# Filesystems & Denny's + LQ scraping

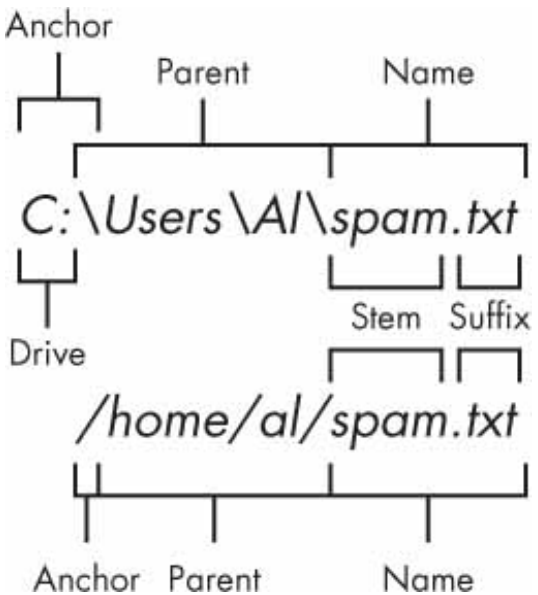
## Lecture 14

Dr. Colin Rundel

# Filesystems

Pretty much all commonly used operating systems make use of a hierarchically structured filesystem.

This paradigm consists of directories which can contain files and other directories (which can then contain other files and directories and so on).



# Absolute vs relative paths

Paths can either be absolute or relative, and the difference is very important. For portability reasons you should almost never use absolute paths.

## Absolute path examples

```
/var/ftp/pub
/etc/samba.smb.conf
/boot/grub/grub.conf
```

## Relative path examples

```
Sta323/filesystem/
data/access.log
filesystem/nelle/pizza.cfg
```

# Special directories

```
1 dir(path = "data/")
```

```
[1] "ak"                "gis"                "lego_sales.rds"
[4] "movies"            "office_ratings.csv" "phone.csv"
[7] "pvec_res.Rdata"    "us"
```

```
1 dir(path = "data/", all.files = TRUE)
```

```
 [1] "."                 ".."                 ".DS_Store"
 [4] "ak"                "gis"                "lego_sales.rds"
 [7] "movies"            "office_ratings.csv" "phone.csv"
[10] "pvec_res.Rdata"    "us"
```

```r
dir(path = "../")
```

```
[1] "css"    "slides"
```

```r
dir(path = "data/../../")
```

```
[1] "css"    "slides"
```

```r
dir(path = "../../")
```

```
 [1] "CLAUDE.md"     "config.yaml"   "data"          "docs"
 [5] "layouts"       "Makefile"      "README.md"     "resources"
 [9] "static"        "website.Rproj"
```

# Home directory and ~

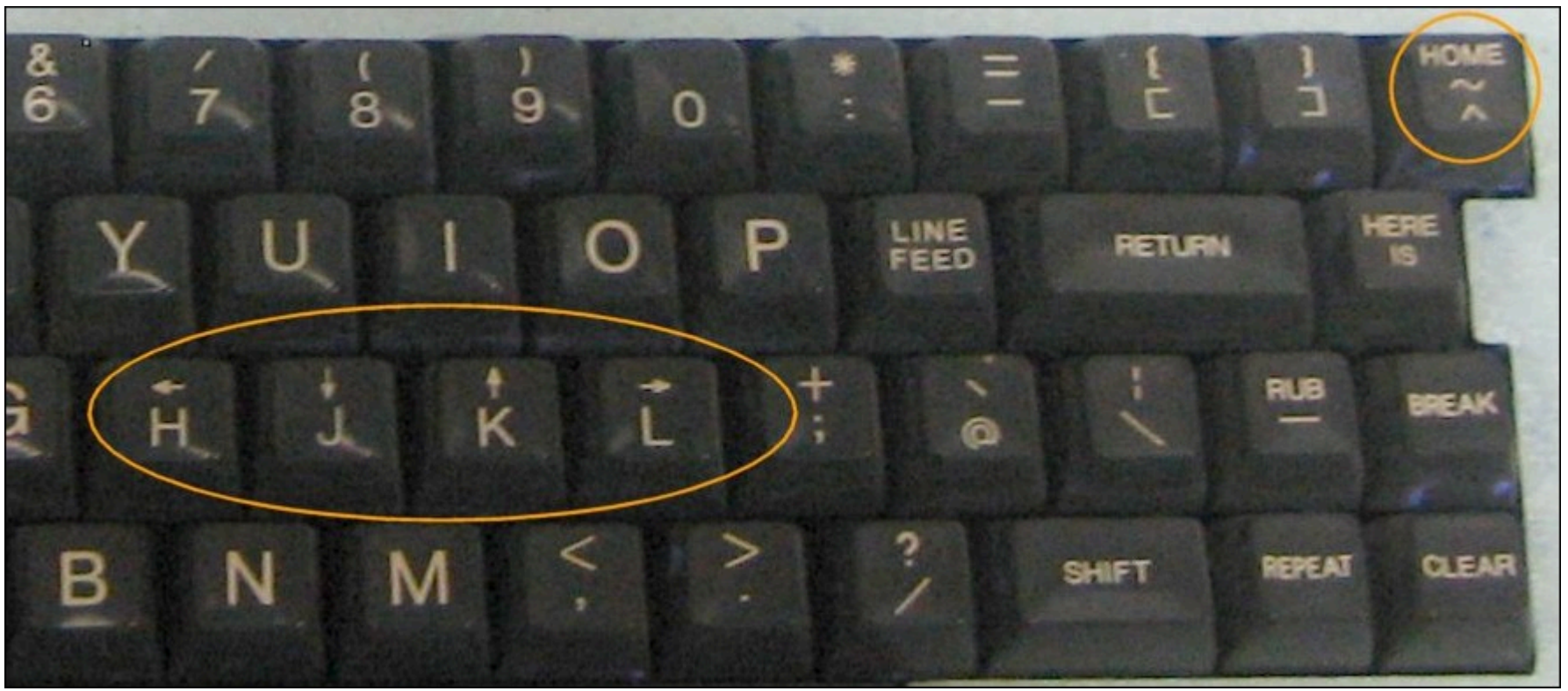
Tilde (~) is a shortcut that expands to the name of your home directory on unix-like systems.

```
1 dir(path = "~/Desktop/Sta323-Sp26/website/static/slides")
```

```
 [1] "data"                      "imgs"
 [3] "Lec01_notes.R"             "Lec01.html"
 [5] "Lec01.pdf"                 "Lec01.qmd"
 [7] "Lec02_notes.R"             "Lec02.html"
 [9] "Lec02.pdf"                 "Lec02.qmd"
[11] "Lec03.html"                "Lec03.pdf"
[13] "Lec03.qmd"                 "Lec04_notes.R"
[15] "Lec04.html"                "Lec04.pdf"
[17] "Lec04.qmd"                 "Lec05_notes.R"
[19] "Lec05.html"                "Lec05.pdf"
[21] "Lec05.qmd"                 "Lec06.html"
[23] "Lec06.pdf"                 "Lec06.qmd"
[25] "Lec07_cache"               "Lec07_files"
[27] "Lec07.html"                "Lec07.pdf"
[29] "Lec07.qmd"                 "Lec08_notes.R"
[31] "Lec08.html"                "Lec08.pdf"
[33] "Lec08.qmd"                 "Lec09_notes.R"
[35] "Lec09.html"                "Lec09.pdf"
[37] "Lec09.qmd"                 "Lec10_notes.R"
[39] "Lec10.html"                "Lec10.pdf"
[41] "Lec10.qmd"                 "Lec11_cache"
[43] "Lec11_files"               "Lec11.html"
[45] "Lec11.pdf"                 "Lec11.qmd"
```

# Why ~?

Below is the keyboard from an ADM-3A terminal from the 1970s,



# Working directories

R (and OSes) have the concept of a working directory, this is the directory where a program / script is being executed and determines the absolute path of any relative paths used.

```
1 getwd()
```

```
[1] "/Users/rundel/Desktop/Sta323-Sp26/website/static/slides"
```

```
1 setwd("~/")
2 getwd()
```

```
[1] "/Users/rundel"
```

If the first line of your R script is

```
setwd("C:\Users\jenny\path\that\only\I\have")
```

I* will come into your office and
SET YOUR COMPUTER ON FIRE 🔥.

* or maybe Timothée Poisot will

# RStudio and Working Directories

Just like R, RStudio also makes use of a working directory for each of your sessions - we haven't had to discuss these yet because when you use an RStudio project, the working directory is automatically set to the directory containing the `Rproj` file.

This makes your project portable as all you need to do is to send the project folder to a collaborator (or push to GitHub) and they can open the project file and have identical *relative* path structure.

# Some useful base R filesystem functions

- `dir()` - list the contents of a directory
- `basename()` - Removes all of the path up to and include the last path separator (`/`)
- `dirname()` - Returns the path up to but excluding the last path separator
- `file.path()` - a useful alternative to `paste0()` when combining paths (and urls) as it will add a `/` when necessary.
- `unlink()` - delete files and or directories
- `dir.create()` - create directories
- `fs` package - collection of filesystem related tools based on unix cli tools (e.g. `ls`)


fs

# The fs package

The fs package provides a cross-platform, uniform interface to file system operations. It wraps system calls with consistent naming and behavior across operating systems.

```
1 library(fs)
```

Key features:

- All functions are vectorized
- Provides consistent & helpful error messages on failure
- Handles paths consistently across Windows, Mac, and Linux
- Consistent naming scheme via function prefixes (`file_`, `dir_`, `path_`)

# Path manipulation - `path()`

The `path()` function constructs file paths "correctly" for the current OS:

```
1 path("data", "myfile.csv")
```

```
data/myfile.csv
```

```
1 path("~", "Desktop/", "/Sta323-Fa25")
```

```
~/Desktop/Sta323-Fa25
```

Other useful path functions:

- `path_abs()` - convert to absolute path
- `path_expand()` - expand `~` in paths
- `path_file()` - extract filename (like `basename()`)
- `path_dir()` - extract directory (like `dirname()`)
- `path_ext()` - extract file extension
- `path_ext_set()` - change file extension

# Listing files and directories

`dir_ls()` lists directory contents with more information than `dir()`:

```
1 dir_ls("data/")
```

```
data/ak                 data/gis                data/lego_sales.rds
data/movies             data/office_ratings.csv data/phone.csv
data/pvec_res.Rdata     data/us
```

file lists can also be filtered via glob, regex, or type:,

```
1 dir_ls("data/", glob = "*.csv")
```

```
data/office_ratings.csv data/phone.csv
```

```
1 dir_ls("data/", type = "file", recurse = TRUE)
```

```
data/ak/states.dbf
data/ak/states.prj
data/ak/states.shp
data/ak/states.shx
data/gis/AnneArundel/AnneArundelN.dbf
data/gis/AnneArundel/AnneArundelN.prj
data/gis/AnneArundel/AnneArundelN.qpj
data/gis/AnneArundel/AnneArundelN.shp
data/gis/AnneArundel/AnneArundelN.shx
data/gis/AnneArundel/AnneArundelN84.dbf
data/gis/AnneArundel/AnneArundelN84.prj
data/gis/AnneArundel/AnneArundelN84.qpj
data/gis/AnneArundel/AnneArundelN84.shp
data/gis/AnneArundel/AnneArundelN84.shx
data/gis/airports/airports.dbf
data/gis/airports/airports.prj
data/gis/airports/airports.sbn
data/gis/airports/airports.sbx
```

```
data/gis/airports/airports.shp
data/gis/airports/airports.shx
data/gis/airports/airports.txt
data/gis/airports/airports.xml
data/gis/nc_counties/nc_counties.dbf
data/gis/nc_counties/nc_counties.prj
data/gis/nc_counties/nc_counties.shp
data/gis/nc_counties/nc_counties.shx
data/gis/nc_districts112.gpkg
data/gis/us_interstates/us_interstates.avl
data/gis/us_interstates/us_interstates.dbf
```

# Creating and deleting

Creating directories:

```
1 dir_create("data/temp")
2 dir_create("data/temp/a/b/c")
```

Deleting files and directories:

```
1 file_delete("data/myfile.csv")
2 dir_delete("data/temp")
```

# Copying and moving

Copy files or directories:

```
1 file_copy("data/file.csv", "backup/file.csv")
2 dir_copy("data/", "backup/")
```

Move (rename) files or directories:

```
1 file_move("data/old.csv", "data/new.csv")
```

Check if files/directories exist:

```
1 file_exists("data/myfile.csv")
2 dir_exists("data/temp")
```

# File metadata

## Get information about files:

```
1 file_info("data/office_ratings.csv")
```

```
# A tibble: 1 × 18
  path         type   size permissions modification_time   user  group device_id
  <fs::path>   <fct> <fs:> <fs::perms> <dttm>              <chr> <chr>     <dbl>
1 …ratings.csv file  7.02K rw-r--r--   2022-08-29 10:47:54 rund… staff  16777229
# ℹ 10 more variables: hard_links <dbl>, special_device_id <dbl>, inode <dbl>,
#   block_size <dbl>, blocks <dbl>, flags <int>, generation <dbl>,
#   access_time <dttm>, change_time <dttm>, birth_time <dttm>
```

## We can use vectorization to get info on multiple files,

```
1 dir_ls("data/") |>
2   file_info()
```

```
# A tibble: 8 × 18
  path       type     size permissions modification_time   user  group device_id
  <fs::path> <fct> <fs::b> <fs::perms> <dttm>              <chr> <chr>     <dbl>
1 data/ak    dire…     192 rwxr-xr-x   2023-10-11 22:08:38 rund… staff  16777229
2 data/gis   dire…     256 rwxr-xr-x   2024-11-13 09:58:14 rund… staff  16777229
3 …sales.rds file   23.07K rw-r--r--   2022-08-29 10:47:54 rund… staff  16777229
4 …ta/movies dire…     128 rwxr-xr-x   2023-10-11 22:08:38 rund… staff  16777229
5 …tings.csv file    7.02K rw-r--r--   2022-08-29 10:47:54 rund… staff  16777229
6 …phone.csv file       99 rw-r--r--   2022-08-29 10:47:54 rund… staff  16777229
7 …res.Rdata file      369 rw-r--r--   2022-08-29 10:47:54 rund… staff  16777229
8 data/us    dire…     320 rwxr-xr-x   2023-10-11 22:08:38 rund… staff  16777229
```

```
# ℹ 10 more variables: hard_links <dbl>, special_device_id <dbl>, inode <dbl>,
#   block_size <dbl>, blocks <dbl>, flags <int>, generation <dbl>,
#   access_time <dttm>, change_time <dttm>, birth_time <dttm>
```

# Denny's and LQ Scraping Demo