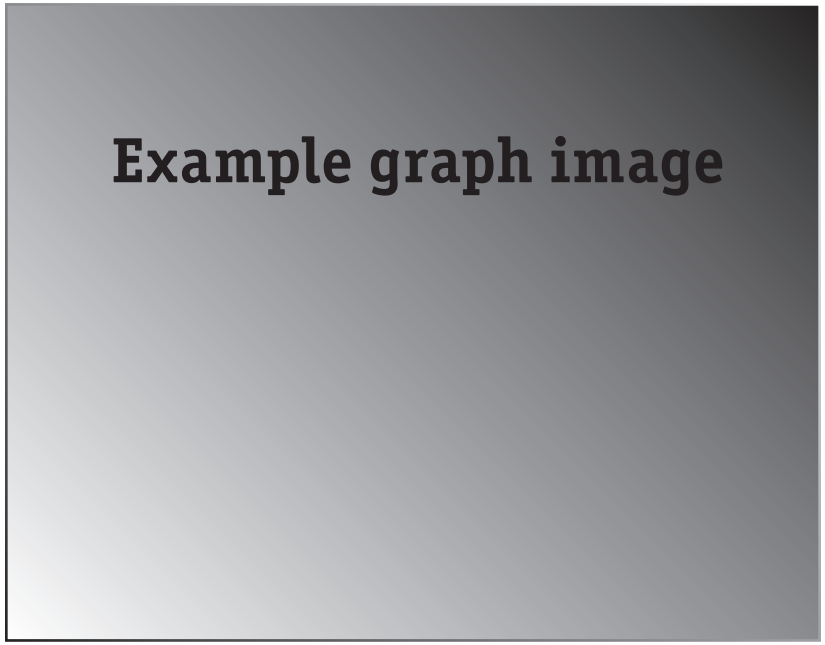
POLISHING YOUR ggplot2 PLOT

a plotting system for R, by Hadley Wickham

Introduction, Contact Info Here

IN THIS GUIDE:



short caption

AXIS PROPERTIES

titles
limits
ticks

Titling plot axes

Labelling the x and y axes is accomplished with `xlab` and `ylab`, or `labs`.

```
p + xlab("City MPG")        sets the x-axis label to City MPG
p + ylab("Highway MPG")     sets the y-axis label to Highway MPG
```

Using the `labs` command sets both the x- and y-axis labels simultaneously:

```
p + labs(x = "City MPG", y = "Highway MPG")
```

Changing axis limits

Setting the x and y axis limits is accomplished with `xlim` and `ylim`.

```
xlim(10,20)                 set continuous x-scale's limit to 10–20
ylim(20,10)                 set continuous y-scale's limit to 10–20
xlim("a", "b", "c")         set discrete x-scale limits to a, b, c
```

Changing axis tick marks

The positions of axis tick marks are set by the `breaks` command.
The labels that appear at tick marks are set by the `labels` command.
If `labels` is set, `breaks` must also be set so the two can be matched correctly.

```
breaks = c(4, 5, 6)                 sets tick marks at even intervals from 4–8
labels = c("four", "five", "six")   sets tick mark labels to four, five, and six
```

LEGEND PROPERTIES

titles
keys

Changing the legend title

The legend title can be set with `labs`:

```
p + labs(colour = "Displacement")   sets the legend title to "Displacement"
```

Changing the legend key values

The values in the legend's key are controlled by the same arguments that control the values and position of axis tick-marks, `breaks` and `labels`.

`breaks` controls the values that appear in the key.
`labels` controls the labels that appear at each value specified by `breaks`.

outline
fill
colours

COLOUR SCALES

Defining the data outline colour scale

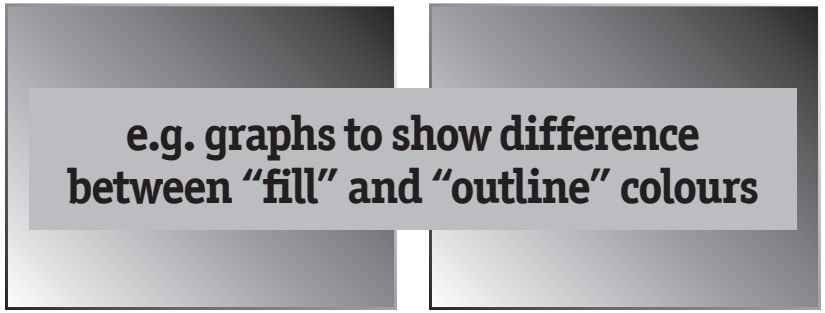
The outline colour scale is defined using the `scale_colour_` command (followed by the type of scale desired).

`scale_colour_gradient(low, high)` set outline scale to a two-colour gradient
`scale_colour_brewer(pal = "set")` set outline scale a brewer palette "set"

Defining the data fill colour scale

The fill colour scale is defined using the `scale_fill_` command (followed by the type of scale desired).

`scale_fill_gradient(low,high)` set fill scale to a two-colour gradient
`scale_fill_brewer(pal = "set")` set fill scale a brewer palette "set"



short caption

Defining colour arguments

The colour arguments of `_gradient` scales are set as named colours (such as "red" and "black"). Arguments to the `_brewer` scale are passed as the names of existing, discrete brewer palettes (a list of which can be found at ???).

`scale_fill_gradient(low = "white", high = "pink")`
creates a (continuous) gradient fill scale from white to pink
`scale_fill_brewer(pal = "Pastel1")`
creates a (discrete) fill scale using the Brewer palette "Pastel1"

Common scale commands and arguments

all commands preceded by `scale_fill` or `scale_colour`

`_gradient(low = "a", high = "b")`
scale is a continuous gradient from "a" to "b"
`_gradient2(low = "a", mid = "b", high = "c")`
scale is a continuous gradient from "a" to "c", with mid colour "b"
`_gradientn(arg)`
scale is a continuous gradient of the colours provided in vector arg
`_brewer(pal = "arg")`
scale is the discrete set of colours from the brewer palette "arg"

THEME ELEMENTS

text
line
rect
blank

THEME ELEMENT		TYPE	DESCRIPTION
axis	.line	segment	line along axis
axis	.text.x	text	x-axis label
axis	.text.y	text	y-axis label
axis	.ticks	segment	axis tick marks
axis	.title.x	text	horizontal tick labels
axis	.title.y	text	vertical tick labels
legend	.background	rect	background of legend
legend	.key	rect	background underneath legend keys
legend	.text	text	legend labels
legend	.title	text	legend name
panel	.background	rect	background of panel
panel	.border	rect	border around panel
panel	.grid.major	line	major grid lines
panel	.grid.minor	line	minor grid lines
panel	.background	rect	background of the entire plot
panel	.title	text	plot title
strip	.background	rect	background of facet labels
strip	.text.x	text	text for horizontal strips
strip	.text.y	text	text for vertical strips

Defining _text elements

`theme_text()` draws labels and headings. The arguments passed control the appearance of text. `family`, `face`, `colour`, `size`, `hjust`, `vjust`, `angle`, and `lineheight` are the controllable attributes of a text element.

`plot.title = theme_text(size = 20, face = "bold")`
sets the plot title in size 20 bold font

Defining _rect elements

`theme_rect()` draws rectangles (mostly for backgrounds). `fill` colour, outline `colour`, `size`, and `linetype` are the controllable attributes of a rect element.

`plot.background = theme_rect(fill = "grey80", colour = NA)`
sets the plot background with a grey80 fill and no outline

Defining _line elements

`theme_line()` elements and `theme_segment()` elements draw lines and segments. (The arguments for each type are the same, but it is important to utilize the proper type, according to the above table). `colour`, `size`, and `linetype` are the controllable attributes of line and segment elements.

`panel.grid.major = theme_line(colour = "red", size = "2")`
sets the major grid to size 2 red lines

Defining _blank elements

`theme_blank()` elements draw nothing. A `theme_blank()` call prevents the item from being shown in the final plot. There are no arguments passed when invoking a `theme_blank()` element.