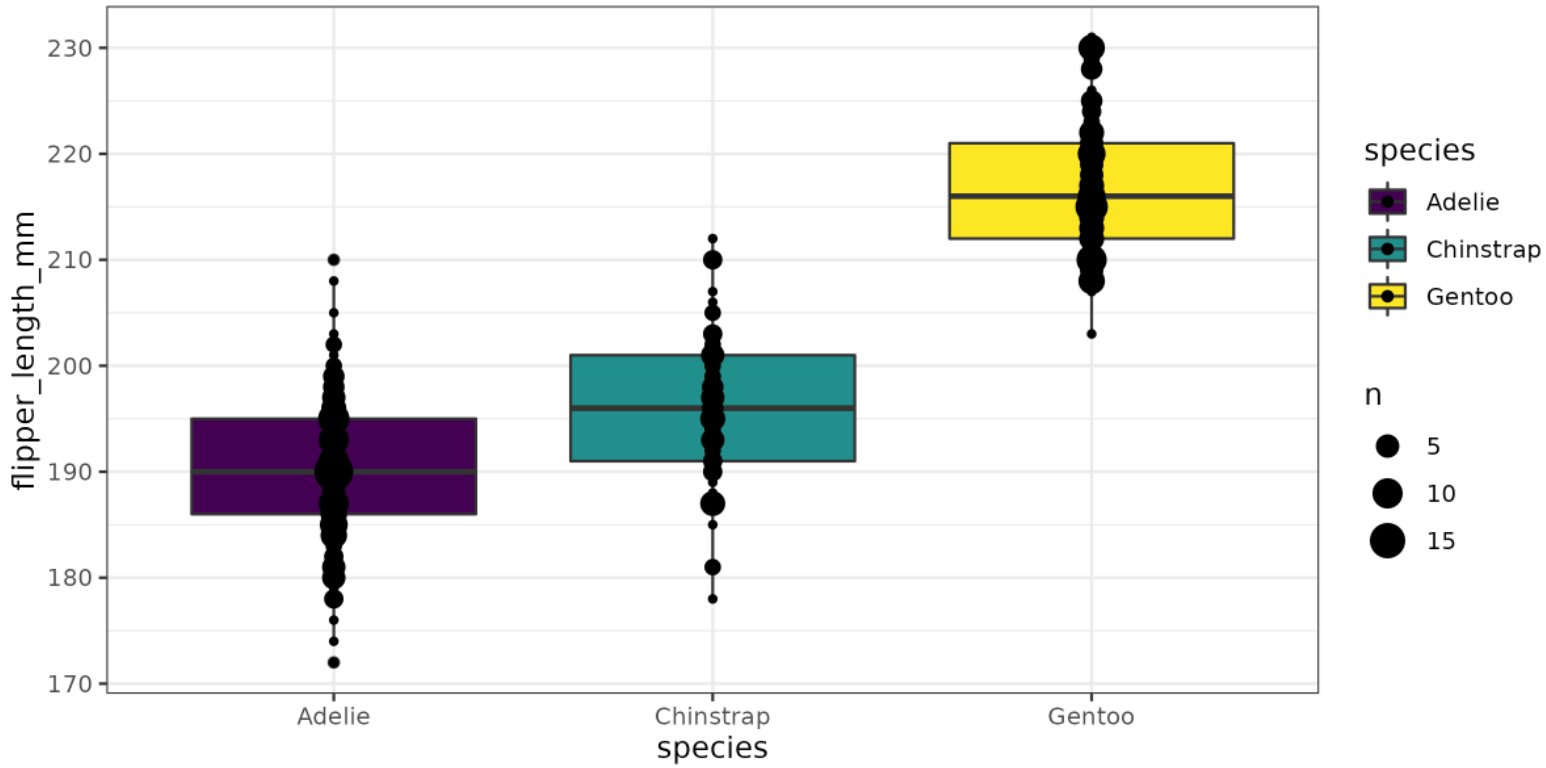
Workshop: Dealing with Data in R

Visualizing Data in R

A primer on `ggplot2`



Steffi LaZerte https://steffilazerte.ca | *Compiled: 2022-01-26*

First things first

Save previous script (consider names like **`intro.R`** or **`1_getting_started.R`**)

Open New File

(make sure you're in the RStudio Project)

Add **`library(tidyverse)`** to the top

Save this new script

consider names like **`figs.R`** or **`2_figures.R`**

Outline

1. Figures with `ggplot2` (A `tidyverse` package)

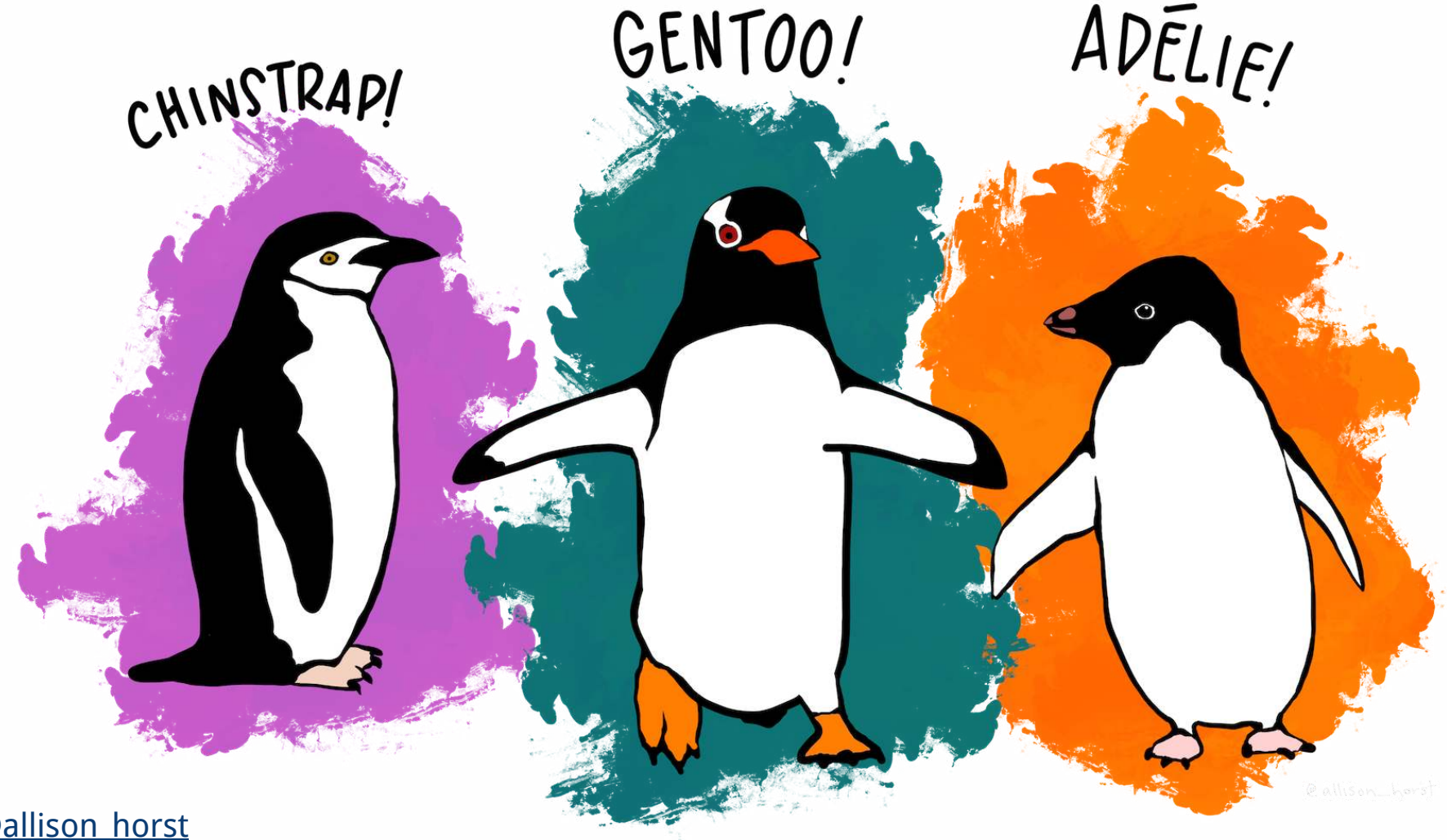
- Basic plot
- Common plot types
- Plotting by categories
- Adding statistics
- Customizing plots
- Annotating plots

2. Combining figures with `patchwork`

3. Saving figures

Artwork by @allison_horst

Our data set: Palmer Penguins!

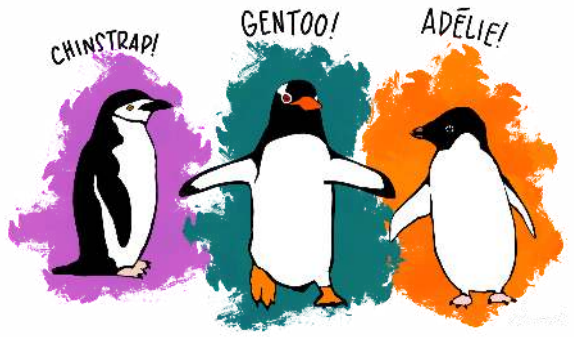


Artwork by @allison_horst

Our data set: Palmer Penguins!


```
library(palmerpenguins)
penguins
```

```
## # A tibble: 344 × 8
##   species island    bill_length_mm bill_depth_mm flipper_length_mm body_mass_g sex     year
##   <fct>   <fct>              <dbl>         <dbl>             <int>       <int> <fct>  <int>
## 1 Adelie  Torgersen           39.1          18.7               181        3750 male    2007
## 2 Adelie  Torgersen           39.5          17.4               186        3800 female  2007
## 3 Adelie  Torgersen           40.3          18                 195        3250 female  2007
## 4 Adelie  Torgersen           NA            NA                  NA          NA <NA>    2007
## 5 Adelie  Torgersen           36.7          19.3               193        3450 female  2007
## 6 Adelie  Torgersen           39.3          20.6               190        3650 male    2007
## 7 Adelie  Torgersen           38.9          17.8               181        3625 female  2007
## 8 Adelie  Torgersen           39.2          19.6               195        4675 male    2007
## # … with 336 more rows
```



Artwork by @allison_horst

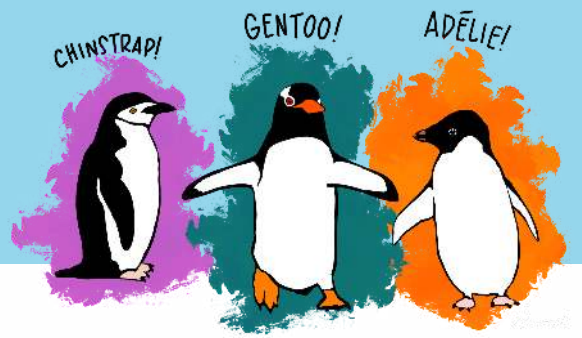
Our data set: Palmer Penguins!


```
library(palmerpenguins)
penguins
```

```
## # A tibble: 344 × 8
##   species island    bill_length_mm bill_depth_mm flipper_length_mm body_mass_g sex     year
##   <fct>   <fct>              <dbl>         <dbl>             <int>       <int> <fct>  <int>
## 1 Adelie  Torgersen           39.1          18.7               181        3750 male    2007
## 2 Adelie  Torgersen           39.5          17.4               186        3800 female  2007
## 3 Adelie  Torgersen           40.3          18                 195        3250 female  2007
## 4 Adelie  Torgersen           NA            NA                  NA          NA <NA>    2007
## 5 Adelie  Torgersen           36.7          19.3               193        3450 female  2007
## 6 Adelie  Torgersen           39.3          20.6               190        3650 male    2007
## 7 Adelie  Torgersen           38.9          17.8               181        3625 female  2007
## 8 Adelie  Torgersen           39.2          19.6               195        4675 male    2007
## # … with 336 more rows
```

Your turn!

1. Run this code and look at the output in the console
2. Run **`view(penguins)`** to see the output in the built-in spreadsheet viewer



Artwork by @allison_horst

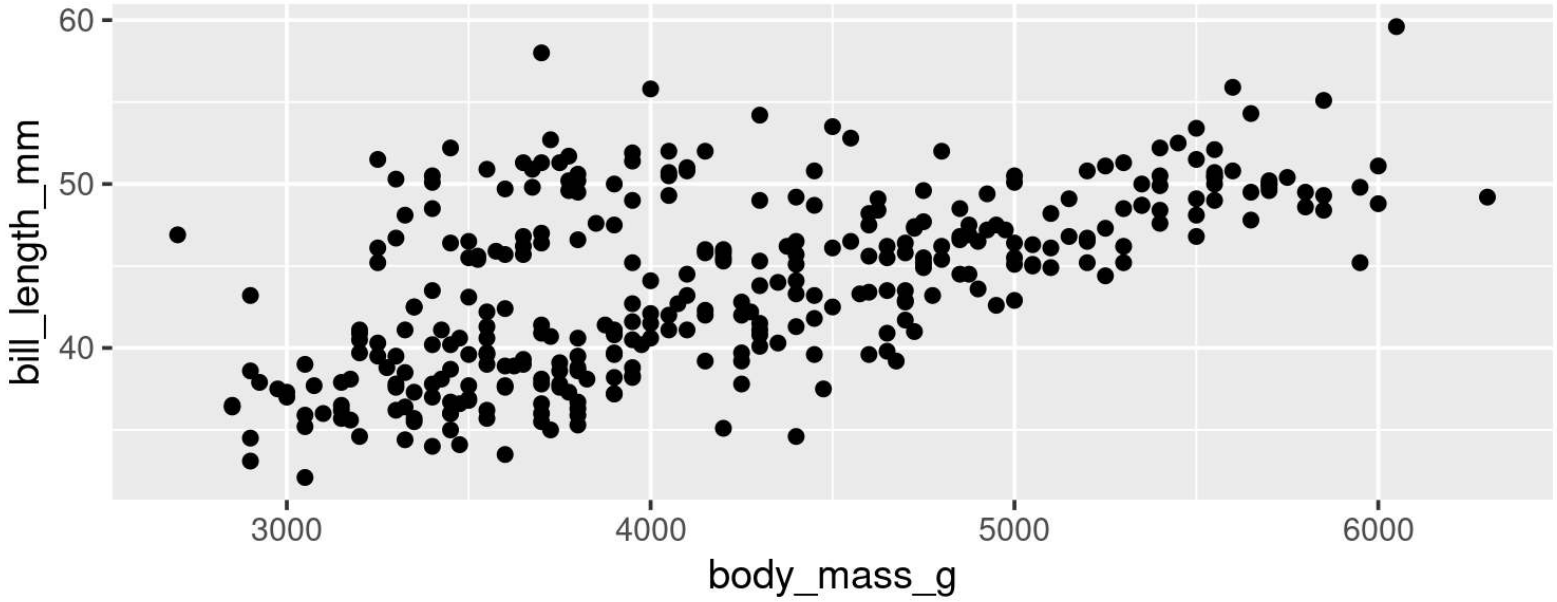
Side Note

Where did the `penguins` data set come from?

- Sometimes R packages contain data
- If you load a package (i.e. `library(palmerpenguins)`) you can use the data
- **Note** that here the data object is called `penguins` (not `palmerpenguins`)
- **Note** this is NOT how you'll load your own data

A basic plot

```
library(palmerpenguins)
library(tidyverse)

ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm)) +
  geom_point()
```



Break it down

```
library(palmerpenguins)
library(tidyverse)

ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm)) +
  geom_point()
```

`library(palmerpenguins)`

- Load the **`palmerguins`** package so we have access to **`penguins`** data

Break it down

```
library(palmerpenguins)
library(tidyverse)

ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm)) +
  geom_point()
```

`library(tidyverse)`

- Load the **`tidyverse`** package (which loads the **`ggplot2`** package and gives us access to the **`ggplot()`** function among others)

Break it down

```
library(palmerpenguins)
library(tidyverse)

ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm)) +
  geom_point()
```

`ggplot()`

- Set the attributes of your plot
- **`data`** = Dataset
- **`aes`** = Aesthetics (how the data are used)
- Think of this as your plot defaults

Break it down

```
library(palmerpenguins)
library(tidyverse)

ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm)) +
  geom_point()
```

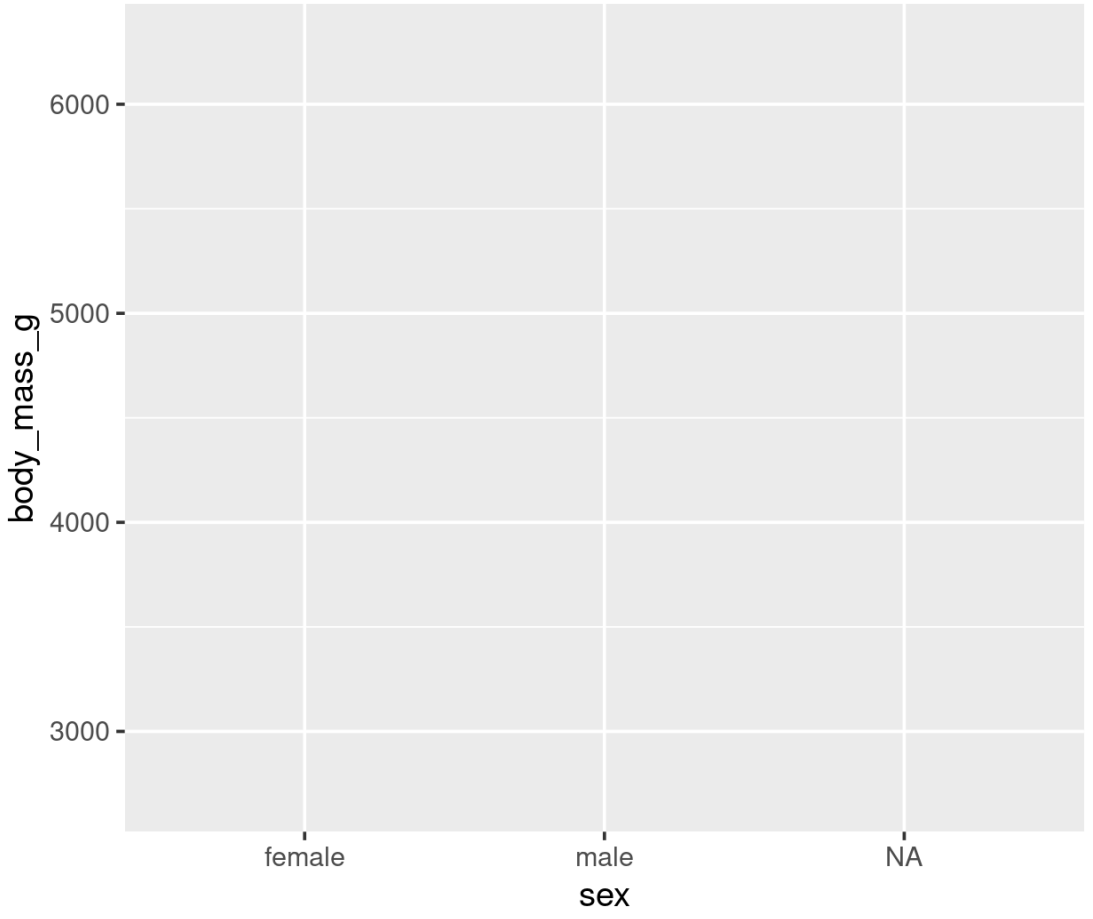
`geom_point()`

- Choose a **`geom`** function to display the data
- Always *added* to a **`ggplot()`** call with **`+`**

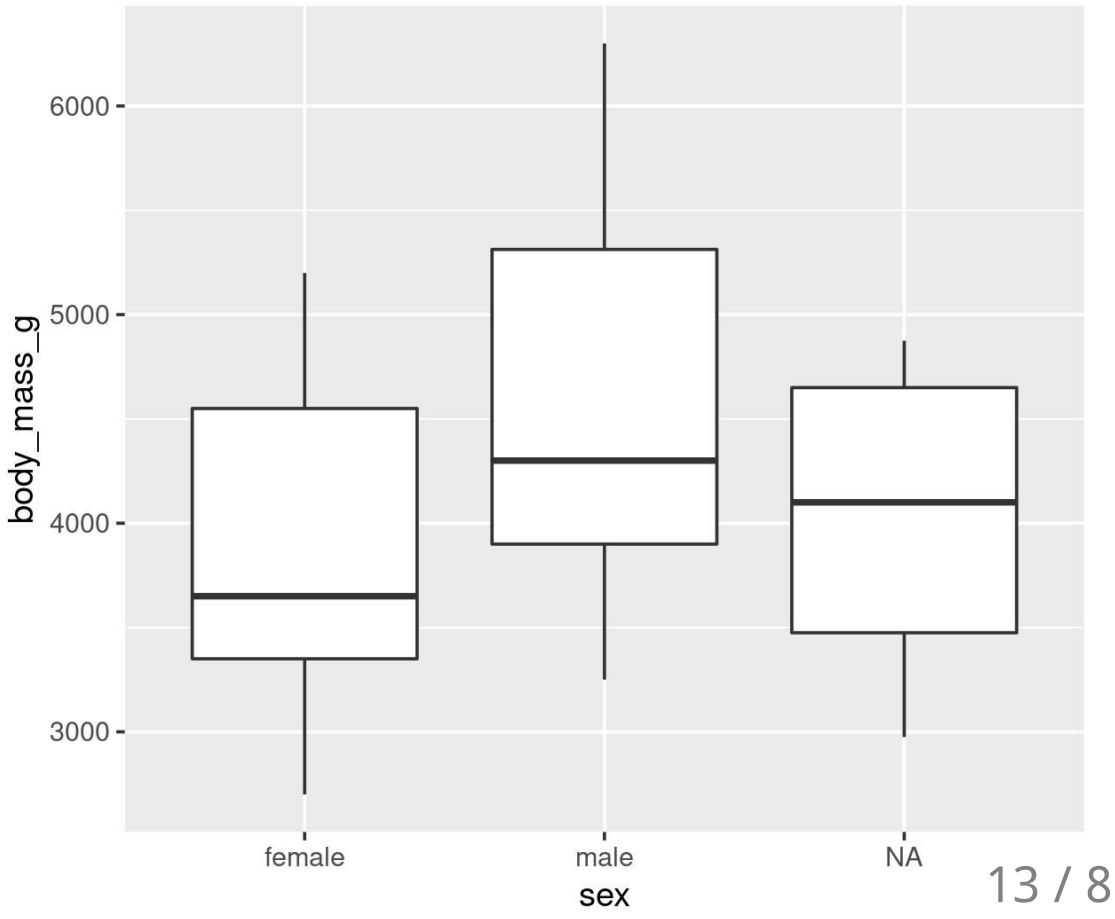
> ggplots are essentially layered objects, starting with a call to **`ggplot()`**

Plots are layered

```
ggplot(data = penguins, aes(x = sex, y = body_mass_g))
```

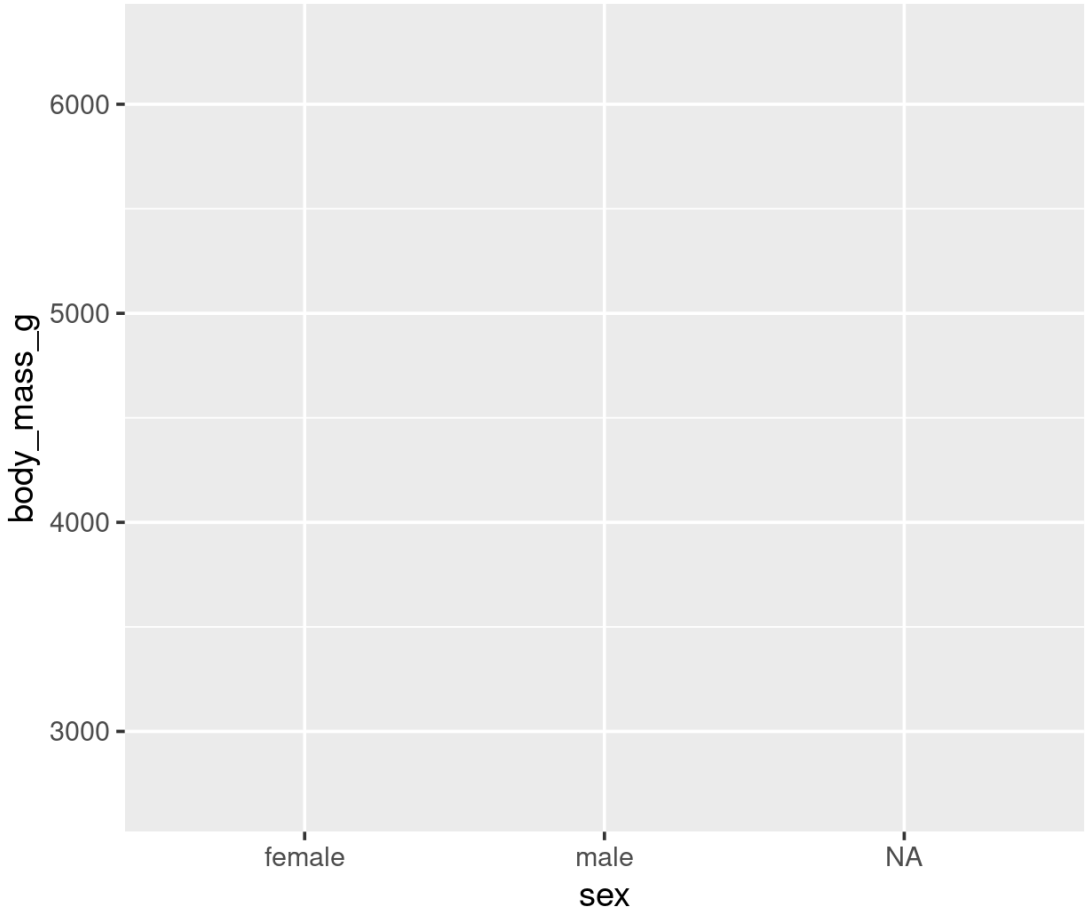



```
ggplot(data = penguins, aes(x = sex, y = body_mass_g)) +
  geom_boxplot()
```

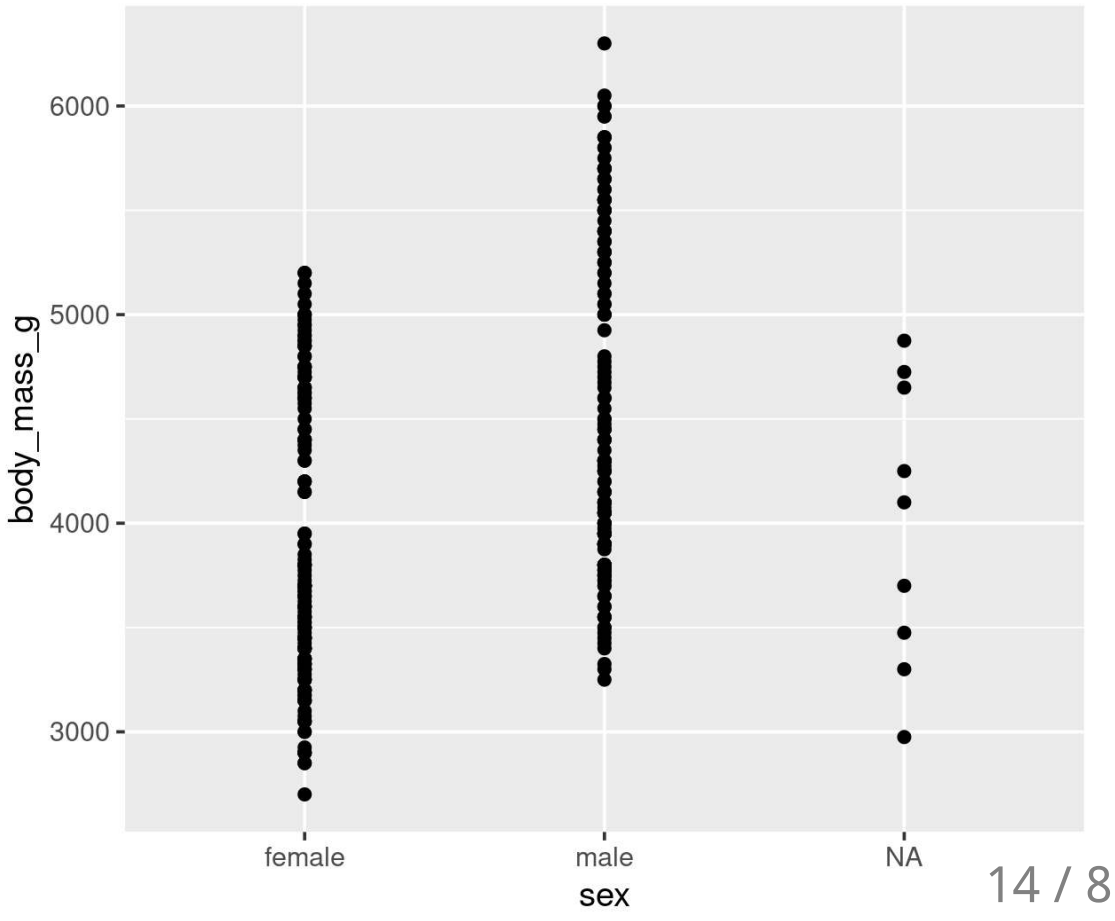


Plots are layered

```
ggplot(data = penguins, aes(x = sex, y = body_mass_g))
```

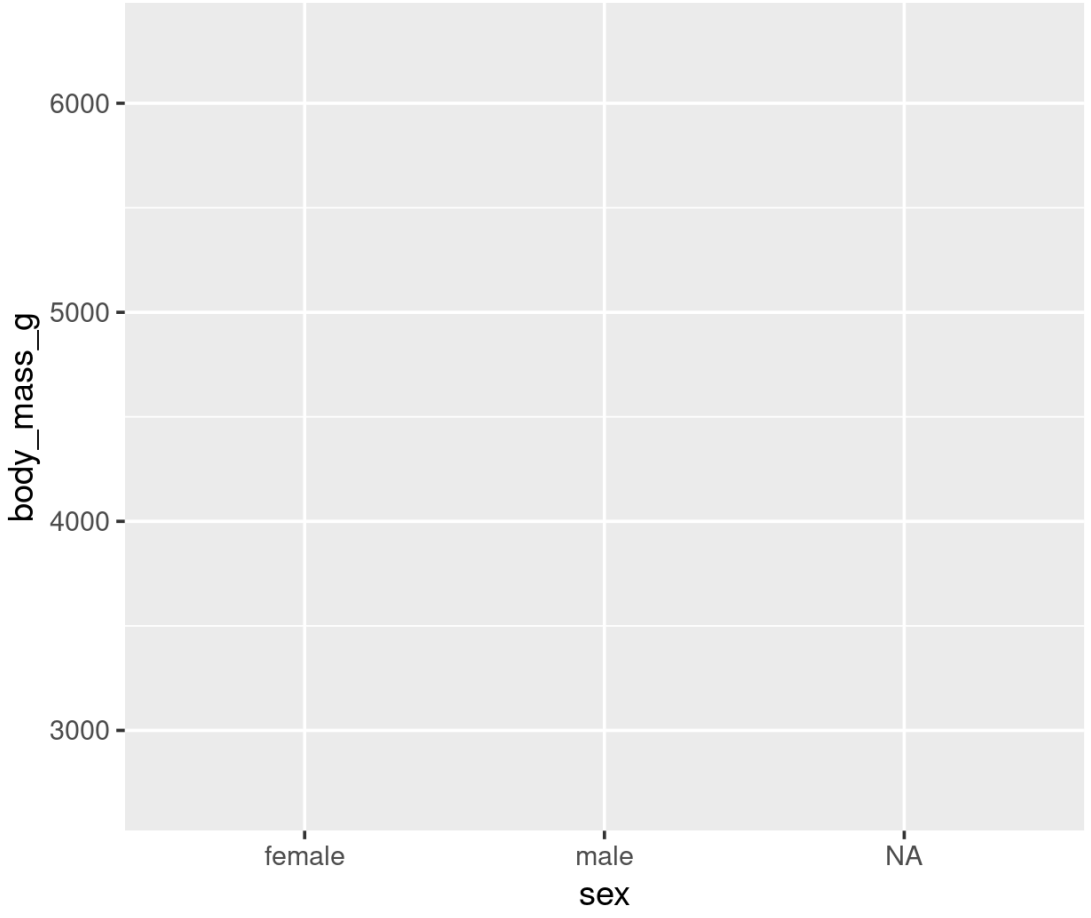



```
ggplot(data = penguins, aes(x = sex, y = body_mass_g)) +
  geom_point()
```

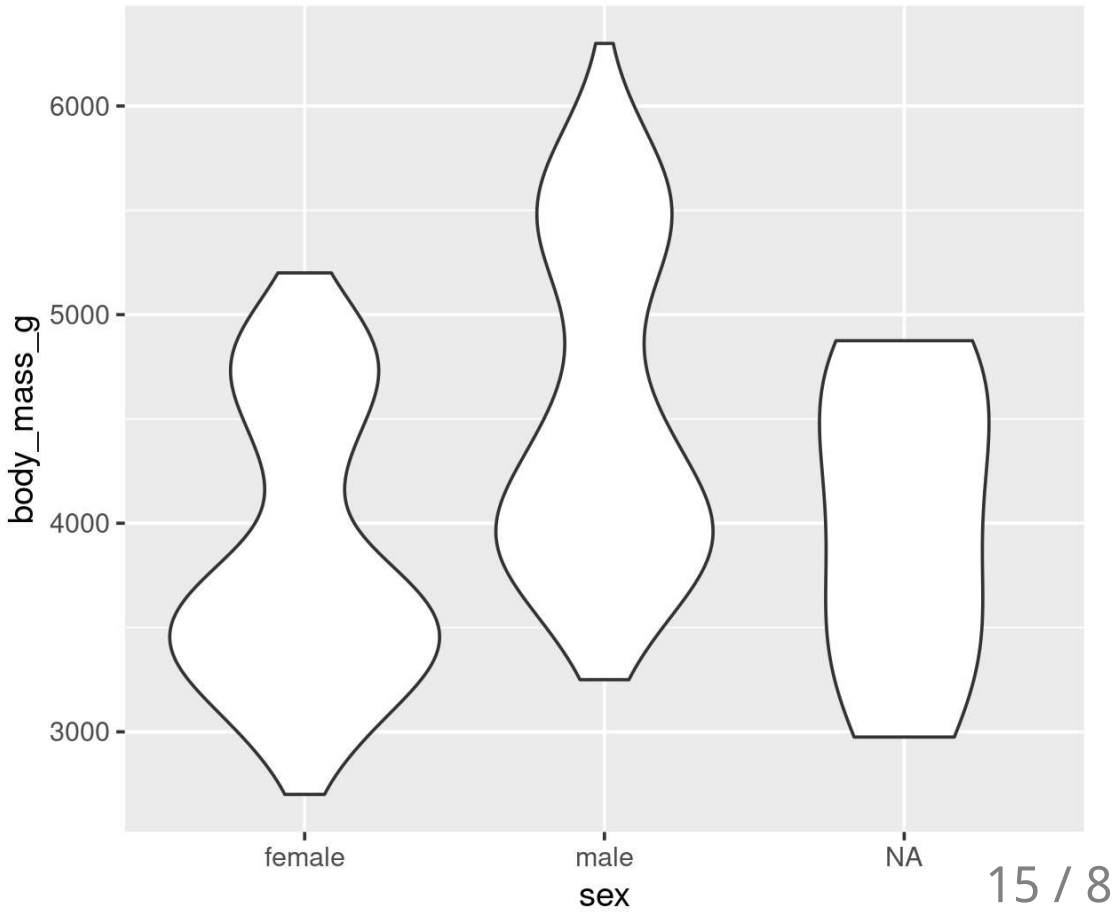


Plots are layered

```
ggplot(data = penguins, aes(x = sex, y = body_mass_g))
```



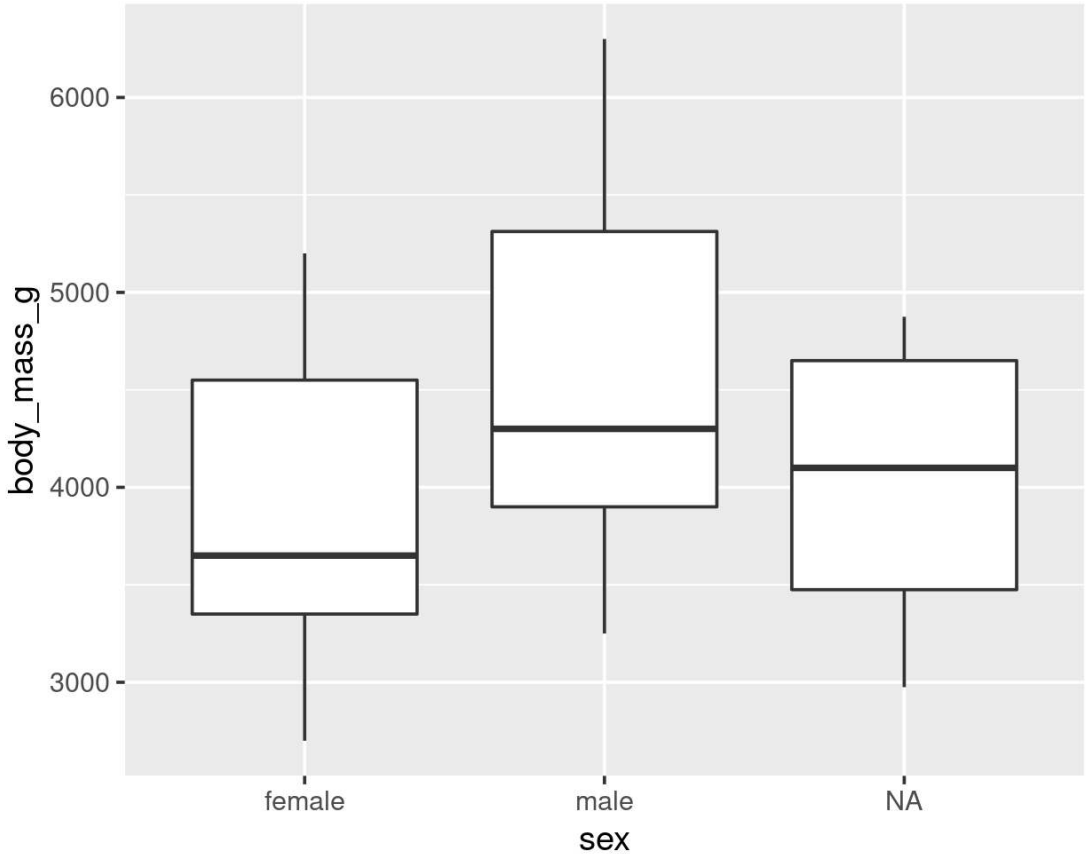

```
ggplot(data = penguins, aes(x = sex, y = body_mass_g)) +
  geom_violin()
```



Plots are layered

You can add multiple layers

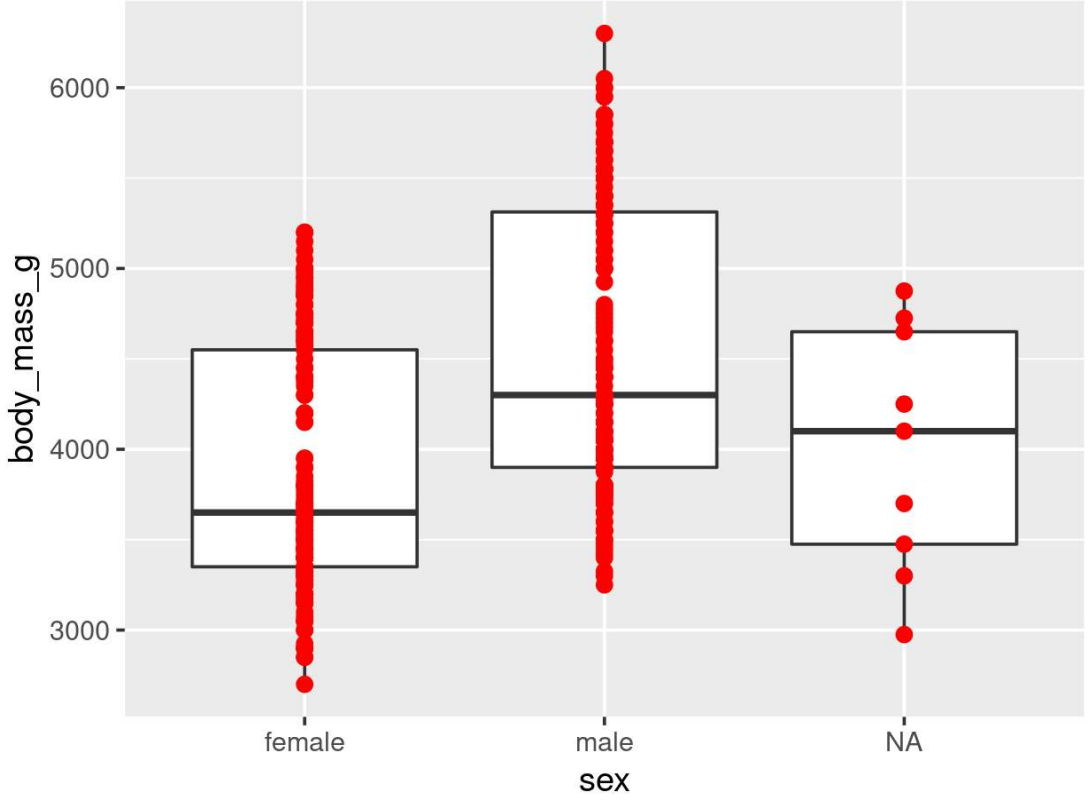
```
ggplot(data = penguins, aes(x = sex, y = body_mass_g)) +
  geom_boxplot()
```



Plots are layered

You can add multiple layers

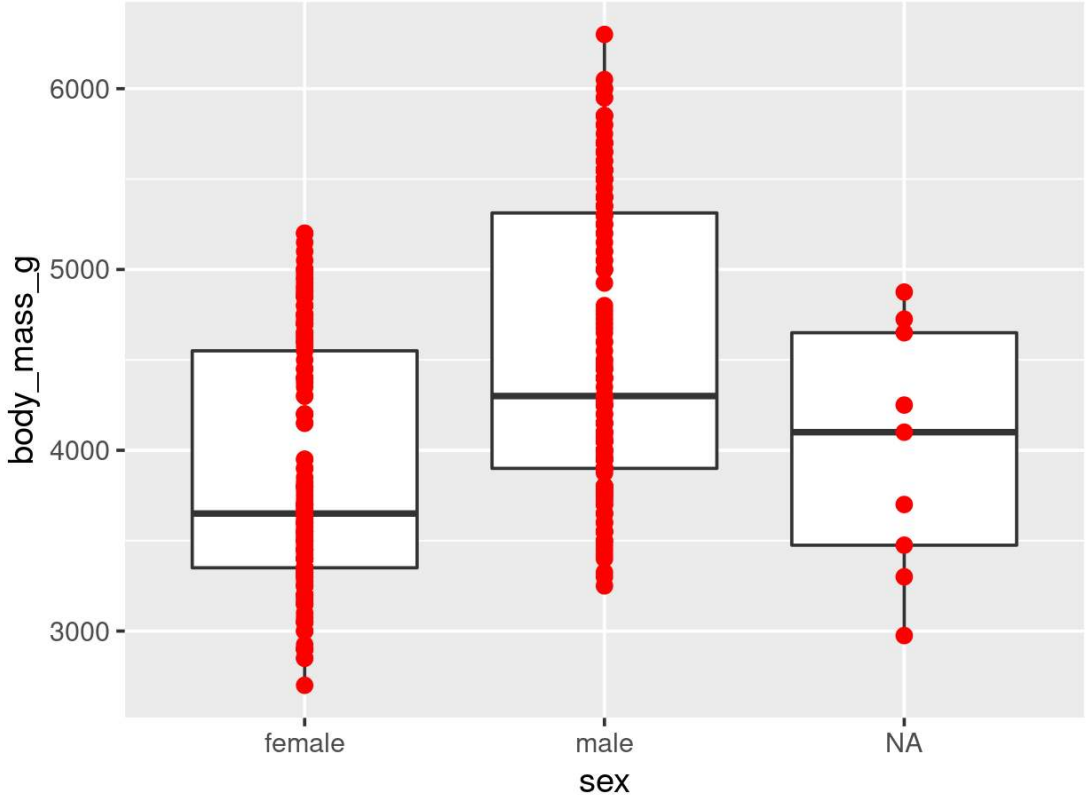
```
ggplot(data = penguins, aes(x = sex, y = body_mass_g)) +
  geom_boxplot() +
  geom_point(size = 2, colour = "red")
```



Plots are layered

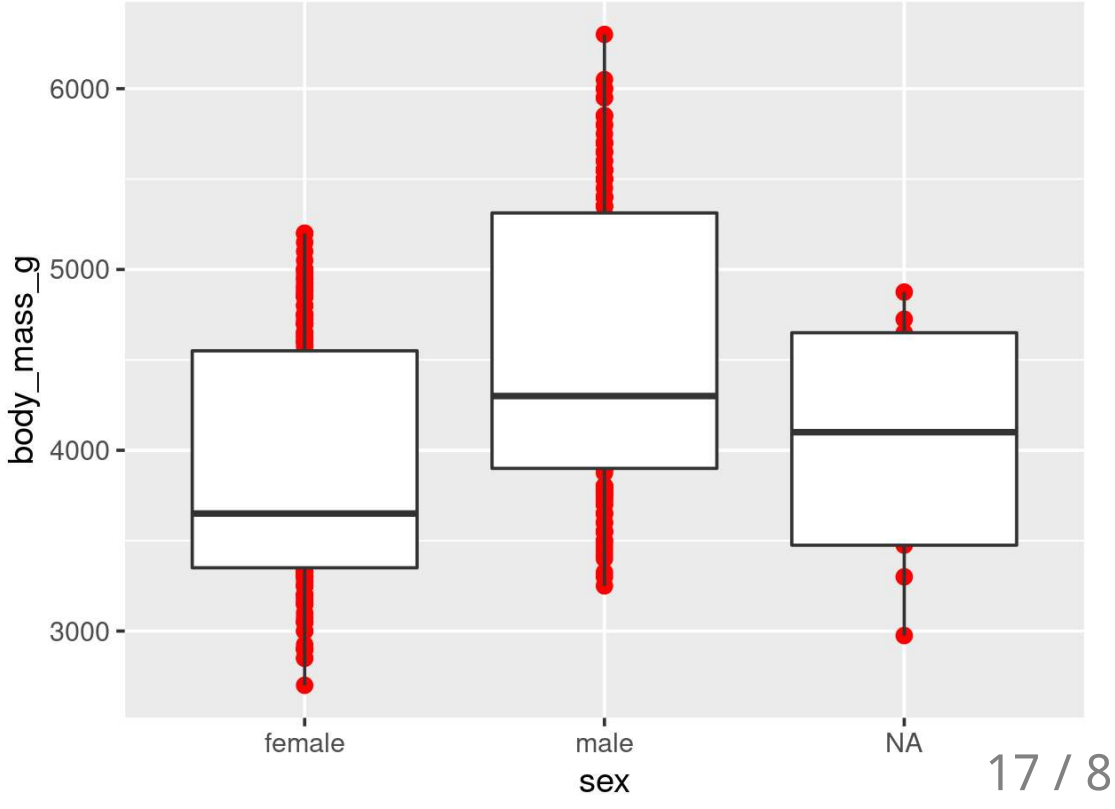
You can add multiple layers

```
ggplot(data = penguins, aes(x = sex, y = body_mass_g)) +
  geom_boxplot() +
  geom_point(size = 2, colour = "red")
```



Order matters

```
ggplot(data = penguins, aes(x = sex, y = body_mass_g)) +
  geom_point(size = 2, colour = "red") +
  geom_boxplot()
```



Plots are objects

Any ggplot can be saved as an object

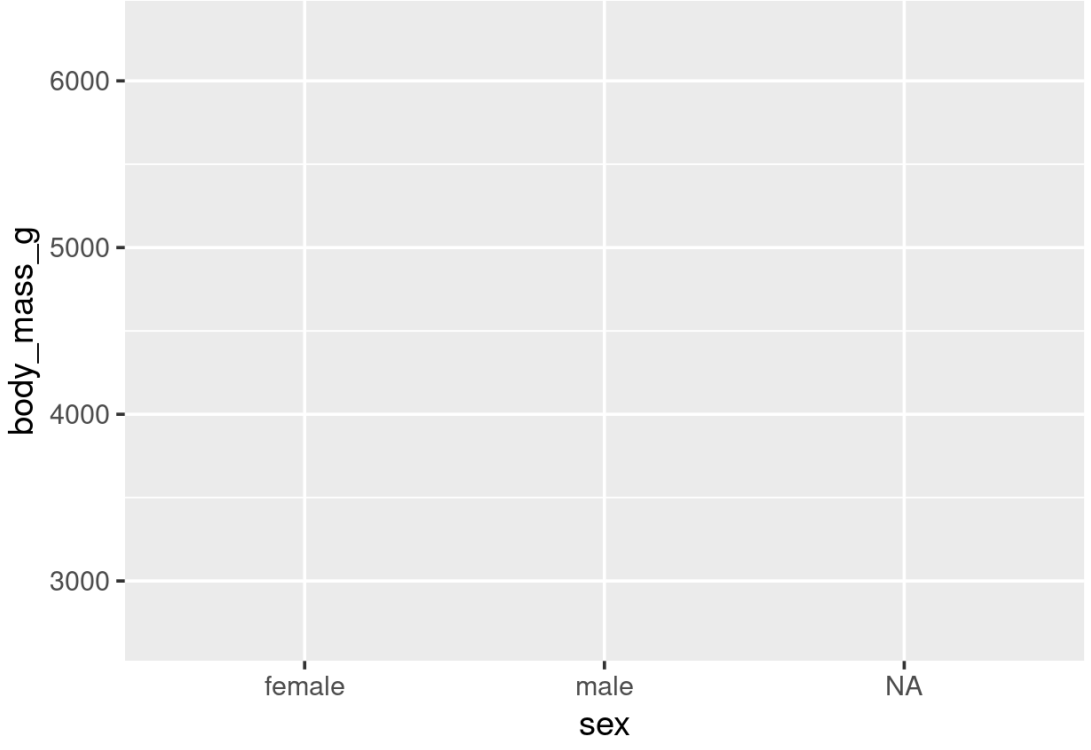
```
g <- ggplot(data = penguins, aes(x = sex, y = body_mass_g))
```

Plots are objects

Any ggplot can be saved as an object

```
g <- ggplot(data = penguins, aes(x = sex, y = body_mass_g))
```

```
g
```

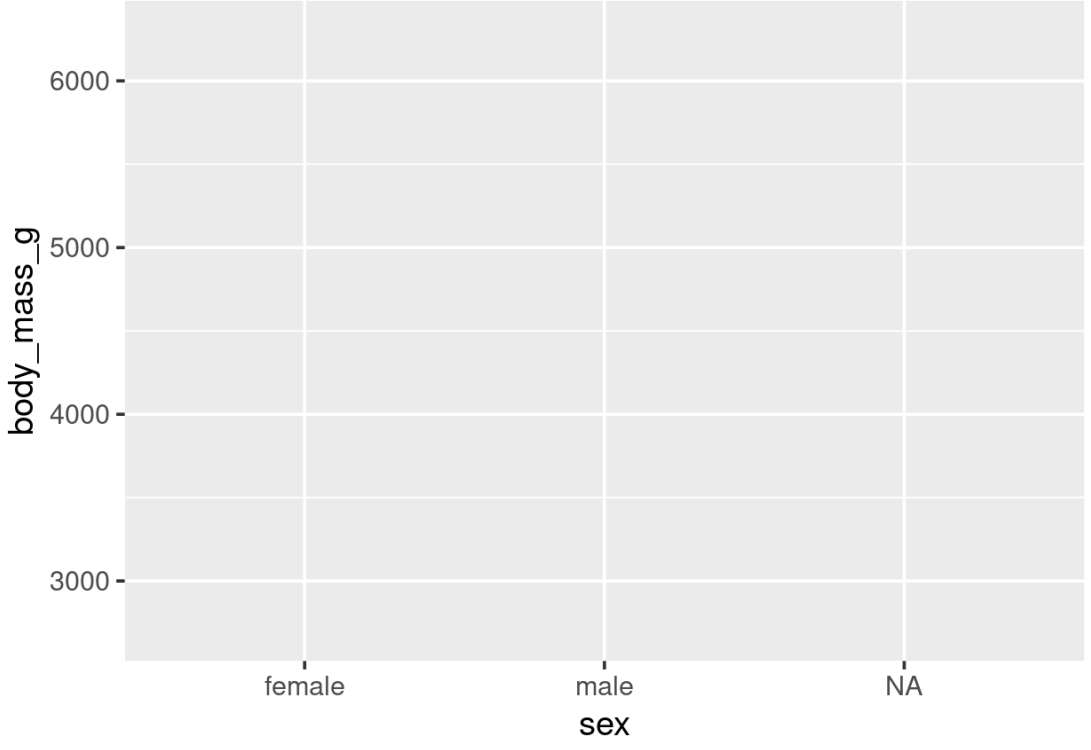


Plots are objects

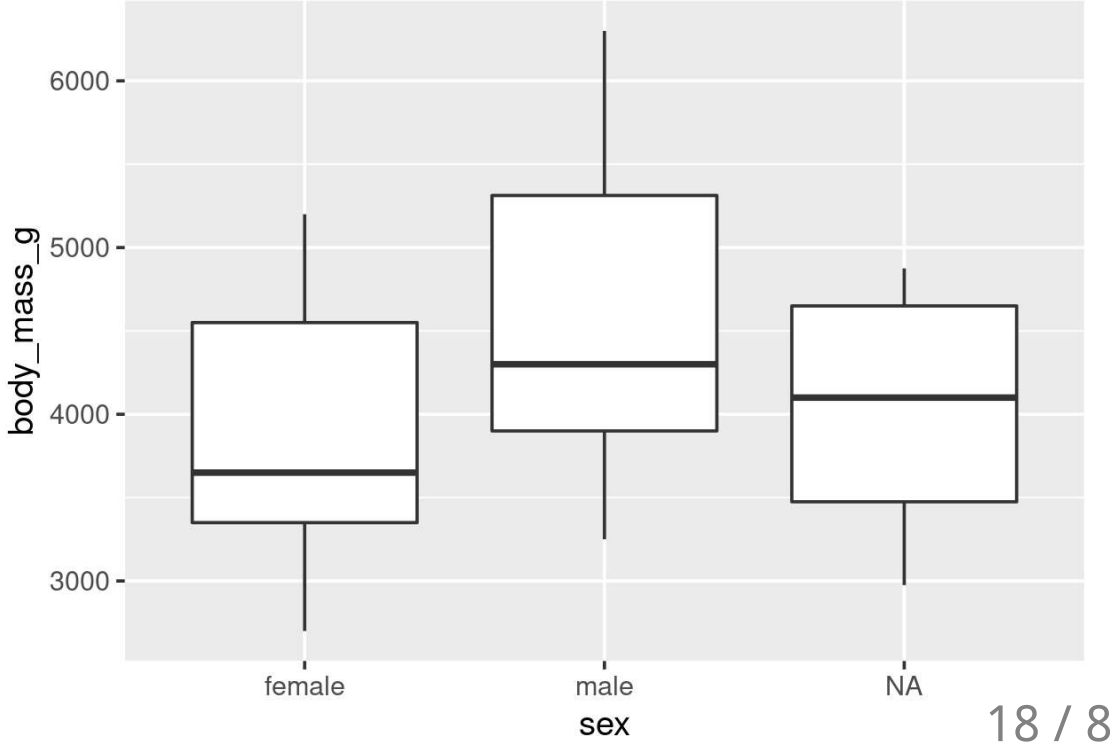
Any ggplot can be saved as an object

```
g <- ggplot(data = penguins, aes(x = sex, y = body_mass_g))
```

```
g
```




```
g + geom_boxplot()
```

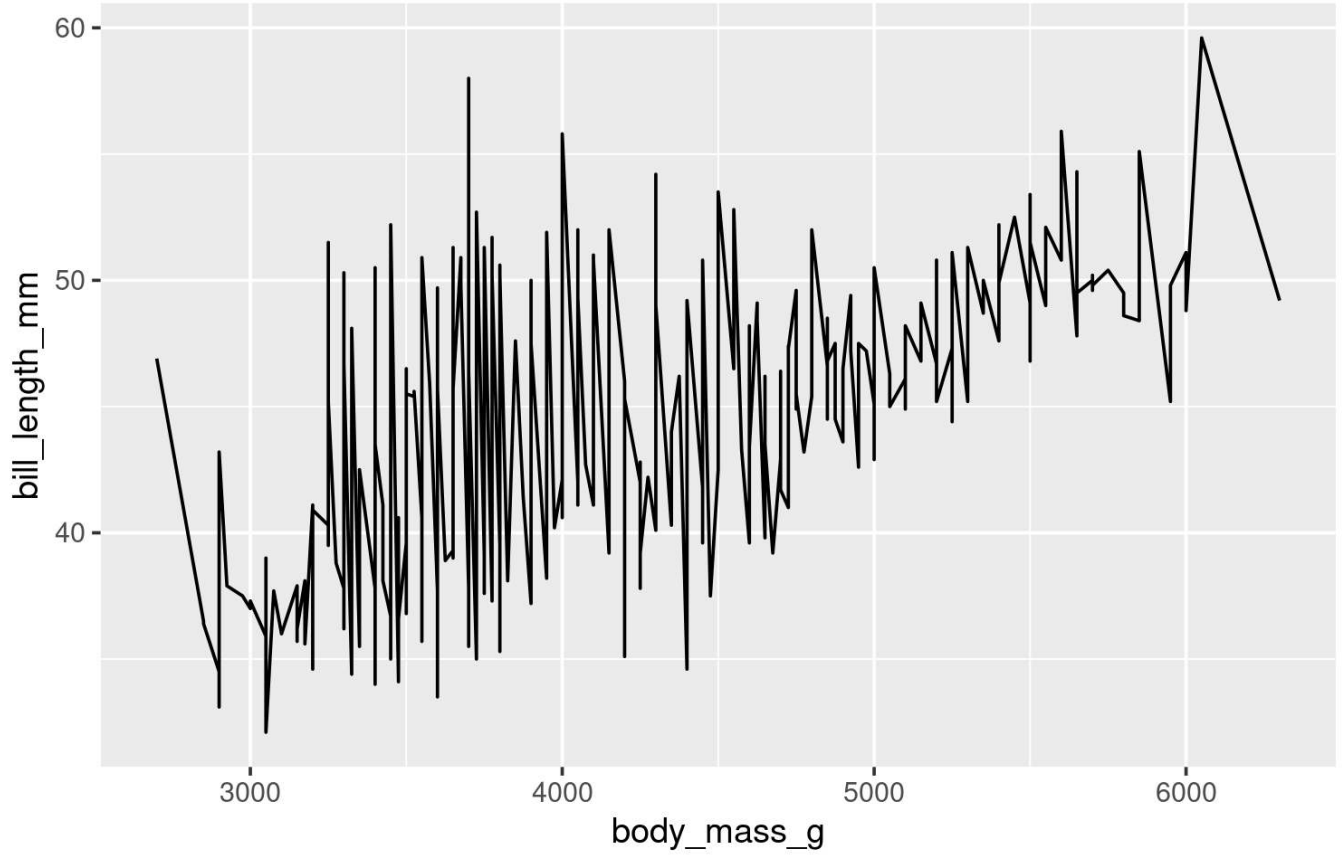


More Geoms

(Plot types)

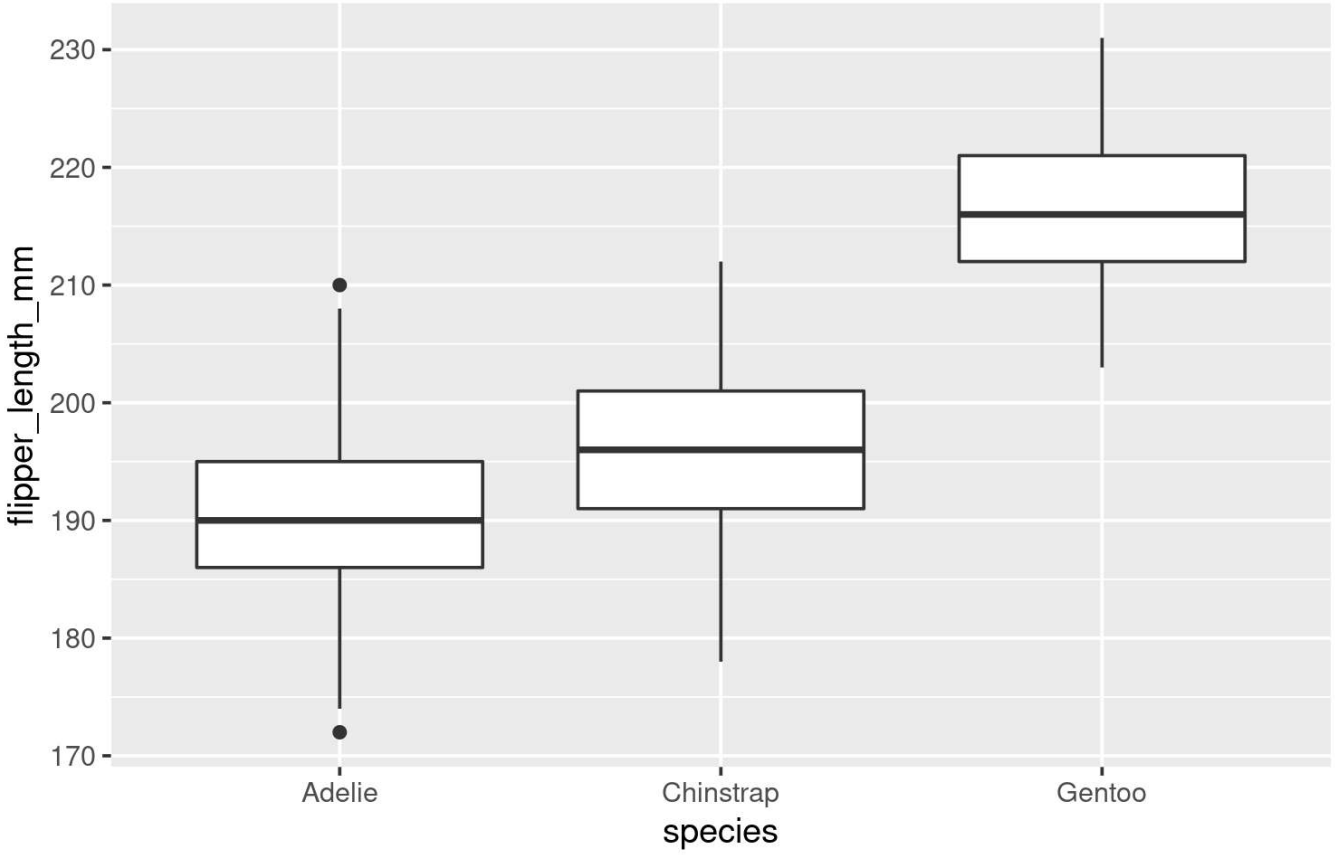
Geoms: Lines

```
ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm)) +
  geom_line()
```



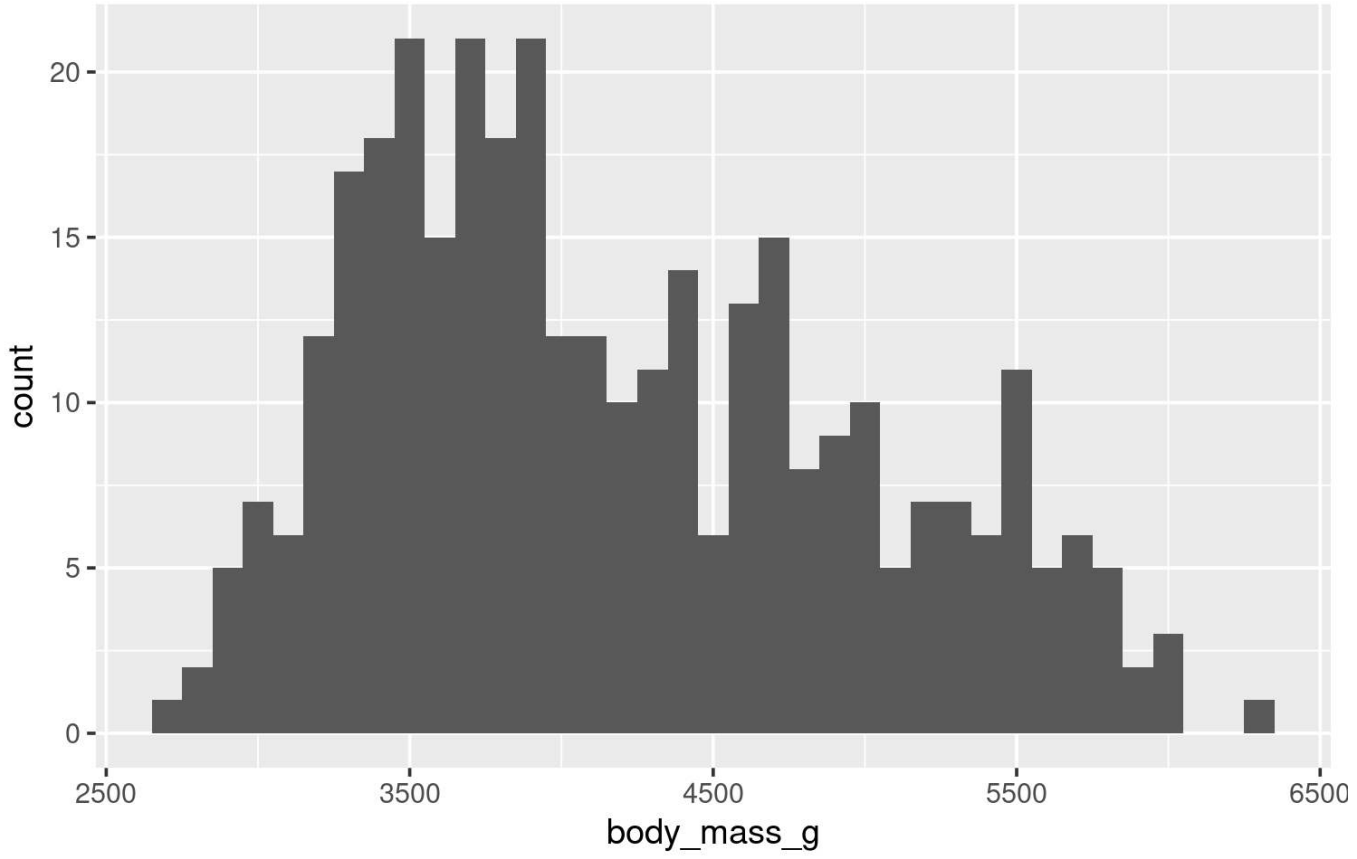
Geoms: Boxplots

```
ggplot(data = penguins, aes(x = species, y = flipper_length_mm)) +
  geom_boxplot()
```



Geoms: Histogram

```
ggplot(data = penguins, aes(x = body_mass_g)) +
  geom_histogram(binwidth = 100)
```



Geoms: Histogram

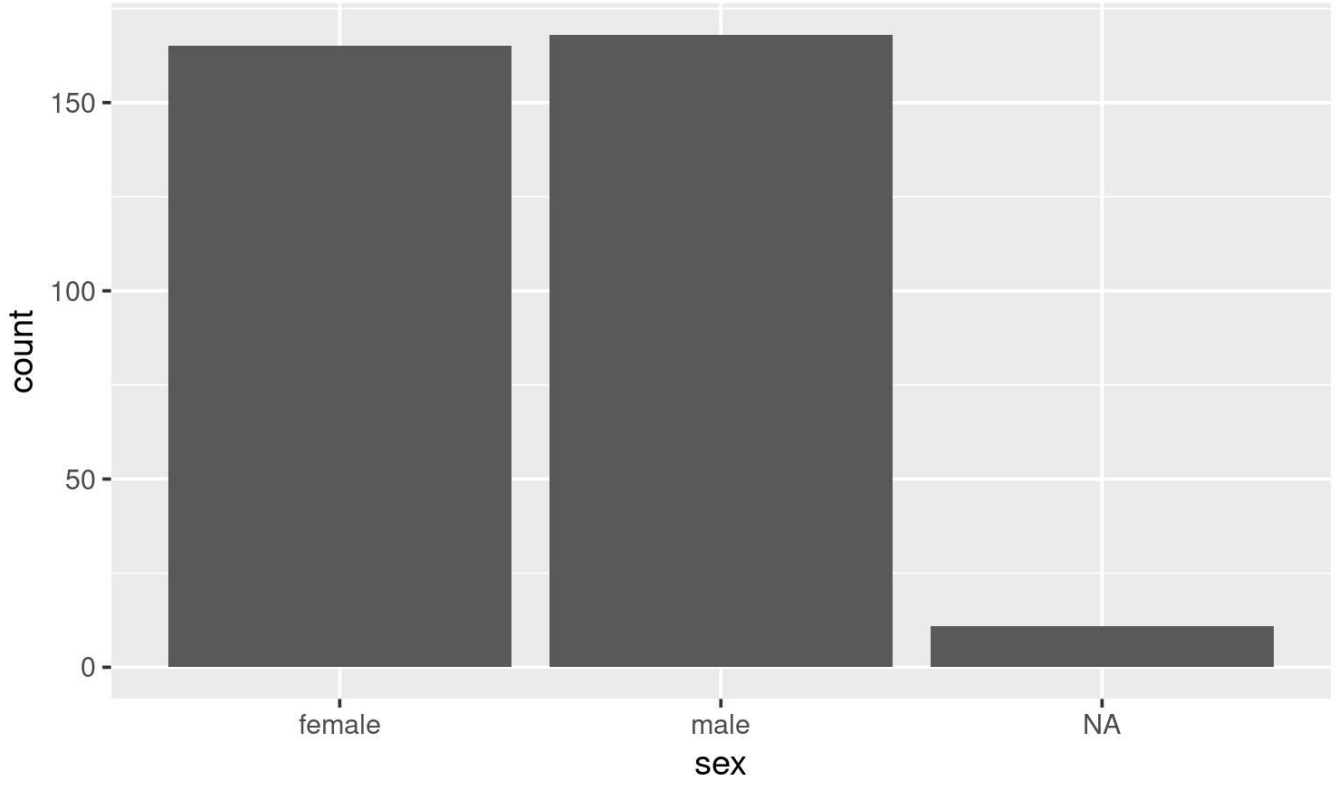
```
ggplot(data = penguins, aes(x = body_mass_g)) +
  geom_histogram(binwidth = 100)
```

Note: We only need 1 aesthetic here (x)

Geoms: Barplots

Let `ggplot` count your data

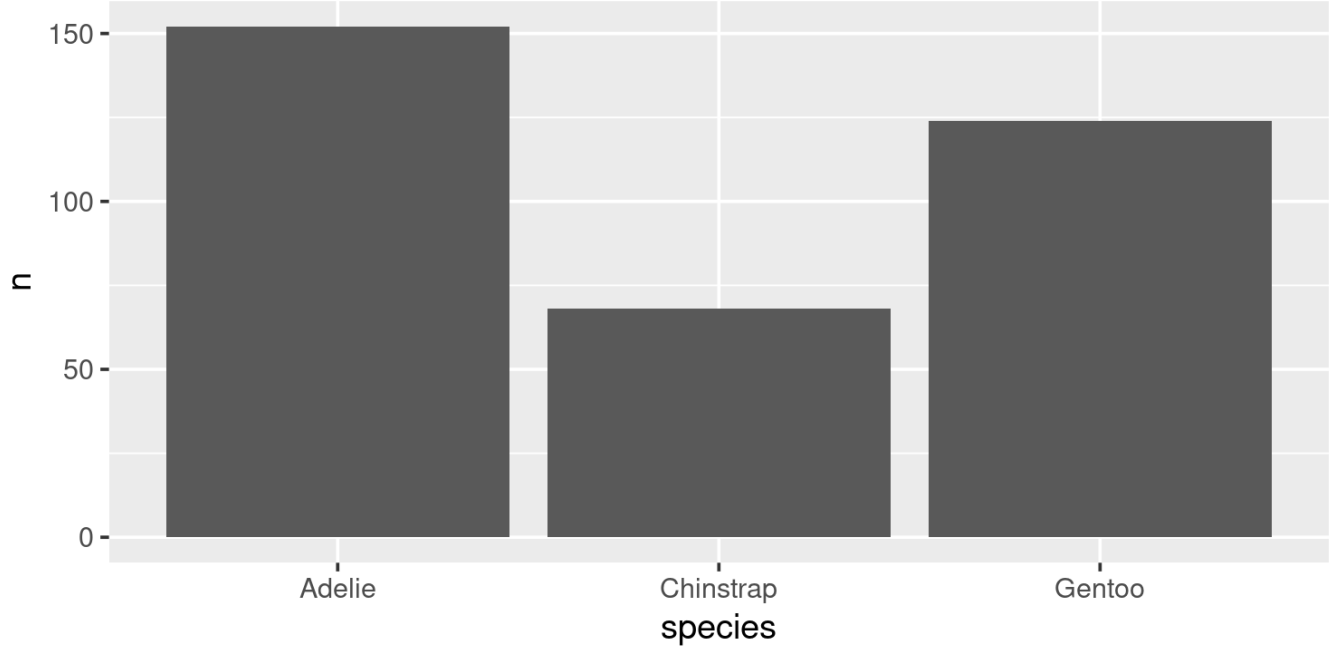
```
ggplot(data = penguins, aes(x = sex)) +
  geom_bar()
```



Geoms: Barplots

You can also provide the counts

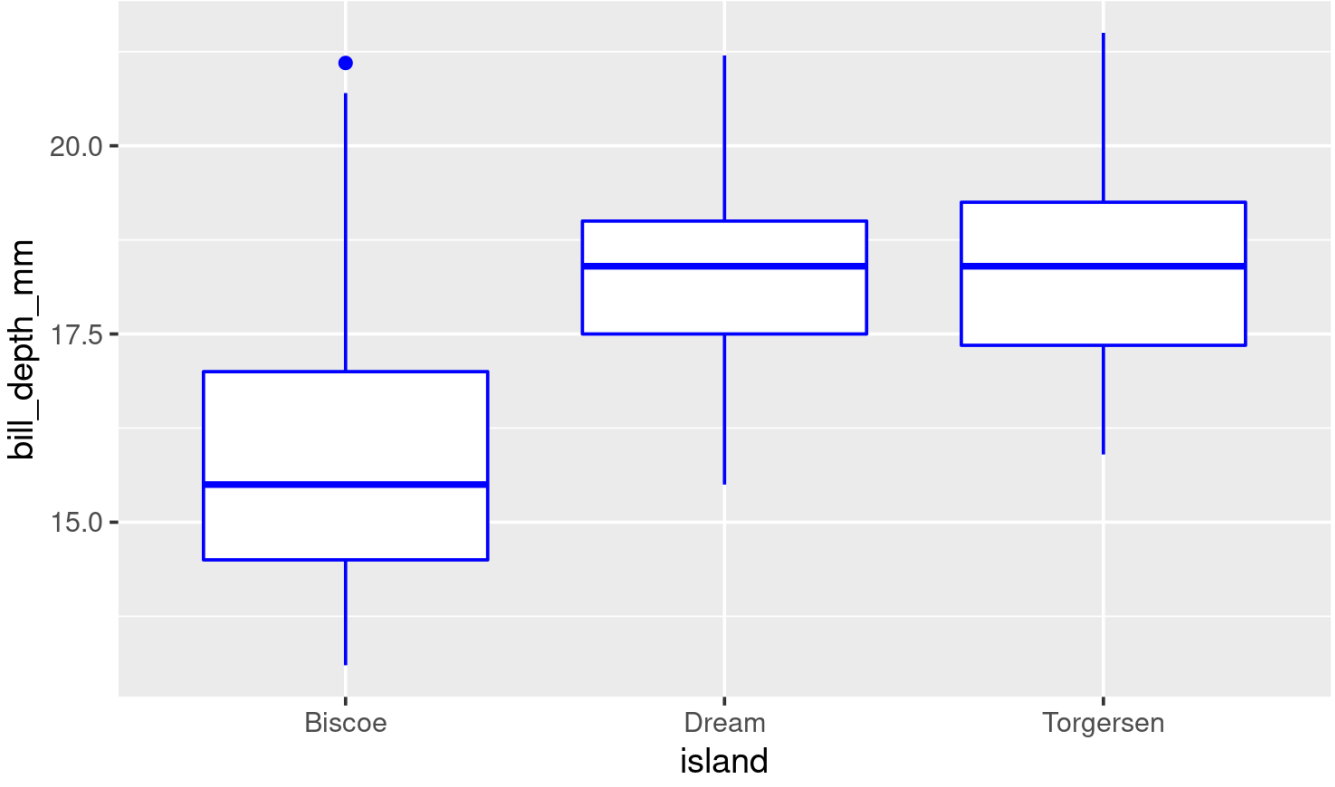
```
# Create our own data frame of counts
species <- count(penguins, species)

ggplot(data = species, aes(x = species, y = n)) +
  geom_bar(stat = "identity")
```



Your Turn: Create this plot

Extra Challenge
Plot points on top

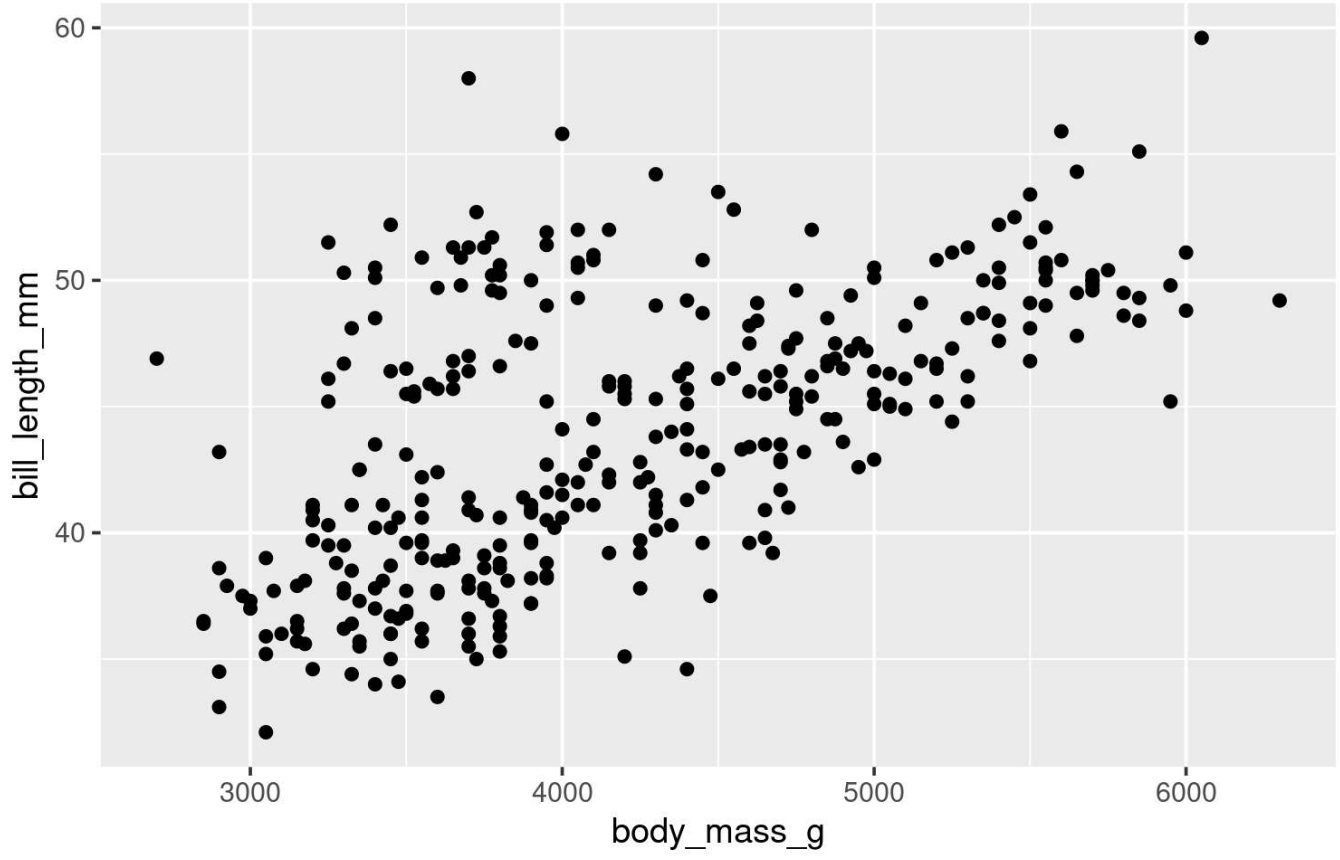
```
library(tidyverse)

ggplot(data =          , aes(x =       , y =              )) +
  geom_          (                )
```



Showing data by group

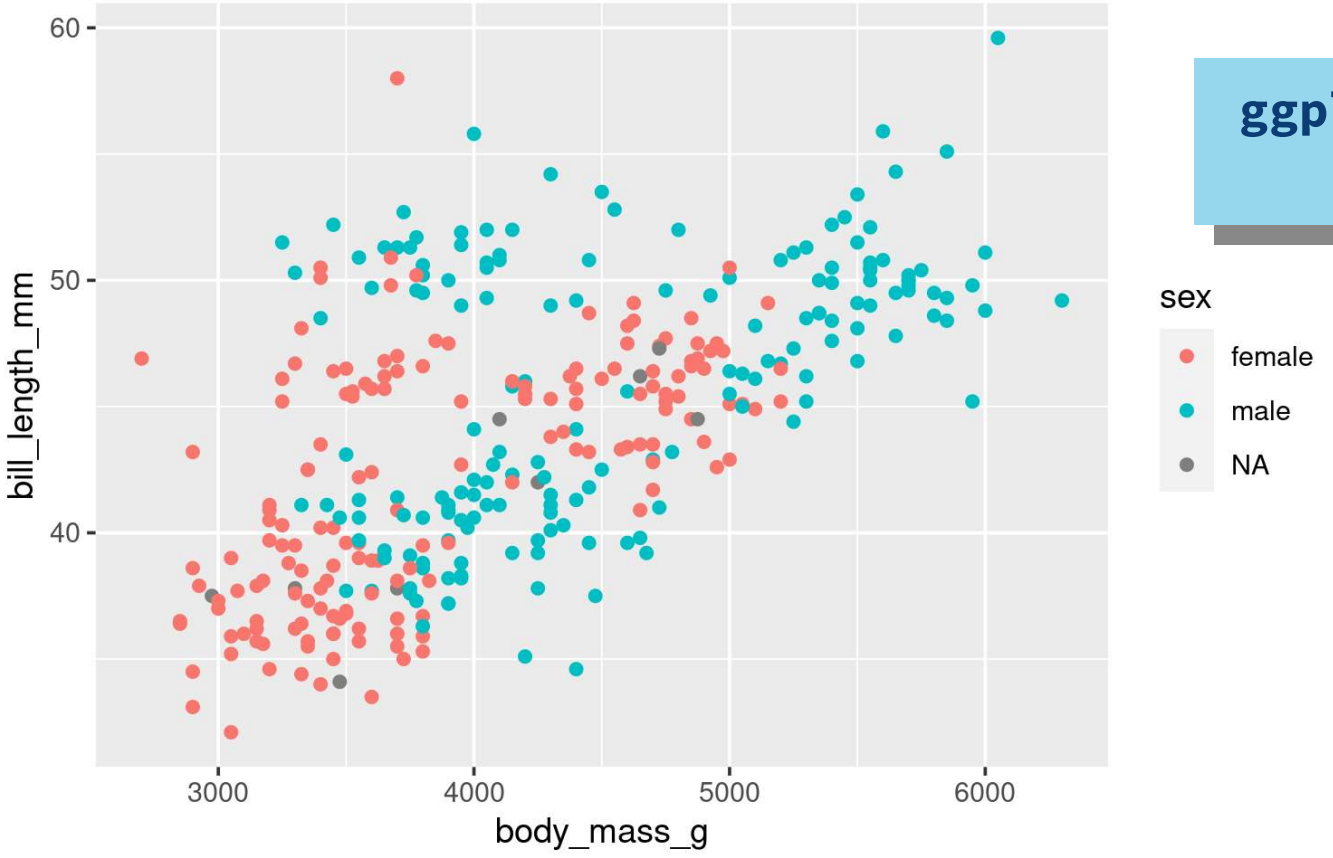
Mapping aesthetics

```
ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm)) +
  geom_point()
```



Mapping aesthetics

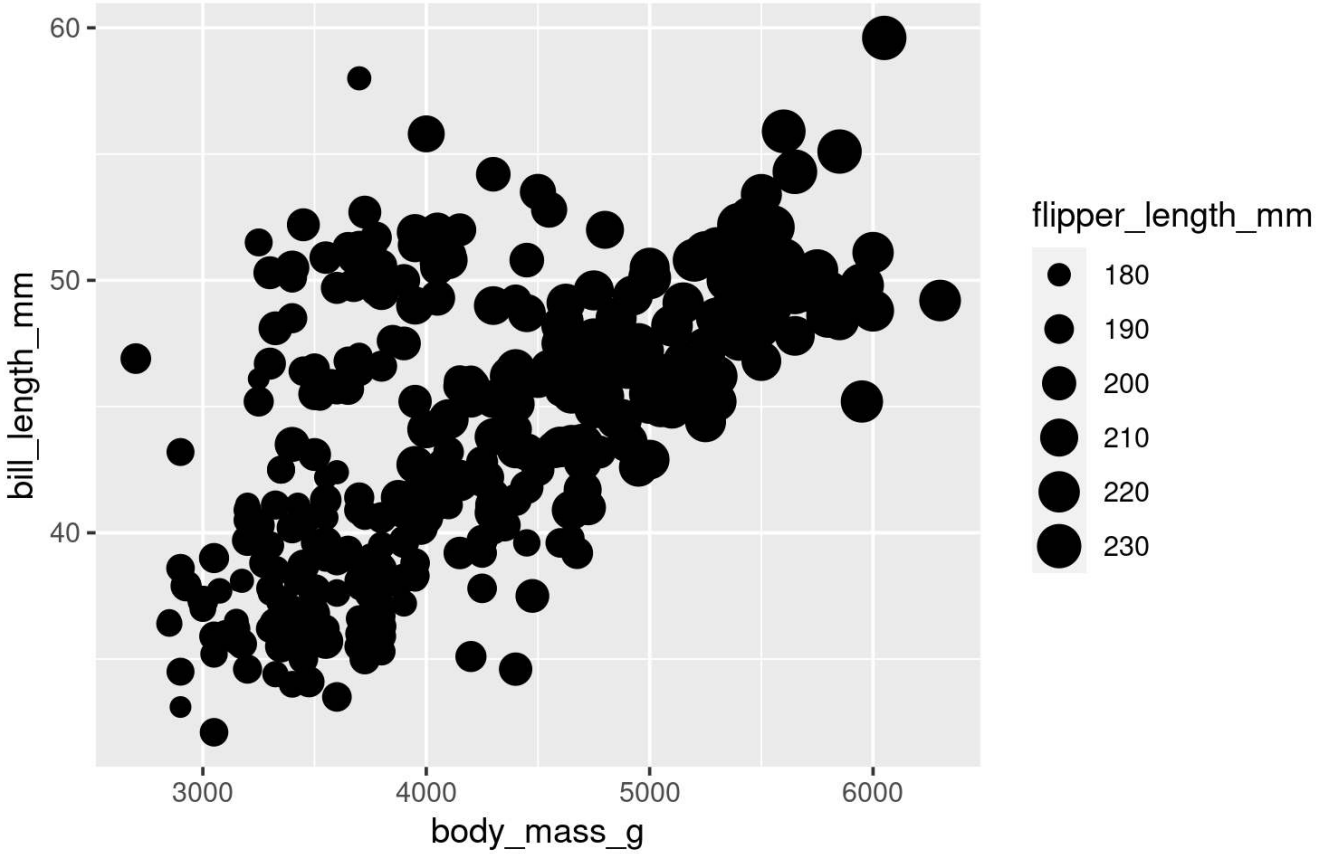
```
ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm, colour = sex)) +
  geom_point()
```



`ggplot()` automatically creates legends

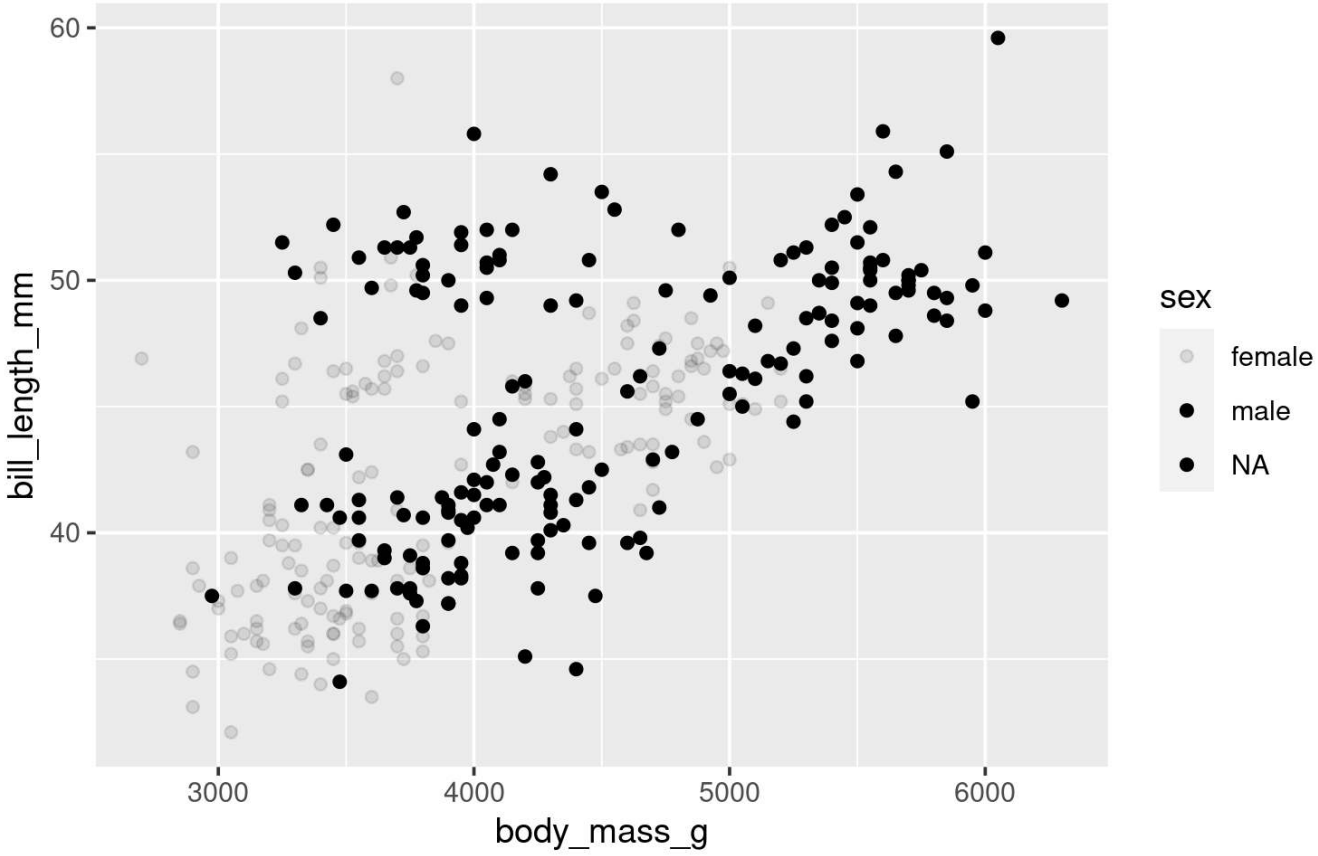
Mapping aesthetics

```
ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm, size = flipper_length_mm)) +
  geom_point()
```



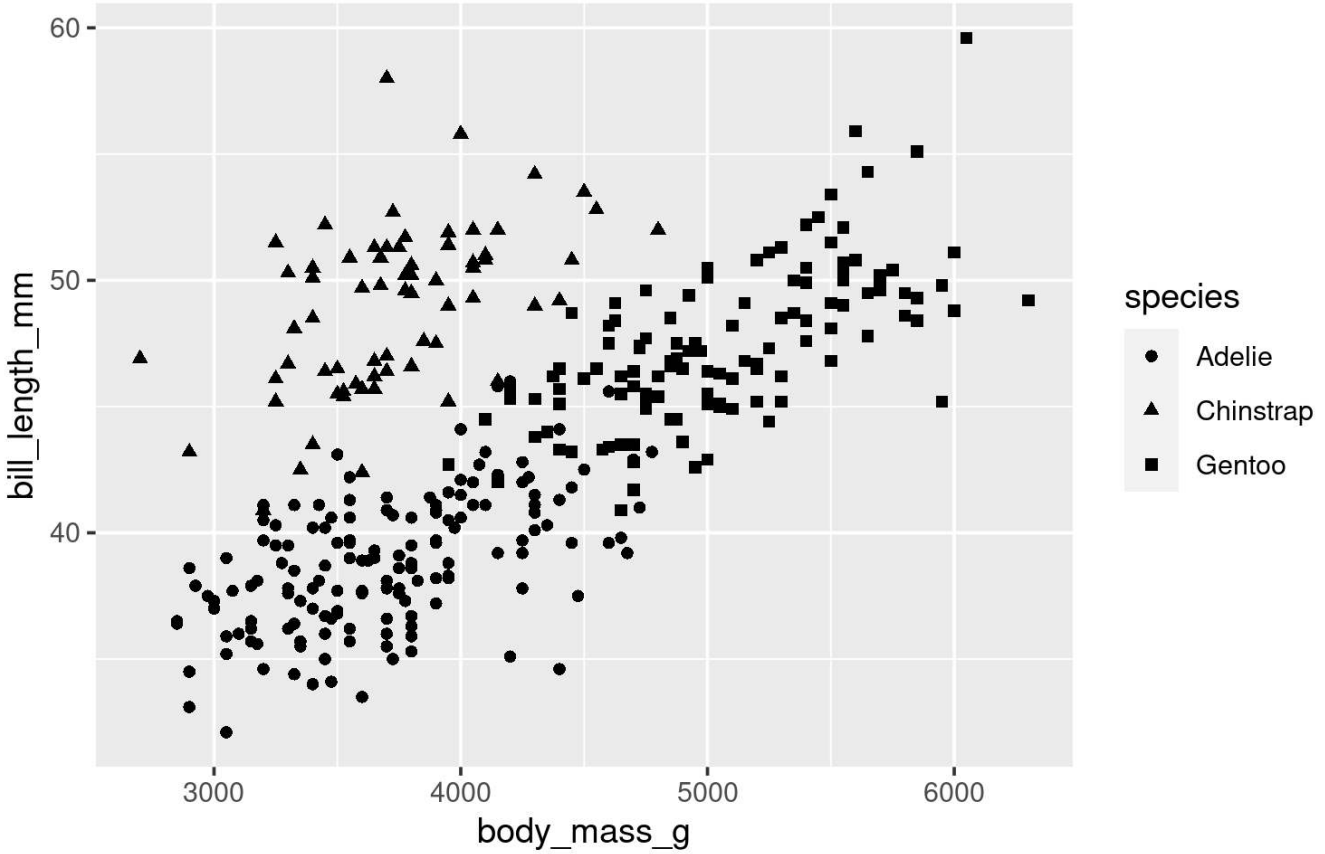
Mapping aesthetics

```
ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm, alpha = sex)) +
  geom_point()
```



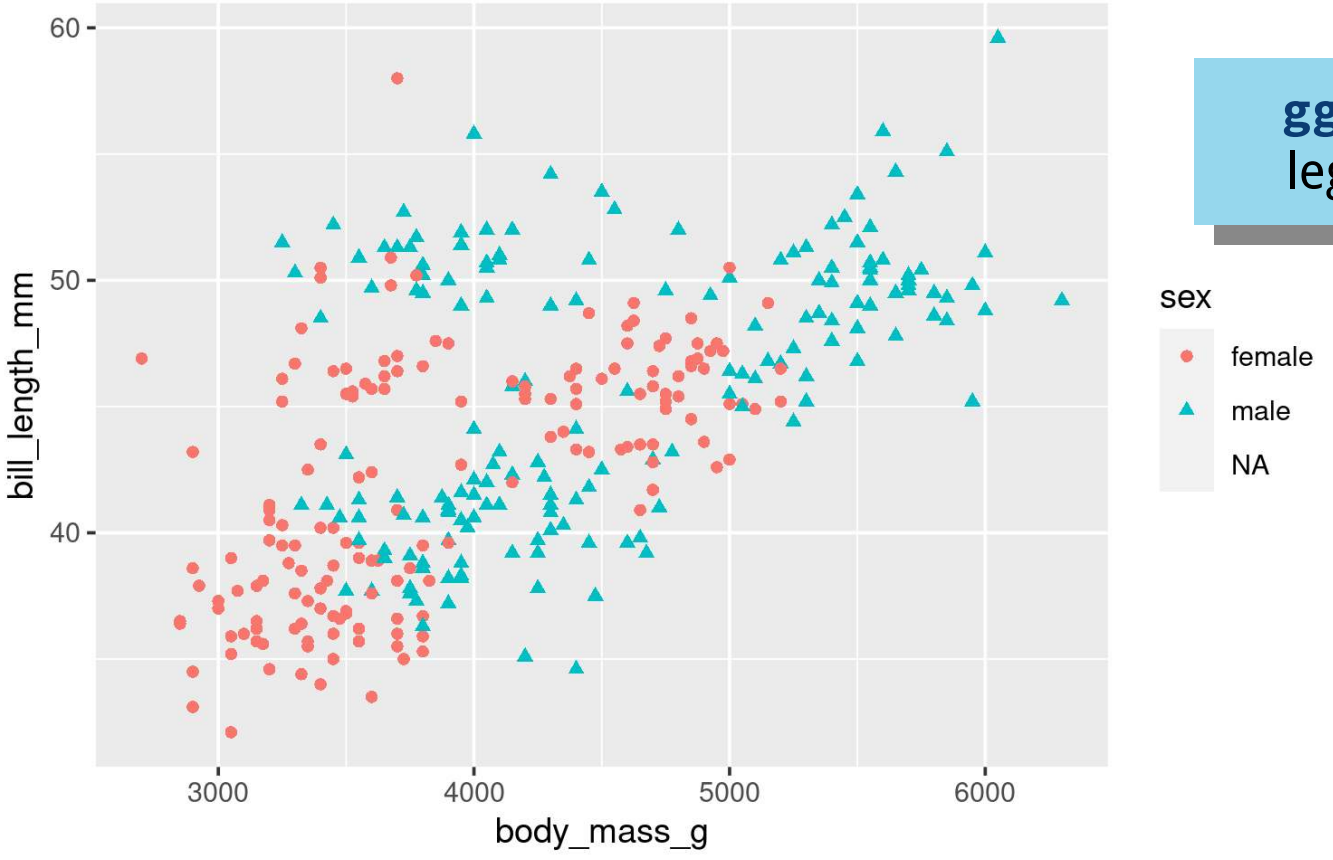
Mapping aesthetics

```
ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm, shape = species)) +
  geom_point()
```



Mapping aesthetics

```
ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm, colour = sex, shape = sex)) +
  geom_point()
```



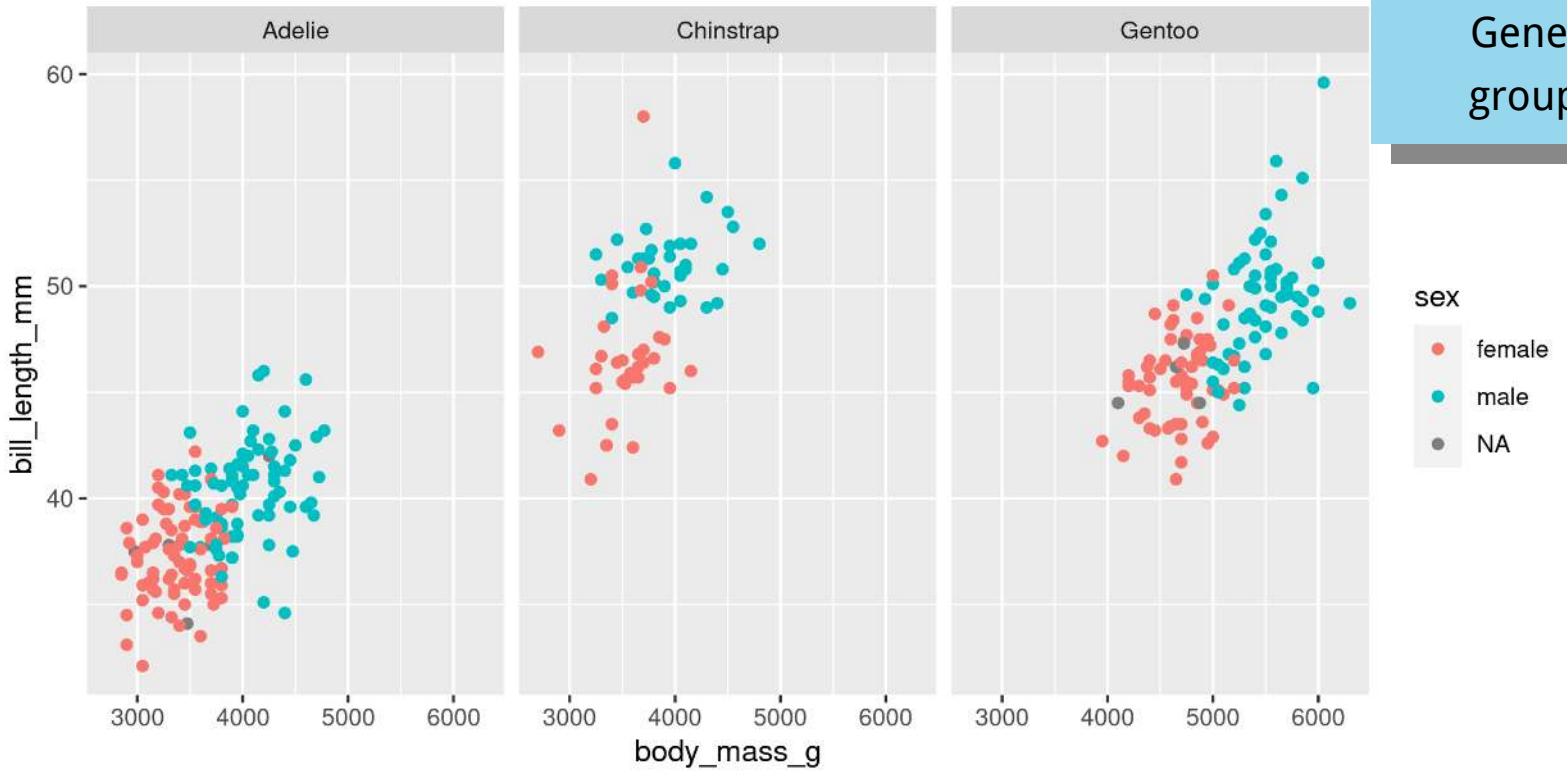
`ggplot()` combines legends where it can

Faceting: `facet_wrap()`

```
ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm, colour = sex)) +
  geom_point() +
  facet_wrap(~ species)
```

Split plots by wrapping

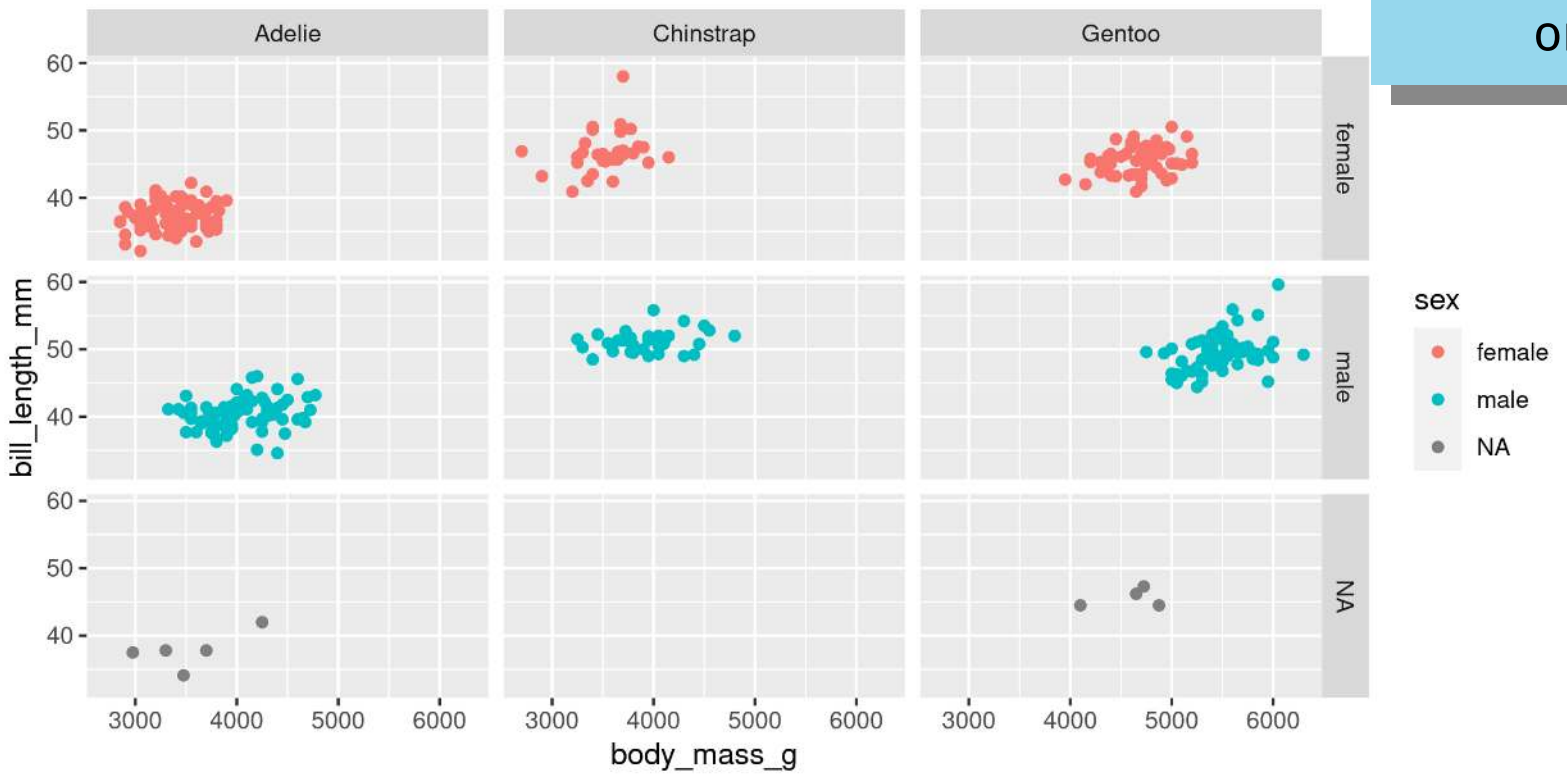
Generally for one grouping variable



Faceting: facet_grid()

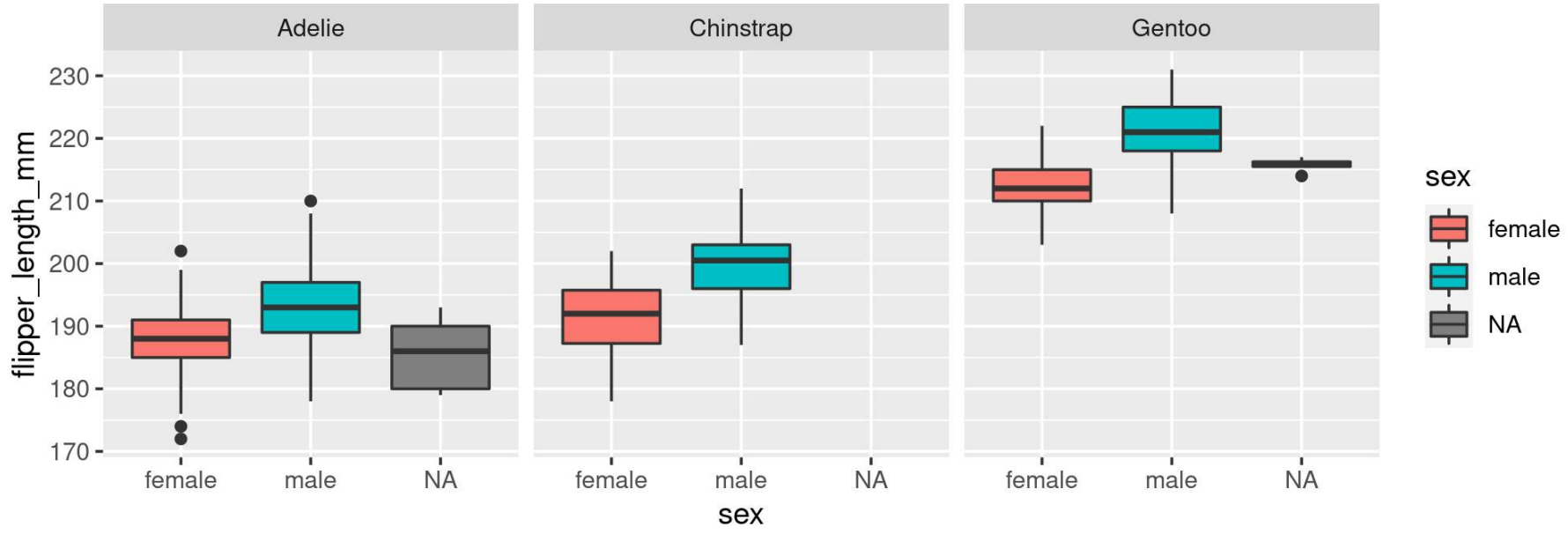
```
ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm, colour = sex)) +
  geom_point() +
  facet_grid(sex ~ species)
```

Split plots by **two** grouping variables on a grid



Your Turn: Create this plot

```
ggplot(data =          , aes(                                    )) +
   +
```

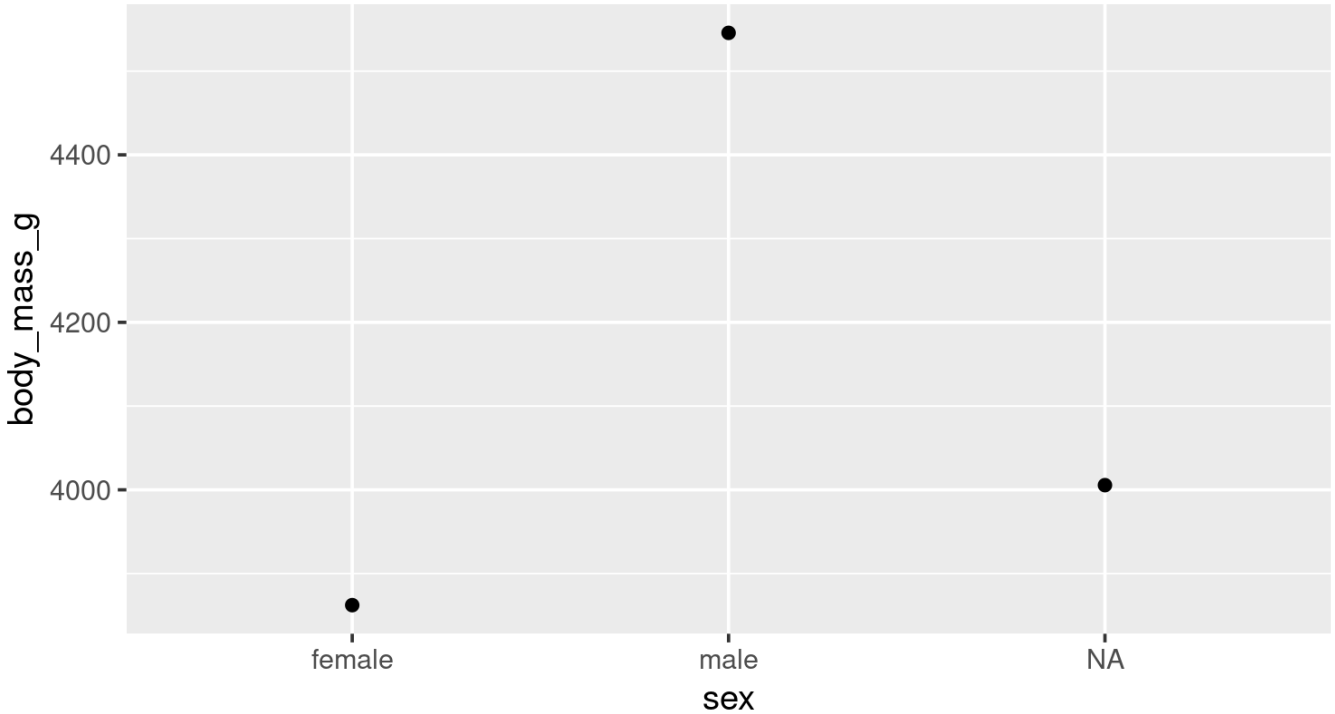


Hint: `colour` is for outlining with a colour, `fill` is for 'filling' with a colour
Extra Challenge: Split boxplots by sex **and** island

Summarizing data

Add data means as points

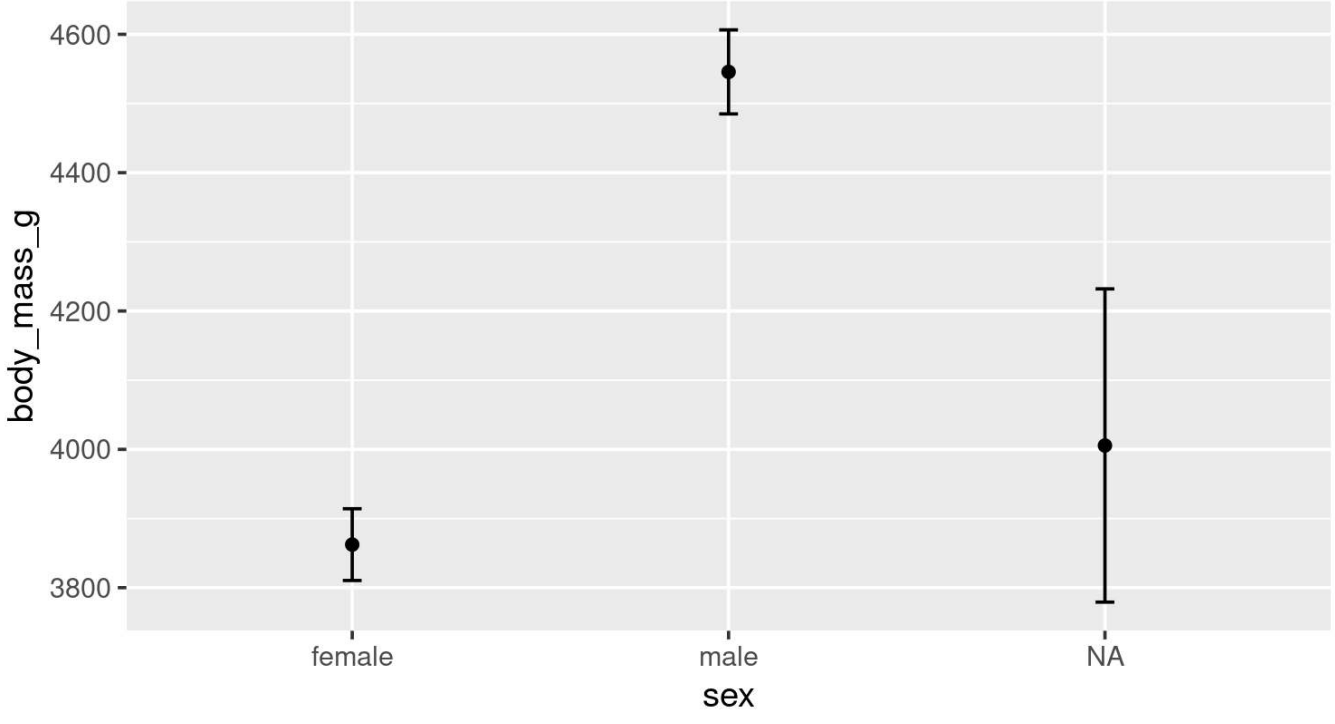
```
ggplot(data = penguins, aes(x = sex, y = body_mass_g)) +
  stat_summary(geom = "point", fun = mean)
```



Summarizing data

Add error bars, calculated from the data

```
ggplot(data = penguins, aes(x = sex, y = body_mass_g)) +
  stat_summary(geom = "point", fun = mean) +
  stat_summary(geom = "errorbar", width = 0.05, fun.data = mean_se)
```

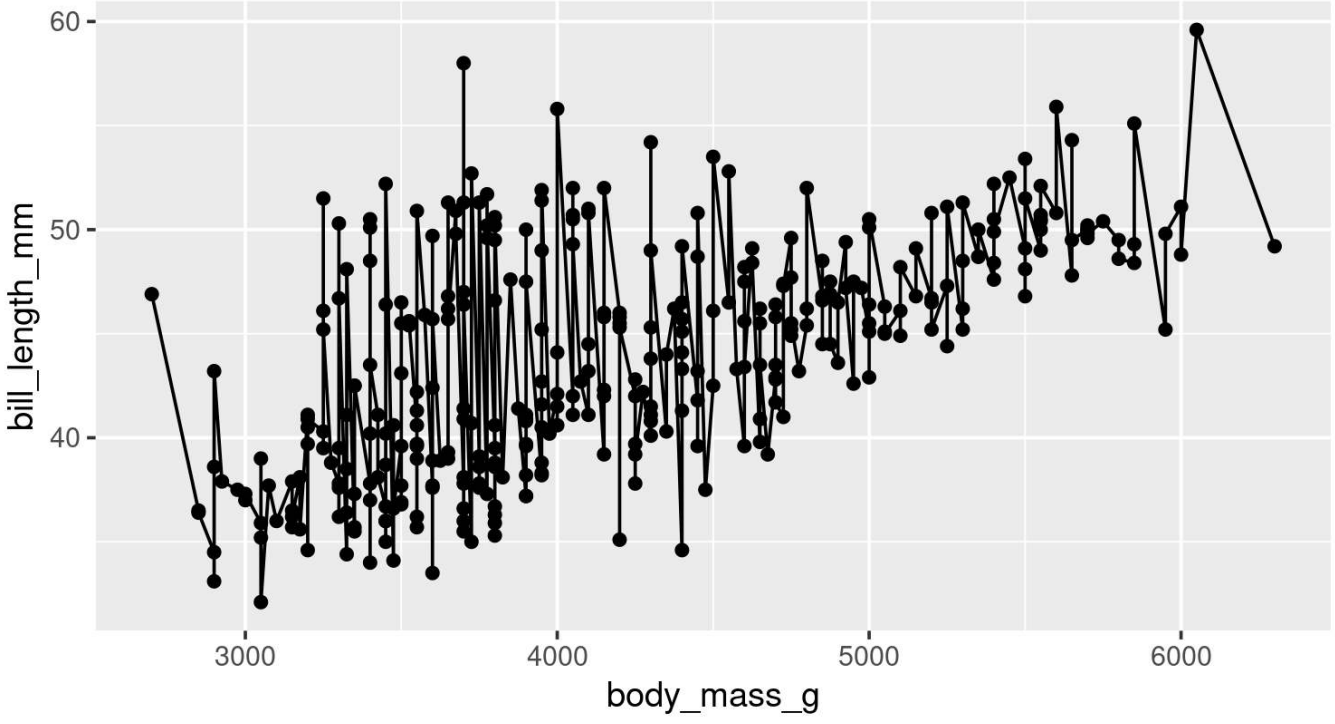


Trendlines / Regression Lines

Trendlines / Regression lines

`geom_line()` is connect-the-dots, not a trend or linear model

```
ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm)) +
  geom_point() +
  geom_line()
```

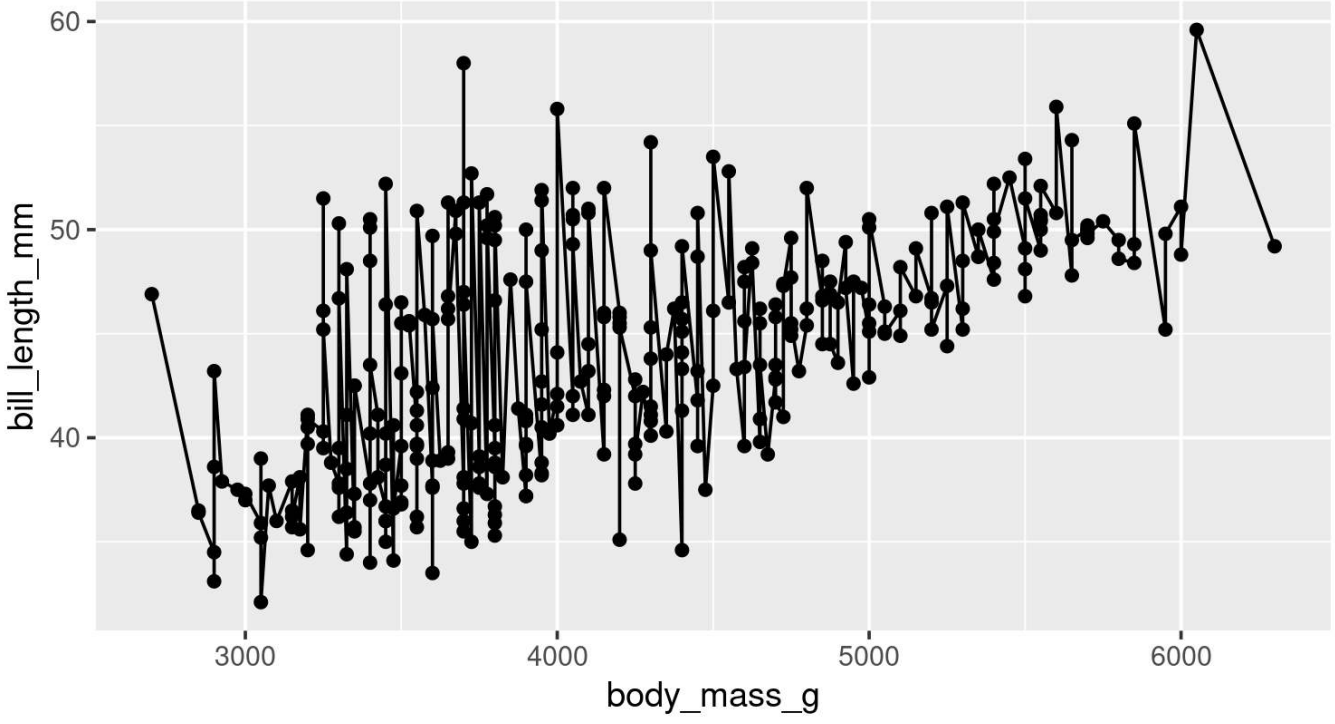


Trendlines / Regression lines

`geom_line()` is connect-the-dots, not a trend or linear model

```
ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm)) +
  geom_point() +
  geom_line()
```

Not what we're looking for

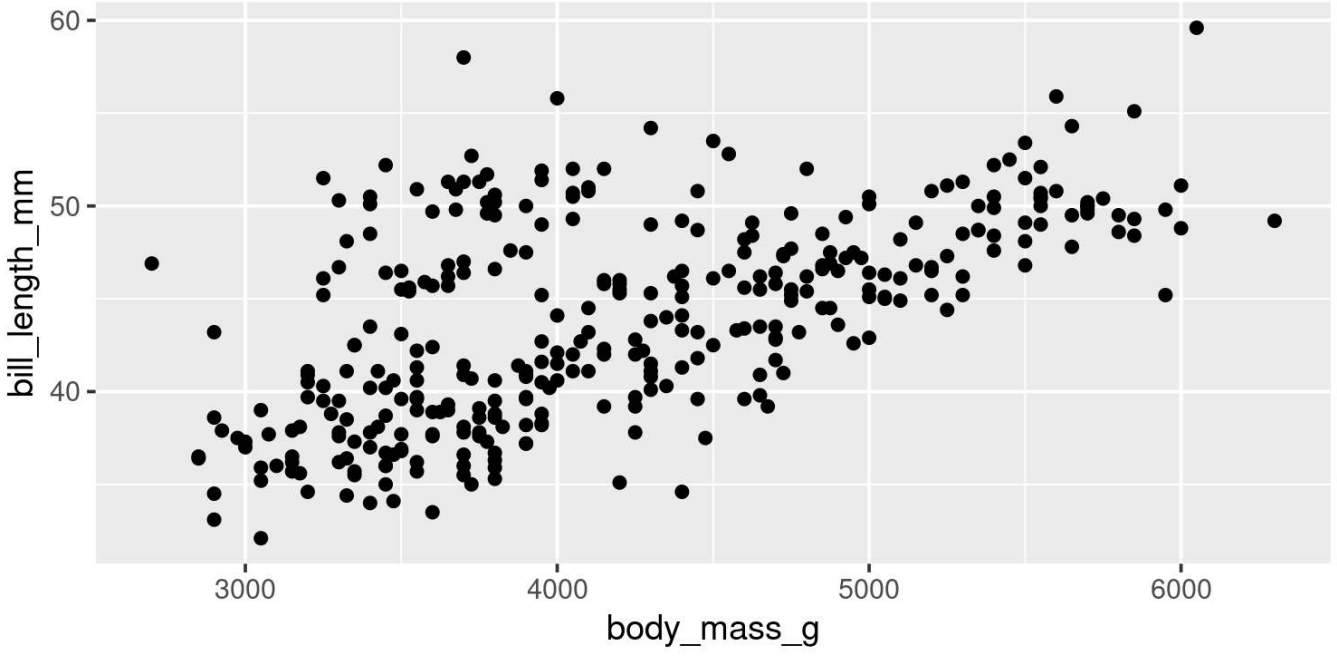


Trendlines / Regression lines

Let's add a trend line properly

Start with basic plot:

```
g <- ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm)) +
  geom_point()
g
```

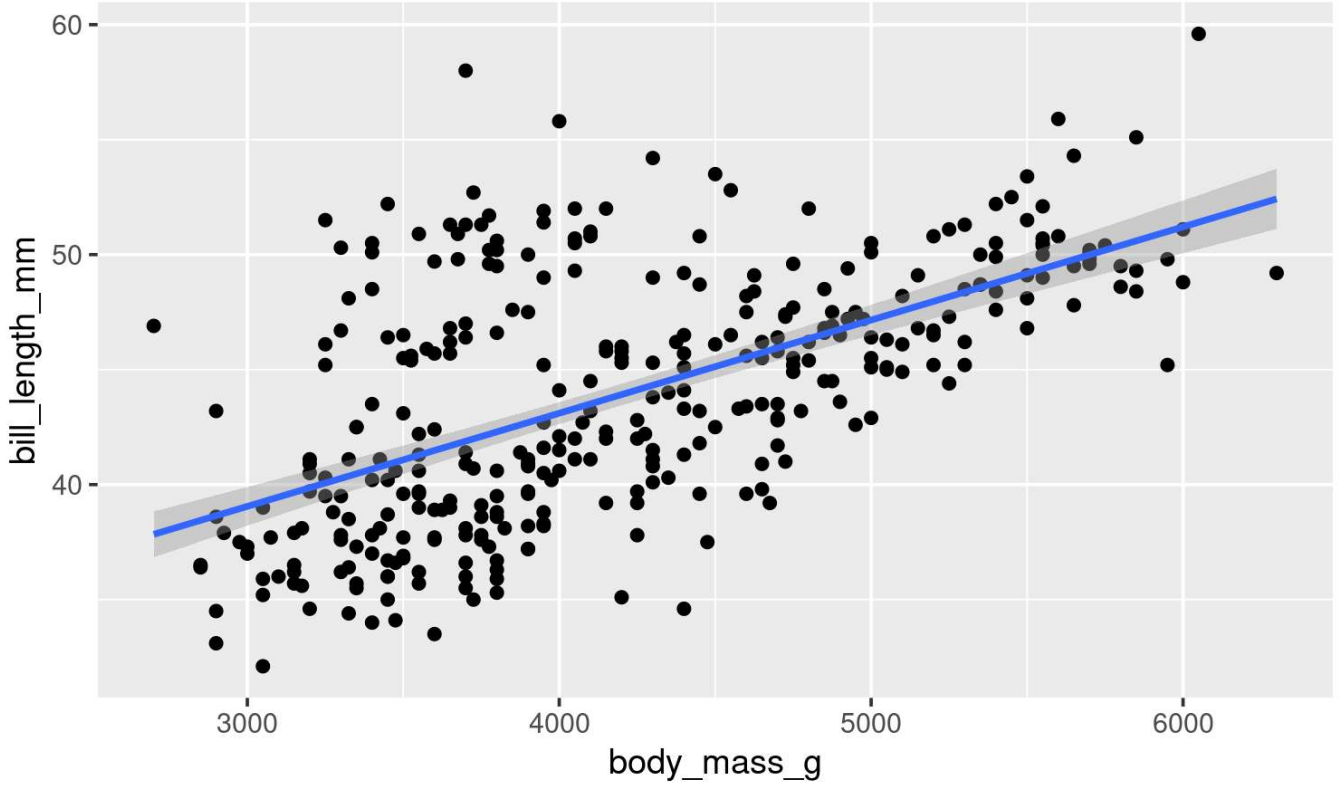


Trendlines / Regression lines

Add the `stat_smooth()`

- `lm` is for "linear model" (i.e. trendline)
- grey ribbon = standard error

```
g + stat_smooth(method = "lm")
```

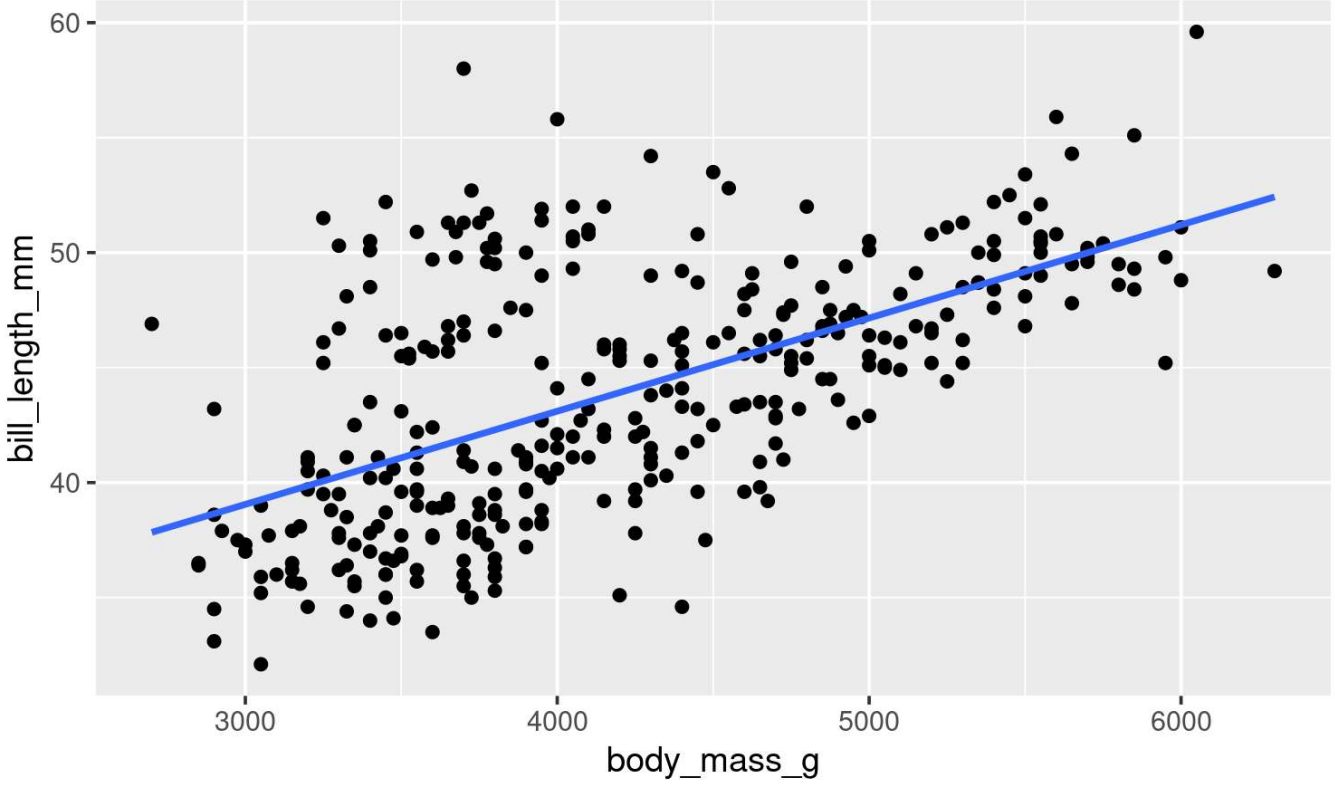


Trendlines / Regression lines

Add the `stat_smooth()`

- remove the grey ribbon **`se = FALSE`**

```
g + stat_smooth(method = "lm", se = FALSE)
```

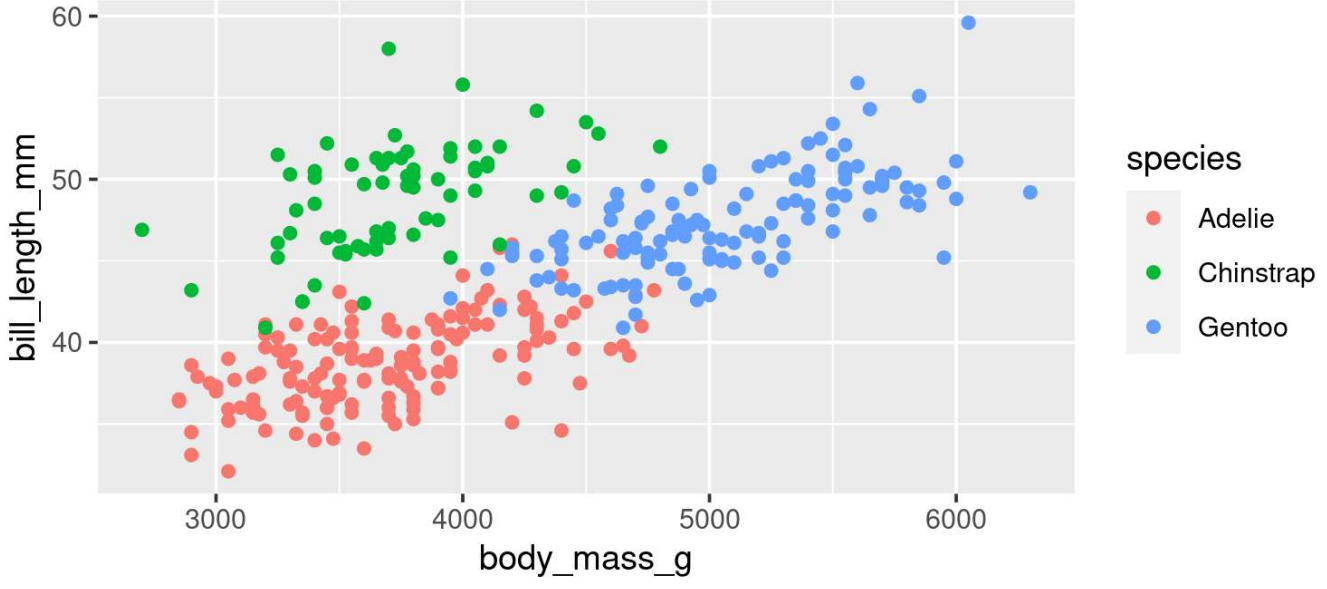


Trendlines / Regression lines

A line for each group

- Specify group (here we use **`colour`** to specify **`species`**)

```
g <- ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm, colour = species)) +
  geom_point()
g
```

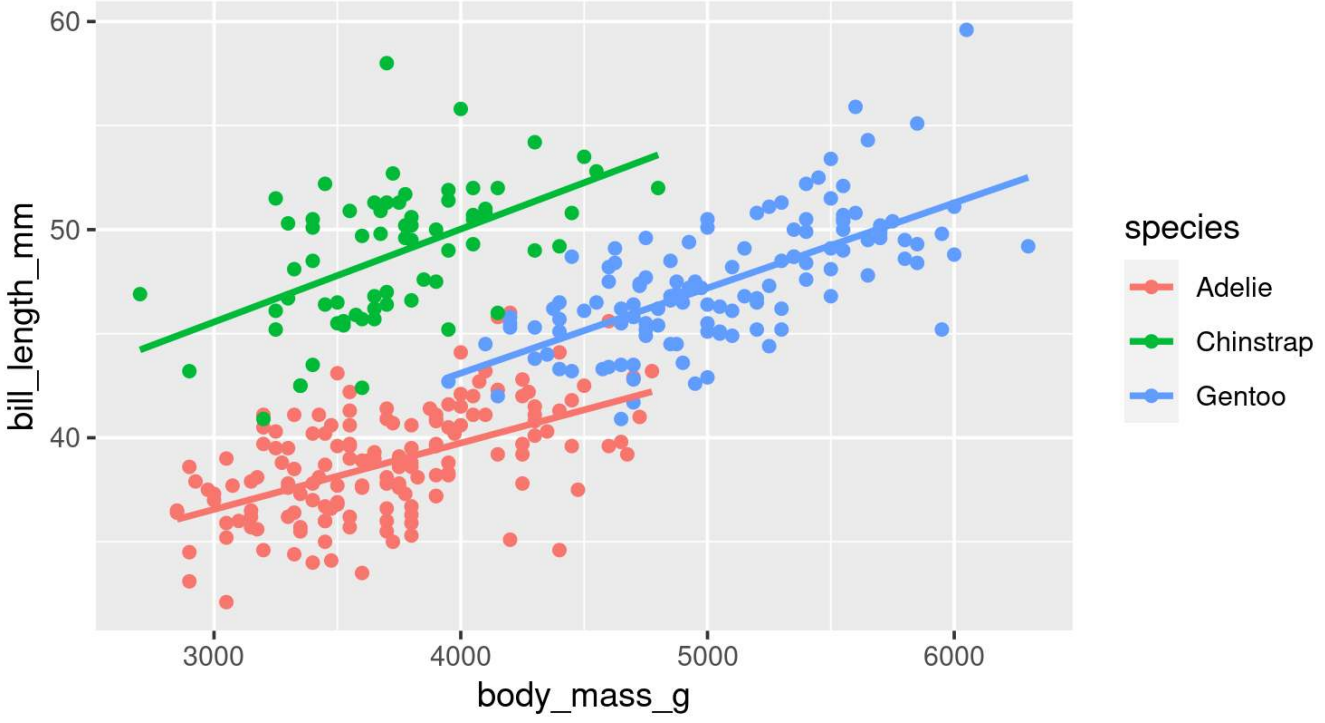


Trendlines / Regression lines

A line for each group

- **`stat_smooth()`** automatically uses the same grouping

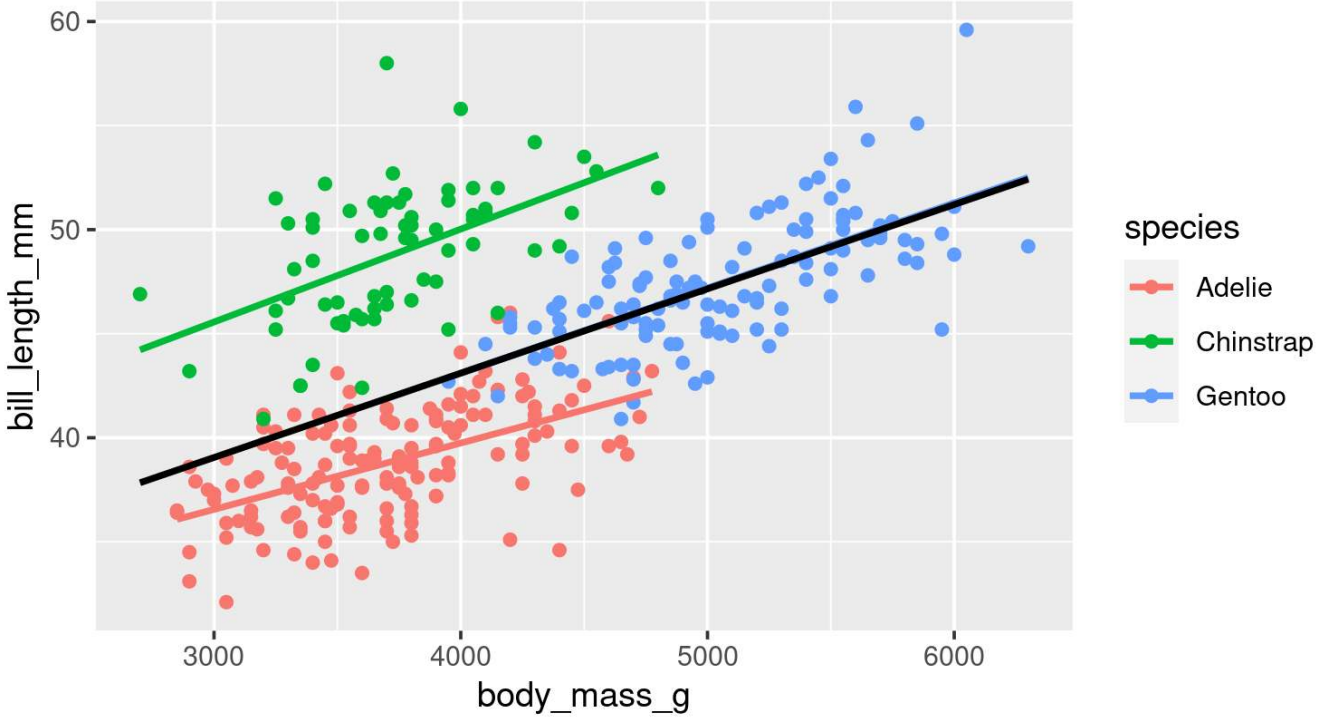
```
g + stat_smooth(method = "lm", se = FALSE)
```



Trendlines / Regression lines

A line for each group AND overall

```
g +
  stat_smooth(method = "lm", se = FALSE) +
  stat_smooth(method = "lm", se = FALSE, colour = "black")
```



Your Turn: Create this plot

- A scatter plot: **Flipper Length** by **Body Mass** grouped by **Species**
- With *a single regression line for the overall trend*

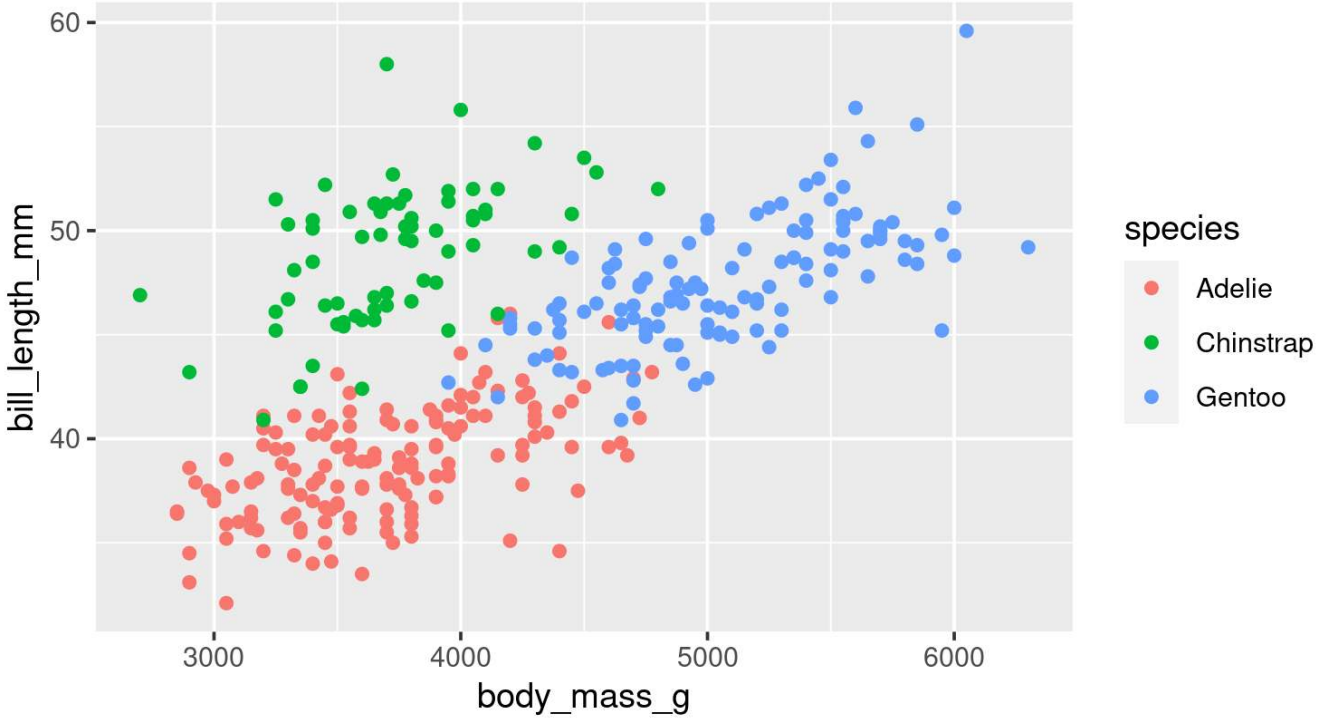
> **Extra Challenge:** Create a separate plot for each sex as well

Customizing plots

Customizing: Starting plot

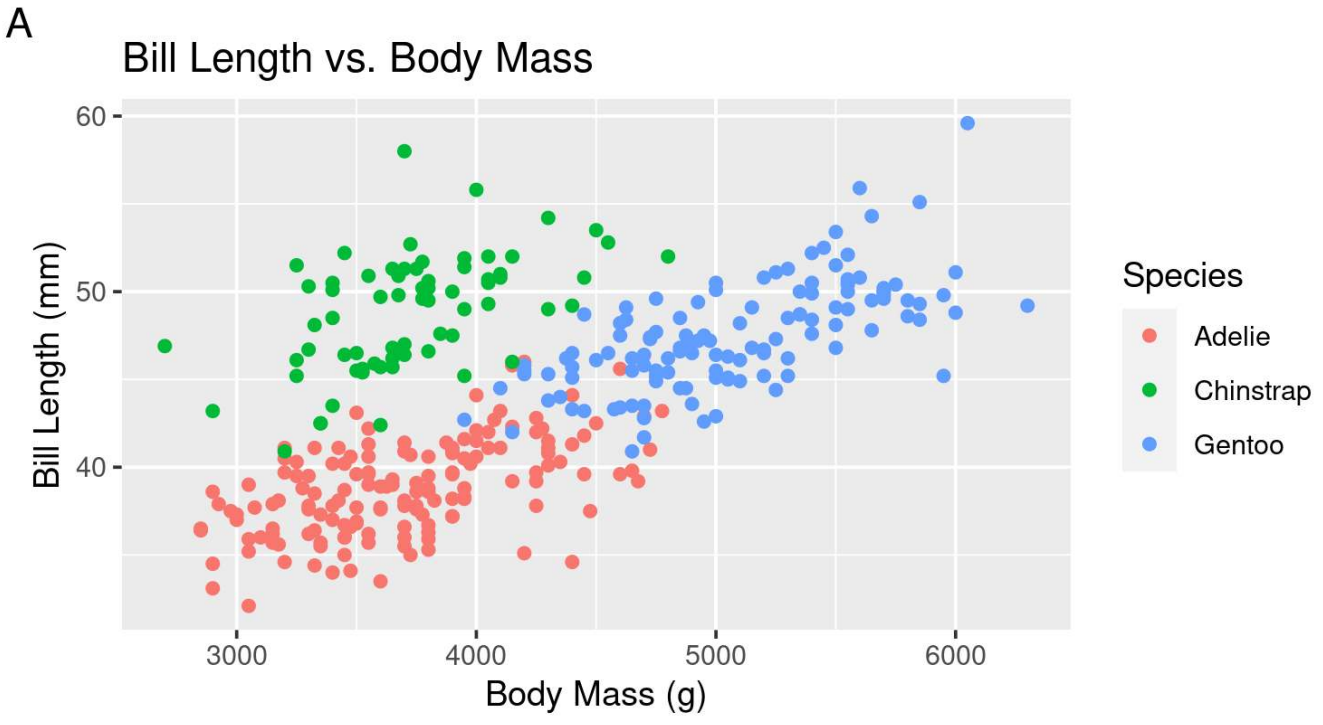
Let's work with this plot

```
g <- ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm, colour = species)) +
  geom_point()
```



Customizing: Labels

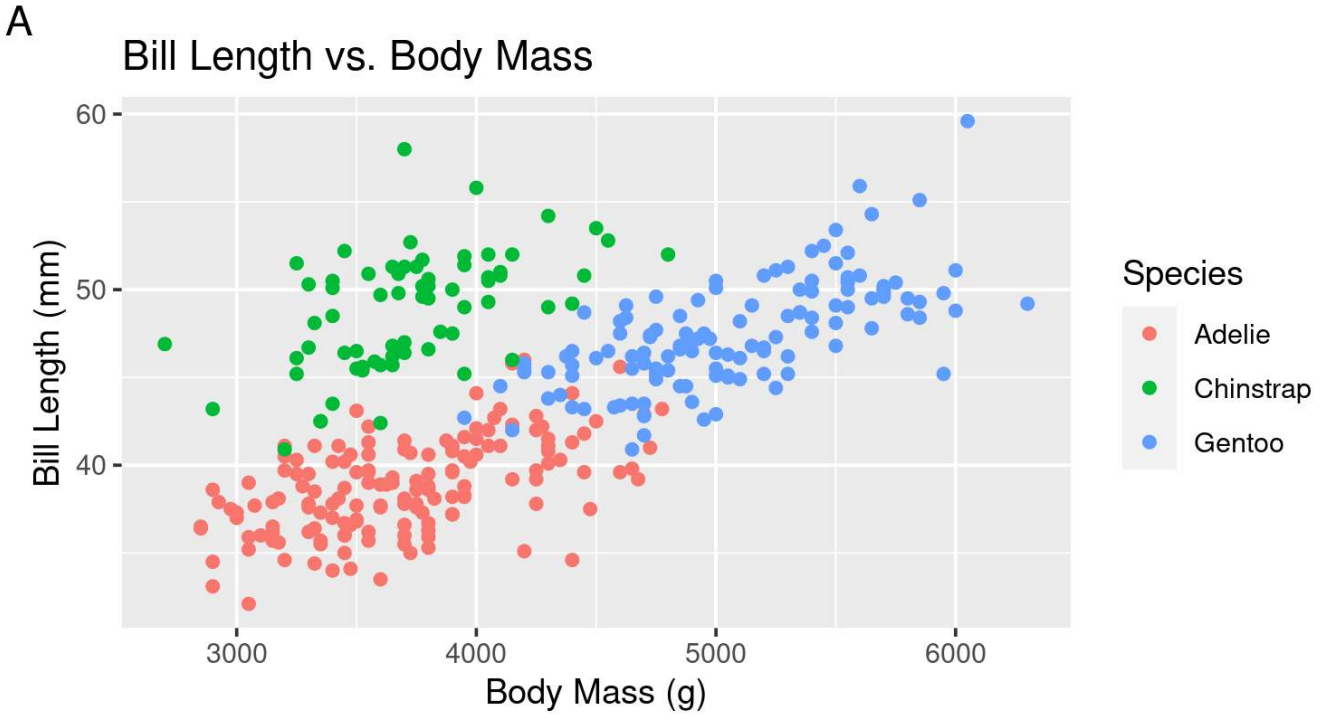
```
g + labs(title = "Bill Length vs. Body Mass",
         x = "Body Mass (g)",
         y = "Bill Length (mm)",
         colour = "Species", tag = "A")
```



Customizing: Labels

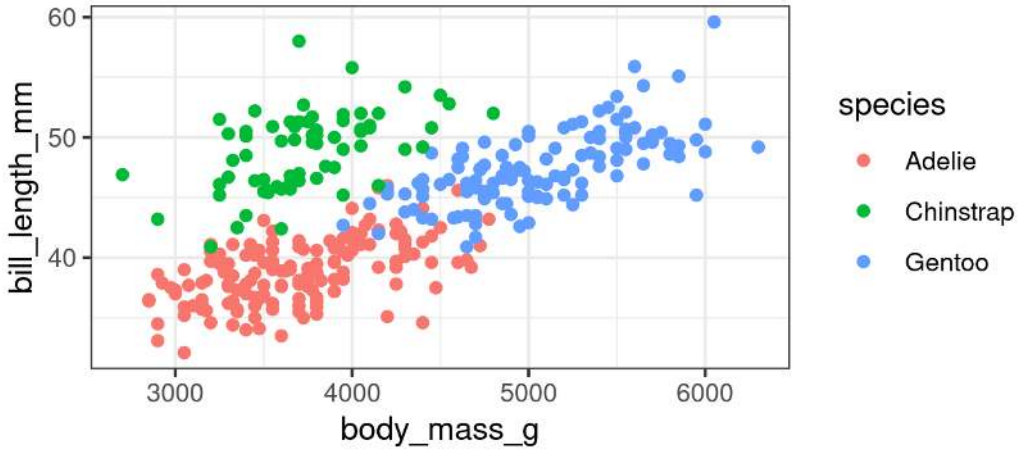
```
g + labs(title = "Bill Length vs. Body Mass",
         x = "Body Mass (g)",
         y = "Bill Length (mm)",
         colour = "Species", tag = "A")
```

Your Turn:
Add proper labels to one of your previous plots

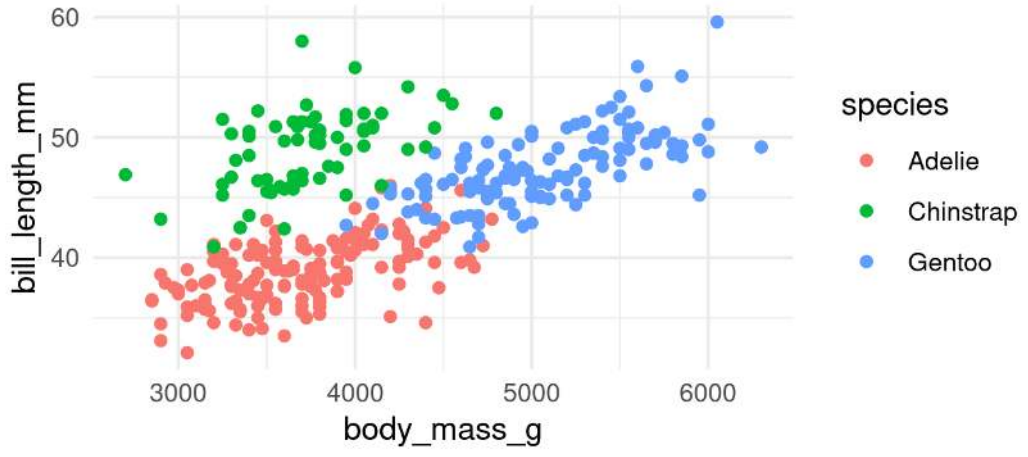


Customizing: Built-in themes

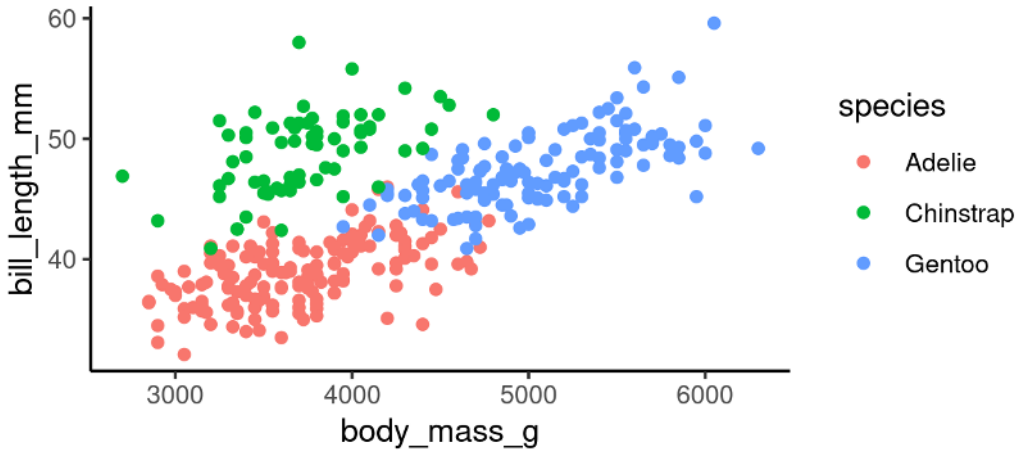
g + theme_bw()



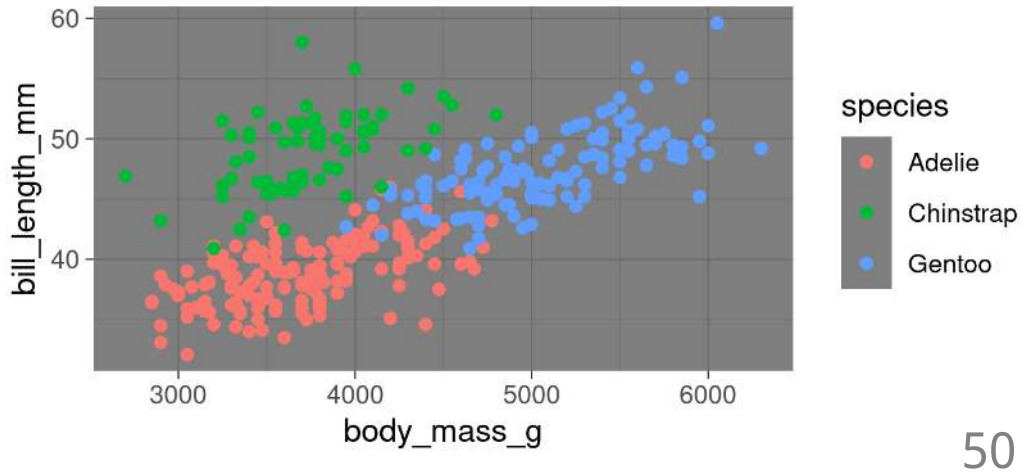
g + theme_minimal()



g + theme_classic()



g + theme_dark()



Customizing: Axes

scale_ + (**x** or **y**) + type (**contiuous**, **discrete**, **date**, **datetime**)

- **scale_x_continuous()**
- **scale_y_discrete()**
- etc.

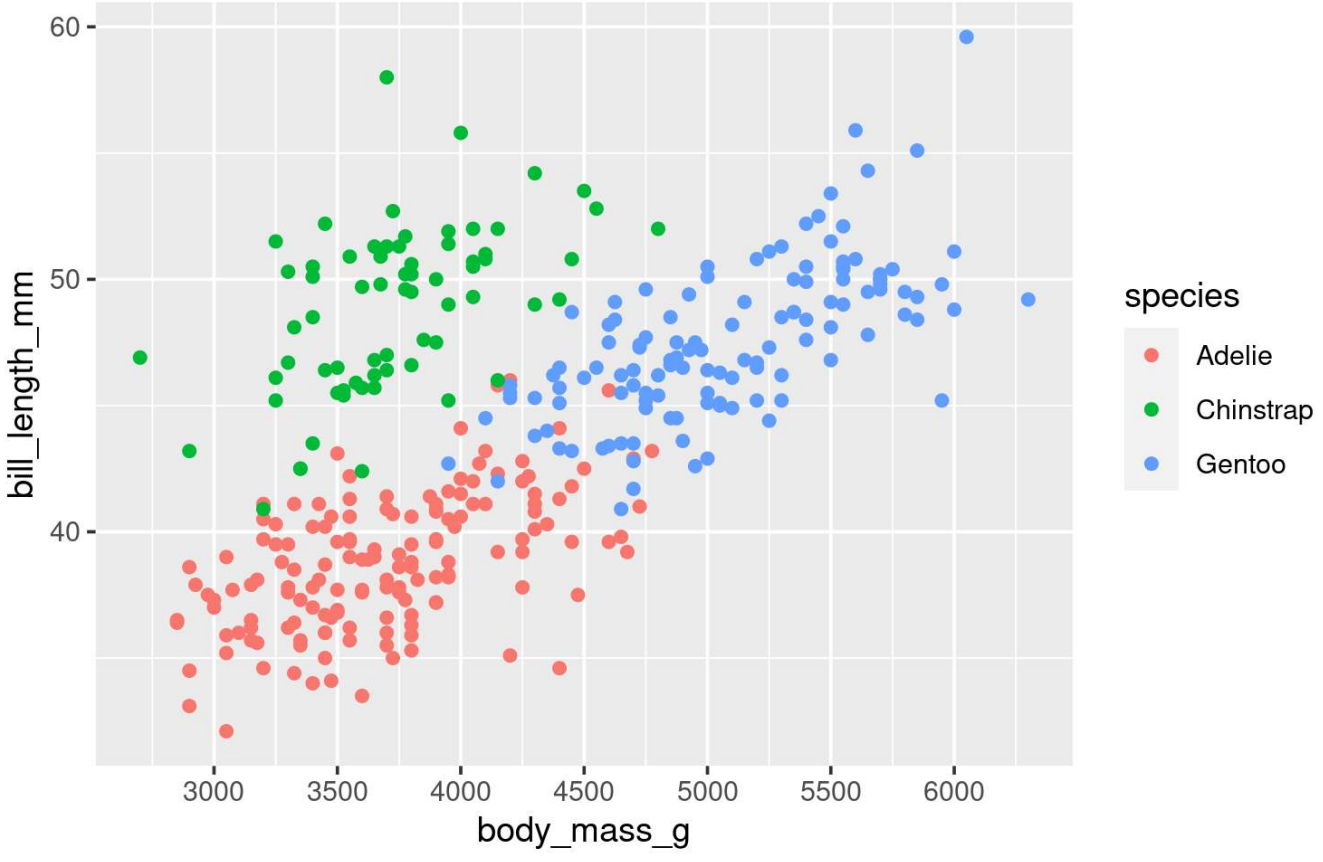
Common arguments

```
g + scale_x_continuous(breaks = seq(0, 20, 10)) # Tick breaks
g + scale_x_continuous(limits = c(0, 15))       # xlim() is a shortcut for this
g + scale_x_continuous(expand = c(0, 0))        # Space between axis and data
```

Customizing: Axes

Breaks

```
g + scale_x_continuous(breaks = seq(2500, 6500, 500))
```



Customizing: Axes

Limits

```
g + scale_x_continuous(limits = c(3000, 4000))
```

Customizing: Axes

Space between origin and axis start

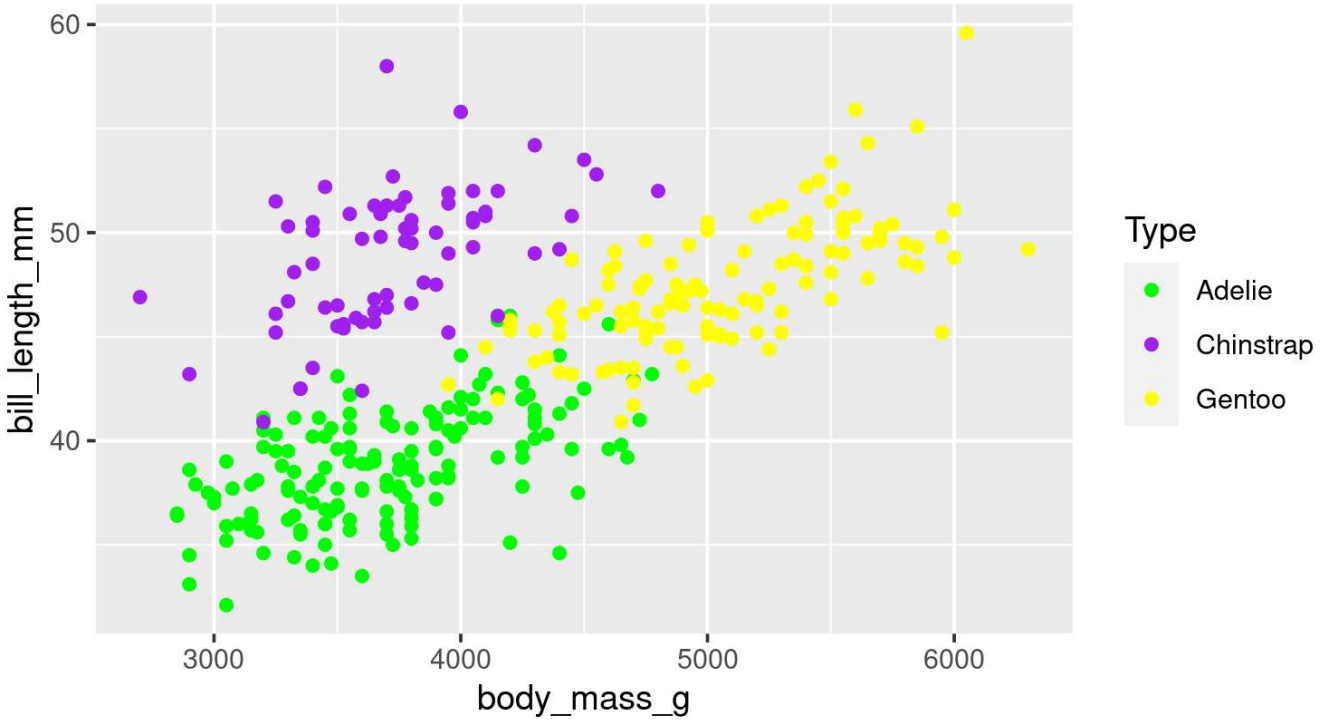
```
g + scale_x_continuous(expand = c(0, 0))
```

Customizing: Aesthetics

Using scales

`scale_` + aesthetic (**`colour`**, **`fill`**, **`size`**, etc.) + type (**`manual`**, **`continuous`**, **`datetime`**, etc.)

```
g + scale_colour_manual(name = "Type", values = c("green", "purple", "yellow"))
```

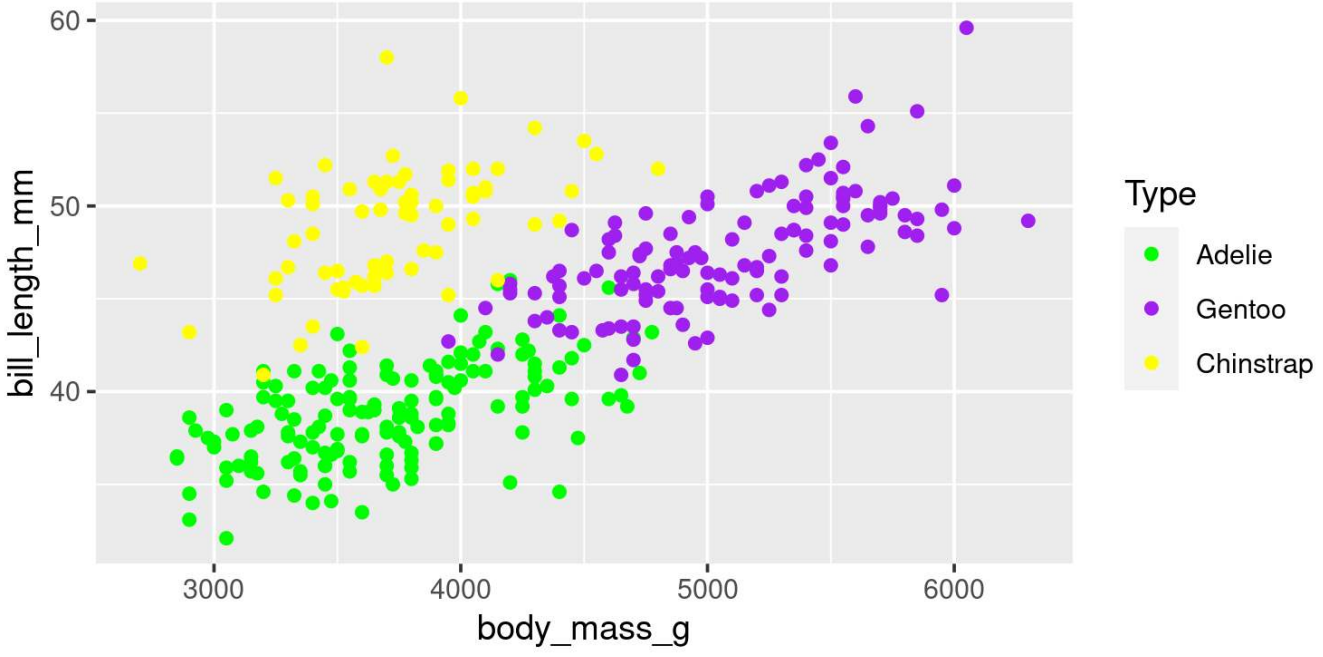


Customizing: Aesthetics

Using scales

Or be very explicit:

```
g + scale_colour_manual(name = "Type", na.value = "black", values = c("Adelie" = "green",
                                                                      "Gentoo" = "purple",
                                                                      "Chinstrap" = "yellow"))
```

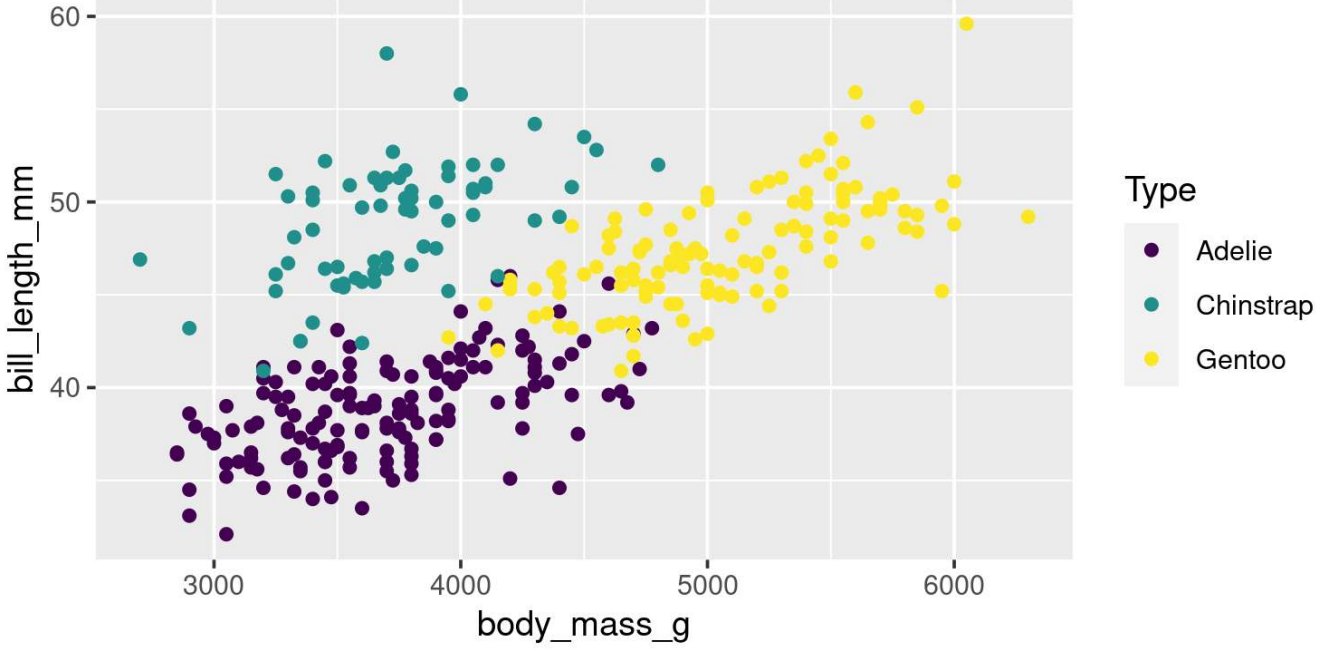


Customizing: Aesthetics

For colours, consider colour-blind-friendly scale

`viridis_d` for "discrete" data

```
ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm, colour = species)) +
  geom_point() +
  scale_colour_viridis_d(name = "Type")
```

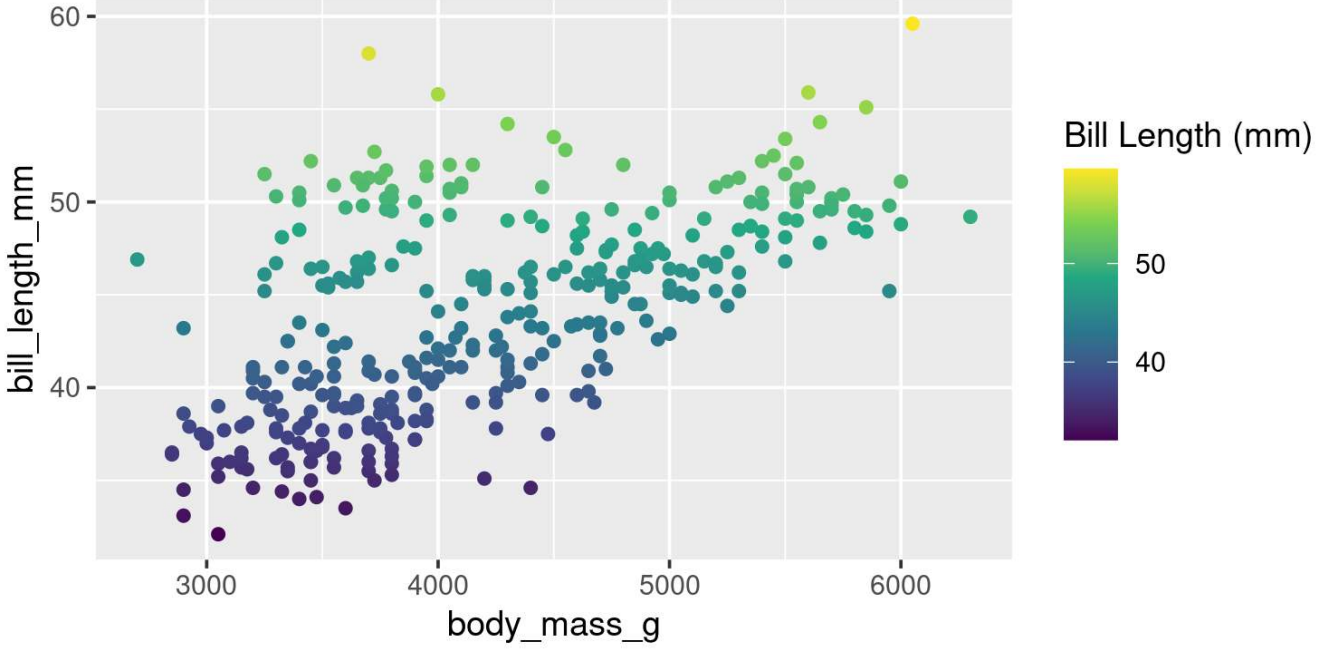


Customizing: Aesthetics

For colours, consider colour-blind-friendly scale

`viridis_c` for "continuous" data

```
ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm, colour = bill_length_mm)) +
  geom_point() +
  scale_colour_viridis_c(name = "Bill Length (mm)")
```

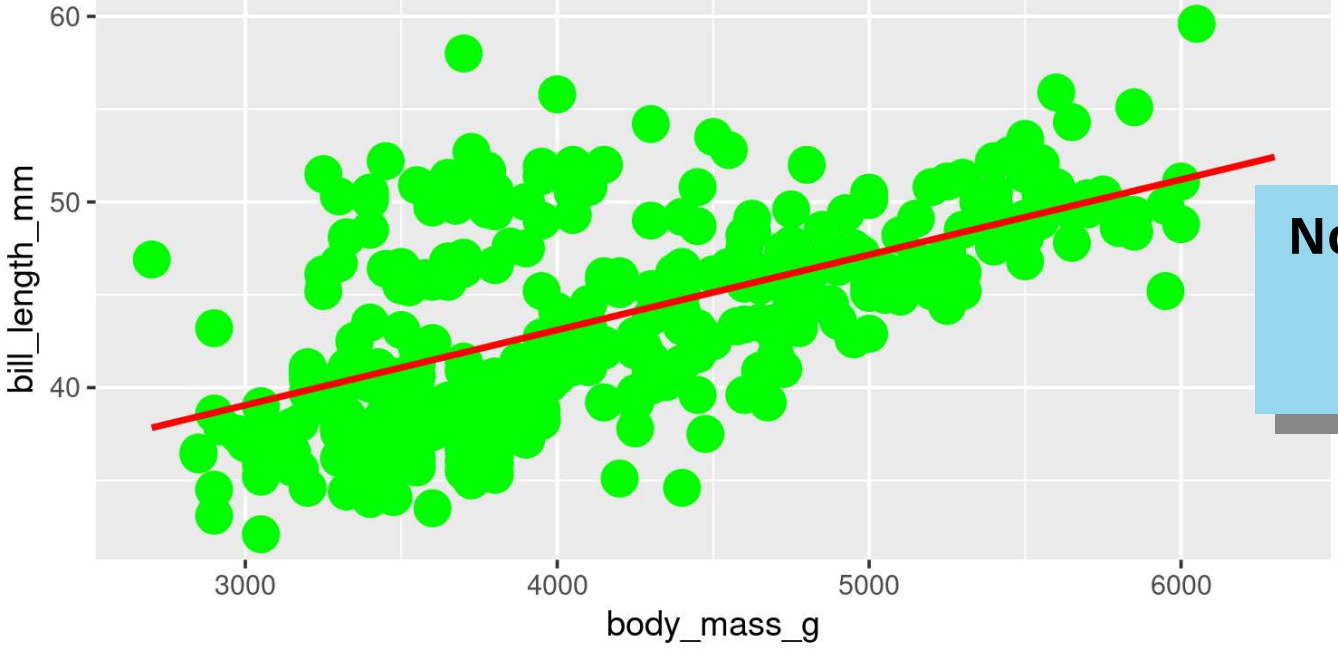


Customizing: Aesthetics

Forcing

Remove the association between a variable and an aesthetic

```
ggplot(data = penguins, aes(x = body_mass_g, y = bill_length_mm, colour = sex)) +
  geom_point(colour = "green", size = 5) +
  stat_smooth(method = "lm", se = FALSE, colour = "red")
```

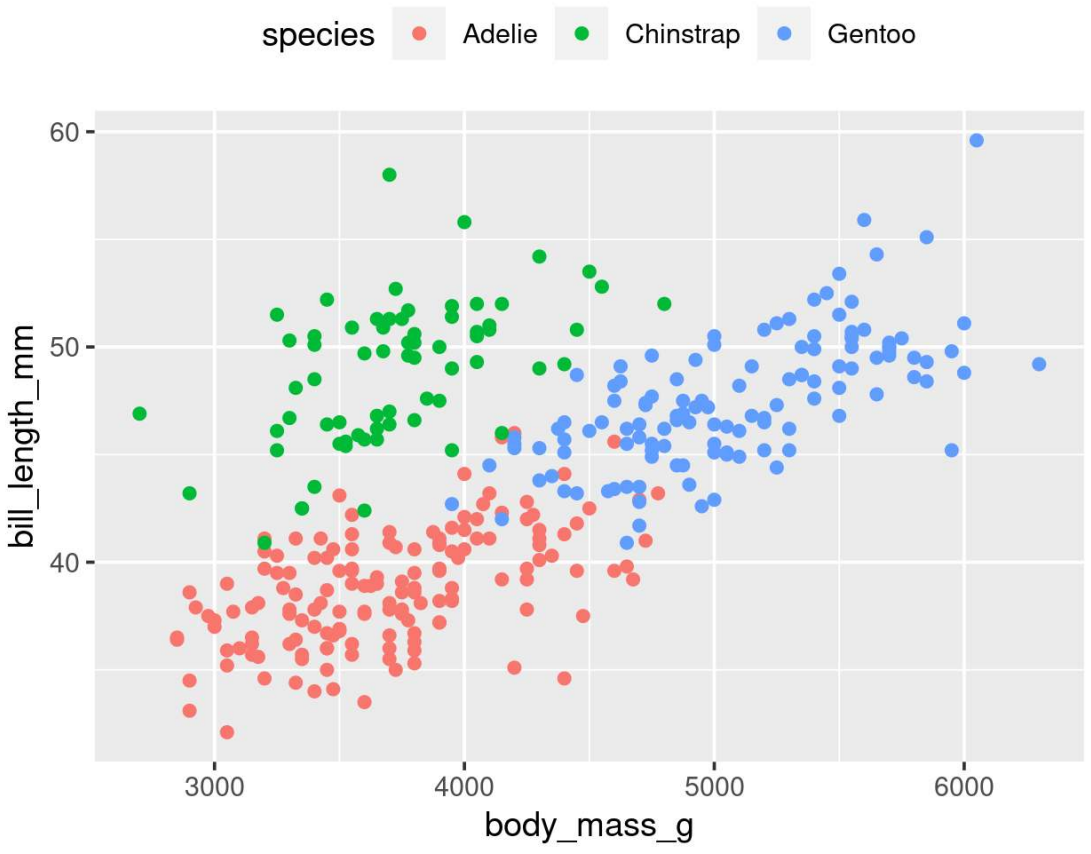


Note: When forcing, aesthetic is not inside **aes()**

Customizing: Legends placement

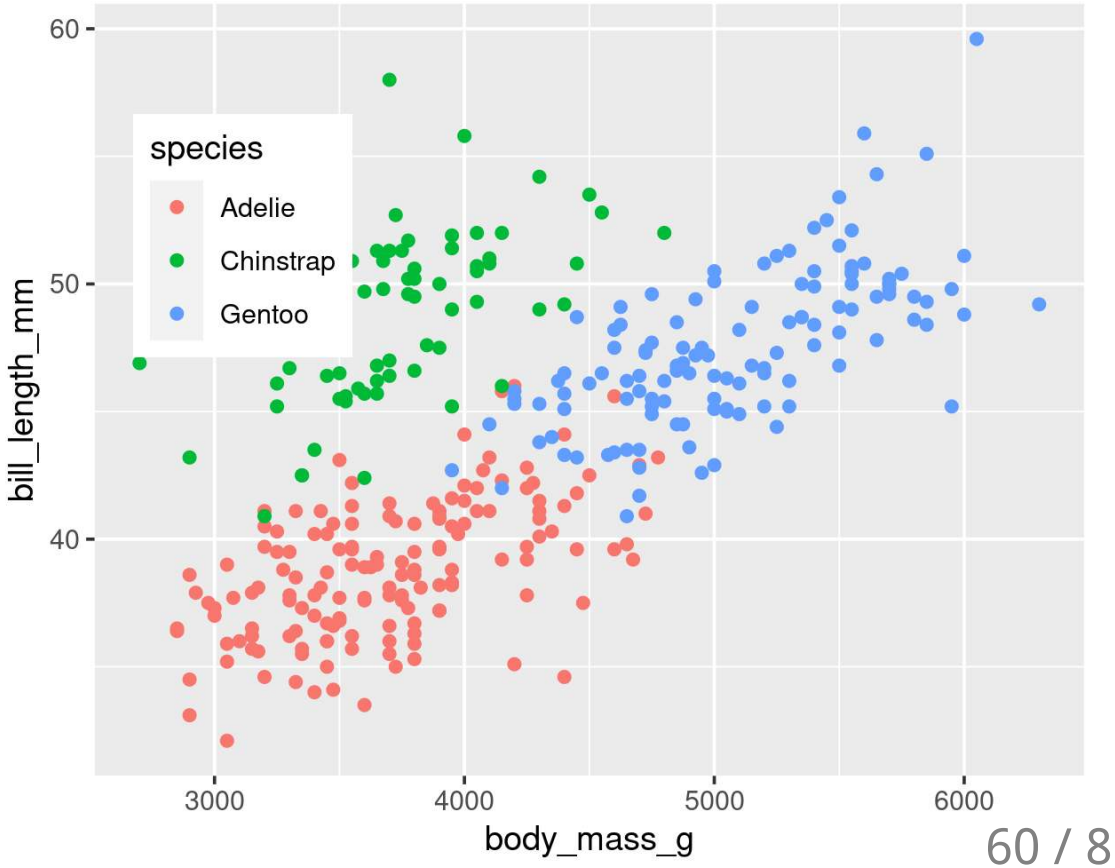
At the: top, bottom, left, right

```
g + theme(legend.position = "top")
```



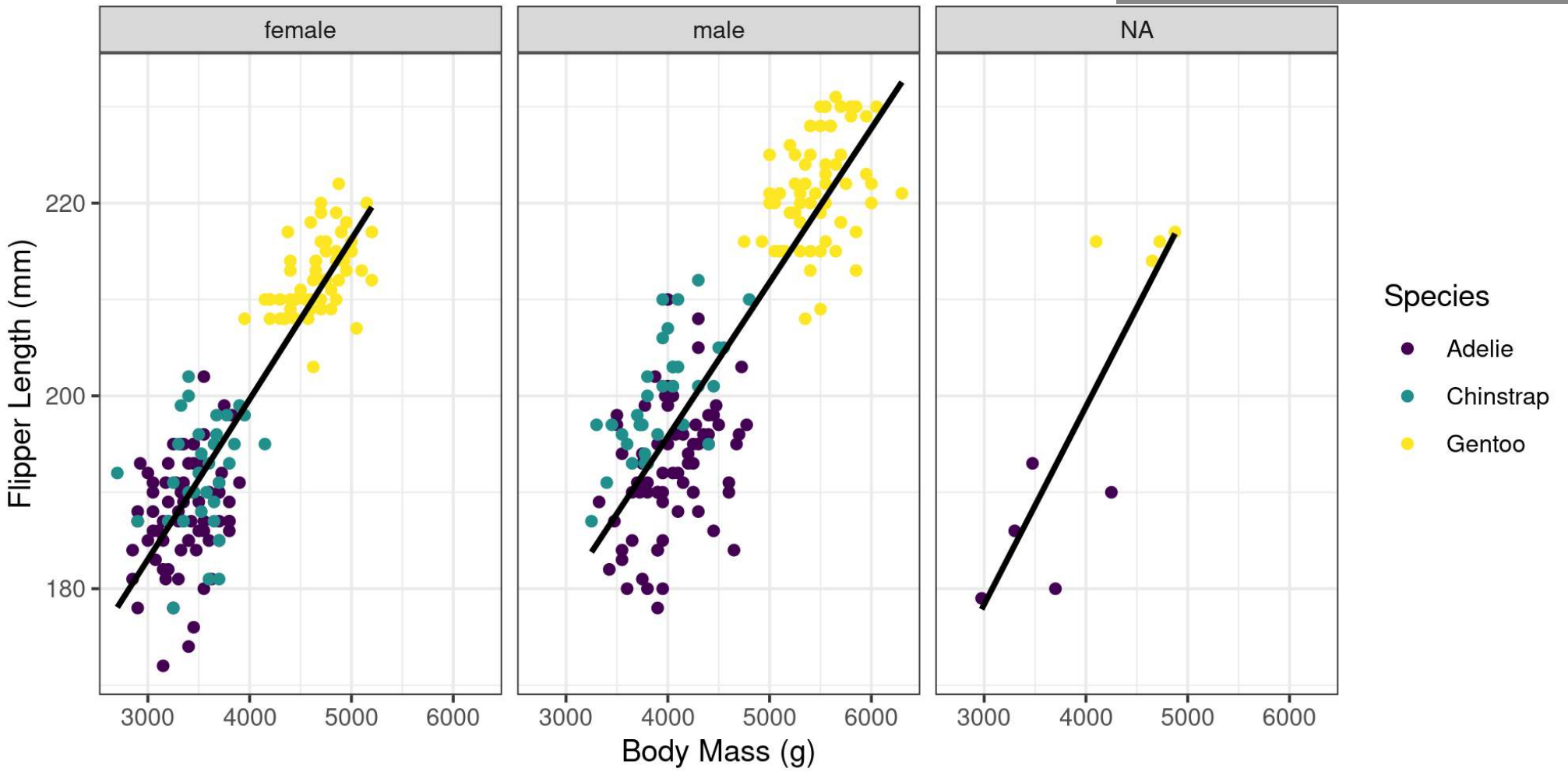
Exactly here

```
g + theme(legend.position = c(0.15, 0.7))
```



Your Turn: Create this plot

Extra Challenge
Create a plot of your own data



Side note: Order of operations

Order of operations

Remember...

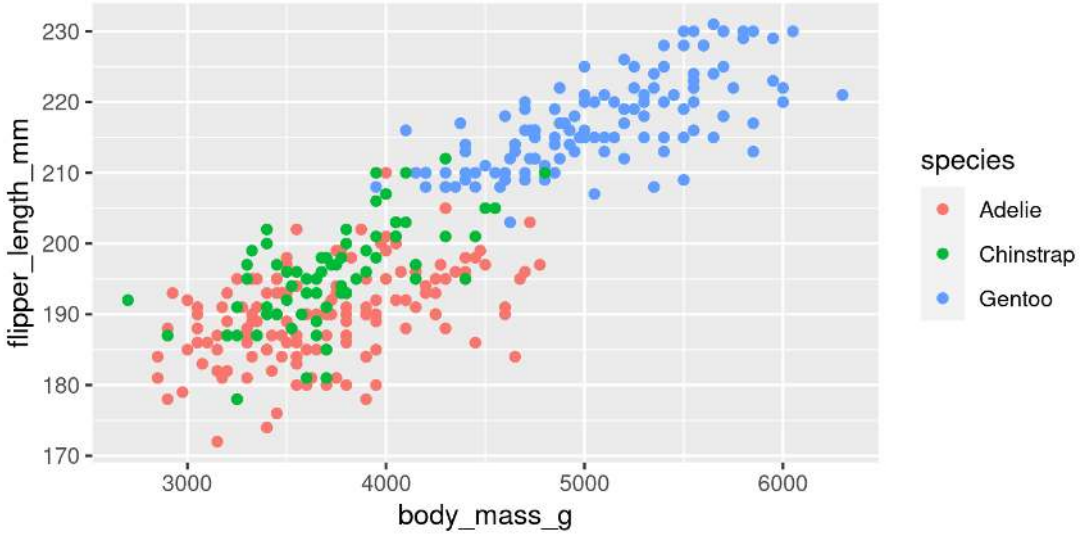
- `ggplot()` is the default line (all options passed down)
- The other lines are *added* with the `+` (options only apply to this line)

Order of operations

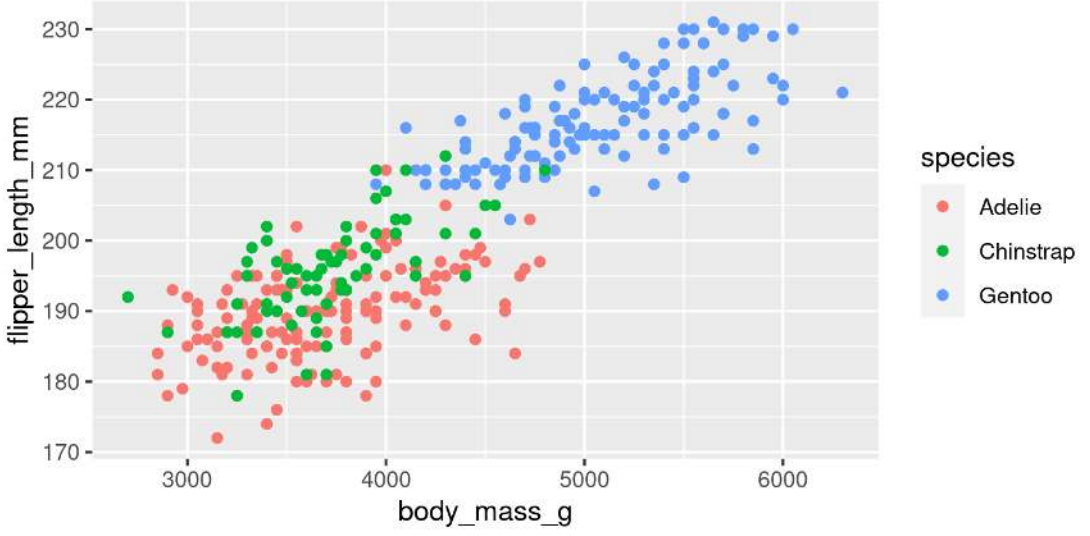
Where to put the `aes()`?

Sometimes it doesn't matter...

```
ggplot(penguins, aes(x = body_mass_g,
                     y = flipper_length_mm,
                     colour = species)) +
  geom_point()
```




```
ggplot(penguins, aes(x = body_mass_g,
                     y = flipper_length_mm)) +
  geom_point(aes(colour = species))
```

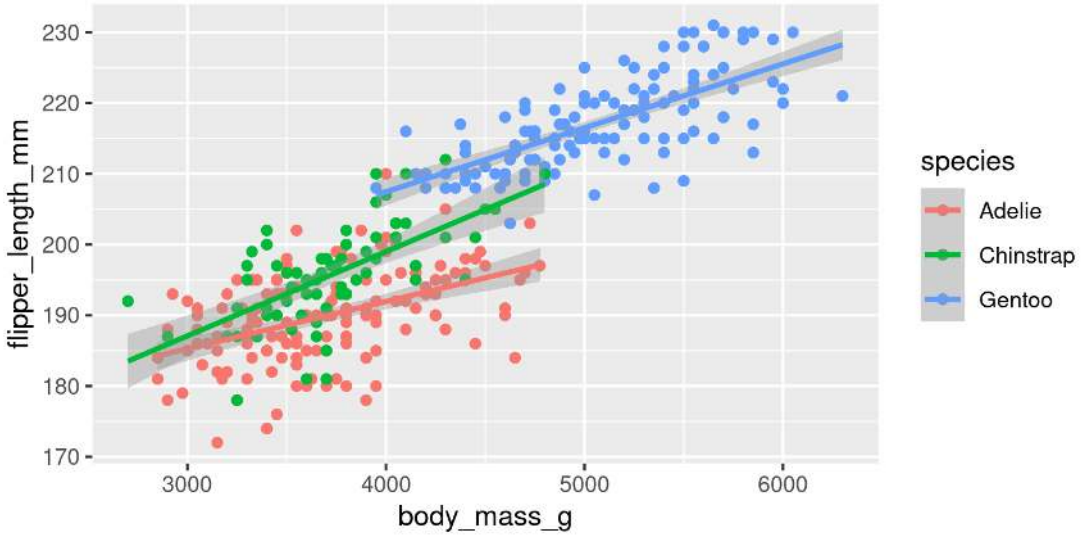


Order of operations

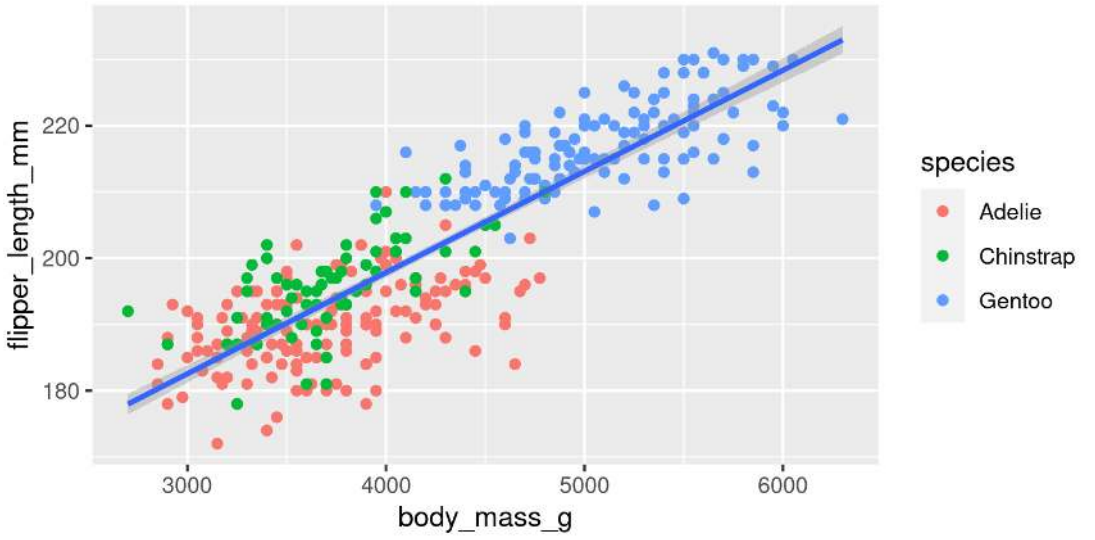
Where to put the aes()?

Sometimes it DOES matter...

```
ggplot(penguins, aes(x = body_mass_g,
                     y = flipper_length_mm,
                     colour = species)) +
  geom_point() +
  stat_smooth(method = "lm")
```




```
ggplot(penguins, aes(x = body_mass_g,
                     y = flipper_length_mm)) +
  geom_point(aes(colour = species)) +
  stat_smooth(method = "lm")
```

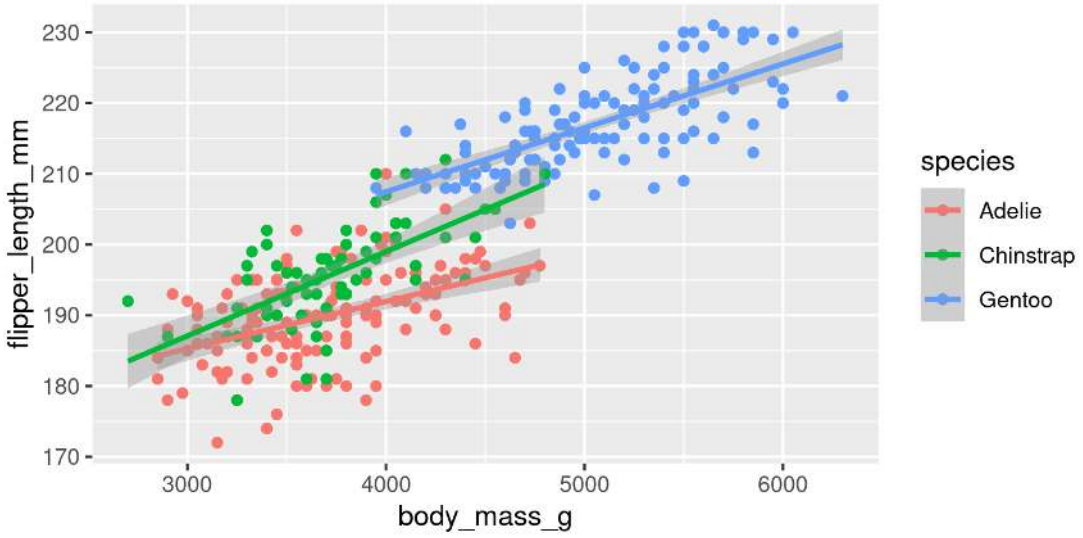


Order of operations

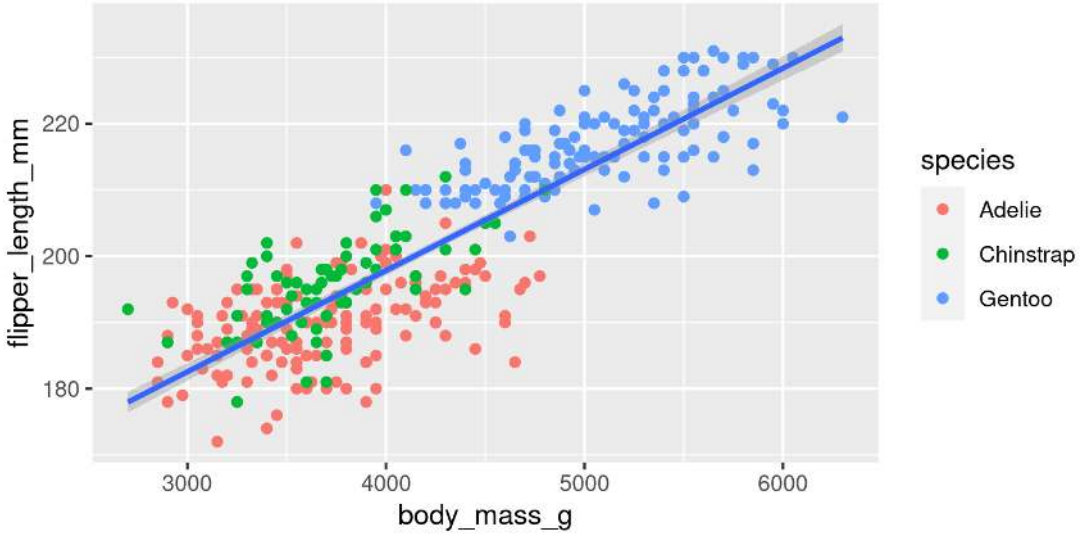
Where to put the `aes()`?

Applies to ALL lines in the ggplot including **`stat_smooth`**

```
                    y = flipper_length_mm,
                    colour = species)) +
  geom_point() +
  stat_smooth(method = "lm")
```




```
ggplot(penguins, aes(x = body_mass_g,
                     y = flipper_length_mm)) +
  geom_point(aes(colour = species)) +
  stat_smooth(method = "lm")
```

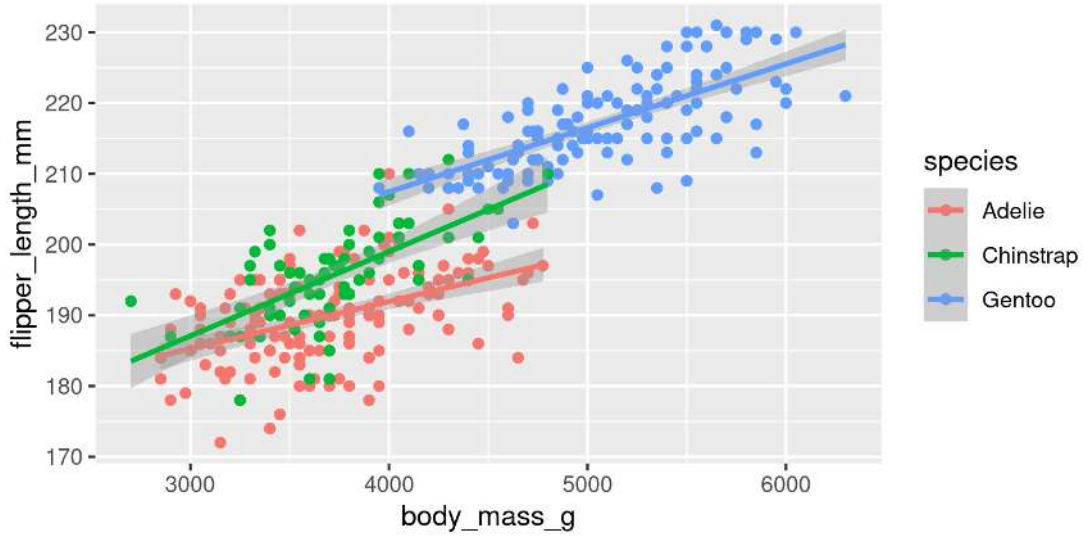


Order of operations

Where to put the `aes()`?

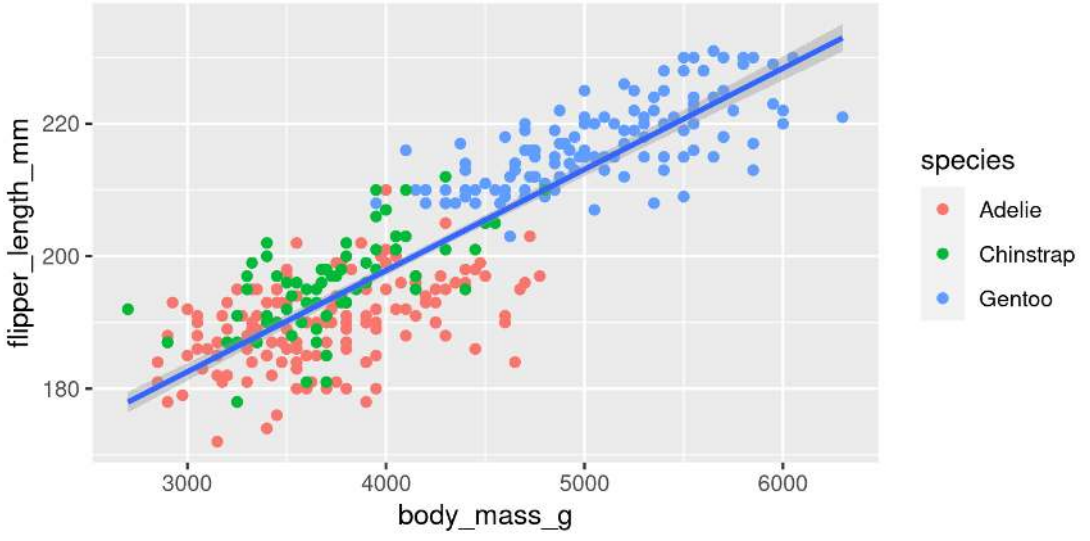
Applies to ALL lines in the ggplot including **`stat_smooth`**

```
                    y = flipper_length_mm,
                    colour = species)) +
  geom_point() +
  stat_smooth(method = "lm")
```



Applies to only the geom_point in the ggplot not **`stat_smooth`**

```
                        y = flipper_length_mm)) +
  geom_point(aes(colour = species)) +
  stat_smooth(method = "lm")
```

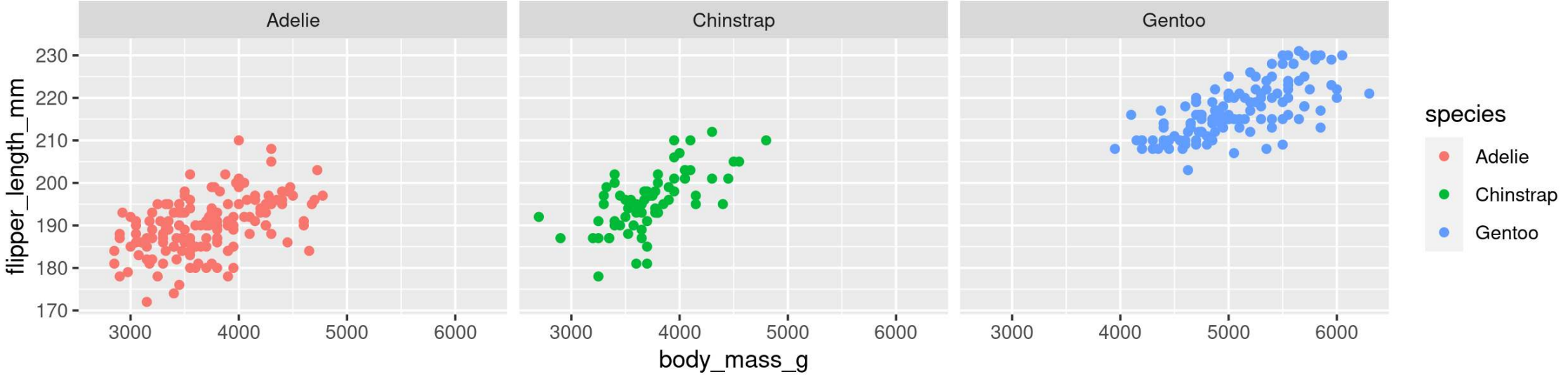


Annotating plots

Annotating

Plot to be annotated: Let's add sample sizes

```
ggplot(data = penguins, aes(x = body_mass_g, y = flipper_length_mm, colour = species)) +
  geom_point() +
  facet_grid(~ species)
```



Annotating

Create data to use in our annotations

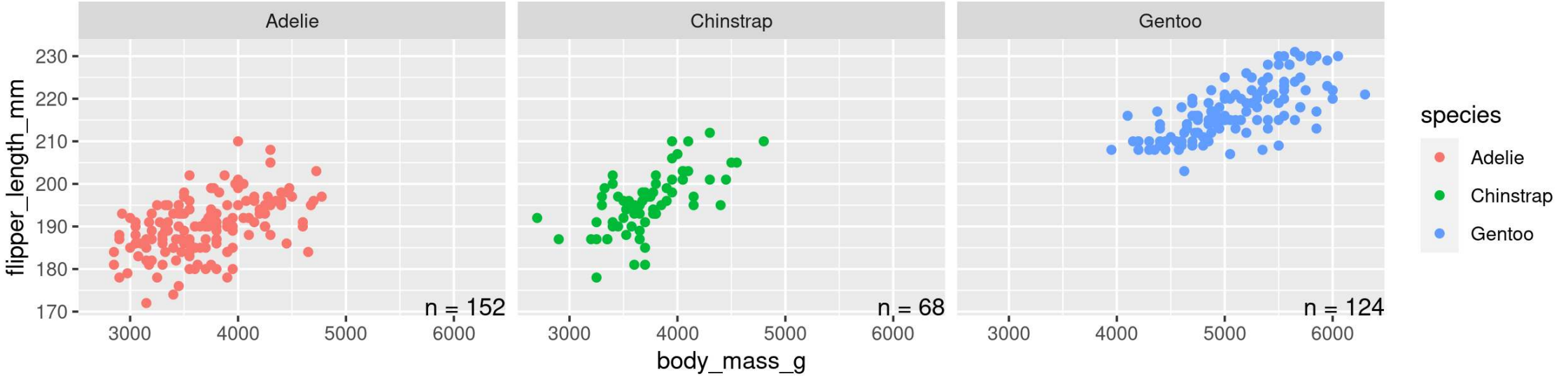
(`mutate()` is covered tomorrow!)

```
library(tidyverse)
n <- count(penguins, species)
n <- mutate(n, text = paste0("n = ", n))
n
```

```
## # A tibble: 3 × 3
##   species       n text
##   <fct>     <int> <chr>
## 1 Adelie      152 n = 152
## 2 Chinstrap    68 n = 68
## 3 Gentoo      124 n = 124
```

Annotating

```
ggplot(data = penguins, aes(x = body_mass_g, y = flipper_length_mm, colour = species)) +
  geom_point() +
  facet_grid(~ species) +
  geom_text(data = n,                    # Use 'n' data set
            x = +Inf, y = -Inf,          # Location relative to plot (right, bottom)
            aes(label = text),           # Map 'text' to label
            hjust = 1, vjust = 0,        # Adjust horizontal and vertical placement
            colour = "black")            # black text
```



Combining plots

Combining plots with `patchwork`

Setup

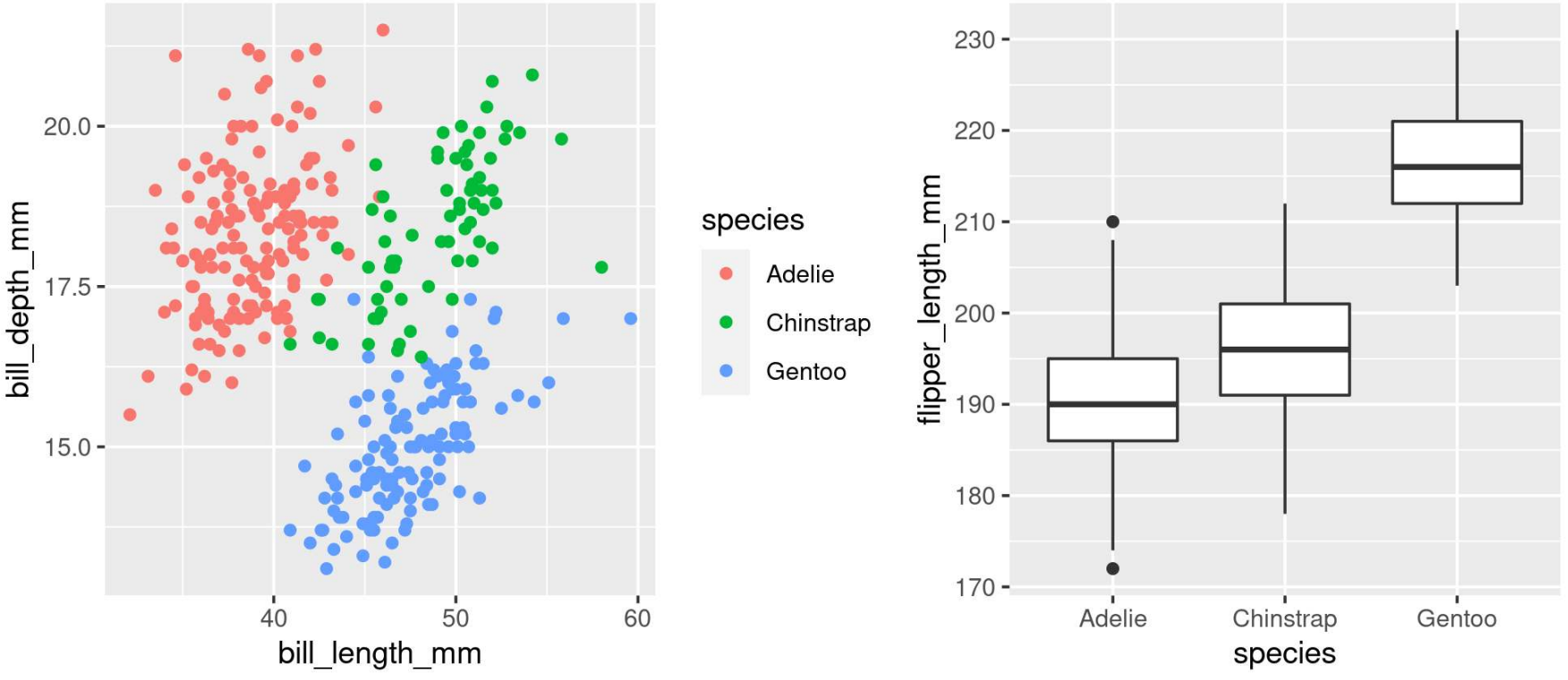
- Load **`patchwork`**
- Create a couple of different plots

```
library(patchwork)

g1 <- ggplot(data = penguins, aes(x = bill_length_mm, y = bill_depth_mm, colour = species)) +
  geom_point()

g2 <- ggplot(data = penguins, aes(x = species, y = flipper_length_mm)) +
  geom_boxplot()

g3 <- ggplot(data = penguins, aes(x = flipper_length_mm, y = body_mass_g, colour = species)) +
  geom_point()
```

Combining plots with `patchwork`

Side-by-Side 2 plots

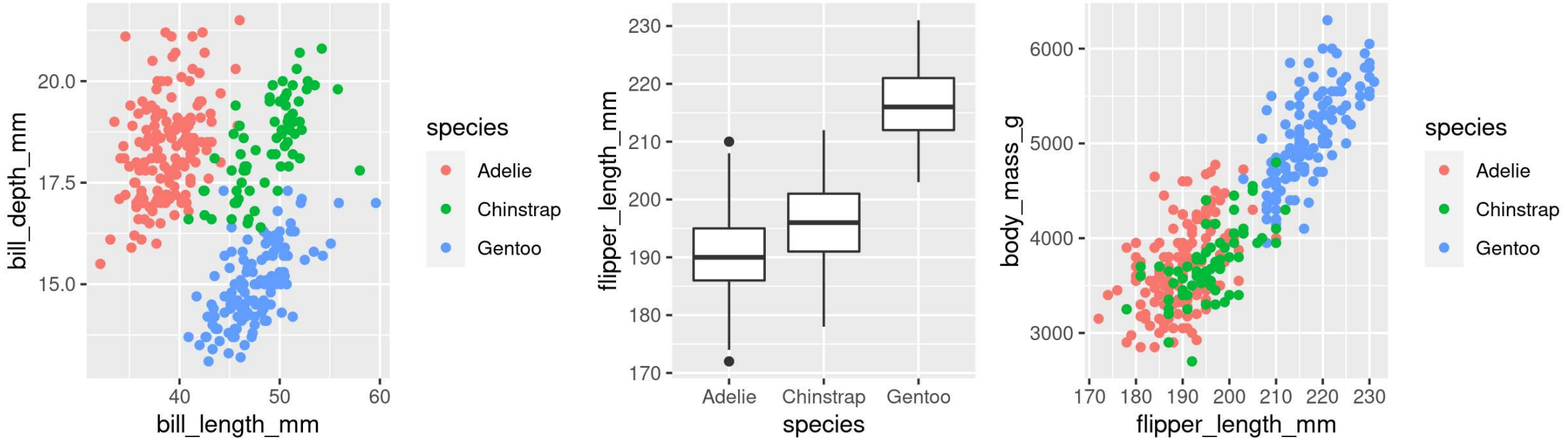
```
g1 + g2
```



Combining plots with `patchwork`

Side-by-Side 3 plots

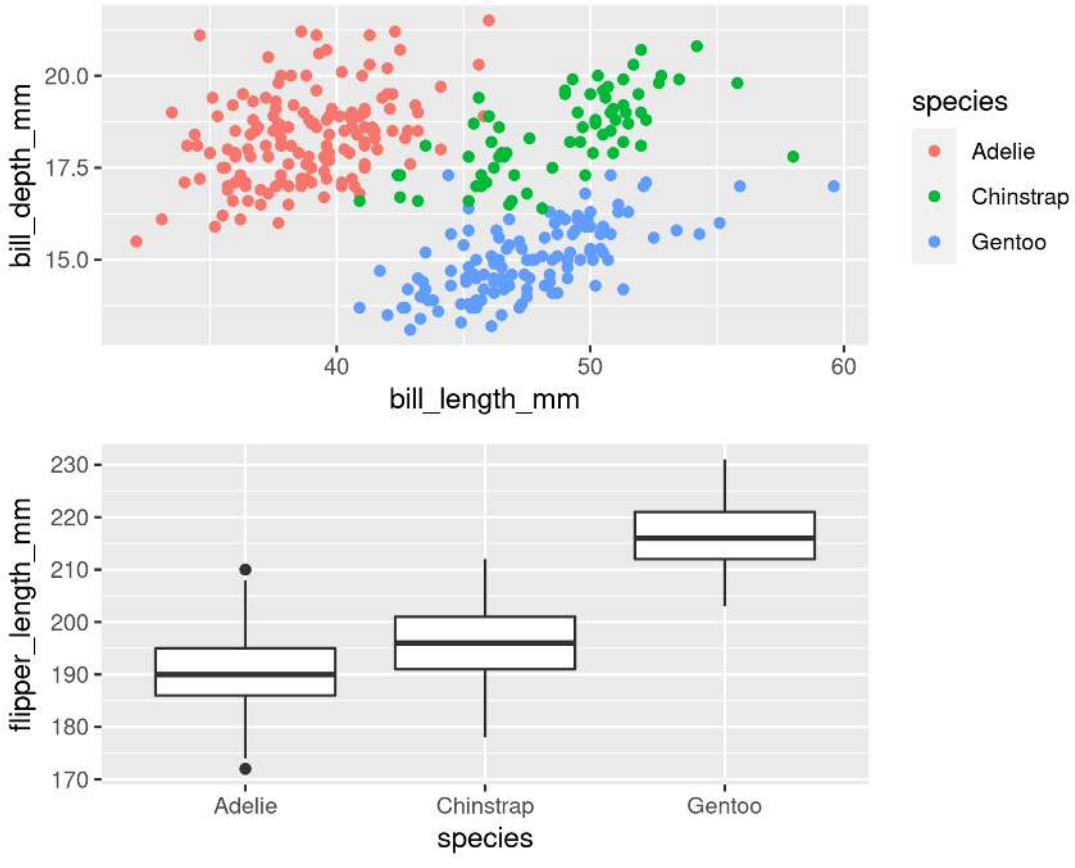
```
g1 + g2 + g3
```



Combining plots with `patchwork`

Stacked 2 plots

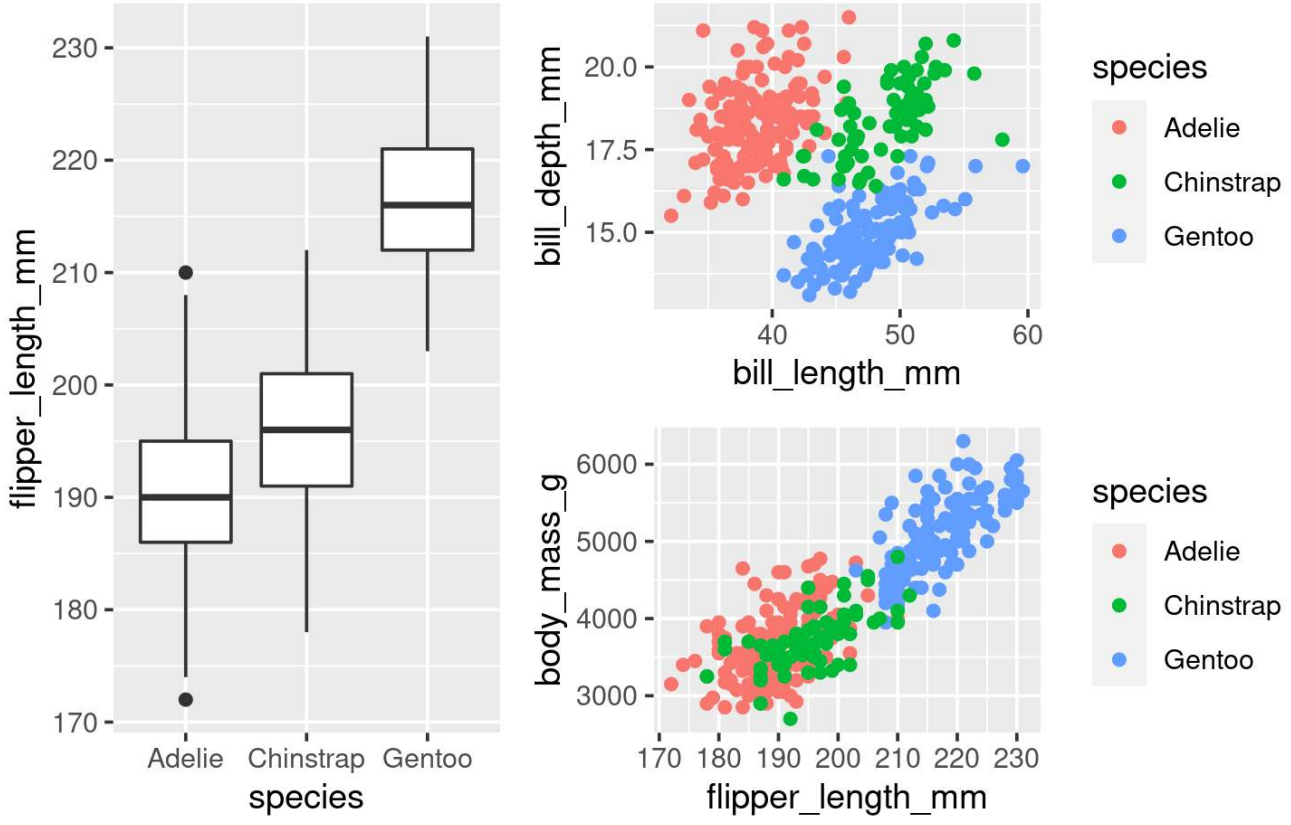
```
g1 / g2
```



Combining plots with `patchwork`

More complex arrangements

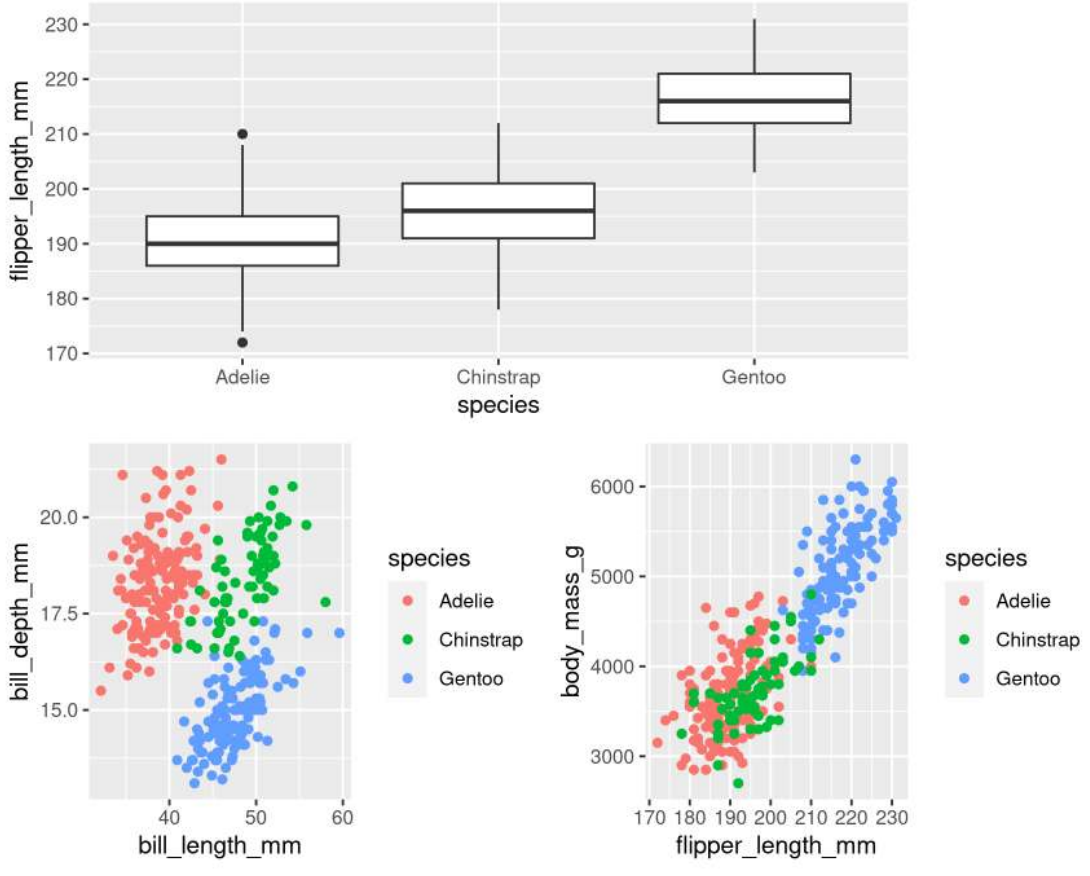
```
g2 + (g1 / g3)
```



Combining plots with `patchwork`

More complex arrangements

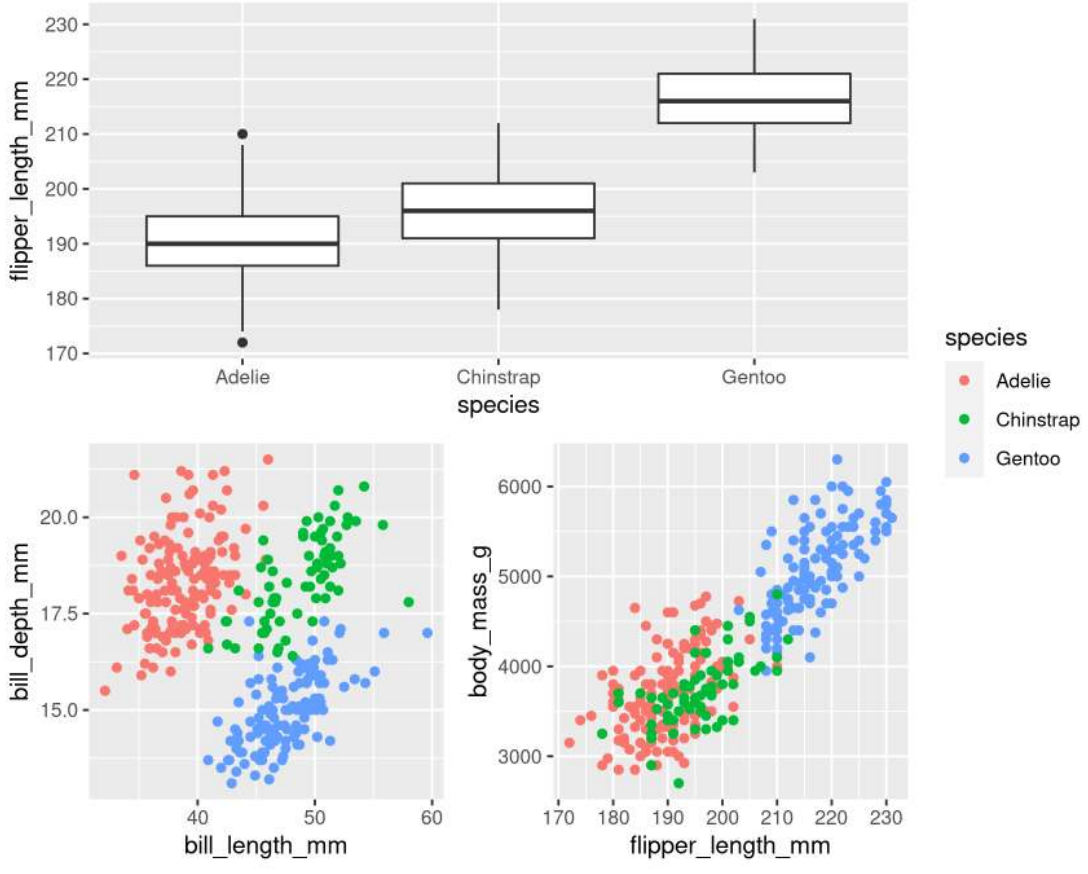
```
g2 / (g1 + g3)
```



Combining plots with `patchwork`

"collect" common legends

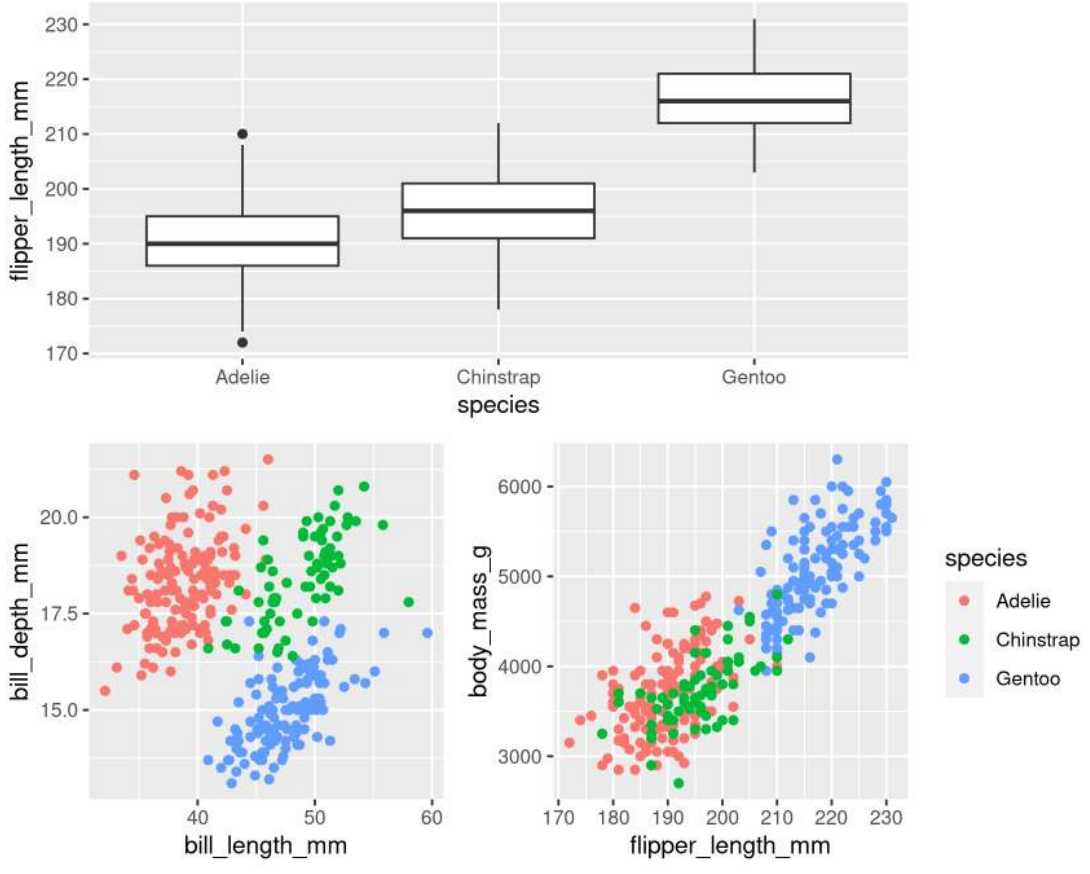
```
g2 / (g1 + g3) + plot_layout(guides = "collect")
```



Combining plots with patchwork

"collect" common legends

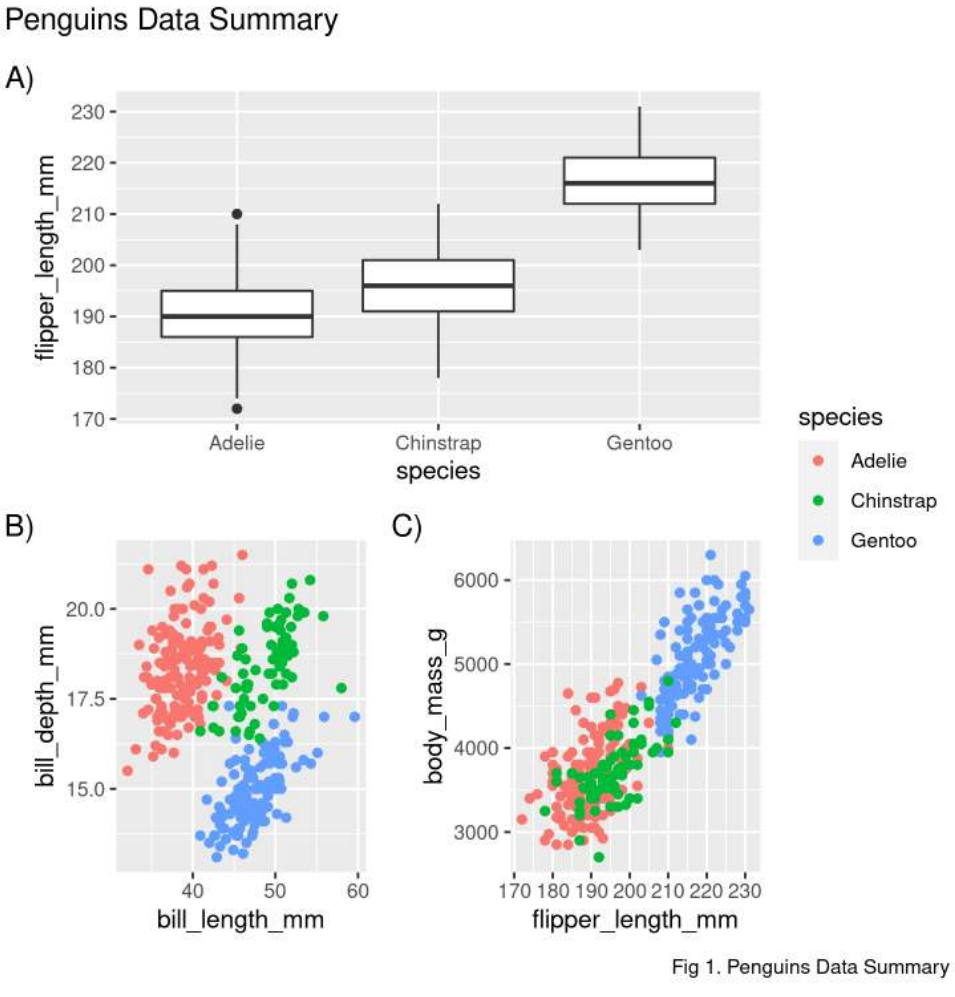
```
g2 / (g1 + g3 + plot_layout(guides = "collect"))
```



Combining plots with patchwork

Annotate

```
g2 / (g1 + g3) +
  plot_layout(guides = "collect") +
  plot_annotation(title = "Penguins Data Summary",
                  caption = "Fig 1. Penguins Data Summary",
                  tag_levels = "A",
                  tag_suffix = ")")
```

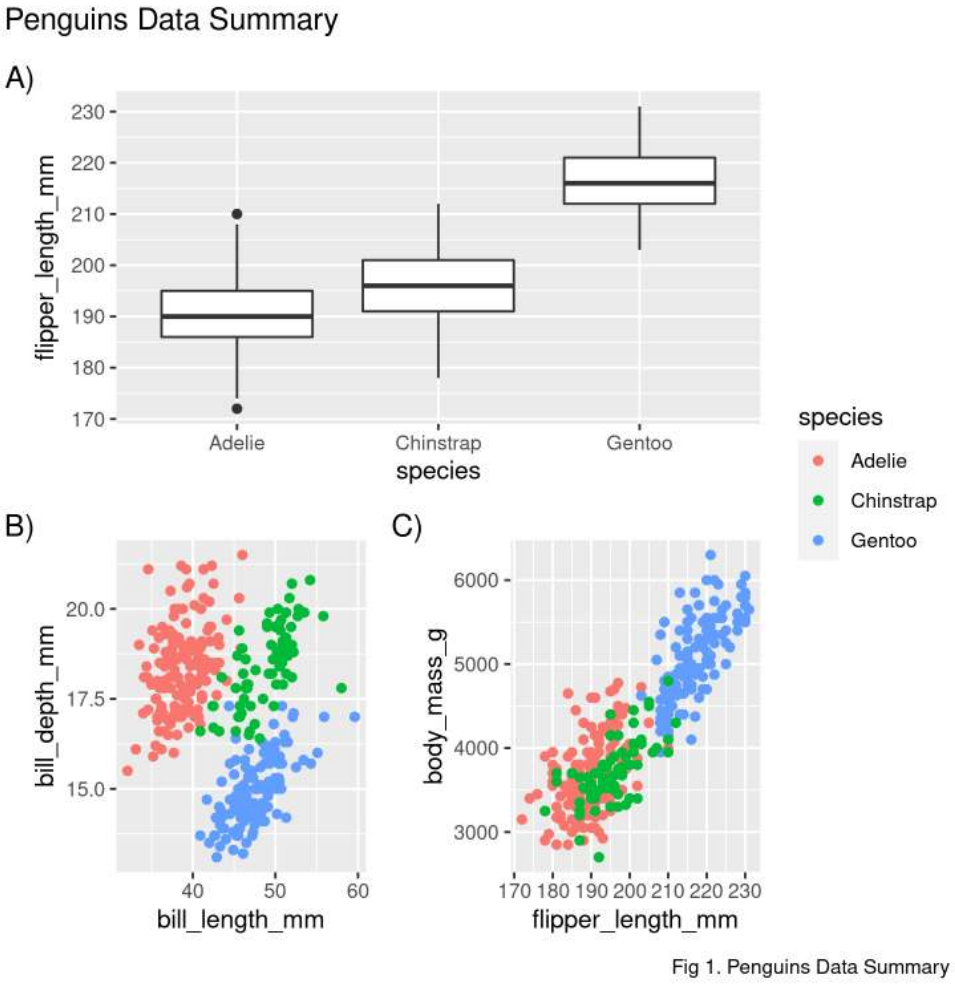


Combining plots with patchwork

Annotate

```
g2 / (g1 + g3) +
  plot_layout(guides = "collect") +
  plot_annotation(title = "Penguins Data Summary",
                  caption = "Fig 1. Penguins Data Summary",
                  tag_levels = "A",
                  tag_suffix = ")")
```

Your Turn: Combine any 3 figures



Saving plots

Saving plots

RStudio Export

Demo

Saving plots

RStudio Export

Demo

`ggsave()`

```
g <- ggplot(penguins, aes(x = sex, y = bill_length_mm, fill = year)) +
  geom_boxplot()

ggsave(filename = "penguins_mass.png", plot = g)
```

```
## Saving 6 x 3.9 in image
```

Saving plots

Publication quality plots

- Many publications require 'lossless' (pdf, svg, eps, ps) or high quality formats (tiff, png)
- Specific sizes corresponding to columns widths
- Minimum resolutions

```
g <- ggplot(penguins, aes(x = sex, y = body_mass_g)) +
  geom_boxplot() +
  labs(x = "Sex", y = "Body Mass (g)") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

ggsave(filename = "penguins_mass.pdf", plot = g, dpi = 300,
       height = 80, width = 129, units = "mm")
```

Wrapping up: Common mistakes

- The **package** is `ggplot2`, the function is just `ggplot()`
- Did you remember to put the `+` at the **end** of the line?
- Order matters! If you're using custom `theme()`'s, make sure you put these lines **after** bundled themes like `theme_bw()`, or they will be overwritten
- Variables like 'year' are treated as continuous, but are really categories
  - Wrap them in `factor()`, i.e. `ggplot(data = penguins, aes(x = factor(year), y = body_mass_g))`

Wrapping up: Common mistakes

I get an error regarding an object that can't be found or aethetic length?

You are probably trying to plot two different datasets, and you make references to variables in the `ggplot()` call that don't exist in one of the datasets:

```
n <- count(penguins, island)

ggplot(data = penguins, aes(x = flipper_length_mm, y = bill_length_mm, colour = species)) +
  geom_point() +
  facet_wrap(~ island) +
  geom_text(data = n, aes(label = n),
            x = -Inf, y = +Inf, hjust = 0, vjust = 1)
```

```
## Error: Aesthetics must be either length 1 or the same as the data (3): colour
```

Wrapping up: Common mistakes

I get an error regarding an object that can't be found or aethetic length?

Either move the aesthetic...

```
ggplot(penguins, aes(x = flipper_length_mm, y = bill_length_mm)) +
  geom_point(aes(colour = species)) +
  facet_wrap(~ island) +
  geom_text(data = n, aes(label = n),
            x = -Inf, y = +Inf, hjust = 0, vjust = 1)
```

Or assign it to NULL where it is missing...

```
ggplot(penguins, aes(x = flipper_length_mm, y = bill_length_mm, colour = species)) +
  geom_point() +
  facet_wrap(~ island) +
  geom_text(data = n, aes(label = n, colour = NULL),
            x = -Inf, y = +Inf, hjust = 0, vjust = 1)
```

Wrapping up: Further reading (all Free!)

- RStudio > Help > Cheatsheets > Data Visualization with ggplot2
- **ggplot2** book v3 by Hadley Wickham, Danielle Navarro, and Thomas Lin Pedersen
- R Graphics Cookbook by Winston Chang
- R for Data Science by Hadley Wickham and Garret Grolemund
  - Chapter on Data Visualization
- Data Visualization: A practical introduction by Kieran Healy
- **patchwork** website
- **gridExtra** Vignette: Arranging Multiple Grobs
  - Alternative to patchwork