

Introduction (recap)

Ádám T. Kocsis (adam.kocsis@fau.de)



Schedule

	Monday	Tuesday	Wednesday	Thursday	Friday
Instructor	Adam	Adam and Wolfgang	Chris and Sebastian	Adam	Adam
Where	Henkestr.	Henkestr.	Tennenlohe	Henkestr.	Henkestr.
Morning	10:00-12:00	9:00-12:00	9:00 – 16:00	8:30-9:30 10:00 – 11:00 Msc Welcome event @ GeoZentrum!	8:30-11:30
Afternoon	13:00-16:00	13:30-16:00 (Wolfgang)		12:30-16:00	12:30-15:00

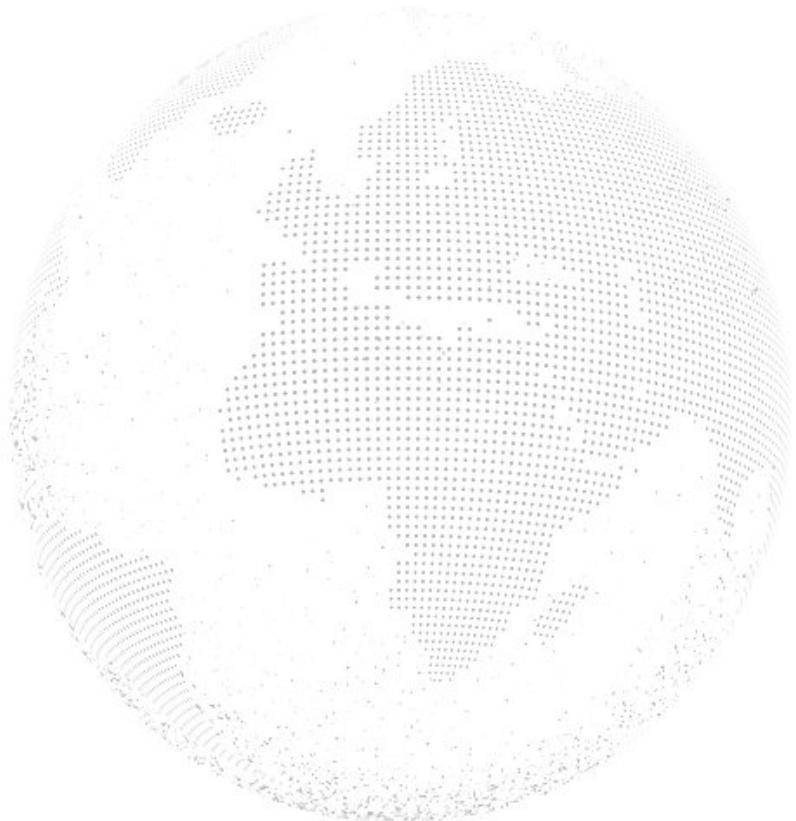
<https://tinyurl.com/4nfsuehb>

Objectives

1. Confidence with **files, directories and paths**
2. Experience with **raster** and **vector graphics**
3. Essential of the **BASH** Shell/Language
4. Better understand **programming** (FOSS)
5. Execute, understand, and modify **R scripts**
6. Write **basic R code**, see the capabilities

Discussions

1. What do we/you use computers for?



Discussions

1. What do we/you use computers for?
2. In (geo)sciences what do you use computers for? What kind of software?



Discussions

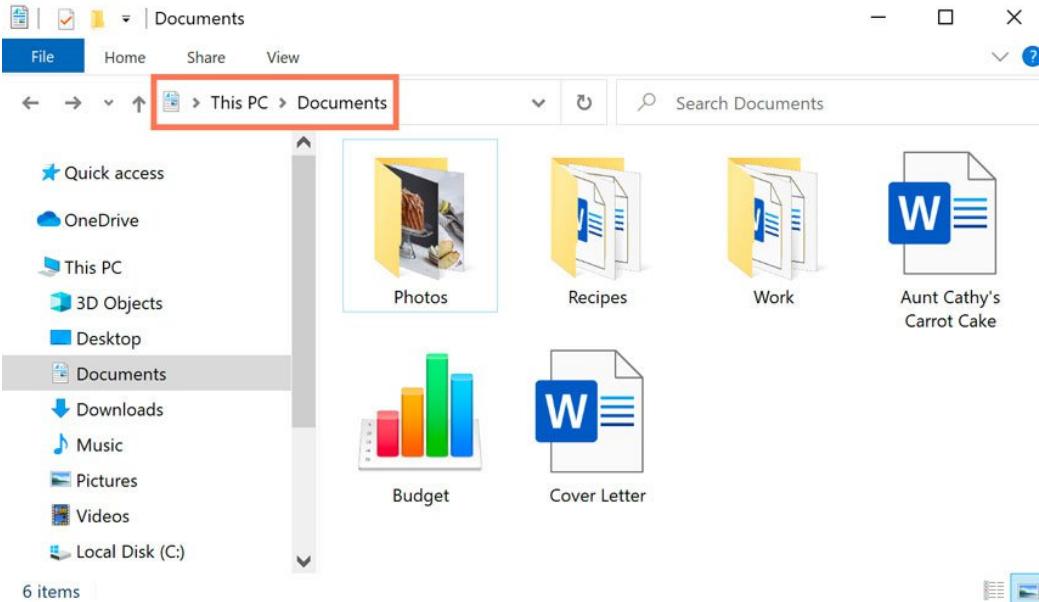
1. What do we/you use computers for?
2. In (geo)sciences what do you use computers for? What kind of software?

Presentation
Publishing, Typesetting
Multimedia consumption
Internet Browsing
Email, Communication
Graphics Vector, Raster, 3D
Software Development
Data Processing/Analysis
Statistical Analysis and Modelling
Computation
GIS
Web Design
File management



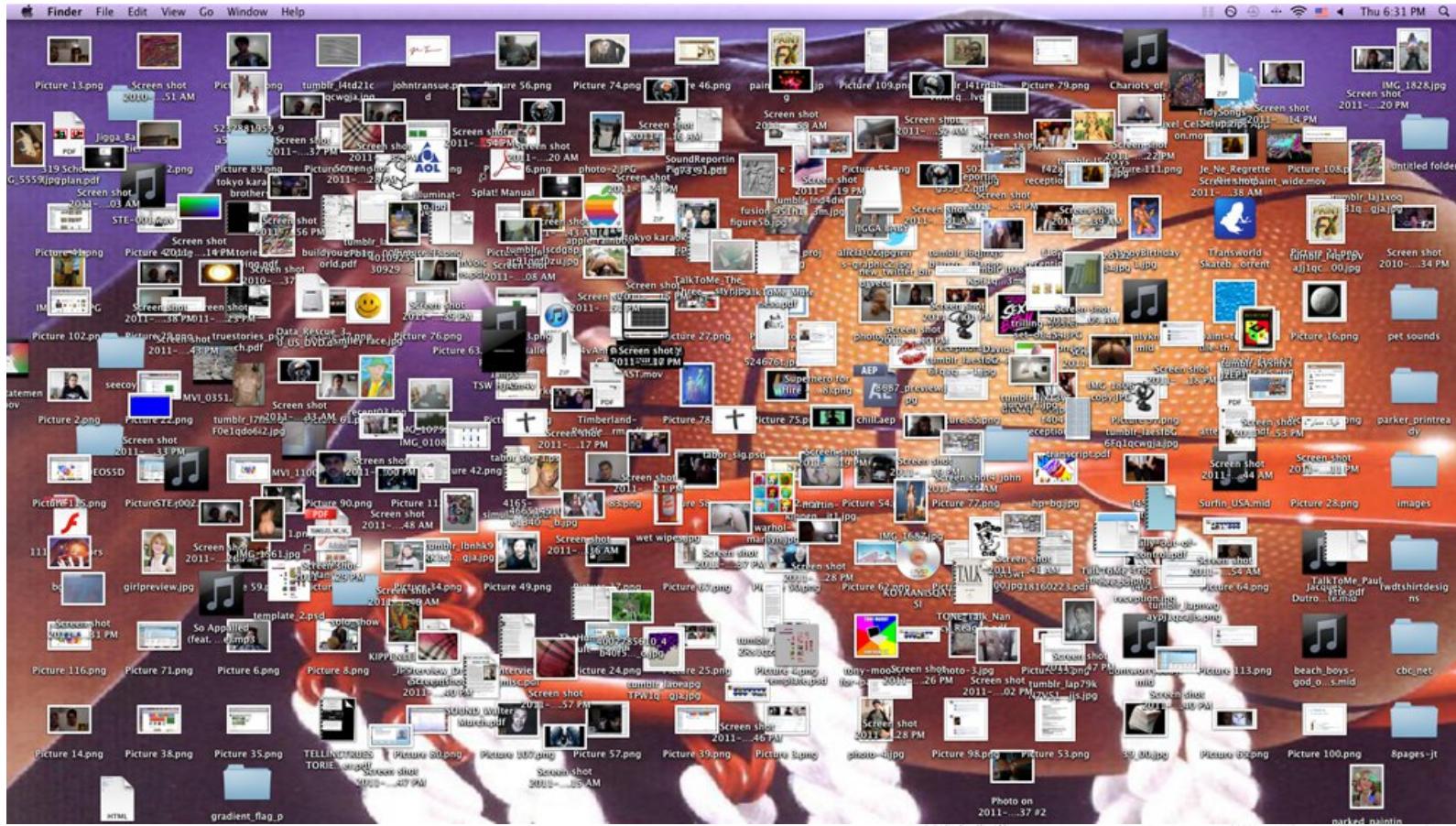
Files and directories

The practical point of view



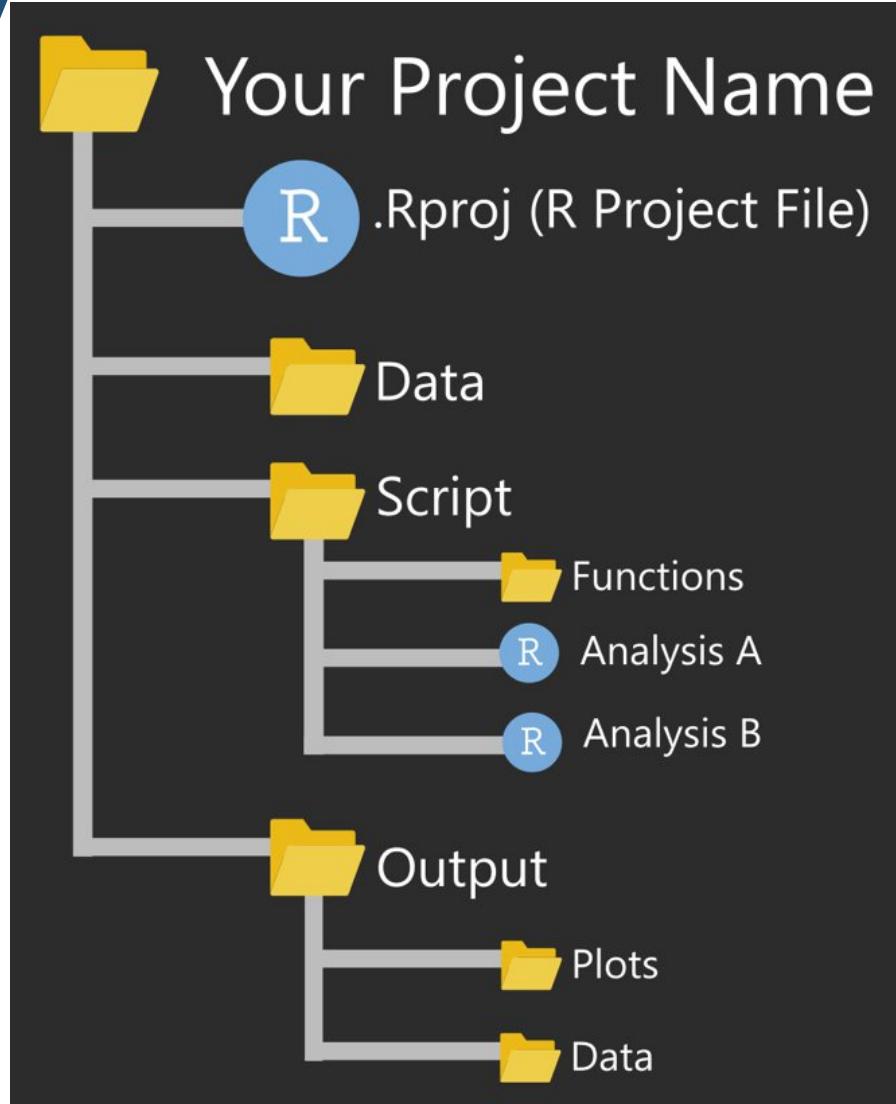
(Almost) everything that you work on is in a file.

Avoid this!



Solution: hierarchy

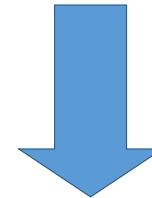
- **Regular file:** cannot contain other files
- **Directory:** special file, does not store anything but **other files (can be empty)**
- Directories **do not actually contain data**, this is just an abstract representation, just references to other files
- Copying vs Moving (renaming) speed difference!



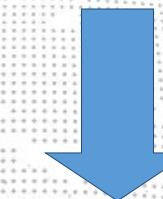
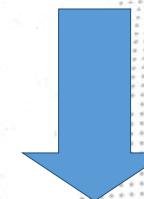
Exercise

- Create a directory that represents this semester of your studies!
- Make directories for every class that you will have this semester!
(including this)

Program/degree

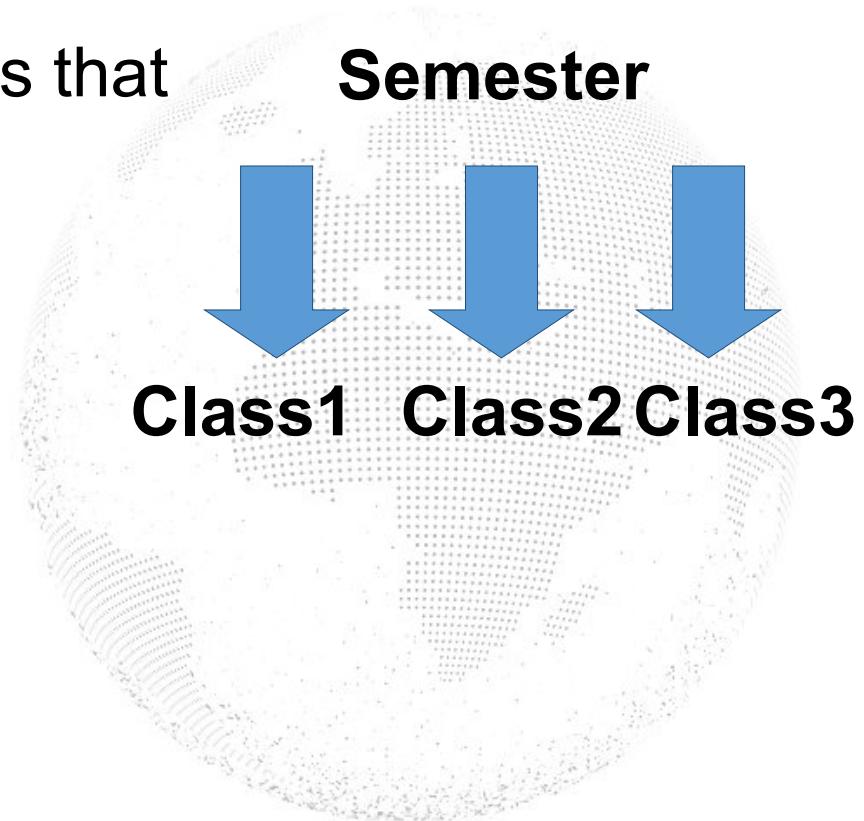


Semester



Class1

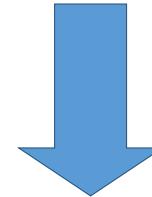
Class2 Class3



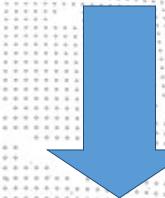
Exercise

- Create a directory that represents this semester of your studies!
- Make directories for every class that you will have this semester!
(including this)

Program/degree



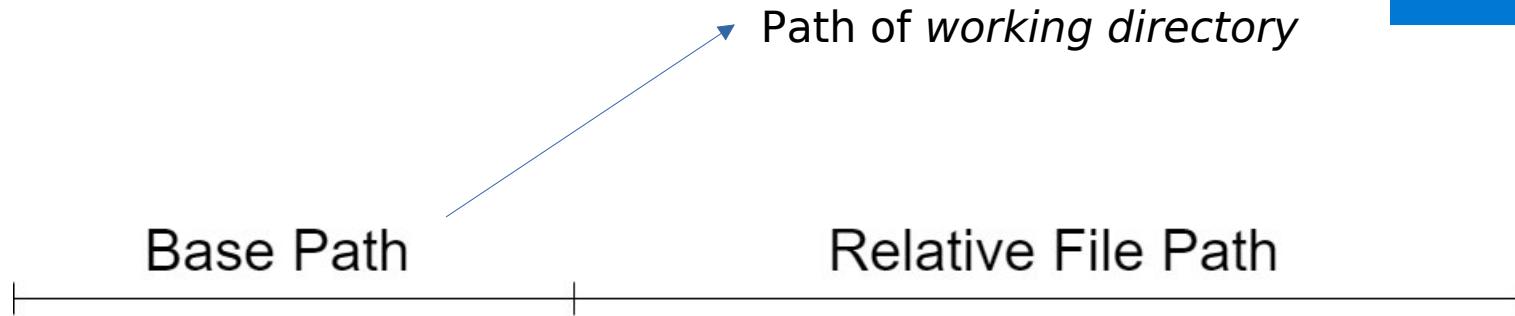
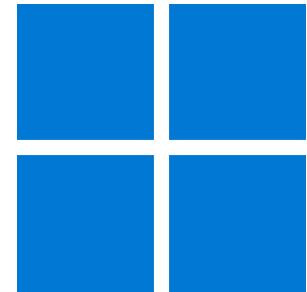
Semester



Class1 Class2 Class3

Where did you create this?

The file's path



C:\Source\project\docs\file.txt

Root

Base

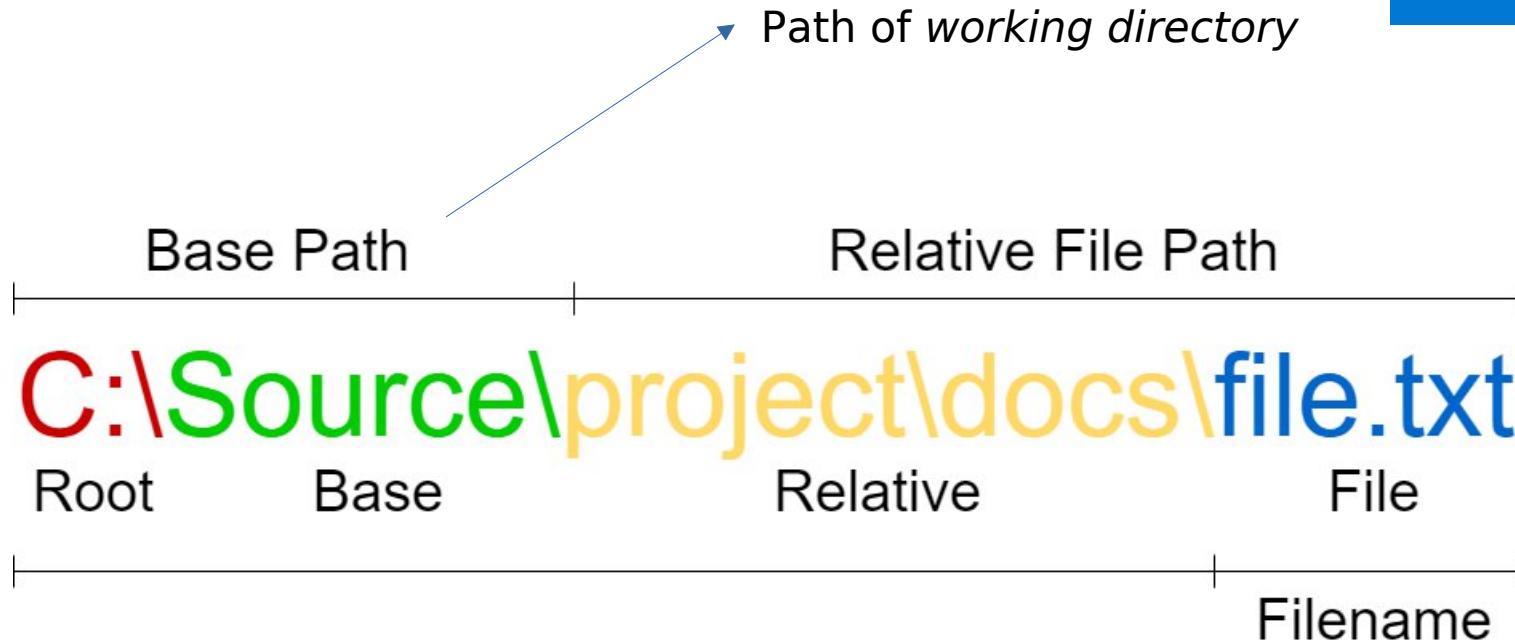
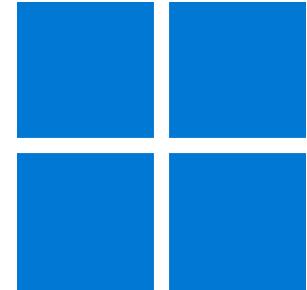
Relative

File

Filename

Absolute path

The file's path



Note: Things are easier if this does not have any spaces!

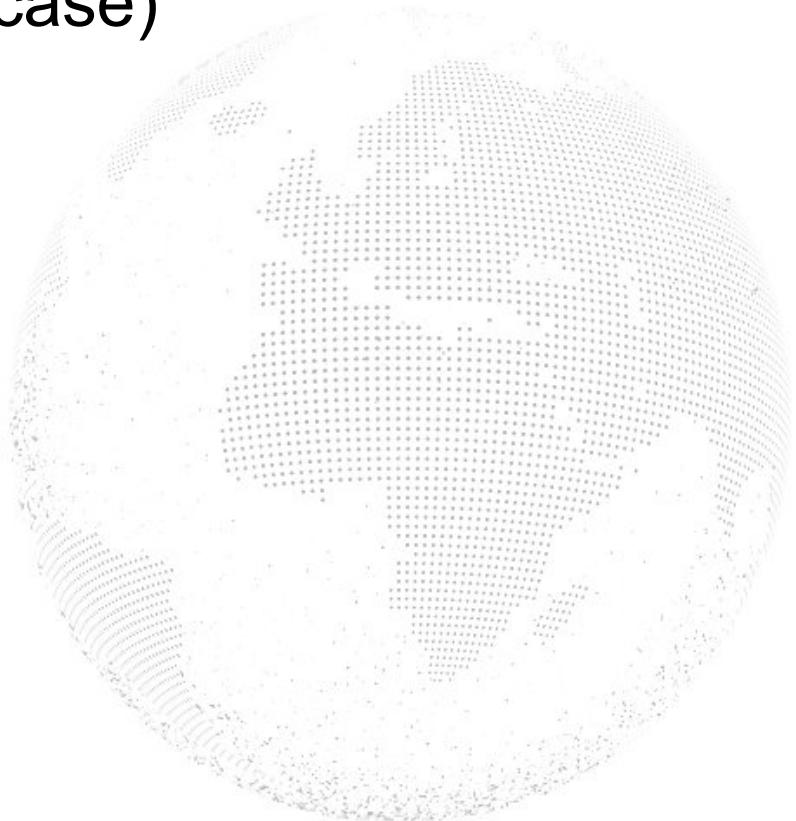
No Spaces?

- Systematic rules to represent text with one word
- Variable/Object names - Language dependent conventions

Naming Convention	Example Format
Pascal Case	PascalCase
Camel Case	camelCase
Snake Case	snake_case
Kebab Case	kebab-case
Flat Case	flatcase
Upper Flat Case	UPPERFLATCASE
Pascal Snake Case	Pascal_Snake_Case
Camel Snake Case	camel_Snake_Case
Screaming Snake Case	SCREAMING_SNAKE_CASE
Train Case	Train-Case
Cobol Case	COBOL-CASE

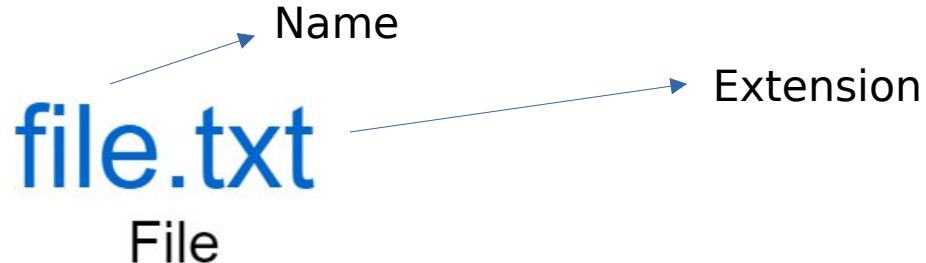
Exercise

Rename your class directories to match one of your preferred cases! (e.g. snake case)



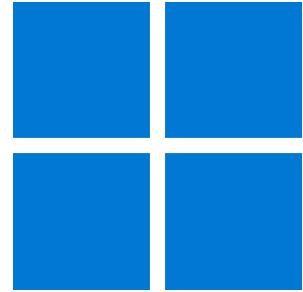
Files don't need to have an extension

The file's extension

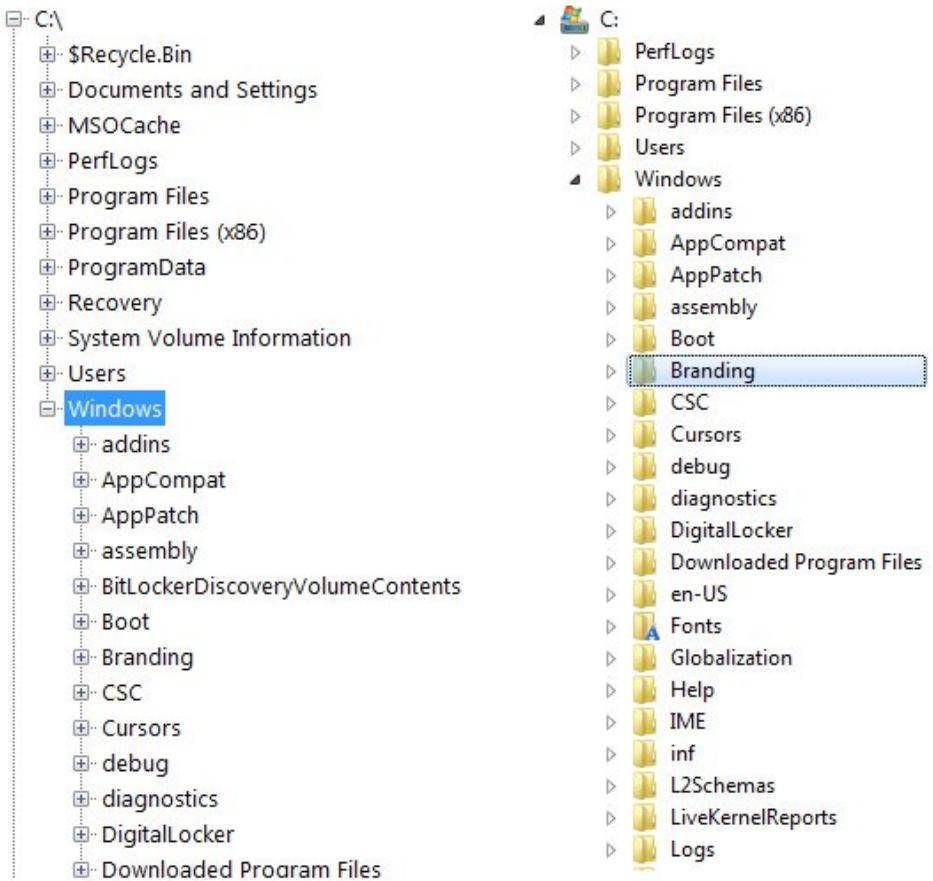


- a) The file extension indicates to the operating system (and you!) how to handle the file.
- b) This is not a hard constraint! Changing the extension will not make the file's contents different in any way!

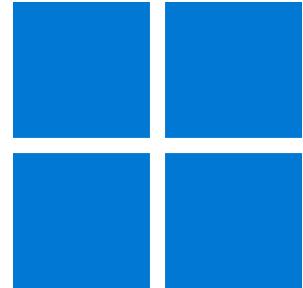
Windows - files



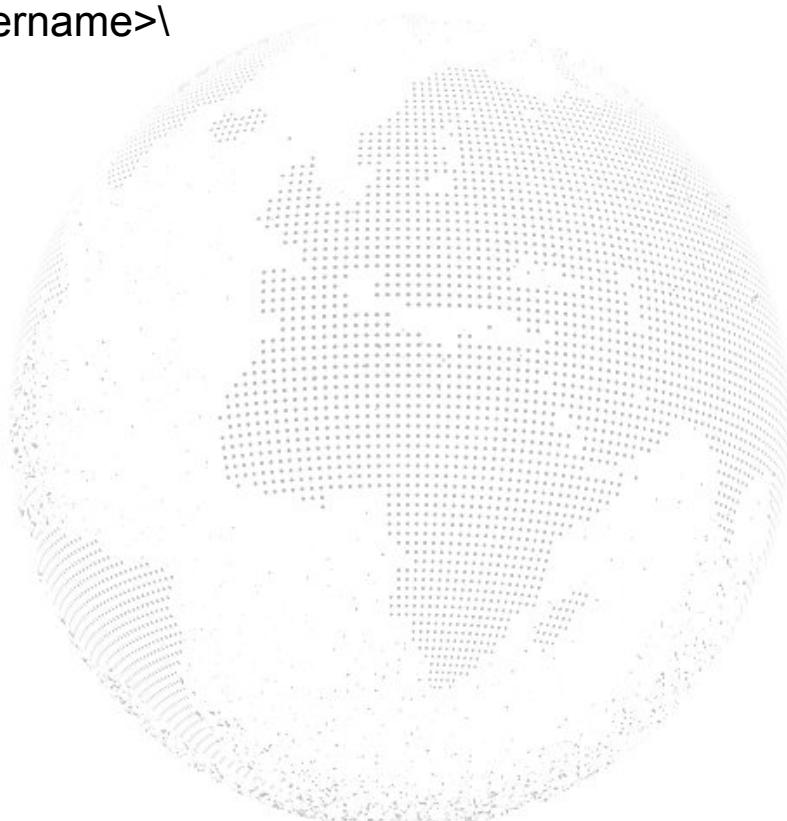
- Files are data items on storage devices
- Multiple roots e.g. C:, each correspond to a **partition**
- Paths use the characteristic **backslash ** character to depict nestedness
- Directories are called “**Folders**”
- Executables: filename.exe
- Total path to “Branding”: C:\Windows\Branding
- Case insensitive!
- FAT32 and NTFS



Windows - files



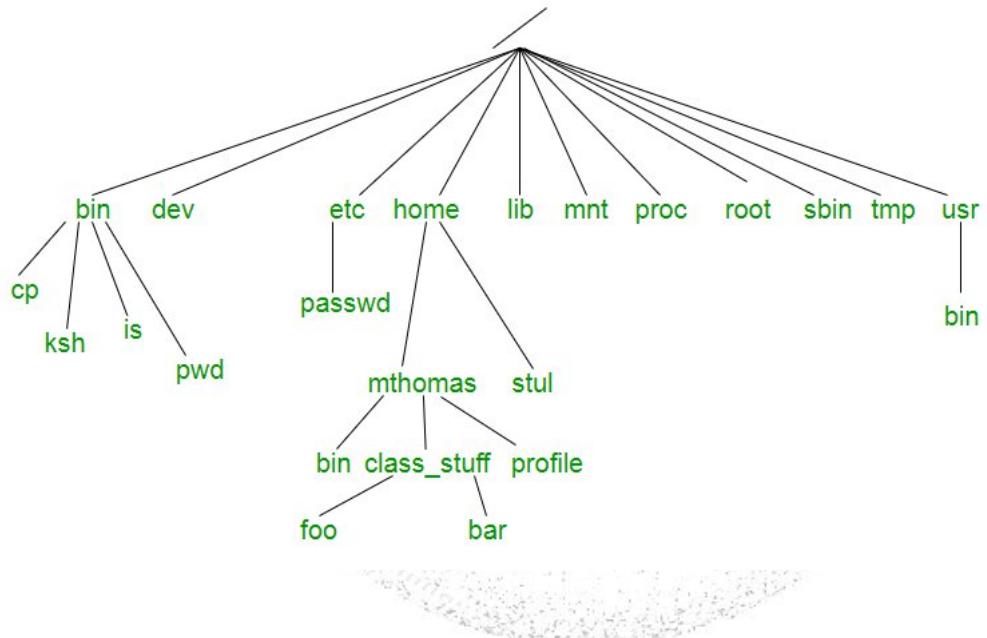
- Paths Always present, if you don't see it
- To make the directory hierarchy novice-friendly, Windows creates “aliases” that look nice, but are not functional!
- User’s home directory is by default: C:\Users\<Username>\
- Desktop: C:\Users\<Username>\Desktop
- The program to view files is “Explorer.exe”



UNIX File system



- Shared for UNIX and UNIX-like systems (GNU/Linux, macOS, Android)
- More abstract: everything in the computer is represented by a file
- ~ Standard directory names
- Nestedness coded with forward slash : /
- File can be anything
- Executables don't have extensions
- Complete path to "bar"
/home/mthomas/class_stuff/bar
- Case sensitive!



Archives

Excellent for storing and transmitting files – entire directory structures

Two processes:

- creating an **archive**: one file from multiple files
- employing **lossless compression**: algorithm to make decrease the size of a file

Examples: zip, rar, gzip, bzip2, tgz (e.g. .tar.gz)



Compression is everywhere!

- Often part of I/O (input/output)
- Multimedia (codecs)



TAR COMMAND EXAMPLES



Exercise

- Go to this page
- Download **data.zip**
- Uncompress the file
- Copy contents into a new directory (e.g. `day_1`) in this class' directory!

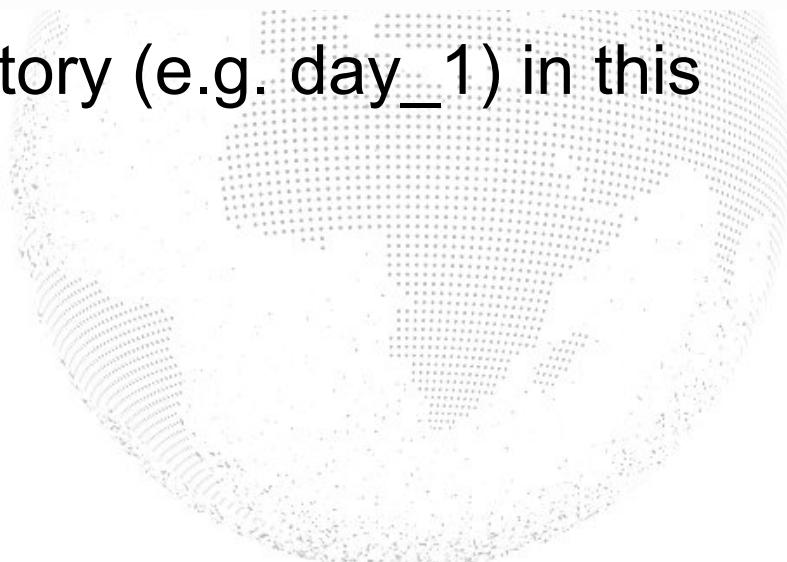
https://adamkocsis.github.io/computers_in_geosciences/

Computers in geosciences Material for the Computers in Geosciences course (2023)

Material for the Computers in Geosciences course (2023)

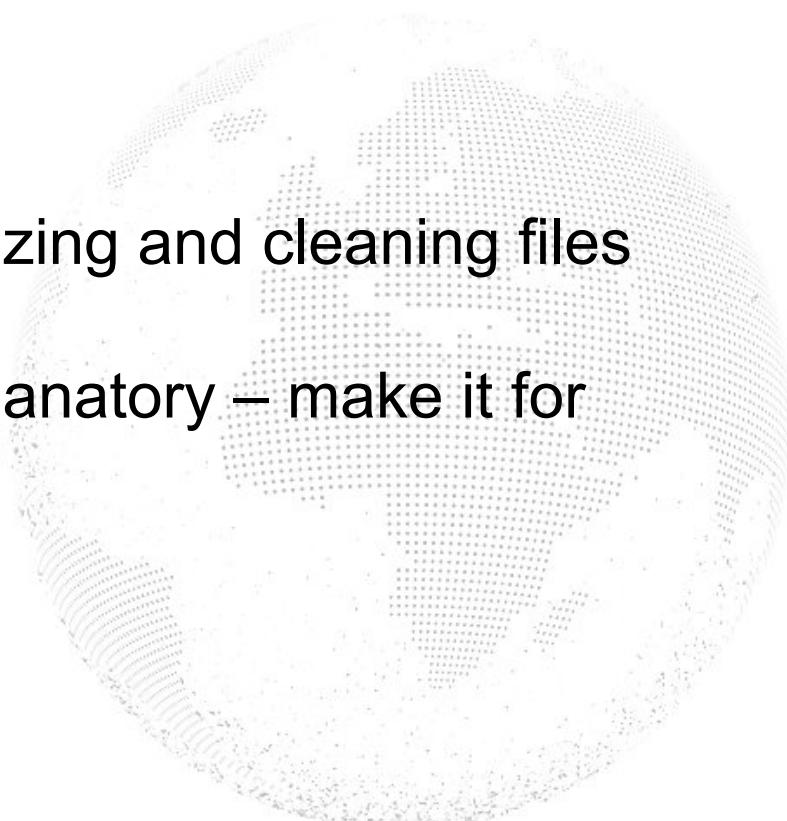
Schedule

Day	Topic	Instructor
October 9 (Monday)	Introduction, Files and BASH essential	Kocsis
October 10 (Tuesday)	Raster and Vector Image processing	Teichert, Schulbert
October 11 (Wednesday)	Open Source software, Programming basics. R as a calculator.	Kocsis
October 12 (Thursday)	R basic features (script reading and modification)	Kocsis
October 13 (Friday)	R basic features (script reading and modification)	Kocsis



Hints and tips for file management

- Keep all your stuff together (separate partition!)
- Logical hierarchy
- Make it portable (Windows!)
- Regularly spend time on organizing and cleaning files
- Naming and grouping: self-explanatory – make it for somebody else (you!)
- Avoid spaces in paths
- Cloud backups!



Novice- vs Expert-friendly tools

No program is perfectly user-friendly! Depending on the task at hand:

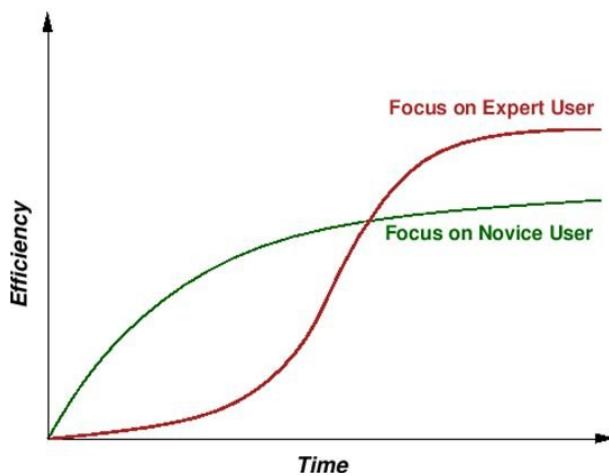
Novice-friendly

- Graphical User Interface (GUI)
- No or very basic training
- Quick learning
- Lower final efficiency
- Visually appealing



Expert-friendly

- Command Line Interpreter (CLI) or **Text-based** interface
- Education/training is necessary
- Eventually higher efficiency
- Visuals: usually invisible!
- Programmable



Working with text is essential

Recommendation: get a code editor!

The image shows the official Visual Studio Code website on the left and the application itself on the right.

Visual Studio Code Website (Left):

- Header: Visual Studio Code, Docs, Updates, Blog, API, Extensions, FAQ, Learn, Search Docs, Download.
- Main Content: "Code editing. Redefined." in large font, "Free. Built on open source. Runs everywhere." below it.
- Downloads: ".deb" (Debian, Ubuntu...), ".rpm" (Red Hat, Fedora...), and "Web, Insiders edition, or other platforms".
- License and Privacy Statement link.

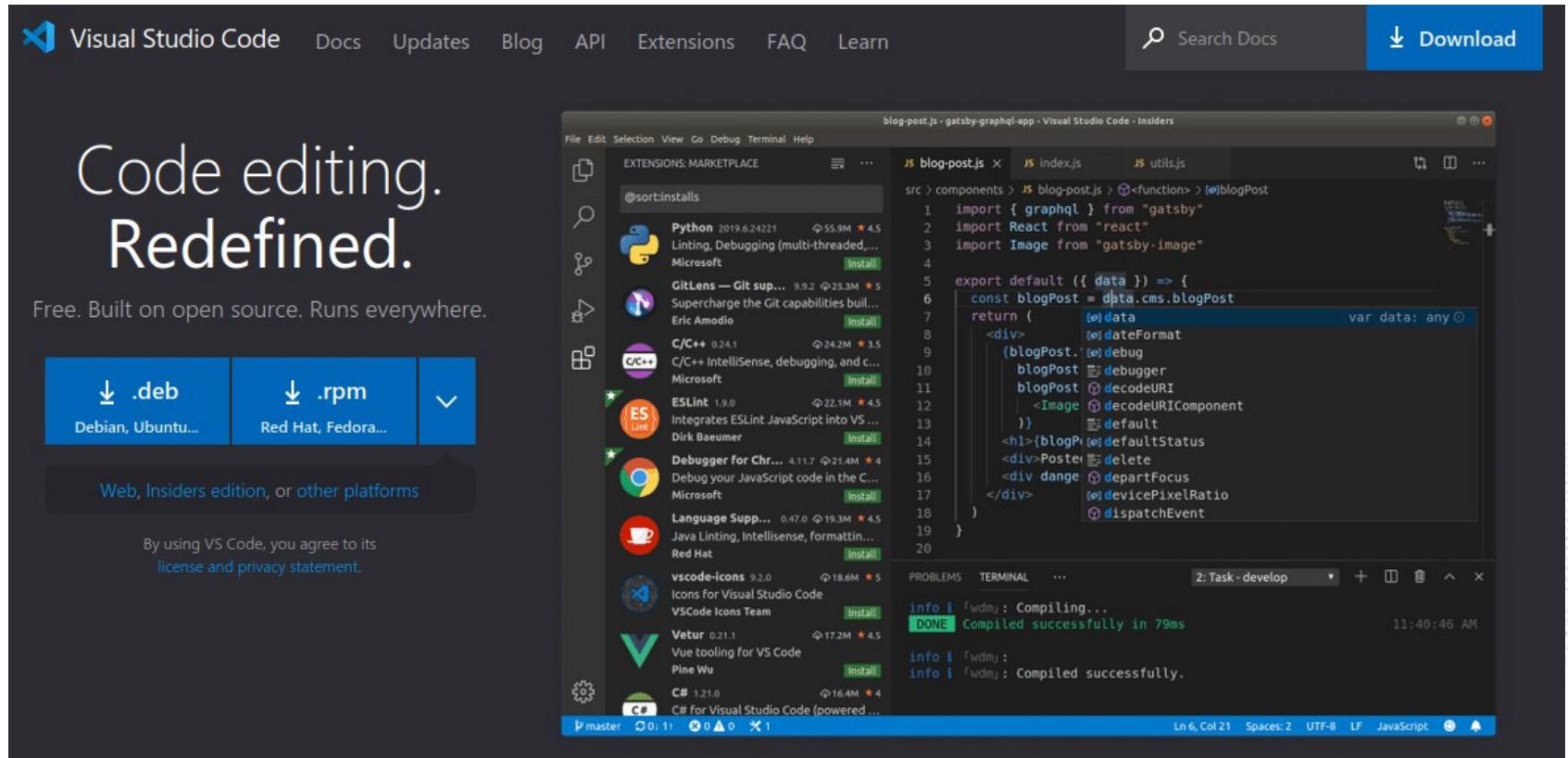
Visual Studio Code Application (Right):

- File Explorer sidebar showing extensions like Python, GitLens, C/C++, ESLint, Debugger, Language Support, etc.
- Code Editor showing a file named "blog-post.js" with some JavaScript code.
- Terminal at the bottom showing build logs: "Compiling...", "Compiled successfully in 79ms", "Compiled successfully.".
- Status bar at the bottom: master, 0:11, 0:0▲0, X:1, Ln 6, Col 21, Spaces: 2, UTF-8, LF, JavaScript.

<https://code.visualstudio.com/>

Working with text is essential

Recommendation: get a code editor!

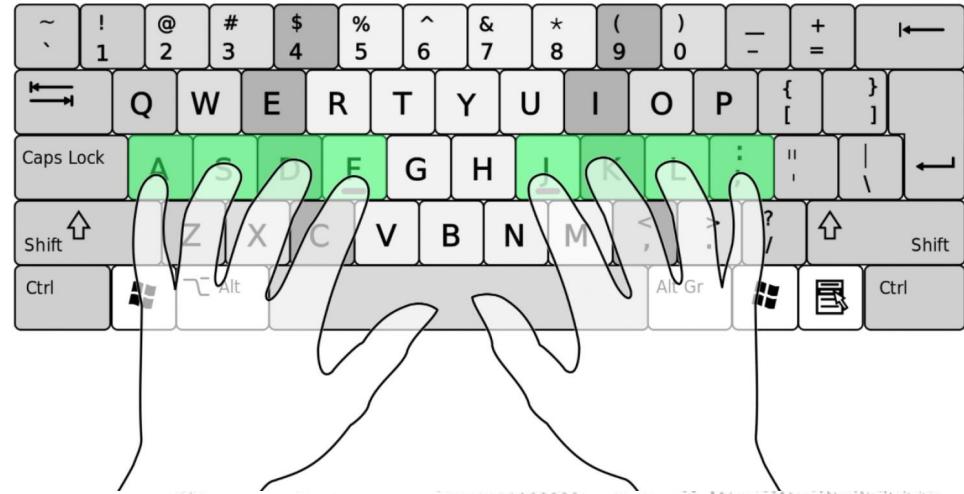
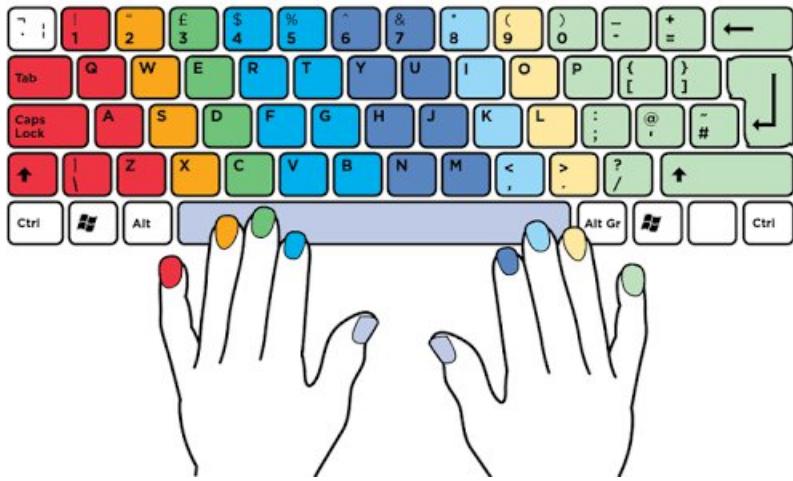


<https://code.visualstudio.com/>

Exercise: install VS Studio Code!

Working with text is essential

Recommendation: learn to touch type, if you don't know



Loads of resources available online!

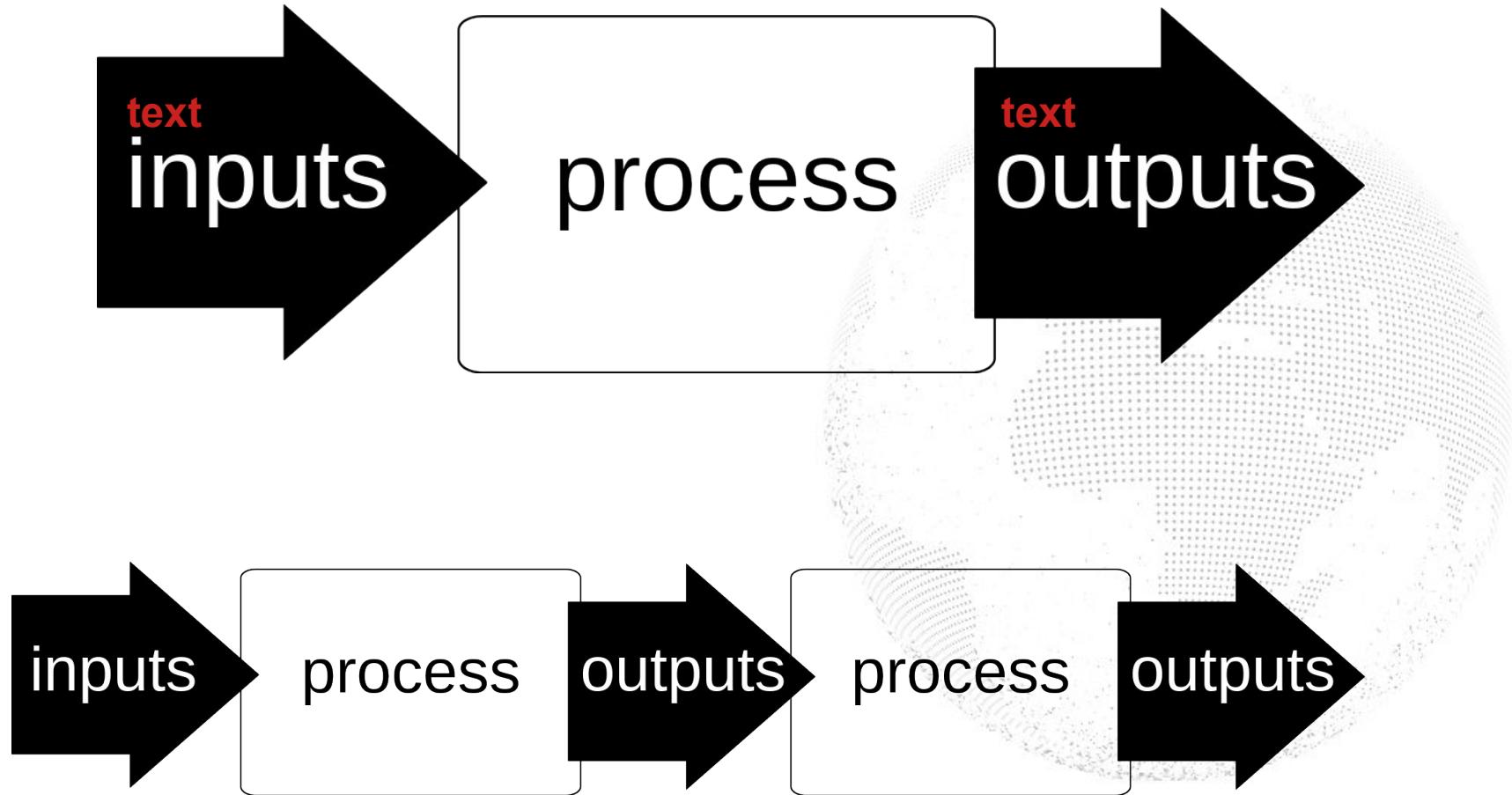
e.g. <https://keybr.com>

R consolidation course!

Enter data < Write instructions

Why text? Universal language

- Easy to connect processes / programs



Why text? Text is actually numbers

Perfect balance between simplicity and complexity

- Text can be represented with numbers, e.g.
ASCII: <https://www.ascii-code.com/>

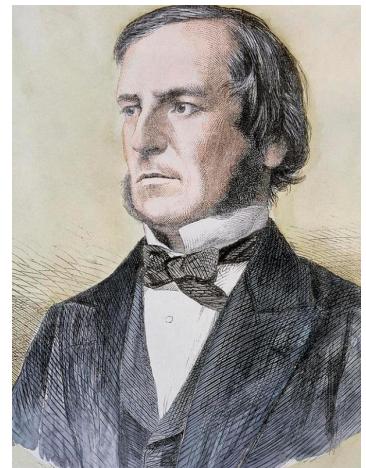
DEC	OCT	HEX	BIN	Symbol	HTML Number	HTML Name	Description
32	040	20	00100000	SP	 		Space
33	041	21	00100001	!	!	!	Exclamation mark
34	042	22	00100010	"	"	"	Double quotes (or speech marks)
35	043	23	00100011	#	#	#	Number sign
36	044	24	00100100	\$	$	$	Dollar
37	045	25	00100101	%	%	%	Per cent sign
38	046	26	00100110	&	&	&	Ampersand
85	125	55	01010101	U	U		Uppercase U
86	126	56	01010110	V	V		Uppercase V
87	127	57	01010111	W	W		Uppercase W
88	130	58	01011000	X	X		Uppercase X
89	131	59	01011001	Y	Y		Uppercase Y
90	132	5A	01011010	Z	Z		Uppercase Z
91	133	5B	01011011	[[[Opening bracket
92	134	5C	01011100	\	\	\	Backslash
93	135	5D	01011101]]]	Closing bracket
94	136	5E	01011110	^	^	^	Caret - circumflex
95	137	5F	01011111	-	_	_	Underscore
96	140	60	01100000	'	`	`	Grave accent
97	141	61	01100001	a	a		Lowercase a
98	142	62	01100010	h	b		Lowercase h

Binary code: 0 and 1

Why binary?

Simplest way to record information

- As type of data: True: (1) and False (0)
- Basis of scientific hypothesis testing – Hypothesis is a statement about reality, that can be answered with true or false. e.g.
- **It is raining outside. (TRUE/FALSE?)**
- Boolean Algebra (Logic)
- Easy to make machines the process information

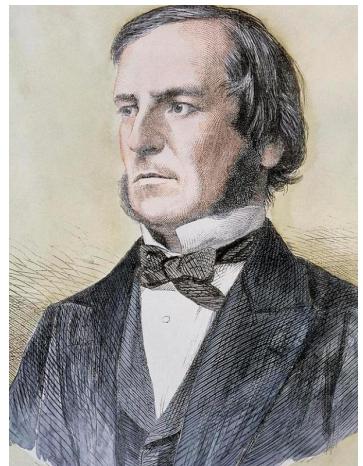


George Boole



Boolean algebra

The logical AND operation



George Boole

Input 1	Input 2	operation	Result
True	True	AND (&)	True
True	False	AND (&)	False
False	True	AND (&)	False
False	False	AND (&)	False

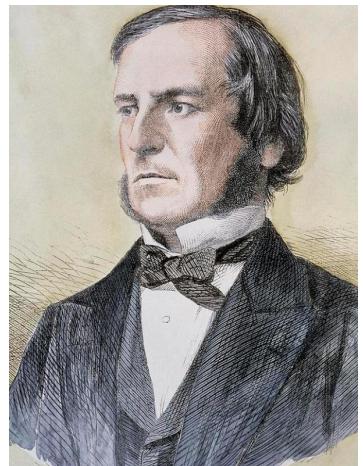
A: True

B: False

A & B = ?

Boolean algebra

The logical OR operation



George Boole

Input 1	Input 2	operation	Result
True	True	OR ()	True
True	False	OR ()	True
False	True	OR ()	True
False	False	OR ()	False

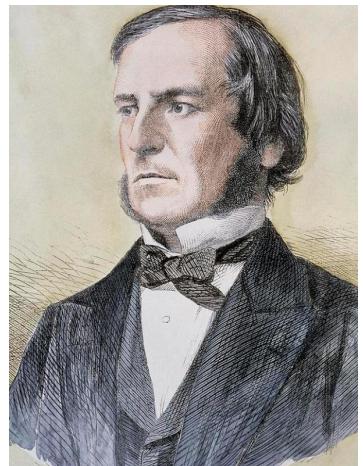
A: True

B: False

$A \mid B = ?$

Boolean algebra

The logical OR operation



George Boole

Input 1	Input 2	operation	Result
True	True	OR ()	True
True	False	OR ()	True
False	True	OR ()	True
False	False	OR ()	False

A: True

B: False

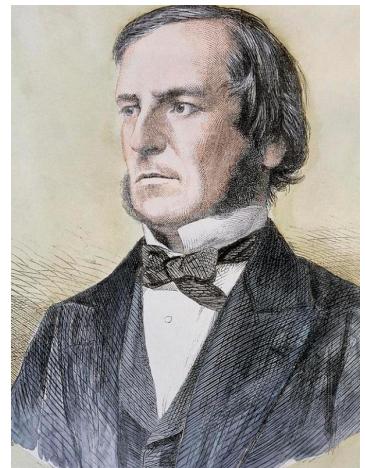
$A \mid B = ?$

C: True

$(A \& B) \mid C = ?$

Boolean algebra

The logical NOT operation



George Boole

Input 1	Input 2	operation	Result
True	False	NOT (!)	False
False	True	NOT (!)	True

A: True

$!A = ?$

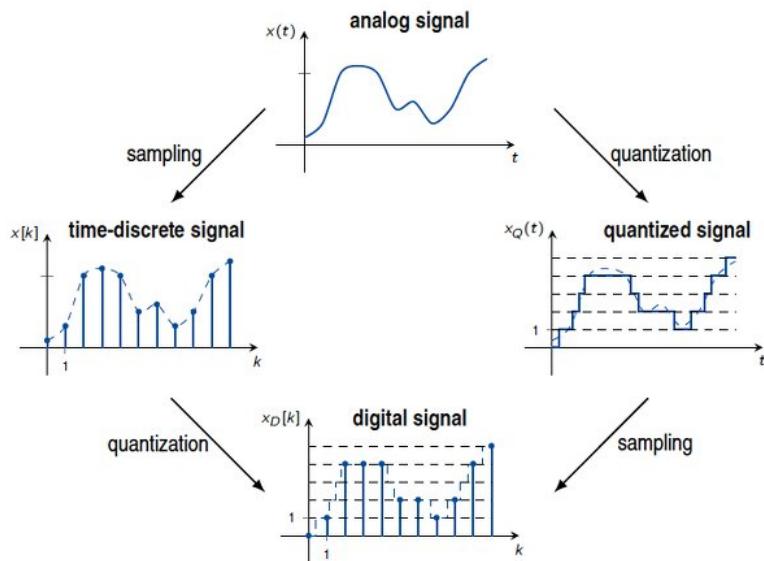
B: True

C: True

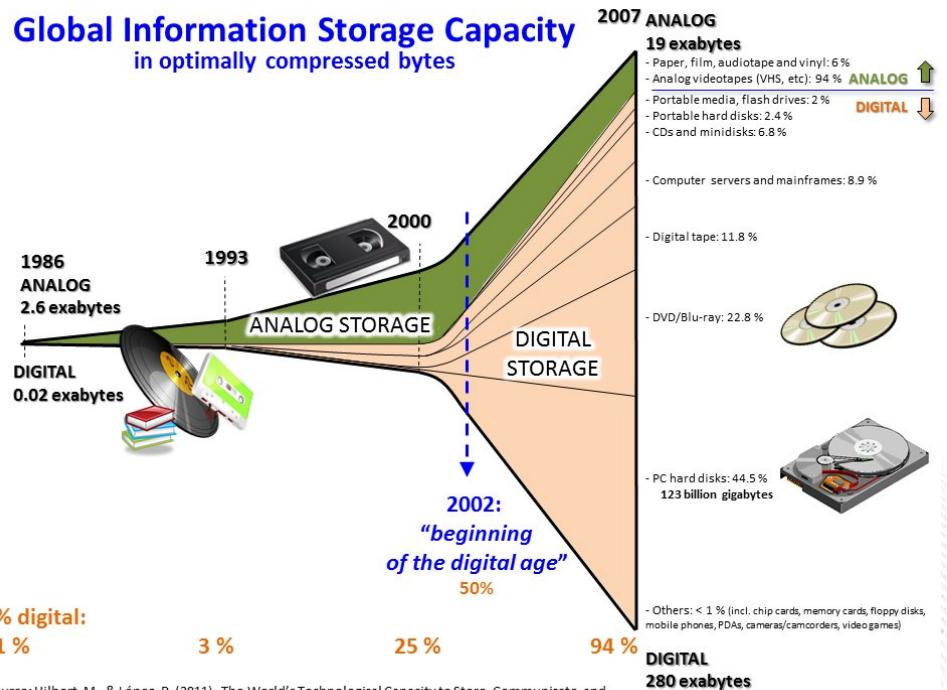
$!(A \& B) | C = ?$

Digital vs Analogue information

Used to build up elementary building blocks of computers



Global Information Storage Capacity
in optimally compressed bytes



Computing and programming

- The concept of calculation: how much is $651/7$?

You have 651 balls.

1. You go through them one-by one.
2. You put every 7th ball in a bin.
3. After done, count the balls. (divisor)



Computing and programming

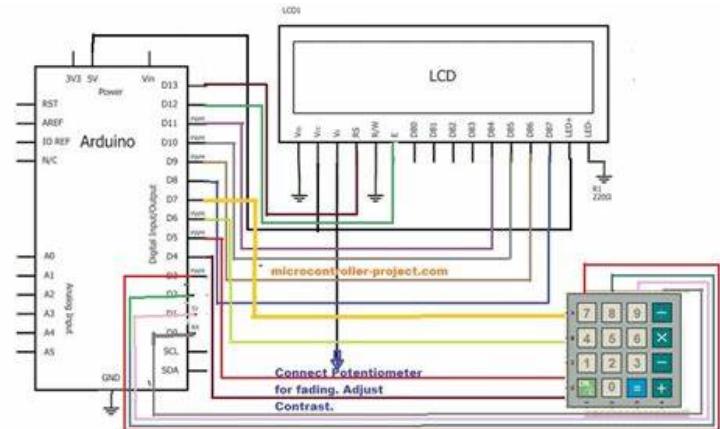


- The concept of calculation: how much is $651/7$?

You have 651 balls.

1. You go through them one-by one.
2. You put every 7th ball in a bin.
3. After done, count the balls. (divisor)

- You can do this with electri

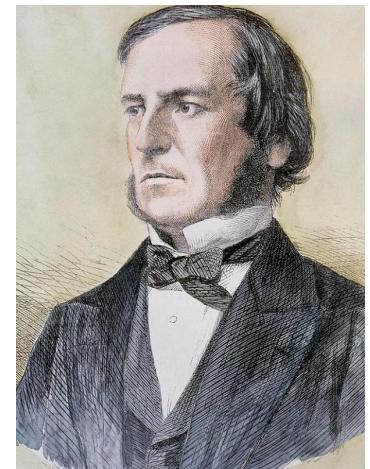


Logic Gates

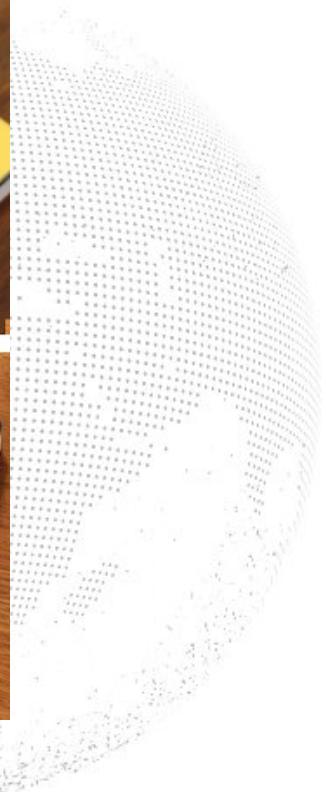
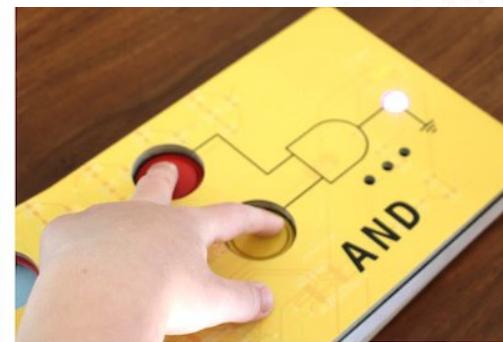
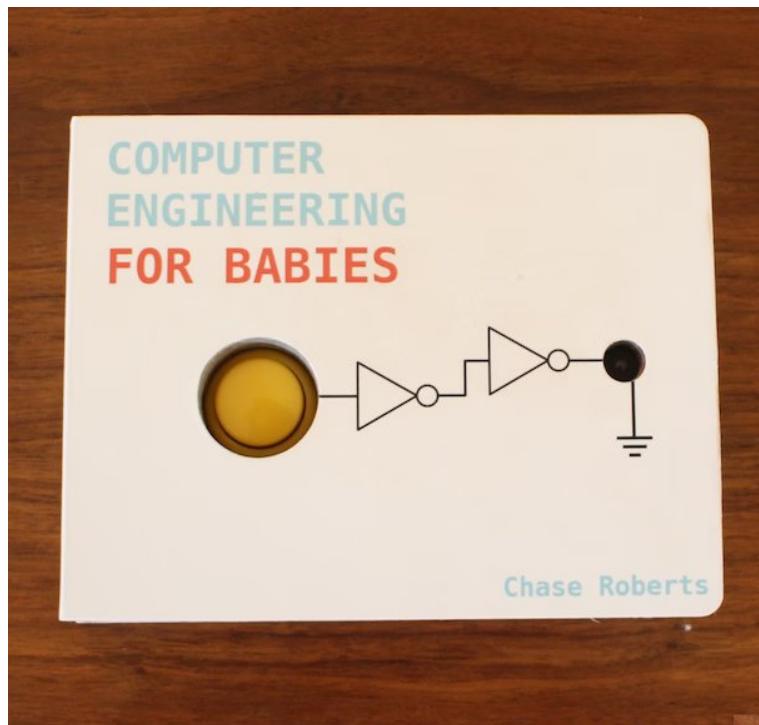
Boolean algebra is relatively simple to implement with physics

True: Electricity!

False: No electricity.



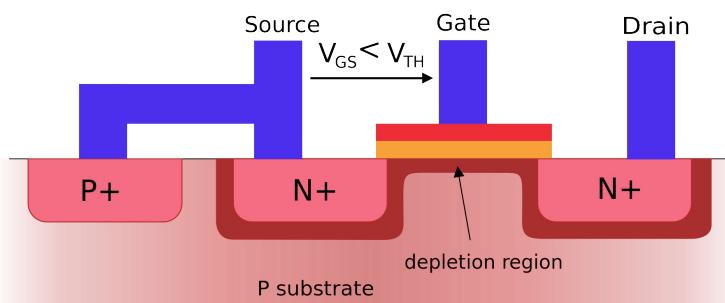
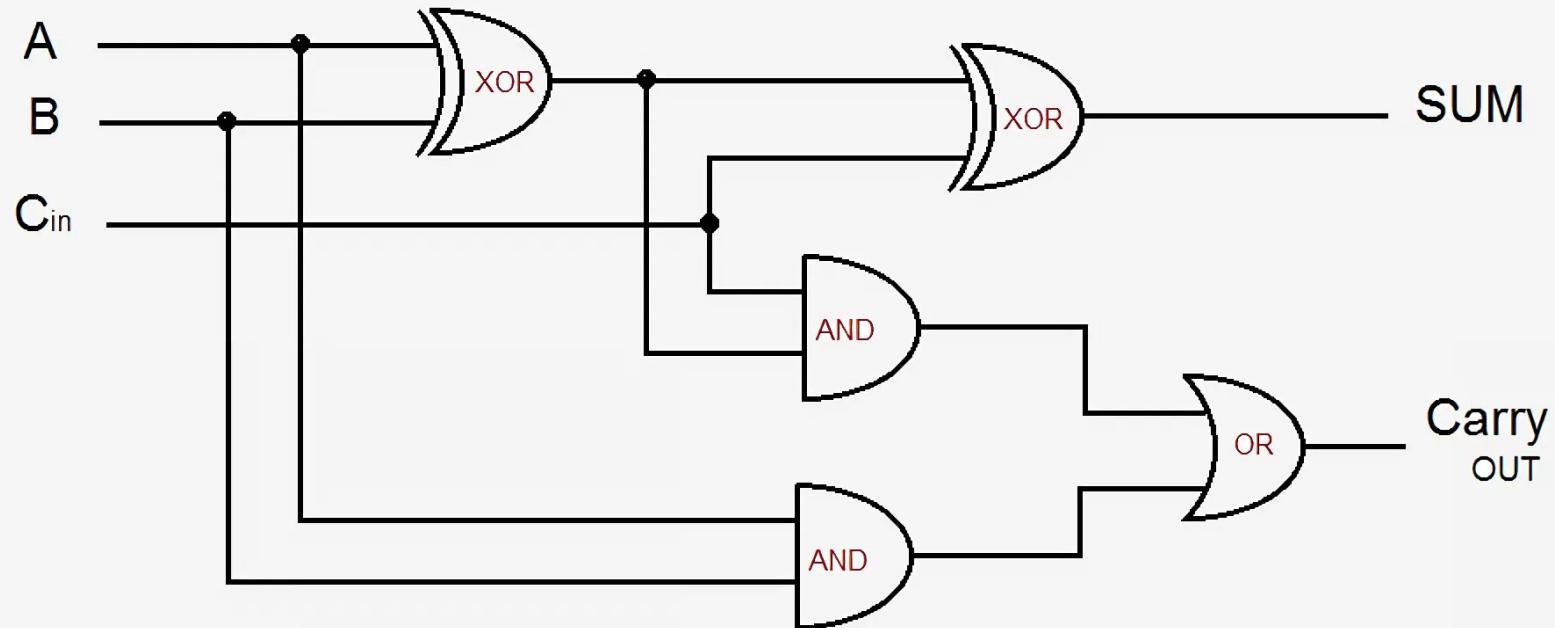
George Boole



https://www.youtube.com/watch?v=_ldfWkZgX1Y

Logic Gates

Used to build up elementary building blocks of computers



Transistors...

<https://en.wikipedia.org/wiki/MOSFET>

Computing and programming

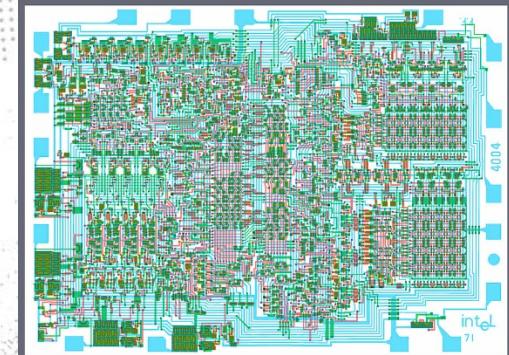
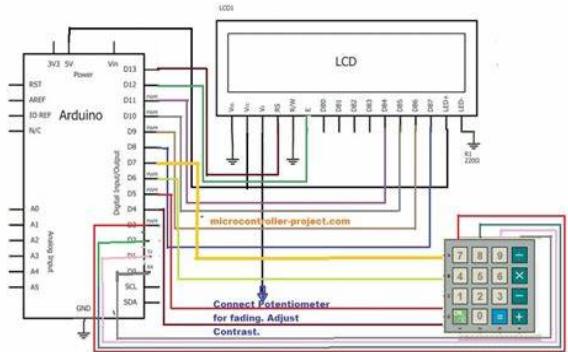


- The concept of calculation: how much is $651/7$?

You have 651 balls.

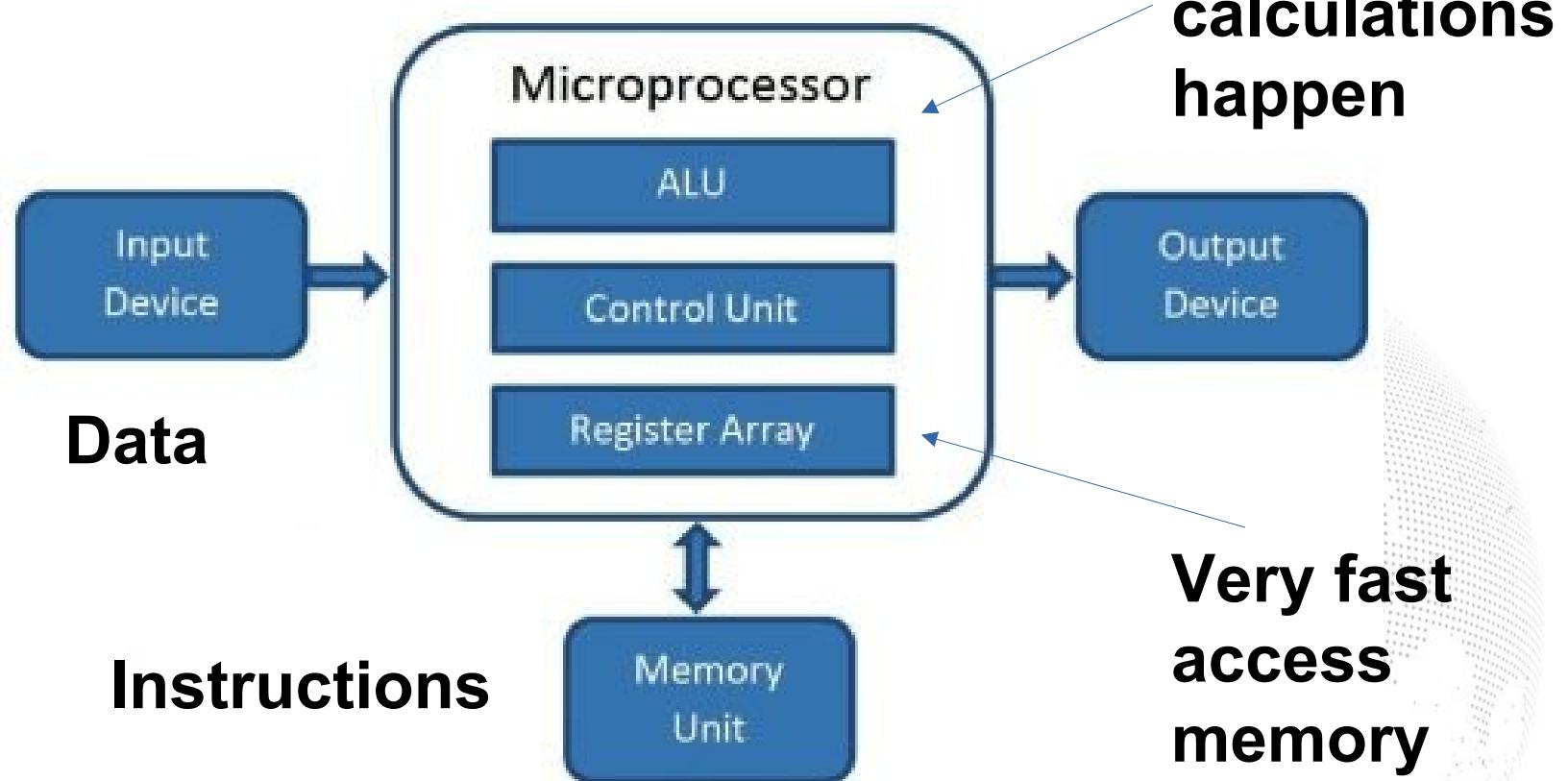
1. You go through them one-by one.
2. You put every 7th ball in a bin.
3. After done, count the balls. (divisor)

- You can do this with electricity
- Use instructions to define a machine that calculates numbers that represent something else (programmable computer)



<https://www.youtube.com/shorts/i2k6jHzK4s>

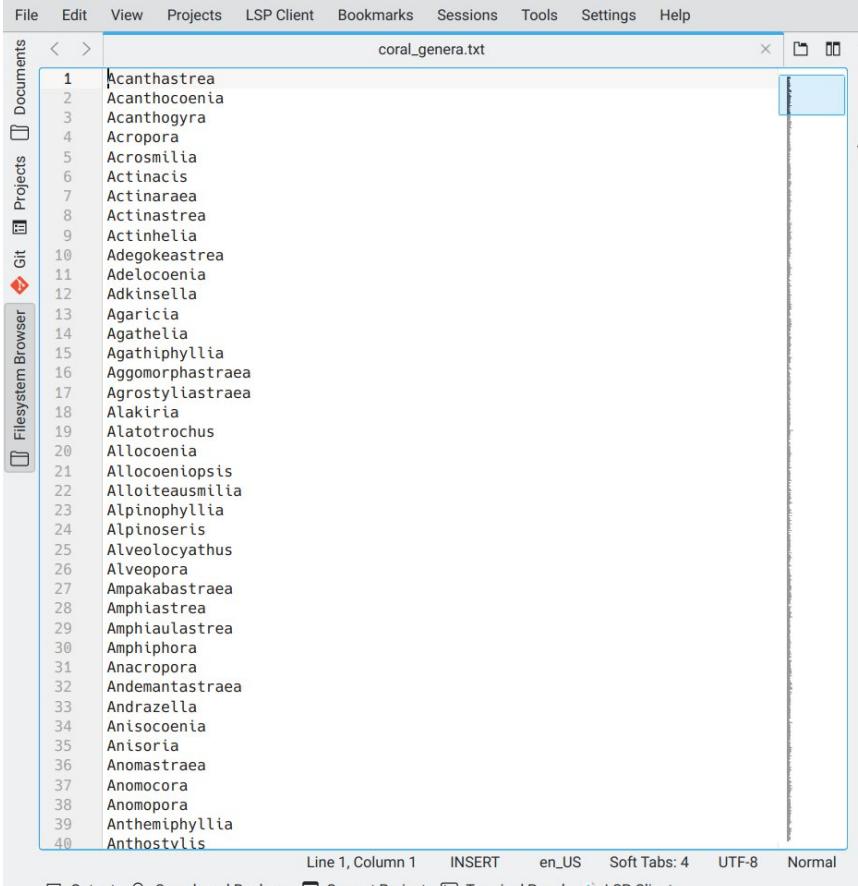
Building up more complex things



Structured text data types

List of entries

- The simplest thing ever
- Convention is to use .txt
- Entries are separated by new lines
- `coral_genera.txt`



The screenshot shows a code editor window with the following details:

- Title Bar:** File Edit View Projects LSP Client Bookmarks Sessions Tools Settings Help
- Document Tab:** coral_genera.txt
- Content:** The file contains a list of 40 coral genera, each preceded by a line number from 1 to 40. The genera listed are:

```
1 Acanthastrea
2 Acanthocenia
3 Acanthogrya
4 Acropora
5 Acrosimilia
6 Actinacis
7 Actinaraea
8 Actinastrea
9 Actinelia
10 Adegokeastrea
11 Adelocoenia
12 Adkinsella
13 Agaricia
14 Agathelia
15 Agathiphyllia
16 Aggomorphastraea
17 Agrostyliastraea
18 Alakiria
19 Alatotrochus
20 Allocenia
21 Alloceniopsis
22 Alloiteausmilia
23 Alpinophyllia
24 Alpinoseris
25 Alveolocyathus
26 Alveopora
27 Ampakabastraea
28 Amphiastrea
29 Amphialastrea
30 Amphiphora
31 Anacropora
32 Andemantastraea
33 Andrazella
34 Anisocoenia
35 Anisoria
36 Anomastraea
37 Anomocora
38 Anomopora
39 Anthemiphyllia
40 Anthostylis
```
- Status Bar:** Line 1, Column 1 INSERT en_US Soft Tabs: 4 UTF-8 Normal
- Bottom Buttons:** Output, Search and Replace, Current Project, Terminal Panel, LSP Client

Structured text data types

(csv) Comma-Separated Values

- Frequently used to represent tabular data
- Rows in lines
- Values separated by commas
- Example: corals.csv



```
"10422","occ","","","","601","Phacops sp.","genus","21701","","Phacops","genus","21701","Pragian","","410.8","407.6","60918"
"10438","occ","","","","602","Proetida indet.","order","21062","","Proetida","order","21062","Pragian","","410.8","407.6","27289"
"10439","occ","","","","602","Phacops sp.","genus","21701","","Phacops","genus","21701","Pragian","","410.8","407.6","27289"
"10440","occ","","","","602","Leonaspis sp.","genus","19814","","Leonaspis","genus","19814","Pragian","","410.8","407.6","27289"
"10558","occ","","","","605","Dalmanites sp.","genus","21523","","Dalmanites","genus","21523","Emsian","","407.6","393.3","60970"
"10559","occ","","","","605","Phacops sp.","genus","21701","","Phacops","genus","21701","Emsian","","407.6","393.3","60970"
"10567","occ","","","","606","Leonaspis sp.","genus","19814","","Leonaspis","genus","19814","Emsian","","407.6","393.3","60984"
"10568","occ","","","","606","Phacopida indet.","order","21421","","Phacopida","order","21421","Emsian","","407.6","393.3","60984"
"10569","occ","","","","606","Dechenella sp.","genus","21144","","Dechenella","genus","21144","Emsian","","407.6","393.3","60984"
"10573","occ","","","","607","Basidechenella sp.","genus","21087","","Basidechenella","genus","21087","Emsian","","407.6","393.3","60984"
"10574","occ","","","","607","Otarion sp.","genus","21275","","Otarion","genus","21275","Emsian","","407.6","393.3","60984"
"10575","occ","","","","607","Proetida indet.","order","21062","","Proetida","order","21062","Emsian","","407.6","393.3","60984"
"10576","occ","","","","607","Terataspis sp.","genus","19765","","Terataspis","genus","19765","Emsian","","407.6","393.3","60984"
"10577","occ","","","","607","Leonaspis sp.","genus","19814","","Leonaspis","genus","19814","Emsian","","407.6","393.3","60984"
"10631","occ","","","","608","Phacops sp.","genus","21701","","Phacops","genus","21701","Emsian","","407.6","393.3","13441"
"10683","occ","","","","612","Phacopida ? indet.","order","21421","","Phacopida","order","21421","Eifelian","","393.3","387.7","17056"
"10704","occ","","","","613","Phacopida ? indet.","order","21421","","Phacopida","order","21421","Eifelian","","393.3","387.7","61511"
"10716","occ","","","","614","Otarion sp.","genus","21275","","Otarion","genus","21275","Eifelian","","393.3","387.7","272"
"10745","occ","","","","616","Leonaspis sp.","genus","19814","","Leonaspis","genus","19814","Eifelian","","393.3","387.7","61110"
"10746","occ","","","","616","Proetus sp.","genus","21327","","Proetus","genus","21327","Eifelian","","393.3","387.7","61110"
"10784","occ","","","","619","Otarion sp.","genus","21275","","Otarion","genus","21275","Eifelian","Givetian","393.3","382.7","61512"
"10811","occ","","","","621","Proetida ? indet.","order","21062","","Proetida","order","21062","Eifelian","Givetian","393.3","382.7","61512"
"10820","occ","","","","622","Dechenella sp.","genus","21144","","Dechenella","genus","21144","Eifelian","Givetian","393.3","382.7","61512"
"10971","occ","","","","627","Dechenella sp.","genus","21144","","Dechenella","genus","21144","Givetian","","387.7","382.7","841"
"10972","occ","","","","627","Phacops sp.","genus","21701","","Phacops","genus","21701","Givetian","","387.7","382.7","841"
"10973","occ","","","","627","Greenops sp.","genus","21588","","Greenops","genus","21588","Givetian","","387.7","382.7","841"
"10974","occ","","","","627","Trimerus sp.","genus","21805","","Trimerus","genus","21805","Givetian","","387.7","382.7","841"
"11038","occ","","","","632","Proetus sp.","genus","21327","","Proetus","genus","21327","Frasnian","","382.7","372.2","61501"
"11099","occ","","","","640","Trimerococephalus sp.","genus","21804","","Trimerococephalus","genus","21804","Late Devonian","","382.7","358.9","60994"
```

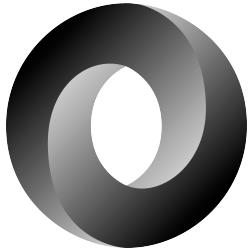
Some structured text “languages”

(csv) Comma-Separated Values (variants)

- Separator can be different (e.g. semicolon (;) or white-space (\t, \s))
- Semicolon-separated example: stages.csv
- Tab-delimited: penguins.tab



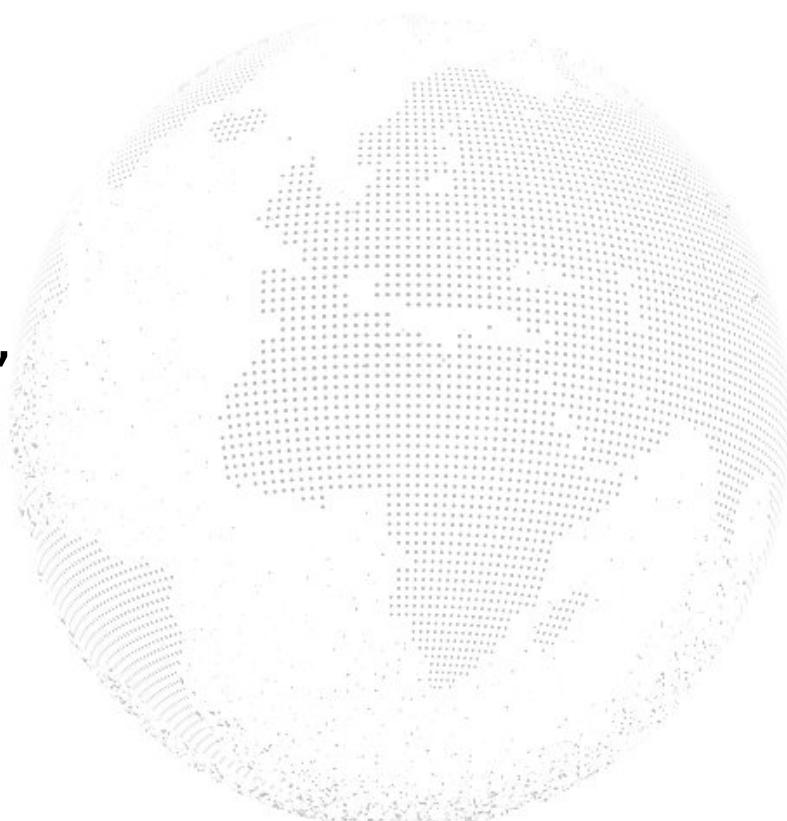
Structured text data types



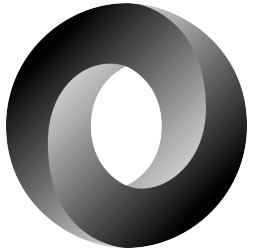
(JSON) JavaScript Object Notation

- Text-based format of key-value pairs

```
{  
  "firstname": "Adam",  
  "lastname": "Kocsis"  
}
```



Structured text data files

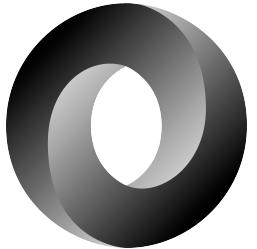


(JSON) JavaScript Object Notation

- Text-based format of key-value pairs
- Allows hierarchical structuring, multiple values/keys ('array')
- Can be made complicated, but very straightforward

```
{  
  "firstName": "John",  
  "lastName": "Smith",  
  "isAlive": true,  
  "age": 27,  
  "address": {  
    "streetAddress": "21 2nd Street",  
    "city": "New York",  
    "state": "NY",  
    "postalCode": "10021-3100"  
  },  
  "phoneNumbers": [  
    {  
      "type": "home",  
      "number": "212 555-1234"  
    },  
    {  
      "type": "office",  
      "number": "646 555-4567"  
    }  
  ],  
  "children": [  
    "Catherine",  
    "Thomas",  
    "Trevor"  
  ],  
  "spouse": null  
}
```

Structured text data files



(YAML) Yet Another Markup Language

- Similar Text-based format that allows hierarchical structuring
- Key-value pairs
- Similar to JSON

```
- name: Computers in Geosciences
  nickname: Computers in Geosciences
  ref: computers
  group:
    - key
    - wahl
  short: "Usually just called the '<i>computers course"
  note: "As many other courses in the Paleobiology ma
  more: "Essential tasks on the computer including imag
  evaluation: "Attendance and participation is required
  administrator: kocsis
  instructors:
    - kiessling
    - teichert
    - kocsis
  ects: 5
  campo: "https://www.campo.fau.de:443/qisserver/pages
  studon: "https://www.studon.fau.de/crs1321742.html"
  photos:
    thumbnail: "images/courses-thumbnails/computers.jpg"
    image:
      title: "images/courses/big/computers.jpg"
  form: "1 week block course."
  type: "practical"
```

Structured text data files



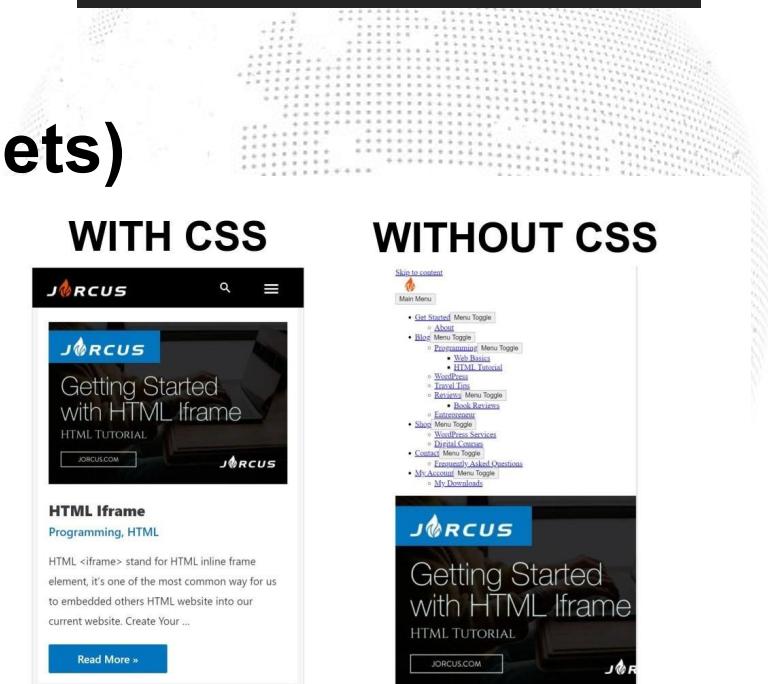
HTML (Hypertext Markup Language)

- Used to structure webpages, based on 'tags'
- Also used for interface development

```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <meta charset="UTF-8">
5      <title>Title goes here</title>
6    </head>
7    <body>
8
9    </body>
10   </html>
```

CSS (Cascading Style Sheets)

- Adding formatting to webpages



The image shows a comparison between two versions of a website: one styled with CSS and one without. The top part of the image shows a dark background with a grid pattern, while the bottom part shows the actual website layout.

WITH CSS: This section shows a screenshot of a website featuring a header with the logo 'JORCUS', a search bar, and a menu icon. Below the header is a main content area with a dark background. The content includes a title 'Getting Started with HTML Iframe', a subtitle 'HTML TUTORIAL', and a 'Read More >' button. The overall design is clean and modern.

WITHOUT CSS: This section shows the same website layout but without CSS styling. The text is less readable due to lack of contrast, and the overall appearance is less polished. The menu on the right is more complex and lacks the visual hierarchy provided by CSS.

Structured text data files

XML (eXtensible Markup Language)



- Storing Arbitrary Data
- Very similar to HTML
- Many file formats are based on this
(OOXML, e.g. MS Office)

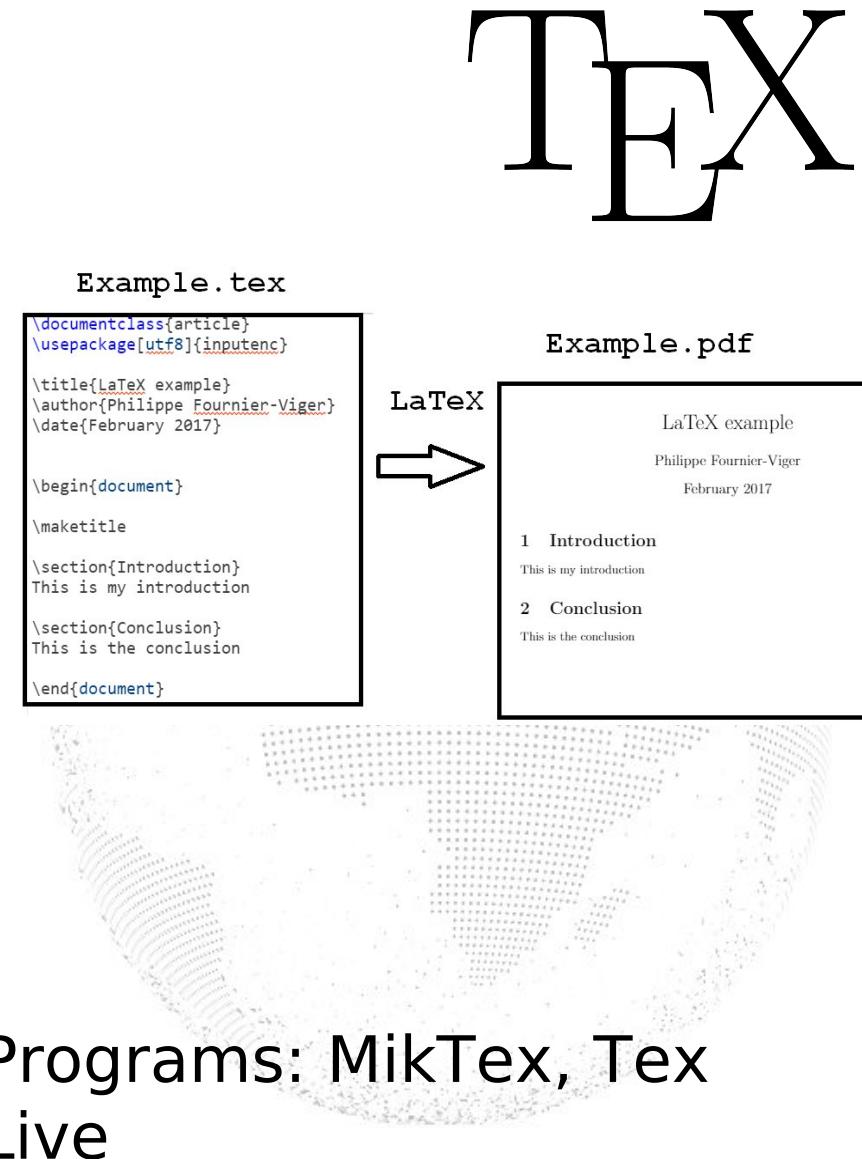


Structured text data files

TeX/LaTeX

- Markup language for typesetting documents (e.g. creating .pdf files)
- LaTeX is a generalized implementation
- Excellent for mathematical expressions

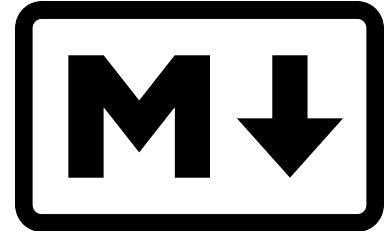
$${}^w\phi = {}^wf^{-1}({}^0f({}^0\phi))$$



Programs: MikTex, Tex Live

Structured text data files

Markdown



- Developed for easier web development
- Very clean and easy syntax
- Frequently used in ‘literate programming’*
- Various flavors (e.g. R-markdown)



*Methodology that combines programming with a documentation language