```
library(shiny)
library(shinydashboard)
# https://fontawesome.com/search?q=network&o=r&m=free
ui <-dashboardPage(
  dashboardHeader(title = 'Warm up'),
  dashboardSidebar(

    sidebarMenu(
      menuItem("DT", tabName = "dt", icon = icon("dashboard")),
      menuItem("MAP", tabName = "map", icon = icon("map")),
      menuItem("Network", tabName = "network", icon = icon("circle-nodes"))

    )



  ),
  dashboardBody(
    tabItems(
      tabItem(tabName = "dt", h1('Table')),
      tabItem(tabName = "map", h1('Map')),
      tabItem(tabName = "network", h1('Network'))
    )
  )
)

server <- function(input, output, session) {

}

shinyApp(ui, server)
```

```
# https://yihui.shinyapps.io/DT-selection/
 ui <- dashboardPage(
  dashboardHeader(title = 'DT row selected'),
  dashboardSidebar(),
  dashboardBody(
    dataTableOutput('dtout'),
    textOutput('selected_rows'),
    plotlyOutput('car_plot')
  )
)
 server <- function(input, output, session) {
    output$dtout = DT::renderDataTable(mtcars)
  output$selected_rows = renderPrint(input$dtout_rows_selected)

  df <- reactive(mtcars[input$dtout_rows_selected,])
  output$car_plot <- renderPlotly({
    p <- plot_ly(df(), x=df()$wt, y=df()$mpg, z=df()$hp,
                 type="scatter3d", mode="markers",
                 color=df()$drat, size=df()$qsec) %>%
      layout(scene=list(
        xaxis = list(title = "Weight (1000 lbs)"),
        yaxis = list(title = "miles per gallon"),
        zaxis = list(title = "Gross horsepower)"))
```

```
library(leaflet)
library(rgdal)
library(shiny)

ui <- dashboardPage(
  dashboardHeader(title = 'Leaflet'),
  dashboardSidebar(),
  dashboardBody(
   leafletOutput('austriamap'),
   textOutput('yourmouseon'),
   textOutput('yourlastclick'),
   textOutput('mapzoomlevel')
  )
)
server <- function(input, output, session) {
  austria <- rgdal::readOGR("austria-with-regions_.geojson")

  output$austriamap <- renderLeaflet({
    leaflet(austria) %>%
      addPolygons(label=~name, layerId = ~name) %>%
      addTiles()
  })

  output$yourmouseon <- renderText( paste0('Your mouse on: ',
input$austriamap_shape_mouseover$id ))

  output$yourlastclick <- renderText( paste0('Your last click was: ',
input$austriamap_shape_click$id))
  output$mapzoomlevel <- renderText(paste0('the zoom level of your map is:',
input$austriamap_zoom ))
```

```
}



library(shiny)
library(networkD3)
library(shinydashboard)

ui <- dashboardPage(
  dashboardHeader(title = 'Network'),
  dashboardSidebar(),
  dashboardBody(
    textOutput('selected_node_out'),
    forceNetworkOutput('forcenetwork_out'),
    sankeyNetworkOutput('sankey_out')
  )
)

server <- function(input, output, session) {
  data(MisLinks)
  data(MisNodes)
  MisNodes$name <- as.character(MisNodes$name)
  MyClickScript <- 'Shiny.onInputChange("selected_node",d.index)'

  # Load energy projection data
  URL <- paste0(
    "https://cdn.rawgit.com/christophergandrud/networkD3/",
    "master/JSONdata/energy.json")
  Energy <- jsonlite::fromJSON(URL)

  output$forcenetwork_out <- renderForceNetwork({
```

```
  forceNetwork(Links = MisLinks, Nodes = MisNodes,
           Source = "source", Target = "target",
           Value = "value", NodeID = "name",
           Group = "group", opacity = 0.8, clickAction = MyClickScript)
  })
 # zoom = T,fontSize = 20,opacityNoHover = 0.9

 output$selected_node_out <- renderText(MisNodes$name[input$selected_node])
  output$sankey_out <- renderSankeyNetwork({
  sankeyNetwork(Links = Energy$links, Nodes = Energy$nodes, Source = "source",
             Target = "target", Value = "value", NodeID = "name",
             units = "TWh", fontSize = 12, nodeWidth = 30)
 })

}

shinyApp(ui, server)
```

```
library(dygraphs)
lungDeaths <- cbind(mdeaths, fdeaths)
dygraph(lungDeaths)

source('functions.R')
library(data.table)
library(TTR)
library(httr)
library(rtsdata)
library(DT)
df <- get_data_by_ticker_and_date('TSLA', Sys.Date()-360, Sys.Date())

lungDeaths <- cbind('close'=ts(df$close), 'open'= ts(df$open))
dygraph(lungDeaths)
```