

Data Visualization with R Shiny tutorial

Ariane Ducellier

University of Washington - Fall 2025

Progress bar

If some computation in `server.R` can take a long time, it is useful to wrap the corresponding code inside the `Shiny withProgress()` function.

In `server.R`

```
withProgress(message = ... ,  
  detail = ' ... ', value = 0,  
  ... function code ...  
  incProgress(1/3)  
  ... function code ...  
  incProgress(1/3)  
  ... function code ...  
  incProgress(1/3)  
  ... function code ...  
})
```

```
Go to File >  
  New File >  
    R Markdown >  
      From Template >  
        Flex Dashboard
```

Click on Knit to see the empty dashboard.

In the first R block, load the libraries and the data:

```
library(flexdashboard)
library(tidyverse)
library(leaflet)
load("geocodedData.Rdata")
```

Change the names of the R Markdown headers and fill the R blocks with the code from `dashboard1.Rmd`.

Click on Knit to see the final dashboard.

Adding shiny to the flexdashboard

Modify the header by adding shiny and using a rows orientation:

```
title: "Flexdashboard 2"  
runtime: shiny
```

We will add one sidebar column:

```
Column {.sidebar}
```

We fill the R block with R shiny code to create a slider and a checkbox as done previously in `ui.R`.

Adding shiny to the flexdashboard

Create a simple row and a row with several tabs:

```
Row
```

```
Row {.tabset}
```

We fill the R block with R shiny code to create plots as done previously in `server.R`.

In this case, the filtering is done for every block of R code. We cannot define a reactive object to filter the years.

Shiny dashboards

```
library(shinydashboard)
header <- dashboardHeader( )
sidebar <- dashboardSidebar()
body <- dashboardBody()
dashboardPage(header, sidebar, body,
  title = NULL,
  skin = c("blue", "black", "purple", "green", "red",
    "yellow"))
```

Adding a menu to the sidebar

```
sidebarMenu(id = NULL,  
  menuItem("Name",  
    icon = ... ,  
    tabName = ... ,  
    badgeLabel = ... ,  
    badgeColor = ... ,  
    ...  
  ),  
  sliderInput( ... )  
)
```

`tabName` will be referred to in the dashboard body to create the corresponding graph.

Improving the UI - Adding a menu to the sidebar

```
tabItems(  
  tabItem(tabName = ... ,  
    fluidRow(  
      box(width = 10,  
        plotOutput("trend"),  
        checkboxInput( ... )),  
      box(width = 2, ... )  
    ),  
  )  
)
```

`tabName` corresponds to the value given in `menuItem` in the sidebar.

Improving the UI - Adding info boxes

In the file `ui.R`:

```
infoBoxOutput(width = 3, "infoYears")
```

In the file `server.R`:

```
output$infoYears = renderInfoBox({  
  infoBox(title,  
    value = NULL,  
    icon = ... ,  
    color = ... ,  
    fill = ...  
  )  
})
```

Improving the UI - Adding icons

```
tabPanel("Trend",  
        plotOutput("trend"),  
        icon = icon("calendar"))
```

```
tabPanel("Summary",  
        textOutput("summary"),  
        icon = icon("user", lib = "glyphicon"))
```

See a list of icons here:

- <https://fontawesome.com/icons>
- <https://icons.getbootstrap.com/>