README

Esta aplicación es un experimento de la herramienta Shiny para explorar el comportamiento de contagio y muertes de Covid-19 en todos los paises, este set de datos lo obtuve de OpenData, y muestra el reporte diario de casos confirmados y muertes de cada pais (tambien tenemos la poblacion total de cada pais, por si deseamos trabajar un analisis de contagio por millon de habitantes, sera en otro caso).

Librerias

```
library(shiny)
library(stringr)
library(ggplot2)
```

Carga de Datos Inicial

Realizamos una carga de datos al momento de iniciar Shiny, esta es la secuencia:

- Descargamos y leemos los datos de la pagina arriba mencionada (Nota: Recordemos que la forma correcta de manejar aplicaciones portables y reproducibles es dejando la ruta por defaul en la que estan los ejecutables del programa)

```
 download.file('https://opendata.ecdc.europa.eu/covid19/casedistribution/csv','descarga.csv')
DataCovid <- read.csv("descarga.csv")
head(DataCovid)
```

```
##      dateRep day month year cases deaths countriesAndTerritories geoId
## 1 05/07/2020   5     7 2020   348      7             Afghanistan    AF
## 2 04/07/2020   4     7 2020   302     12             Afghanistan    AF
## 3 03/07/2020   3     7 2020   186     33             Afghanistan    AF
## 4 02/07/2020   2     7 2020   319     28             Afghanistan    AF
## 5 01/07/2020   1     7 2020   279     13             Afghanistan    AF
## 6 30/06/2020  30     6 2020   271     12             Afghanistan    AF
##   countryterritoryCode popData2019 continentExp
## 1                  AFG    38041757         Asia
## 2                  AFG    38041757         Asia
## 3                  AFG    38041757         Asia
## 4                  AFG    38041757         Asia
## 5                  AFG    38041757         Asia
## 6                  AFG    38041757         Asia
```

- Asignamos el formato correcto a la fecha

```
DataCovid$dateRep<-as.Date(DataCovid$dateRep,format='%d/%m/%Y')
```

- Convertimos los nombres de los paises en Texto

```
DataCovid$countriesAndTerritories<-as.character(DataCovid$countriesAndTerritories)
head(unique(DataCovid$countriesAndTerritories))
```

```
## [1] "Afghanistan" "Albania"     "Algeria"     "Andorra"     "Angola"
## [6] "Anguilla"
```

- Identificamos los nombres unicos de los paises para poder calcular el total acumulado de casos y total acumulado de falledicos por fecha.

```
  data<-DataCovid
  totalCases<-data[0,0]
  countries<-unique(data$countriesAndTerritories)
    for(countrie in countries)  {
        dataAux<-data[which(data$countriesAndTerritories==countrie),]
        dataAux<-dataAux[order(dataAux$dateRep),]
        dataAux<-cbind(dataAux,0,0)
        names<-c(colnames(data),'TotalCase','totalDeath')
        colnames(dataAux)<-names
        rownames(dataAux)<-c(1:nrow(dataAux))
        for( i in 1:nrow(dataAux)){
            if(i==1) dataAux$totalDeath[i]=dataAux$deaths[i] else dataAux$totalDeath[i]=dataAux$deaths[
        }
        for( i in 1:nrow(dataAux)){
            if(i==1) dataAux$TotalCase[i]=dataAux$cases[i] else dataAux$TotalCase[i]=dataAux$cases[i]+da
        }
        dataAux<- dataAux[which(dataAux$TotalCase!=0),]
        dataAux$dayPerInit<-c(1:nrow(dataAux))
        totalCases<-rbind(totalCases,dataAux)
    }
    head(totalCases)
```

```
##       dateRep day month year cases deaths countriesAndTerritories geoId
## 57 2020-02-25  25     2 2020     1      0             Afghanistan    AF
## 58 2020-02-26  26     2 2020     0      0             Afghanistan    AF
## 59 2020-02-27  27     2 2020     0      0             Afghanistan    AF
## 60 2020-02-28  28     2 2020     0      0             Afghanistan    AF
## 61 2020-02-29  29     2 2020     0      0             Afghanistan    AF
## 62 2020-03-01   1     3 2020     0      0             Afghanistan    AF
##    countryterritoryCode popData2019 continentExp TotalCase totalDeath
## 57                  AFG    38041757         Asia         1          0
## 58                  AFG    38041757         Asia         1          0
## 59                  AFG    38041757         Asia         1          0
## 60                  AFG    38041757         Asia         1          0
## 61                  AFG    38041757         Asia         1          0
## 62                  AFG    38041757         Asia         1          0
##    dayPerInit
## 57          1
## 58          2
## 59          3
## 60          4
## 61          5
## 62          6
```

- Realizamos un barrido por los datos y determinamos aquellos registros que son antes de la fecha en la que se confirmo el primer contagiado por cada pais

```
totalCases<-totalCases[-which(totalCases$TotalCase==0),]
head(totalCases[c(1,7,13,14,12)])
```

```
## [1] dateRep                 countriesAndTerritories totalDeath
## [4] dayPerInit              TotalCase
## <0 rows> (or 0-length row.names)
```

- Y por ultimo, creamos una funcion que grafica los fallecidos y los casos confirmados (Para efectos de este documento, solamente mostraremos la grafica de Confirmados)

```
graficar<-function(name){
   dataset<-totalCases
  countriData<-dataset[0,0]
  for(cty in name){
    dataAux<-dataset[which(dataset$countriesAndTerritories==cty),]
    countriData<-rbind(countriData,dataAux)
  }
  gg1<-ggplot(countriData,aes(x=dayPerInit,y=TotalCase))
  gg1<-gg1+geom_line(aes(color=countriesAndTerritories,linetype=countriesAndTerritories))
  gg1<- gg1+ylab("Dias desde Primer Confirmado")+xlab("Total de Casos Confirmados")
  gg1
}
```

Ejemplo

```
graficar(c('El_Salvador','Guatemala','Nicaragua','Costa_Rica','Honduras'))
```

Incrustandolo en Shiny

Para incrustar en Shiny y poder interactuar con la informacion, debemos separar por partes el codigo y hacerlo compatible con la libreria.

En la Parte de las interfaces, hemos descargado y leido los datos y hemos identificado cada uno de los nombres de los paises para poder interactuar con ellos.

En la parte del server, he agregado la carga nuevamente del archivo y toda la transformacion de datos, y posteriormente en el llamado de los objetos output, he colocado la grafica de los paises seleccionador en el listado de nombres.

Entiendo que esta parte es muy compleja de entender, pero shiny es facil de trabajar, pero complejo de explicar.

Antes de entrar a la aplicacion Shiny abajo incrustada, aqui he decidido solamente hacer una descarga del archivo, asi que la linea debajo del comando "ui=fluidPage(" debe ser descomentada para que shiny pueda descargar el set de datos y trabajarlo.

Dejo el link de Github con el codigo fuente y el link de shiny.io para que puedan hecharle un ojo a la aplicacion

```
shinyApp(

  ui = fluidPage(
#  download.file('https://opendata.ecdc.europa.eu/covid19/casedistribution/csv','descarga.csv'),
```

```
  # Application title
  titlePanel("Seleccione los paises que desea graficar"),
  selectInput(
      "Ciudades",
      "Variable:",
      unique(as.character(DataCovid <- read.csv("descarga.csv")$countriesAndTerritories)),
      multiple = TRUE
  ),


      mainPanel(
          plotOutput('plotConfirmados'),
          plotOutput('plotMuertes')
  )
),

server = function(input, output) {
  DataCovid <- read.csv("descarga.csv")
  DataCovid$dateRep<-as.Date(DataCovid$dateRep,format='%d/%m/%Y')
  DataCovid$countriesAndTerritories<-as.character(DataCovid$countriesAndTerritories)
  # unique(DataCovid$countriesAndTerritories)
  data<-DataCovid
  # countrie<-'Guatemala'
  # class(DataCovid)
  #?data.frame
  # data<-DataCovid
  totalCases<-data[0,0]
  countries<-unique(data$countriesAndTerritories)

  for(countrie in countries)  {

      dataAux<-data[which(data$countriesAndTerritories==countrie),]
      dataAux<-dataAux[order(dataAux$dateRep),]
      dataAux<-cbind(dataAux,0,0)

      names<-c(colnames(data),'TotalCase','totalDeath')

      colnames(dataAux)<-names
      rownames(dataAux)<-c(1:nrow(dataAux))

      for( i in 1:nrow(dataAux)){
          # i=85
          if(i==1) dataAux$totalDeath[i]=dataAux$deaths[i] else dataAux$totalDeath[i]=dataAux$deaths[
      }
      for( i in 1:nrow(dataAux)){
          # i=85
          if(i==1) dataAux$TotalCase[i]=dataAux$cases[i] else dataAux$TotalCase[i]=dataAux$cases[i]+da
      }
      dataAux<- dataAux[which(dataAux$TotalCase!=0),]
      dataAux$dayPerInit<-c(1:nrow(dataAux))
      totalCases<-rbind(totalCases,dataAux)
```

```
    }
    totalCases<-totalCases[-which(totalCases$TotalCase==0),]
    output$plotConfirmados <- renderPlot({

        # generate bins based on input$bins from ui.R
        dataset<-  totalCases
        # dataset<-resultTotales
        # cty<-'Italy'
        # name<-unique(as.character(resultTotales$countriesAndTerritories))[1:3]

        countriData<-dataset[0,0]
        for(cty in input$Ciudades){
            dataAux<-dataset[which(dataset$countriesAndTerritories==cty),]
            countriData<-rbind(countriData,dataAux)
            # View(countriData)
        }
        gg1<-ggplot(countriData,aes(x=dayPerInit,y=TotalCase))
        gg1<-gg1+geom_line(aes(color=countriesAndTerritories,linetype=countriesAndTerritories))
        gg1

    })
    output$plotMuertes <- renderPlot({

        # generate bins based on input$bins from ui.R
        dataset<-  totalCases
        # dataset<-resultTotales
        # cty<-'Italy'
        # name<-unique(as.character(resultTotales$countriesAndTerritories))[1:3]

        countriData<-dataset[0,0]
        for(cty in input$Ciudades){
            dataAux<-dataset[which(dataset$countriesAndTerritories==cty),]
            countriData<-rbind(countriData,dataAux)
            # View(countriData)
        }
        gg1<-ggplot(countriData,aes(x=dayPerInit,y=totalDeath))
        gg1<-gg1+geom_line(aes(color=countriesAndTerritories,linetype=countriesAndTerritories))
        gg1

    })
  },

  options = list(height = 500)
)
```

Shiny applications not supported in static R Markdown documents