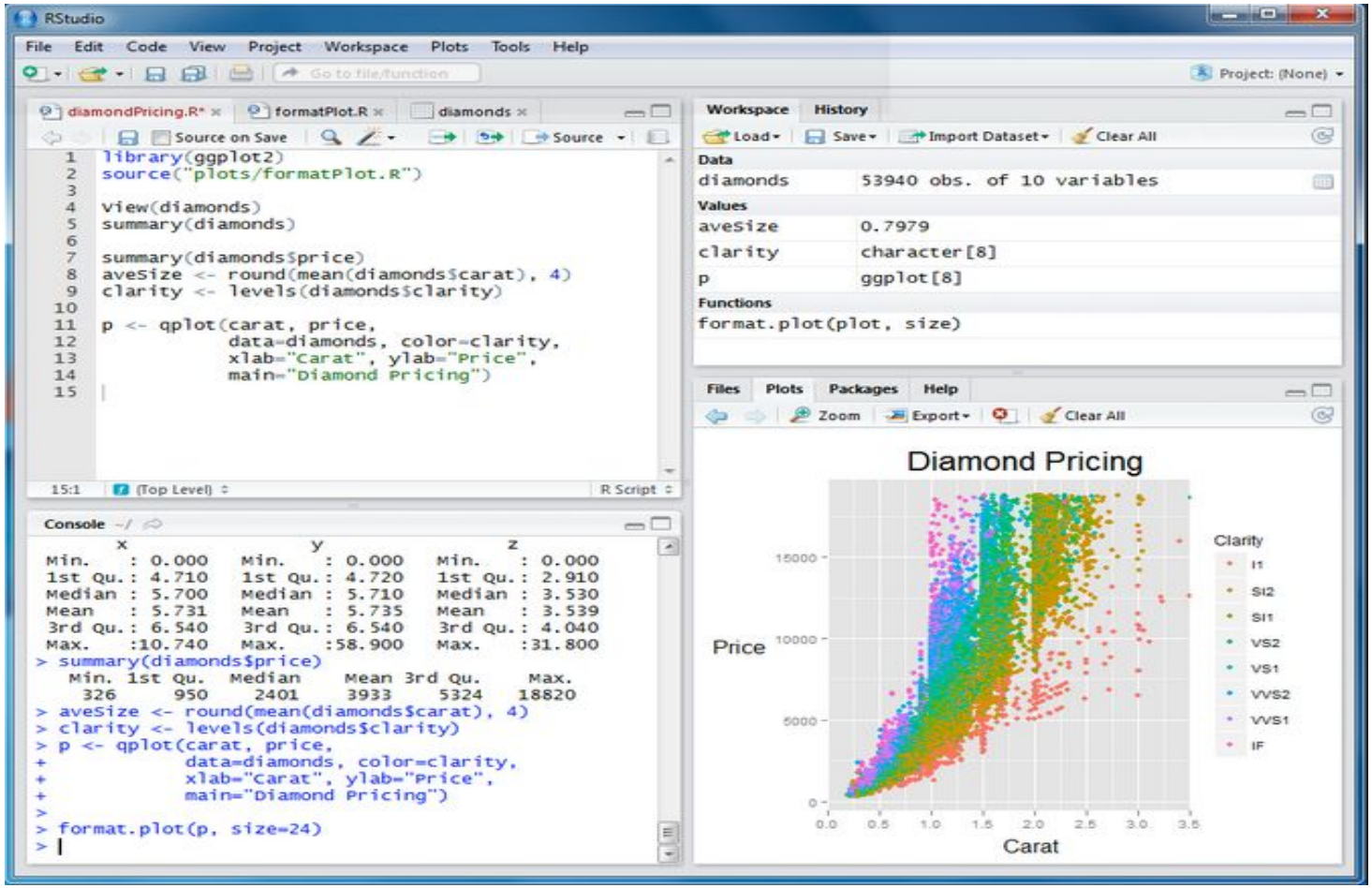
RStudio addins: a shortcut to your favourite functionalities

Andrea Melloncelli
andrea.melloncelli@quantide.com

sponsored by QUANTIDE knowledge from data

R & RStudio



sponsored by QUANTIDE knowledge from data

R Package

Definition:

Packages are collections of **R** functions, data, and compiled code in a well-**defined** format. The directory where **packages** are stored is called the library. **R** comes with a standard set of **packages**. Others are available for download and installation. Once installed, they have to be loaded into the session to be used.

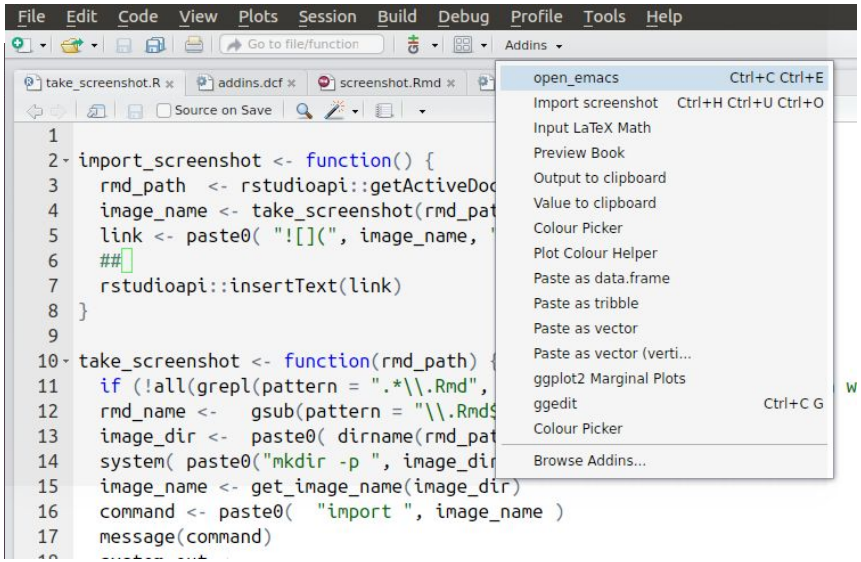
How to install packages:

```
# install a package
install.packages(“devtools”)


# install from github (linux)
devtools::install_github("r-lab-milano/shinyGadgetExample")
```

Source

Addin

Definition: it is a binding between an R function and its entry in the Addin menu.

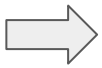


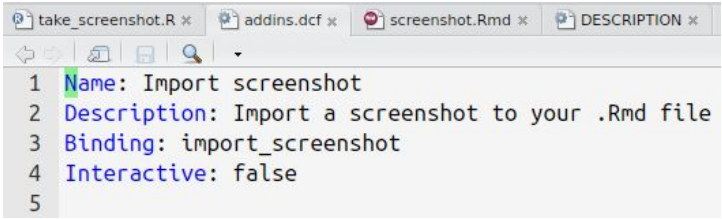
Goals:

1. Run your R function from RStudio GUI with your mouse.
2. Give it a keyboard shortcuts.

All you need is to create the file:

`<package>/inst/rstudio/addins.dcf`





[RStudio Addin]

Make an Addin

1. Create a package
   a. File > New project
   b. new directory
   c. R package
   d. (“create a git repository” if you know and have git, otherwise unflag it)
   e. Create project
2. create a new file and write down the code you need encapsulated in functions
3. save it as ./R/main.R
4. add a file and save it as ./inst/rstudio/addins.dcf
5. fill it as explained before
6. Press Ctrl + Shift + B and install the new package
7. Check in the addin menu if the new entry has appeared

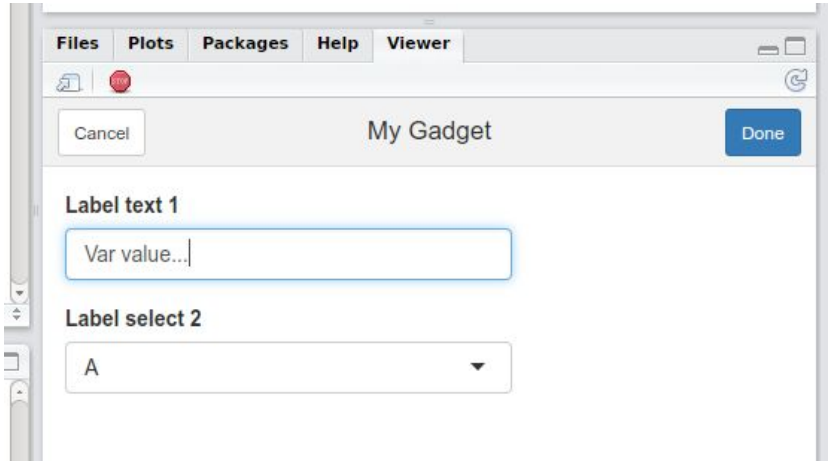
Gadgets

Definition: it is a small shiny Grafical User Interface (GUI) that helps the RStudio user.

```
library(shiny)
library(miniUI)

myGadgetFunc <- function() {

  ui <- miniPage(
    gadgetTitleBar("My Gadget"),
    miniContentPanel(
      # Define layout, inputs, outputs
    )
  )

  server <- function(input, output, session) {
    # Define reactive expressions, outputs, etc.

    # When the Done button is clicked, return a
value
    observeEvent(input$done, {
      returnValue <- ...
      stopApp(returnValue)
    })
  }

  runGadget(ui, server)
}
```

Goals:

1. Give a quick GUI to your R functions
2. Guide the user through Visual information



Shiny Gadgets Example

RStudio API

The rstudioapi package allows you to interact with RStudio directly from the R code you write.

Usage:

get the full path of the active file:

```
current_file_path  <- rstudioapi::getActiveDocumentContext()$path
```

or insert a string in the cursor position:

```
rstudioapi::insertText("your clever text here")
```

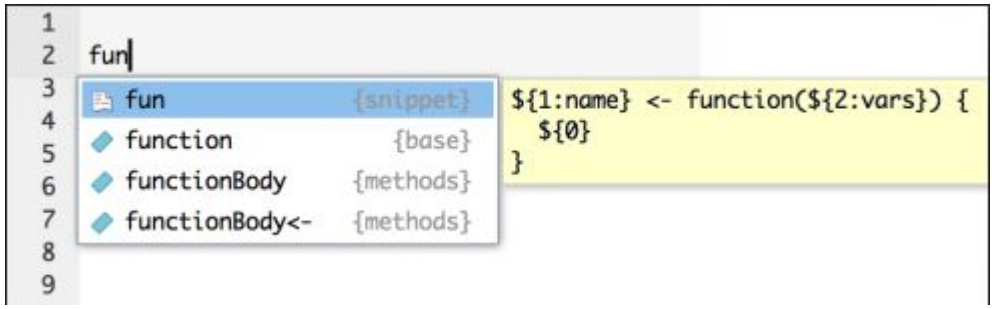
Installation:

```
install.packages("rstudioapi", type = "source")
```

Rstudioapi

Snippet

Definition: it is a short text that RStudio expands into a template text.



Goals:

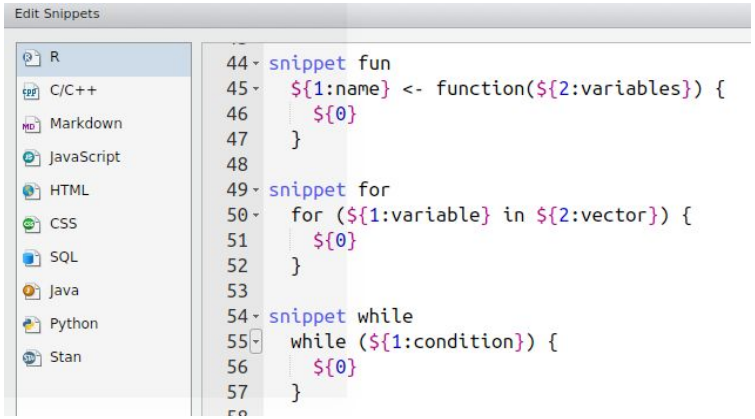
1. Write less
2. Move the cursor in the right place with a “tab”

Use snippets:

1. write the snippet name: `fun`
2. press Tab or Shift+Tab
3. then Tab to move

Customize snippets:

Edit *Snippets* button in *Global Options -> Code*



RStudio Snippets Code Snippets

Snippet examples

```
snippet shiny
    library(shiny)
    ui <- fluidPage(
        ${1:ui}
    )

    server <- function(input, output, session){
      ${2:server}
    }
    shinyApp(ui = ui, server = server)
```

```
snippet gg
    ggplot(${1:data}, aes(${2:aes})) +
      geom_${3:geom}()
```

Snippet example - source