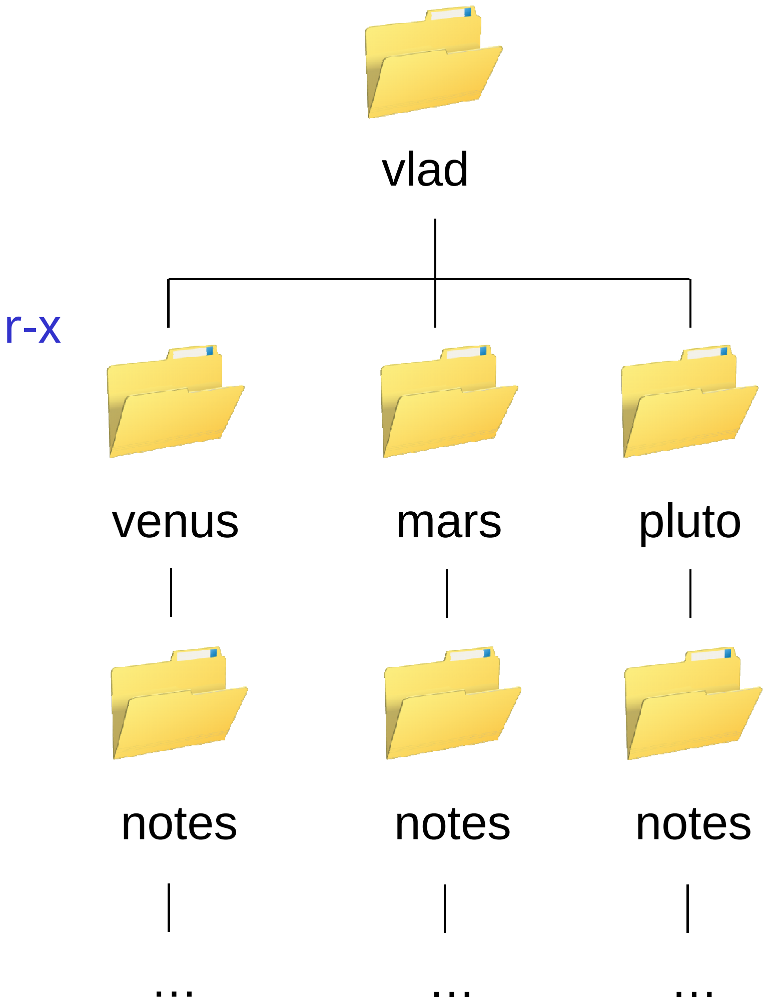
the directory



What does "execute" mean for directories?

Gives the right to *traverse*

the directory

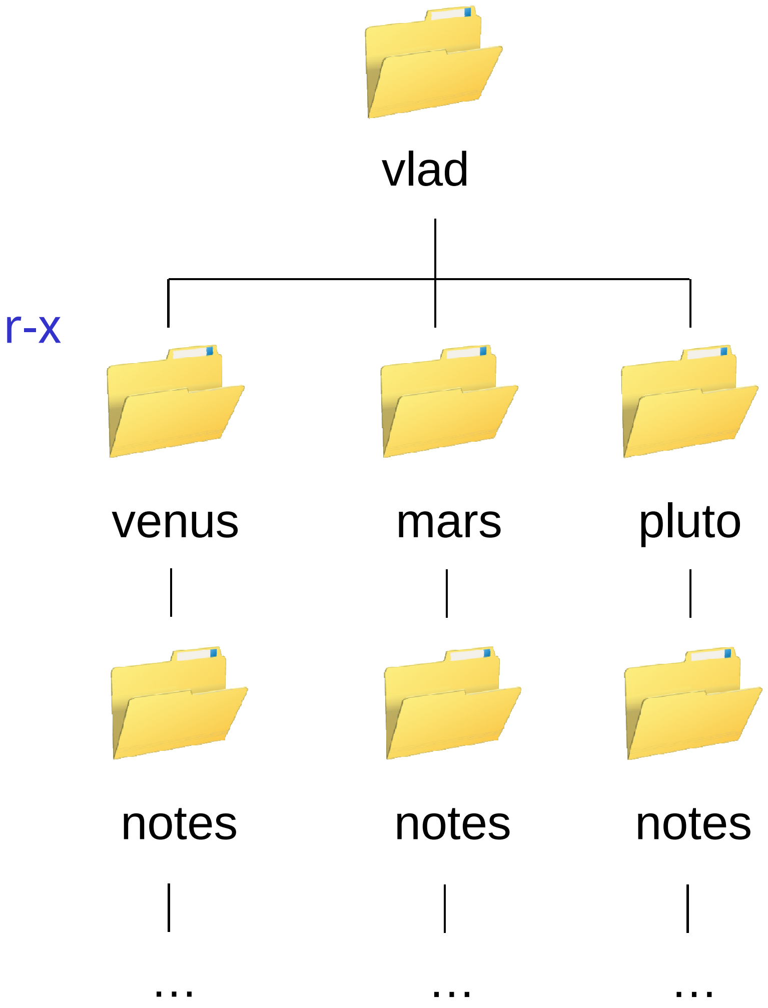


What does "execute" mean for directories?

Gives the right to *traverse*

the directory

```
$ ls venus venus/notes
```

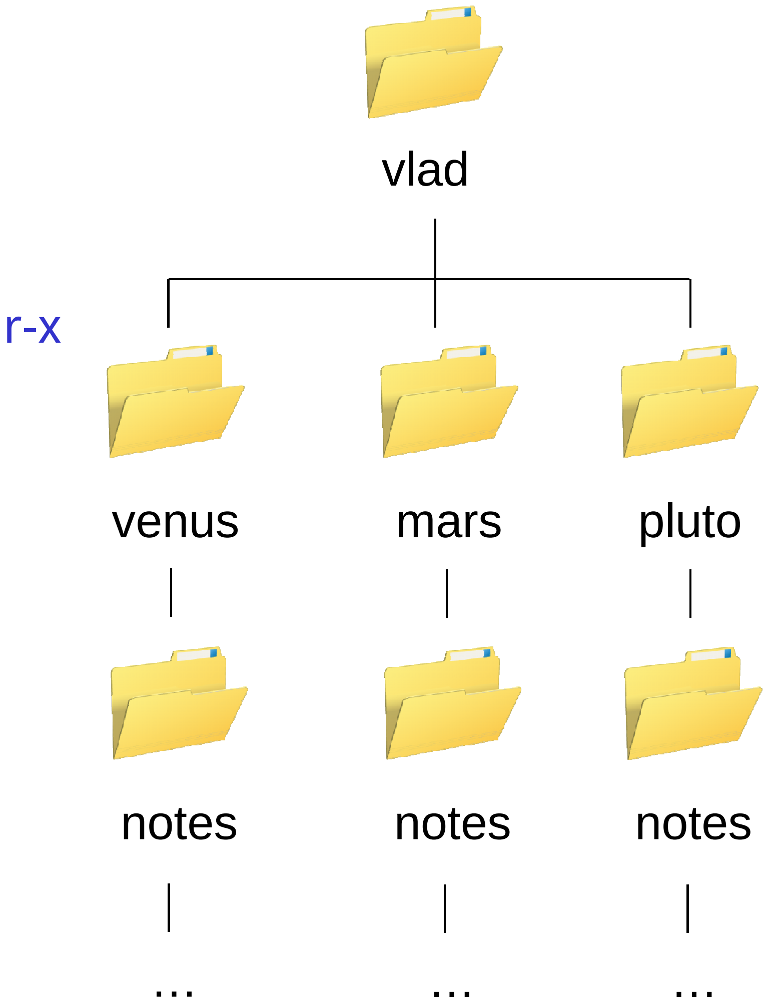


What does "execute" mean for directories?

Gives the right to *traverse*

the directory

```
$ ls venus venus/notes
```
✓



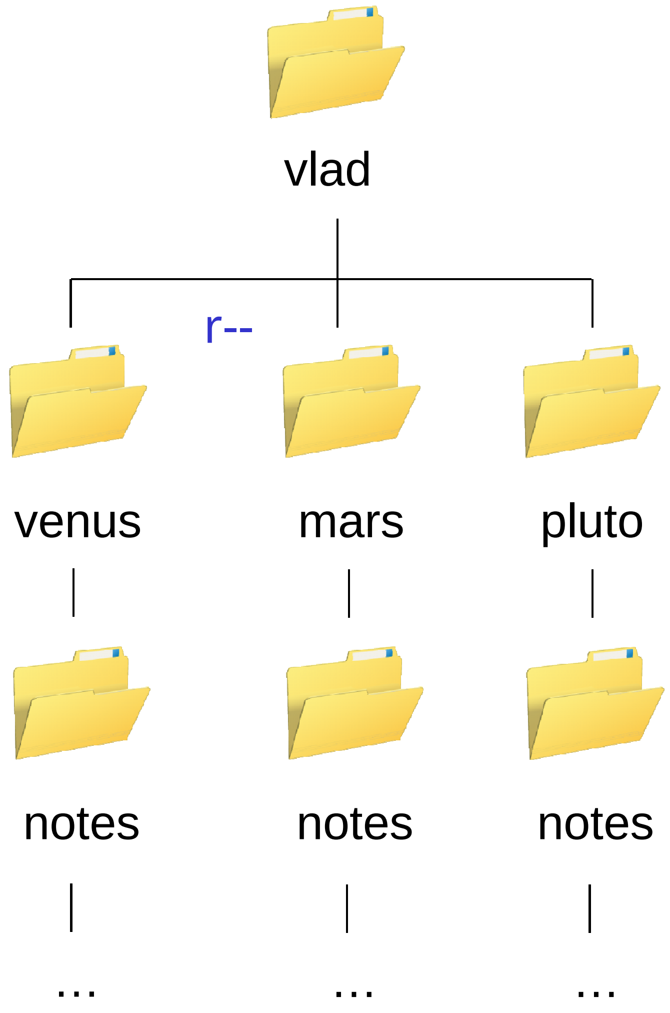
What does "execute" mean for directories?

Gives the right to *traverse*

the directory

```
$ ls venus venus/notes
```

✓

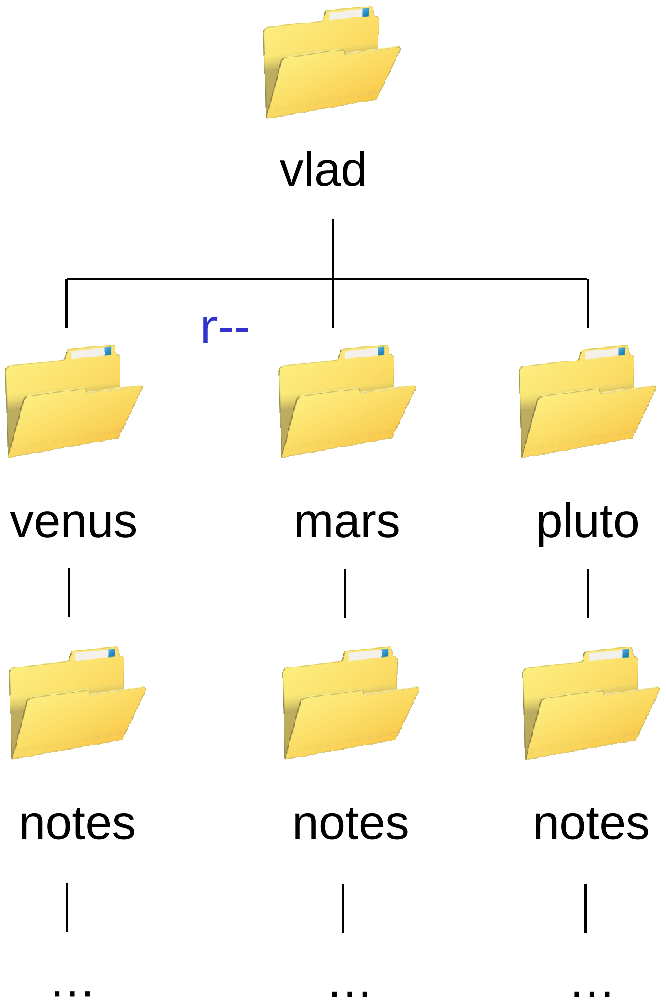


What does "execute" mean for directories?

Gives the right to *traverse*

the directory

```
$ ls venus venus/notes     ✓
$ ls mars mars/notes       ✓
```



What does "execute" mean for directories?

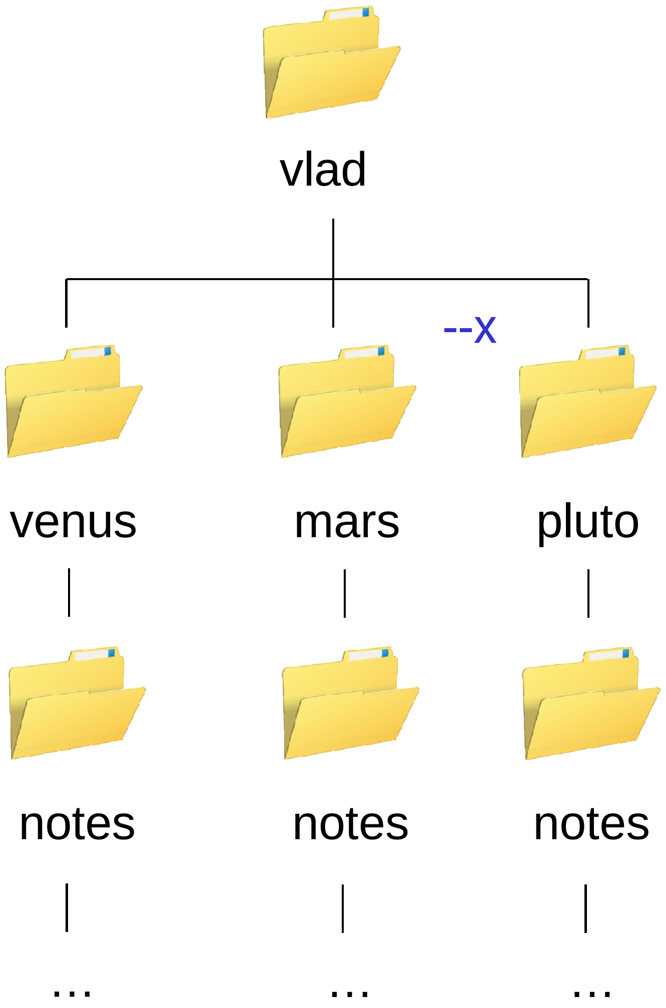
Gives the right to *traverse*

the directory

```
$ ls venus venus/notes
$ ls mars mars/notes
```

✓

✓

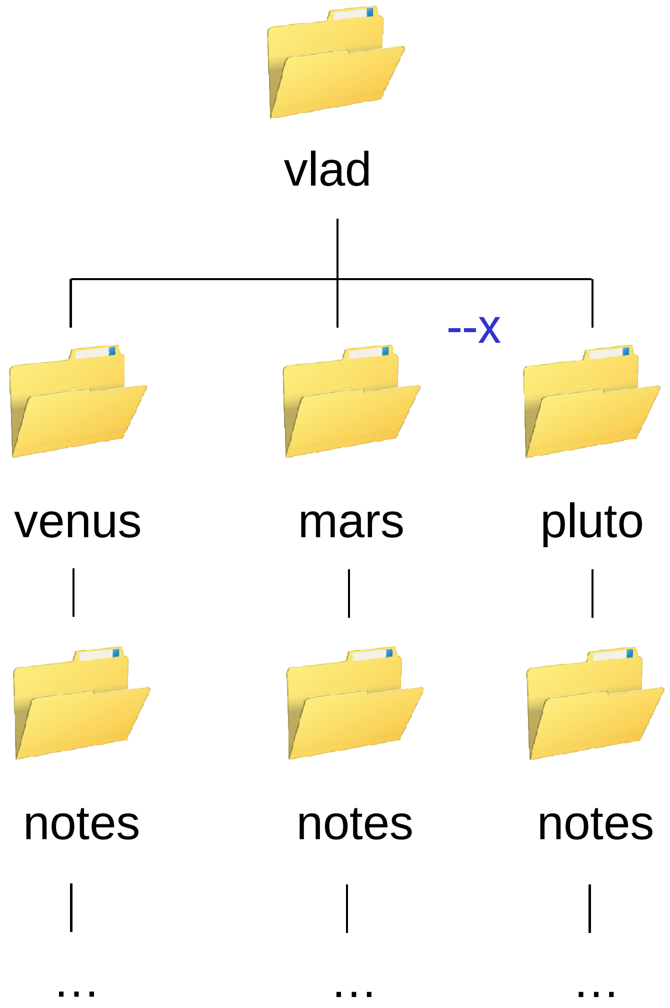


What does "execute" mean for directories?

Gives the right to *traverse*

the directory

```
$ ls venus venus/notes      ✓
$ ls mars mars/notes        ✓
$ ls pluto                  ✗
```

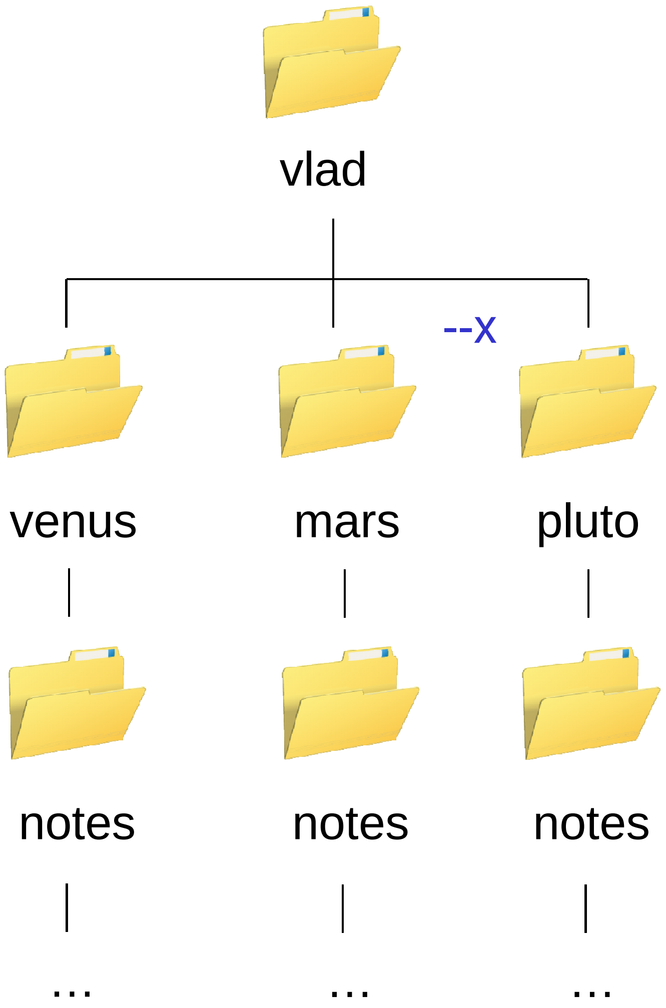


What does "execute" mean for directories?

Gives the right to *traverse*

the directory

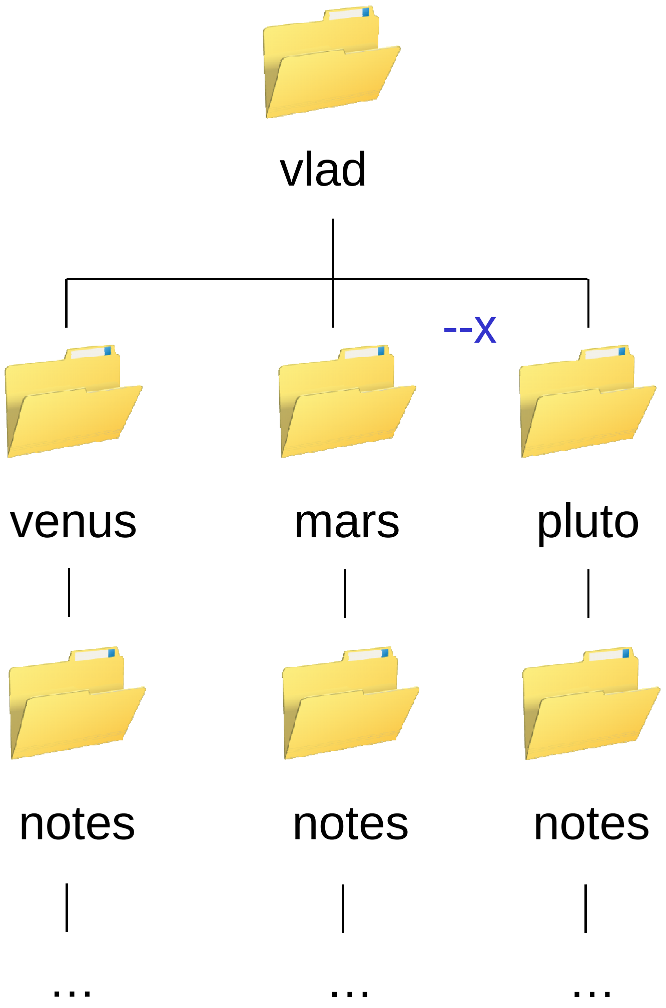
```
$ ls venus venus/notes     ✓
$ ls mars mars/notes       ✓
$ ls pluto                 ✗
$ ls pluto/notes
```



What does "execute" mean for directories?

Gives the right to *traverse* the directory

```
$ ls venus venus/notes     ✓
$ ls mars mars/notes       ✓
$ ls pluto                 ✗
$ ls pluto/notes           ✓
```



vlad

--x

venus

mars

pluto

notes

notes

notes

…

…

…

Change permission with `chmod` (<u>ch</u>ange <u>mod</u>e)

Change permission with chmod (change mode)

```
$ ls -l final.grd
-rwxrwxrwx 1 vlad bio  4215  2010-08-29 22:30 final.grd
```

Change permission with chmod (change mode)

```
$ ls -l final.grd
-rwxrwxrwx 1 vlad bio  4215  2010-08-29 22:30 final.grd
```

Everyone can read it

Change permission with chmod (change mode)

```
$ ls -l final.grd
-rwxrwxrwx 1 vlad bio  4215  2010-08-29 22:30 final.grd
```

Everyone can read it

Modify it

Change permission with chmod (change mode)

```
$ ls -l final.grd
-rwxrwxrwx 1 vlad bio  4215  2010-08-29 22:30 final.grd
```

↑

Everyone can read it

Modify it

Try to run it (which probably doesn't make sense)

Change permission with chmod (change mode)

```
$ ls -l final.grd
-rwxrwxrwx 1 vlad bio  4215  2010-08-29 22:30 final.grd
$ chmod u=rw final.grd
$
```

Change permission with chmod (change mode)

```
$ ls -l final.grd
-rwxrwxrwx 1 vlad bio  4215  2010-08-29 22:30 final.grd
$ chmod u=rw final.grd
$
```

User (u) has read-write (rw)

Change permission with chmod (change mode)

```
$ ls -l final.grd
-rwxrwxrwx 1 vlad bio  4215  2010-08-29 22:30 final.grd
$ chmod u=rw final.grd
$ ls -l final.grd
-rw-rwxrwx 1 vlad bio  4215  2010-08-30 08:19 final.grd
$
```

Change permission with chmod (change mode)

```
$ ls -l final.grd
-rwxrwxrwx 1 vlad bio  4215  2010-08-29 22:30 final.grd
$ chmod u=rw final.grd
$ ls -l final.grd
-rw-rwxrwx 1 vlad bio  4215  2010-08-30 08:19 final.grd
$ chmod g=r final.grd; ls -l final.grd
-rw-r--rw- 1 vlad bio  4215  2010-08-30 08:19 final.grd
$
```

Change permission with chmod (change mode)

```
$ ls -l final.grd
-rwxrwxrwx 1 vlad bio  4215  2010-08-29 22:30 final.grd
$ chmod u=rw final.grd
$ ls -l final.grd
-rw-rwxrwx 1 vlad bio  4215  2010-08-30 08:19 final.grd
$ chmod g=r final.grd; ls -l final.grd
-rw-r--rw- 1 vlad bio  4215  2010-08-30 08:19 final.grd
$
```

Use ';' to put multiple commands on a single line

Change permission with chmod (change mode)

```
$ ls -l final.grd
-rwxrwxrwx 1 vlad bio  4215  2010-08-29 22:30 final.grd
$ chmod u=rw final.grd
$ ls -l final.grd
-rw-rwxrwx 1 vlad bio  4215  2010-08-30 08:19 final.grd
$ chmod g=r final.grd; ls -l final.grd
-rw-r--rw- 1 vlad bio  4215  2010-08-30 08:19 final.grd
$ chmod a= final.grd; ls -l final.grd
-rw-r----- 1 vlad bio  4215  2010-08-30 08:20 final.grd
```

Change permission with chmod (change mode)

```
$ ls -l final.grd
-rwxrwxrwx 1 vlad bio  4215  2010-08-29 22:30 final.grd
$ chmod u=rw final.grd
$ ls -l final.grd
-rw-rwxrwx 1 vlad bio  4215  2010-08-30 08:19 final.grd
$ chmod g=r final.grd; ls -l final.grd
-rw-r--rw- 1 vlad bio  4215  2010-08-30 08:19 final.grd
$ chmod a= final.grd; ls -l final.grd
-rw-r----- 1 vlad bio  4215  2010-08-30 08:20 final.grd
```

No permissions at all

Again, things are different on Windows

Again, things are different on Windows

Permissions defined by Access Control Lists (ACLs)

Again, things are different on Windows

Permissions defined by Access Control Lists (ACLs)

A list of (who, what) pairs

Again, things are different on Windows

Permissions defined by Access Control Lists (ACLs)

A list of (who, what) pairs

More flexible...

Again, things are different on Windows

Permissions defined by Access Control Lists (ACLs)

A list of (who, what) pairs

More flexible...

...but more complex to administer and understand

Again, things are different on Windows

Permissions defined by Access Control Lists (ACLs)

A list of (who, what) pairs

More flexible...

...but more complex to administer and understand

Some flavors of Unix provide ACLs, but hardly anyone uses them

Create your own commands

Create your own commands

```
$ cat > smallest
```

Create your own commands

```
$ cat > smallest
```

No input file specified, so read from keyboard

Create your own commands

```
$ cat > smallest
```

Send output to a file called `smallest`

Create your own commands

```
$ cat > smallest
wc -l *.pdb | sort | head -1
```

Create your own commands

```
$ cat > smallest
wc -l *.pdb | sort | head -1
^D
$
```

Create your own commands

```
$ cat > smallest
wc -l *.pdb | sort | head -1
^D
$
```

Ctrl-D means "end of input" in Unix

Create your own commands

```
$ cat > smallest
wc -l *.pdb | sort | head -1
^D
$
```

Ctrl-D means "end of input" in Unix

Ctrl-Z does the same thing in Windows

Create your own commands

```
$ cat > smallest
wc -l *.pdb | sort | head -1
^D
$ chmod u+x smallest
$
```

Create your own commands

```
$ cat > smallest
wc -l *.pdb | sort | head -1
^D
$ chmod u+x smallest
$
```

Give the user owner permission to run this file

Create your own commands

```
$ cat > smallest
wc -l *.pdb | sort | head -1
^D
$ chmod u+x smallest
$ ./smallest
```

Create your own commands

```
$ cat > smallest
wc -l *.pdb | sort | head -1
^D
$ chmod u+x smallest
$ ./smallest
```

Put `./` at the front to be sure of running the `smallest` that it's *this* directory

Create your own commands

```
$ cat > smallest
wc -l *.pdb | sort | head -1
^D
$ chmod u+x smallest
$ ./smallest
  9  methane.pdb
$
```

Create your own commands

```
$ cat > smallest
wc -l *.pdb | sort | head -1
^D
$ chmod u+x smallest
$ ./smallest
  9  methane.pdb
$
```

Try doing *that* with a desktop full of GUIs

software carpentry

created by

Greg Wilson

August 2010