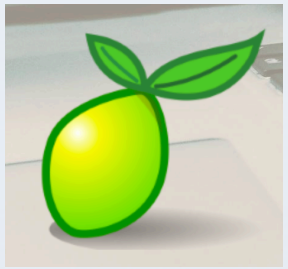
# Useful open science tools to do research in psychiatry

Thomas Gargot
thomas_gargot@hotmail.com
EPA Congress,
Tuesday March 6th

# Useful open science tools to do research in psychiatry

**Randomizer.org**

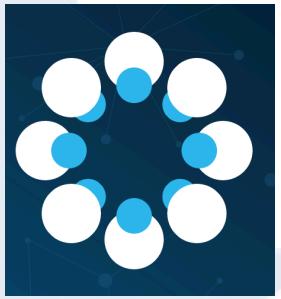
**Collect data**
Limesurvey



RStudio®

**Analyse data**

**Share data**
OSF and
GitHub



equator
network

**Write an article**


**Make the knowledge more visible**

# Goals of the workshop

- Understand the challenges of open science
- Learn the concepts of some tools
- Consider your own research/projects
- Rate cakes

# Who I am

- Psychiatrist in Paris
- PhD Student on Computer Science :

  use of IT in school

La pitié Salpétrière

- Interest in IT and psychotherapy (CBT)
- Master in Cognitive science in ENS

- Former IT secretary of EFPT
- Psychotherapy WG chairman

EUROPEAN FEDERATION OF PSYCHIATRIC TRAINEES

# Disclosure

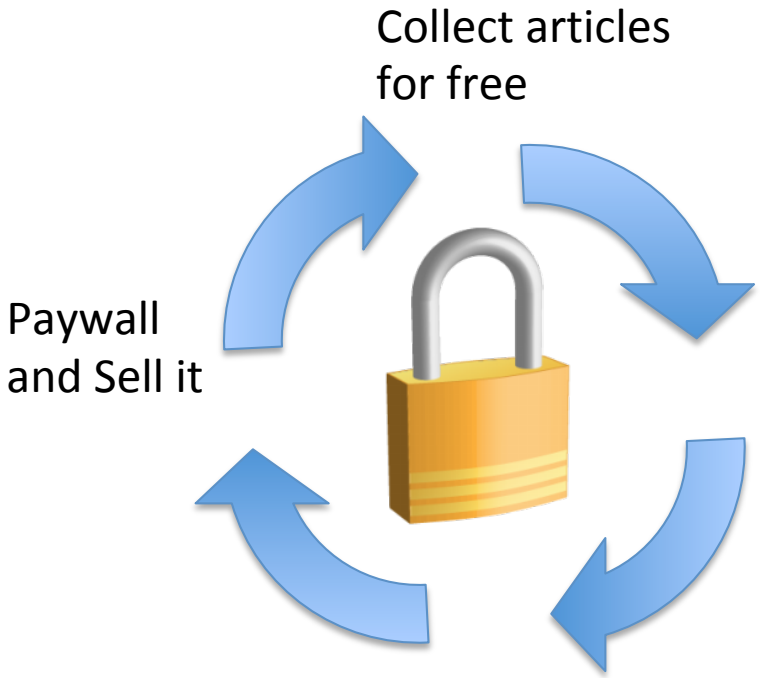
- None

# Ethical issues

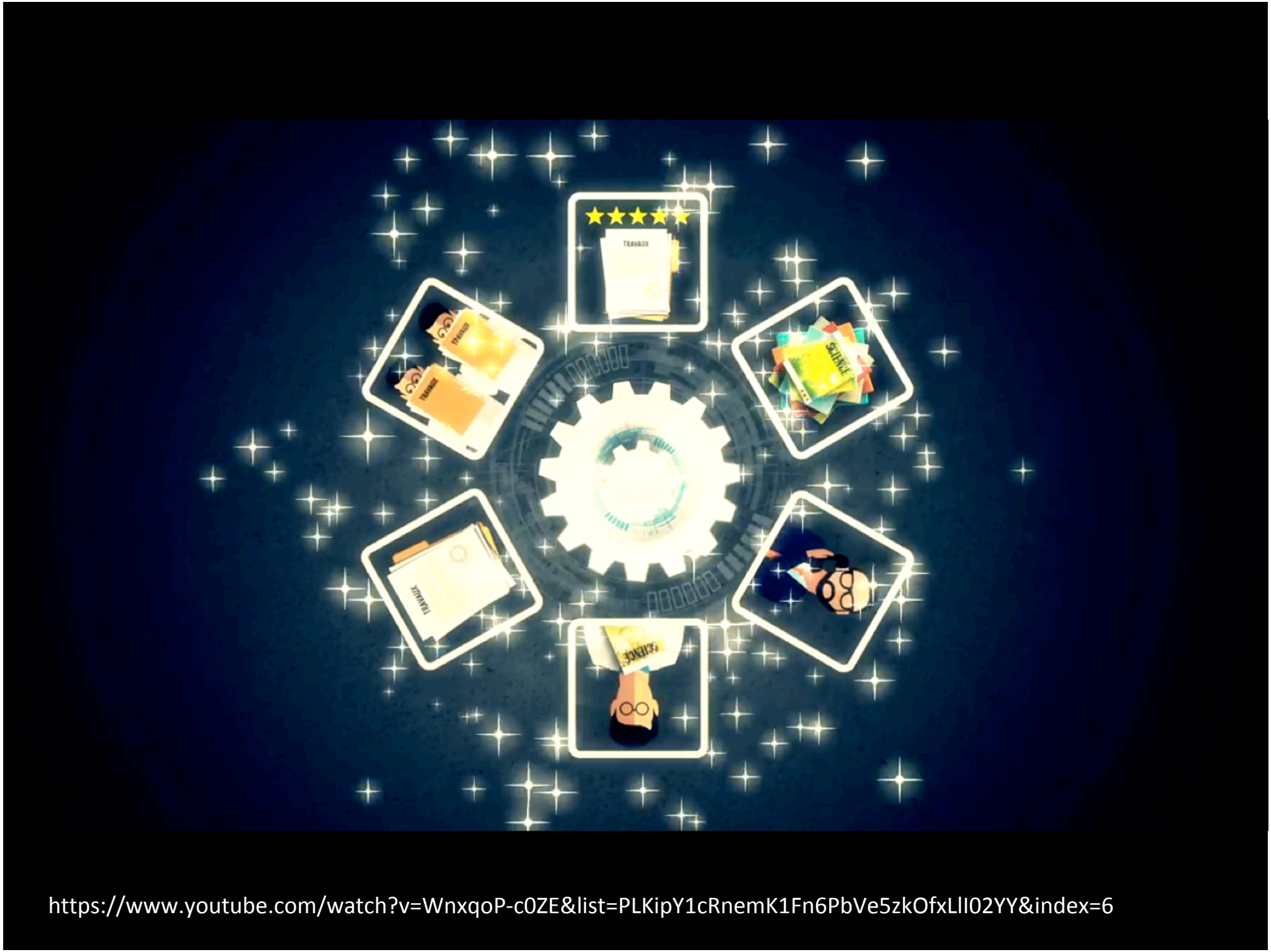
# Problems

- **Cost:** How much will it cost to run this study?
- **Method**: What do they mean when they did this analysis?
- **Licence**: Does someone has the last version of this software ?
- **Memory**: Where is my data, how to re-use, how share it?
- **Paywall**: I can’t get this article
- **Visibility**: Nobody cares about our work

# Publication lucrative vicious circle




Research privatization




https://www.youtube.com/watch?v=WnxqoP-c0ZE&list=PLKipY1cRnemK1Fn6PbVe5zkOfxLlI02YY&index=6

# Goals

- Decrease costs
- Improve transparency
- Improve accessibility to softwares, articles, and re-use of data
- Improve visibility for public

➔ Is it really possible ?!?

# Why is there a need for open science?

nature human behaviour

PERSPECTIVE

PUBLISHED: 10 JANUARY 2017 | VOLUME: 1 | ARTICLE NUMBER: 0021

OPEN

## A manifesto for reproducible science

Marcus R. Munafò[1,2*], Brian A. Nosek[3,4], Dorothy V. M. Bishop[5], Katherine S. Button[6], Christopher D. Chambers[7], Nathalie Percie du Sert[8], Uri Simonsohn[9], Eric-Jan Wagenmakers[10], Jennifer J. Ware[11] and John P. A. Ioannidis[12,13,14]

Data from many fields suggests reproducibility is lower than is desirable[8–14]; one analysis estimates that 85% of biomedical research efforts are wasted[14], while 90% of respondents to a recent survey in *Nature* agreed that there is a 'reproducibility crisis'[15].

https://www.nature.com/articles/s41562-016-0021

# Information world

- Share information don’t decrease the ownership of the sharer
- Rather, better accessibility improve possibilities of not anticipated use (hacking) and feedback
- Open softwares are more flexible and have fewer bugs that closed/proprietary one

# Free / Open?


**Open software**: Free use, study, modification, duplication and sharing

≠ Freemium strategy (demonstration)

≠ centralisation and user data collection

https://www.qwant.com/?q=you%20are%20the%20product&t=images&o=0:0a530ac5ad25daeb329762e4a2da88f3

# An example of Open science pathway

Hands-on

# Research protocol

- Make a data base of cookies tasting
- Rate them in an international context
- Select the best cakes in the world


Sampling process during EPA congress, Madrid, 2016

# Useful open science tools to do research in psychiatry

Randomizer.org

# Randomizer.org

# Randomizer.org


## RESULTS

PRINT DOWNLOAD CLOSE

1 Set of 10 Numbers
Range: From 1 to 2

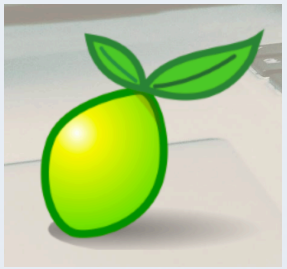
**Set #1**

2, 2, 2, 1, 1, 2, 2, 1, 2, 1

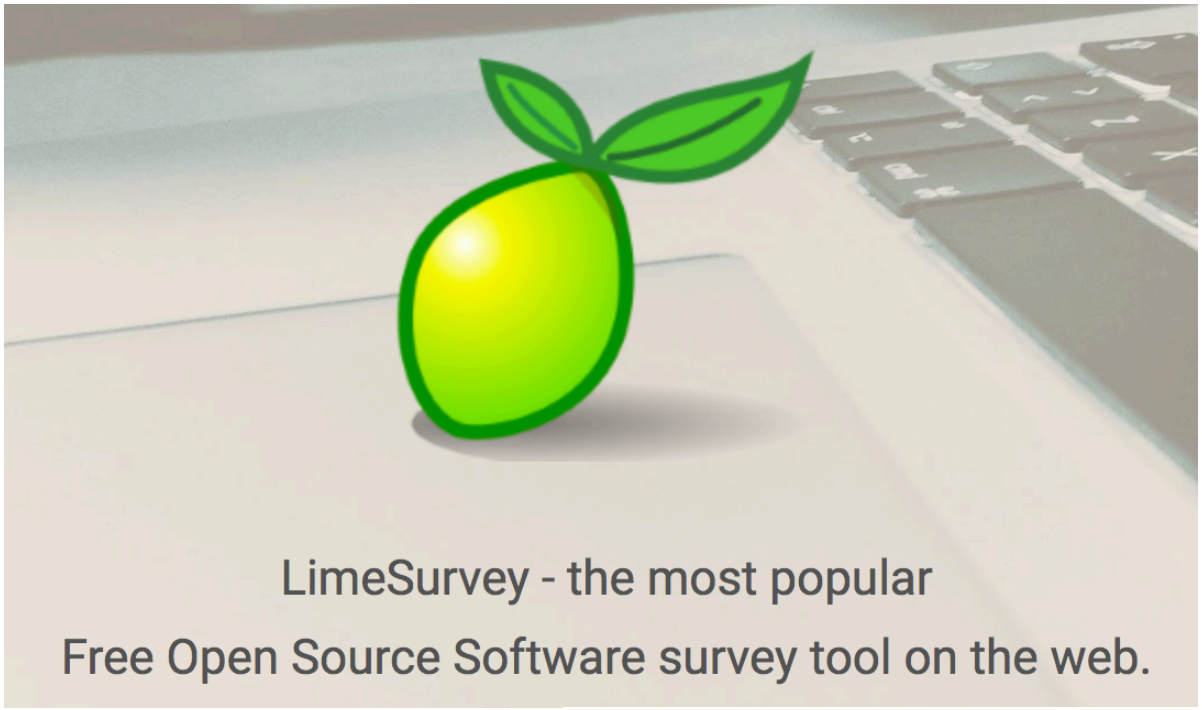
# Useful open science tools to do research in psychiatry

**Randomizer.org**

**Collect data**
Limesurvey



# Limesurvey




Create professional online surveys
LimeService
"Truth fears no questions."

http://thecakereports.limequery.com

How does it look like ?

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ● No answer

How does it taste ?

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ● No answer

How is the texture ?

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ● No answer

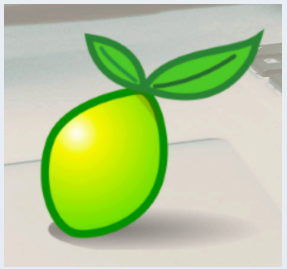
How is the packaging ?

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ● No answer

# Useful open science tools to do research in psychiatry

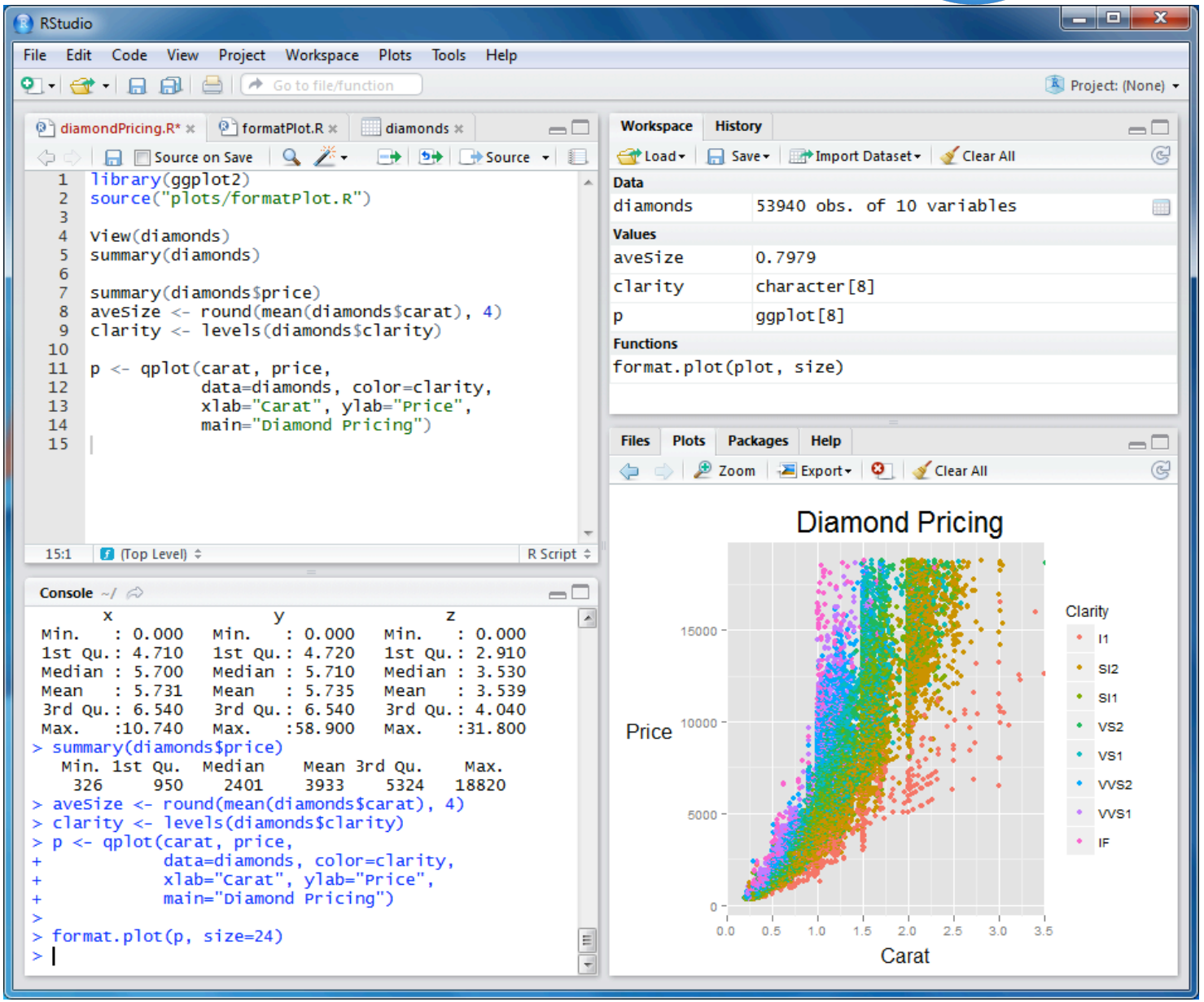
**Randomizer.org**

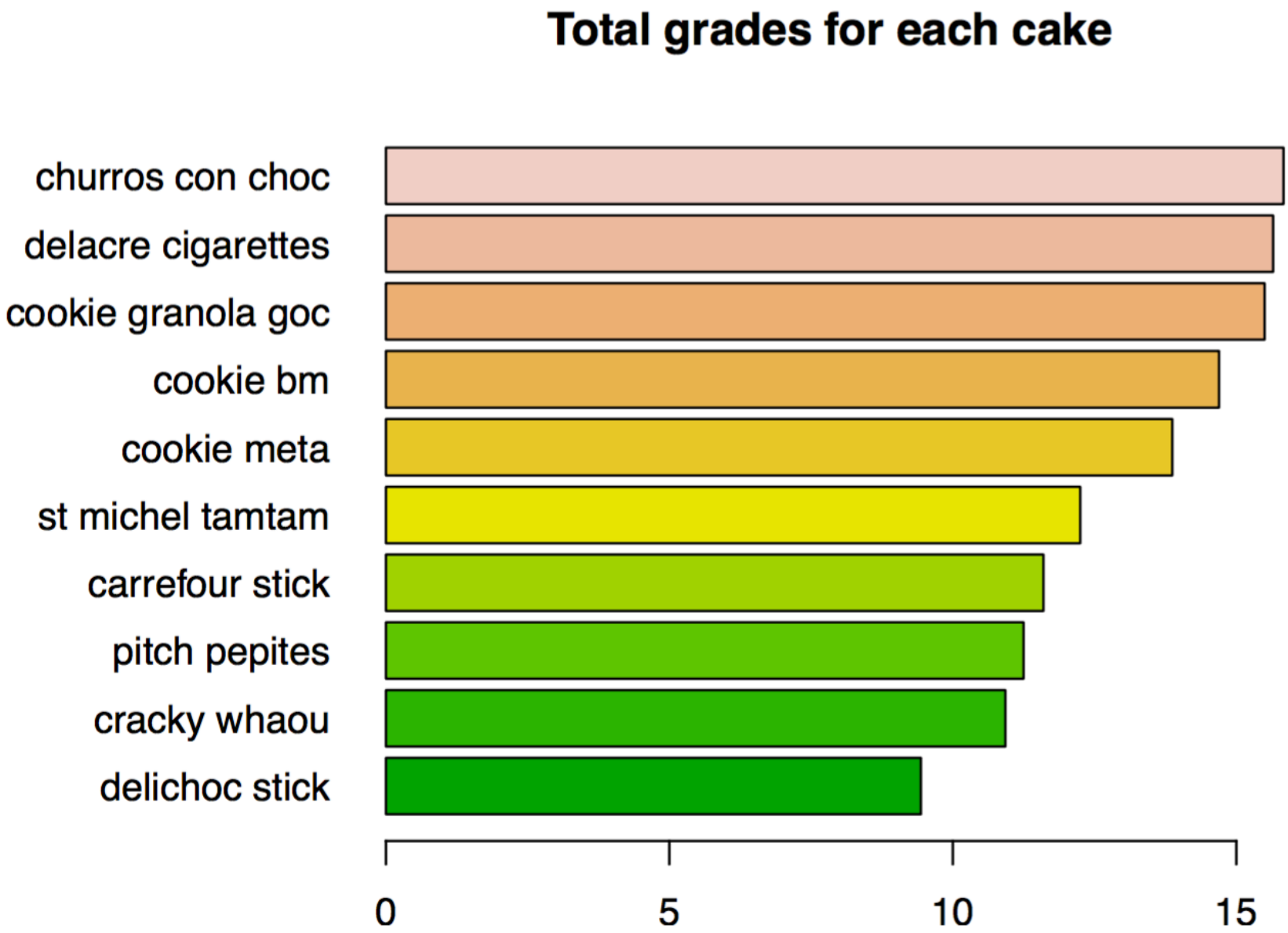
**Collect data**
Limesurvey



**Analyse data**

RStudio®


RStudio
File Edit Code View Project Workspace Plots Tools Help
Go to file/function
Project: (None)
diamondPricing.R*
formatPlot.R
diamonds
Source on Save
Source
1 library(ggplot2)
2 source("plots/formatPlot.R")
3
4 view(diamonds)
5 summary(diamonds)
6
7 summary(diamonds$price)
8 aveSize <- round(mean(diamonds$carat), 4)
9 clarity <- levels(diamonds$clarity)
10
11 p <- qplot(carat, price,
12 data=diamonds, color=clarity,
13 xlab="Carat", ylab="Price",
14 main="Diamond Pricing")
15
15:1 (Top Level) R Script
Workspace History
Load Save Import Dataset Clear All
Data
diamonds 53940 obs. of 10 variables
Values
aveSize 0.7979
clarity character[8]
p ggplot[8]
Functions
format.plot(plot, size)
Files Plots Packages Help
Zoom Export Clear All
Diamond Pricing
Price
Carat
Clarity
I1
SI2
SI1
VS2
VS1
VVS2
VVS1
IF
Console ~/
x y z
Min. : 0.000 Min. : 0.000 Min. : 0.000
1st Qu.: 4.710 1st Qu.: 4.720 1st Qu.: 2.910
Median : 5.700 Median : 5.710 Median : 3.530
Mean : 5.731 Mean : 5.735 Mean : 3.539
3rd Qu.: 6.540 3rd Qu.: 6.540 3rd Qu.: 4.040
Max. :10.740 Max. :58.900 Max. :31.800
> summary(diamonds$price)
Min. 1st Qu. Median Mean 3rd Qu. Max.
326 950 2401 3933 5324 18820
> aveSize <- round(mean(diamonds$carat), 4)
> clarity <- levels(diamonds$clarity)
> p <- qplot(carat, price,
+ data=diamonds, color=clarity,
+ xlab="Carat", ylab="Price",
+ main="Diamond Pricing")
>
> format.plot(p, size=24)
>


Total grades for each cake
churros con choc
delacre cigarettes
cookie granola goc
cookie bm
cookie meta
st michel tamtam
carrefour stick
pitch pepites
cracky whaou
delichoc stick
0
5
10
15

# Connect Limesurvey with R ? : Shinyapps

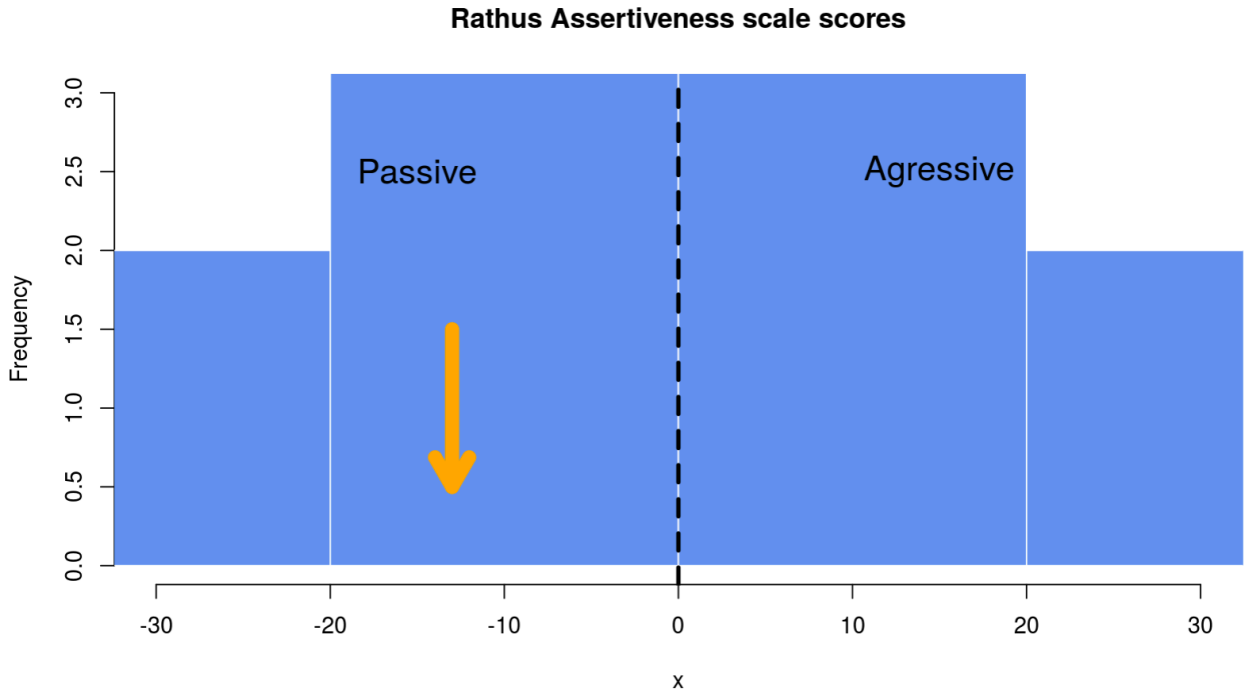
## Assertiveness Rathus Scale

**What is your pseudo ?**

thog1

**Number of observations to view:**

10




```
Your pseudo is thog1. Your score is -13. You are passive.
```

A score toward -90 is a sign than the subject has a lack of assertivity (passive style).

A score around 0 is found in normal assertive behaviour.

https://ouphix.shinyapps.io/affiDeSoi/

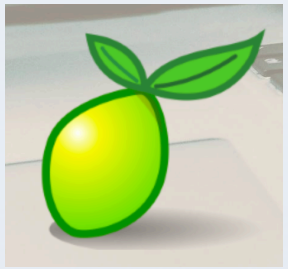
# Python

- A little more fundamental and less user friendly, very used by computer scientists.
- Python has very large possibilities :
  - Collect data during a psychological experiment with pygame
  - Data analysis
  - Code a robot
  - Make a website

# Useful open science tools to do research in psychiatry

**Randomizer.org**

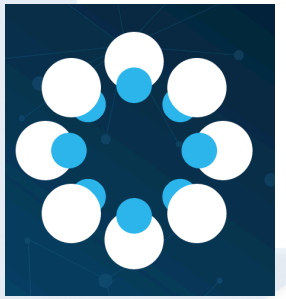
**Collect data**
Limesurvey



RStudio®

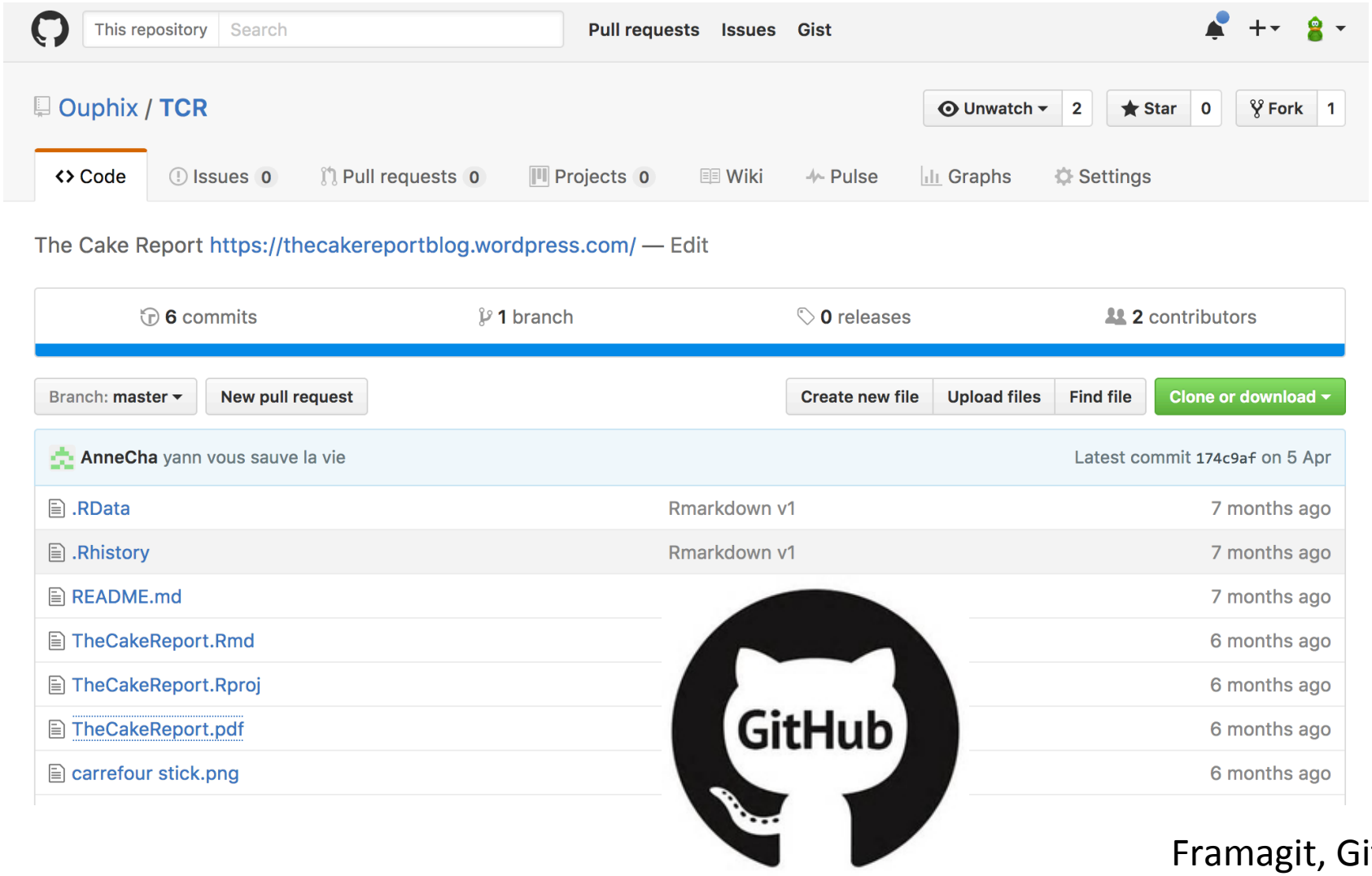
**Analyse data**

**Share data**
OSF and
GitHub



# Git Hub

- Share data

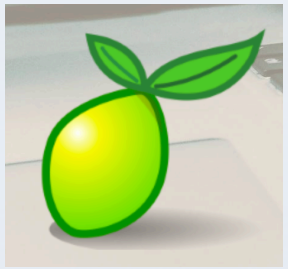



Framagit, Gitlab

# Useful open science tools to do research in psychiatry

**Randomizer.org**

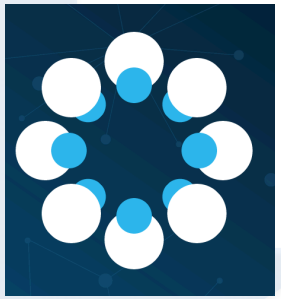
**Collect data**
Limesurvey



RStudio®

**Analyse data**

**Share data**
OSF and
GitHub



equator
network

**Write an article**

# Equator Network

equator network

**Enhancing the QUAlity and Transparency Of health Research**

EQUATOR resources in Portuguese | Spanish

Home | Library | Toolkits | Courses & events | News | Blog | Librarian Network | About us | Contact

**Essential resources for writing and publishing health research**

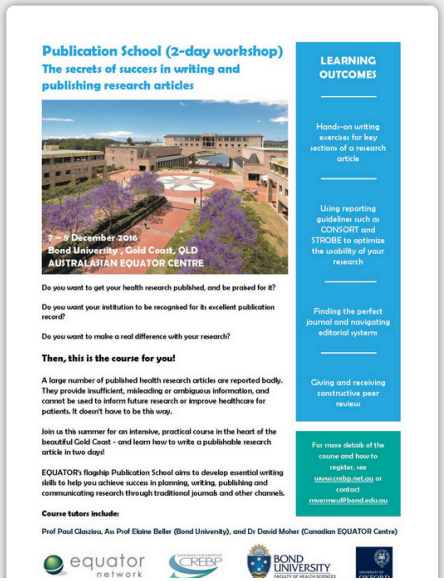
## Library for health research reporting

The Library contains a comprehensive searchable database of reporting guidelines and also links to other resources relevant to research reporting.

- Search for reporting guidelines
- Not sure which reporting guideline to use?
- Reporting guidelines under development
- Visit the library for

## Reporting guidelines for main study types

| | | | |
|---|---|---|---|
| Randomised trials | CONSORT | Extensions | Other |
| Observational studies | STROBE | Extensions | Other |
| Systematic reviews | PRISMA | Extensions | Other |
| Case reports | CARE | | Other |
| Qualitative research | SRQR | COREQ | Other |
| Diagnostic / prognostic studies | STARD | TRIPOD | Other |
| Quality improvement studies | SQUIRE | | Other |
| Economic evaluations | CHEERS | | Other |
| Animal pre-clinical studies | ARRIVE | | Other |
| Study protocols | SPIRIT | PRISMA-P | Other |



# Collaborative writting

Overleaf PROJECT HISTORY & REVISIONS SHARE PDF JOURNALS & SERVICES SIGN UP SIGN IN

Source Rich Text Edit Find § B I π Preview Manual Auto up-to-date and saved

```
1  \documentclass[a4paper]{article}
2
3  %% Language and font encodings
4  \usepackage[english]{babel}
5  \usepackage[utf8x]{inputenc}
6  \usepackage[T1]{fontenc}
7
8  %% Sets page size and margins
9  \usepackage[a4paper,top=3cm,bottom=2cm,left=3cm,right=3cm,marginparwidth=1.75cm]
   {geometry}
10
11 %% Useful packages
12 \usepackage{amsmath}
13 \usepackage{graphicx}
14 \usepackage[colorinlistoftodos]{todonotes}
15 \usepackage[colorlinks=true, allcolors=blue]{hyperref}
16
17 \title{Your Paper}
18 \author{You}
19
20 \begin{document}
21 \maketitle
22
```

# Your Paper

You

December 14, 2017

**Abstract**

Your abstract.

## 1 Introduction

Your introduction goes here! Some examples of commonly used commands and features are listed below, to help you get started. If you have a question, please use the help menu ("?") on the top bar to search for help or ask us a question.

## 2 Some examples to get started

### 2.1 How to add Comments

Comments can be added to your project by clicking on the comment icon in the toolbar above. To reply to a comment, simply click the reply button in the lower right corner of the comment, and you can close them when you're done.

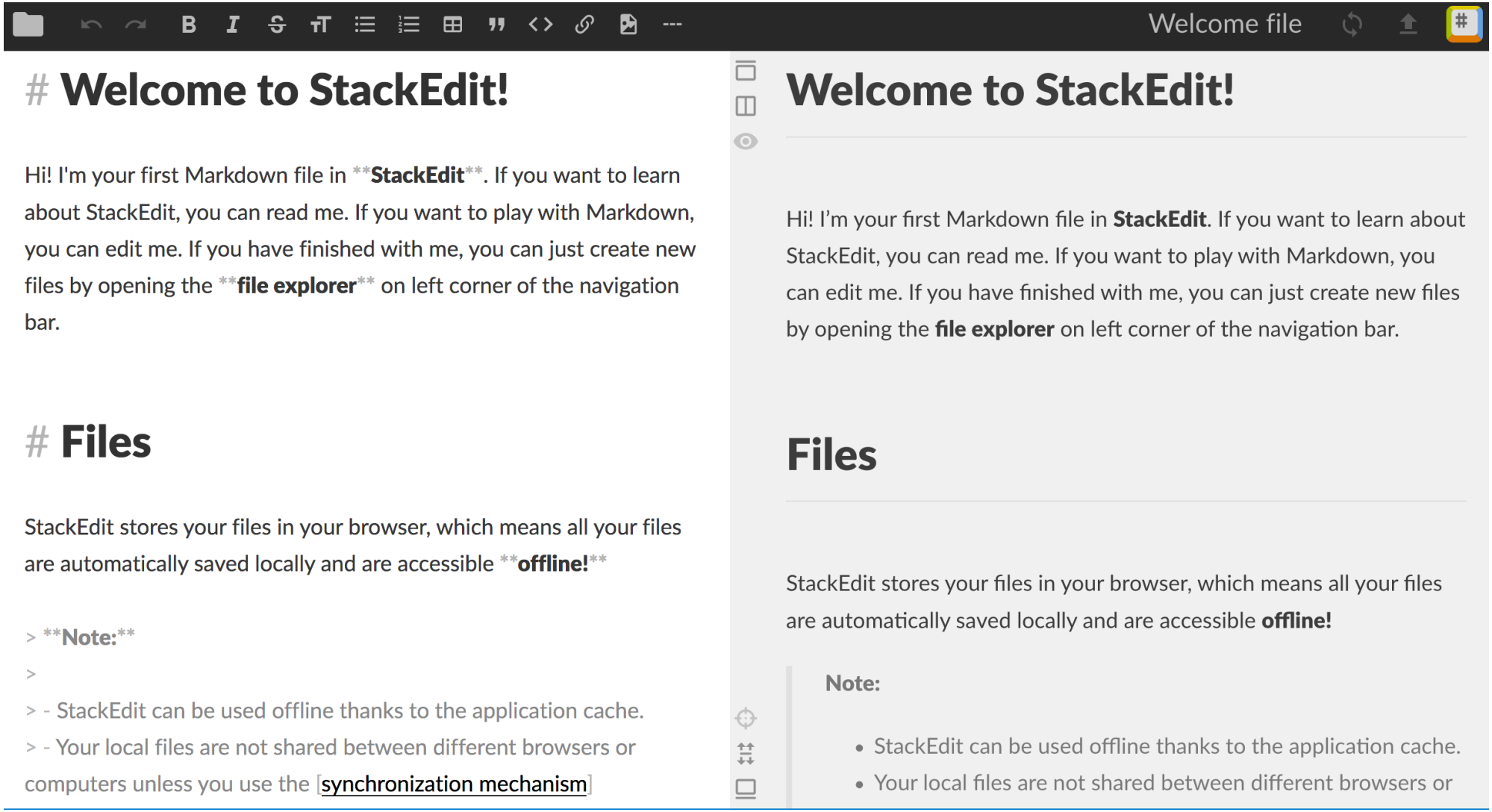
### 2.2 How to include Figures

First you have to upload the image file from your computer using the upload link the project menu. Then use the includegraphics command to include it in your document. Use the figure environment and the caption command to add a number and a caption to your figure. See the code for Figure 1 in this section for an example.

### 2.3 How to add Tables

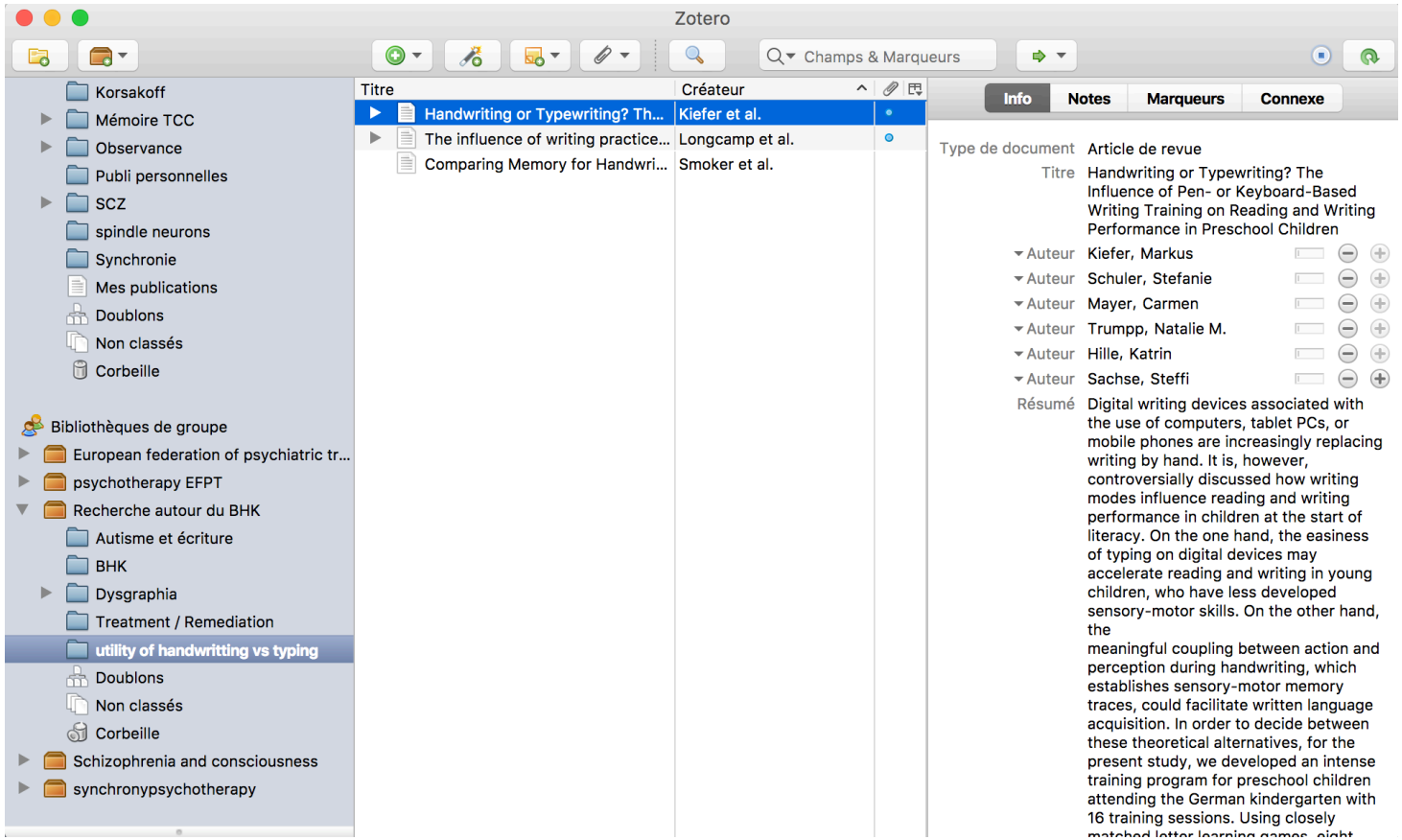
Use the table and tabular commands for basic tables — see Table 1, for example.

Figure 1: This frog was uploaded via the project menu.


Welcome file
# Welcome to StackEdit!
Hi! I'm your first Markdown file in **StackEdit**. If you want to learn about StackEdit, you can read me. If you want to play with Markdown, you can edit me. If you have finished with me, you can just create new files by opening the **file explorer** on left corner of the navigation bar.
# Files
StackEdit stores your files in your browser, which means all your files are automatically saved locally and are accessible **offline!**
> **Note:**
>
> - StackEdit can be used offline thanks to the application cache.
> - Your local files are not shared between different browsers or computers unless you use the [synchronization mechanism]
Welcome to StackEdit!
Hi! I'm your first Markdown file in StackEdit. If you want to learn about StackEdit, you can read me. If you want to play with Markdown, you can edit me. If you have finished with me, you can just create new files by opening the file explorer on left corner of the navigation bar.
Files
StackEdit stores your files in your browser, which means all your files are automatically saved locally and are accessible offline!
Note:
StackEdit can be used offline thanks to the application cache.
Your local files are not shared between different browsers or

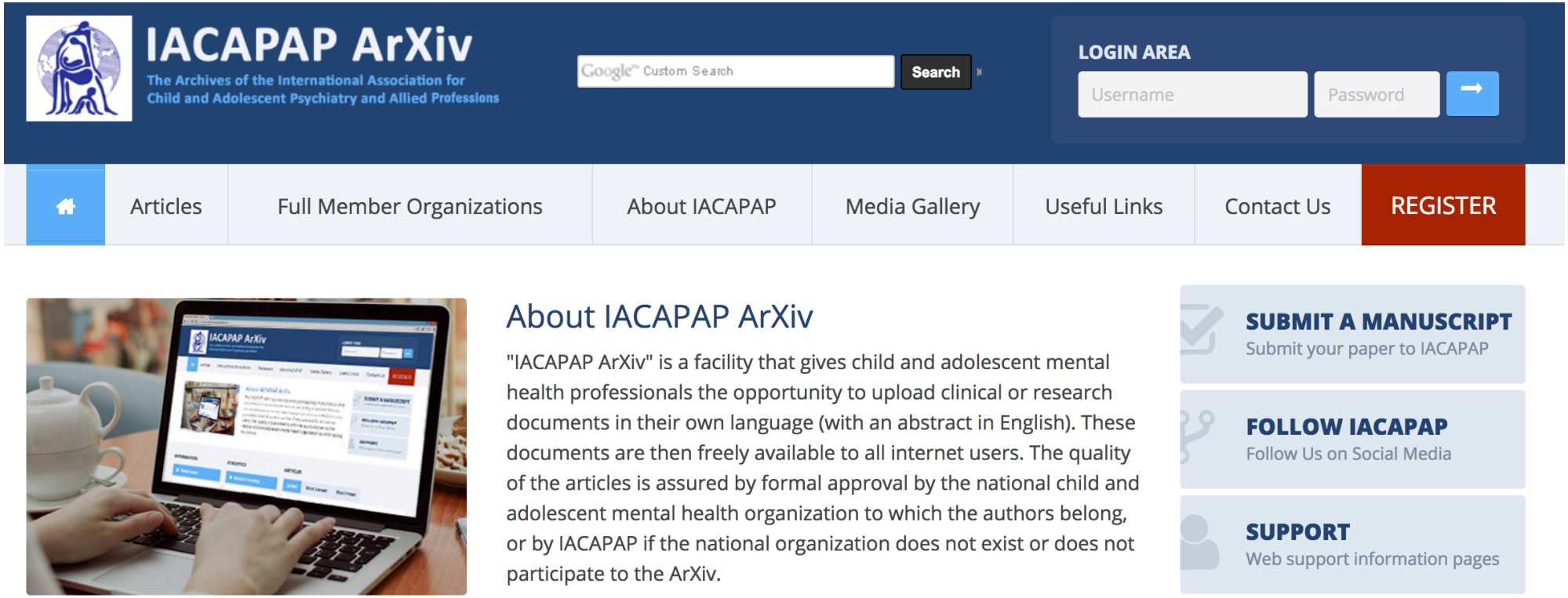
# zotero

- Select, gather, write and share bibliography



# New ways to publish ?

# ArXiv : Preprints

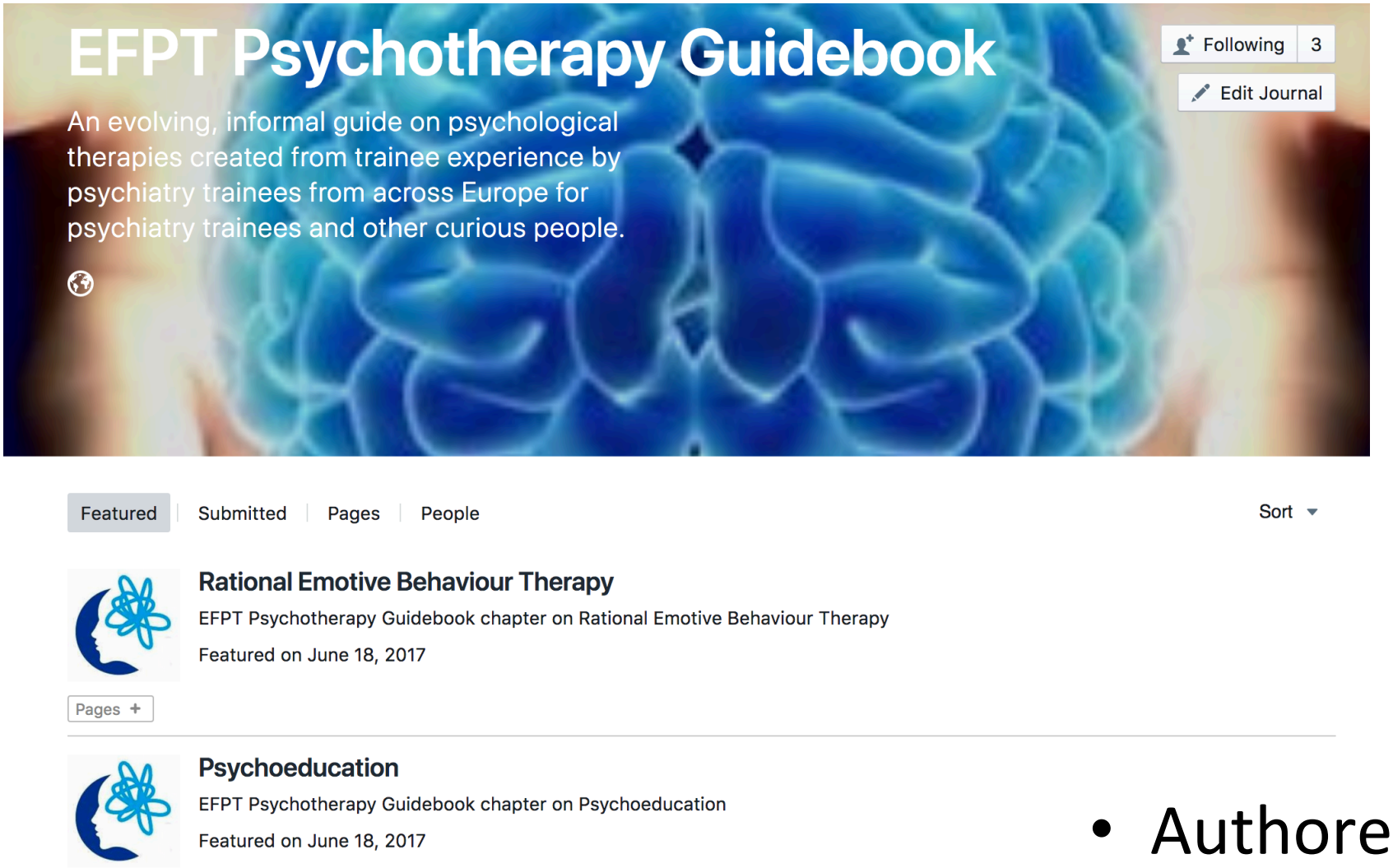



bioRxiv beta

PsyArXiv

SOC ARXIV
Open archive of the social sciences

# https://www.pubpub.org/




- Authorea

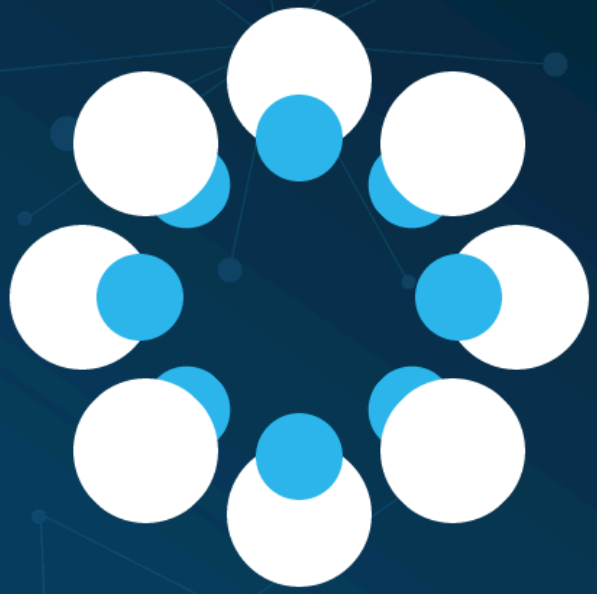
# Several way of practicing Open Access

- Not only Open Access Gold : journal where you need to pay to publish (new publisher's strategy)
- Use Open Repository as HAL in France, Zenodo, etc.
- Don't forget the licences (Creative Commons, etc)

| LICENSES | TERMS |
| --- | --- |
|   | **Attribution**<br>Others can copy, distribute, display, perform and remix your work if they credit your name as requested by you |
|   | **No Derivative Works**<br>Others can only copy, distribute, display or perform verbatim copies of your work |
|   | **Share Alike**<br>Others can distribute your work only under a license identical to the one you have chosen for your work |
|   | **Non-Commercial**<br>Others can copy, distribute, display, perform or remix your work but for non-commercial purposes only. |
|   | |
|   | |

# Open Science Framework

A scholarly commons to connect the entire research cycle

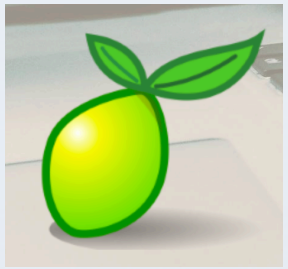


# Share with general public

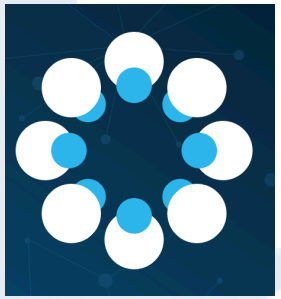
# Useful open science tools to do research in psychiatry

**Randomizer.org**

**Collect data**
Limesurvey




**Analyse data**



**Share data**
OSF and
GitHub



**Write an article**


**Make the knowledge more visible**



WIKIPEDIA
The Free Encyclopedia

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file
Special pages
Permanent link
Page information

# Biscuit

From Wikipedia, the free encyclopedia

*See also: Biscuit (bread) and Cookie*
*For other uses, see Biscuit (disambiguation).*

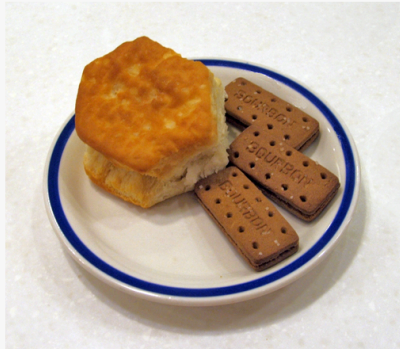
This article **may lack focus or may be about more than one topic**. Please help improve this article, possibly by splitting the article and/or by introducing a disambiguation page, or discuss this issue on the talk page. *(May 2016)*

**Biscuit** is a term used for a diverse variety of baked, commonly flour-based food products. The term is applied to two distinct products in North America and the Commonwealth of Nations and Europe. The North American biscuit is typically a soft, leavened quick bread, and is covered in the article Biscuit (bread). This article covers the other type of biscuit, which is typically hard, flat and unleavened.

**Contents** [hide]



**Biscuit**



American biscuit (left) and one variety of British biscuit (right) – the American biscuit is

https://en.wikipedia.org/

WIKIPEDIA
The Free Encyclopedia

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file
Special pages
Permanent link
Page information

# Biscuit

From Wikipedia, the free encyclopedia

*See also: Biscuit (bread) and Cookie*
*For other uses, see Biscuit (disambiguation).*

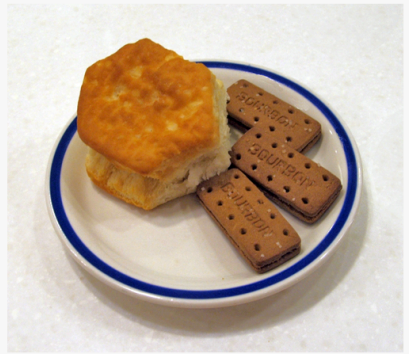
This article **may lack focus or may be about more than one topic**. Please help improve this article, possibly by splitting the article and/or by introducing a disambiguation page, or discuss this issue on the talk page. *(May 2016)*

**Biscuit** is a term used for a diverse variety of baked, commonly flour-based food products. The term is applied to two distinct products in North America and the Commonwealth of Nations and Europe. The North American biscuit is typically a soft, leavened quick bread, and is covered in the article Biscuit (bread). This article covers the other type of biscuit, which is typically hard, flat and unleavened.

**Contents** [hide]



**Biscuit**



American biscuit (left) and one variety of British biscuit (right) – the American biscuit is

https://en.wikipedia.org/


WordPress.org
Search WordPress.org
Showcase
Themes
Plugins
Mobile
Support
Get Involved
About
Blog
Hosting
Download WordPress
Meet WordPress
WordPress is open source software you can use to create a beautiful website, blog, or app.
European Federation of Psychiatric Train...
0
New
Security
Hi, Thomas Gargot
Screen Options
Help
Dashboard
Jetpack
Posts
All Posts
Add New
Categories
Tags
Media
Pages
Comments
Events
Feedback
Add a New Post
Wordfence could not get an API key from the Wordfence scanning servers when it activated. You can try to fix this by going to the Wordfence "options" page and hitting "Save Changes". This will cause Wordfence to retry fetching an API key for you. If you keep seeing this error it usually means your WordPress server can't connect to our scanning servers. You can try asking your WordPress host to allow your WordPress server to connect to noc1.wordfence.com.
Enter title here
Add Media
Add Contact Form
Visual
Text
b
i
link
b-quote
del
ins
img
ul
ol
li
code
more
close tags
contact form
proofread
Publish
Save Draft
Preview
Status: Draft Edit
Visibility: Public Edit
Publish immediately Edit
Publicise: Not Connected
Show
Publish

# Badges



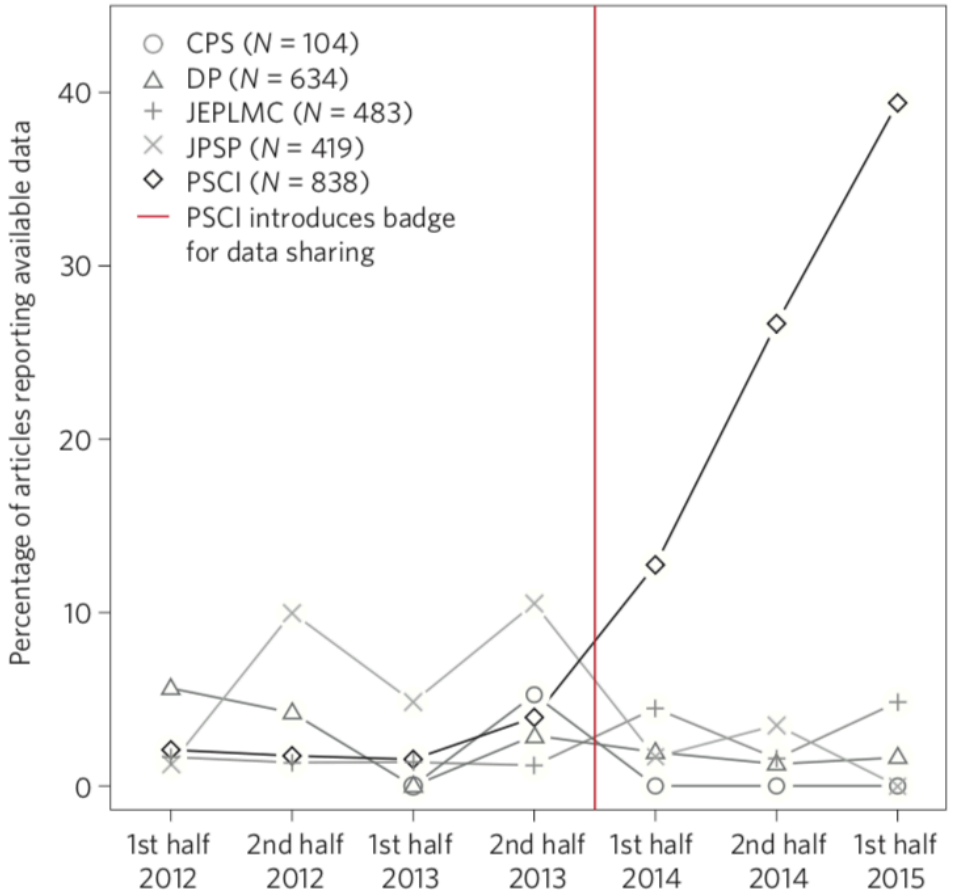
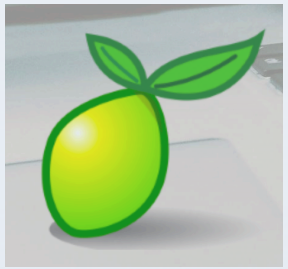




**Figure 2 | The impact of introducing badges for data sharing.** In January 2014, the journal *Psychological Science* (PSCI) introduced badges for articles with open data. Immediately afterwards, the proportion of articles with open data increased steeply, and by October 2015, 38% of articles in *Psychological Science* had open data. For comparison journals (*Clinical Psychological Science* (CPS), *Developmental Psychology* (DP), *Journal of Experimental Psychology: Learning, Memory and Cognition* (JEPLMC) and *Journal of Personality and Social Psychology* (JPSP)) the proportion of articles with open data remained uniformly low. Figure adapted from ref. 75, PLoS.

# Useful open science tools to do research in psychiatry

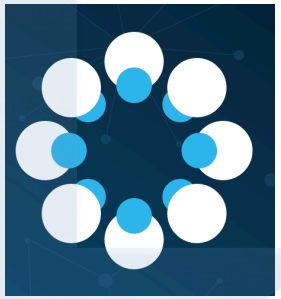
**Randomizer.org**

**Collect data :**
Limesurvey




**Analyse data**

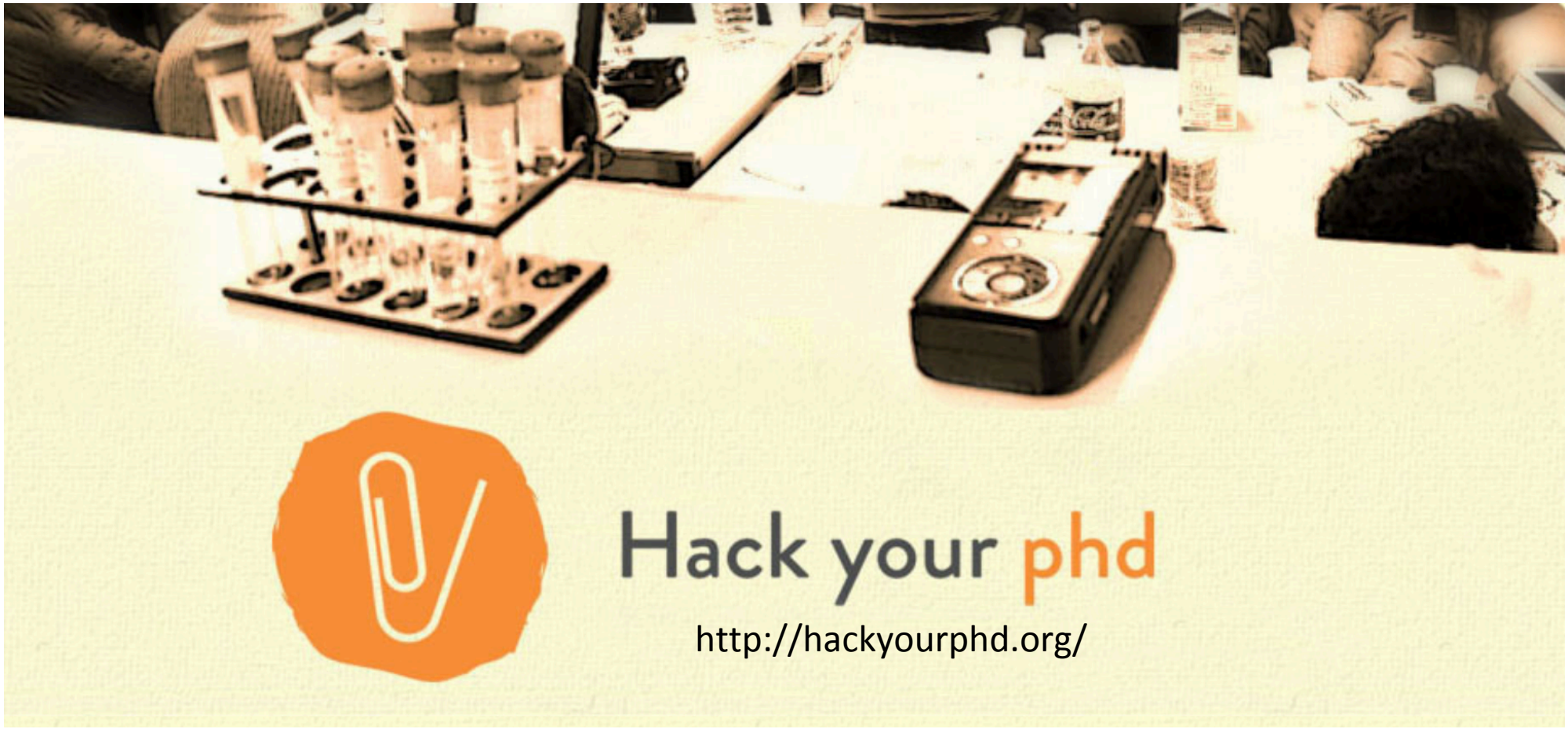
**Share data**
OSF and
GitHub




**Write an article**


**Make the knowledge more visible**

# Thank you for your attention




thomas_gargot@hotmail.com

# Base R
Cheat Sheet

## Getting Help

### Accessing the help files

**?mean**
Get help of a particular function.
**help.search('weighted mean')**
Search the help files for a word or phrase.
**help(package = 'dplyr')**
Find help for a package.

### More about an object

**str(iris)**
Get a summary of an object's structure.
**class(iris)**
Find the class an object belongs to.

## Using Packages

**install.packages('dplyr')**
Download and install a package from CRAN.

**library(dplyr)**
Load the package into the session, making all its functions available to use.

**dplyr::select**
Use a particular function from a package.

**data(iris)**
Load a built-in dataset into the environment.

## Working Directory

**getwd()**
Find the current working directory (where inputs are found and outputs are sent).

**setwd('C://file/path')**
Change the current working directory.

**Use projects in RStudio to set the working directory to the folder you are working in.**

## Vectors

### Creating Vectors

| | | |
|---|---|---|
| c(2, 4, 6) | 2 4 6 | Join elements into a vector |
| 2:6 | 2 3 4 5 6 | An integer sequence |
| seq(2, 3, by=0.5) | 2.0 2.5 3.0 | A complex sequence |
| rep(1:2, times=3) | 1 2 1 2 1 2 | Repeat a vector |
| rep(1:2, each=3) | 1 1 1 2 2 2 | Repeat elements of a vector |

### Vector Functions

**sort(x)**
Return x sorted.
**rev(x)**
Return x reversed.
**table(x)**
See counts of values.
**unique(x)**
See unique values.

### Selecting Vector Elements

#### By Position

| | |
|---|---|
| x[4] | The fourth element. |
| x[-4] | All but the fourth. |
| x[2:4] | Elements two to four. |
| x[-(2:4)] | All elements except two to four. |
| x[c(1, 5)] | Elements one and five. |

#### By Value

| | |
|---|---|
| x[x == 10] | Elements which are equal to 10. |
| x[x < 0] | All elements less than zero. |
| x[x %in% c(1, 2, 5)] | Elements in the set 1, 2, 5. |

#### Named Vectors

| | |
|---|---|
| x['apple'] | Element with name 'apple'. |

## Programming

### For Loop

```
for (variable in sequence){
   Do something
}
```

Example

```
for (i in 1:4){
   j <- i + 10
   print(j)
}
```

### While Loop

```
while (condition){
   Do something
}
```

Example

```
while (i < 5){
   print(i)
   i <- i + 1
}
```

### If Statements

```
if (condition){
   Do something
} else {
   Do something different
}
```

Example

```
if (i > 3){
   print('Yes')
} else {
   print('No')
}
```

### Functions

```
function_name <- function(var){
   Do something
   return(new_variable)
}
```

Example

```
square <- function(x){
   squared <- x*x
   return(squared)
}
```

## Reading and Writing Data

Also see the **readr** package.

| Input | Ouput | Description |
|---|---|---|
| df <- read.table('file.txt') | write.table(df, 'file.txt') | Read and write a delimited text file. |
| df <- read.csv('file.csv') | write.csv(df, 'file.csv') | Read and write a comma separated value file. This is a special case of read.table/ write.table. |
| load('file.RData') | save(df, file = 'file.Rdata') | Read and write an R data file, a file type special for R. |

### Conditions

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| a == b | Are equal | a > b | Greater than | a >= b | Greater than or equal to | is.na(a) | Is missing |
| a != b | Not equal | a < b | Less than | a <= b | Less than or equal to | is.null(a) | Is null |

## Types

Converting between common data types in R. Can always go from a higher value in the table to a lower value.

| | | |
|---|---|---|
| `as.logical` | `TRUE, FALSE, TRUE` | Boolean values (TRUE or FALSE). |
| `as.numeric` | `1, 0, 1` | Integers or floating point numbers. |
| `as.character` | `'1', '0', '1'` | Character strings. Generally preferred to factors. |
| `as.factor` | `'1', '0', '1', levels: '1', '0'` | Character strings with preset levels. Needed for some statistical models. |

## Maths Functions

| | | | |
|---|---|---|---|
| `log(x)` | Natural log. | `sum(x)` | Sum. |
| `exp(x)` | Exponential. | `mean(x)` | Mean. |
| `max(x)` | Largest element. | `median(x)` | Median. |
| `min(x)` | Smallest element. | `quantile(x)` | Percentage quantiles. |
| `round(x, n)` | Round to n decimal places. | `rank(x)` | Rank of elements. |
| `signif(x, n)` | Round to n significant figures. | `var(x)` | The variance. |
| `cor(x, y)` | Correlation. | `sd(x)` | The standard deviation. |

## Variable Assignment

```
> a <- 'apple'
> a
[1] 'apple'
```

## The Environment

| | |
|---|---|
| `ls()` | List all variables in the environment. |
| `rm(x)` | Remove x from the environment. |
| `rm(list = ls())` | Remove all variables from the environment. |

**You can use the environment panel in RStudio to browse variables in your environment.**

## Matrices

`m <- matrix(x, nrow = 3, ncol = 3)`
Create a matrix from x.


`m[2, ]` - Select a row


`m[ , 1]` - Select a column


`m[2, 3]` - Select an element

`t(m)`
Transpose

`m %*% n`
Matrix Multiplication

`solve(m, n)`
Find x in: m * x = n

## Lists

`l <- list(x = 1:5, y = c('a', 'b'))`
A list is a collection of elements which can be of different types.

| `l[[2]]` | `l[1]` | `l$x` | `l['y']` |
|---|---|---|---|
| Second element of l. | New list with only the first element. | Element named x. | New list with only element named y. |

Also see the **dplyr** package.

## Data Frames

`df <- data.frame(x = 1:3, y = c('a', 'b', 'c'))`
A special case of a list where all elements are the same length.

| x | y |
|---|---|
| 1 | a |
| 2 | b |
| 3 | c |

**List subsetting**


`df$x`


`df[[2]]`

*Understanding a data frame*

`View(df)` See the full data frame.

`head(df)` See the first 6 rows.

**Matrix subsetting**


`df[ , 2]`


`df[2, ]`

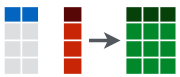

`df[2, 2]`

`nrow(df)`
Number of rows.

`ncol(df)`
Number of columns.

`dim(df)`
Number of columns and rows.

**`cbind`** - Bind columns.



**`rbind`** - Bind rows.

## Strings

Also see the **stringr** package.

| | |
|---|---|
| `paste(x, y, sep = ' ')` | Join multiple vectors together. |
| `paste(x, collapse = ' ')` | Join elements of a vector together. |
| `grep(pattern, x)` | Find regular expression matches in x. |
| `gsub(pattern, replace, x)` | Replace matches in x with a string. |
| `toupper(x)` | Convert to uppercase. |
| `tolower(x)` | Convert to lowercase. |
| `nchar(x)` | Number of characters in a string. |

## Factors

`factor(x)`
Turn a vector into a factor. Can set the levels of the factor and the order.

`cut(x, breaks = 4)`
Turn a numeric vector into a factor by 'cutting' into sections.

## Statistics

`lm(y ~ x, data=df)`
Linear model.

`glm(y ~ x, data=df)`
Generalised linear model.

`summary`
Get more detailed information out a model.

`t.test(x, y)`
Perform a t-test for difference between means.

`pairwise.t.test`
Perform a t-test for paired data.

`prop.test`
Test for a difference between proportions.

`aov`
Analysis of variance.

## Distributions

| | Random Variates | Density Function | Cumulative Distribution | Quantile |
|---|---|---|---|---|
| Normal | `rnorm` | `dnorm` | `pnorm` | `qnorm` |
| Poisson | `rpois` | `dpois` | `ppois` | `qpois` |
| Binomial | `rbinom` | `dbinom` | `pbinom` | `qbinom` |
| Uniform | `runif` | `dunif` | `punif` | `qunif` |

## Plotting

Also see the **ggplot2** package.


`plot(x)`
Values of x in order.


`plot(x, y)`
Values of x against y.


`hist(x)`
Histogram of x.

## Dates

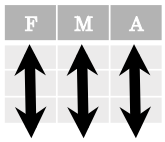
See the **lubridate** package.

# Data Wrangling with pandas Cheat Sheet

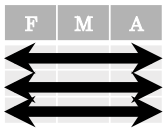
http://pandas.pydata.org

## Tidy Data – A foundation for wrangling in pandas

In a tidy data set:




&




Each **variable** is saved in its own **column**

Each **observation** is saved in its own **row**

Tidy data complements pandas's **vectorized operations**. pandas will automatically preserve observations as you manipulate variables. No other format works as intuitively with pandas.


M * A

## Syntax – Creating DataFrames

| | a | b | c |
|---|---|---|---|
| 1 | 4 | 7 | 10 |
| 2 | 5 | 8 | 11 |
| 3 | 6 | 9 | 12 |

```
df = pd.DataFrame(
        {"a" : [4 ,5, 6],
         "b" : [7, 8, 9],
         "c" : [10, 11, 12]},
        index = [1, 2, 3])
```

Specify values for each column.

```
df = pd.DataFrame(
    [[4, 7, 10],
     [5, 8, 11],
     [6, 9, 12]],
    index=[1, 2, 3],
    columns=['a', 'b', 'c'])
```

Specify values for each row.

| | | a | b | c |
|---|---|---|---|---|
| n | v | | | |
| d | 1 | 4 | 7 | 10 |
| | 2 | 5 | 8 | 11 |
| e | 2 | 6 | 9 | 12 |

```
df = pd.DataFrame(
        {"a" : [4 ,5, 6],
         "b" : [7, 8, 9],
         "c" : [10, 11, 12]},
index = pd.MultiIndex.from_tuples(
        [('d',1),('d',2),('e',2)],
          names=['n','v'])))
```

Create DataFrame with a MultiIndex

## Method Chaining

Most pandas methods return a DataFrame so that another pandas method can be applied to the result. This improves readability of code.

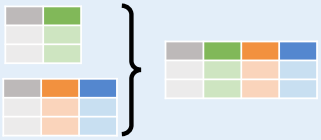
```
df = (pd.melt(df)
        .rename(columns={
                'variable' : 'var',
                'value' : 'val'})
        .query('val >= 200')
    )
```

## Reshaping Data – Change the layout of a data set

**pd.melt(df)**
Gather columns into rows.

**df.pivot(columns='var', values='val')**
Spread rows into columns.

**pd.concat([df1,df2])**
Append rows of DataFrames



**pd.concat([df1,df2], axis=1)**
Append columns of DataFrames

**df.sort_values('mpg')**
Order rows by values of a column (low to high).

**df.sort_values('mpg',ascending=False)**
Order rows by values of a column (high to low).

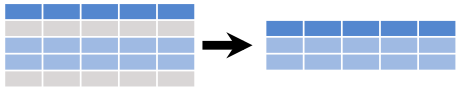
**df.rename(columns = {'y':'year'})**
Rename the columns of a DataFrame

**df.sort_index()**
Sort the index of a DataFrame

**df.reset_index()**
Reset index of DataFrame to row numbers, moving index to columns.

**df.drop(['Length','Height'], axis=1)**
Drop columns from DataFrame

## Subset Observations (Rows)



**df[df.Length > 7]**
Extract rows that meet logical criteria.

**df.drop_duplicates()**
Remove duplicate rows (only considers columns).

**df.head(n)**
Select first n rows.

**df.tail(n)**
Select last n rows.

**df.sample(frac=0.5)**
Randomly select fraction of rows.

**df.sample(n=10)**
Randomly select n rows.

**df.iloc[10:20]**
Select rows by position.

**df.nlargest(n, 'value')**
Select and order top n entries.

**df.nsmallest(n, 'value')**
Select and order bottom n entries.

| Logic in Python (and pandas) | | | |
|---|---|---|---|
| < | Less than | != | Not equal to |
| > | Greater than | df.column.isin(*values*) | Group membership |
| == | Equals | pd.isnull(*obj*) | Is NaN |
| <= | Less than or equals | pd.notnull(*obj*) | Is not NaN |
| >= | Greater than or equals | &,\|,~,^,df.any(),df.all() | Logical and, or, not, xor, any, all |

## Subset Variables (Columns)

**df[['width','length','species']]**
Select multiple columns with specific names.

**df['width']** *or* **df.width**
Select single column with specific name.

**df.filter(regex='*regex*')**
Select columns whose name matches regular expression *regex*.

| regex (Regular Expressions) Examples | |
|---|---|
| '\.' | Matches strings containing a period '.' |
| 'Length$' | Matches strings ending with word 'Length' |
| '^Sepal' | Matches strings beginning with the word 'Sepal' |
| '^x[1-5]$' | Matches strings beginning with 'x' and ending with 1,2,3,4,5 |
| ''^(?!Species$).*' | Matches strings except the string 'Species' |

**df.loc[:,'x2':'x4']**
Select all columns between x2 and x4 (inclusive).

**df.iloc[:,[1,2,5]]**
Select columns in positions 1, 2 and 5 (first column is 0).

**df.loc[df['a'] > 10, ['a','c']]**
Select rows meeting logical condition, and only the specific columns .

# Summarize Data

**`df['w'].value_counts()`**
Count number of rows with each unique value of variable

**`len(df)`**
# of rows in DataFrame.

**`df['w'].nunique()`**
# of distinct values in a column.

**`df.describe()`**
Basic descriptive statistics for each column (or GroupBy)

pandas provides a large set of **summary functions** that operate on different kinds of pandas objects (DataFrame columns, Series, GroupBy, Expanding and Rolling (see below)) and produce single values for each of the groups. When applied to a DataFrame, the result is returned as a pandas Series for each column. Examples:

**`sum()`**
Sum values of each object.

**`count()`**
Count non-NA/null values of each object.

**`median()`**
Median value of each object.

**`quantile([0.25,0.75])`**
Quantiles of each object.

**`apply(function)`**
Apply function to each object.

**`min()`**
Minimum value in each object.

**`max()`**
Maximum value in each object.

**`mean()`**
Mean value of each object.

**`var()`**
Variance of each object.

**`std()`**
Standard deviation of each object.

# Group Data

**`df.groupby(by="col")`**
Return a GroupBy object, grouped by values in column named "col".

**`df.groupby(level="ind")`**
Return a GroupBy object, grouped by values in index level named "ind".

All of the summary functions listed above can be applied to a group. Additional GroupBy functions:

**`size()`**
Size of each group.

**`agg(function)`**
Aggregate group using function.

The examples below can also be applied to groups. In this case, the function is applied on a per-group basis, and the returned vectors are of the length of the original DataFrame.

**`shift(1)`**
Copy with values shifted by 1.

**`rank(method='dense')`**
Ranks with no gaps.

**`rank(method='min')`**
Ranks. Ties get min rank.

**`rank(pct=True)`**
Ranks rescaled to interval [0, 1].

**`rank(method='first')`**
Ranks. Ties go to first value.

**`shift(-1)`**
Copy with values lagged by 1.

**`cumsum()`**
Cumulative sum.

**`cummax()`**
Cumulative max.

**`cummin()`**
Cumulative min.

**`cumprod()`**
Cumulative product.

# Windows

**`df.expanding()`**
Return an Expanding object allowing summary functions to be applied cumulatively.

**`df.rolling(n)`**
Return a Rolling object allowing summary functions to be applied to windows of length n.

# Handling Missing Data

**`df.dropna()`**
Drop rows with any column having NA/null data.

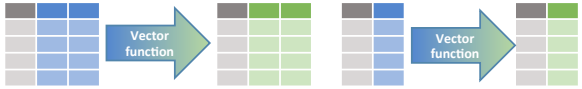
**`df.fillna(value)`**
Replace all NA/null data with value.

# Make New Columns

**`df.assign(Area=lambda df: df.Length*df.Height)`**
Compute and append one or more new columns.

**`df['Volume'] = df.Length*df.Height*df.Depth`**
Add single column.

**`pd.qcut(df.col, n, labels=False)`**
Bin column into n buckets.




pandas provides a large set of **vector functions** that operate on all columns of a DataFrame or a single selected column (a pandas Series). These functions produce vectors of values for each of the columns, or a single Series for the individual Series. Examples:

**`max(axis=1)`**
Element-wise max.

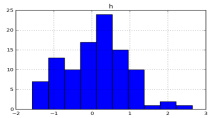
**`clip(lower=-10,upper=10)`**
Trim values at input thresholds

**`min(axis=1)`**
Element-wise min.

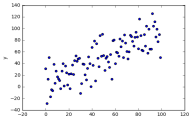
**`abs()`**
Absolute value.

# Plotting

**`df.plot.hist()`**
Histogram for each column



**`df.plot.scatter(x='w',y='h')`**
Scatter chart using pairs of points



# Combine Data Sets

adf

| x1 | x2 |
|---|---|
| A | 1 |
| B | 2 |
| C | 3 |

+

bdf

| x1 | x3 |
|---|---|
| A | T |
| B | F |
| D | T |

=

## Standard Joins

| x1 | x2 | x3 |
|---|---|---|
| A | 1 | T |
| B | 2 | F |
| C | 3 | NaN |

**`pd.merge(adf, bdf, how='left', on='x1')`**
Join matching rows from bdf to adf.

| x1 | x2 | x3 |
|---|---|---|
| A | 1.0 | T |
| B | 2.0 | F |
| D | NaN | T |

**`pd.merge(adf, bdf, how='right', on='x1')`**
Join matching rows from adf to bdf.

| x1 | x2 | x3 |
|---|---|---|
| A | 1 | T |
| B | 2 | F |

**`pd.merge(adf, bdf, how='inner', on='x1')`**
Join data. Retain only rows in both sets.

| x1 | x2 | x3 |
|---|---|---|
| A | 1 | T |
| B | 2 | F |
| C | 3 | NaN |
| D | NaN | T |

**`pd.merge(adf, bdf, how='outer', on='x1')`**
Join data. Retain all values, all rows.

## Filtering Joins

| x1 | x2 |
|---|---|
| A | 1 |
| B | 2 |

**`adf[adf.x1.isin(bdf.x1)]`**
All rows in adf that have a match in bdf.

| x1 | x2 |
|---|---|
| C | 3 |

**`adf[~adf.x1.isin(bdf.x1)]`**
All rows in adf that do not have a match in bdf.

ydf

| x1 | x2 |
|---|---|
| A | 1 |
| B | 2 |
| C | 3 |

+

zdf

| x1 | x2 |
|---|---|
| B | 2 |
| C | 3 |
| D | 4 |

=

## Set-like Operations

| x1 | x2 |
|---|---|
| B | 2 |
| C | 3 |

**`pd.merge(ydf, zdf)`**
Rows that appear in both ydf and zdf (Intersection).

| x1 | x2 |
|---|---|
| A | 1 |
| B | 2 |
| C | 3 |
| D | 4 |

**`pd.merge(ydf, zdf, how='outer')`**
Rows that appear in either or both ydf and zdf (Union).

| x1 | x2 |
|---|---|
| A | 1 |

**`pd.merge(ydf, zdf, how='outer', indicator=True) .query('_merge == "left_only"') .drop(['_merge'],axis=1)`**
Rows that appear in ydf but not zdf (Setdiff).