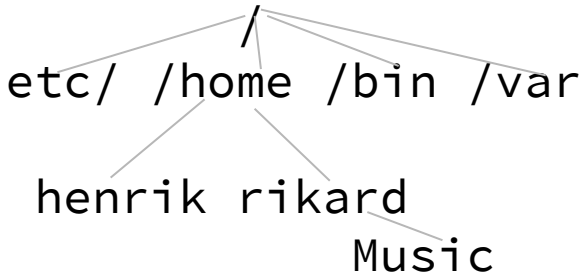
Introduction to Bash video lecture

02 File system

File system

- In order to work in the terminal using Bash, you need to understand the concept of a file syste
- An abstraction of a single-rooted tree hierarchy
- There is a *root directory* called / (slash)
- Every directory can have files and directories (creating an hierarchy)



Parent directory

- Every directory (except /) has a *parent directory*
- If / contains home then / is the parent directory of home
- A directory can have only one parent directory (a file or directory cannot exist in more than one directory - it has only one place in the file system)
- This allows us to identify a file or directory uniquely using a *path* to the file or directory
- There are two kinds of paths
  - Absolute paths - start from / and all the way “down” to the file or directory
  - Relative paths - start from some other directory and leads “up” or “down” to the file or directory

Absolute paths

- Start from the root directory and leads down to the file or directory
  `/home/rikard/music/classical/bach/cantata_bwv_105.mp3`
- The directories between the root and the file or directory are separated by slashes
  `/home/rikard/music/classical/bach/cantata_bwv_105.mp3`
- An absolute path is *canonical* (“the one, true path”) and works from any directory, since you cannot mistake what it means
- Compare:
  -Meet me in Paris
  -Paris, Texas or Paris, France?
  -No, Paris, Denmark!

Relative paths

- Assumes prior knowledge, a context, of where you start from
- In /home/rikard the following relative path:
  `music/classical/bach/cantata_bwv_105.mp3`
  means the same as the absolute path:
  `/home/rikard/music/classical/bach/cantata_bwv_105.mp3`
- Never start with / (only absolute paths start from the root)
- Can be a problem when used in a script or program (the script will use the directory it was run from as the starting point)
- Use .. signifying “up” to parent
- Use . signifying “here” (current directory) and
- ~ Signifying “your home directory”

Using .. to signify parent

- From `/home/rikard/music/classical/bach/` to `/home/rikard/music/classical/beethoven/` you need to go “up” to `/home/rikard/music/classical/` and down to `/home/rikard/music/classical/beethoven`
- Using a relative path from bach you would say `../beethoven` (first up to parent, then down to beethoven)

Using . to signify “here” or current directory

- You want to move `an_die_freude.mp3` from `/home/rikard/music/classical/bach/` to `/home/rikard/music/classical/beethoven/`
- Using a relative path from `beethoven` you would say `cp ../bach/an_die_freude.mp3 .`
- Meaning move the file from bach/ in my parent directory to here
- “here” meaning `/home/rikard/music/classical/beethoven/` since that is your current directory

Using ~ (tilde) to signify your home

- Every user has a home directory, typically in /home/username where username is your user's name (e.g. rikard on Rikard's computer)
- Allows us to have many users on one computer
- Keeps the users' files and directories separated an private
- You end up in your home directory when you start a shell
- You can use ~ as a shortcut for the absolute path to your home

```
rikard@newdelli:~/opt/progund/intro-it/bash-scripting-intro$ echo ~
/home/rikard
rikard@newdelli:~/opt/progund/intro-it/bash-scripting-intro$ cd ~
rikard@newdelli:~$
```

- Notice how the prompt changes

Paths

- Commands involving files or directories use paths to understand exactly what file or directory you are talking about
- For instance if you copy or move a file, the command needs to know:
  - What file do you want to copy/move?
  - Where do you want to copy/move it?
- You absolutely need to understand absolute and relative paths if you are to study or work within IT
- Take time to study and learn this, it is one of the most common sources for confusion in programming courses and operating systems courses

Current directory

- An abstraction to let us think that we “are in a directory”
- Is typically shown as a part of the prompt
- Stored in a shell variable called $PWD

```
rikard@newdelli:~/opt/progund/intro-it/bash-scripting-intro$ echo $PWD
/home/rikard/opt/progund/intro-it/bash-scripting-intro
```

- You can use . (a single dot) to signify current directory (or “here”)

  `cp /path/to/some/file .` (copy some file here)
- A relative path starts (implicitly or explicitly) from the current directory

  `cp some/file ~/my_files/` same as:

  `cp ./some/file ~/my_files/`
- Use the `pwd` command to print the path to current directory

Scripts and commands have a current directory

- When you run a command from a directory, the command knows where you were when you issued the command (your current directory)
- This means that commands that use paths can work
- If you say `ls` the command will list the files in the current directory (so it needs to know which it is)
- If you say `cat file` (cat is used to print text files to the terminal), the command needs to know the current directory in order to find the file to print (it could be anywhere, and there could be hundreds of files with the same name in hundreds of directories)
- This is one reason for why it is important to understand the file system

Home directory

- Every user has one, e.g. `/home/rikard /home/henrik`
- This is where you do all work (mostly), which is why you are dropped there when you start a shell (starting a terminal or logging in remotely)
- Using `cd` (change directory) without an argument takes you home
- Stored in the `$HOME` environment variable
- So commonly used that they invented the ~ alias for it

Summary

- The filesystem is an abstraction of a tree with / as the root
- Every directory and file (except /) is in exactly one directory, called “parent directory”
- This means that you can uniquely identify a file or directory using an absolute path (the address of the file starting from the root) or a relative path (starting with the current directory)
- Users have a home directory (with the nickname ~)
- In paths, you can use .. to signify parent directory and . to signify current directory
- When running the shell you “are in a directory” which is called current directory