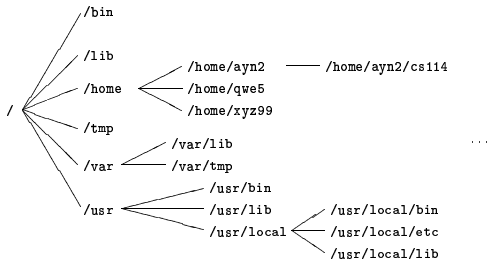
Unix directory structure



In Unix, directories form a tree structure. At the top of the three, there is a "root" directory /. Each directory can contain files, subdirectories, **or both**.

Absolute paths

One way of referring to files and directories is by giving an *absolute* path to it from the root of the tree — e.g. `/home/ayn2/Syllabus.txt` or `/home/ayn2/homeworks`. If you want to emphasize that something is a directory, you can put a / at the end — `/home/ayn2/homeworks/`. Notice that absolute paths always start with a "/"

Managing current directory - `cd`

- When you log in, your current directory is your *home* directory.
- `pwd` **p**rints your current **w**orking **d**irectory
- `cd` **c**hanges you current **d**irectory. You can either run `cd` *path* or just run `cd` to get back to your home directory.
- Usually you can refer to your home directory as `$HOME` or simply `~`. For example, `cd ~/cs114`

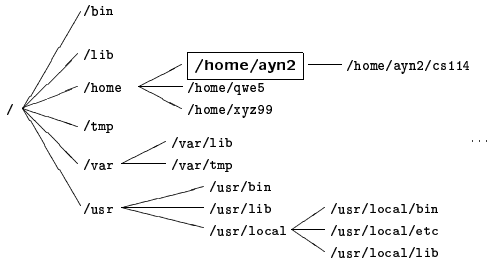
Listing files - `ls`

`ls` *[options] directory* lists files and subdirectories in *directory*. Without a *directory* argument, it will list files and subdirectpries in current directory. Possible options include:

- `-l` long listing
- `-a` list all files and subdirectories (including the "dot" ones)
- `-A` list all files and subdirectories, except for `.` and `..`

Relative paths

Paths can be also specified relative to the current directory.



From the current directory we can go:

- Down the tree. For example, `cs114`, `cs114/classlist`. The single dot also stands for the current directory, so to emphasize that a path is relative, we may say `./cs114`, `./cs114/classlist`
- Up the tree — `..` stands for going one level up. So, `..` is same as `/home` and `../..` is the same as /.
- First up and then down. For example, `../xyz99` is the same as `/home/xyz99`

Manipulating files and directories

- `mkdir` *directory* creates *directory*
- `rmdir` *directory* removes *directory* (only if it's empty and you are not inside it).
- `touch` *file* creates an empty *file*.
- `rm` *file* removes *file*.
- `mv` *old_name new_name* renames a file (and moves it if names refer to different directories).
- `mv` $file_1$ $file_2$ ... *directory* moves files into a directory.
- `cp` *old_name new_name* copies a file.
- `cp` $file_1$ $file_2$ ... *directory* copies files into a directory.