Library(plotly)

Scatter plot

P<-
plot_ly(data=iris,x=~Sepal.Length,y=~Petal.Length
)

```
add_annotations(p,text="length",data=NULL,inherit=TRUE)
```

```
show the table for plotly object
add_table(p,rownames=TRUE,data=NULL,inherit=TRUE)
```

with color

```
plot_ly(economics,x=~date,y=~unempmed,color=I("red"),showlegend=FALSE)
```

To show using lines rather than scatter p lot

```
p<-plot_ly(economics,x=~date,y=~uempmed,color=I("red"),showlegend=FALSE)
add_lines(p)

add_markers(p) %sscatter points%
```

Customized mode to traces

```
add_trace(p, type = "scatter", mode = "markers+lines")
```

3d visualization

```
p<-plot_ly(economics,x=~date,y=~uempmed,z=~psavert,color=I("red"),showlege
nd=FALSE)
```

```
Multiple attributes in 2 D
```

```
trace_0 <- rnorm(100, mean = 5)
trace_1 <- rnorm(100, mean = 0)
trace_2 <- rnorm(100, mean = -5)
x <- c(1:100)
```

```
data <- data.frame(x, trace_0, trace_1, trace_2)


p <- plot_ly(data, x = ~x) %>%
  add_trace(y = ~trace_0, name = 'trace 0',mode = 'lines') %>%
  add_trace(y = ~trace_1, name = 'trace 1', mode = 'lines+markers') %>%
  add_trace(y = ~trace_2, name = 'trace 2', mode = 'markers')
```

Using different symbols

```
p <- plot_ly(data = iris, x = ~Sepal.Length, y = ~Petal.Length, type =
'scatter',
  mode = 'markers', symbol = ~Species, symbols = c('circle','x','o'),
  color = I('black'), marker = list(size = 10))
```

p

Color, size, hover

```
d <- diamonds[sample(nrow(diamonds), 1000), ]

p <- plot_ly(
  d, x = ~carat, y = ~price,
  # Hover text:
  text = ~paste("Price: ", price, '$<br>Cut:', cut),
  color = ~carat, size = ~carat
)
```

Table

```
plot_ly(type='table',header=list(values=list(list('<b
>Reg Num</b>'),list('<b>Name</b>')),line=list(color='
#508794'),fill=list(color='#112344'),align=c('left','
center'),font=list(color='white',size=12)),cells=list
(values=list(c('R1','R2','R3'),c('X1','X2','X3')),fon
t=list(color='#508794',size=15)))
```

Table from dataframe

```
headerValues <- list()
for (i in (0:ncol(mtcars))) {
   name <- names(mtcars)[i]
   headerValues[i] <- name
}

headerValues <- append(headerValues, "<b>Cars</b>", after = 0)

cellValues <- list()
for (i in (0:ncol(mtcars))) {
  row <- mtcars[i]
  cellValues[i] <- row
}

cellValues <- append(cellValues, list(rownames(mtcars)), after = 0)

p <- plot_ly(
  type = 'table',
  header = list(
    values = headerValues,
  align = c('left', rep('center', ncol(mtcars))),
  line = list(width = 1, color = 'black'),
  fill = list(color = 'rgb(235, 100, 230)'),
  font = list(family = "Arial", size = 14, color = "white")
  ),
  cells = list(
    values = cellValues,
    align = c('left', rep('center', ncol(mtcars))),
    line = list(color = "black", width = 1),
    fill = list(color = c('rgb(235, 193, 238)', 'rgba(228, 222, 249,
0.65)')),
    font = list(family = "Arial", size = 12, color = c("black"))
  ))
```

Tables : using functions, plotly

```
createTable <- function(df, tableHeight = 50){

  # Create the value parameters
  # Headers
  nms <- lapply(names(df), function(x){
    return(paste0("<b>", x, "</b>"))
  })

  nms <- append(nms, "<b>Rows</b>", after = 0)
  headerValues <- lapply(nms, function(x){return(list(x))})

  # Cell values
  names(df) <- NULL
  cellValues <- apply(df, 2, function(x){return(list(x))})
  cellValues <- lapply(cellValues, function(x){return(unlist(x))})

  cellValues <- append(cellValues, list(rownames(df)), after = 0)
  # Create the list to pass to plot_ly()
  header <- list(
    values = headerValues,

    # Formatting
    line = list(color = '#DFE8F3'),
    align = c('left', rep('center', ncol(df))),
    font = list(color = '#ffffff', size = 16),
    fill = list(color = '#999999')
  )

  cells <- list(
    values = cellValues,

    # Formatting
    line = list(color = '#DFE8F3'),
    align = c('left', rep('right', ncol(df))),
    font = list(color = c('#262626'), size = 14),
    fill = list(color = c("#d9d9d9", rep("#ffe6cc", ncol(df)))),
    height = tableHeight
  )

  # Create table in plotly
  p <- plot_ly(
    type = "table",
    header = header,
    cells = cells,
    width = 1200,
    height = 1600) %>%

    layout(xaxis = list(zeroline = F, showgrid = F, showticklabels = F),
           yaxis = list(zeroline = F, showgrid = F, showticklabels = F))

  return(p)
}

p <- createTable(mtcars)
p
```