Go to your cute local repository

Warning : never make a git pull if your last merge request has not been accepted, or if you believe that your local repo has modifications that are not in the forked repo

If ok: git pull gael master to get the last version of cute

Create a new .R file, using the name of the function to deal with without "fun\_", e.g., *info.R* for the fun\_info function

Go to your cute\_little\_R\_functions local repository

Then: git pull gael master to get the last version of cute\_little\_R\_functions (normally you should not have anything made by you here)

Open the cute\_little\_R\_functions.docx

Copy the whole desired function, for instance, fun\_info, and paste the code into the new *info.R* file.

Open the *info.R* file in RStudio

Reindent:

Mac: Command-A then Command-I

Windows: Ctrl-A then Ctrl-I

Then perform the following check and modifications:

# Arguments of the function: one per line

Example:

**fun\_info <- function(**

**data,**

**n = NULL,**

**warn.print = TRUE**

**){**

And not

**fun\_info <- function(data, n = NULL, warn.print = TRUE){**

# In the description section (large part with lines starting by #)

## Check that all these categ are here

*# AIM*

*# WARNINGS*

*# ARGUMENTS*

*# RETURN*

*# REQUIRED PACKAGES*

*# REQUIRED FUNCTIONS FROM THE cute PACKAGE*

*# EXAMPLE*

*# DEBUGGING*

Otherwise, signal it. For instance, if Warnings is not present, add # WARNINGS with 4 empty liens before, and 4 after.

*# AIM*

*# Provide a broad description of an object*

*# WARNINGS*

*# ARGUMENTS*

*# data: object to analyse*

*# n: positive integer value indicating the n first number of elements to display per compartment of the output list (i.e., head(..., n)). Write NULL to return all the elements. Does not apply for the $STRUCTURE compartment output*

*# warn.print: logical. Print potential warnings at the end of the execution? If FALSE the warning messages are added in the output list as an additional compartment (or NULL if no message).*

## Each line must start by capital letter, except arguments names in argument section

## In the argument section, update all argument description, saying, character vector, etc, as in fun\_gg\_boxplot

## In the return section, update all as in fun\_gg\_boxplot

## In the example section, 1) copy-paste all the examples in a new file, named *examples\_info.R* for *info.R*, 2) keep a single example in the example section.

## In the debbuging section, keep only one line

# Check that all these comment categs are in the code of the function, and check also the "must" written in red below

*# function name*

**#** *must have* **arg.user.setting** *inside*

*# end function name*

*# required function checking*

*# see boxplot*

*# end required function checking*

*# reserved words (to avoid bugs)*

*# end reserved words (to avoid bugs)*

*# arg with no default values*

*# see boxplot*

*# end arg with no default values*

*# argument primary checking*

*# end argument primary checking*

*# second round of checking and data preparation*

*# management of NA arguments*

**#** *must have* **arg.user.setting** *inside*

*# end management of NA arguments*

*# management of NULL arguments*

*# end management of NULL arguments*

*# code that protects set.seed() in the global environment*

*# end code that protects set.seed() in the global environment*

*# warning initiation*

*# end warning initiation*

*# other checkings*

*# end other checkings*

*# reserved word checking*

*# end reserved word checking*

*# end second round of checking and data preparation*

*# package checking*

*# end package checking*

*# main code*

*# output*

*# must have* **if(warn.print == TRUE & ! is.null(warn)){** *inside*

*# end output*

*# end main code*

Otherwise, signal it. For instance, if *# function name ... # end function name*

is not present, add it with 4 empty liens before, and 4 after:

*# function name*

*# end function name*

*examples\_info.R*

**# 4) example sheet as in fun\_gg\_boxplot**

**# 5) test the function with debugging\_tools\_for\_r\_dev**

**# 6) use fun\_test()**

**# 7) check all(, na.rm = TRUE) and any(, na.rm = TRUE), notably in if() that does not like NA result**

**# 8) write at the beginning of the function:**

*# todo list check OK*

*# Check r\_debugging\_tools-v1.4.R OK*

*# Check fun\_test() 20201107 (see cute\_checks.docx) OK*

*# example sheet OK*

*# check all and any OK*

*# -> clear to go Apollo*

*# -> transferred into the cute package*

*# Do not modify this function in cute\_little\_R\_function anymore*

**# 9) transfer the clean function into the cute repo**

**###### WARNING:**

**# 1) do not forget to remove all the fun\_ in the cute repo**

**# 2) call the cute package before the cute\_little\_r\_toolbow, because some functions are now only in the cute package**