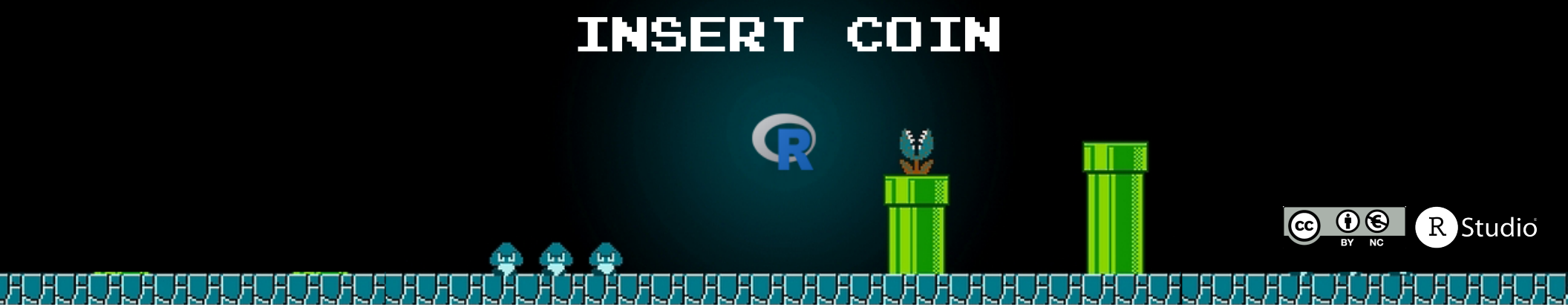
LOADING

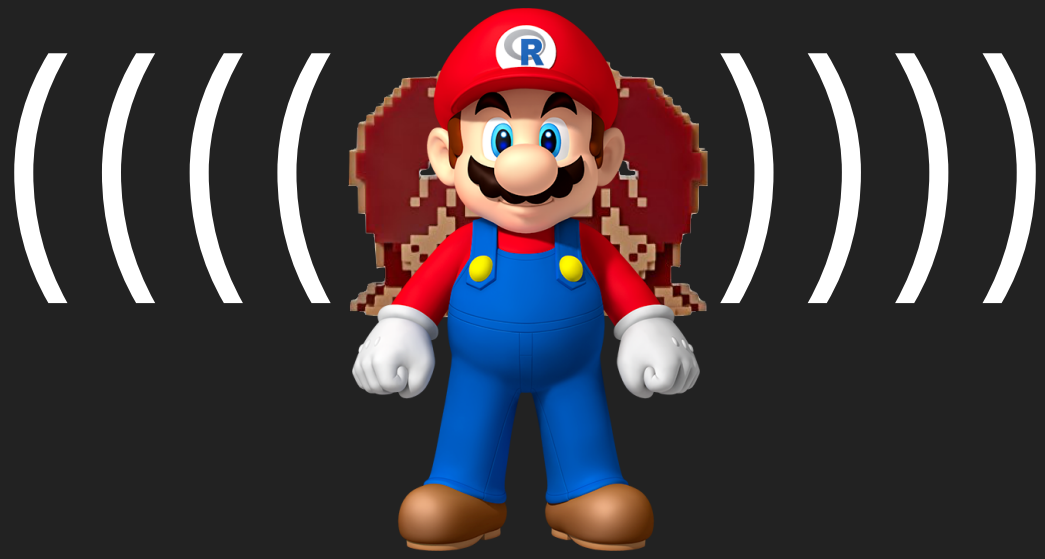
RSTUDIO::CONF 2017

WRITING READABLE CODE WITH PIPES

Bob Rudis • [Master] Chef des Données de Sécurité @ Rapid7



RStudio



RStudio

PIPES : QUICK REFRESHER

```
Package: magrittr
Type: Package
Title: magrittr - a forward-pipe operator for R
Version: 1.0.0
Date: 2014-01-19
Author: Stefan Milton Bache <stefan@stefanbache.dk> and
    Hadley Wickham <h.wickham@gmail.com>
Description: Provides a mechanism for chaining commands with a
    new forward-pipe operator. Ceci n'est pas un pipe.
```

%>%

RStudio

81

export("%>%")

DT
DiagrammeR
GerminaR
HydeNet
Momocs
SciencesPo
analogsea
archivist
binomen
blscrapeR
bpa
ckanr
curlconverter
dat
datadr
ddpcr
dendextend
diffrprojectswidget
diffrprojects
dplyr
dygraphs
emil
forcats
fulltext
gRain
gRbase
geojson
geojsonio
ggvis
gistr
gmailr
gogamer
googleAnalyticsR
highcharter
httping
igraph
jqr
lawn
leaflet
leafletCN
lightsout
magrittr
mason
metacoder
metricsgraphics
modelr
multipanelfigure
nlstimedist
operators
pdp
pixiedust
plotly
poppr
ptstem
purrr
qdap
rbokeh
rdrop2
request
rex
rgbif
rhandsontable
rrr
rscorecard
rslp
rtext
rvest
saeSim
scrubr
simmer
simulator
sparklyr
srvyr
stringr
tableHTML
testthat
text2vec
tidyr
tigris
timevis
tmaptools
vcfR
vegalite
vembedr
visNetwork
webshot
wellknown

What does "*forward chaining*" look like?

```
object %>% operation() → result
```

```
object %>% operation() → result
```

```
object %>% operation() → result
```

```
object %>% operation() → result
```

DOESN'T NEED TO BE THE SAME CLASS/ MODE/TYPE AS OBJECT

RStudio

```
result ← operation(object)
```

```
operation <- function(data, param1, …) {
  data <- do_stuff_with(data)
  data <- do_even_morestuff_with(data)
  data
}
```

"DATA FIRST"

```
operation <- function(data, param1, …) {
  data <- do_stuff_with(data)
  data <- do_even_morestuff_with(data)
  data
}
```

"DATA NOT FIRST"

```
lm(formula, data, subset, weights, na.action,
   method = "qr", model = TRUE, x = FALSE,
   y = FALSE, qr = TRUE, singular.ok = TRUE,
   contrasts = NULL, offset, ...)
```

```
str(mtcars)
```

```
mtcars %>% str()
```

```
mtcars %>% str() # DON'T DO THIS
```

```
mtcars %>% summary() # OR THIS
```

LES RÈGLES DE LA TUYAUTERIE DE hrbrmstr

(hrbrmstr's Rules of Piping)

- The chain should be > 1

(((())))

```
cat(crayon::red(
  paste0(sprintf("I <3 %s",
    map_chr(
      c("dGhpcyBhdWRpZW5jZQ==", "bXkgZmFtaWx5", "Ug=="),
      ~rawToChar(
        openssl::base64_decode(.)))), collapse="\n")))
```


RStudio

(((())))

```r
msg <- c("dGhpcyBhdWRpZW5jZQ==", "bXkgZmFtaWx5", "Ug==")

decoded_msg <- lapply(msg, openssl::base64_decode)
decoded_msg <- sapply(decoded_msg, rawToChar,
                      USE.NAMES = FALSE)

output <- paste0(sprintf("I <3 %s", decoded_msg),
                 collapse="\n")
output <- crayon::red(output)

cat(output)
```

```r
c("dGhpcyBhdWRpZW5jZQ==", "bXkgZmFtaWx5", "Ug==") %>%
  map(openssl::base64_decode) %>%
  map_chr(rawToChar) %>%
  sprintf(fmt = "I <3 %s") %>%
  paste0(collapse = "\n") %>%
  crayon::red() %>%
  cat()
```

I <3 this audience
I <3 my family
I <3 R

Before we go any further...

```
library(tidyverse)
```

```
library(tidyverse)
```

- ggplot2 (data vis)
- dplyr (data manip)
- tidyr (data tidying)
- readr (data import)
- purrr (functional prog)
- tibble (data.frame++)

RStudio

Real-world example

Read in a directory of CSV/JSON/*whatev* into a data frame

```
cowrie_2017-01-03T004539.json.gz
cowrie_2017-01-03T044239.json.gz
cowrie_2017-01-03T065255.json.gz
cowrie_2017-01-03T010619.json.gz
cowrie_2017-01-03T044746.json.gz
cowrie_2017-01-03T073818.json.gz
cowrie_2017-01-03T014616.json.gz
cowrie_2017-01-03T045817.json.gz
```

```
library(ndjson)
library(anytime)

do.call(rbind, lapply(list.files("data/honeypot",
pattern="cowrie.*json.gz", full.names=TRUE), function(x) {
  df <- as.data.frame(stream_in(x))
  df <- df[,c("timestamp", "src_ip", "src_port", "sensor",
"session", "dst_port", "eventid", "username", "password")]
  df$timestamp <- anytime(df$timestamp)
  df
})) -> cowrie_df
```

```r
stream_and_filter <- function(x) {
  df <- as.data.frame(stream_in(x))
  df <- df[,c("timestamp", "src_ip", "src_port", “sensor”,
              "session", "dst_port", "eventid", “username",
              "password")]
  df$timestamp <- anytime(df$timestamp)
  df
}

fils <- list.files("data/honeypot",
                   pattern="cowrie.*json.gz", full.names=TRUE)
cowrie_df_list <- lapply(fils, stream_and_filter)
cowrie_df <- do.call(rbind, cowrie_df_list)
```

```
list.files("data/honeypot", pattern = "cowrie.*json.gz",
                            full.names = TRUE) %>%
  map_df(stream_in) %>%
  select(timestamp, src_ip, src_port, sensor, session,
         dst_port, eventid, username, password) %>%
  mutate(timestamp = anytime(timestamp))
```

LES RÈGLES DE LA TUYAUTERIE DE hrbrmstr

(hrbrmstr's Rules of Piping)

- The chain should be > 1
- A pipe group should be designed to accomplish a unified task

```
list.files("data/honeypot", pattern = "cowrie.*json.gz",
                            full.names = TRUE) %>%
  map_df(stream_in) %>%
  select(timestamp, src_ip, src_port, sensor, session,
         dst_port, eventid, username, password) %>%
  mutate(timestamp = anytime(timestamp))
```

```
filter(mtcars, cyl==6) %>%
  count(gear) %>%
  ggplot(aes(gear, n)) + geom_col()
```

```
count(iris, Species, sort=TRUE) %>%
  mutate(Species=factor(Species, levels=Species)) %>%
  ggplot(aes(Species, n)) + geom_col()
```

```r
filter(satellites, is_active) %>%
  count(agency_full_name, sort=TRUE) %>%
  mutate(ct=scales::comma(n),
         pct=scales::percent(n/sum(n))) %>%
  select(-n) %>%
  print(n=20)
```

```
get_flu_data("national", years=2010:2016) %>%
  mutate(week=from_yr_wk(YEAR, WEEK)) %>%
  gather(age_group, count, starts_with("AGE")) %>%
  ggplot(aes(week, count, group = age_group)) +
  geom_line(aes(color = age_group)) +
  scale_y_continuous(label=scales::comma, limits=c(0,20000)) +
  facet_wrap(~age_group, scales="free") +
```

```
GET("https://www.federalregister.gov/articles/search.csv",
    query=list(
      `conditions[agency_ids][]`=254,
      `conditions[publication_date][gte]`="01/01/2006",
      `conditions[publication_date][lte]`="1/13/2017",
      `conditions[term]`="6039G",
      `conditions[type][]`="NOTICE")) %>%
  stop_for_status() %>%
  content(as="parsed") %>%
  filter(grepl("^Quarterly", title)) -> reg_df
```

```
read_fwf(satellites, na=c("N/A"), col_types=satcat_cols,
         fwf_positions(satcat_cols_start,
                       satcat_cols_end, satcat_col_names)) %>%
  mutate(multiple=(multiple=="M"),
         payload=(payload=="*")) %>%
  left_join(satcat_launch_sources, by="source") %>%
  left_join(satcat_launch_sites, by="launch_site") %>%
  left_join(satcat_op_status, by="op_status_code") %>%
  mutate(is_active=(op_status_code %in%
                    c("+", "P", "B", "S", "X"))) -> satcat
```

LES RÈGLES DE LA TUYAUTERIE DE hrbrmstr

(hrbrmstr's Rules of Piping)

- The chain should be > 1
- A pipe group should be designed to accomplish a unified task
- It's OK to change object class/type/mode

```
list.files("data/honeypot", pattern = "cowrie.*json.gz",
                            full.names = TRUE) %>%
  map_df(stream_in) %>%
  select(timestamp, src_ip, src_port, sensor, session,
         dst_port, eventid, username, password) %>%
  mutate(timestamp = anytime(timestamp))
```

LES RÈGLES DE LA TUYAUTERIE DE hrbrmstr

(hrbrmstr's Rules of Piping)

- The chain should be > 1
- A pipe group should be designed to accomplish a unified task
- It's OK to change object class/type/mode
- Be data-source aware
- Pipe operations should be "atomic"

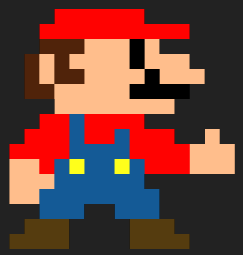
```
list.files("data/honeypot", pattern = "cowrie.*json.gz",
                                  l.names = TRUE) %>%
  map_df(stream_in) %>%
  select(timestamp, sr                  port, sensor, session,
         dst_port, eventid, username, password) %>%
  mutate(timestamp = anytime(timestamp))
```

```r
list.files("data/honeypot", pattern = "cowrie.*json.gz",
                            full.names = TRUE) %>%
  map_df(stream_in) %>%
  select(timestamp, src_ip, src_port, sensor, session,
         dst_port, eventid, username, password) %>%
  mutate(timestamp = anytime(timestamp),
         date = as.Date(timestamp)) %>%
  perform_something_else() %>%
  count(date, username, password, sort = TRUE) %>%
  mutate(creds=sprintf("%s:%s", username, password)) %>%
  ggplot2(aes(creds, n) + geom_col()
```

```
list.files("data/honeypot", pattern = "cowrie.*json.gz",
                            full.names = TRUE) %>%
  map_df(stream_in) %>%
  select(timestamp, src_ip, src_port, sensor, session,
         dst_port, eventid, username, password) %>%
  mutate(timestamp = anytime(timestamp),
         date = as.Date(timestamp)) %>%
💥 perform_something_else() %>%
  count(date, username, password, sort = TRUE) %>%
  mutate(creds=sprintf("%s:%s", username, password)) %>%
📊 ggplot2(aes(creds, n) + geom_col()
```

```r
list.files("data/honeypot", pattern = "cowrie.*json.gz",
                            full.names = TRUE) %>%
  map_df(stream_in) %>%
  select(timestamp, src_ip, src_port, sensor, session,
         dst_port, eventid, username, password) %>%
  mutate(timestamp = anytime(timestamp),
         date = as.Date(timestamp)) %>%
  perform_something_else() %>%
  count(date, username, password, sort = TRUE) %>%
  mutate(creds=sprintf("%s:%s", username, password)) %>%
  ggplot2(aes(creds, n) + geom_col()
```

```r
list.files("data/honeypot", pattern = "cowrie.*json.gz",
                            full.names = TRUE) %>%
  map_df(stream_in) %>%
  select(timestamp, src_ip, src_port, sensor, session,
         dst_port, eventid, username, password) %>%
  mutate(timestamp = anytime(timestamp),
         date = as.Date(timestamp)) -> df
```



https://github.com/be5invis/Iosevka

```
read_html("https://example.com/page.html") %>%
    html_nodes("p") %>%
    html_text() -> some_text
```

```
read_csv("https://example.com/data.csv") %>%
    mutate(...) %>%
    count(...) -> df
```

LES RÈGLES DE LA TUYAUTERIE DE hrbrmstr

(hrbrmstr's Rules of Piping)

- The chain should be > 1
- A pipe group should be designed to accomplish a unified task
- It's OK to change object class/type/mode
- Be data-source aware
- Pipe operations should be "atomic"
- Pipe (briefly) in pipes

LES RÈGLES DE LA TUYAUTERIE DE hrbrmstr

(hrbrmstr's Rules of Piping)

- The chain should be > 1
- A pipe group should be designed to accomplish a unified task
- It's OK to change object class/type/mode
- Be data-source aware
- Pipe operations should be "atomic"
- Pipe (briefly) in pipes

(((())))

```
income <- c("$130,000 - $134,999", "$55,000 - $59,999",
            "$90,000 - $94,999", "$70,000 - 74,999",
            "$10,000 - $14,999", "$25,000 - $29,999",
            "$20,000 - $24,999", "$25,000 - $29,999",
            "$40,000 - $44,999")

sapply(strsplit(income,' - '),
  function(x) mean(as.numeric(gsub('[,\\$]', '', x))))

## [1] 132499.5  57499.5  92499.5  72499.5  12499.5
## [6]  27499.5  22499.5  27499.5  42499.5
```

```
income %>%
  strsplit(" - ") %>%
  map(gsub, pattern="[,\\$]", replacement="") %>%
  map(as.numeric) %>%
  map_dbl(mean)
```

```
income_tmp <- strsplit(" - ")
income_tmp <- map(income_tmp,
            stri_replace_all_regex, "[,\\$]", "")
income_tmp <- map(income_tmp, as.numeric)
mean_incomes <- map_dbl(mean)
```

```
strsplit(income, " - ") %>%
  map_dbl(~gsub("[,\\$]", "", .) %>%
            as.numeric() %>%
            mean())
```

LES RÈGLES DE LA TUYAUTERIE DE hrbrmstr

(hrbrmstr's Rules of Piping)

- The chain should be > 1
- A pipe group should be designed to accomplish a unified task
- It's OK to change object class/type/mode
- Pipe operations should be "atomic"
- Be data-source aware
- Pipe (briefly) in pipes
- Don't be reticent to create new verbs

```r
dollar_to_numeric <- function(x) {
  map(x, ~as.numeric(gsub("[,\\$]", "", .)))
}

compute_means <- function(x) { map_dbl(x, mean) }

strsplit(income, " - ") %>%
  dollar_to_numeric() %>%
  compute_means()
```

Programs are meant to be read by humans and only incidentally for computers to execute.

—Knuth

LES RÈGLES DE LA TUYAUTERIE DE hrbrmstr

(hrbrmstr's Rules of Piping)

- The chain should be > 1
- A pipe group should be designed to accomplish a unified task
- It's OK to change object class/type/mode
- Be data-source aware
- Pipe operations should be "atomic"
- Pipe (briefly) in pipes
- Don't be reticent to create new verbs
- Keep them **short**

W W M D

WHAT WOULD Mario DO?


USE PIPES

Ceci n'est pas une pipe.

HRBRMSTR	RSTUDIO::CONF	SLIDES	TIME
20170113	0 x 5	66	20M

TALK OVER

BOB RUDIS
[MASTER] CHIEF DATA SCIENTIST @ RAPID7
EMAIL: BOB@RUD.IS
BLOG: RUD.IS/B
TWITTER: @HRBRMSTR
GITHUB.COM/HRBRMSTR/RSTUDIOCONF2017

RStudio