R in Grenoble

Working sessions
One Thursday per month
At Imag or on Zoom 17:00-18:00

R in Grenoble - 25 May 2021

Mayeul Kauffmann (USMB) - Nicolas Ricci (HES-SO)

The main objective of the seminar is to allow R/Shiny application designers to collect information (logs, usability feedback...) from its visitors, to improve their R/Shiny applications. The example will be taken from the CaDyCo project. CaDyCo (Dynamic Collaborative Mapping [of university training]) is a Franco-Swiss InterReg project, led by the USMB (Université Savoie Mont Blanc). The UGA is associated to this project, which (in addition) hosts the hardware infrastructure (Gricad: Gitlab, Winter, Data Lake...); R is used not only for Shiny but also for data collection and preparation (these points will also be discussed). (CaDyCo is for the time being concerned with "engineering sciences" training, in the broad sense).
The session will proceed as follows:
1. Brief introduction
2. Discovery of the use of an R/Shiny dashboard by the seminar participants, with a short questionnaire (knowledge of the users; task of finding some information).
3. How to build a small R/Shiny application
4. How to record individual logs on the behaviour of each user
5. How to study accurate system logs on system resources consumed according to user behaviour
6. Analysis of the data collected during point 2 above. Indeed, precise logs will be collected while the participants discover an R/Shiny dashboard (with a task to complete: find some information).

This will allow participants to understand how visitors interact with an R/Shiny application, in order to improve usability. And in passing, analyse the system resources, to know what is greedy, what are the needs... (so as not to break down the day a service is launched to a wider audience).

Contact :
mayeul.kauffmann @univ-smb.fr

L'objectif principal du séminaire est de permettre aux concepteurs d'applications R/Shiny de collecter des informations (logs, retours sur l'utilisabilité...) auprès de ses visiteurs, pour améliorer leurs applications R/Shiny. L'exemple sera pris du projet CaDyCo. CaDyCo (Cartographie Dynamique Collaborative [des formations universitaires]) est un projet InterReg franco-suisse, piloté par l'USMB. L'UGA est associée à ce projet, qui (en outre) héberge l'infrastructure matérielle (Gricad: Gitlab, Winter, Data Lake…) ; R y est utilisé non seulement pour Shiny mais aussi pour la collecte et la préparation des données (ces points seront aussi évoqués). (Sur la thématique, CaDyCo concerne pour l'instant les formations en « sciences de l'ingénieur », au sens large).

La séance se déroulera ainsi:
1. Brève introduction
2. Découverte de l'utilisation d'un tableau de bord R/Shiny par les participants au séminaire, avec un bref questionnaire (connaissance des utilisateurs ; tâche de recherche de quelques informations).
3. Comment construire une petite application R/Shiny

4. Comment enregistrer des logs individualisés sur le comportement de chaque utilisateur
5. Comment étudier des logs système précis sur les ressources systèmes consommés en fonction des comportements des utilisateurs
6. Analyse des données collectées pendant le point 2 ci-dessus. En effet, des logs précis seront collectés pendant que les participants découvrent un tableau de bord R/Shiny (avec une tâche à remplir : trouver quelques informations).

Cela permettra aux participants de comprendre comment les visiteurs intéragissent avec une application R/Shiny, afin d'améliorer l'utilisabilité. Et au passage analyser les ressources systèmes, histoire de savoir ce qui est gourmand, quels sont les besoins... (pour ne pas tomber en panne le jour du lancement d'un service auprès d'un public plus large).

Content

CaDyCo..4
Online survey and graphic dashboard, made with R/Shiny......................................6
General architecture : Client – Server - R/Shiny..7
Comparing R/Shiny in Docker with virtual machines..8
# Required packages to follow the tutorial..9
# Example 1A - A simple Shiny app with a slider..10
# Example 1B - Shiny app with a slider. Pofiling & improving...............................11
# Example 2 - A simple Shiny app – looking inside Shiny with reactlog...............12
# Example 3 : A surveys with Shiny (shinysurveys)...13

CaDyCo

https://www.univ-smb.fr/recherche/projets-de-recherche/cadyco/

CaDyCo: Dynamic mapping and collaborative adaptation of the educational offer

An innovative collaborative decision-making platform for the adaptation of the higher education offer in the Franco-Swiss territory to the needs of the local ecosystem

This innovative tool has been developed mainly for university managers, but also for future applicants for these courses, socio-economic players in the Franco-Swiss territory, and a cross-border network of partners. It aims to increase the adequacy of the higher education offer to the needs of the local socio-economic fabric.

The ambition of CaDyCo Formation is to bring together all the engineering science programmes offered by the institutions in the cross-border area on a common platform, by organising in an entirely new way the presentation of this offer in the form of a dynamic cartography, which meets the needs of future students, companies, the educational teams of the institutions or the local authorities. Through the integration of additional indicators from the cross-border territory, the users of the platform will be able to have a real decision-making tool specific to each category of actor to orientate themselves in the offer, choose a training course, imagine relevant developments or new training courses adapted to the needs of the territory.

CaDyCo : Cartographie dynamique et adaptation collaborative de l’offre de formation

Une plateforme collaborative innovante d’aide à la décision pour l’adaptation de l’offre de formation supérieure du territoire franco-suisse aux besoins de l’écosystème local

Cet outil innovant est développé à destination principalement des responsables universitaires mais aussi des futurs candidats à ces formations, des acteurs socio-économiques du territoire franco genevois, et d’un réseau transfrontalier de partenaires. Il vise à augmenter l’adéquation de l’offre de formations supérieures aux besoins du tissu socio-économique local.

CaDyCo Formation a pour ambition de réunir l’ensemble de l’offre de formation en Sciences de l’ingénieur proposée par les établissements de l’espace transfrontalier sur une plateforme commune, en organisant d’une manière entièrement nouvelle la présentation de cette offre sous la forme d’une cartographie dynamique, qui répond aux besoins des futurs étudiants, des entreprises, des équipes pédagogiques des établissements ou des collectivités. À travers l’intégration d’indicateurs complémentaires issus du territoire transfrontalier, les utilisateurs de la plateforme pourront disposer d’un véritable outil d’aide à la décision propre à chaque catégorie d’acteur pour s’orienter dans l’offre, choisir une formation, imaginer des évolutions pertinentes ou des créations nouvelles de formations adaptées aux besoins du territoire.

Online survey and graphic dashboard, made with R/Shiny

(we’ll collect the logs and answers and look at them together just afterwards)

Please go to :

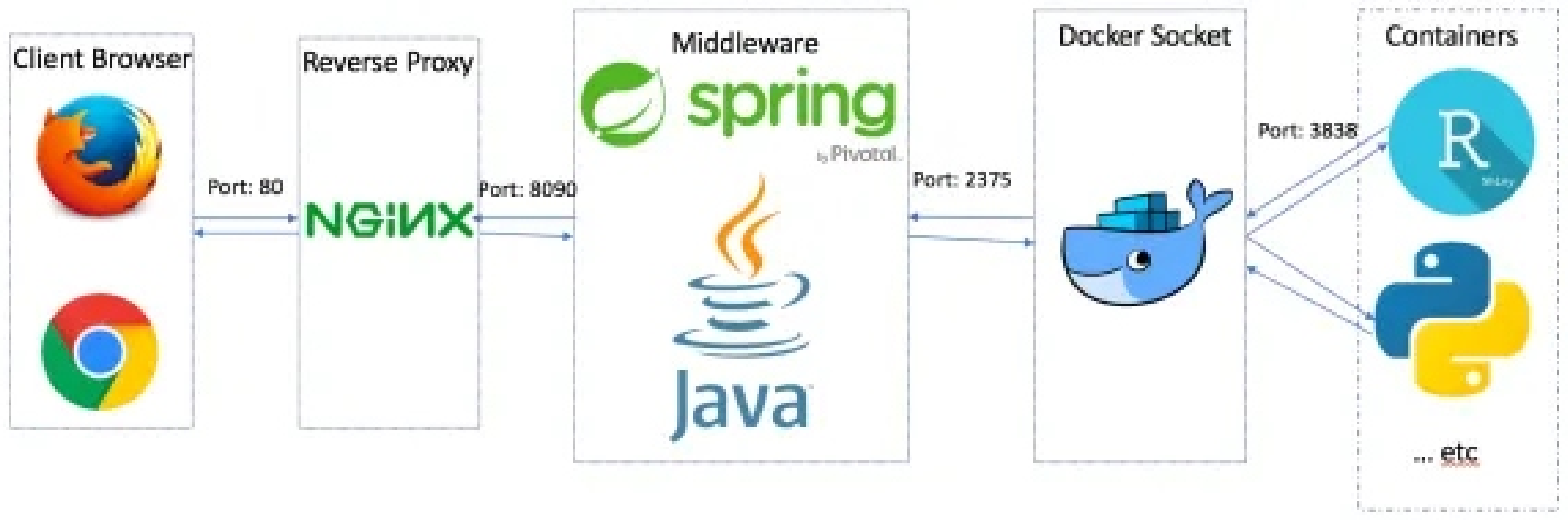
http://cadyco.education:8081

then to :

http://cadyco.education:8081/devenir_des_etudiants_survey_ring_14/

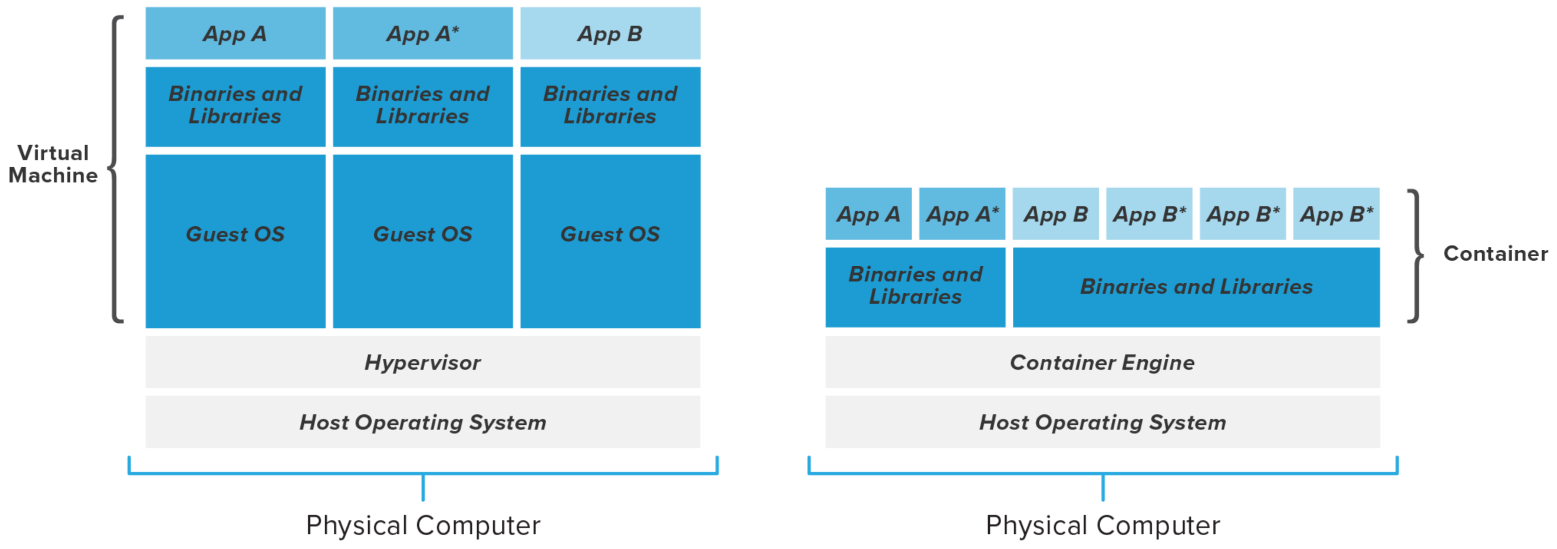
General architecture : Client – Server - R/Shiny

The architecture is similar to the one below (replacing the « spring » middleware by Liferay, engineered by Pentila)



https://www.r-bloggers.com/2017/10/how-to-use-shiny-containers-with-shinyproxy/

Comparing R/Shiny in Docker with virtual machines



Source : https://www.f5.com/services/resources/white-papers/f5-and-containerization

# Required packages to follow the tutorial

```
install.packages(c(
"magrittr",
"shiny",
"profvis",
"reactlog",
"shinysurveys")
```

# Example 1A - A simple Shiny app with a slider

```r
library(shiny)
# Define UI for app that draws a histogram

ui <- fluidPage(
  titlePanel("Hello Shiny!"),

  # Sidebar layout with input and output definitions
  sidebarLayout(
    sidebarPanel(
      sliderInput(inputId = "bins",
                  label = "Number of bins:",
                  min = 1, max = 10,
                  value = 5)
    ),
    mainPanel(

      # Output: Histogram ----
      plotOutput(outputId = "distPlot")
    )))

server <- function(input, output) {

  # Histogram of the Old Faithful Geyser Data with requested number of bins.
  # Since it is wrapped in renderPlot it is re-executed when inputs (input$bins) change
  output$distPlot <- renderPlot({
    x    <- rep(faithful$waiting, 100000) # we replicate 1000000 times the dataset, to increase the workload
    bins <- seq(min(x), max(x), length.out = input$bins + 1)
    hist(x, breaks = bins, col = "#75AADB", border = "white",
         xlab = "Waiting time to next eruption (in mins)",
         main = "Histogram of waiting times")
  })
}

shinyApp(ui, server)
```

# Example 1B - Shiny app with a slider. Pofiling & improving

```r
library(magrittr) ; library(shiny) # shiny 1.6 is needed for bindCache

# Uncomment to read the data only once (outside of the server function) to speed things up:
# x <- rep(faithful$waiting, 100000)

# Define UI for app that draws a histogram
ui <- fluidPage(
  titlePanel("Hello Shiny!"),
  # Sidebar layout with input and output definitions
  sidebarLayout(
    sidebarPanel(
      sliderInput(inputId = "bins",
                  label = "Number of bins:",
                  min = 1, max = 10,
                  value = 5) ),
    mainPanel(
      # Output: Histogram ----
      plotOutput(outputId = "distPlot"))))
server <- function(input, output) {
  # Histogram of the Old Faithful Geyser Data with requested number of bins.
  # Since it is wrapped in renderPlot it is re-executed when inputs (input$bins) change
  output$distPlot <- renderPlot({
    # Uncomment above line 4 and comment out below to read the data only once
    x    <- rep(faithful$waiting, 100000) # we replicate 1000000 times the dataset, to increase the workload and
better see perf. improvements
    bins <- seq(min(x), max(x), length.out = input$bins + 1)
    hist(x, breaks = bins, col = "#75AADB", border = "white",
         xlab = "Waiting time to next eruption (in mins)",
         main = "Histogram of waiting times")
  }) ## %>% bindCache(input$bins)   # Uncomment to enable caching
}


## profvis::profvis(runApp(  # Uncomment both lines for profiling
shinyApp(ui, server)
## ))
```

# Example 2 - A simple Shiny app – looking inside Shiny with reactlog

```
library(shiny)
f <- function() {
  Sys.sleep(1)
  g()
  1
}
g <- function() {
  Sys.sleep(1)
}

ui <- fluidPage(
  actionButton("x", "Push me"),
  textOutput("y"),
  plotOutput("yPlot")
)
server <- function(input, output, session) {
  output$y <- eventReactive(input$x, f())
  output$yPlot <- renderPlot(hist(f()))
}

reactlog::reactlog_enable()
runApp(shinyApp(ui, server))
shiny::reactlogShow()
```

# Example 3 : A surveys with Shiny (shinysurveys)

```
# shinysurveys_v1.R
# For more recent version of library :
# install.packages("devtools")
# remotes::install_github("jdtrat/shinysurveys", ref = "26837fbb3365a56b861feca40502eeaa721e303c") # 5 May 2021
library(shiny)
library(shinysurveys)
df <- data.frame(question = c("What is your favorite food?", "Rank the software according to your
preference:", "Rank the software according to your preference:"),
                 option = c("Your Answer",  "R", "S+"),
                 input_type = c("text", "rank_{{2}}", "rank_{{2}}"),
                 input_id = c("favorite_food", "software", "software"),
                 dependence = rep(NA, 3),
                 dependence_value = rep(NA, 3),
                 required = rep(F, 3)
                 )

ui <- fluidPage(
  surveyOutput(df = df,
               survey_title = "Hello, World!",
               survey_description = "Welcome! This is a demo survey showing off the {shinysurveys}
package.")
)

server <- function(input, output, session) {
  renderSurvey(df = df)

  observeEvent(input$submit, {
    showModal(modalDialog(
      title = "Congrats, you completed your first shinysurvey!",
      "You can customize what actions happen when a user finishes a survey using input$submit."
    ))
  })
}

shinyApp(ui, server)
```