Module 4. Customizable and interactive dashboards for the communication of results

BARÇA INNOVATION HUB
Universitas

- Unit 4.1 Customizable and interactive dashboards for the communication of results
- Activities
- References

Unit 4.1 Customizable and interactive dashboards for the communication of results

In previous courses, we have highlighted the large amount of information available to us as sports scientists with the development of new technology and the emergence of new working methods derived from scientific research. There is a growing need to analyse parameters to make the best possible decisions that positively affect our players or organization's performance.

We have also looked at the different types of results communication: from simple graphs that allow us to represent directly and in a summarized way what we want to show to automated reports that allow us repeatability in the analysis at different points of the season or for different players or teams. Finally, we mentioned tools for visualization or distribution of results that are more interactive or that allow to keep track of a data history/flow.

These tools differ from the rest in that they do not respond to the immediate need to present the results in a certain way and they do not intend to standardize the way in which they are visualized. They

do however incorporate multiple sources of information. In the case of sports performance data, it can range from physical performance data, injuries and medical history to participation or results in competition. Not only do these data come from multiple data sources, but they also typically have a longer recorded period of this same data.

Having a tool available to the end user that facilitates data exploration and interacts with the different sources of information allows us to provide more context on the element we are exploring, to see which variables may be affecting each other and what the trend is over time.

In the field of physical performance data analysis, these tools are commonly referred to as AMS (athlete management/monitoring systems). In many cases, these enable the recording of information directly in them, but above all they enable multiple variables of information about the athlete or the team to be available. There are multiple commercial options depending on the type of data or technology used and the particularities or functionalities will vary depending on the provider company.

As described by Compton et al. (2019), the aim of developing an athlete monitoring system is to assist in decision-making processes related to planning and adjusting training, as well as to reduce the risk of injury. They also emphasize that it is essential to know how

data should be analysed and what significant parameters should be explored in order to extract and present information efficiently.

Another fundamental aspect discussed by Coutts et al. (2018), is the need to develop dashboards or AMS according to a pre-decided conceptual framework, in addition to them being integrated with the organization's vision. They explain that we should determine the need we want to address in accordance with the methodology and work philosophy, to establish the design and structure of the tool.

The main solutions provided by an AMS are as follows.

- Data centralization - data from different sources on the same platform.
- Easy access to data for communication.
- Interactivity for the exploration of results.
- Decision-making guidance.
- Contextualization of values got.
- Effectively sharing of results.

As we can see, to achieve these objectives, automated reports or simple graphs are not enough, which is why there are multiple tools for the creation of interactive visualizations such as Tableau or Power BI. RStudio is a software capable of developing tools such as AMS.

The advantages of using RStudio instead of other tools is its great capacity for customization and creation of dashboards or interactive tools adapted to specific needs. In addition, we will be able to easily integrate any type of analysis or data transformation that we carry out on them. As it is a free tool, the absence of economic cost is a great point in favour in many cases.

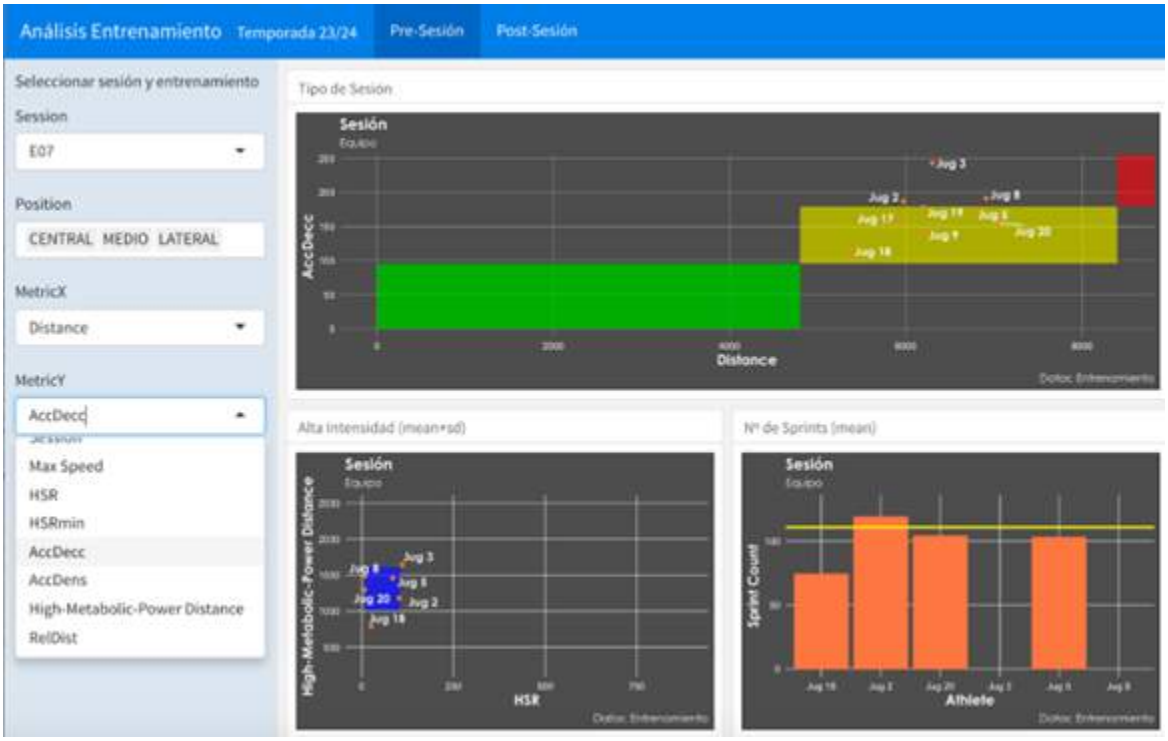
RShiny is the library within RStudio that will allow us to create web apps or interactive dashboards. It offers customization of each of the parts and the integration of data analysis and visualization.

We have already seen in previous modules some examples about these visualizations, but we will show two more examples of possible dashboard designs that can be applied to the sports scientist's context.

In this first image, we see a dashboard created for the analysis of a training session; in this case, to be display the physical demands from data got through GPS devices. As we can see in the panels on the left, there are certain elements that foster user interaction with the app or dashboard. This is its fundamental element. The user can select, among others, the session to be analysed, the players of a certain position or other specific variables.

As we have already mentioned, there are a large number of variables and metrics at our disposal after each session or training session. Depending on our objectives, time of the season or the player's condition, we will be more interested in knowing the results of one variable or another. The dashboard permits this interaction and guides decision-making, determining the player's position compared to the rest of the team.

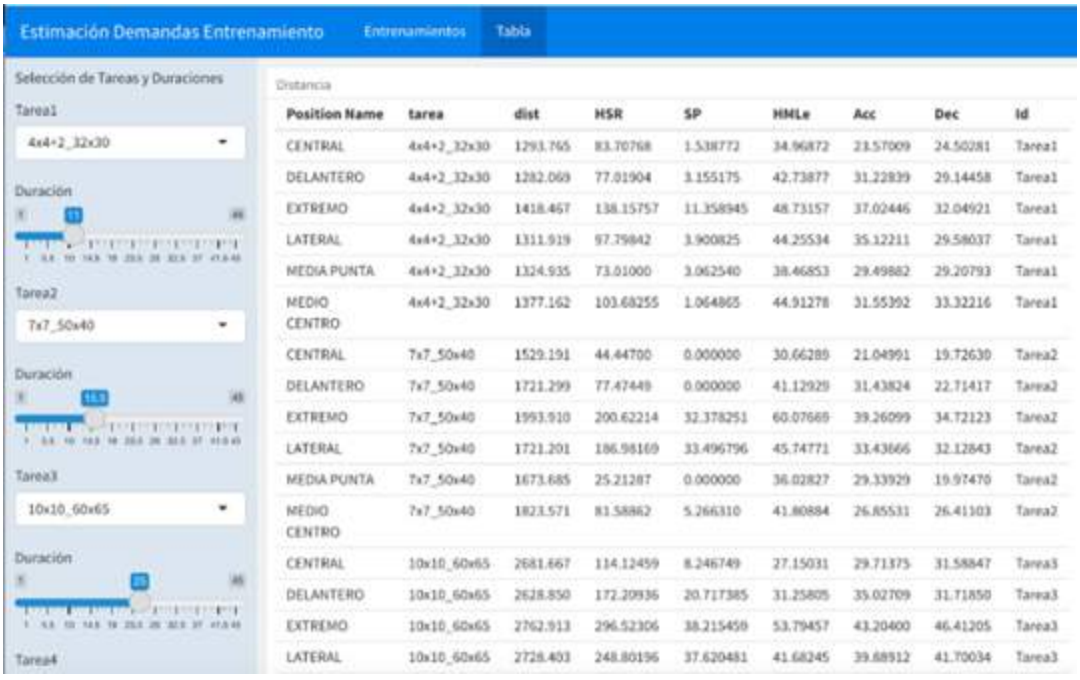
Figure 1. RStudio



Source: RStudio screenshot (Allaire, 2011). RStudio app screenshot, author's production

In this second example, the same dashboard format is presented but with a different objective. This app aims to help in pre-workout decision-making. Using historical team data and statistical relationships between duration, space, and player density, we can estimate the load of each of the positions on the different performance metrics we select. In this way, we can interact with the dashboard, selecting tasks and durations to establish a plan before training. The same principle can be applied to readaptation sessions or internal load variables.

Figure 2. Table in RStudio



Position Name	tarea	dist	HSR	SP	HMLe	Acc	Dec	Id
CENTRAL	4x4+2_32x30	1293.765	83.70768	1.538772	34.96872	23.57009	24.50281	Tarea1
DELANTERO	4x4+2_32x30	1282.069	77.01904	3.155175	42.73877	31.22839	29.14458	Tarea1
EXTREMO	4x4+2_32x30	1418.467	138.15757	11.358945	48.73157	37.02446	32.04921	Tarea1
LATERAL	4x4+2_32x30	1311.919	97.79842	3.900825	44.25534	35.12211	29.58037	Tarea1
MEDIA PUNTA	4x4+2_32x30	1324.935	73.01000	3.062540	38.46853	29.49882	29.20793	Tarea1
MEDIO CENTRO	4x4+2_32x30	1377.162	103.68255	1.064865	44.91278	31.55392	33.32216	Tarea1
CENTRAL	7x7_50x40	1529.191	44.44700	0.000000	30.66289	21.04991	19.72630	Tarea2
DELANTERO	7x7_50x40	1721.299	77.47449	0.000000	41.12929	31.43824	22.71417	Tarea2
EXTREMO	7x7_50x40	1993.910	200.62214	32.378251	60.07665	39.26099	34.72123	Tarea2
LATERAL	7x7_50x40	1721.201	186.98169	33.496796	45.74771	33.43666	32.12843	Tarea2
MEDIA PUNTA	7x7_50x40	1673.685	25.21287	0.000000	36.02827	29.33929	19.97470	Tarea2
MEDIO CENTRO	7x7_50x40	1823.571	81.58862	5.266310	41.80884	26.85531	26.41103	Tarea2
CENTRAL	10x10_60x65	2681.667	114.12459	8.246749	27.15031	29.71375	31.58847	Tarea3
DELANTERO	10x10_60x65	2628.850	172.20936	20.717385	31.25805	35.02709	31.71850	Tarea3
EXTREMO	10x10_60x65	2762.913	296.52306	38.215459	53.79457	43.20400	46.41205	Tarea3
LATERAL	10x10_60x65	2728.403	248.80196	37.620481	41.68245	39.88912	41.70034	Tarea3

Source: RStudio screenshot (Allaire, 2011). RStudio app screenshot, author's production

RShiny: Fundamental Parts

All processes, analysis or use of functions in RStudio follow the same pattern - we have to know the names of the functions we use, their structure and their arguments. To use them effectively, we shall repeat the process multiple times, know the possible causes of the errors we may get and become familiar with the results. RStudio is a tool with a certain learning difficulty at the beginning due to its complexity, and RShiny, as part of RStudio, shares this characteristic. It is an advanced functionality, so we need to understand its basic principles and structure to use it as a starting point for larger apps and more advanced projects. There are multiple resources available to enlarge the tool and adjust it to our professional needs.

There are two main components of every Shiny app (Wickham, 2021).

- UI: This is the user interface. In this part, we will have to make relevant adjustments to define the app's appearance, from the distribution of the graphics or selection inputs to the colours or pages of the app.
- Server: This element defines the way in which the app works, i.e. the code or data processed and the way it is displayed.

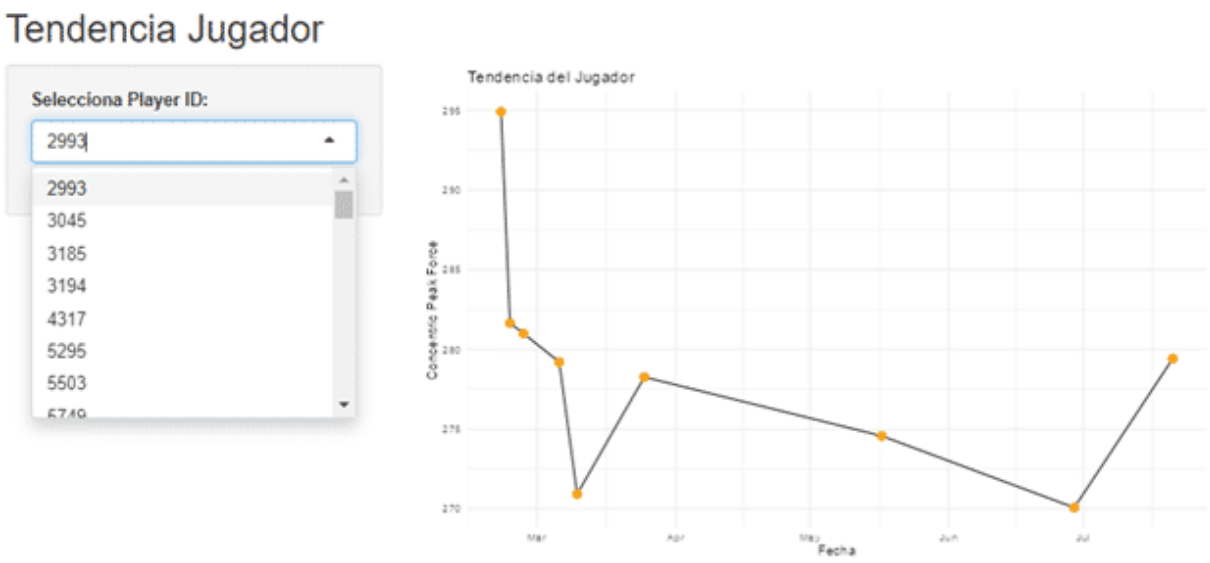
We will need to define each of these elements with their corresponding structure in order to use the app. Even though we will see some examples on the video material, we will explain the logic of

this app through an example, to appreciate what is happening in each of the parts.

The aim was to represent the trend of a player's values in a metric analysed using a force platform. The app looks like this.

On the left, we place an element where we can filter by the player identifier. When the ID is selected the result of the graph will be automatically updated.

Figure 3. Player Trend in RStudio



Source: RStudio screenshot (Allaire, 2011). RStudio app screenshot, author's production

Regarding the code that corresponds to each of the parts,

first, we imported the libraries we want to work with and the database used. In this case, it is data that we have already used before - there are columns that indicate the date, the player's identifier and each of the analysis variables.

Figure 4. Database

```
library(tidyverse)
library(shiny)
library(lubridate)


datos_salto <- read.csv("forceplates.csv",sep=",")


ui <- fluidPage(
  titlePanel("Tendencia Jugador"),
  sidebarLayout(
    sidebarPanel(
      selectInput("player", "Selecciona Player ID:", choices = unique(datos_salto$player_id))
    ),
    mainPanel(
      plotOutput("trendPlot")
    )
  )
)
```

Source: prepared by the author.

The second step is to define the user interface.

- fluidPage: this function will adjust the dimensions of the graphics and elements to the size of the window in which they are being viewed.

- titlePanel: the title that we want to be displayed in the app.
- sidebarLayout-sidebarPanel: this defines the elements that we want to be displayed on the left side of the app in order to create different filters. Below there are some other examples of items that we can select for apps based on different needs.
- selectInput: It has been decided that we are going to use an element that allows you to select a specific player from the list of all indicators.
    - "Player": this will be the indicator that we will use in the "server" part.
    - "Select Player ID": this is the text we want to display above the selection element.
    - "Choices": are all the possible options that we want the user to have (in this case, all the indicators of the players got using the unique() function that we have seen in previous courses).
- mainPanel: this is what we want to be displayed in the main display panel. In this case, a graph (plotOutput) to which we decide to assign the name "trendPlot".

Figure 5. Server element

```
server <- function(input, output) {
  output$trendPlot <- renderPlot({

    player_data <- datos_salto %>%
      filter(player_id == input$player)

    ggplot(player_data, aes(x = ymd(test_date), y = concentric_peak_force)) +
      geom_line(group="player") +
      geom_point(size=3,color="orange") +
      labs(title = paste("Tendencia del Jugador"),
           x = "Fecha",
           y = "Concentric Peak Force")+
      theme_minimal()
  })
}

shinyApp(ui = ui, server = server)
```

Source: prepared by the author.

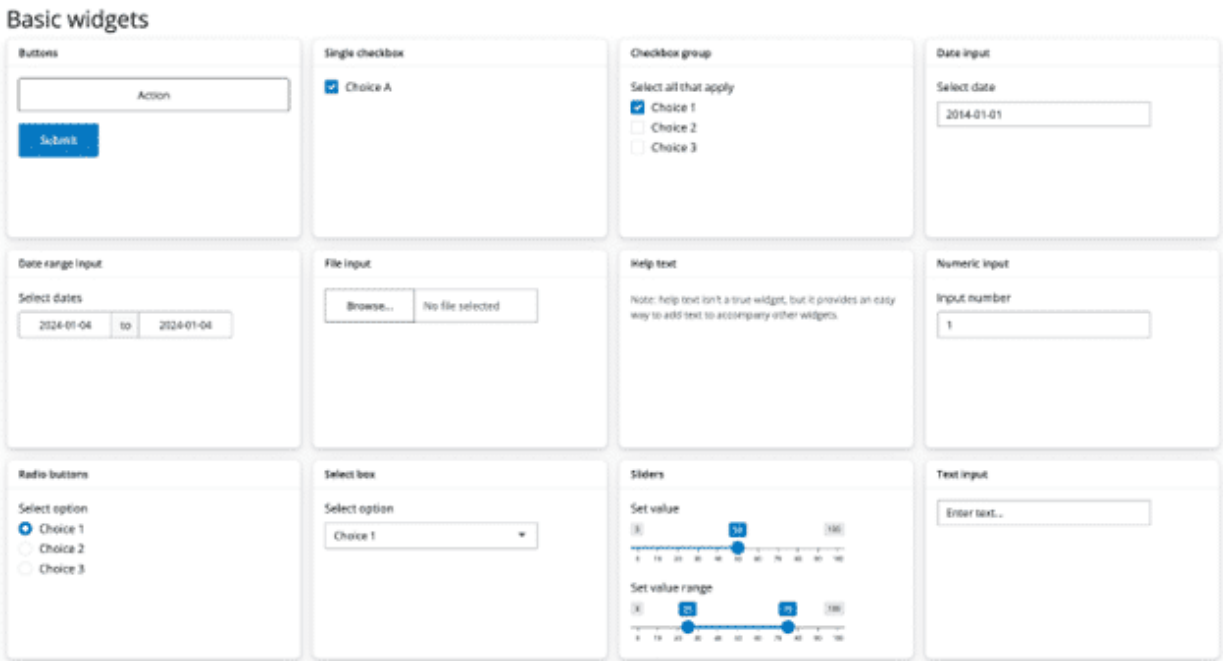
To define the "server" element, we need to create a function that follows the structure shown. In this case, it contains "input" (the selection of the player) and "output" (the graph we want to show).

- output$trendPlot <- renderPlot: this has the same name that we have decided to assign in the UI; If we had more than one chart to display, the name would indicate which visualization we are modifying.
- player_data: we adjust our database to filter only the desired player; to do this, we use input$player and then use the same name "player" that we have decided in the UI.
- Ggplot graph: in the same way as we seen in previous modules, a graph is made to display the information in the desired way.

Finally, the shinyApp function is used with both elements created to launch the visualization of the app.

This is a very simple and introductory example that serves as a starting point for the creation of more complex tools. There are many resources available to get familiar with the possibilities offered by this library. On the https://shiny.posit.co/ page we can find examples of visualizations, as well as descriptions of types of elements (widgets [interactive modules]) that we can use in the user interface for our visualizations.

Figure 6. Widgets



Source: Screenshot taken from https://shiny.posit.co/

It is important to establish a goal from which to build the apps, in the same way as with any other part of the analysis process. Knowing the strengths of RStudio and RShiny provides us with skills that can be key to adding value to the sports performance field.

CONTINUE

Activities

Which of the following statements are advantages of using RStudio over other tools?

- [ ] It has a great capacity for customization and creation of dashboards.
- [ ] It facilitates the integration of any type of data analysis or transformation.
- [ ] It is a payment tool with advanced options.
- [ ] It is a free tool, at no economic cost.
- [ ] It adapts to specific needs through interactive tools.

SUBMIT

Which of the following statements about athlete monitoring or management systems (AMS) is correct?

- [ ] They enable information recording directly in them.
- [ ] They only work with a specific type of data.
- [ ] They offer multiple variables of information about athletes or teams.
- [ ] All AMS tools have the same functionalities.
- [ ] The functionalities of AMS may vary depending on the provider company.

SUBMIT

There are two main components of every Shiny app (Wickham, 2021). Which are they?

- [ ] UI
- [ ] Server
- [ ] Player_data
- [ ] RenderPlot
- [ ] FluidServer

SUBMIT

References

Allaire, J. (2011). RStudio (2023.06.1) [herramienta de software]. Posit.

Compton, H., Delaney, J., Duthie, G. & Dascombe, B. (2019). Developing Athlete Monitoring Systems in Team Sports: Data Analysis and Visualization. International Journal of Sports Physiology and Performance, 14(6), 698–705. https://doi.org/10.1123/ijspp.2018-0169

Coutts, A., Crowcroft, S. & Kempton, T. (2018). Developing athlete monitoring systems: Theoretical basis and practical applications en M. Kellmann & J. Beckmann (Eds.), Sport, Recovery and Performance: Interdisciplinary Insights (pp. 19-32). Routledge.

Wickham, H. (2021). Mastering Shiny. O'Reilly Media, Inc.

Posit team. (2023). RStudio: Integrated Development Environment for R. Posit Software, PBC, Boston, MA. URL: http://www.posit.co/.

CONTINUE