Chapter 8 : Themes

library(ggplot2)  
library(gridExtra)

## 8.1 Introduction

give you control over things likes fonts, ticks, panel strips, and backgrounds. you use code like *plot + theme(element.name = element function())*.

Theming system is composed of 4 main components.

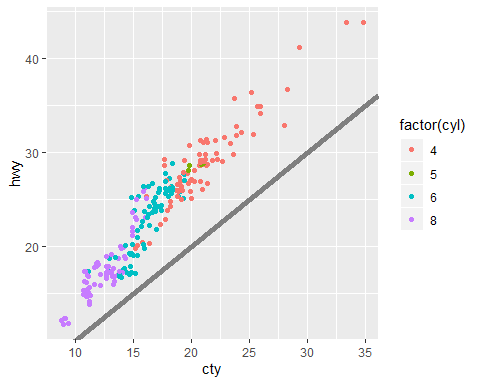
- Theme **elements** specify the non-data elements.For example, the *plot.title* element controls the appearance of the plot title; *axis.ticks.x*, the ticks on the x axis;

- Each element is associated with an **element function** which describes the visual properties of the element. For example, *element\_text()* sets the font size, colour and face of text elements like *plot.title*.

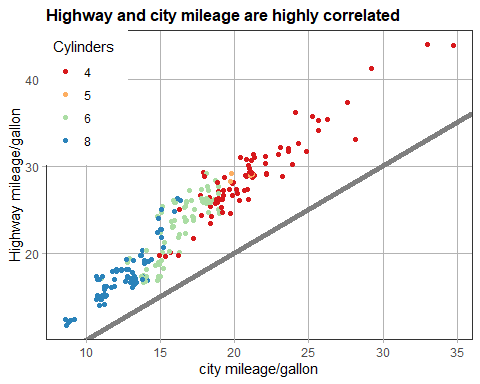
- The *theme()* function which allows you to override the default theme elements by calling element functions, like *theme(plot.title = element text(colour = “red”)).*

- Complete **themes**, like *theme\_grey()* set all of the theme elements to values designed to work together harmoniously.

base <- ggplot(mpg, aes(cty, hwy, color = factor(cyl))) +  
 geom\_jitter() +  
 geom\_abline(colour = "grey50", size = 2)  
base



base +   
 labs(  
 x = "city mileage/gallon",  
 y = "Highway mileage/gallon",  
 colour = "Cylinders",  
 title = "Highway and city mileage are highly correlated"  
 ) +  
 scale\_color\_brewer(type = "seq", palette = "Spectral") +  
 theme\_bw() +  
 theme(  
 plot.title = element\_text(face = "bold", size = 12),  
 legend.background = element\_rect(fill = "white", size = 4, colour = "white"),  
 legend.justification = c(0, 1),  
 legend.position = c(0, 1),  
 axis.ticks = element\_line(colour = "grey70", size = 0.2),  
 panel.grid.major = element\_line(colour = "grey70", size = 0.2),  
 panel.grid.minor = element\_blank()  
 )



## 8.2 Complete Themes

There are 7 other themes! (default : *theme\_grey()* )

- *theme\_bw()* : a variation on *theme\_grey()* that uses a white background and thin grey grid lines.

- *theme\_linedraw()* : A theme with only black lines of various widths on white backgrounds, reminiscent of a line drawing.

- *theme\_light()* : similar to *theme\_linedraw()* but with light grey lines and axes, to direct more attention towards the data.

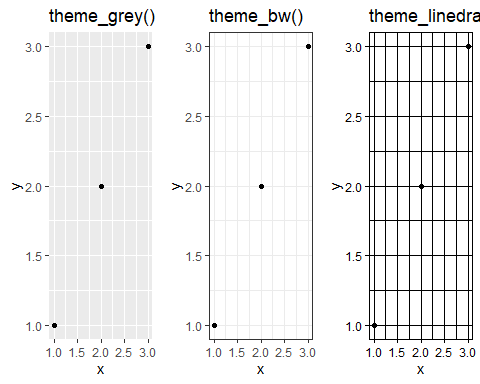
- *theme\_dark()* : the dark cousin of *theme\_light()*, with similar line sizes but a dark background. Useful to make thin coloured lines pop out.

- *theme\_minimal()* : A minimalistic theme with no background annotations.

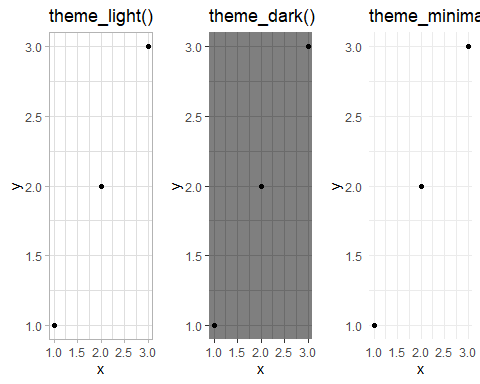
- *theme\_classic()* : A classic-looking theme, with x and y axis lines and no gridlines. = *theme\_void()*: A completely empty theme.

df <- data.frame(x = 1:3, y = 1:3)  
base <- ggplot(df, aes(x, y)) + geom\_point()  
  
b1 = base + theme\_grey() + ggtitle("theme\_grey()")  
b2 = base + theme\_bw() + ggtitle("theme\_bw()")  
b3 = base + theme\_linedraw() + ggtitle("theme\_linedraw()")  
b4 = base + theme\_light() + ggtitle("theme\_light()")  
b5 = base + theme\_dark() + ggtitle("theme\_dark()")  
b6 = base + theme\_minimal() + ggtitle("theme\_minimal()")  
b7 = base + theme\_classic() + ggtitle("theme\_classic()")  
b8 = base + theme\_void() + ggtitle("theme\_void()")

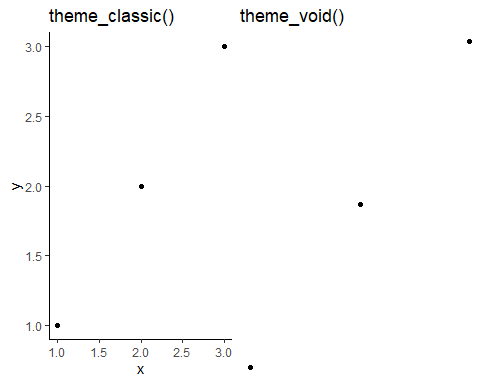
grid.arrange(b1,b2,b3, ncol = 3)



grid.arrange(b4,b5,b6, ncol = 3)



grid.arrange(b7,b8, ncol = 2)

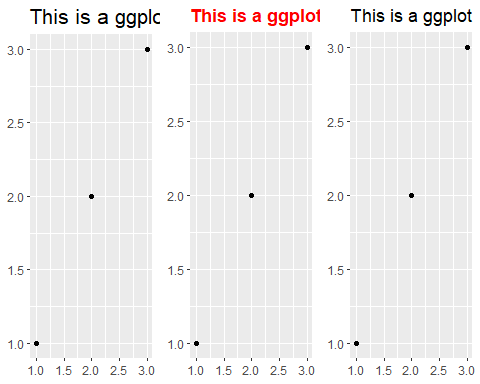


## 8.3 Modifying Theme Components

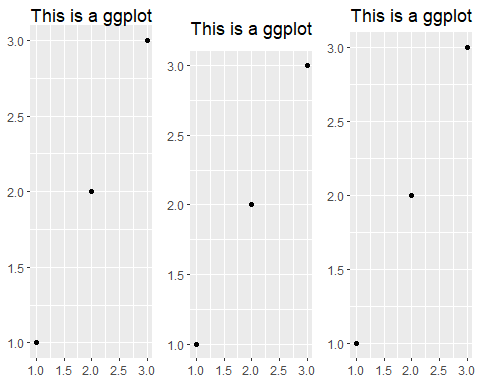
There are 4 basic types of built-in element functions : text, lines, rectangles, blank

*- element\_text()* draws labels and headings.

base\_t <- base + labs(title = "This is a ggplot") + xlab(NULL) + ylab(NULL)  
b1 = base\_t + theme(plot.title = element\_text(size = 16))  
b2 = base\_t + theme(plot.title = element\_text(face = "bold", colour = "red"))  
b3 = base\_t + theme(plot.title = element\_text(hjust = 1))  
grid.arrange(b1,b2,b3,ncol = 3)

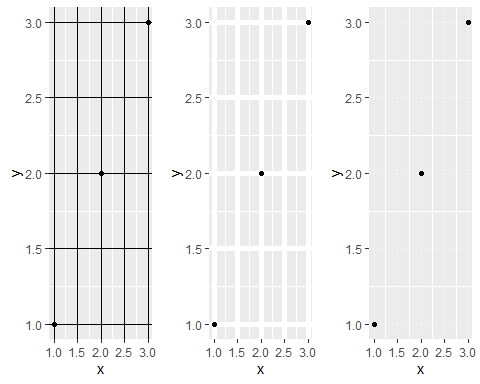


# The margins here look asymmetric because there are also plot margins  
b1 = base\_t + theme(plot.title = element\_text(margin = margin()))  
b2 = base\_t + theme(plot.title = element\_text(margin = margin(t = 10, b = 10)))  
b3 = base\_t + theme(axis.title.y = element\_text(margin = margin(r = 10)))  
  
grid.arrange(b1,b2,b3,ncol = 3)



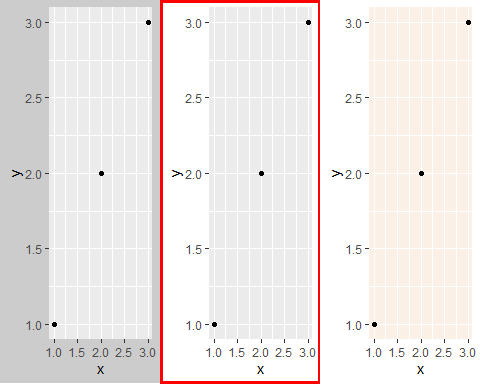
*- element\_line()* draws lines parameterised by colour, size and linetype

b1 = base + theme(panel.grid.major = element\_line(colour = "black"))  
b2 = base + theme(panel.grid.major = element\_line(size = 2))  
b3 = base + theme(panel.grid.major = element\_line(linetype = "dotted"))  
  
grid.arrange(b1,b2,b3,ncol = 3)



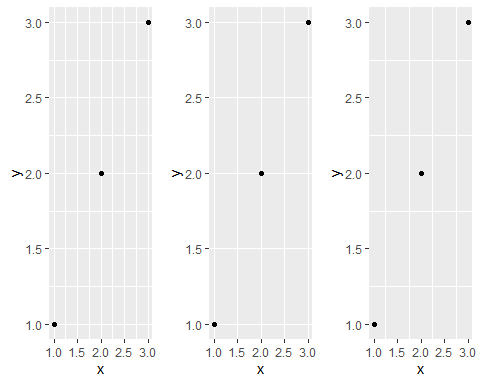
*- element\_rect()* draws rectangles, mostly used for backgrounds, parameterised by *fill* colour and border *colour*, *size* and *linetype*.

b1 = base + theme(plot.background = element\_rect(fill = "grey80", colour = NA))  
b2 = base + theme(plot.background = element\_rect(colour = "red", size = 2))  
b3 = base + theme(panel.background = element\_rect(fill = "linen"))  
  
grid.arrange(b1,b2,b3,ncol = 3)

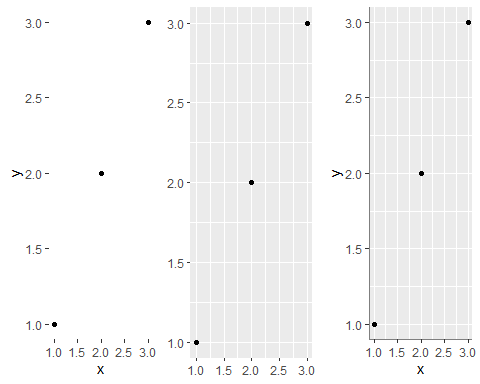


*- element\_blank()* draws nothing.

b1 = base  
b2 = base + theme(panel.grid.minor = element\_blank())  
b3 = base + theme(panel.grid.major = element\_blank())  
  
grid.arrange(b1,b2,b3,ncol = 3)



b1 = base + theme(panel.background = element\_blank())  
b2 = base + theme(  
 axis.title.x = element\_blank(),  
 axis.title.y = element\_blank()  
)  
b3 = base + theme(axis.line = element\_line(colour = "grey50"))  
  
grid.arrange(b1,b2,b3,ncol = 3)



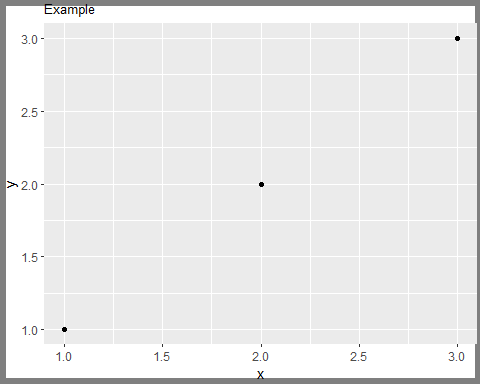
## 8.4 Theme Elements

There are so many elements. Roughly grouped into five category : plot, axis, legend, panel, facet

### 8.4.1 Plot Elements

|  |  |  |
| --- | --- | --- |
| Element | Setter | Description |
| plot.background | element\_rect() | Plot backgroud |
| plot.title | element\_text() | Plot title |
| plot.margin | margin() | Margins around plot |
|  |  |  |

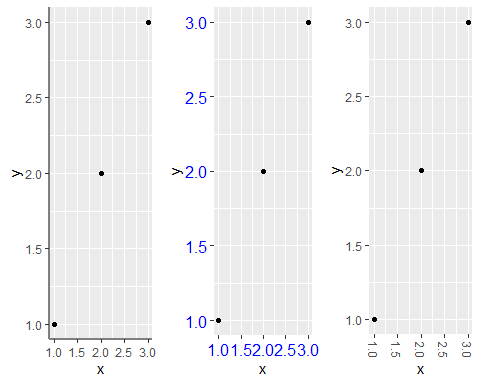
base + labs(title = "Example") +theme(  
 plot.background = element\_rect(colour = "grey50", size = 4),  
 plot.title = element\_text(size = 10),  
 plot.margin = margin(2, 2, 2, 2)  
)



### 8.4.2 Axis Elements

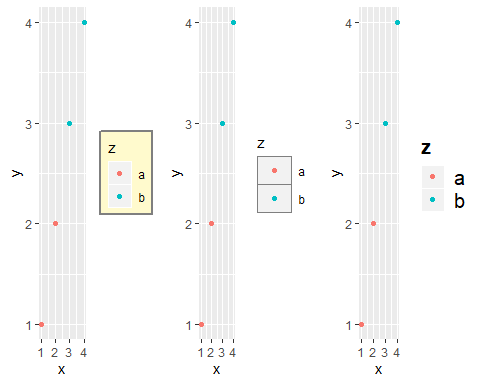
axis.text (and axis.title) comes in three forms: axis.text, axis.text.x, and axis.text.y.

df <- data.frame(x = 1:3, y = 1:3)  
base <- ggplot(df, aes(x, y)) + geom\_point()  
# Accentuate the axes  
b1 = base + theme(axis.line = element\_line(colour = "grey50", size = 1))  
# Style both x and y axis labels  
b2 = base + theme(axis.text = element\_text(color = "blue", size = 12))  
# Useful for long labels  
b3 = base + theme(axis.text.x = element\_text(angle = -90, vjust = 0.5))  
  
grid.arrange(b1,b2,b3,ncol=3)



### 8.4.3 Legend Elements

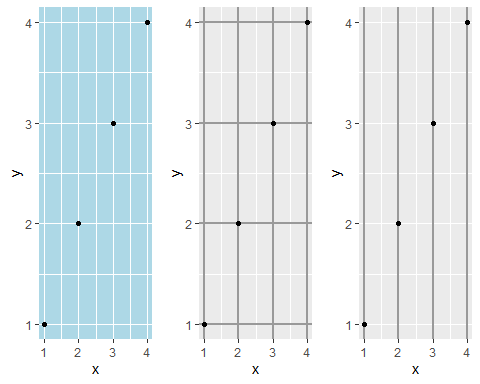
df <- data.frame(x = 1:4, y = 1:4, z = rep(c("a", "b"), each = 2))  
  
base <- ggplot(df, aes(x, y, colour = z)) + geom\_point()  
  
b1 = base + theme(  
 legend.background = element\_rect(  
 fill = "lemonchiffon",  
 colour = "grey50",  
 size = 1  
)  
)  
b2 = base + theme(  
 legend.key = element\_rect(color = "grey50"),  
 legend.key.width = unit(0.9, "cm"),  
 legend.key.height = unit(0.75, "cm")  
)  
b3 = base + theme(  
 legend.text = element\_text(size = 15),  
 legend.title = element\_text(size = 15, face = "bold")  
)  
grid.arrange(b1,b2,b3,ncol=3)



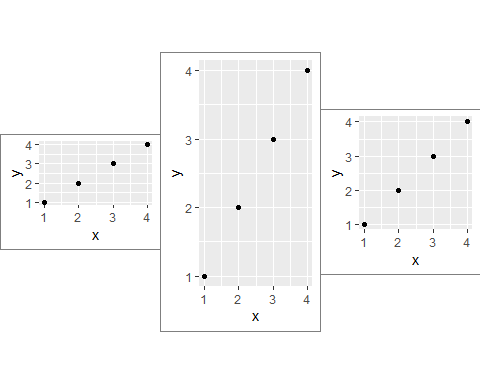
### 8.4.4 Panel Elements

The main difference between *panel.background* and *panel.border* is that the background is drawn underneath the data, and the border is drawn on top of it. For that reason, you’ll always need to assign *fill = NA* when overriding *panel.border.*

base <- ggplot(df, aes(x, y)) + geom\_point()  
# Modify background  
b1 = base + theme(panel.background = element\_rect(fill = "lightblue"))  
# Tweak major grid lines  
b2 = base + theme(panel.grid.major = element\_line(color = "gray60", size = 0.8))  
# Just in one direction  
b3 = base + theme(panel.grid.major.x = element\_line(color = "gray60", size = 0.8))  
  
grid.arrange(b1,b2,b3,ncol=3)



base2 <- base + theme(plot.background = element\_rect(colour = "grey50"))  
# Wide screen  
b1 = base2 + theme(aspect.ratio = 9 / 16)  
# Long and skiny  
b2 = base2 + theme(aspect.ratio = 2 / 1)  
# Square  
b3 = base2 + theme(aspect.ratio = 1)  
  
grid.arrange(b1,b2,b3,ncol=3)



### 8.4.5 Facetting Elements

Element *strip.text.x* affects both *facet\_wrap()* or *facet grid();* *strip.text.y* only affects *facet\_grid().*

df <- data.frame(x = 1:4, y = 1:4, z = c("a", "a", "b", "b"))  
base\_f <- ggplot(df, aes(x, y)) + geom\_point() + facet\_wrap(~z)  
b1 = base\_f  
b2 = base\_f + theme(panel.margin = unit(0.5, "in"))

## Warning: `panel.margin` is deprecated. Please use `panel.spacing` property  
## instead

b3 = base\_f + theme(  
 strip.background = element\_rect(fill = "grey20", color = "grey80", size = 1),  
 strip.text = element\_text(colour = "white")  
)  
  
grid.arrange(b1,b2,b3,ncol=3)

