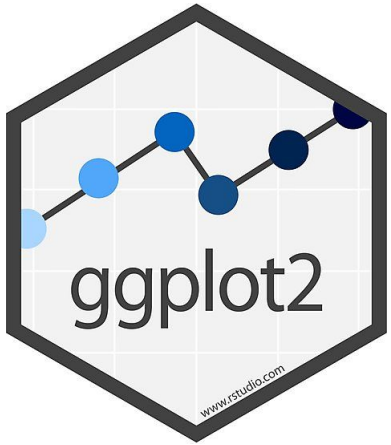
```
library(dplyr)

rladies_global %>%
  filter(city == 'Buenos Aires')
```

Ladies Buenos Aires

Romina Mendez
@r0mymendez

GGPLOT2



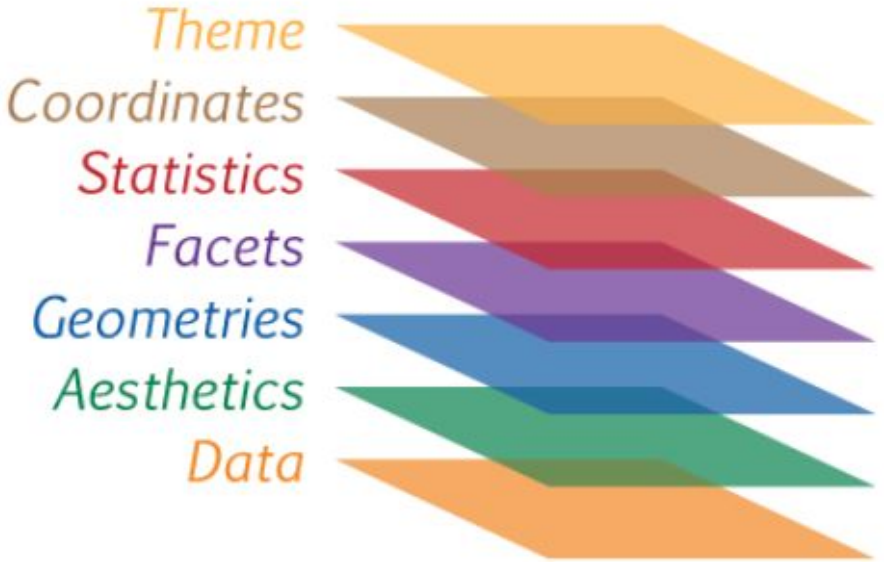

```
library(ggplot2)
```

El paquete ggplot2 utiliza un enfoque altamente modular para los gráficos, que le permite construir y personalizar las gráficas más fácilmente.

El paquete fue creado por Hadley Wickham, quien también escribió ggplot2: Elegant Graphics for Data Analysis (Springer, 2009). Dicho libro explica el paradigma detrás de ggplot2 y cómo usar las funciones del paquete.

GGPLOT2: CAPAS

Las capas de una gráfica ggplot pueden ser las siguientes:



GGPLOT2: CAPAS

Cada capa tiene una funcionalidad diferentes que se van adicionando para poder realizar la gráfica:

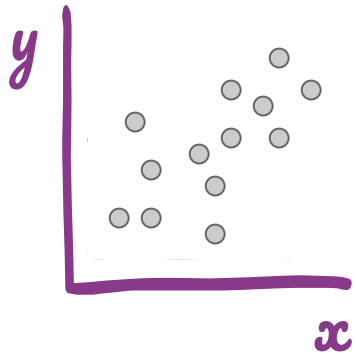
- **DATA:** Es la primer capa y se define el set de datos que se va utilizar
- **AESTHETICS:** En la siguiente capa nos permite especificar las características, las columnas (es decir, la dimensión) que queremos trazar.
  Observación: estas dos capas no grafican nada, solo realizan la selección de los datos y los ejes.
- **GEOMETRICS:** En esta capa definimos las formas que pretendemos usar para presentar los datos usando ggplot. Después de agregar esta capa, el ggplot sabe cómo mostrar los datos.

GGPLOT2: CAPAS

Cada capa tiene una funcionalidad diferentes que se van adicionando para poder realizar la gráfica:

- **FACETS:** En esta capa se permite poder realizar varios gráficos generando una agrupación en base a una variable categórica.
- **STATISTICS**: En esta capa se pueden agregar alguna medida estadísticas a un gráfico.
- **COORDINATES:** A menudo se usa para aplicar el límite en el eje x o el eje y para jugar con la relación x vs y, por lo tanto, personalizar la imagen según sea necesario.
- **THEME:** En esta capa se permite aplicar diferentes estilos, los mismos se pueden personalizar pero ggplot2 tiene algunos definidos en funciones.

{ Ahora vamos a crear nuestro primer gráfico con ggplot2 }

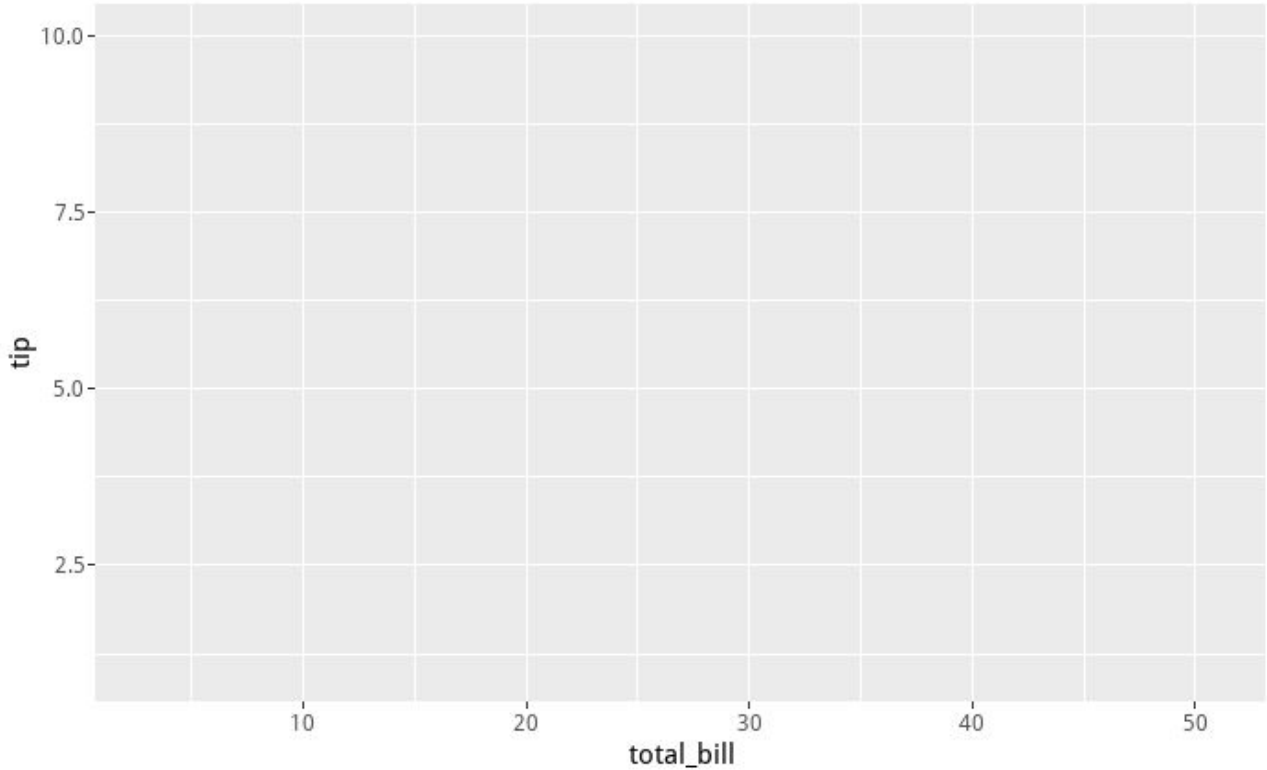


{ En primer lugar creamos las capas de datos y de estéticas }

```
library(ggplot2)
library(reshape2)

df.tips=tips

ggplot(     df.tips  ,      aes(x=total_bill  ,  y=tip)    )
```

aesthetics

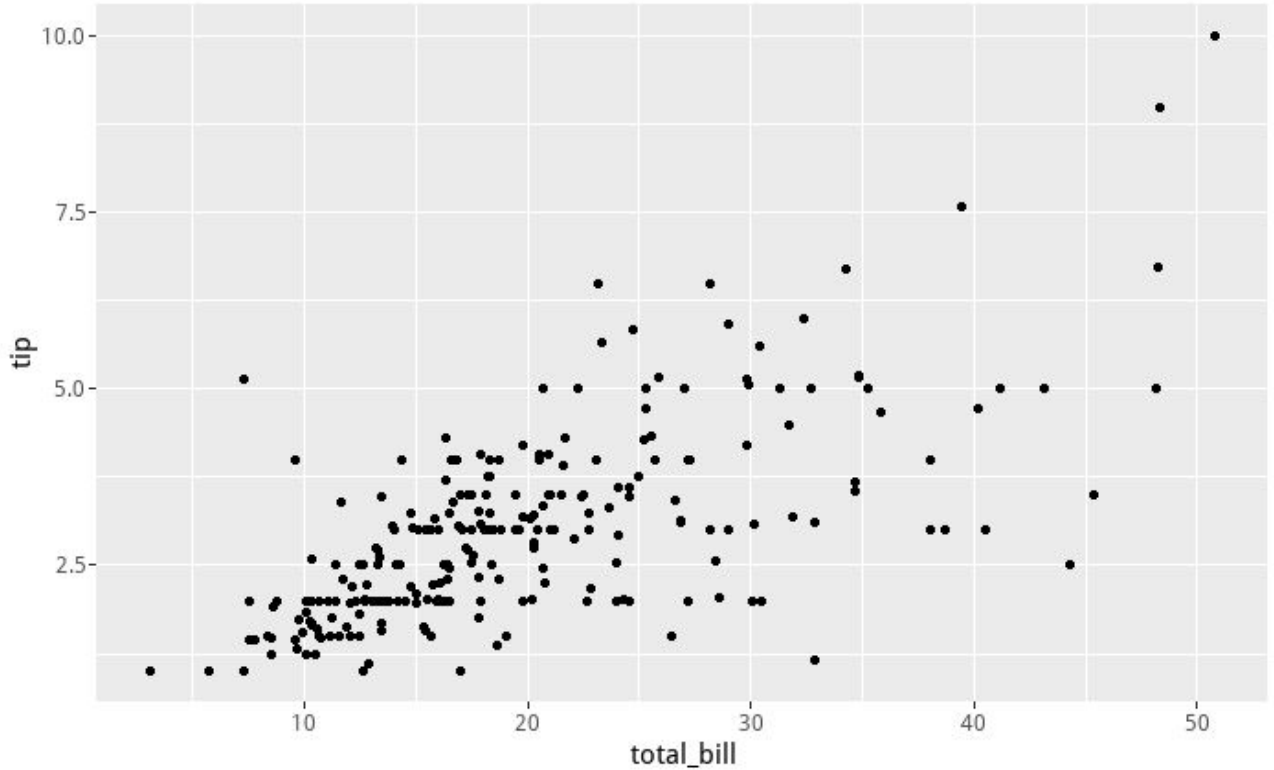
Dataset


10.0
7.5
5.0
2.5
tip
10
20
30
40
50
total_bill

{ Ahora agreguemos la siguiente capa, que es el tipo de gráfico a realizar }

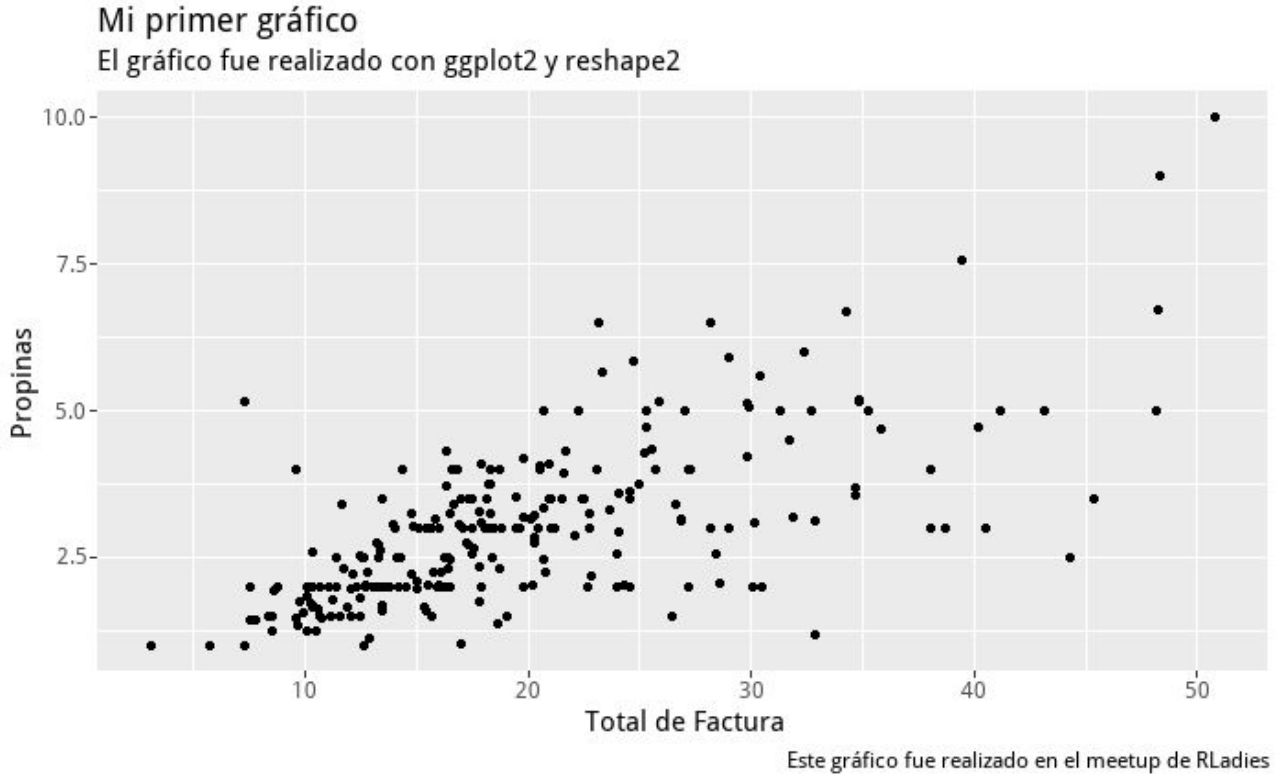
```
library(ggplot2)
library(reshape2)

ggplot(  df.tips  ,  aes(x=total_bill  ,  y=tip)  ) +
  geom_point()
```

geometria o tipo de grafico


10.0
7.5
5.0
2.5
tip
10
20
30
40
50
total_bill

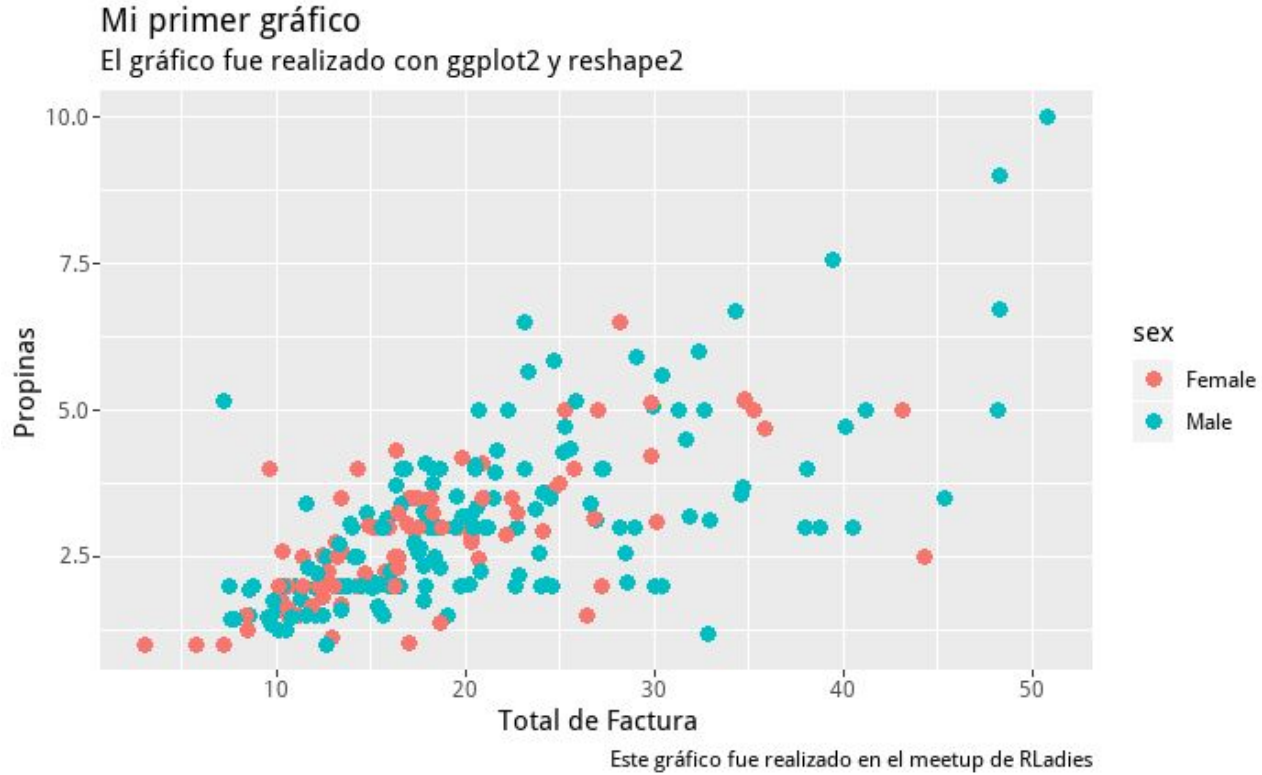
{ Podemos agregar las leyendas de títulos, subtítulos, pie y ejes a nuestro gráfico }

```
library(ggplot2)
library(reshape2)

ggplot(     df.tips  ,      aes(x=total_bill  ,  y=tip)    )  +
  geom_point() +

  labs ( title='Mi primer gráfico',
       subtitle='El gráfico fue realizado con ggplot2 y reshape2',
       x='Total de Factura',
       y='Propinas',
       caption='Este gráfico fue realizado en el meetup de RLadies')
       )
```


Mi primer gráfico
El gráfico fue realizado con ggplot2 y reshape2
Propinas
10.0
7.5
5.0
2.5
10
20
30
40
50
Total de Factura
Este gráfico fue realizado en el meetup de RLadies

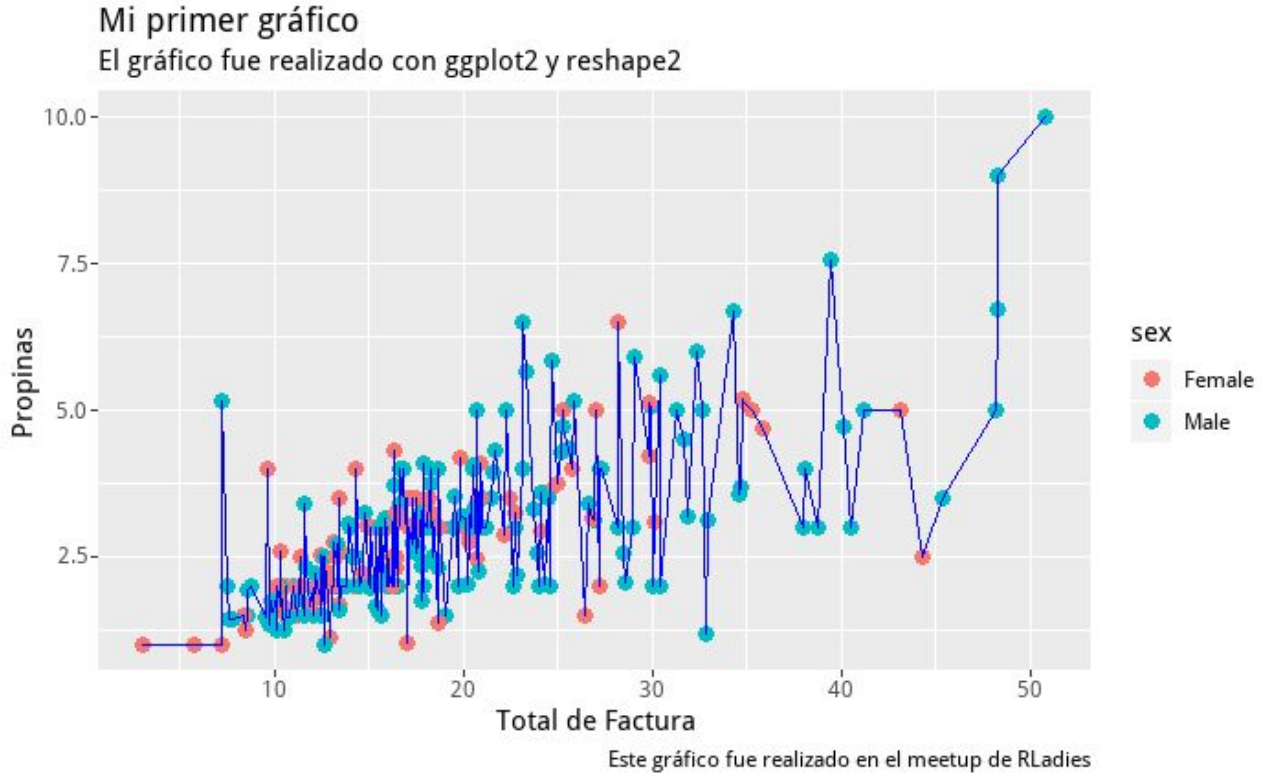
{ Ahora podemos cambiar el tamaño y pintar nuestros puntos en base a otra variable}

```
library(ggplot2)
library(reshape2)

ggplot( df.tips ,aes(x=total_bill,y=tip, color=sex) )+
  geom_point ( size=3 )+
  labs(title='Mi primer gráfico',
       subtitle='El gráfico fue realizado con ggplot2 y reshape2',
       x='Total de Factura',
       y='Propinas',
        caption=paste0('Este gráfico fue realizado en el meetup de RLadies')
       )
```


Mi primer gráfico
El gráfico fue realizado con ggplot2 y reshape2
Propinas
10.0
7.5
5.0
2.5
10
20
30
40
50
Total de Factura
Este gráfico fue realizado en el meetup de RLadies
sex
Female
Male

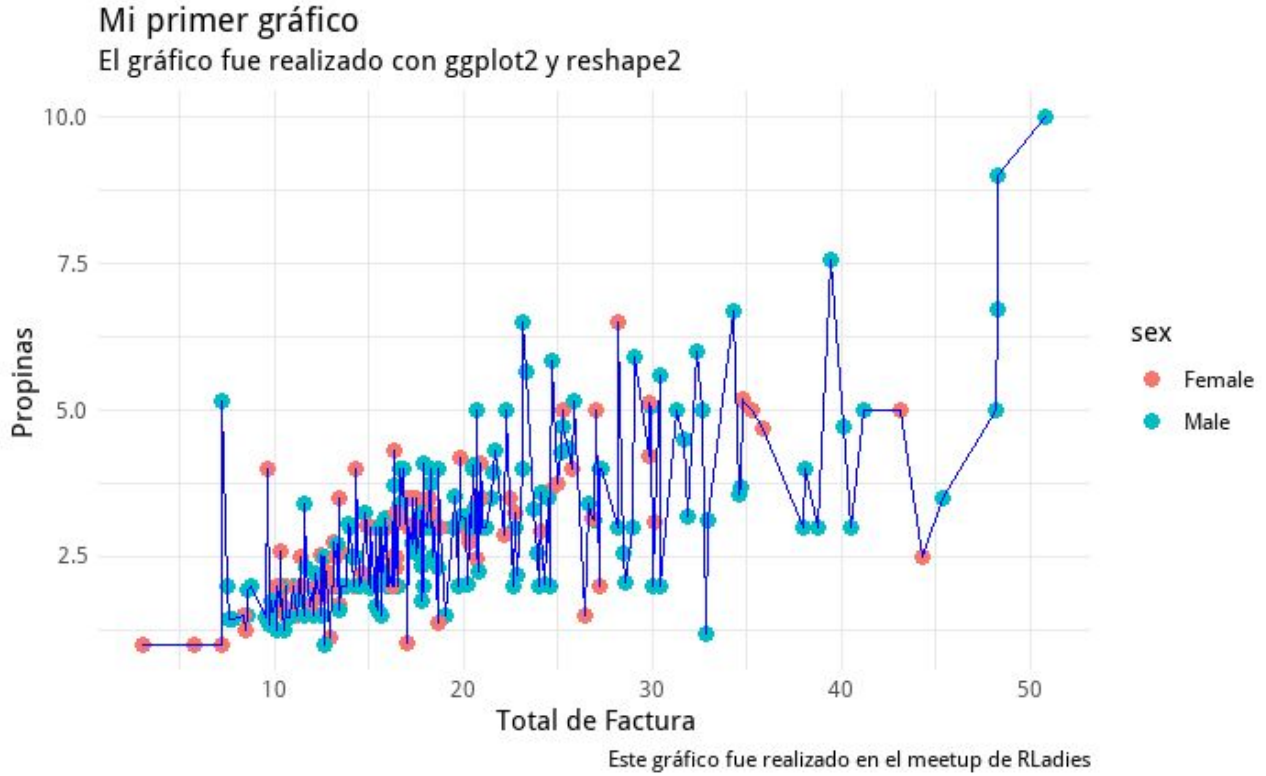
{ Ggplot2 nos permite agregar más de un tipo de geometría, en este caso utilizaremos geom_line }

```
library(ggplot2)
library(reshape2)

ggplot(tips,aes(x=total_bill,y=tip, color=sex) ) +
  geom_point ( size=3 ) +
  geom_line ( color='blue' ) +
  labs(title='Mi primer gráfico',
       subtitle='El gráfico fue realizado con ggplot2 y reshape2',
       x='Total de Factura',
       y='Propinas',
       caption='Este gráfico fue realizado en el meetup de RLadies')
```


Mi primer gráfico
El gráfico fue realizado con ggplot2 y reshape2
Propinas
10.0
7.5
5.0
2.5
10
20
30
40
50
Total de Factura
Este gráfico fue realizado en el meetup de RLadies
sex
Female
Male

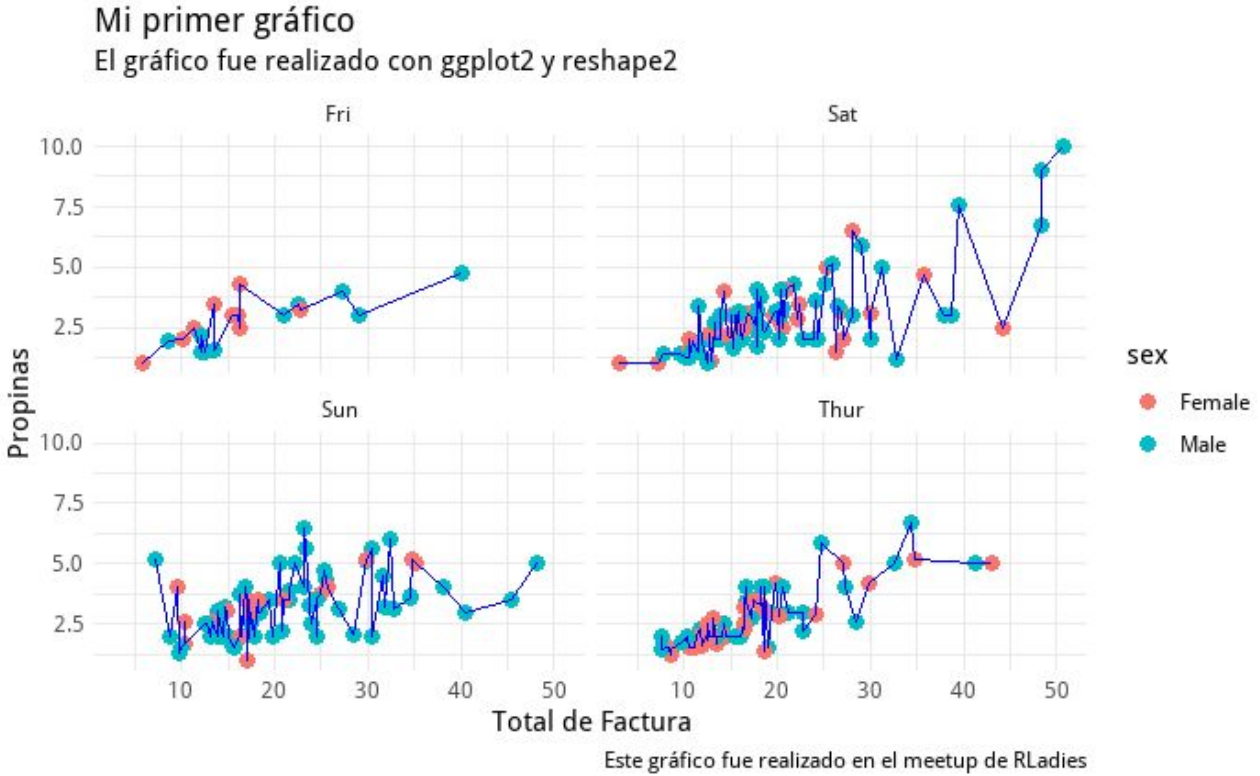
{ Otra capa que podemos modificar fácilmente es la de theme y permite personalizar aún más nuestro gráfico}

```
library(ggplot2)
library(reshape2)

ggplot(tips,aes(x=total_bill,y=tip, color=sex) ) +
  geom_point ( size=3 ) +  geom_line ( color='blue' ) +
  labs(title='Mi primer gráfico',
       subtitle='El gráfico fue realizado con ggplot2 y reshape2',
       x='Total de Factura',
       y='Propinas',
        caption='Este gráfico fue realizado en el meetup de RLadies') +
       theme_minimal()
```

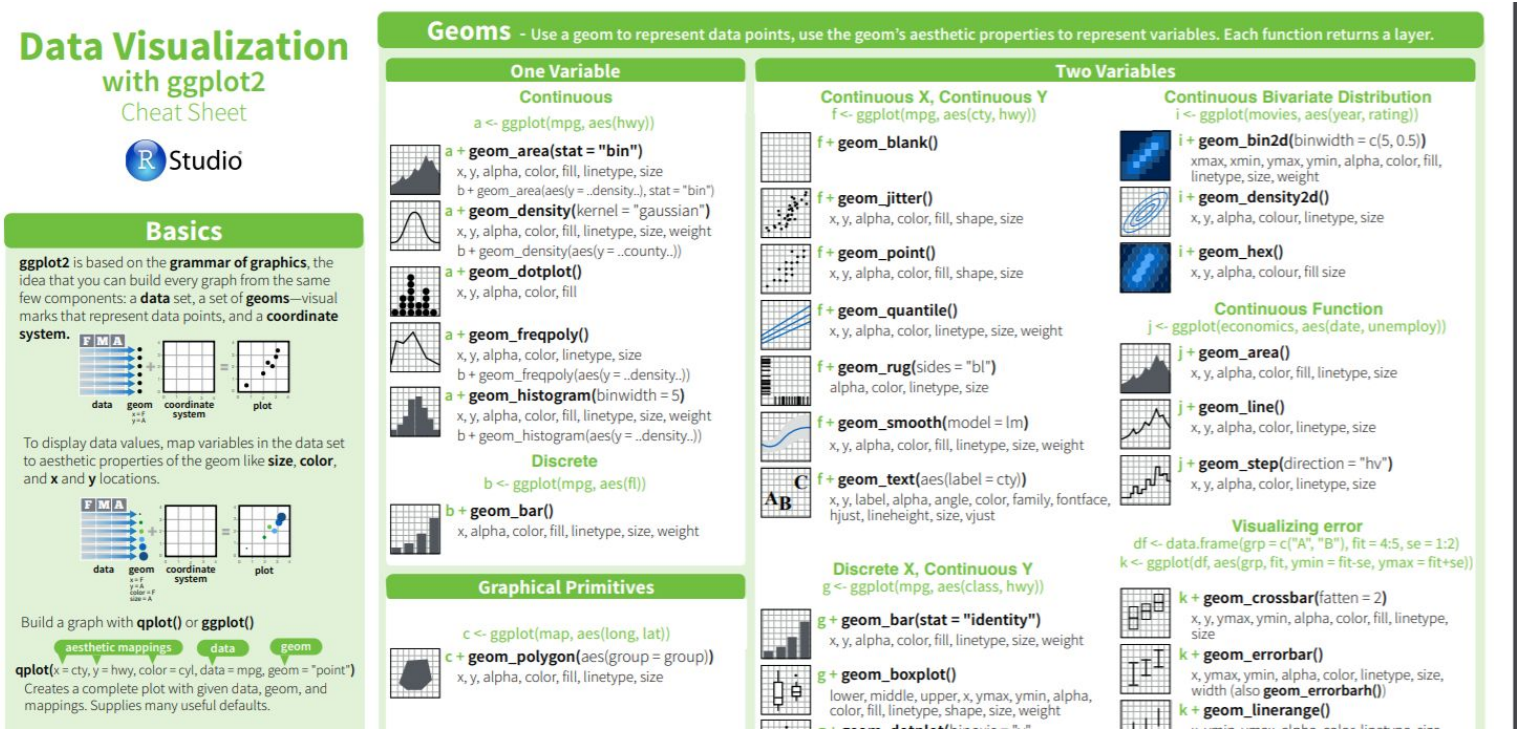

Mi primer gráfico
El gráfico fue realizado con ggplot2 y reshape2
Propinas
10.0
7.5
5.0
2.5
10
20
30
40
50
Total de Factura
Este gráfico fue realizado en el meetup de RLadies
sex
Female
Male

{ La capa denominada facet permite crear paneles para dividir datos y poder hacer comparativas de los mismos }

```
library(ggplot2)
library(reshape2)

ggplot(tips,aes(x=total_bill,y=tip, color=sex) ) +
  geom_point ( size=3 ) +  geom_line ( color='blue') +
  labs(title='Mi primer gráfico',
       subtitle='El gráfico fue realizado con ggplot2 y reshape2',
       x='Total de Factura',
       y='Propinas',
        caption='Este gráfico fue realizado en el meetup de RLadies') +
       theme_minimal ()  +
       facet_wrap (.~day)
```


Mi primer gráfico
El gráfico fue realizado con ggplot2 y reshape2
Fri
Sat
Sun
Thur
Propinas
10.0
7.5
5.0
2.5
10
20
30
40
50
Total de Factura
sex
Female
Male
Este gráfico fue realizado en el meetup de RLadies

{ Ggplot2 Cheat Sheet }



Data Visualization with ggplot2 Cheat Sheet

RStudio

Basics

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same few components: a **data** set, a set of **geoms**—visual marks that represent data points, and a **coordinate system.**

To display data values, map variables in the data set to aesthetic properties of the geom like **size**, **color**, and **x** and **y** locations.

Build a graph with **qplot()** or **ggplot()**

aesthetic mappings · data · geom

qplot(x = cty, y = hwy, color = cyl, data = mpg, geom = "point")
Creates a complete plot with given data, geom, and mappings. Supplies many useful defaults.

Geoms - Use a geom to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

One Variable

Continuous

a <- ggplot(mpg, aes(hwy))

a + **geom_area(stat = "bin")**
x, y, alpha, color, fill, linetype, size
b + geom_area(aes(y = ..density..), stat = "bin")

a + **geom_density**(kernel = "gaussian")
x, y, alpha, color, fill, linetype, size, weight
b + geom_density(aes(y = ..county..))

a + **geom_dotplot()**
x, y, alpha, color, fill

a + **geom_freqpoly()**
x, y, alpha, color, linetype, size
b + geom_freqpoly(aes(y = ..density..))

a + **geom_histogram**(binwidth = 5)
x, y, alpha, color, fill, linetype, size, weight
b + geom_histogram(aes(y = ..density..))

Discrete

b <- ggplot(mpg, aes(fl))

b + **geom_bar()**
x, alpha, color, fill, linetype, size, weight

Graphical Primitives

c <- ggplot(map, aes(long, lat))

c + **geom_polygon**(aes(group = group))
x, y, alpha, color, fill, linetype, size

Two Variables

Continuous X, Continuous Y

f <- ggplot(mpg, aes(cty, hwy))

f + **geom_blank()**

f + **geom_jitter()**
x, y, alpha, color, fill, shape, size

f + **geom_point()**
x, y, alpha, color, fill, shape, size

f + **geom_quantile()**
x, y, alpha, color, linetype, size, weight

f + **geom_rug**(sides = "bl")
alpha, color, linetype, size

f + **geom_smooth**(model = lm)
x, y, alpha, color, fill, linetype, size, weight

f + **geom_text**(aes(label = cty))
x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

Discrete X, Continuous Y

g <- ggplot(mpg, aes(class, hwy))

g + **geom_bar**(stat = "identity")
x, y, alpha, color, fill, linetype, size, weight

g + **geom_boxplot()**
lower, middle, upper, x, ymax, ymin, alpha, color, fill, linetype, shape, size, weight

Continuous Bivariate Distribution

i <- ggplot(movies, aes(year, rating))

i + **geom_bin2d**(binwidth = c(5, 0.5))
xmax, xmin, ymax, ymin, alpha, color, fill, linetype, size, weight

i + **geom_density2d()**
x, y, alpha, colour, linetype, size

i + **geom_hex()**
x, y, alpha, colour, fill size

Continuous Function

j <- ggplot(economics, aes(date, unemploy))

j + **geom_area()**
x, y, alpha, color, fill, linetype, size

j + **geom_line()**
x, y, alpha, color, linetype, size

j + **geom_step**(direction = "hv")
x, y, alpha, color, linetype, size

Visualizing error

df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)
k <- ggplot(df, aes(grp, fit, ymin = fit-se, ymax = fit+se))

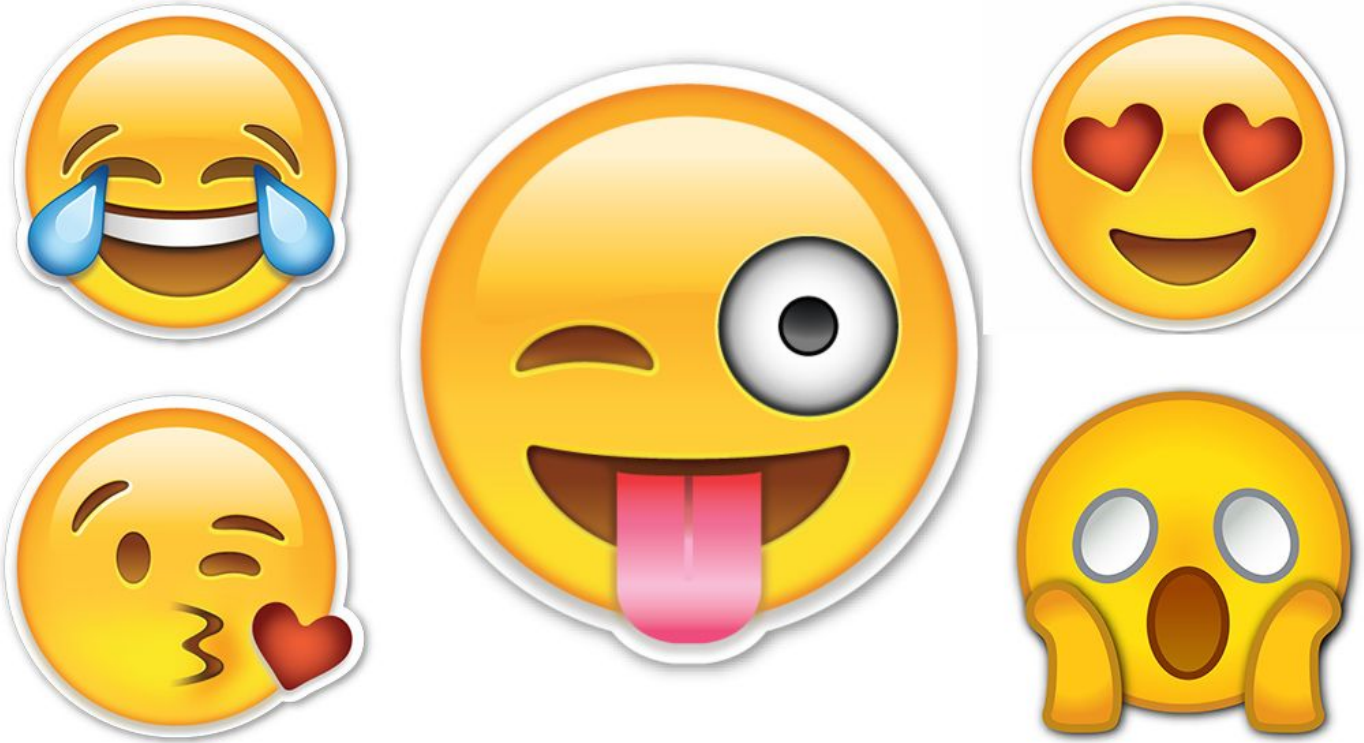
k + **geom_crossbar**(fatten = 2)
x, y, ymax, ymin, alpha, color, fill, linetype, size

k + **geom_errorbar()**
x, ymax, ymin, alpha, color, linetype, size, width (also **geom_errorbarh()**)

k + **geom_linerange()**

https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf

EMOJIS



{ Para comenzar a utilizar emojis debemos buscar con alguna descripción los emojis, para lo cual utilizamos la función search_emoji() }

```
library(emojifont)
```

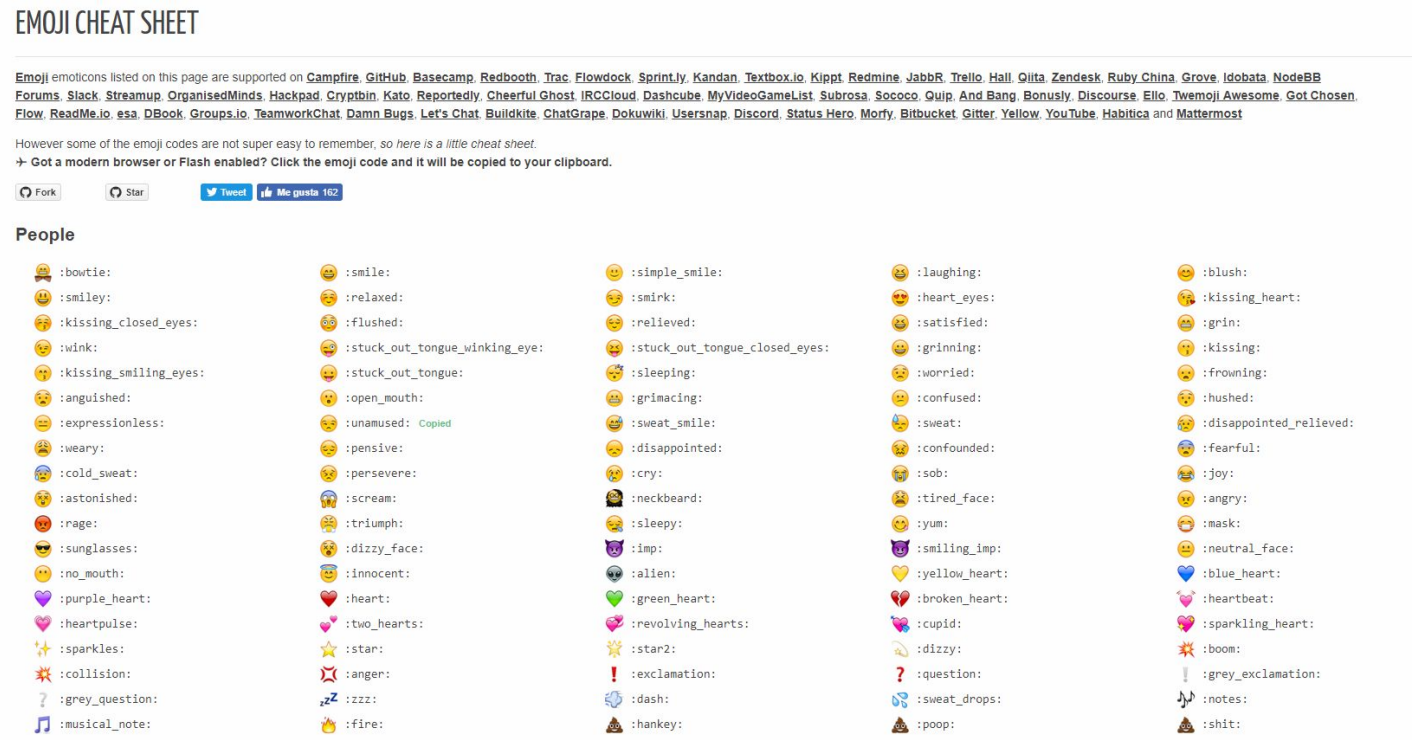
```
# Buscar emojis
search_emoji('smile')
```

```
## [1] "smiley" "smile" "sweat_smile" "smiley_cat"  "smile_cat"
```

:smile: :sweat_smile: :smile_cat:

:smiley: :smiley_cat:

{ Otra opción es buscar en la cheat sheet de emoji el alias }



EMOJI CHEAT SHEET

Emoji emoticons listed on this page are supported on Campfire, GitHub, Basecamp, Redbooth, Trac, Flowdock, Sprint.ly, Kandan, Textbox.io, Kippt, Redmine, JabbR, Trello, Hall, Qiita, Zendesk, Ruby China, Grove, Idobata, NodeBB Forums, Slack, Streamup, OrganisedMinds, Hackpad, Cryptbin, Kato, Reportedly, Cheerful Ghost, IRCCloud, Dashcube, MyVideoGameList, Subrosa, Sococo, Quip, And Bang, Bonusly, Discourse, Ello, Twemoji Awesome, Got Chosen, Flow, ReadMe.io, esa, DBook, Groups.io, TeamworkChat, Damn Bugs, Let's Chat, Buildkite, ChatGrape, Dokuwiki, Usersnap, Discord, Status Hero, Morfy, Bitbucket, Gitter, Yellow, YouTube, Habitica and Mattermost

However some of the emoji codes are not super easy to remember, *so here is a little cheat sheet.*

✈ **Got a modern browser or Flash enabled? Click the emoji code and it will be copied to your clipboard.**

Fork Star Tweet Me gusta 162

People

:bowtie:	:smile:	:simple_smile:	:laughing:	:blush:
:smiley:	:relaxed:	:smirk:	:heart_eyes:	:kissing_heart:
:kissing_closed_eyes:	:flushed:	:relieved:	:satisfied:	:grin:
:wink:	:stuck_out_tongue_winking_eye:	:stuck_out_tongue_closed_eyes:	:grinning:	:kissing:
:kissing_smiling_eyes:	:stuck_out_tongue:	:sleeping:	:worried:	:frowning:
:anguished:	:open_mouth:	:grimacing:	:confused:	:hushed:
:expressionless:	:unamused: Copied	:sweat_smile:	:sweat:	:disappointed_relieved:
:weary:	:pensive:	:disappointed:	:confounded:	:fearful:
:cold_sweat:	:persevere:	:cry:	:sob:	:joy:
:astonished:	:scream:	:neckbeard:	:tired_face:	:angry:
:rage:	:triumph:	:sleepy:	:yum:	:mask:
:sunglasses:	:dizzy_face:	:imp:	:smiling_imp:	:neutral_face:
:no_mouth:	:innocent:	:alien:	:yellow_heart:	:blue_heart:
:purple_heart:	:heart:	:green_heart:	:broken_heart:	:heartbeat:
:heartpulse:	:two_hearts:	:revolving_hearts:	:cupid:	:sparkling_heart:
:sparkles:	:star:	:star2:	:dizzy:	:boom:
:collision:	:anger:	:exclamation:	:question:	:grey_exclamation:
:grey_question:	:zzz:	:dash:	:sweat_drops:	:notes:
:musical_note:	:fire:	:hankey:	:poop:	:shit:

https://www.webfx.com/tools/emoji-cheat-sheet/

{ Una vez seleccionado el alias, utiliza la función emoji para retornar el valor unicode }

```
library(emojifont)
```

```
search_emoji('smile')
```

```
## [1] "smiley" "smile" "sweat_smile" "smiley_cat" "smile_cat"
```

```
emoji('smiley_cat')
```

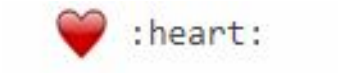
```
"\U0001f638"
```

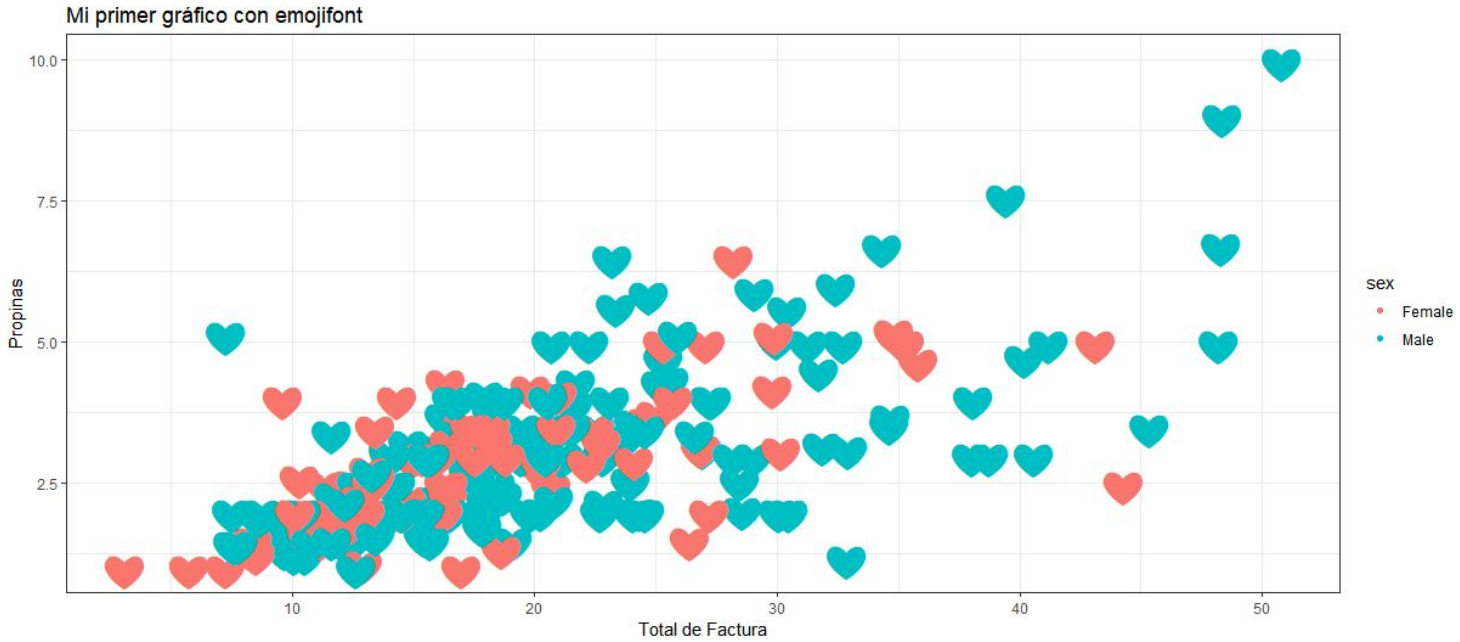
Emojifont

{ Ahora vamos hacer gráficos con EMOJIS! }

{ Necesitamos generar una columna en el dataset con el valor unicode del emoji seleccionado }

```
library(ggplot2)
library(reshape2)
library(emojifont)

tips=tips
tips$emoji=emoji("heart")

ggplot(tips,
    aes(x=total_bill,y=tip,color=sex,label=emoji))+
    geom_point()+
    geom_text(family="EmojiOne", size=12,show.legend = F) +
labs(title='Mi primer gráfico',
     x='Total de Factura',
     y='Propinas')
)
```



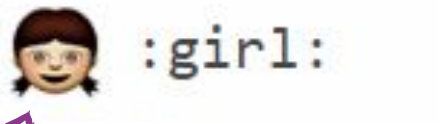

Mi primer gráfico con emojifont
Propinas
10.0
7.5
5.0
2.5
10
20
30
40
50
Total de Factura
sex
Female
Male

Emojifont

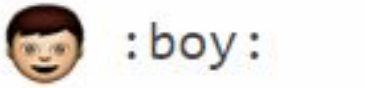
{ También podemos utilizar varios emoji en base al valor de una variable, utilizando la función ifelse() }

```
tips$emoji=ifelse(tips$sex=='Female',
                  emoji('girl'),
                  emoji('boy')
                 )
```

True



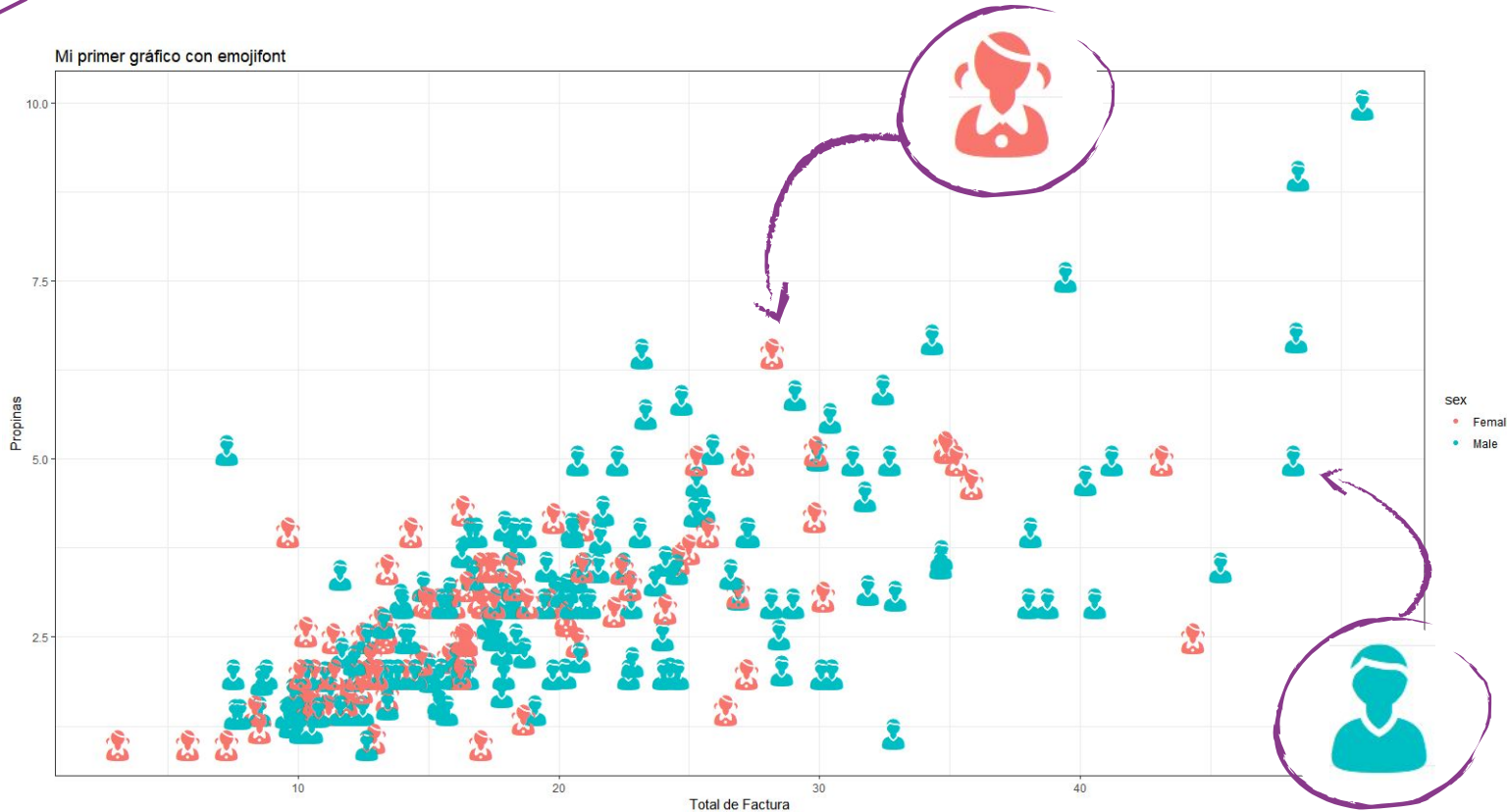
False



{ Ahora nuevamente generemos nuestro gráfico y veamos que la función va a graficar dos tipos diferentes de emoji en base al valor de la variable }

```
ggplot(tips,
    aes(x=total_bill,y=tip,color=sex,label=emoji))+
    geom_point()+
    geom_text(family="EmojiOne", size=12,show.legend = F) +
labs(title='Mi primer gráfico',
     x='Total de Factura',
     y='Propinas'))
```

Emojifont


Mi primer gráfico con emojifont
Propinas
10.0
7.5
5.0
2.5
10
20
30
40
Total de Factura
sex
Female
Male

Plotly es un paquete open source te permite crear gráficos interactivos en javascript.
Asimismo nos permite convertir fácilmente graficos ggplot a su versión interactiva en plotly, solo utilizando la funcion **ggplotly().**

Para más información:

👉 https://plotly-r.com/index.html
👉 https://plot.ly/r/

PLOTLY

```
library(ggplot2)
library(reshape2)
library(plotly)

p= ggplot(tips,aes(x=total_bill,y=tip,color=sex))+
        geom_point(size=3)+
        geom_line( color='blue' )+
        theme_minimal() +
        labs(title='Mi primer gráfico',
                subtitle='El gráfico fue realizado con
                                ggplot2 y  reshape2',
                 x='Total de Factura',
                 y='Propinas') +
          facet_wrap(.~day)
```

```
ggplotly(p)
```

¡¡Muchas Gracias!!