File System Tree

STAT 133

Gaston Sanchez

Department of Statistics, UC–Berkeley

gastonsanchez.com
github.com/gastonstat
Course web: gastonsanchez.com/stat133

Managing Files

File Management

File management is crucial for any data analysis project
Common types of files:

- Data files
- Code files (e.g. functions)
- Analysis files
- Presentation and Report files

Also, many tools such as R, LaTeX, markdown, etc require knowing where files are located in your computer

File Management

Good file managment allows you to:

- find things more easily
- make changes more easily
- benefit from work you've already done
- be understood by others
- collaborate with others

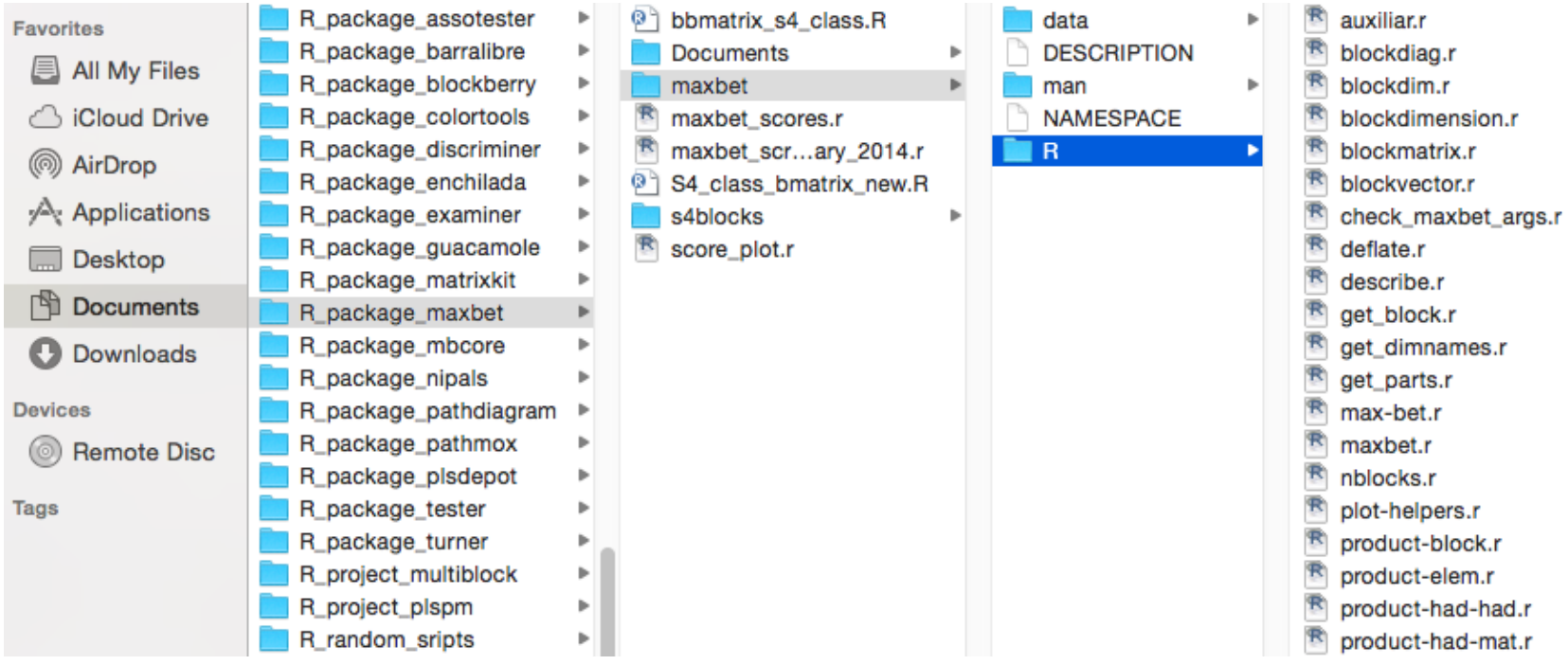
Why File Management?

Common tasks

- Access and organize your files
- Control creation of files
- Control deletion of files
- Control relocation of files
- Control modification of files

How does our computer organize files?

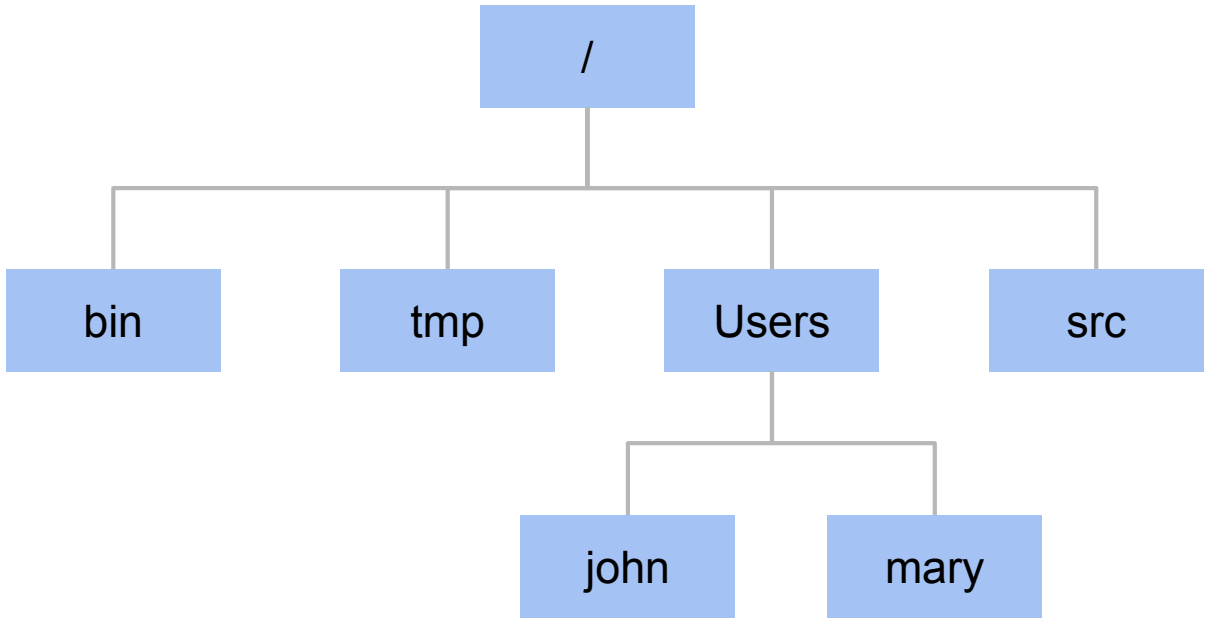
Files and Directories



Organization of Files

- The computer organizes files within directories
- Directories and files follow a tree structure
- A tree structure is a hierarchical structure
- Hierarchical means that directories are located inside other directories

Files and Directories



Filesystem

- The nested hierarchy of folders and files on your computer is called the **filesystem**
- The filesystem follows a tree-like structure

Directories

Files and Directories

There are two special directories in UNIX-like OS:

- The top level directory, named "/", called the **root directory**
- The **home directory**, named ~, which contains all your files

Root Directories

- A root directory is the first level in a disk (such as a hard drive)
- It is the root out of which the file tree "grows"
- All other directories are subdirectories of the root directory
- On Unix-like system, including Macs, the root directory is denoted by a forward slash: /

Root Directory

- The root directory is the most includive folder on the system
- The root directory serves as the container of all other files and folders
- A Unix-based system (e.g. OS X) has a single root directory
- Windows users usually have multiple roots (`C:`, `D:`, etc)

Root Directories on Windows

- On Windows computers you can have multiple root directories (one for each storage device)
- On Windows, root directories are given a drive letter assignment
- On Windows, the most common root directory is `C:\` (denoting the C partition of the hard drive)

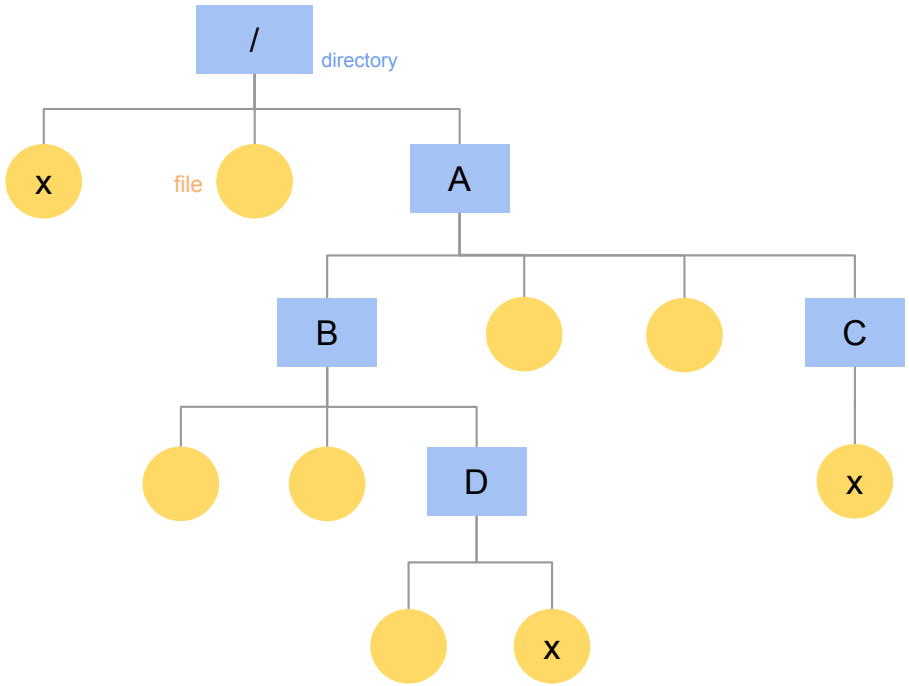
Home Directory

- User's personal files are found in the `/Users` directory
- e.g. mine is `/Users/Gaston`
- A user directory is the **home** directory

Subdirectories and Parent Directories

- We store files in subdirectories of the root directory
- Inside these subdirectories may be further subdirectories and so on
- A directory containing other directories is referred to as the **parent directory**
- Directories inside other directories are referred to as **child directories**

Directories and Subdirectories

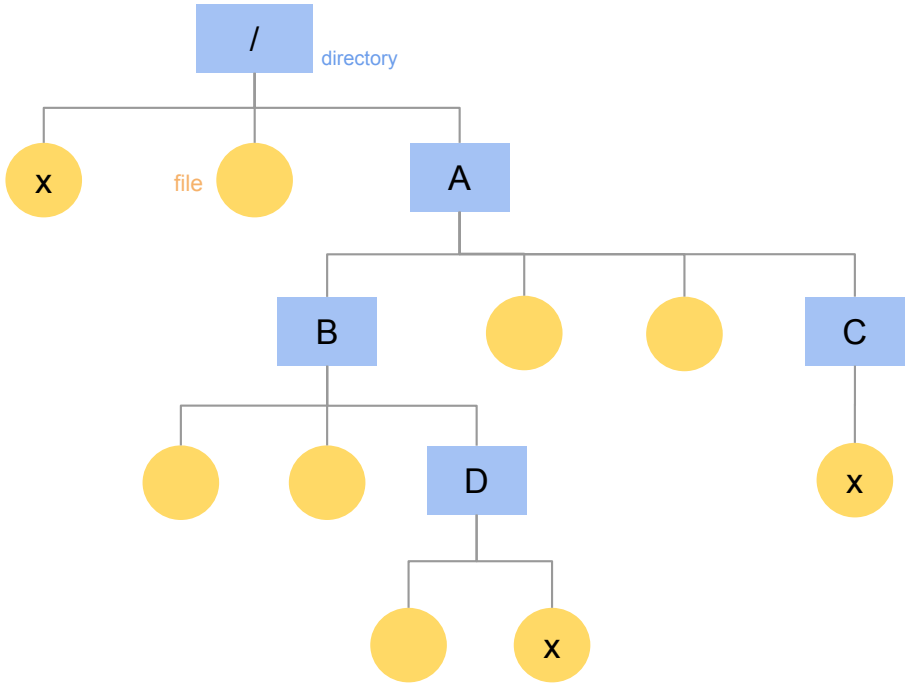


- `A` is a child directory of the root directory
- `A` is the parent directory of `B` and `C`

Working Directory

- Another special type of directory is the so-called **working directory**
- The working directory is the current directory where you perform any task
- If you go to your Desktop, then the `Desktop` is your current directory
- When you use R, the working directory is the directory where the program automatically looks for files

Working Directory



If you are standing in B, then this is your working directory

Paths

Path

- Each file and directory has a unique name in the filesystem
- Such unique name is called a **path**
- The path is simply the desription of where something is located in the filesystem

Filesystem

- The path is a list of directory names separated by slashes
- If the path is for a file, then the last element of the path is the file's name
- e.g. `/Users/Gaston/Documents/data.txt`
- A path can be **absolute** or **relative**

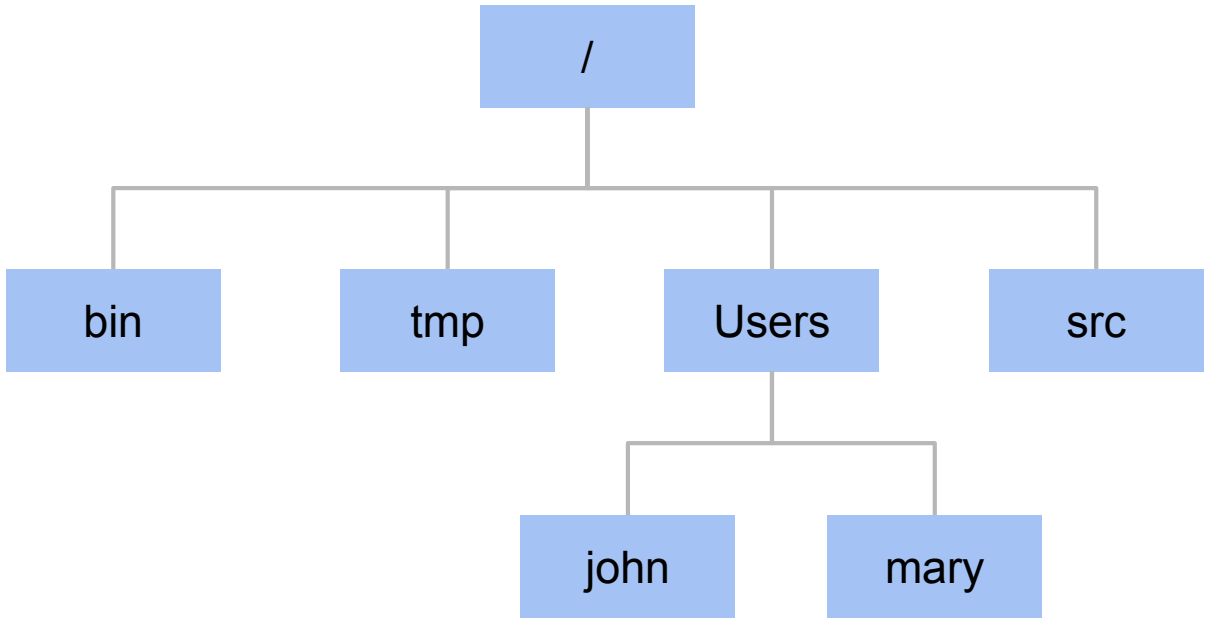
Paths

- An **absolute path** is a complete and unambiguous description of where something is in relation to the **root**
- If a path begins with a slash (i.e. the root), then it's called an absolute path
- A **relative** describes where a folder or file is in relation to another folder (typically the working directory)
- If a path does not begin with a slash, then it is a relative path

Paths

- There are two special relative paths: `.` and `..`
- The single period `.` refers to the current directory
- The two periods means the parent directory, one level above
- For instance, if the current directory is `/Users/XYZ/abc`, then `.` refers to this directory, and `..` refers to `/Users/XYZ`

Files and Directories

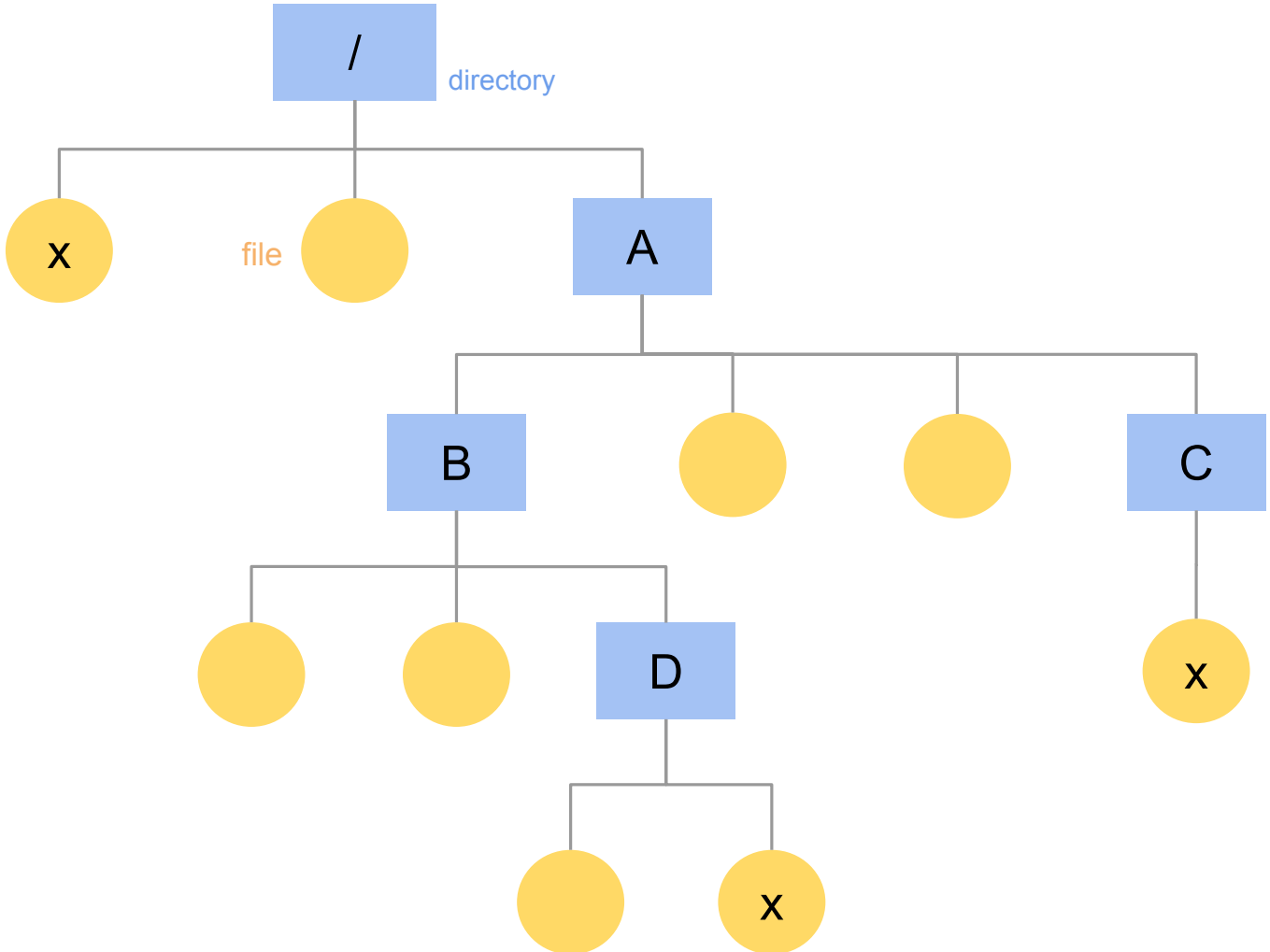


Path Names

Full path name

- path from the top level directory, `/`, to the file or directory of interest
- For `mary` the full pathname is: `/Users/mary`

Files and Directories

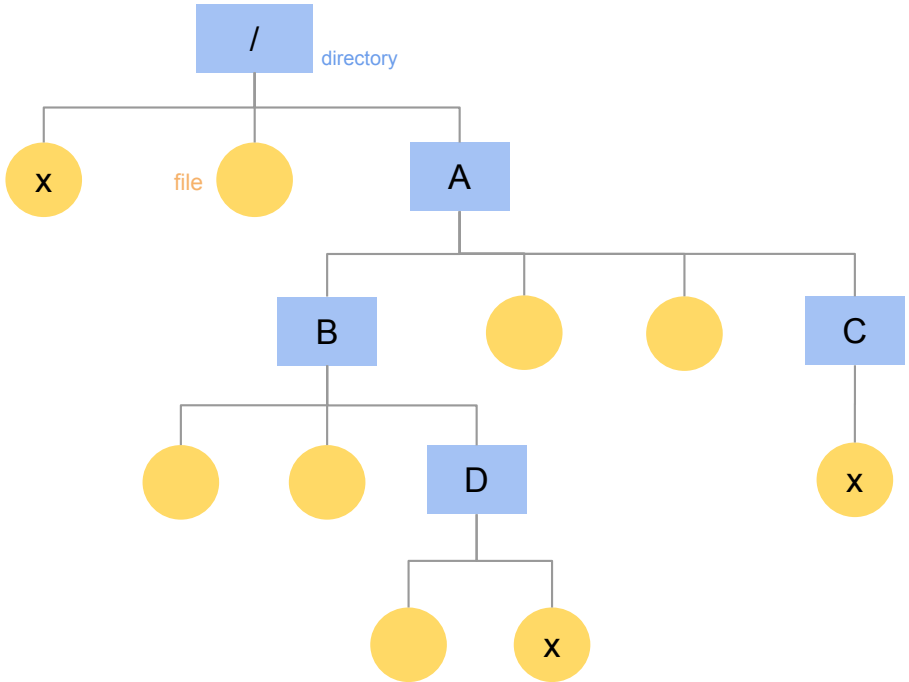


Path Names

Relative path name

- path from the current directory to the file or directory of interest
- Relative path to `D` from `A`: `B/D`
- Equivalently: `./B/D` (`.` refers to current directory)

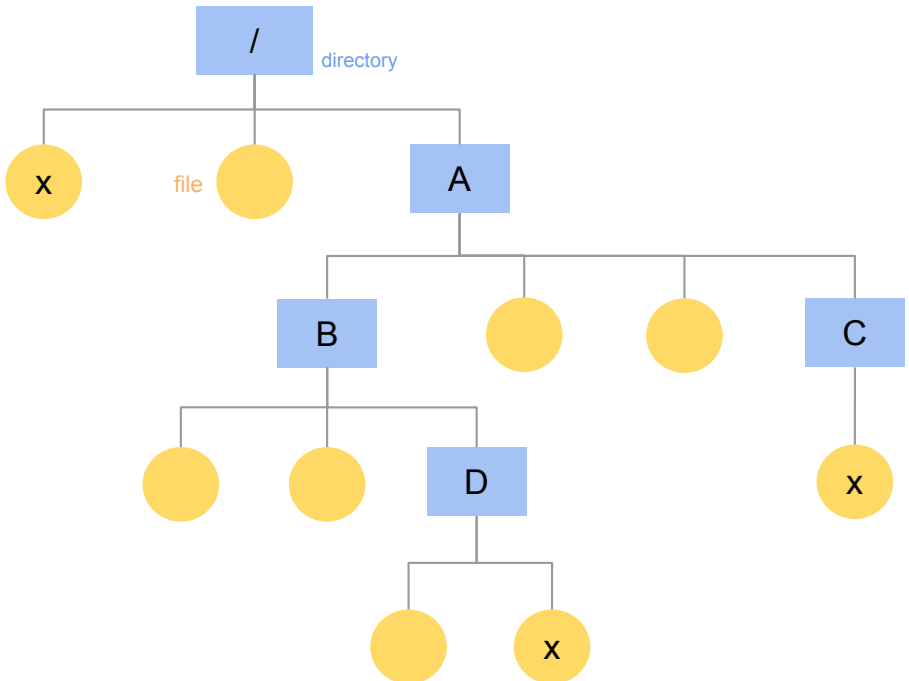
Relative Path Names



Relative path to `D` from `A`: `B/D`

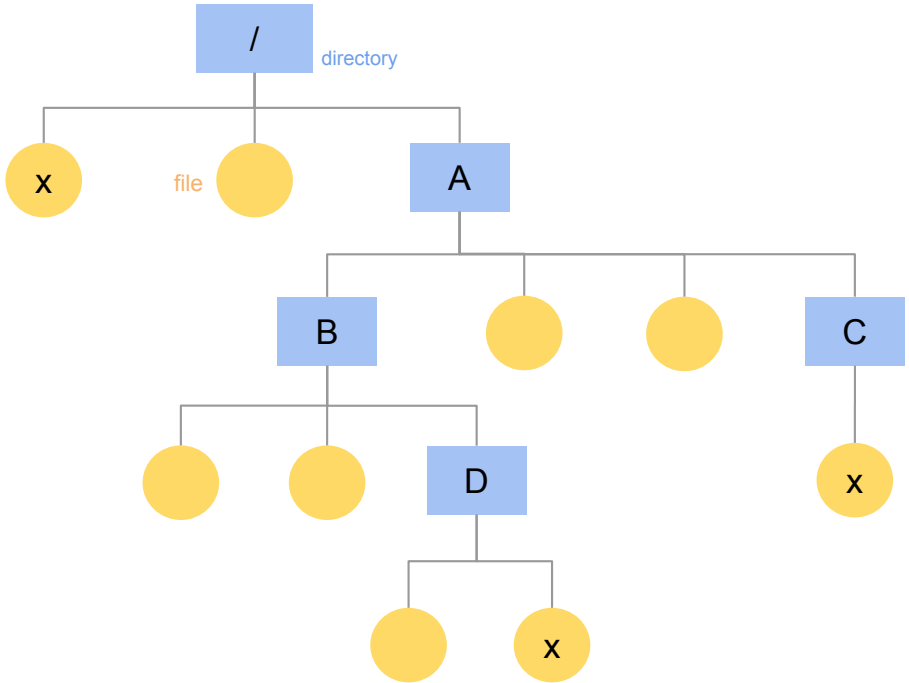
Equivalently: `./B/D` (`.` refers to current directory)

Relative Path Names



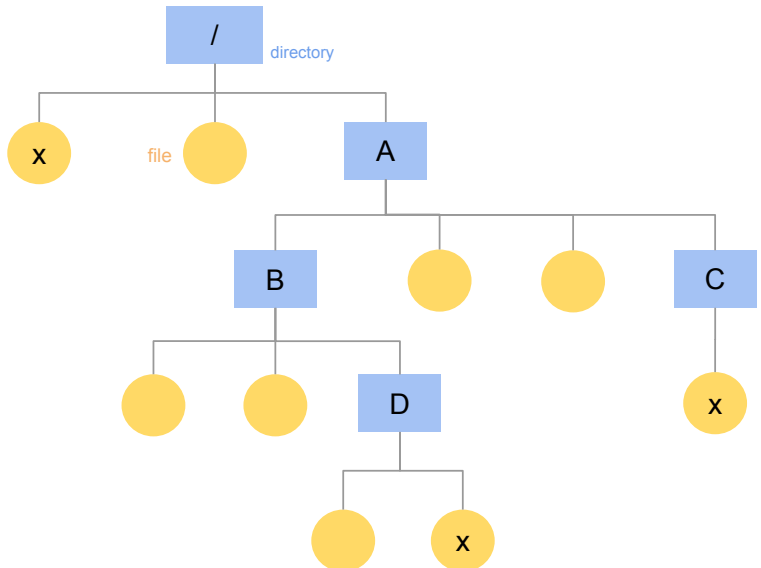
Relative path to `D` from `C`: `../B/D`
(.. refers to parent directory)

Relative Path Names



Relative path to `x` at the top from within `C`?

Relative Path Names



Relative path to `x` at the top from within `C`?

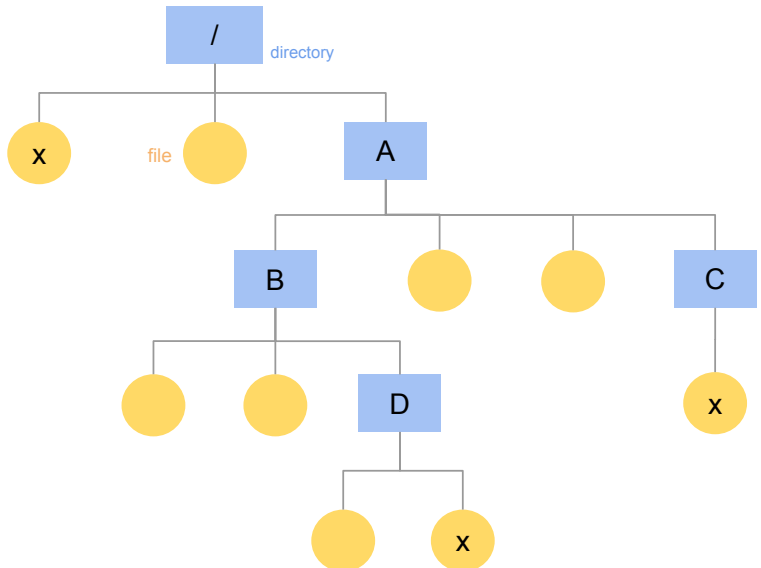
a) `../A/x`

b) `../../x`

c) `../x`

d) `/x`

Relative Path Names



Relative path to `x` in `D` from within `C`?

a) `../D/x`

b) `../B/D/x`

c) `../../A/B/D/x`

d) `/A/B/D/x`

Filesystem

- Root Directory
- Home Directory
- Working Directory
- Directory Tree
- Absolute path names
- Relative path names

File Manipulation Commands in R

R File Management Functions

function	description
`getwd()`	shows the current working directory
`list.files()`	see all the files and subdirectories in the current working directory
`setwd()`	sets the current working directory
`dir.create()`	create a new directory
`file.create()`	create a new blank file
`cat()`	create a new file and put text into it, or append text to an existing file
`file.append()`	attempts to append two files
`unlink()`	delete files and directories
`file.rename()`	rename a file or move a file
`file.copy()`	copy a file to another directory
`file.exists()`	check whether a file exists

getwd()

`getwd()` allows you to find your current working directory

```
# working directory (for these slides)
getwd()


## [1] "/Users/gaston/Dropbox/course_stat133/stat133/slides/27-file-system"
```

list.files()

`list.files()` displays the files and subdirectories of the working directory

```
# files and directories in my working directory
wd <- list.files()
head(wd)
```

```
## [1] "27-file-system-concordance.tex" "27-file-system.log"
## [3] "27-file-system.nav"             "27-file-system.pdf"
## [5] "27-file-system.Rnw"             "27-file-system.snm"
```

list.files()

You can also specify a different `path`

```
# contents in the stat133 github repo
list.files(path = '~/Documents/stat133/stat133')

## character(0)
```

setwd()

`setwd()` allows you to set a working directory (this is where R will look for files and subdirectories)

```
# setting a working directory
setwd('~/Documents/Consulting')
```

setwd()

Assuming that there is a subdirectory `Data` inside `Consulting`, we could read a file like so:

```
# setting a working directory
df <- read.csv('Data/dataset.csv')
```

dir.create()

`dir.create()` allows you to create a new directory

```
# new directory
dir.create('/Users/john/Documents/stat133/HW6')

# new directory (Windows)
dir.create('C:\\Documents\\stat133\\HW6')
```

file.create()

`file.create()` allows you to create a new blank file

```
# new file 'functions.R'
file.create('/Users/john/Documents/stat133/HW6/functions.R')

# new file (on Windows)
file.create('C:\\Documents\\stat133\\HW6\\functions.R')
```

cat()

`cat()` can be used to create a new file and put text into it

```
# new file 'functions.R'
cat('# Homework 6',
    '\n# Your name',
    '\n# Description',
    file = '/Users/john/Documents/stat133/HW6/myscript.R')
```

file.append()

`file.append()` attempts to append two files

```
# append two files
file.append('data1.csv', 'data2.csv')
```

unlink()

`unlink()` deletes files and directories (warning: deletion is permanently)

```
# delete a file
unlink('/Users/john/Documents/stat133/HW6/myscript.R')
```

file.rename()

`file.rename()` renames a file

```
# rename a file
file.rename(from = 'script.R', to = 'analysis.R')
```

file.rename()

`file.rename()` can also be used to fully move a file form one directory to another

```
# move a file
file.rename(from = 'old-project/analysis.R',
            to = 'new-project/analysis.R')
```

file.copy()

`file.copy()` copies a file to another directory

```
# move a file
file.copy(from = 'old-project/analysis.R',
          to = 'new-project/analysis.R')
```

file.exists()

`file.exists()` checks whether a file exists

```
# checking existance of a file
file.exists('homework05_instructions.pdf')
```

Related functions

- `file.info()`
- `file.mode()`
- `file.mtime()`
- `file.size()`
- `file.access()`
- `system()`