

# INTRODUCTION SHINY APPS

Aurélien Nicosia,  
EIOM, ULaval

A series of horizontal lines in teal and light blue colors, stacked and slightly offset, extending from the right edge of the slide towards the center.

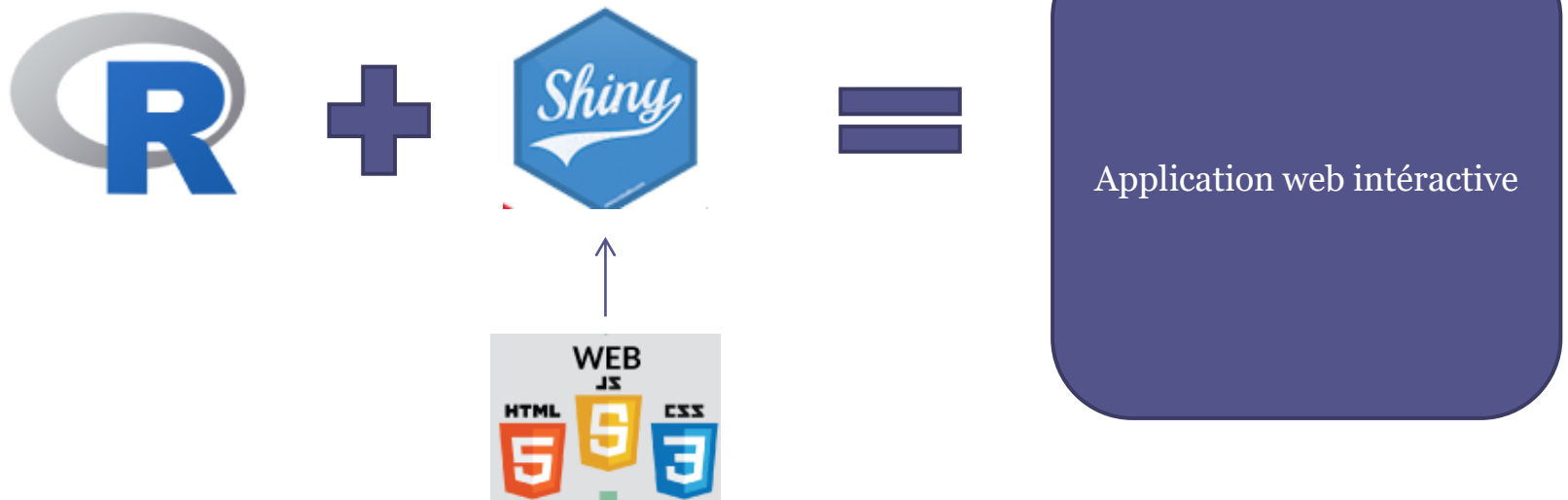
# Plan de la formation

- Introduction
- Qu'est-ce qu'une Shiny app?
- Architecture de l'application
- Inputs, Outputs et réactivité
- Apparence de l'application
- Partager son app

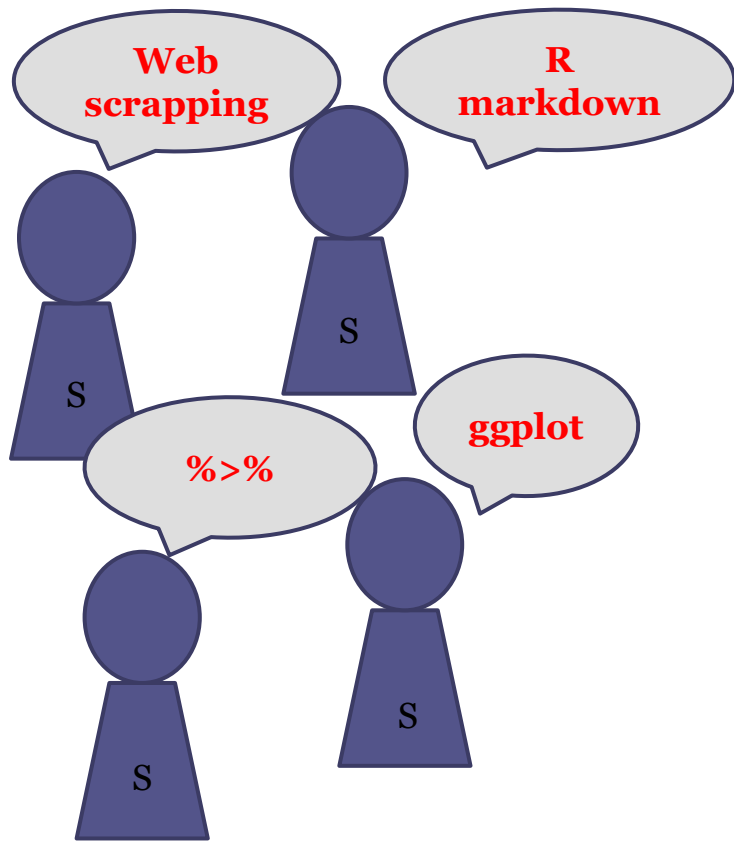
# Qu'est-ce qu'une Shiny app?

- **Un environnement d'application web pour R**
- Transformer vos analyses en application web réactives.
- Aucune connaissance de HTML, CSS, ou JavaScript n'est requise.
- Partageable avec des gens qui n'ont pas R.
- Shiny est un projet de R studio

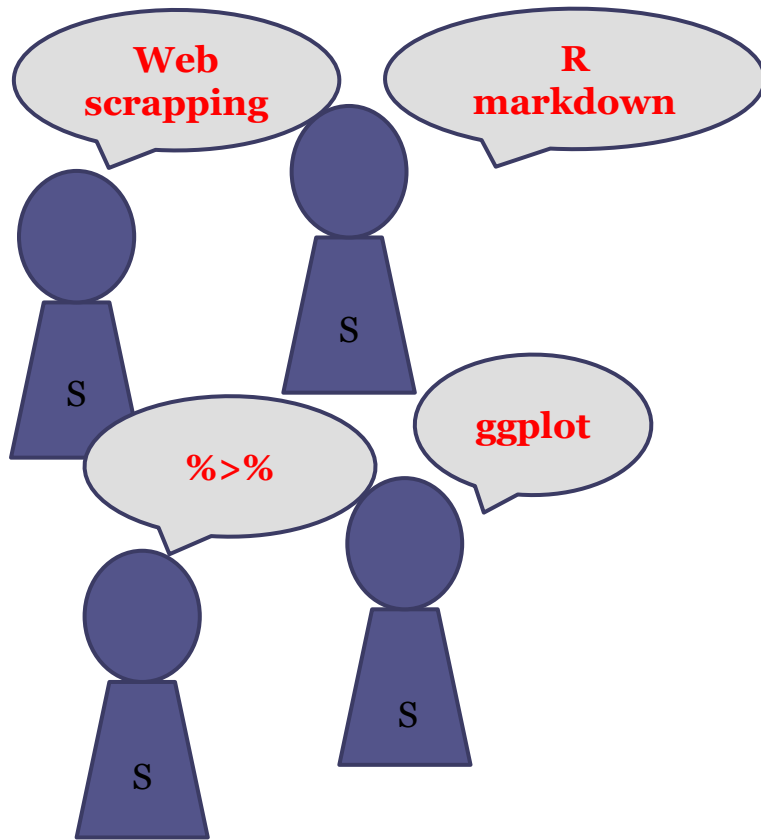
# Les shiny App



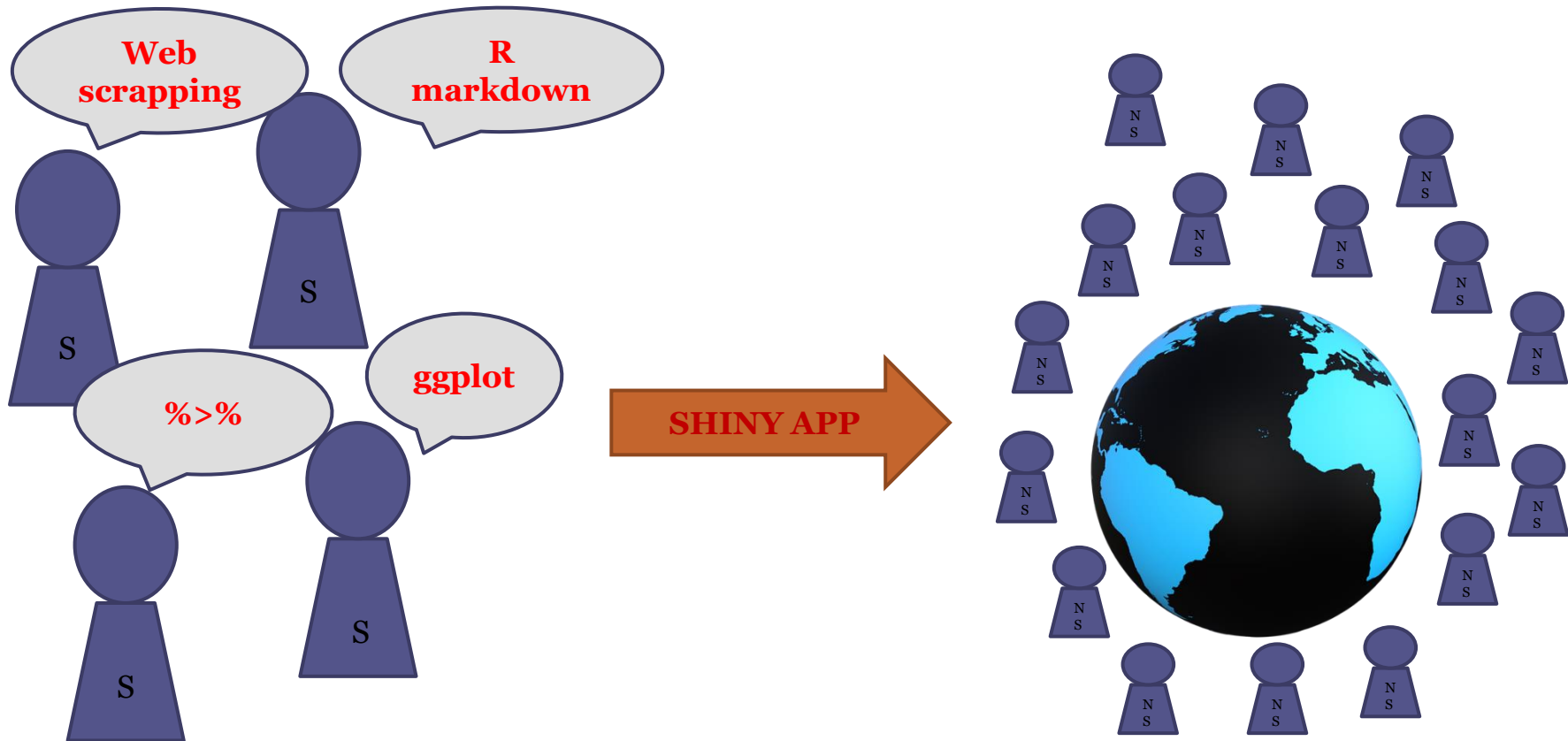
# Intérêt des shiny



# Intérêt des shiny



# Intérêt des shiny



En un mot

**PARTAGER**



# Des exemples sur le web

- [R studio shiny contest](#)

# ShinyDashboard

- Facile d'utiliser shiny pour faire des dashboards.

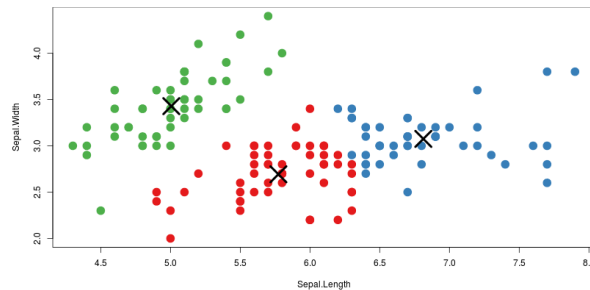
## Shiny

Iris k-means clustering

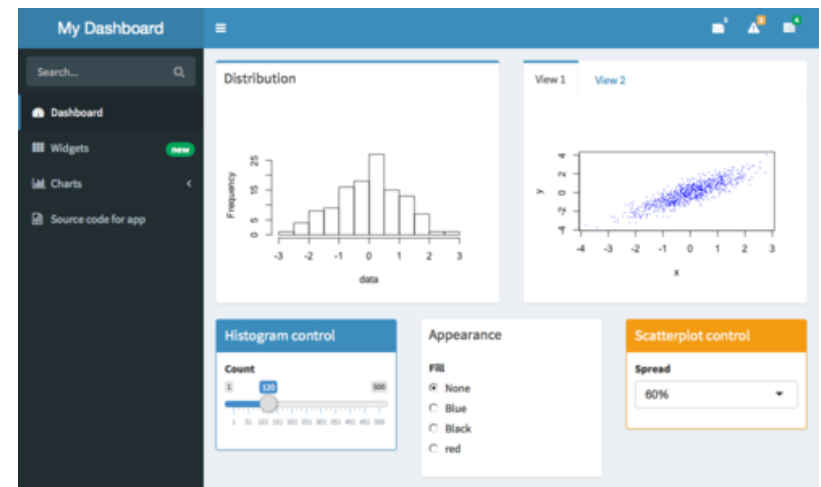
X Variable  
Sepal.Length

Y Variable  
Sepal.Width

Cluster count  
3



## ShinyDashboard



# Architecture de l'app



Nom de l'app

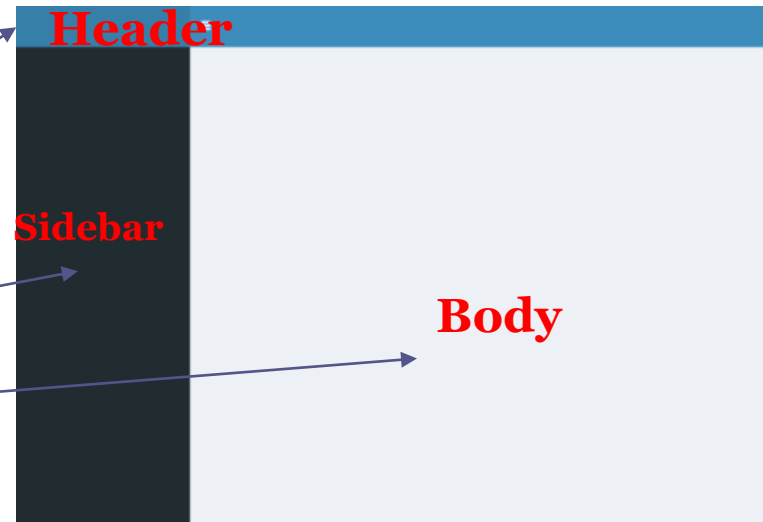
- ui.R
- server.R
- Global.R
- DESCRIPTION
- README
- Autres\_fichiers
- www

- ui.R, server.R et global.R doivent toujours s'appeler de cette façon.
- DESCRIPTION ET README pour aider au partage.
- www est un dossier contenant les images, base de données...

# User-interface

- Le script ui.R user-interface contrôle l'agencement et l'apparence de votre.
- Choix des couleurs, menus, sous-menus, titre...

```
## ui.R ##  
library(shinydashboard)  
  
dashboardPage(  
  dashboardHeader(),  
  dashboardSidebar(),  
  dashboardBody()  
)
```



# Ui.R

```
library(shinydashboard)
```

```
dashboardPage(  
  dashboardHeader(  
    ... code .....,  
    ... code .....,  
  ),  
  dashboardSidebar(  
    ... code .....,  
    ... code .....,  
  ),  
  dashboardBody(  
    ... code .....,  
    ... code .....,  
  )  
)
```

Succession de code telle que du texte, des boîtes, des images, tables...etc. qui va créer l'app.

- Il faut séparer les lignes par des virgules
- Shiny comprend chaque ligne comme du code HTML sans que ça n'en soit!


# Ui.R

- Il existe plusieurs formats de texte que l'on peut inclure dans l'app.
- Header: Titre App + messages, notifications et tâches.
- Sidebard: logo + lien web ainsi que des menus et sous-menus, avec icônes.
- Le Body consiste en une succession de boite rangée en ligne ou colonne.

# Le server

- Le script server.R contient les instructions que votre ordinateur à besoin pour construire votre app.

```
server <- function(input, output) { }
```



Inclure tous les éléments dans la construction de votre Shiny app, ex: création d'un graphique...

# Inputs et Outputs

- Le User-Interface est principalement un rassemblement d'inputs et d'output.
  - Input: entrée faite par l'utilisateur
  - Output: Sortir pour l'utilisateur.
- 
- De façon générale, les inputs sont définies dans le ui et les outputs dans le serveur.



# Inputs et Outputs

- Un input est déclaré dans le ui.R sous un label *inputId*, ensuite il est utilisé dans le server avec `input$inputId`.
- Un output est créer dans le server avec un label *outputId* avec `output$outputId`, on le fait afficher dans le ui avec son *outputId*

# Inputs et Outputs

Ui.R

Définit input 'x'

Affiche output 'z'

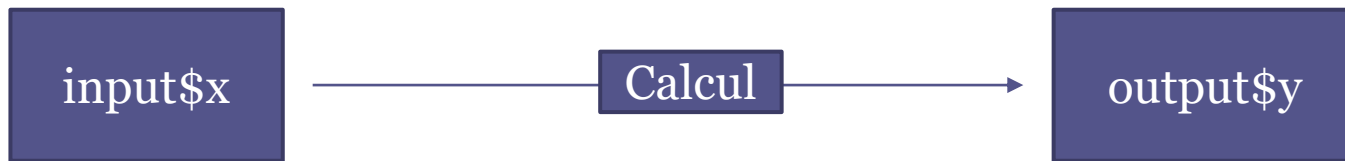
Server.R

Utilise avec input\$x

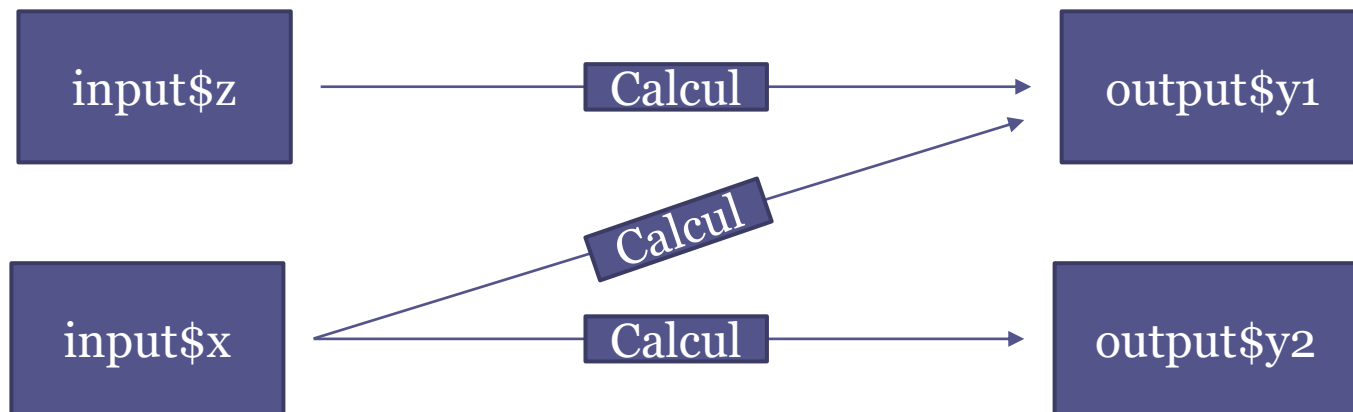
Définit output\$z

# Inputs, Outputs

- Avec l'input nommé "x" on crée l'output "y":



- L'input "x" et "z" crée l'output "y1" et "y2":



# Les différents inputs

- Dans le ui, on définit l'input ID par:

```
fun_input_ui(inputId = 'ID' , ....)
```

La fonction `fun_input_ui` dépend du type d'input que vous souhaitez

# Les différents inputs

## Button

Action

`actionButton()`

## Single checkbox

☒ Choice A

`checkboxInput()`

## Checkbox group

☒ Choice 1  
☐ Choice 2  
☐ Choice 3

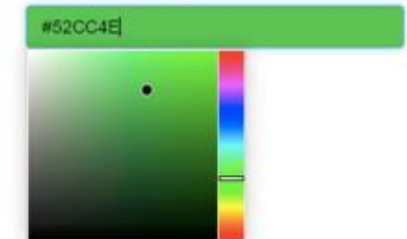
`checkboxGroupInput()`

## Date input

2014-01-01

`dateInput()`

## Colour input



`colourpicker::colourInput()`

## Date range

2014-01-24 to 2014-01-24

`dateRangeInput()`

## File input

Choose File No file chosen

`fileInput()`

## Numeric input

1

`numericInput()`

## Password Input

.....

`passwordInput()`

## Radio buttons

☒ Choice 1  
☐ Choice 2  
☐ Choice 3

`radioButtons()`

## Select box

Choice 1

`selectInput()`

## Sliders



`sliderInput()`

## Text input

Enter text...

`textInput()`

## Text area

Multiple lines  
of text

`textAreaInput()`

# Les différents outputs

- On définit un output dans le server.R avec:

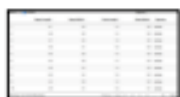
```
output$ID <- fun_output_server({  
  .... création output ....  
})
```

- On l'affiche grâce au ui.R avec: `fun_output_ui("ID")`
- Fun\_output\_server et ui dépendent du type d'output que l'on veut!

# Les différents outputs

Server.R

ui.R



**DT::renderDataTable**(expr,  
options, callback, escape,  
env, quoted)

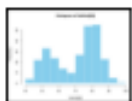


**dataTableOutput**(outputId, icon, ...)



**renderImage**(expr, env, quoted, deleteFile)

**imageOutput**(outputId, width, height, click,  
dblclick, hover, hoverDelay, hoverDelayType,  
brush, clickId, hoverId, inline)



**renderPlot**(expr, width, height, res, ..., env,  
quoted, func)

**plotOutput**(outputId, width, height, click,  
dblclick, hover, hoverDelay, hoverDelayType,  
brush, clickId, hoverId, inline)

Source: local data frame [3 rows x 2 variables]  
# A tibble: 3 x 2  
# Sepal.Length Petal.Length  
# <dbl> <dbl>  
1 5.1 0.2  
2 4.9 0.1  
3 5.4 1.7

**renderPrint**(expr, env, quoted, func,  
width)

**verbatimTextOutput**(outputId)

**renderTable**(expr, ..., env, quoted, func)

**tableOutput**(outputId)

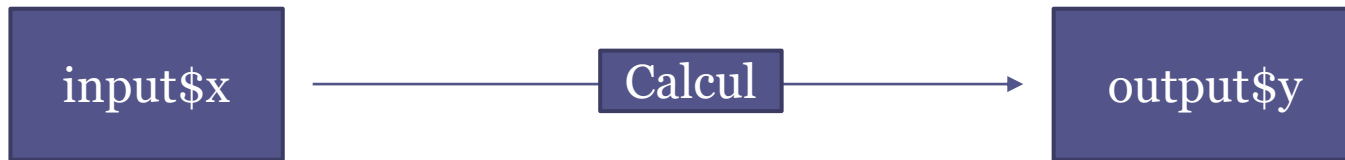
foo

**renderText**(expr, env, quoted, func)

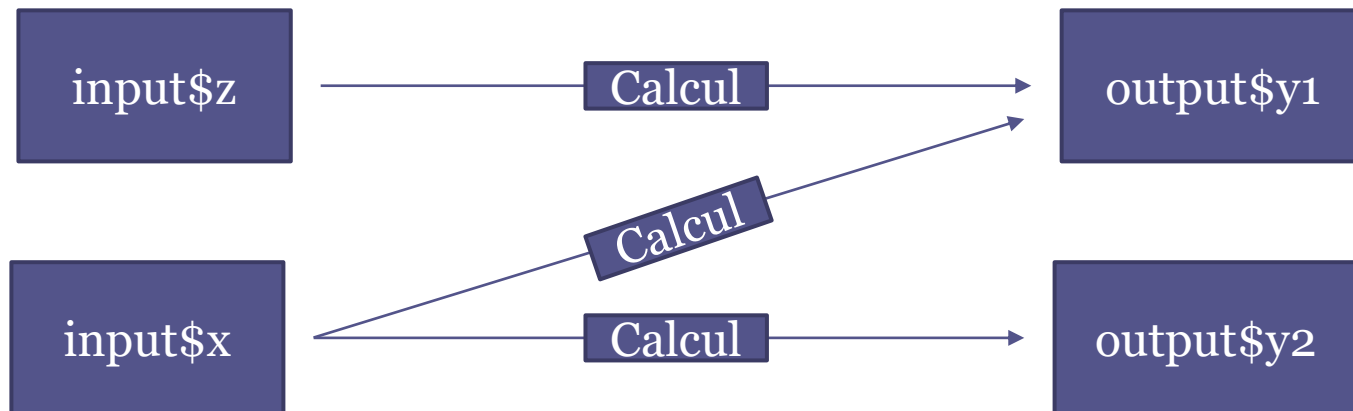
**textOutput**(outputId, container, inline)

# Réactivité

- Avec l'input nommé "x" on crée l'output "y":



- L'input "x" et "z" crée l'output "y1" et "y2":

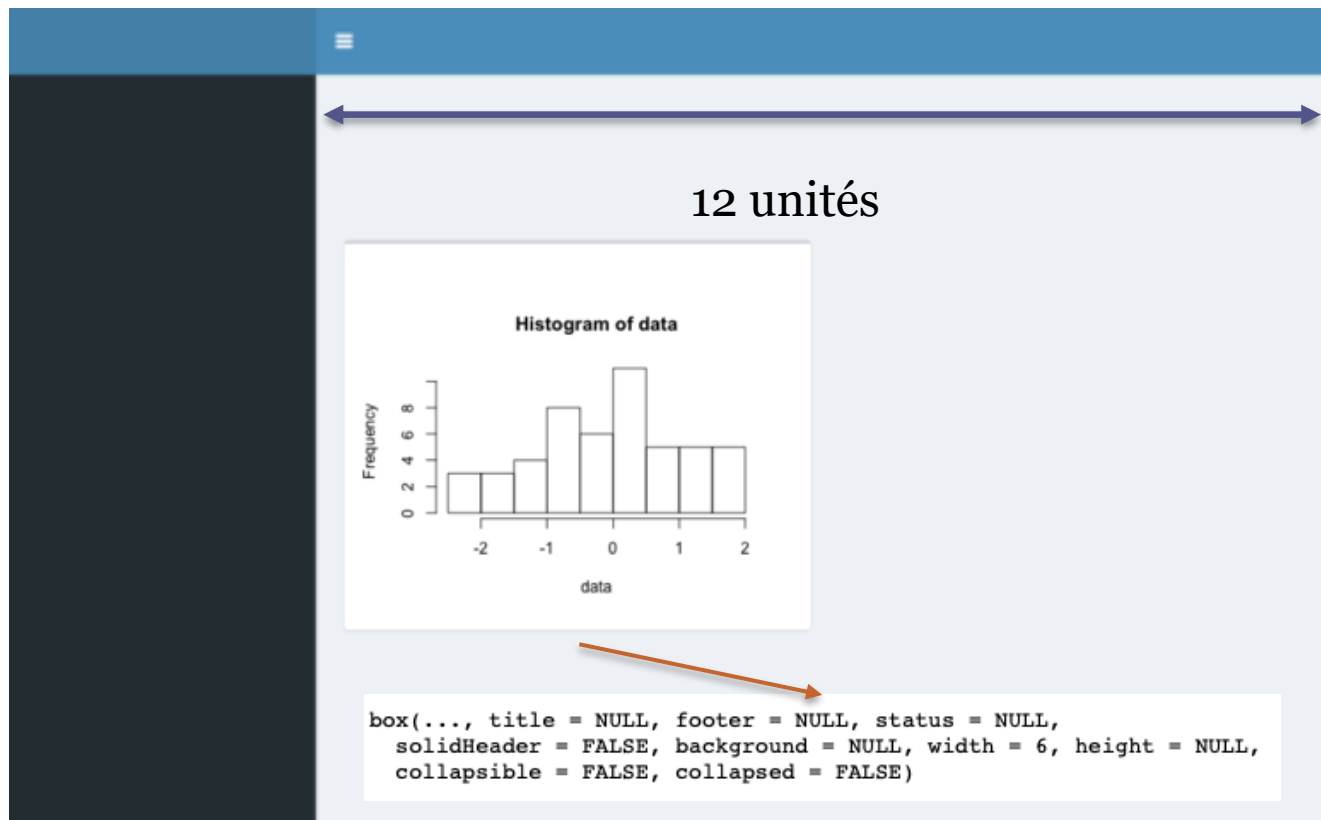




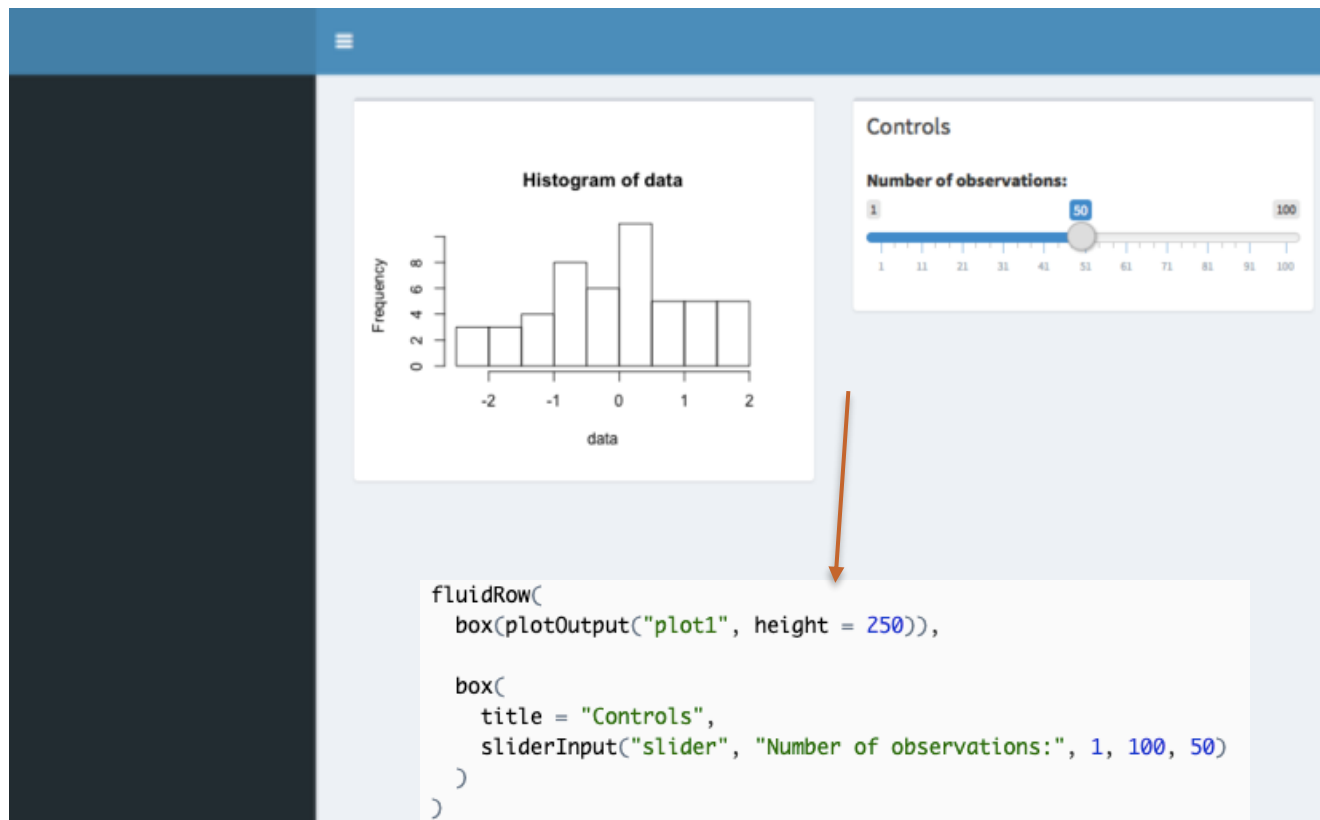
# Exemple

- On demande un nom de fichier xlsx à l'utilisateur et ça ressort la moyenne et la somme de la première colonne.
- Input: nom du fichier
- Output 1: moyenne; Output2: somme de la première colonne
- Attention: on ne veut pas télécharger le fichier Excel 2 fois pour les deux outputs.

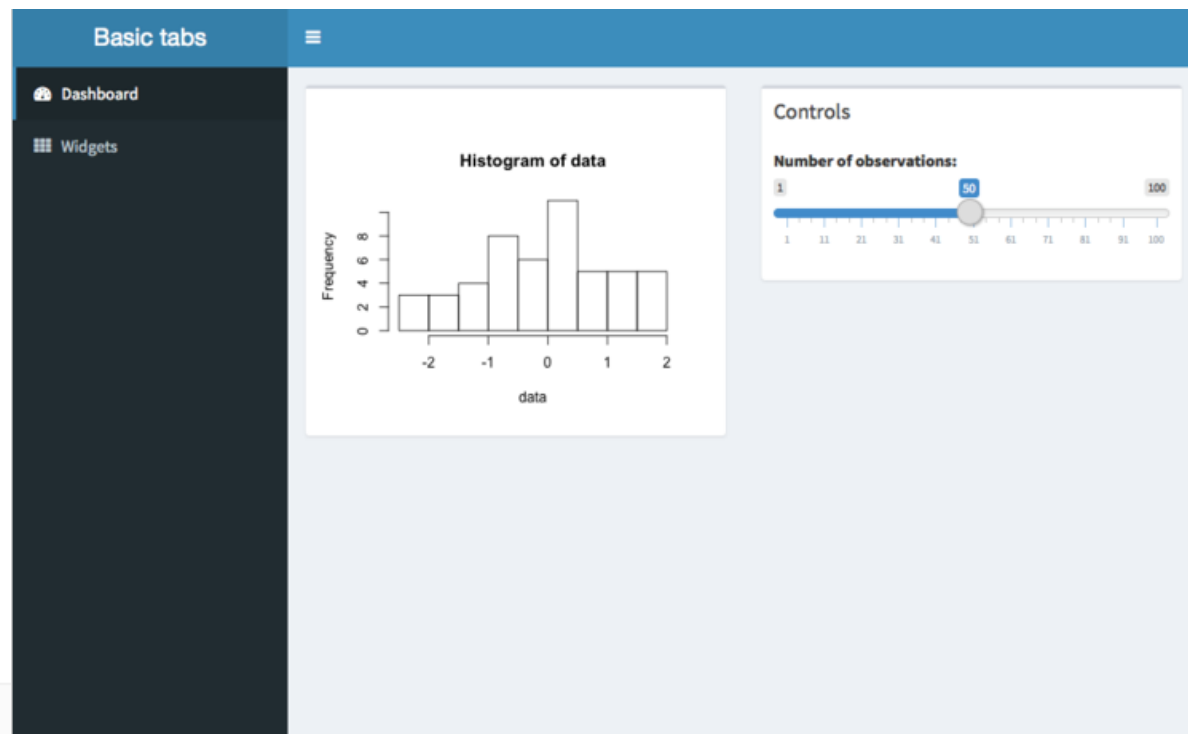
# Apparence des dashboards



# Apparence des dashboards



# Des menus

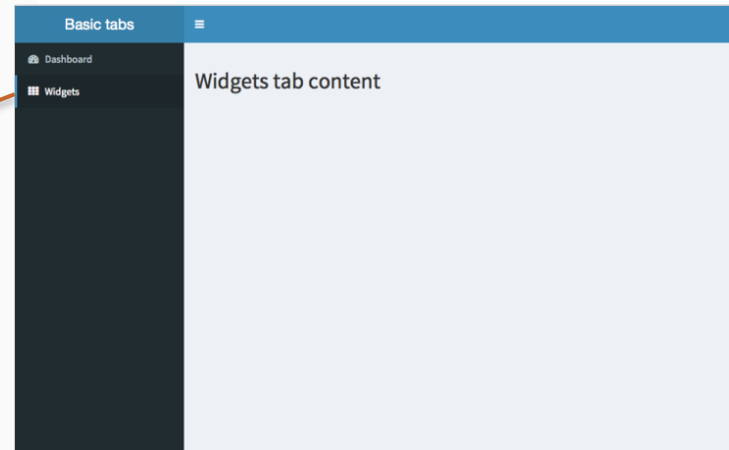
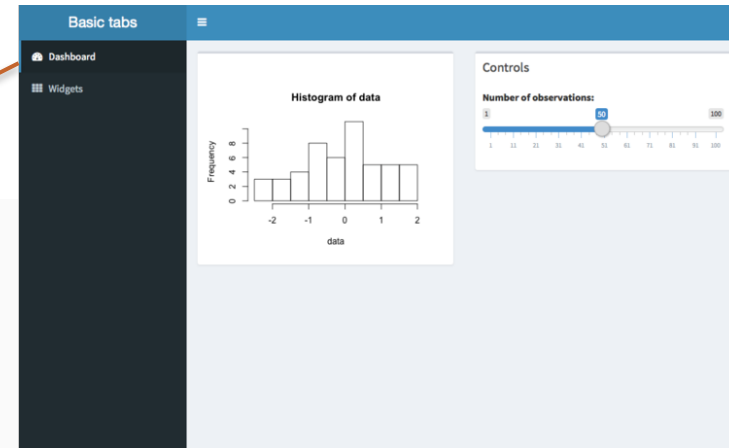


```
## Sidebar content
dashboardSidebar(
  sidebarMenu(
    menuItem("Dashboard", tabName = "dashboard", icon = icon("dashboard")),
    menuItem("Widgets", tabName = "widgets", icon = icon("th"))
  )
)
```

# Chaque chose à sa place

```
## Body content
dashboardBody(
  tabItems(
    # First tab content
    tabItem(tabName = "dashboard",
      fluidRow(
        box(plotOutput("plot1", height = 250)),

        box(
          title = "Controls",
          sliderInput("slider", "Number of observations:", 1, 100, 50)
        )
      )
    ),
    # Second tab content
    tabItem(tabName = "widgets",
      h2("Widgets tab content")
    )
  )
)
```

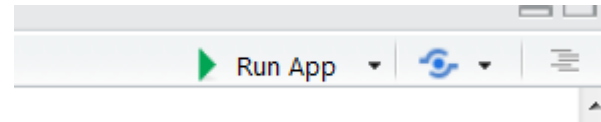


# Les possibilités

- <http://rstudio.github.io/shinydashboard/structure.html>

# Partager son app

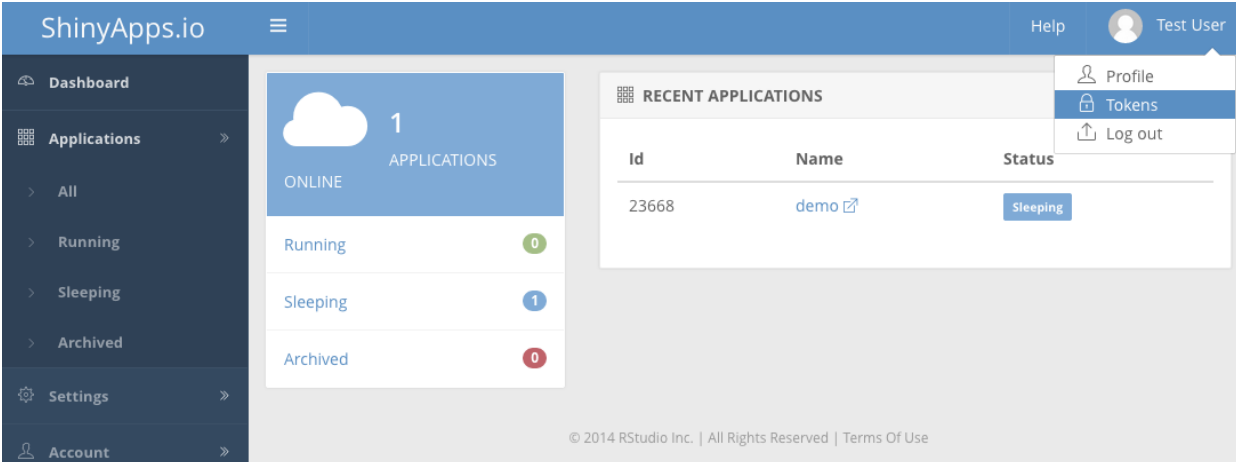
- Envoyer le dossier à quelqu'un et la personne peut charger l'app dans Rstudio avec



- Sur des serveurs...
  1. [Shiny Server Open Source](#) (Linux)
  2. [Shinyapps.io](#) (gratuit pour 5 apps et 25 heures par mois d'utilisation)
  3. Votre propre serveur

# Shinyapps.io

- Très facile!
- 1. Créer un compte sur shinyapps.io
- 2. Installer le package rsconnect
- 3. Dans votre compte Shinyapps.io, aller dans Tokens



The screenshot shows the ShinyApps.io dashboard. The left sidebar contains navigation links: Dashboard, Applications, Settings, and Account. The main content area displays a summary of applications (1 ONLINE) and a table of recent applications. The 'Applications' section shows counts for Running (0), Sleeping (1), and Archived (0). The 'Recent Applications' table lists one application with ID 23668, Name 'demo', and Status 'Sleeping'. A user menu in the top right corner shows options for Profile, Tokens, and Log out.

Id	Name	Status
23668	demo	Sleeping

© 2014 RStudio Inc. | All Rights Reserved | Terms Of Use



# Shinyapps.io

- 4. Copier et lancer dans RStudio

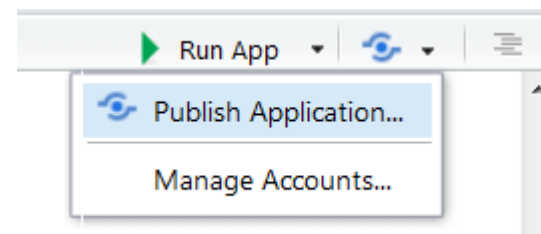
The `shinyapps` package must be authorized to your account using a token and secret. To do this, click the copy button below and we'll copy the whole command you need to your clipboard. Just paste it into your console to authorize your account. Once you've entered the command successfully in R, that computer is now authorized to deploy applications to your ShinyApps.io account.

```
shinyapps::setAccountInfo(name='mytest',  
                           token='[REDACTED]',  
                           secret='<SECRET>')
```

Show secret

 Copy to clipboard

- 5. Publier une application:  
`rsconnect::deployApp()`



# Références

- <http://shiny.rstudio.com/>
- <https://rstudio.github.io/shinydashboard/>

**À vous de jouer maintenant!**

- **Merci de votre attention**