Data visualization with ggplot2

Just show me the data!

```r
head(my_data, 10)
```

```
## # A tibble: 10 × 2
##        x     y
##    <dbl> <dbl>
##  1  55.4  97.2
##  2  51.5  96.0
##  3  46.2  94.5
##  4  42.8  91.4
##  5  40.8  88.3
##  6  38.7  84.9
##  7  35.6  79.9
##  8  33.1  77.6
##  9  29.0  74.5
## 10  26.2  71.4
```

```r
mean(my_data$x)
```

```
## [1] 54.26327
```

Seems reasonable

```r
mean(my_data$y)
```

```
## [1] 47.83225
```

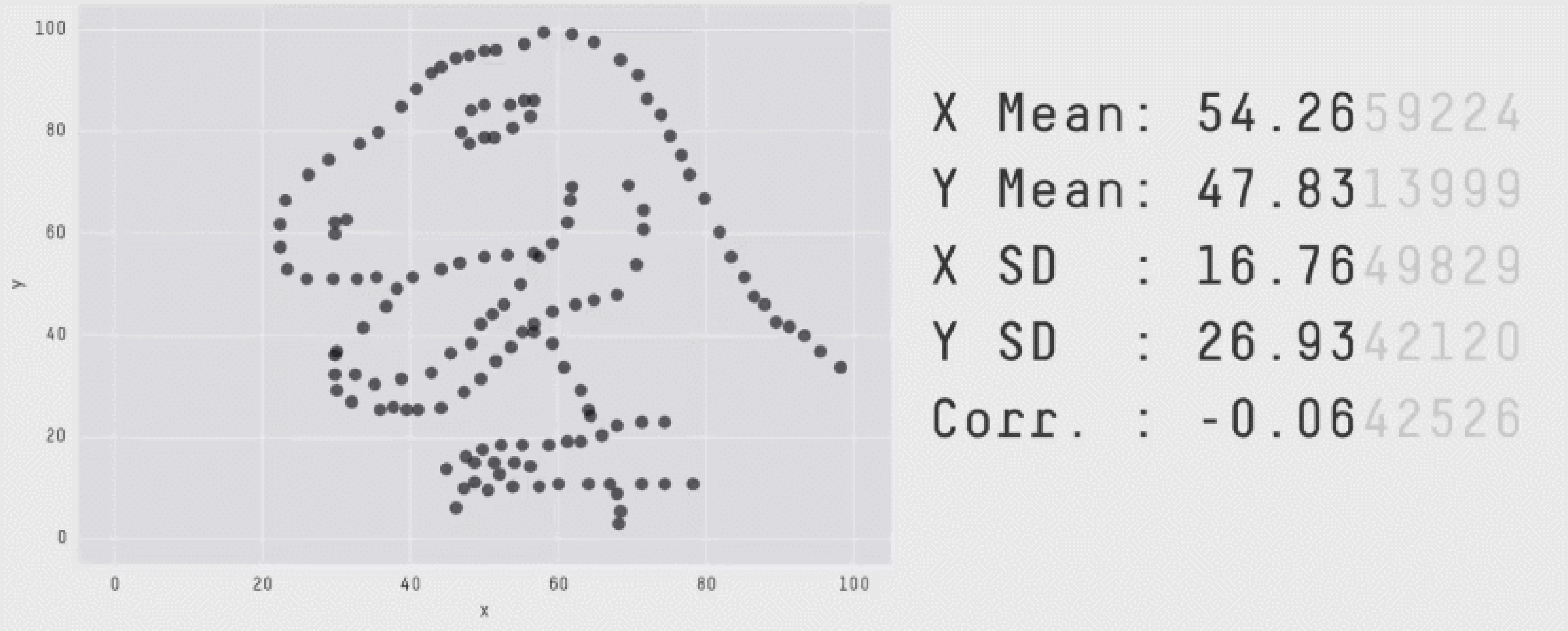
Seems reasonable

```r
cor(my_data$x, my_data$y)
```

```
## [1] -0.06447185
```

No correlation

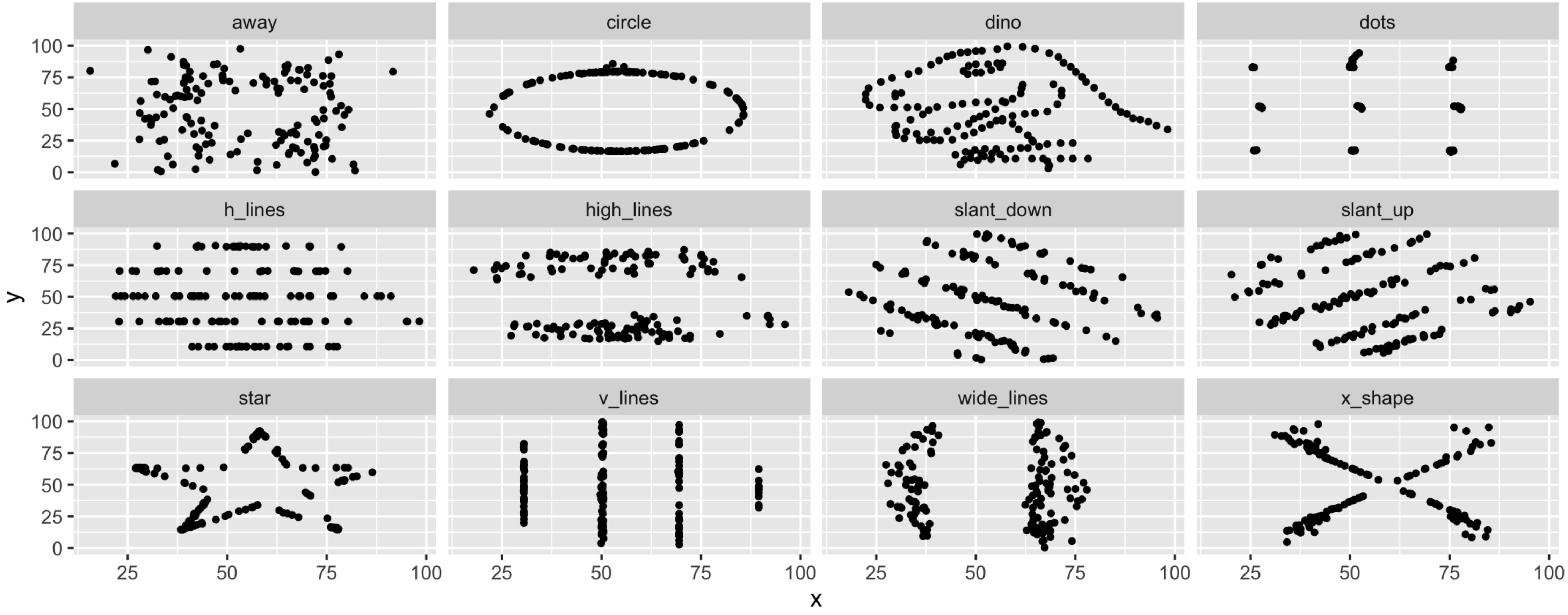
oh no



The Datasaurus Dozen

Raw data is not enough

Each of these has the same mean, standard deviation, variance, and correlation



BMI and daily steps

> Consider the following (alternative, not null) hypotheses:
>
> 1. There is a difference in the mean number of steps between women and men
> 2. The correlation coefficient between steps and BMI is negative for women
> 3. The correlation coefficient between steps and BMI is positive for men
>
> Think about which test to use and calculate the corresponding p-value.
>
> What conclusions can you draw from the data?

```r
library(tidyverse)
bmi_data <- read_csv("data/bmi_data.csv")

head(bmi_data)
```

```
## # A tibble: 6 × 3
##     bmi  steps sex
##   <dbl>  <dbl> <chr>
## 1  27.9   401.  Male
## 2  28.4  6204.  Male
## 3  12.4  8723.  Female
## 4  24.5 11241.  Male
## 5  17.5  5109.  Female
## 6  23.5    73.0 Female
```

```r
t.test(steps ~ sex, data = bmi_data)
```

```
## 
##     Welch Two Sample t-test
## 
## data:  steps by sex
## t = -6.5215, df = 1759.9, p-value = 9.069e-11
## alternative hypothesis: true difference in means
between group Female and group Male is not equal to 0
## 95 percent confidence interval:
##  -1408.8005  -757.3441
## sample estimates:
## mean in group Female   mean in group Male
##             6769.378             7852.450
```

```r
bmi_data %>%
  group_by(sex) %>%
  summarize(correlation = cor(bmi, steps))
```

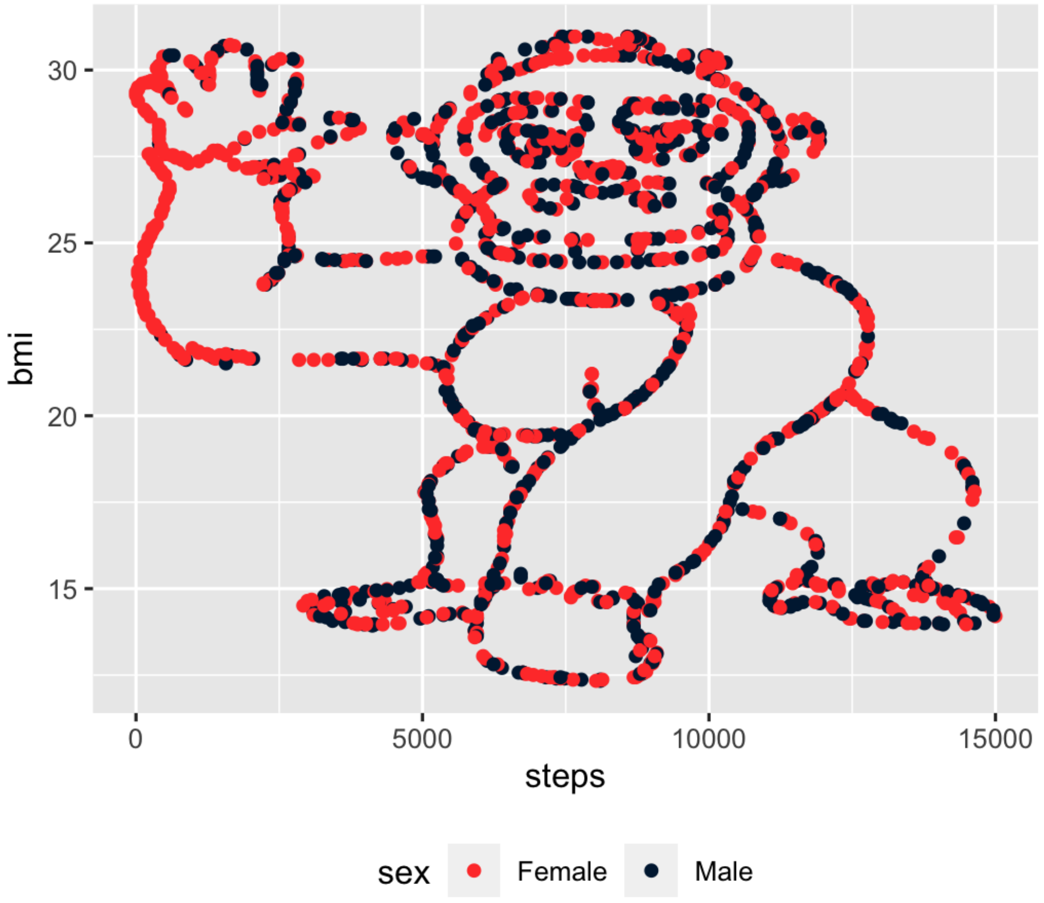
```
## # A tibble: 2 × 2
##   sex    correlation
##   <chr>        <dbl>
## 1 Female      -0.306
## 2 Male        -0.192
```

Raw numbers are not enough!

Examine the data appropriately!

What do you notice?

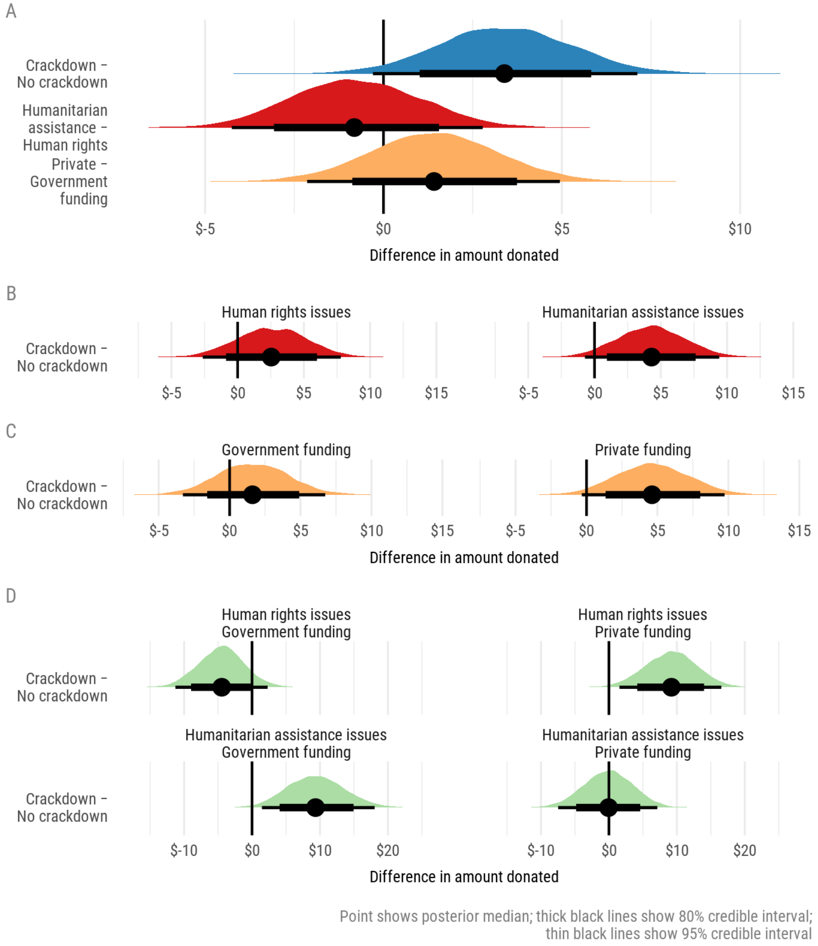
What conclusions can you draw from the data?



Beauty is necessary to see patterns

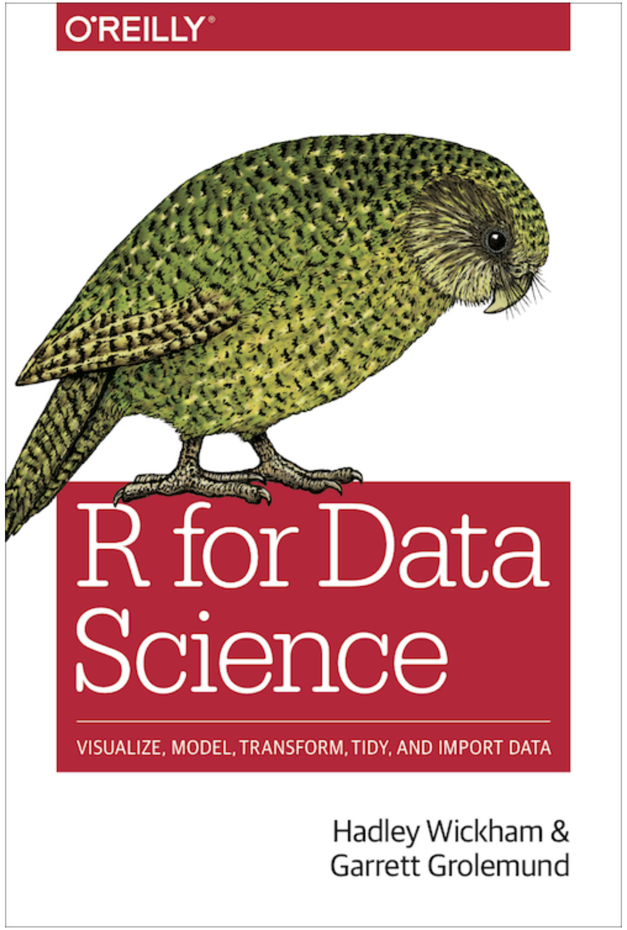
Table 2: Mean values and differences in means for amount donated in "crackdown" (treatment) and "no crackdown" (control) conditions; values represent posterior medians

H_{1b}	$Amount_{Treatment}$	$Amount_{Control}$	Δ	$\%\Delta$	$p(\Delta \neq 0)$
Crackdown − No crackdown	16.34	12.93	3.39	26.3%	0.97
Humanitarian assistance − Human rights	14.06	14.85	-0.82	-5.5%	0.67
Private − Government funding	15.13	13.71	1.42	10.4%	0.79
H_{2b} and H_{3b}	$Amount_{Crackdown}$	$Amount_{No\ crackdown}$	Δ	$\%\Delta$	$p(\Delta \neq 0)$
Human rights issues	17.4	14.86	2.54	17.2%	0.83
Humanitarian assistance issues	15.91	11.68	4.3	36.9%	0.95
Government funding	13.83	12.24	1.61	13.1%	0.74
Private funding	18.95	14.23	4.62	32.4%	0.97
H_{2b} and H_{3b} (nested)	$Amount_{Crackdown}$	$Amount_{No\ crackdown}$	Δ	$\%\Delta$	$p(\Delta \neq 0)$
Human rights issues, Government funding	10.56	15.15	-4.46	-29.5%	0.91
Human rights issues, Private funding	23.76	14.5	9.19	63.8%	0.99
Humanitarian assistance issues, Government funding	21.42	11.89	9.35	77.9%	0.99
Humanitarian assistance issues, Private funding	15.69	15.72	-0.05	-0.3%	0.51



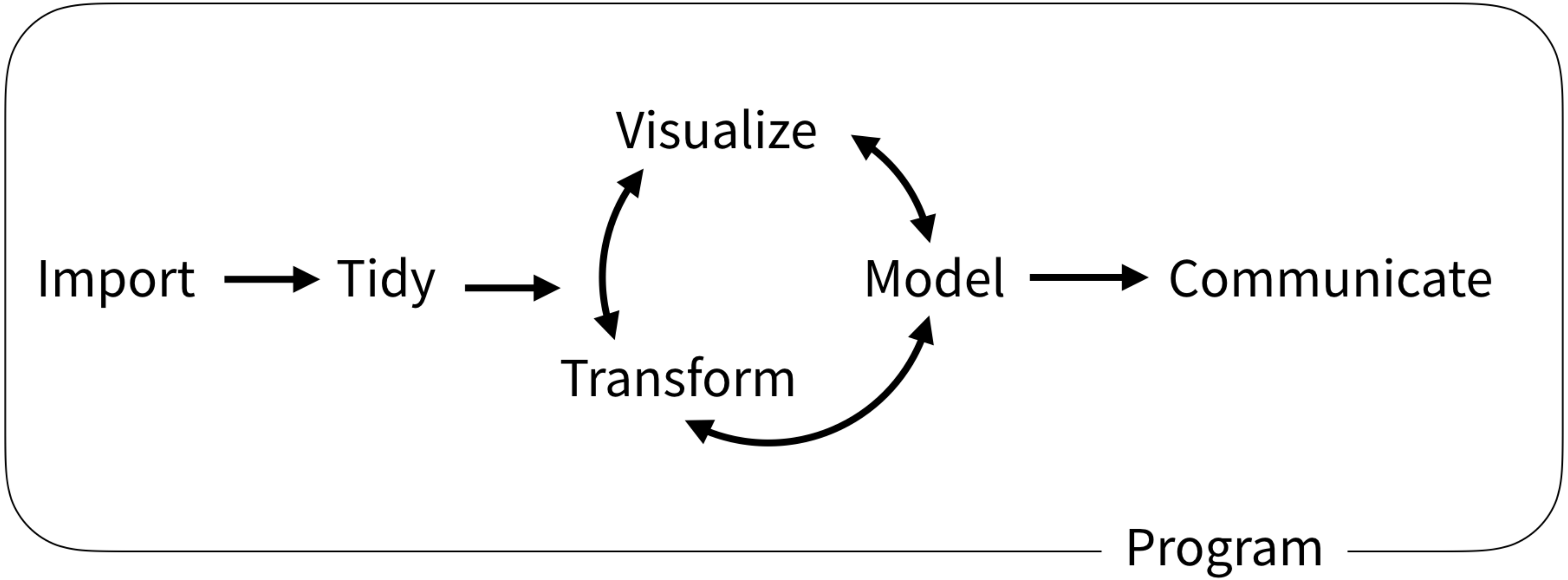
Beauty is necessary for finding truth

Applied data science



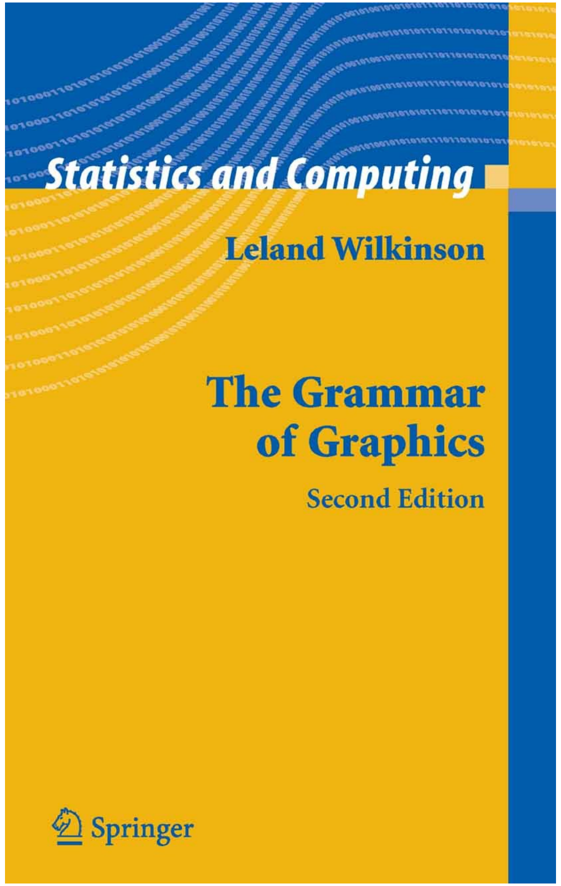
R for Data Science, free online!

Applied data science



The Grammar of Graphics

Mapping data to aesthetics



Aesthetic

Visual property of a graph

Position, shape, color, etc.

Data

A column in a dataset

Your turn #1

Watch this video

andhs.co/rosling

Make a list of all the variables shown in the graph (think about columns in a dataset)

Make a list of how those variables are shown in the graph (think about the graph's aesthetics and geometries)

05:00


0
1
2
3
4
5
6
7
8
Children per woman
TEDx
organized TED event

Mapping data to aesthetics

Data	Aesthetic	Geometry
Wealth (GDP/capita)	Position (x-axis)	Point
Health (Life expectancy)	Position (y-axis)	Point
Continent	Color	Point
Population	Size	Point
Year	Time	Animation

Mapping data to aesthetics

Data	**`aes()`**	**geom**
Wealth (GDP/capita)	`x`	`geom_point()`
Health (Life expectancy)	`y`	`geom_point()`
Continent	`color`	`geom_point()`
Population	`size`	`geom_point()`
Year	`transition`	`transition_time()`

ggplot() template

```
ggplot(data = DATA) +
  GEOM_FUNCTION(mapping = aes(AESTHETIC MAPPINGS))
```

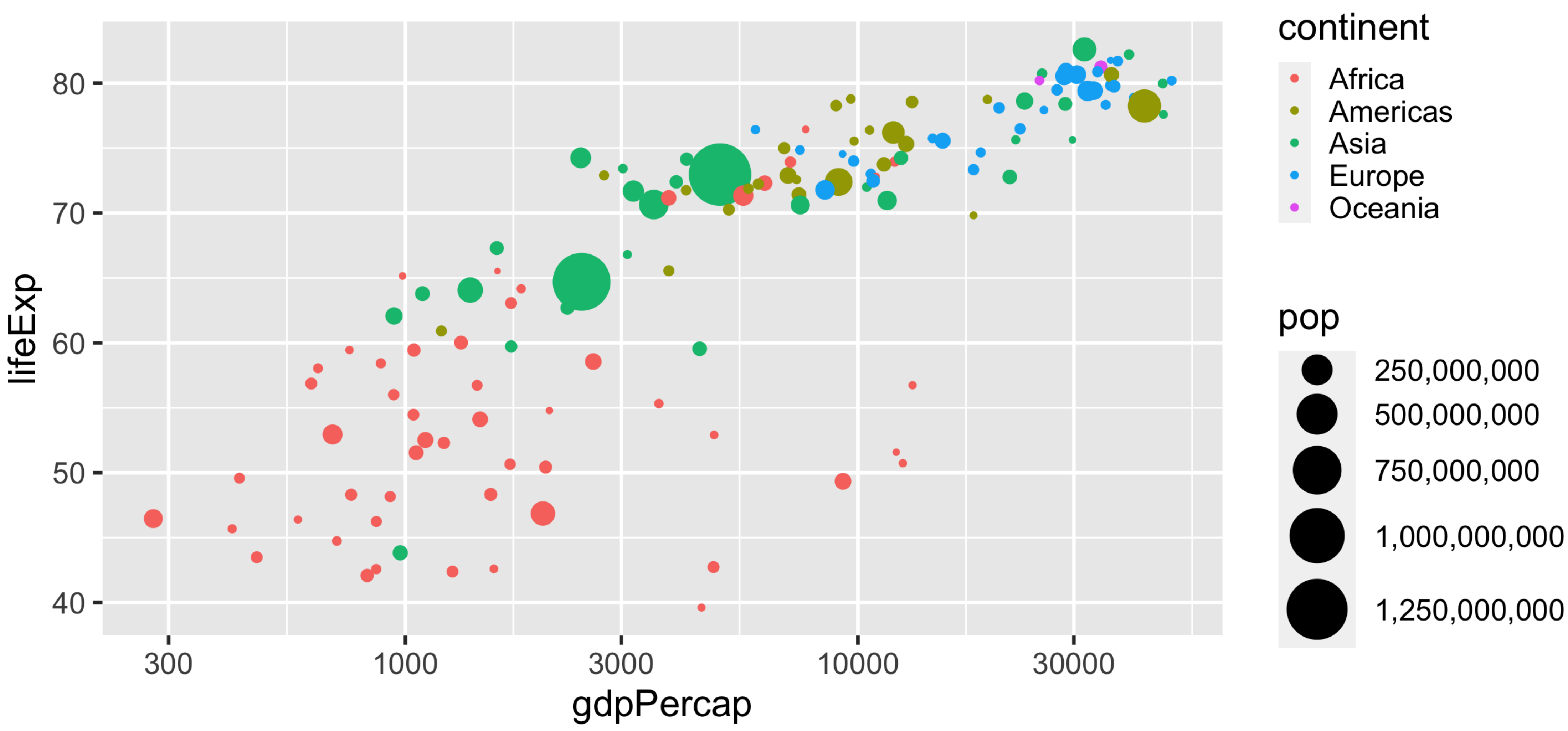
```
ggplot(data = gapminder_2007) +
  geom_point(mapping = aes(x = gdpPercap,
                           y = lifeExp,
                           color = continent,
                           size = pop)))
```

This is a dataset named `gapminder_2007`:

country	continent	gdpPercap	lifeExp	pop
Afghanistan	Asia	974.5803384	43.828	31889923
Albania	Europe	5937.029526	76.423	3600523
…	…	…	…	…

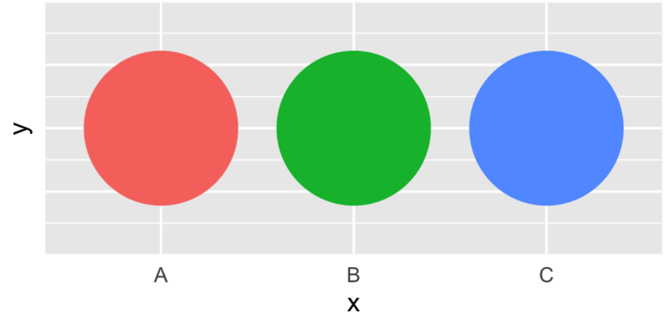
```
ggplot(data = gapminder_2007,
       mapping = aes(x = gdpPercap, y = lifeExp,
                     color = continent, size = pop)) +
  geom_point() +
  scale_x_log10()
```

Health and wealth

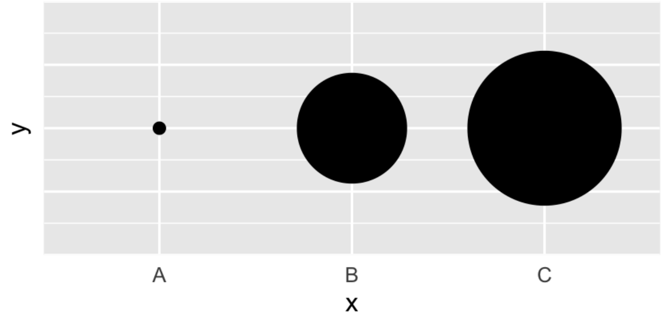


Aesthetics

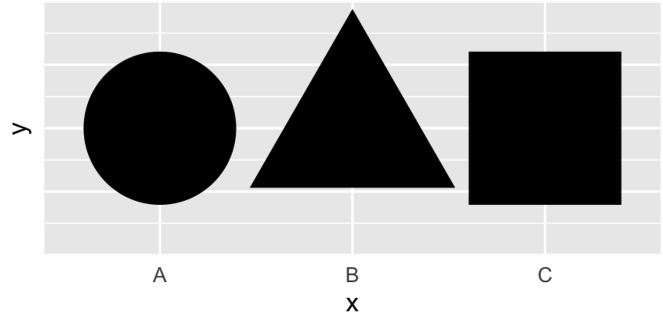
`color` (discrete)



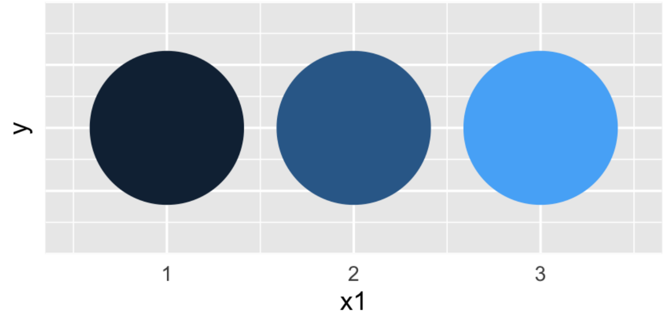
`size`



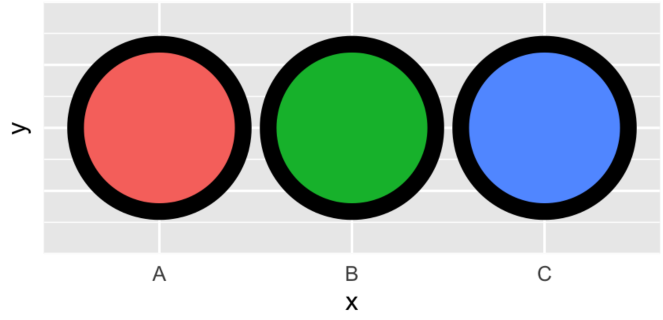
`shape`



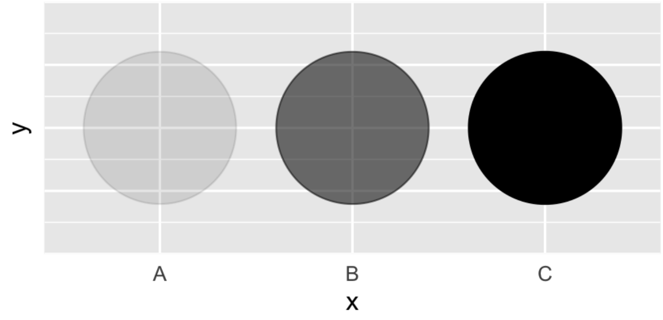
`color` (continuous)

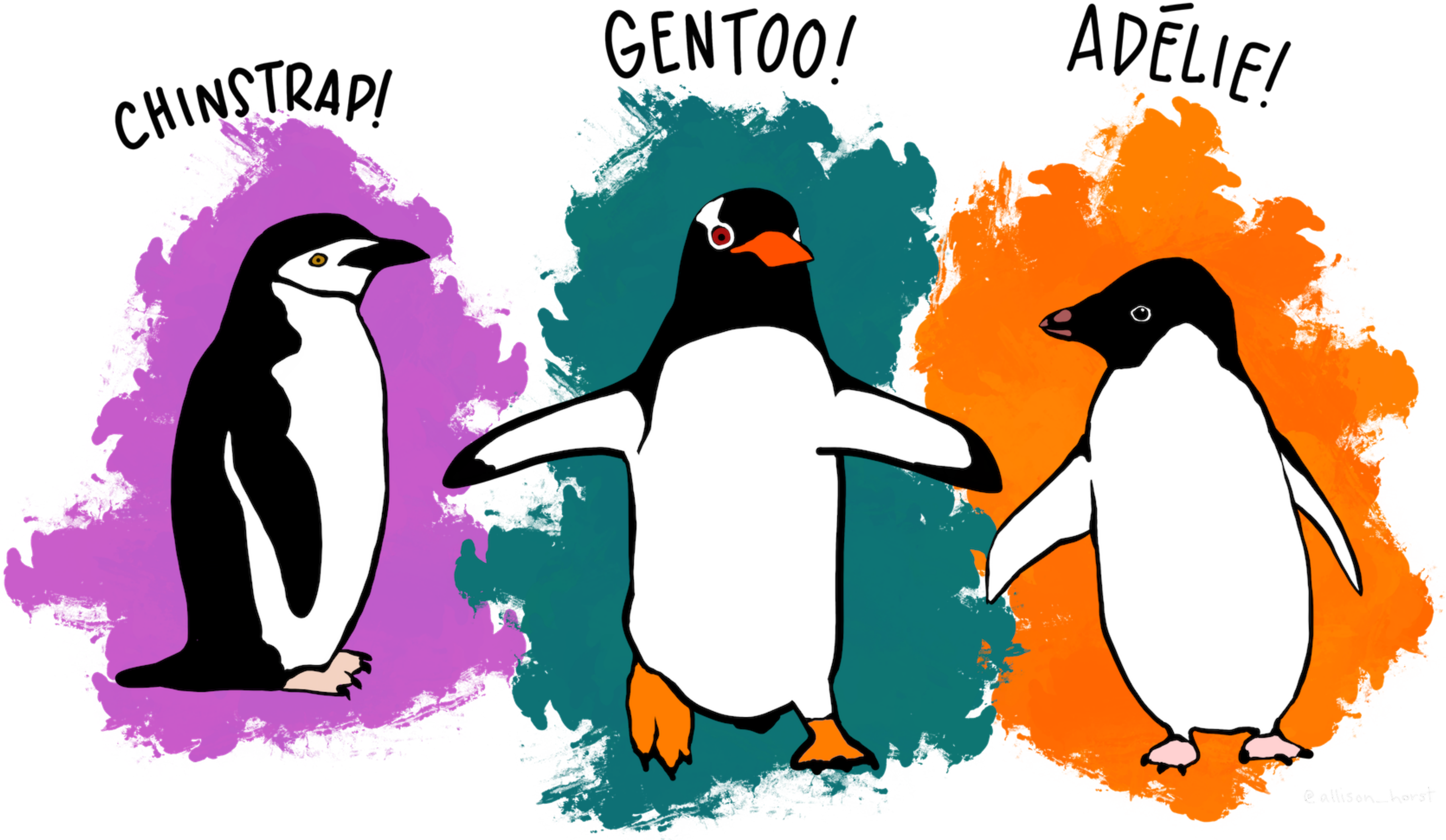


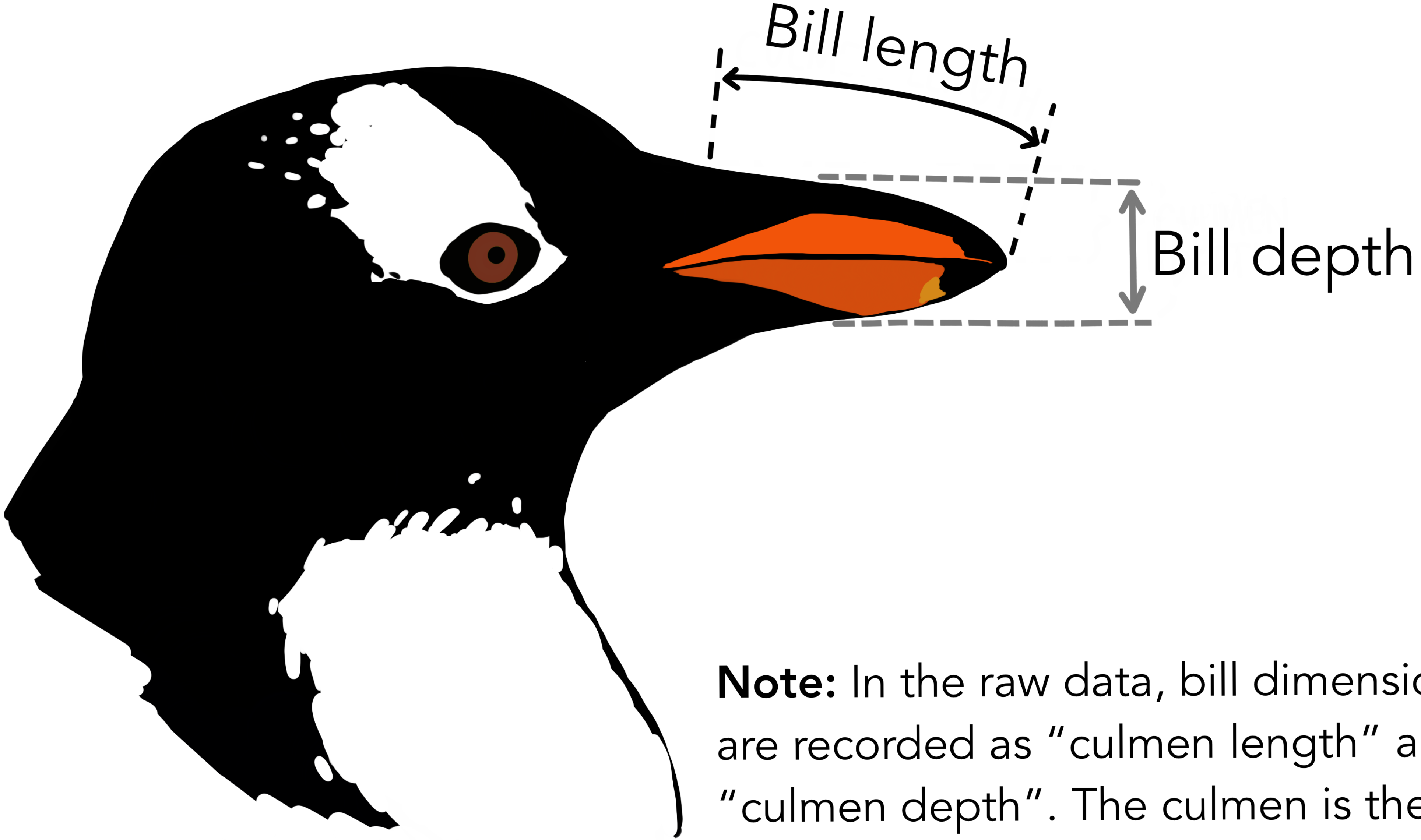
`fill`



`alpha`

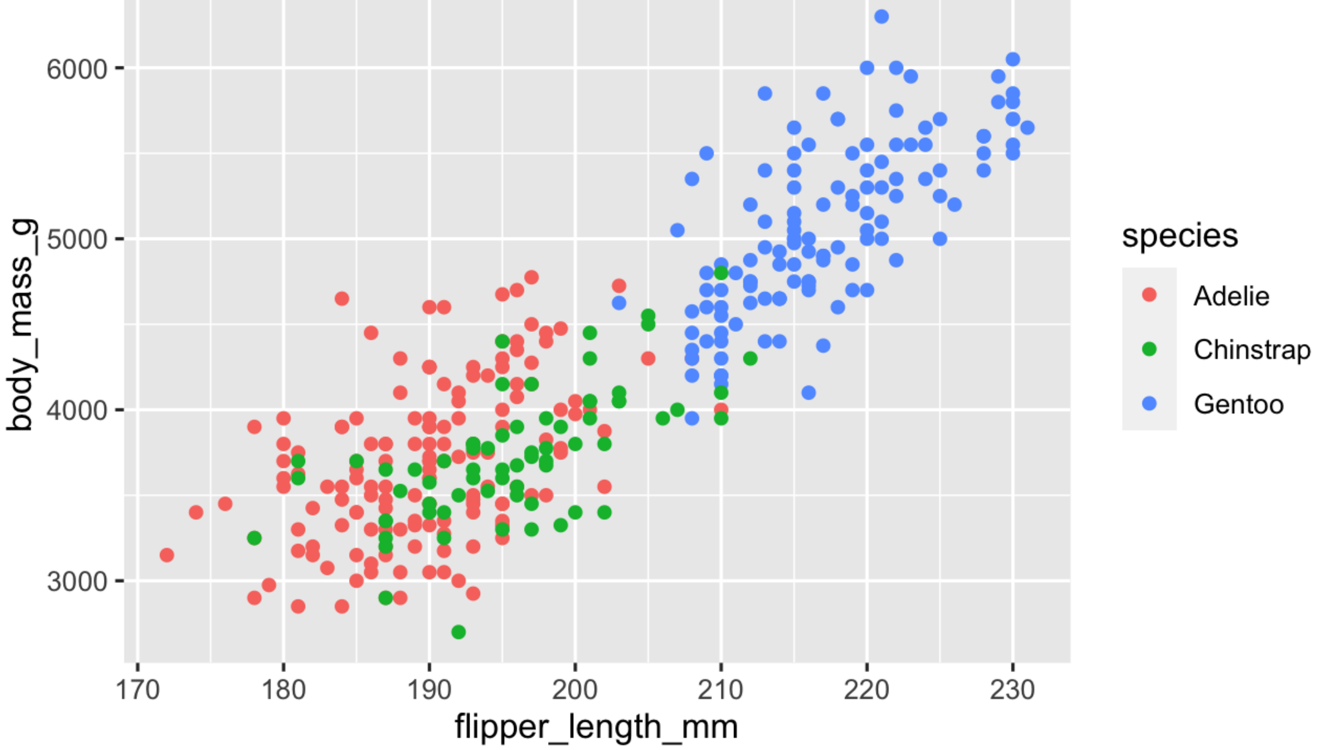



CHINSTRAP!
GENTOO!
ADÉLIE!



Note: In the raw data, bill dimensions are recorded as “culmen length” and “culmen depth”. The culmen is the dorsal ridge atop the bill.

```r
ggplot(data = penguins) +
  geom_point(mapping = aes(x = flipper_length_mm,
                           y = body_mass_g, color = species))
```



Your turn #2

Add color, size, alpha, and shape aesthetics to your graph.

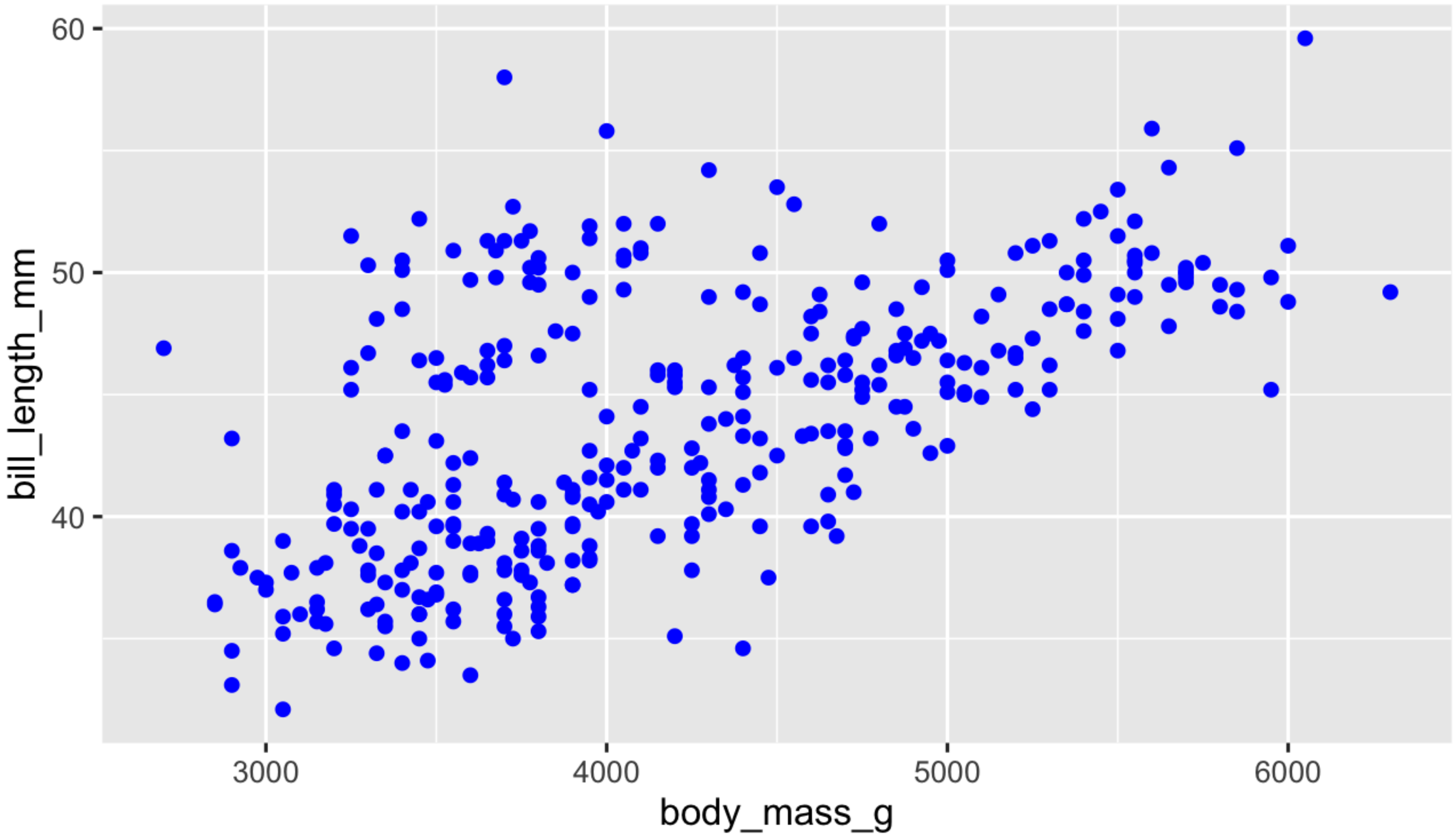
Experiment!

Do different things happen when you map aesthetics to discrete and continuous variables?

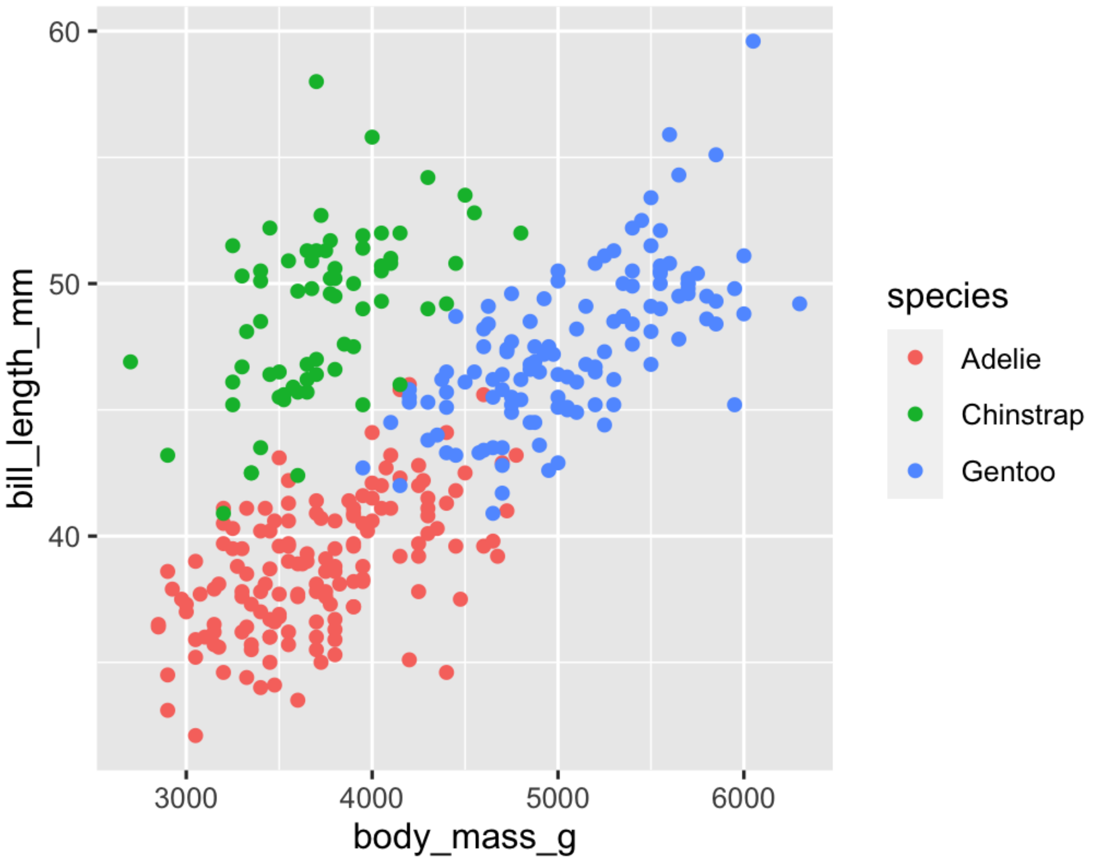
What happens when you use more than one aesthetic?

04:00

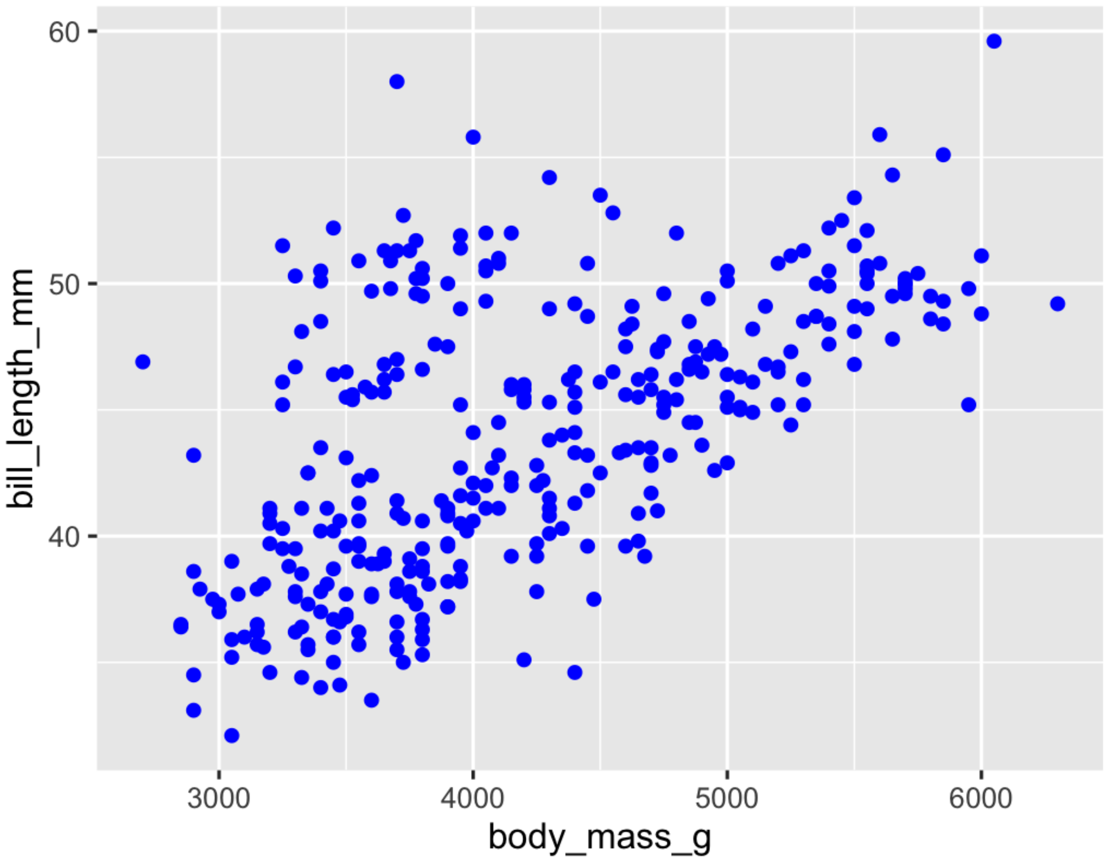
How would you make this plot?



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

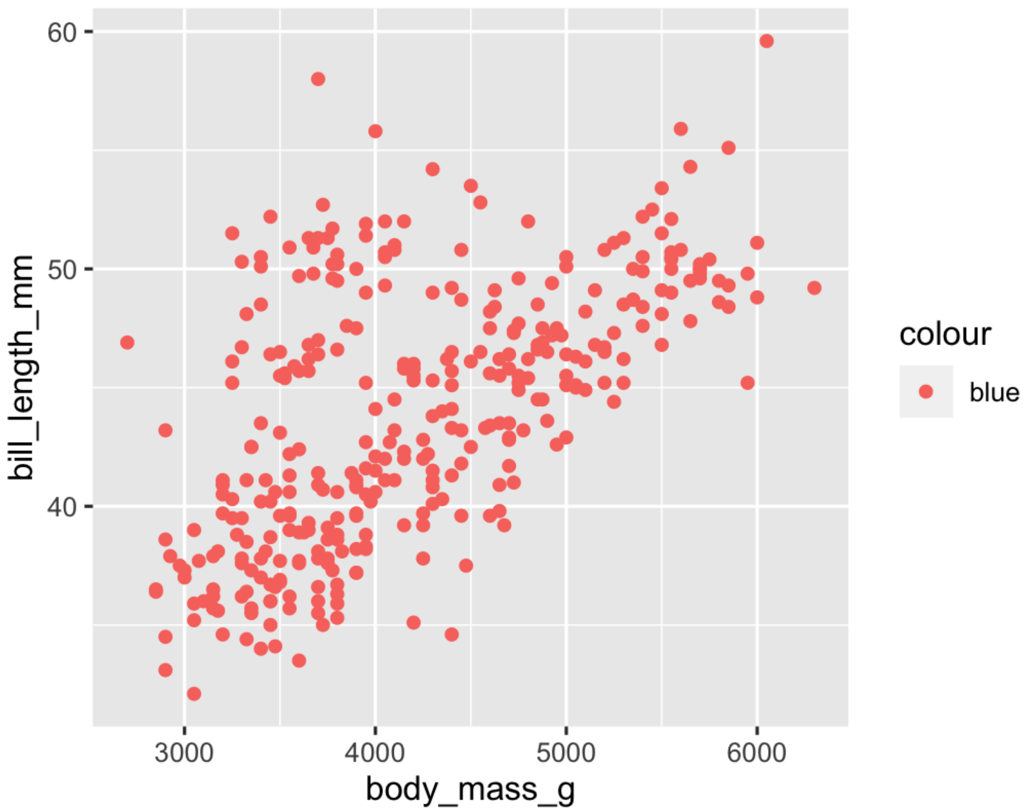


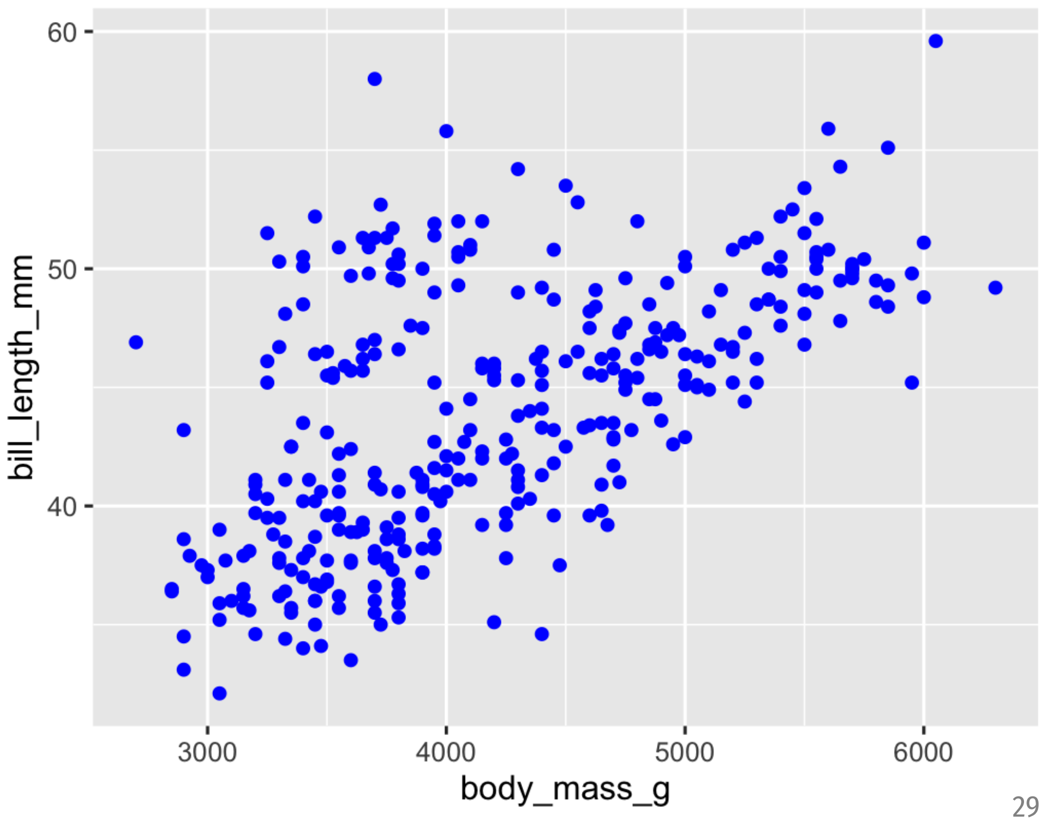
```r
ggplot(penguins) +
  geom_point(aes(x = body_mass_g,
                 y = bill_length_mm),
             color = "blue")
```



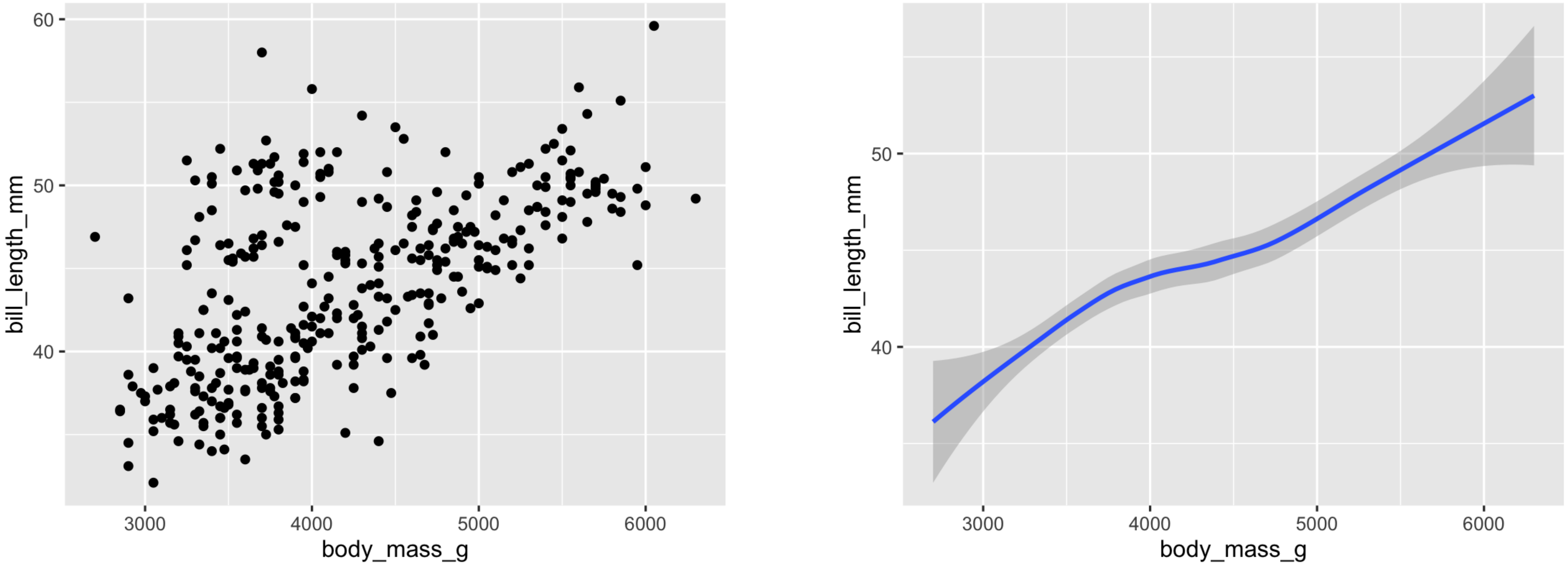
```r
ggplot(penguins) +
  geom_point(aes(x = body_mass_g,
                 y = bill_length_mm,
                 color = "blue"))
```

```r
ggplot(penguins) +
  geom_point(aes(x = body_mass_g,
                 y = bill_length_mm)
             color = "blue")
```





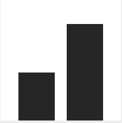
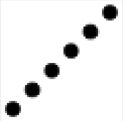
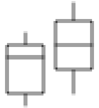
Same aesthetics, different geoms



Geoms

```
ggplot(data = DATA) +
  GEOM_FUNCTION(mapping = aes(AESTHETIC MAPPINGS))
```

Possible geoms

	Example geom	What it makes
	`geom_col()`	Bar charts
	`geom_text()`	Text
	`geom_point()`	Points
	`geom_boxplot()`	Boxplots
	`geom_sf()`	Maps

Possible geoms

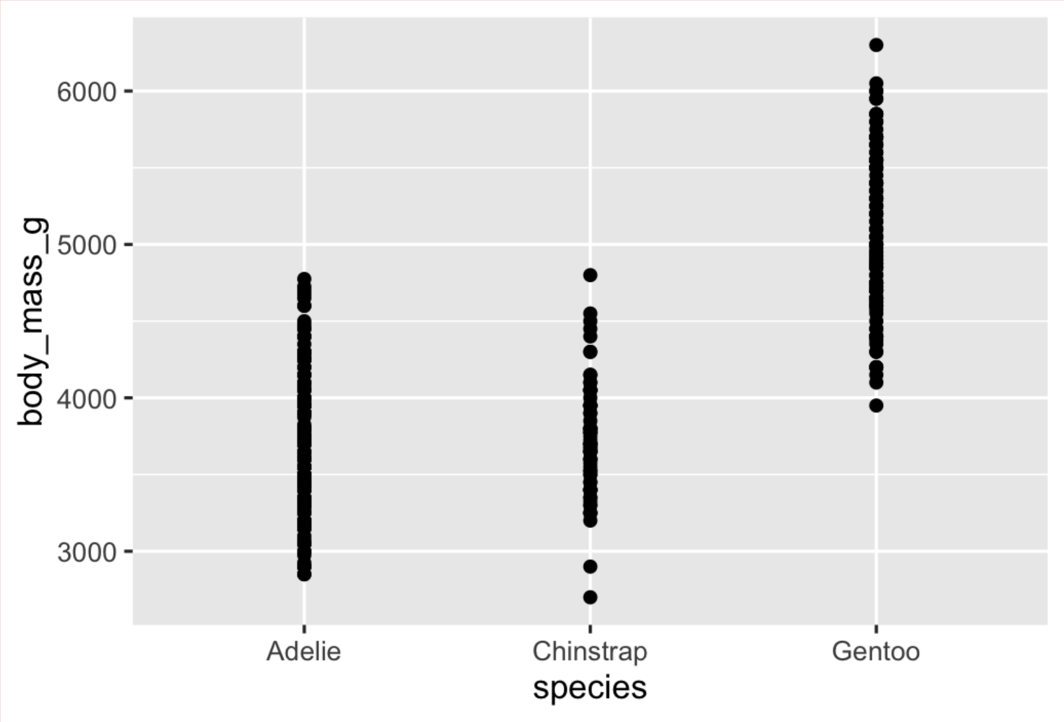
There are dozens of possible geoms!

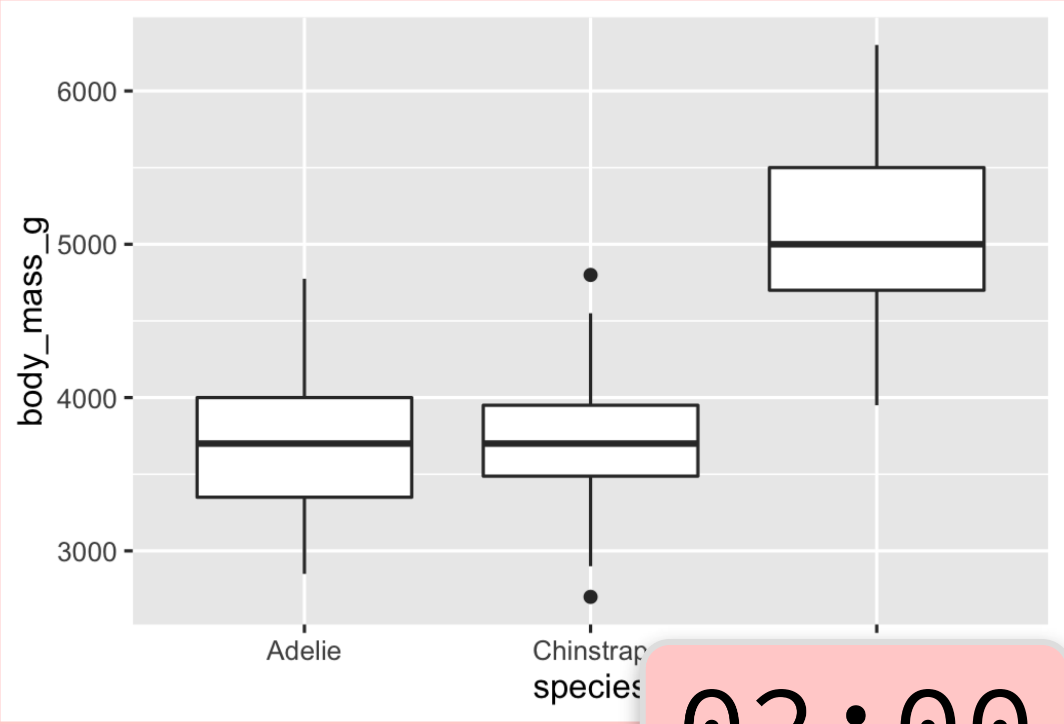
See the ggplot2 documentation for
complete examples of all the different geom layers

Also see the ggplot cheatsheet

Your turn #3

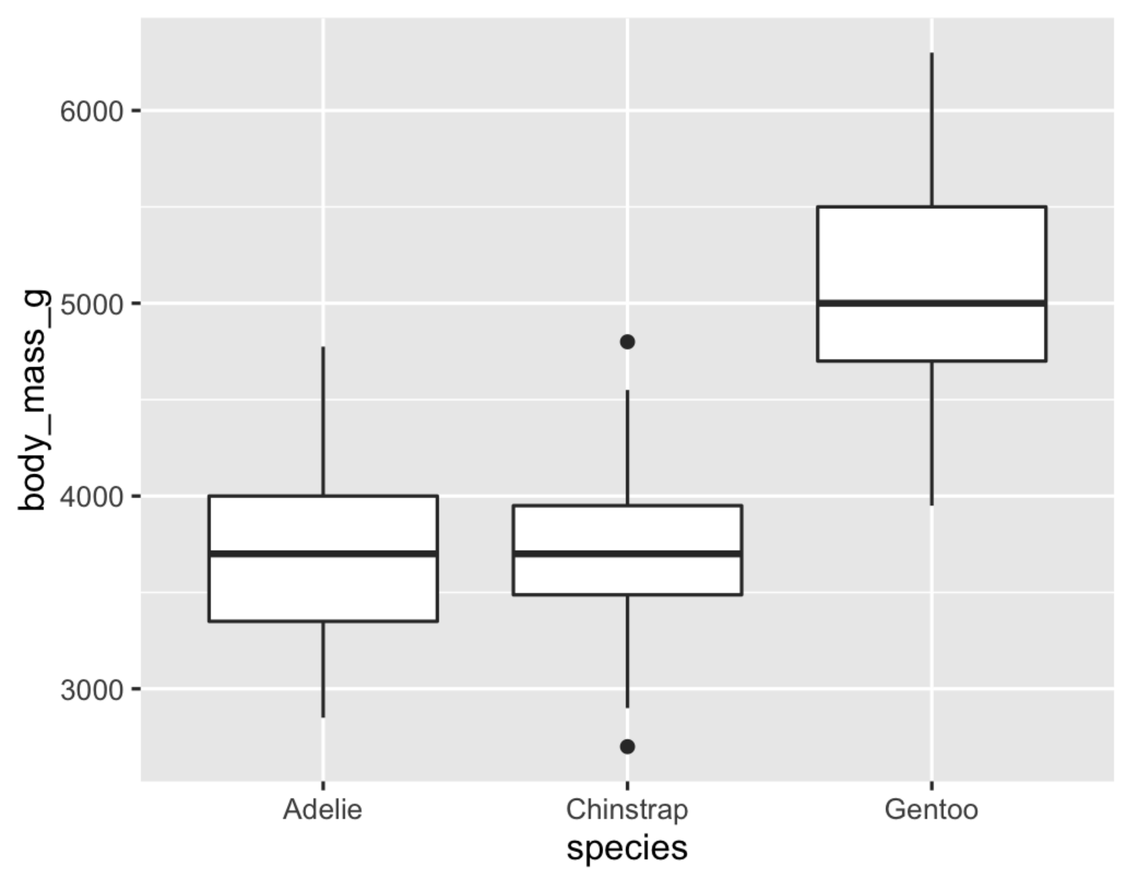
Replace this scatterplot with boxplots. Use the cheatsheet.





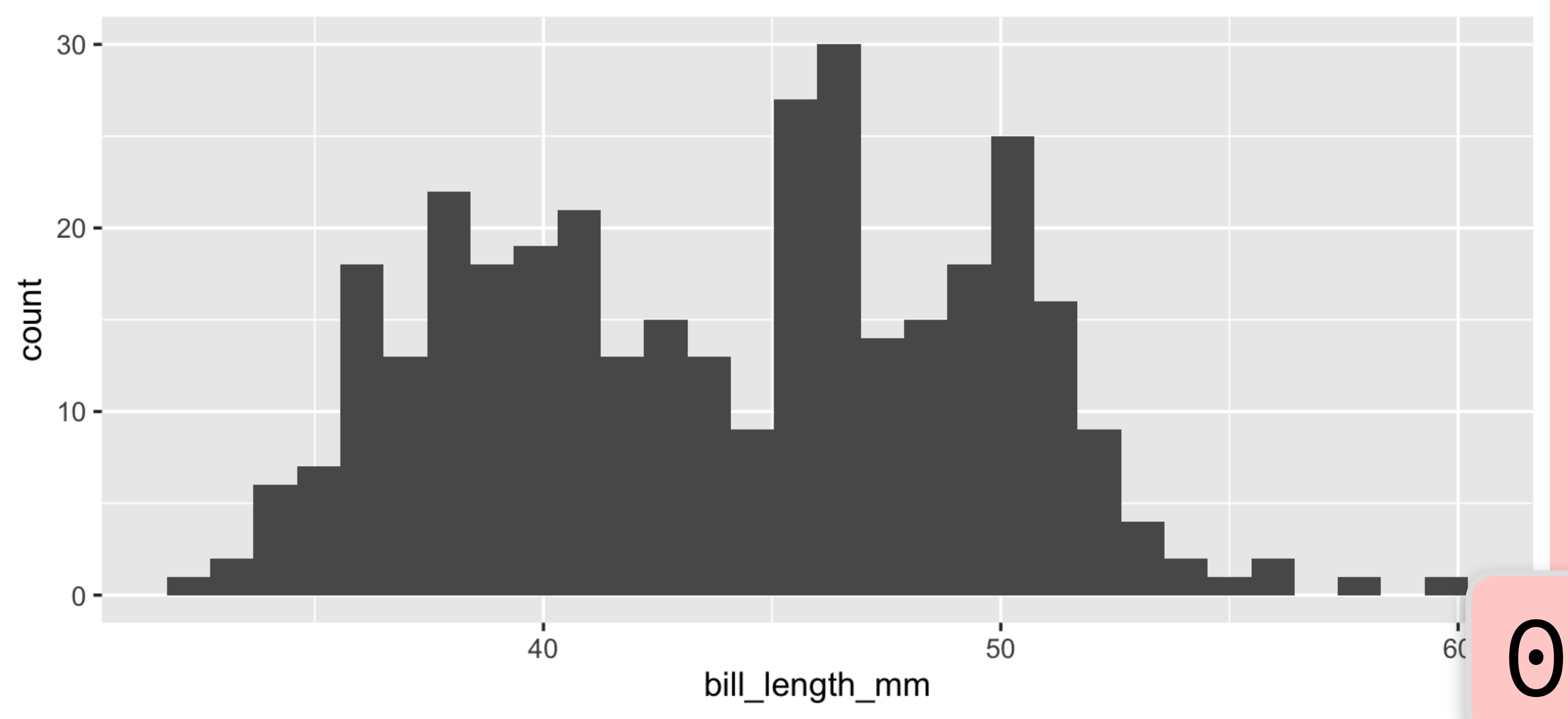
02:00

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



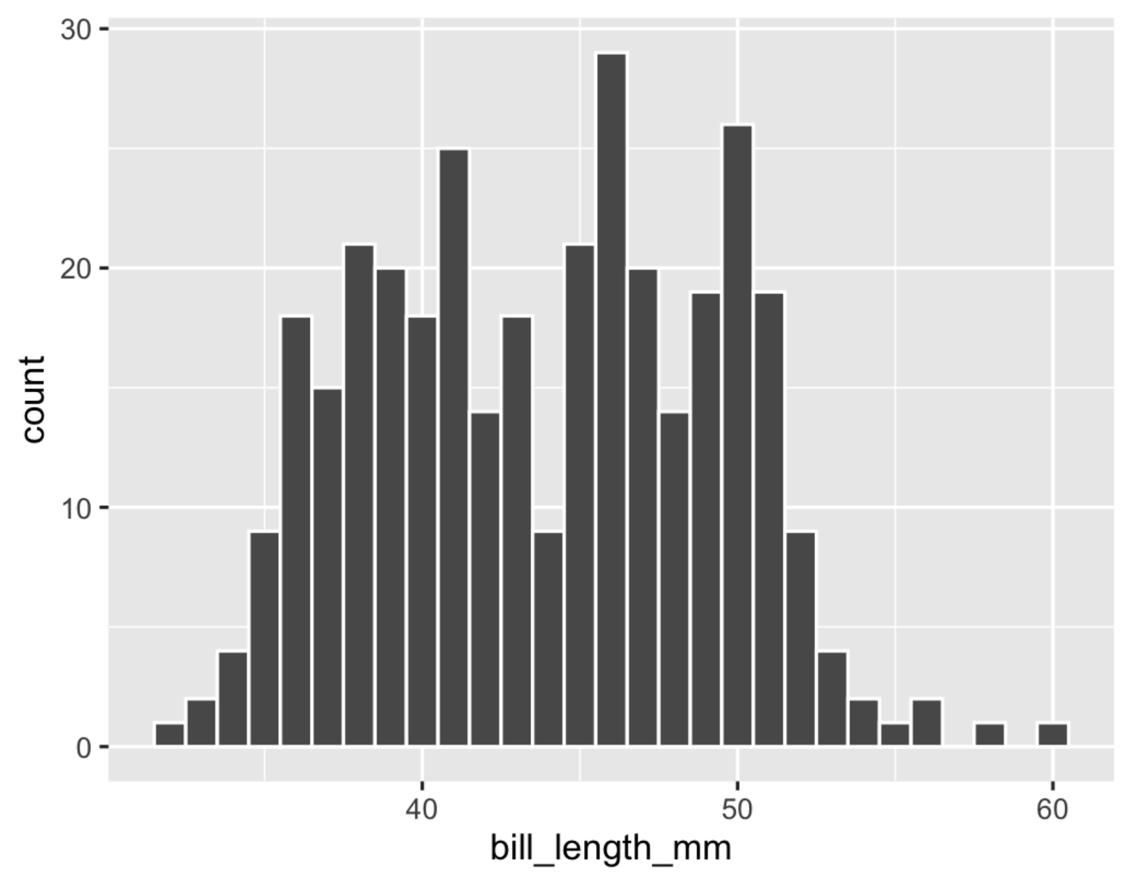
Your turn #4

Make a histogram of `bill_length_mm`. Use the cheetsheet.
Hint: don't supply a `y` variable.



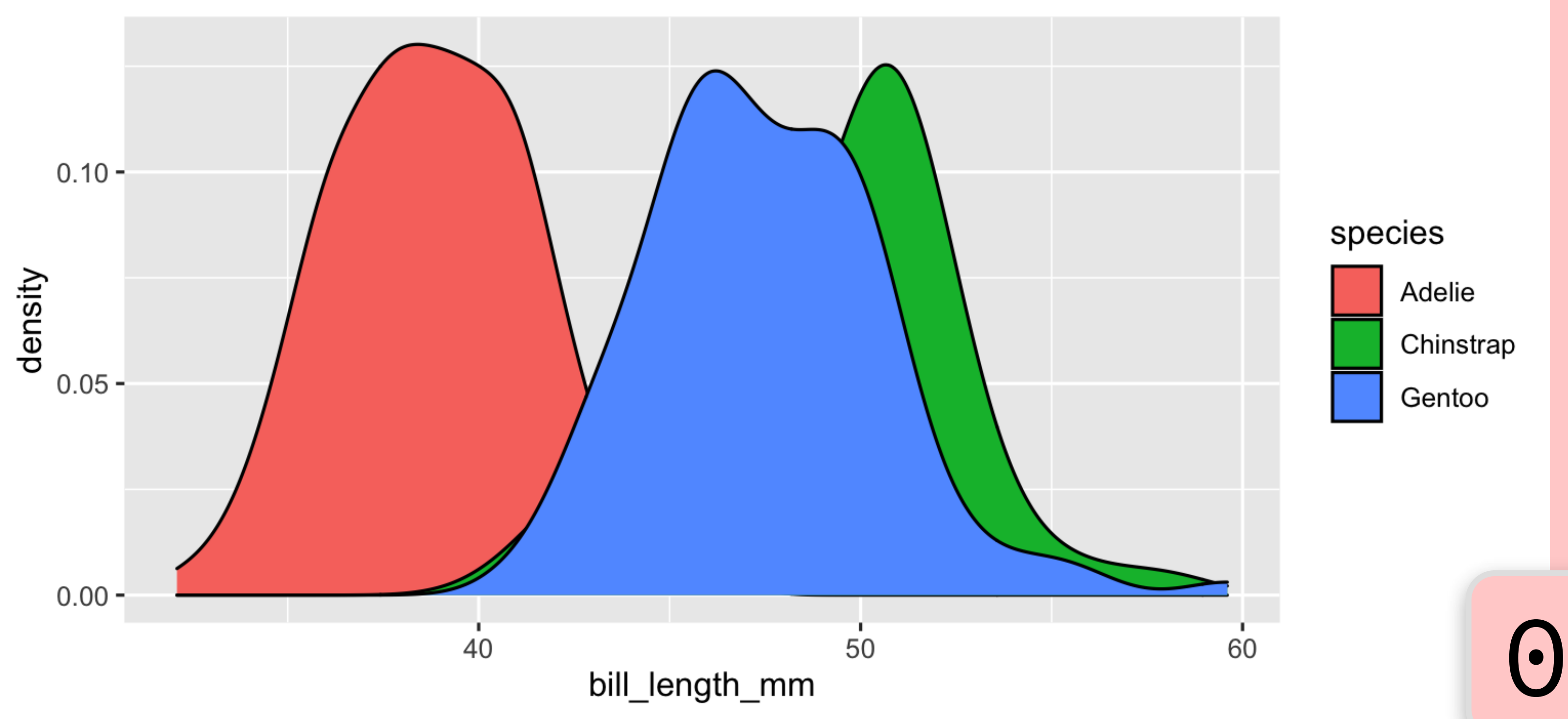
02:00

```r
ggplot(penguins) +
  geom_histogram(aes(x = bill_length_mm),
                 binwidth = 1,
                 color = "white")
```



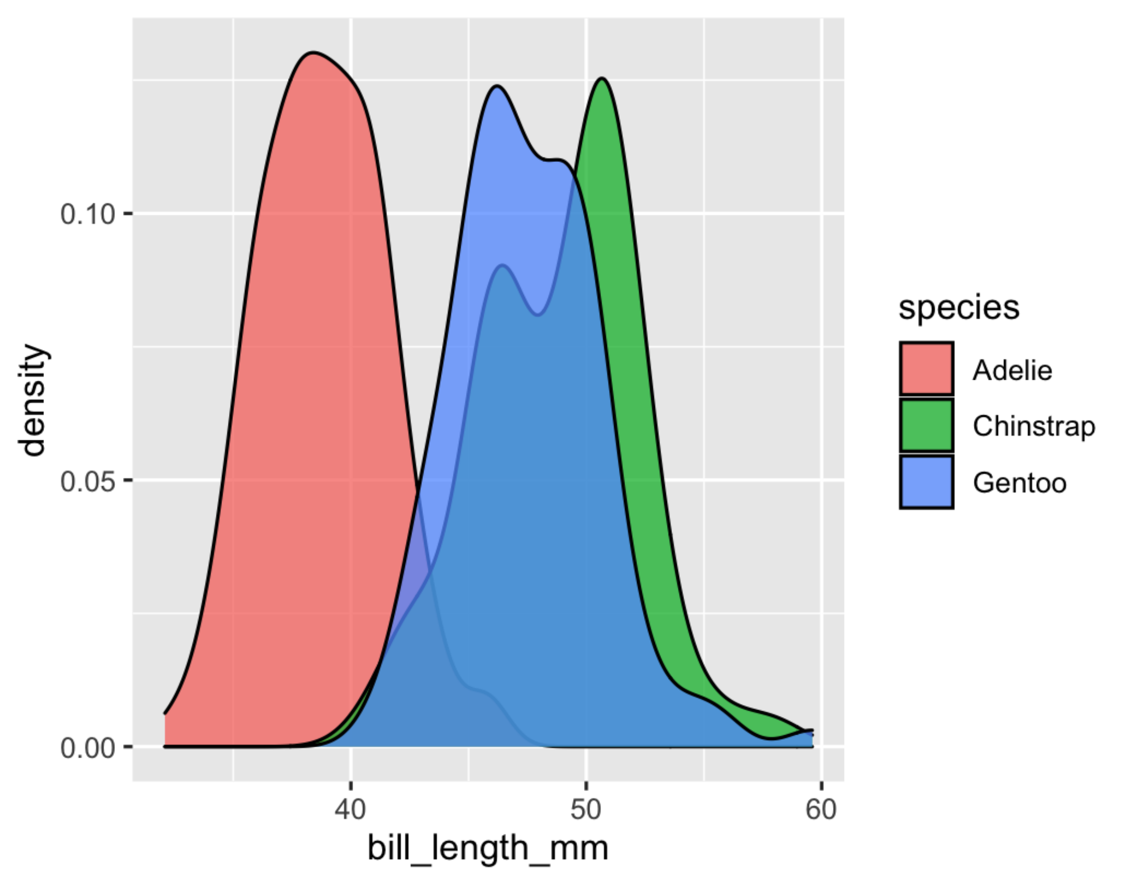
Your turn #5

Make this density plot of `bill_length_mm` filled by `species`. Use the cheatsheet. Hint: don't supply a `y` variable.

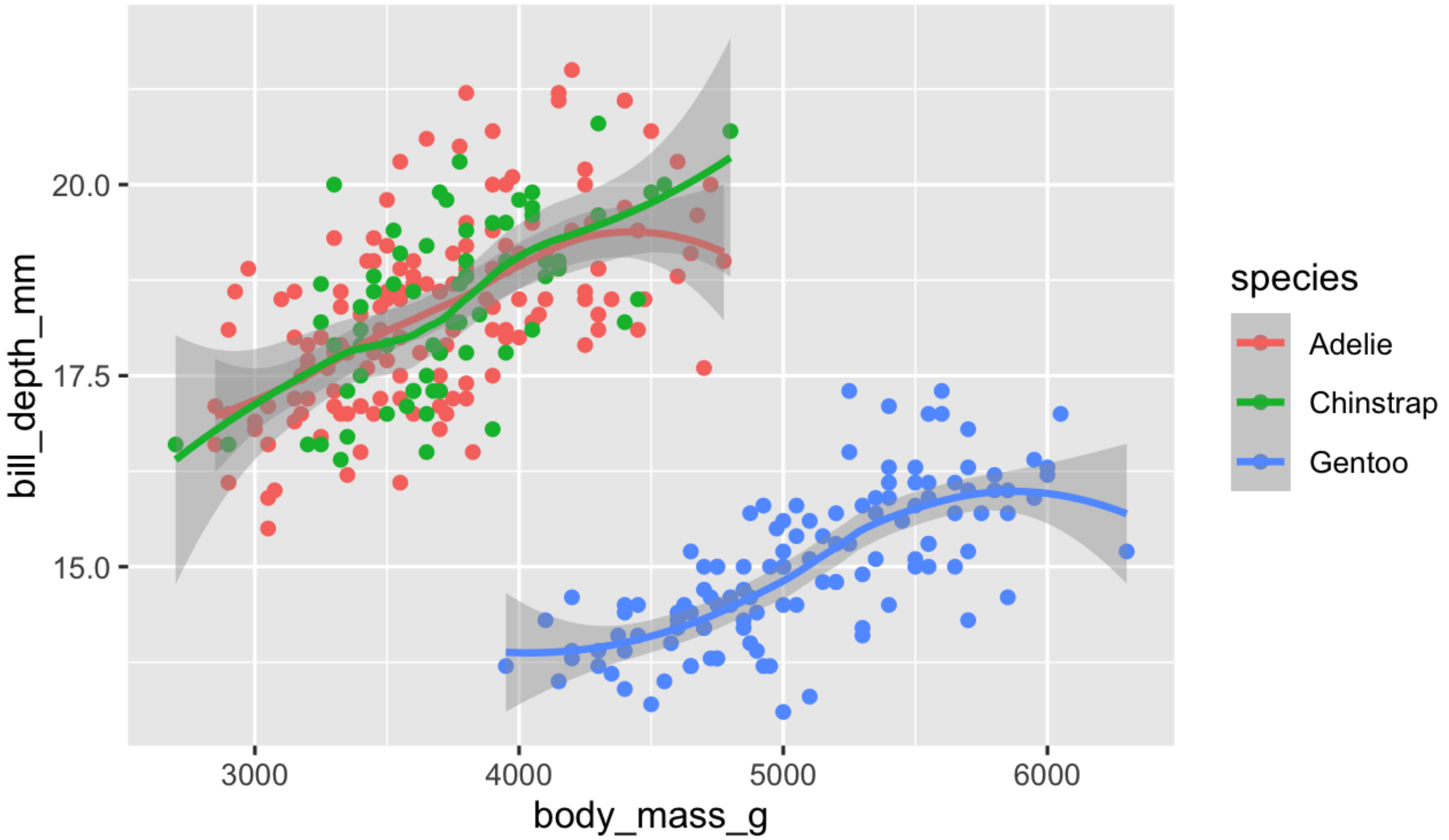


02:00

```r
ggplot(penguins) +
  geom_density(aes(x = bill_length_mm,
                   fill = species),
               alpha = 0.75)
```



Complex graphs!



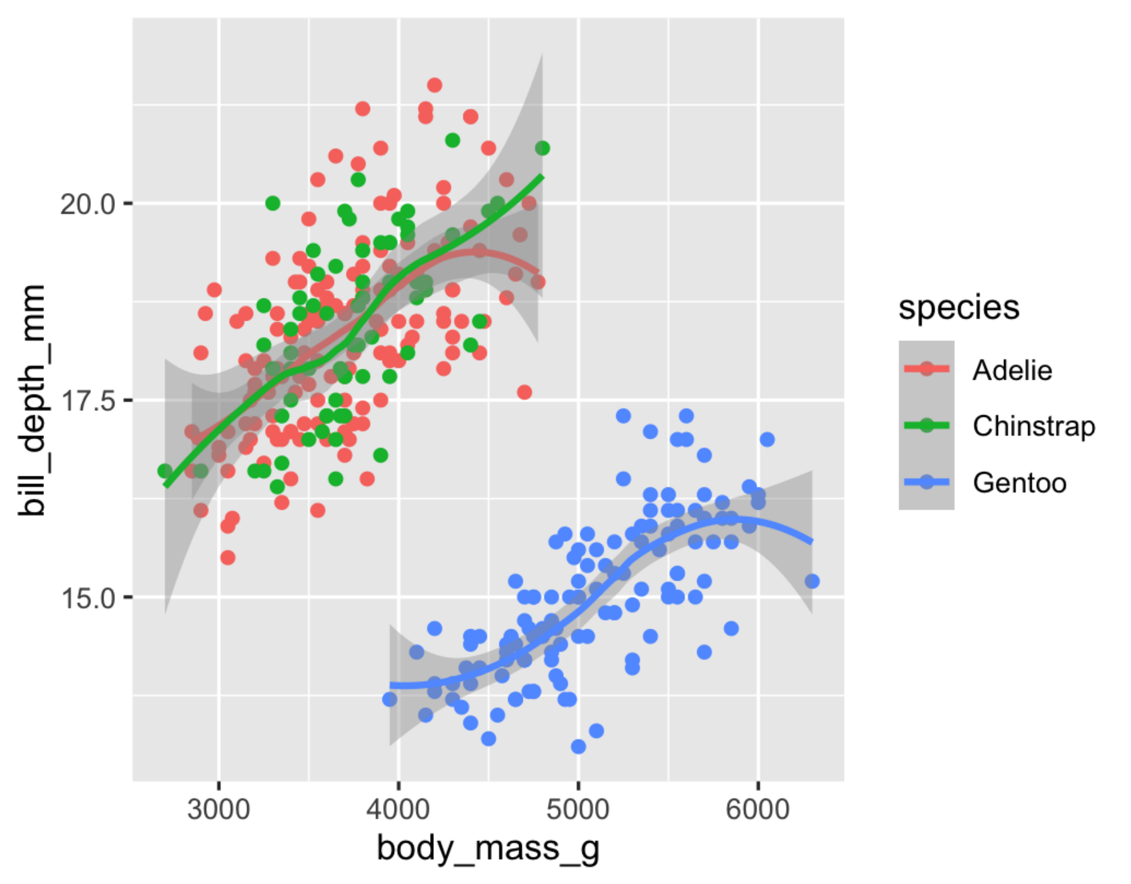
Your turn #6

Predict what this code will do. Then run it.

```
ggplot(data = penguins) +
  geom_point(mapping = aes(x = body_mass_g,
                           y = bill_depth_mm,
                           color = species)) +
  geom_smooth(mapping = aes(x = body_mass_g,
                            y = bill_depth_mm,
                            color = species))
```

01:00

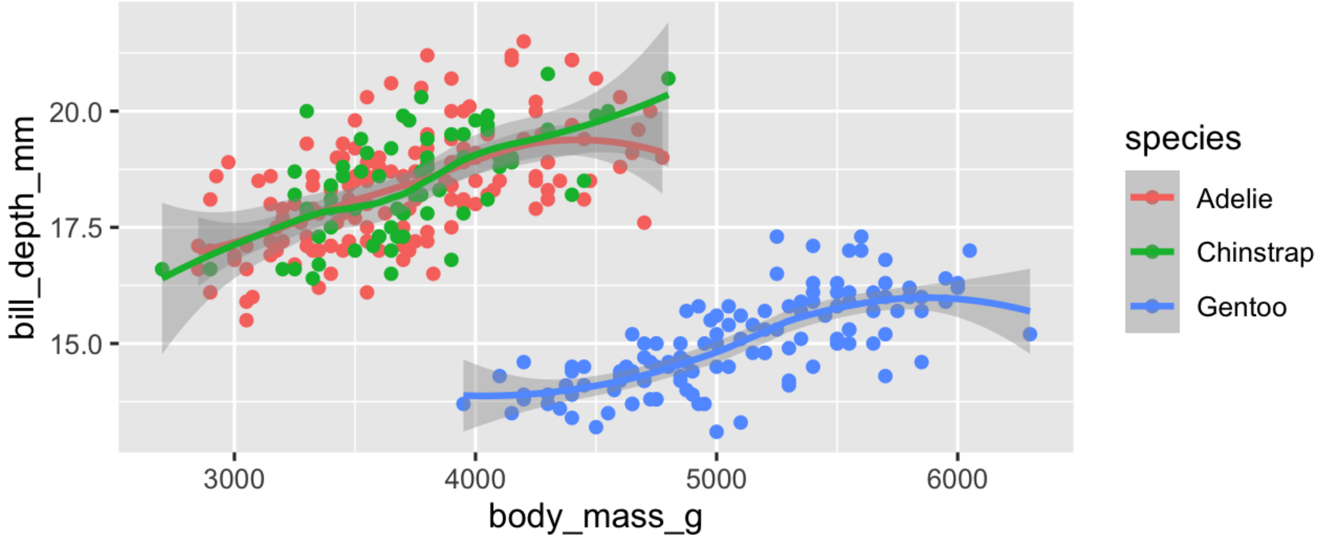
```r
ggplot(data = penguins) +
  geom_point(aes(x = body_mass_g,
                 y = bill_depth_mm,
                 color = species)) +
  geom_smooth(aes(x = body_mass_g,
                  y = bill_depth_mm,
                  color = species))
```



Global vs. local

Any aesthetics in `ggplot()` will show up in all `geom_` layers

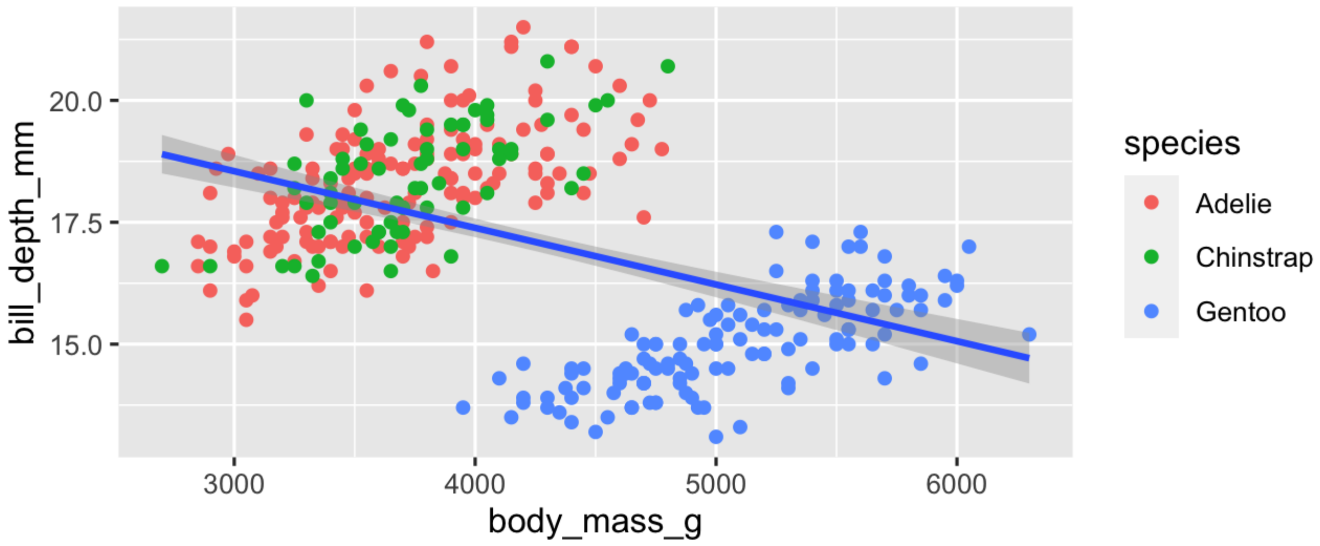
```
ggplot(penguins, aes(x = body_mass_g, y = bill_depth_mm, color = species)) +
  geom_point() +
  geom_smooth()
```



Global vs. local

Any aesthetics in `geom_` layers only apply to that layer

```
ggplot(penguins, mapping = aes(x = body_mass_g, y = bill_depth_mm)) +
  geom_point(mapping = aes(color = species)) +
  geom_smooth(method = "lm")
```



Gammar components as layers

So far we know about data, aesthetics, and geometries

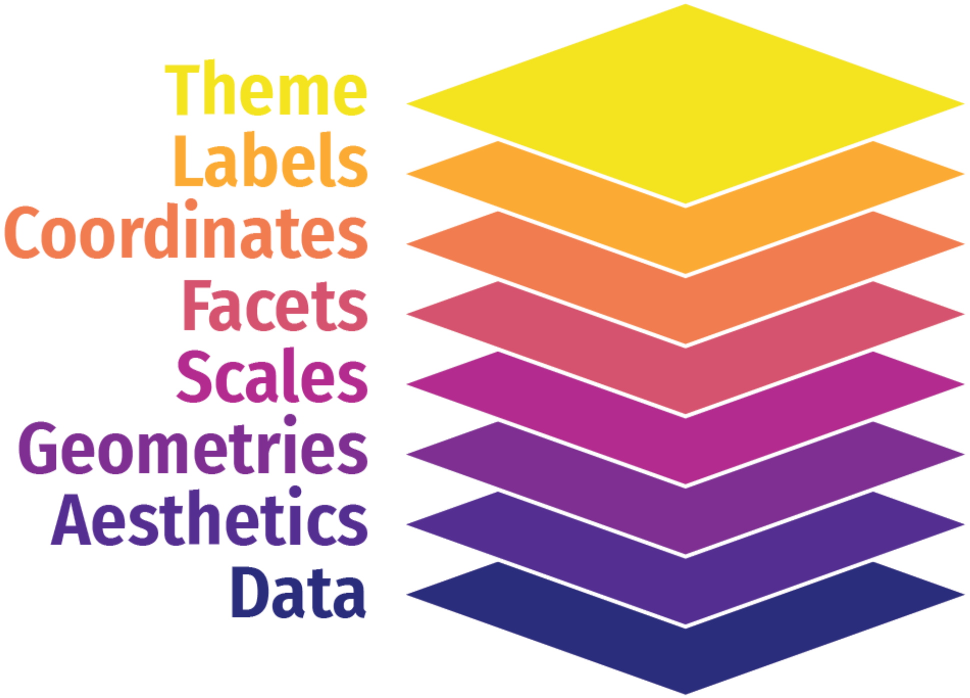
Think of these components as **layers**

Add them to foundational `ggplot()` with `+`

Additional layers

There are many of other grammatical layers we can use to describe graphs!

We sequentially add layers onto the foundational `ggplot()` plot to create complex figures

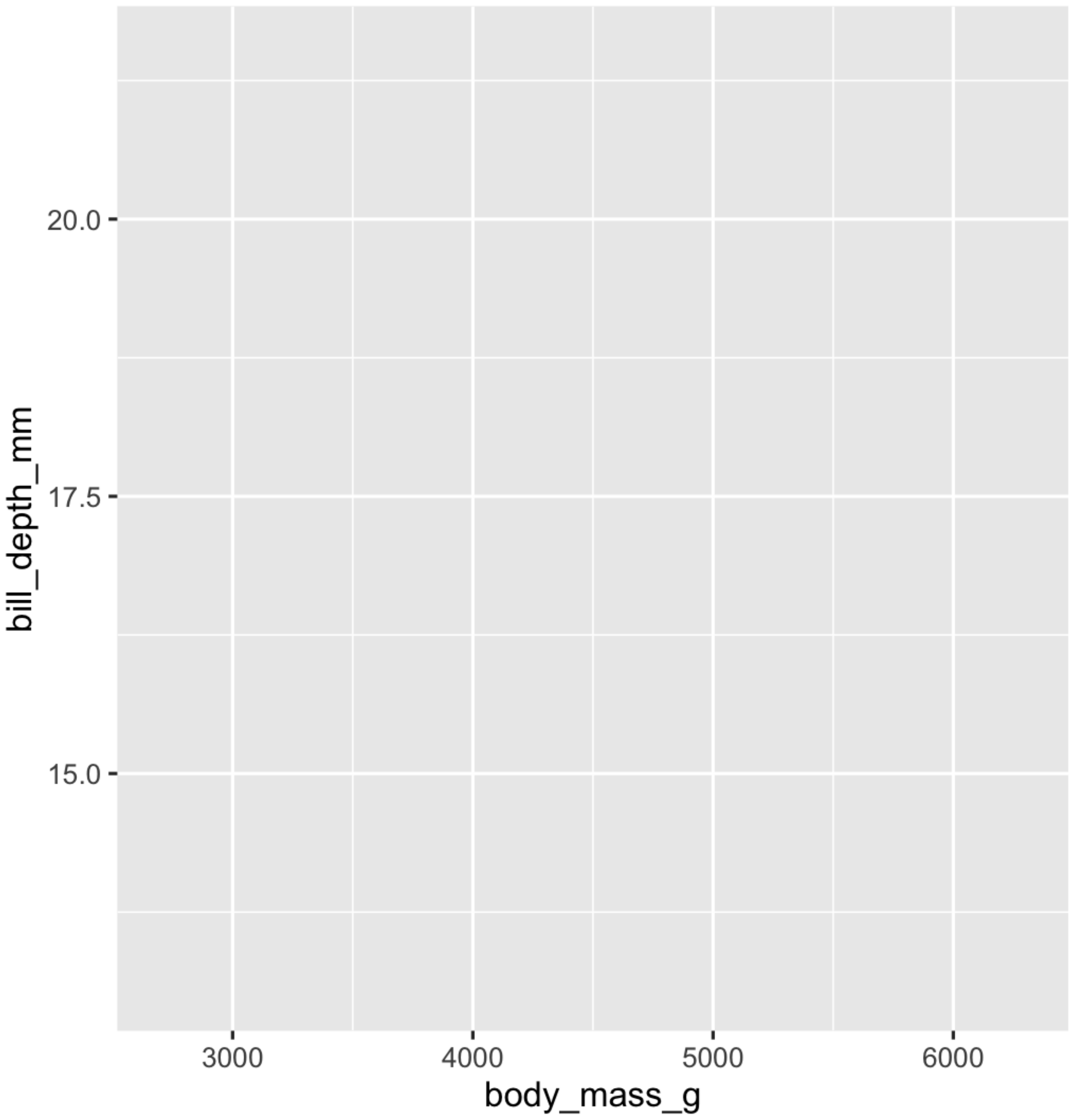


Putting it all together

We can build a plot sequentially to see how each grammatical layer changes the appearance

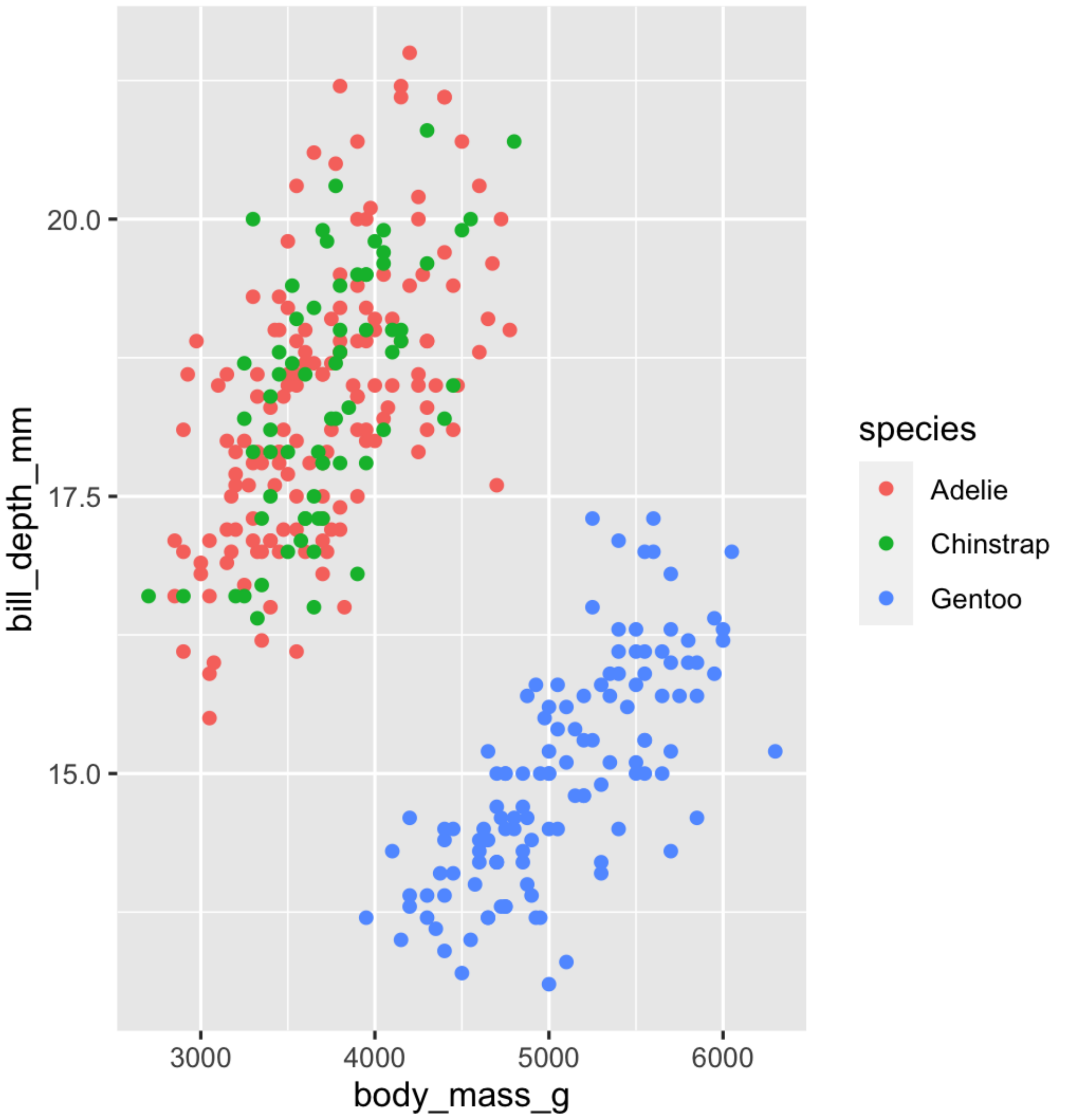
Start with data and aesthetics

```
ggplot(data = penguins,
       mapping = aes(x = body_mass_g,
                     y = bill_depth_mm,
                     color = species))
```



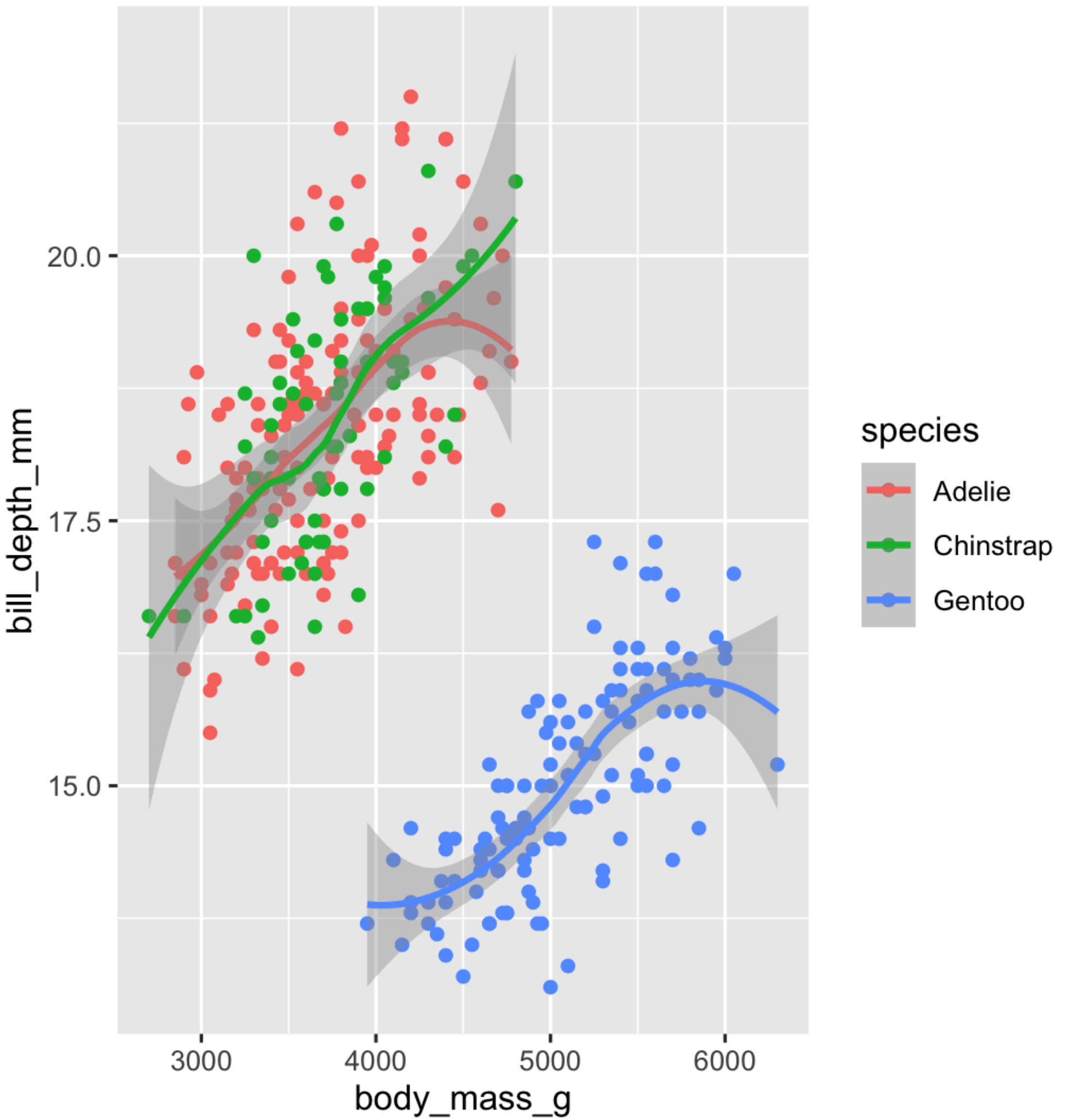
Add a point geom

```
ggplot(data = penguins,
       mapping = aes(x = body_mass_g,
                     y = bill_depth_mm,
                     color = species)) +
  geom_point()
```



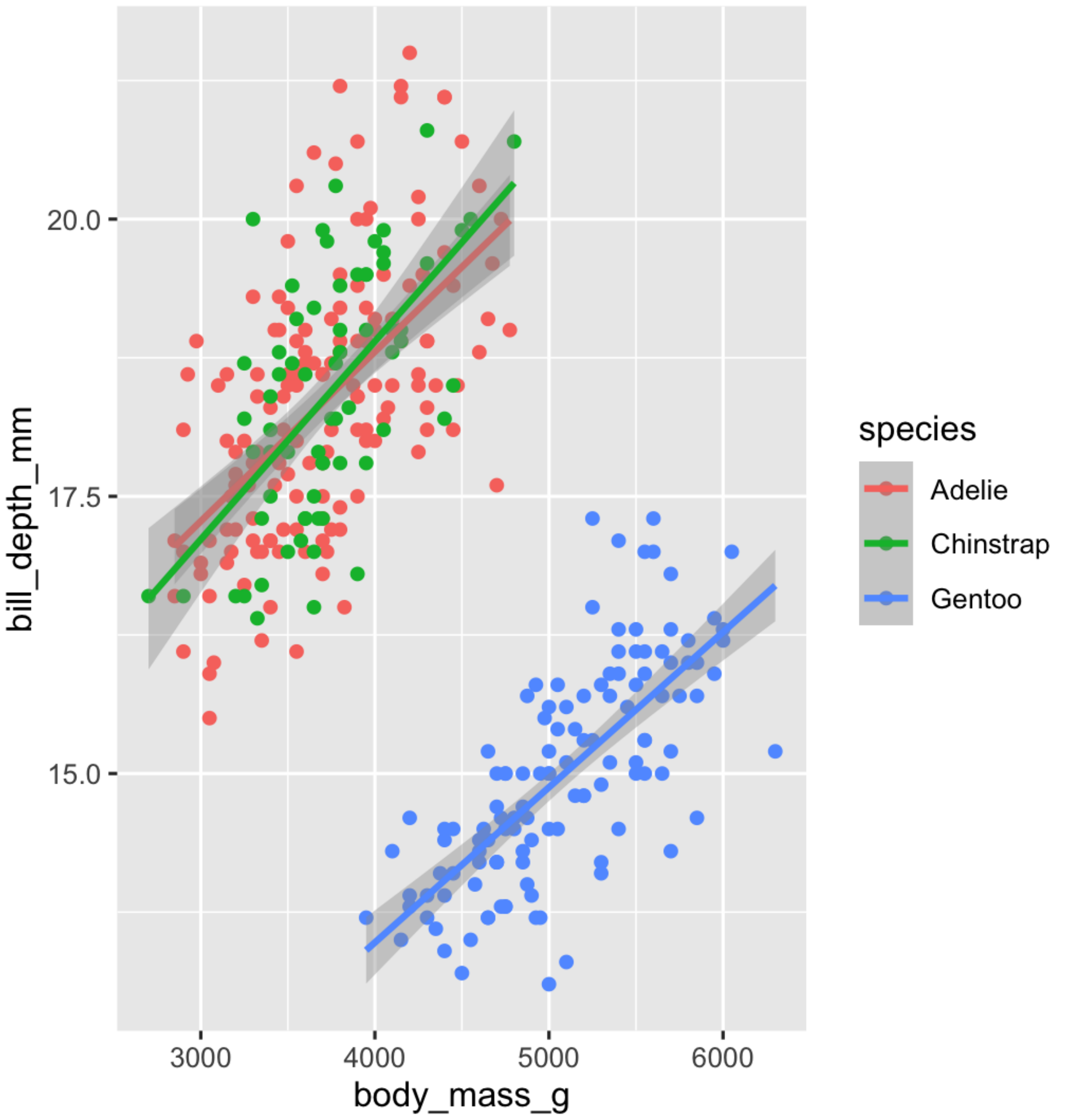
Add a smooth geom

```
ggplot(data = penguins,
       mapping = aes(x = body_mass_g,
                     y = bill_depth_mm,
                     color = species)) +
  geom_point() +
  geom_smooth()
```



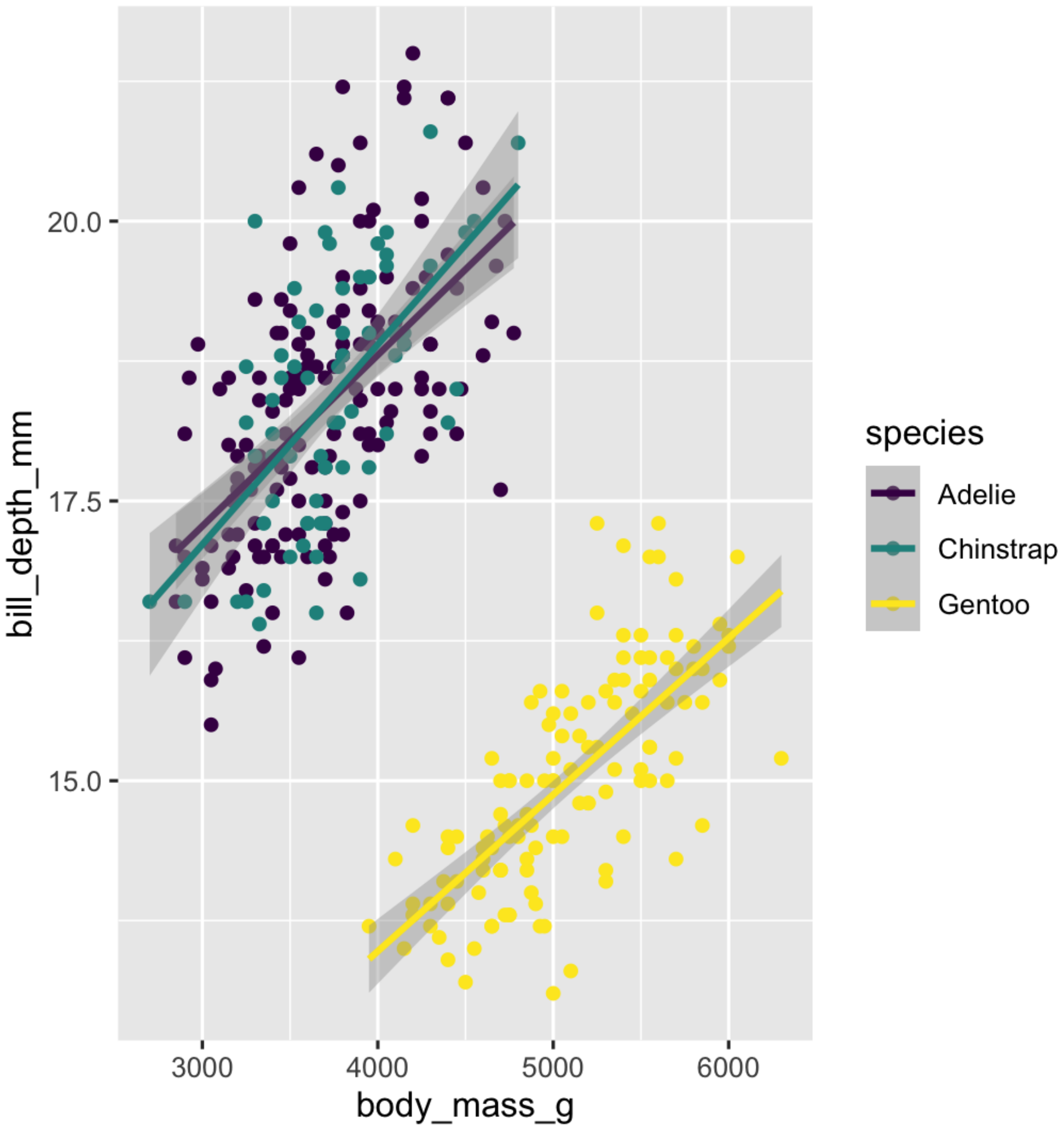
Make it straight

```
ggplot(data = penguins,
       mapping = aes(x = body_mass_g,
                     y = bill_depth_mm,
                     color = species)) +
  geom_point() +
  geom_smooth(method = "lm")
```



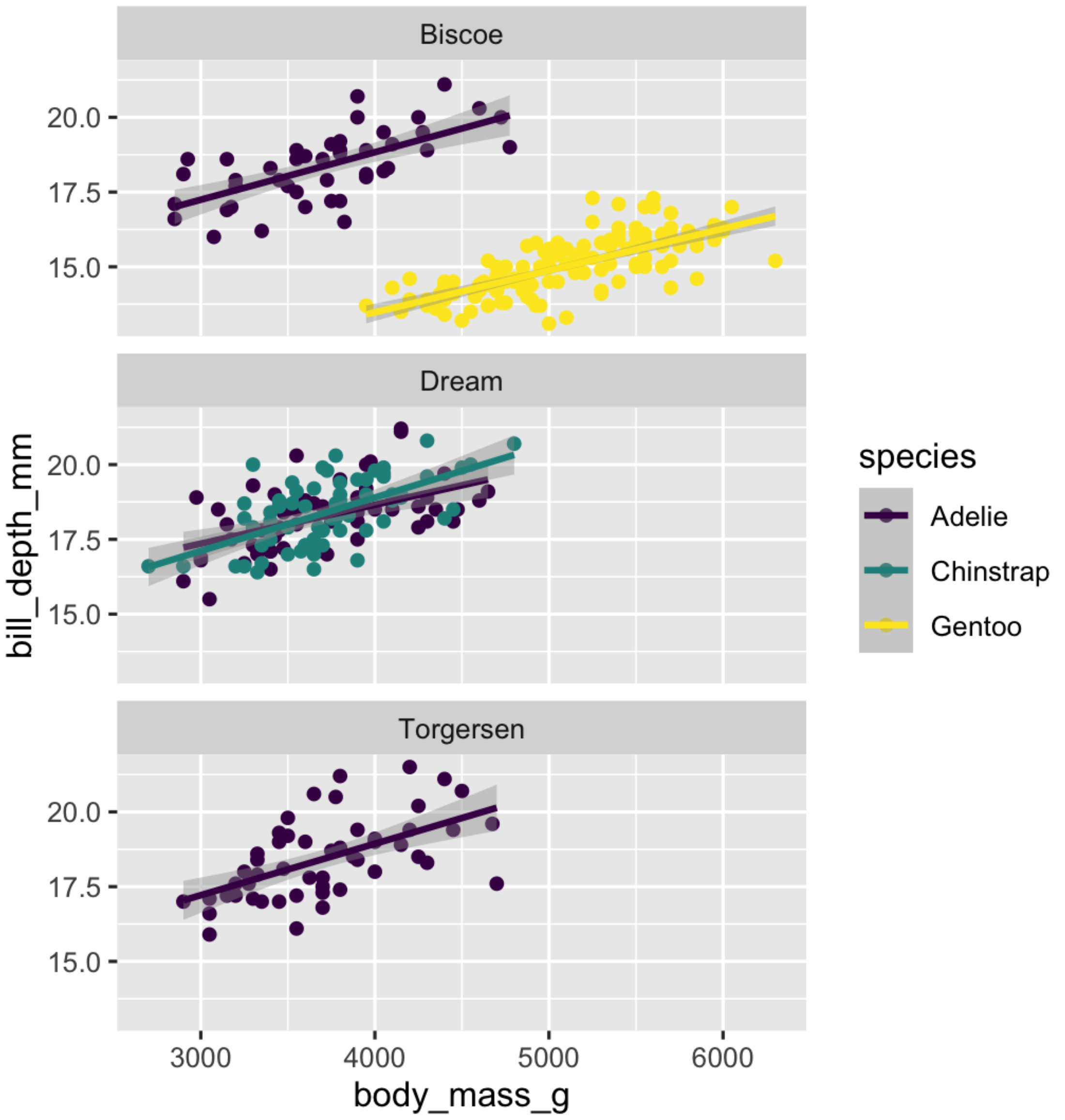
Use a viridis color scale

```r
ggplot(data = penguins,
       mapping = aes(x = body_mass_g,
                     y = bill_depth_mm,
                     color = species)) +
  geom_point() +
  geom_smooth(method = "lm") +
  scale_color_viridis_d()
```



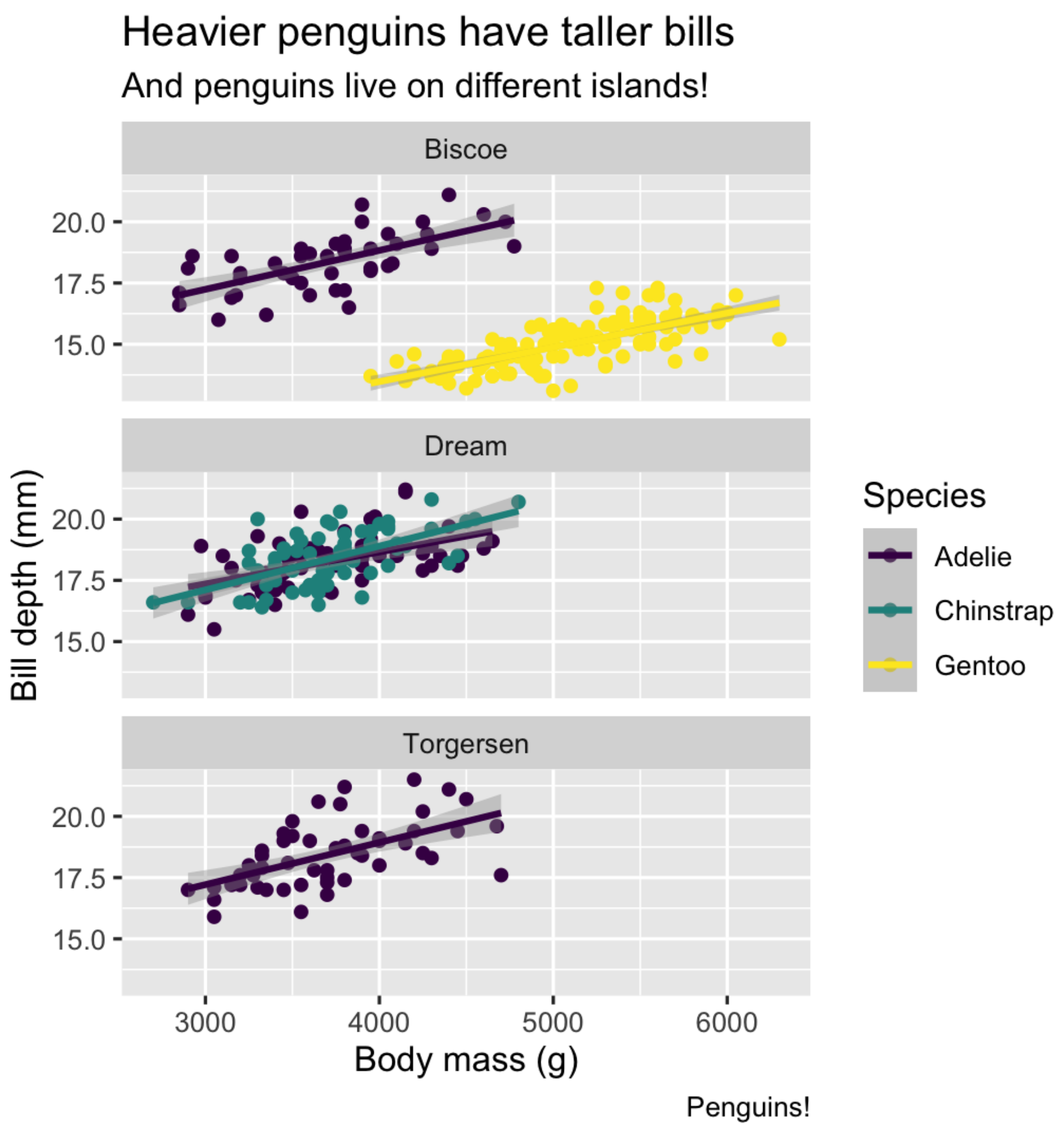
Facet by island

```r
ggplot(data = penguins,
       mapping = aes(x = body_mass_g,
                     y = bill_depth_mm,
                     color = species)) +
  geom_point() +
  geom_smooth(method = "lm") +
  scale_color_viridis_d() +
  facet_wrap(vars(island), ncol = 1)
```



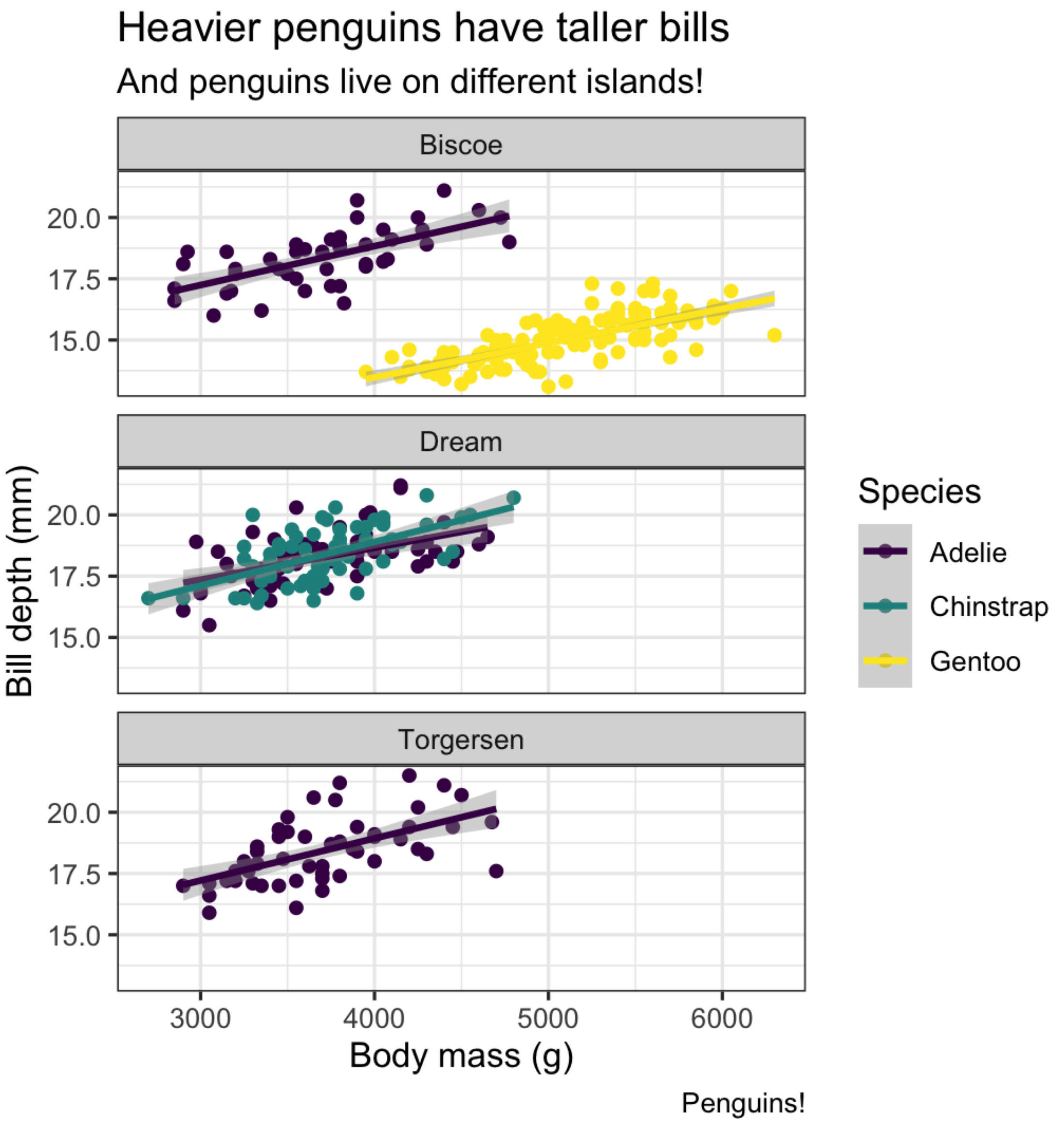
Add labels

```
ggplot(data = penguins,
       mapping = aes(x = body_mass_g,
                     y = bill_depth_mm,
                     color = species)) +
  geom_point() +
  geom_smooth(method = "lm") +
  scale_color_viridis_d() +
  facet_wrap(vars(island), ncol = 1) +
  labs(x = "Body mass (g)", y = "Bill depth
       color = "Species",
       title = "Heavier penguins have taller
       subtitle = "And penguins live on diffe
       caption = "Penguins!")
```



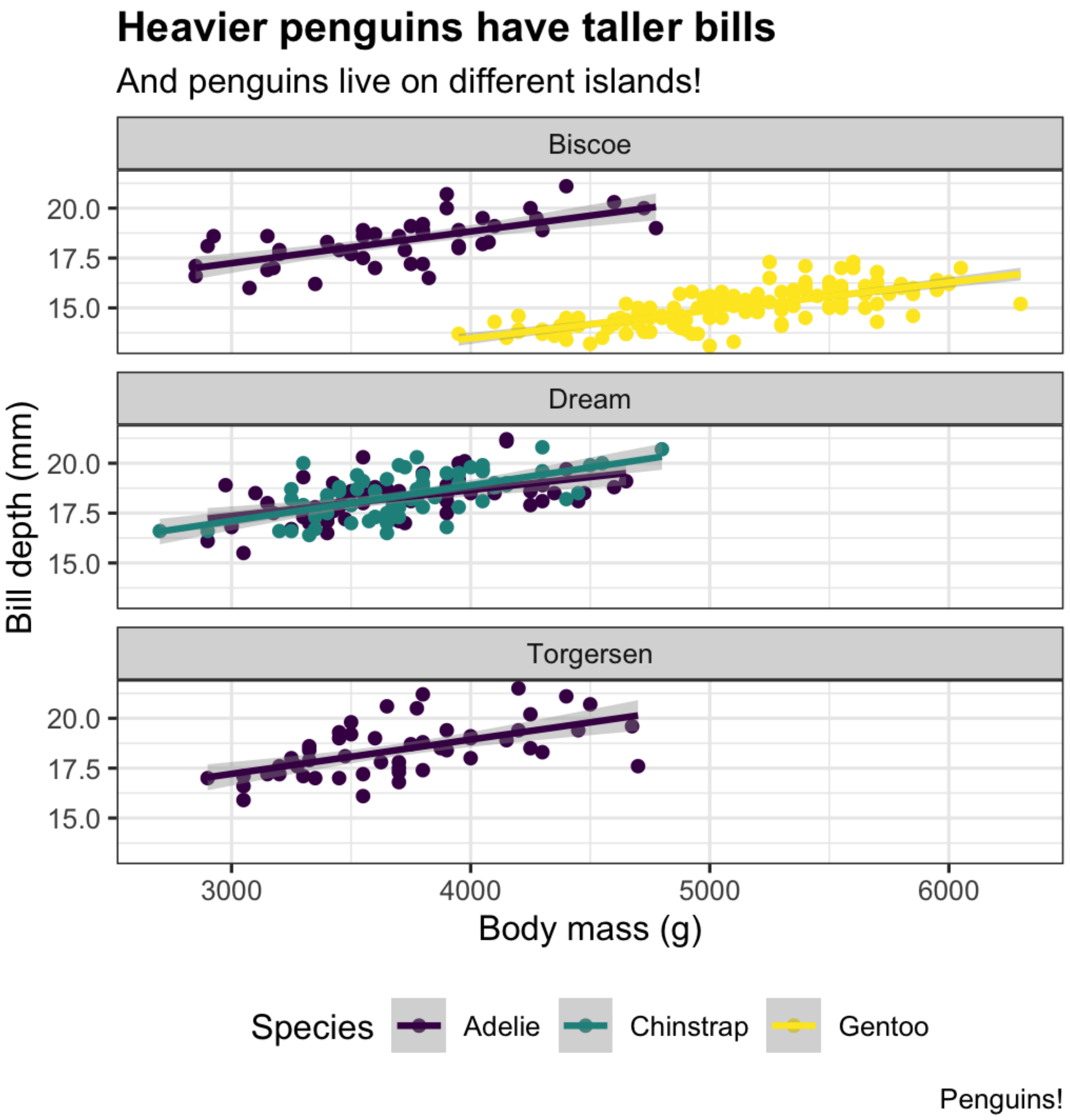
Add a theme

```
ggplot(data = penguins,
       mapping = aes(x = body_mass_g,
                     y = bill_depth_mm,
                     color = species)) +
  geom_point() +
  geom_smooth(method = "lm") +
  scale_color_viridis_d() +
  facet_wrap(vars(island), ncol = 1) +
  labs(x = "Body mass (g)", y = "Bill depth
       color = "Species",
       title = "Heavier penguins have taller
       subtitle = "And penguins live on diffe
       caption = "Penguins!") +
  theme_bw()
```



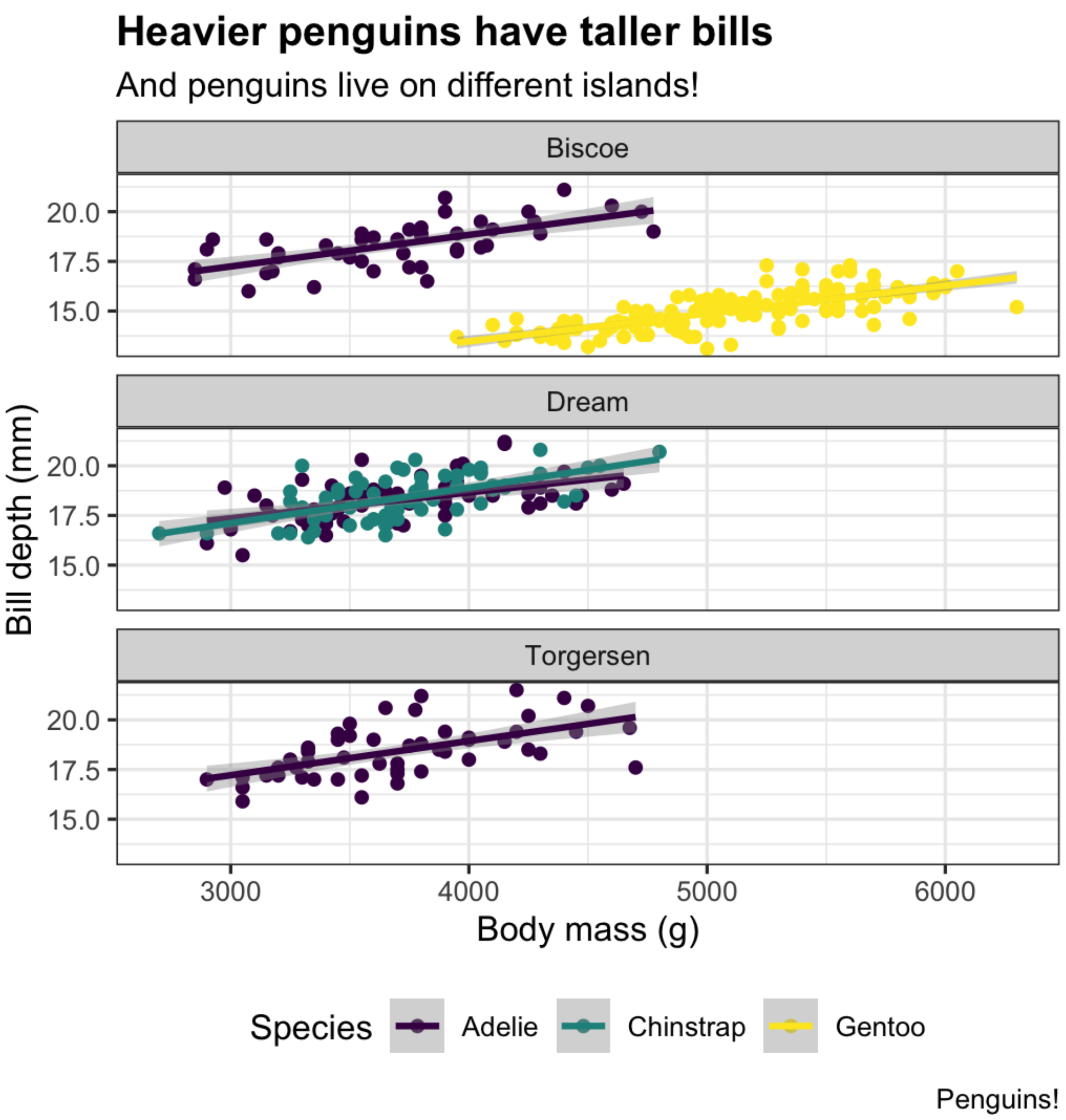
Modify the theme

```
ggplot(data = penguins,
       mapping = aes(x = body_mass_g,
                     y = bill_depth_mm,
                     color = species)) +
  geom_point() +
  geom_smooth(method = "lm") +
  scale_color_viridis_d() +
  facet_wrap(vars(island), ncol = 1) +
  labs(x = "Body mass (g)", y = "Bill depth
       color = "Species",
       title = "Heavier penguins have taller
       subtitle = "And penguins live on diffe
       caption = "Penguins!") +
  theme_bw() +
  theme(legend.position = "bottom",
        plot.title = element_text(face = "bo
```

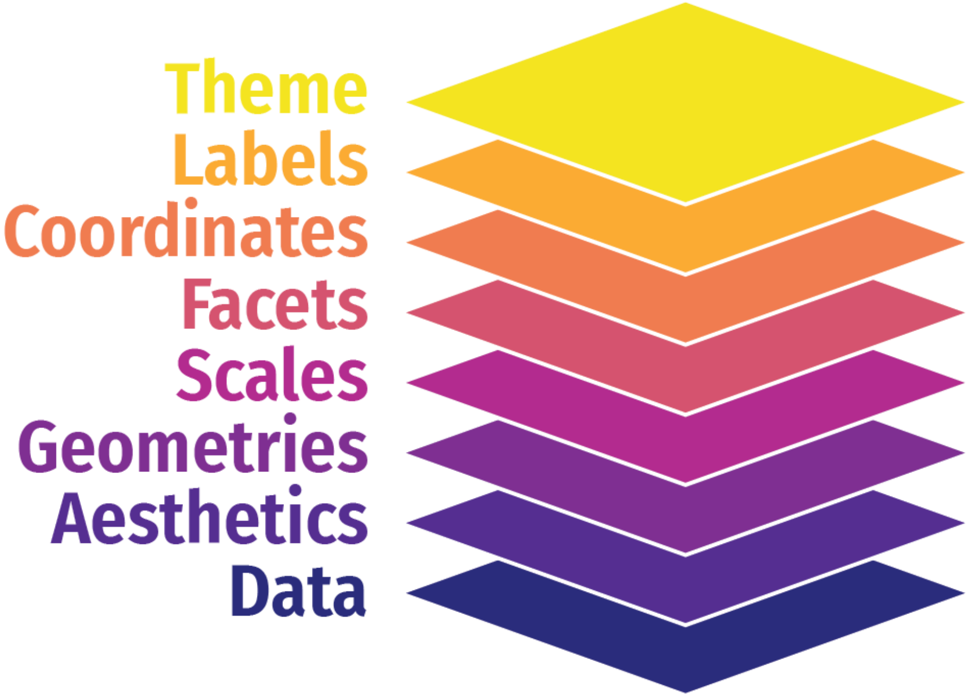


Finished!

```
ggplot(data = penguins,
       mapping = aes(x = body_mass_g,
                     y = bill_depth_mm,
                     color = species)) +
  geom_point() +
  geom_smooth(method = "lm") +
  scale_color_viridis_d() +
  facet_wrap(vars(island), ncol = 1) +
  labs(x = "Body mass (g)", y = "Bill depth
       color = "Species",
       title = "Heavier penguins have taller
       subtitle = "And penguins live on diffe
       caption = "Penguins!") +
  theme_bw() +
  theme(legend.position = "bottom",
        plot.title = element_text(face = "bo
```



So many possibilities!



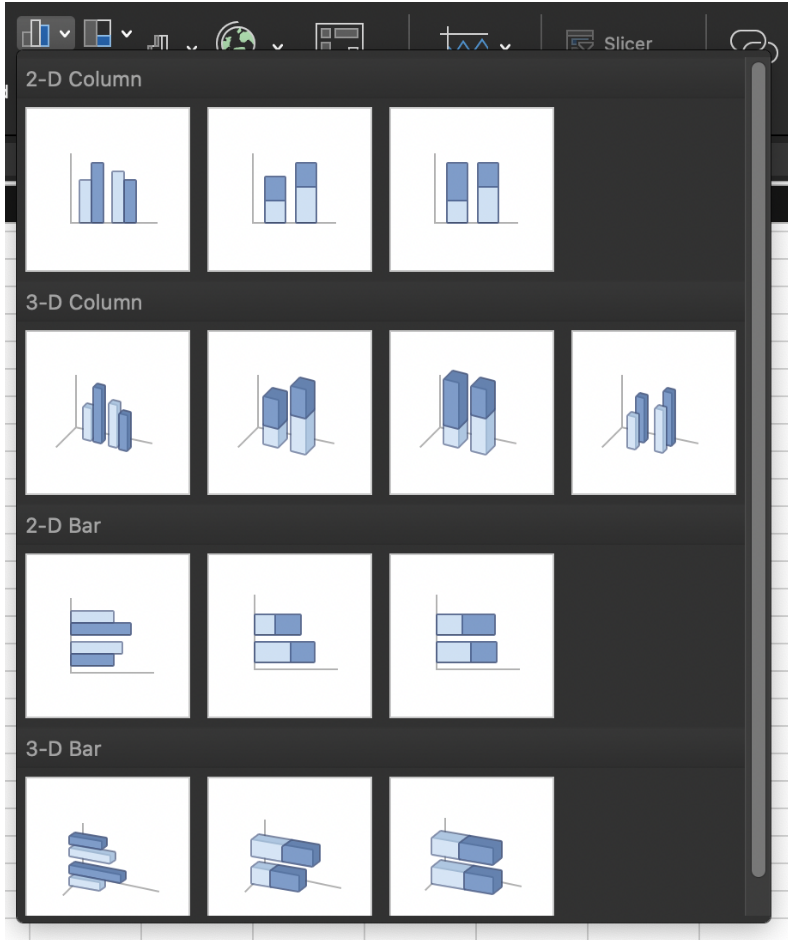
These were just a few examples of layers!

See the ggplot2 documentation for complete examples of everything you can do

A true grammar

With the grammar of graphics, we don't talk about specific chart *types*

Hunt through Excel menus for a stacked bar chart and manually reshape your data to work with it

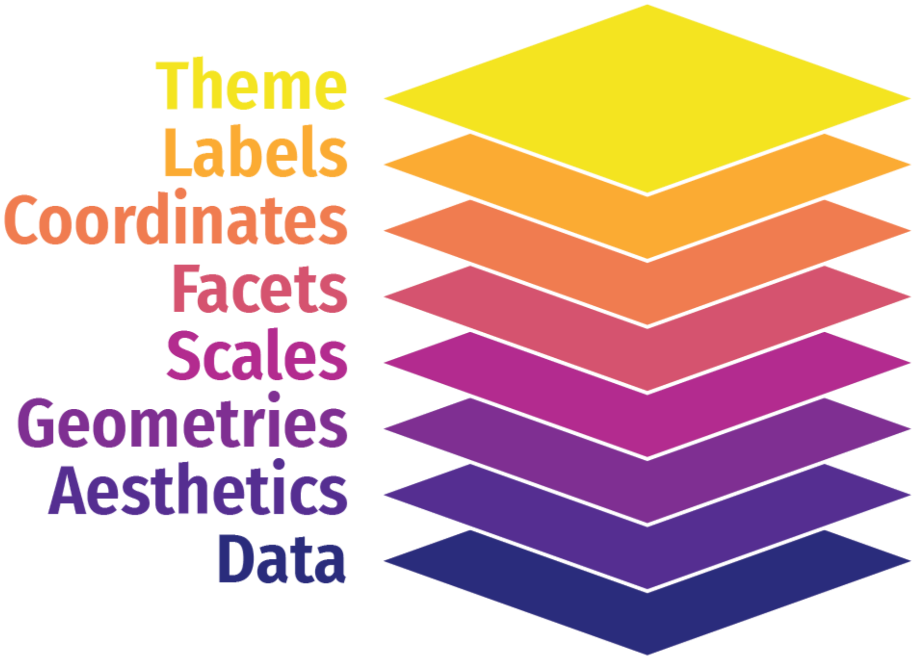


A true grammar

With the grammar of graphics, we *do* talk about specific chart *elements*

Map a column to the x-axis, fill by a different variable, and `geom_col()` to get stacked bars

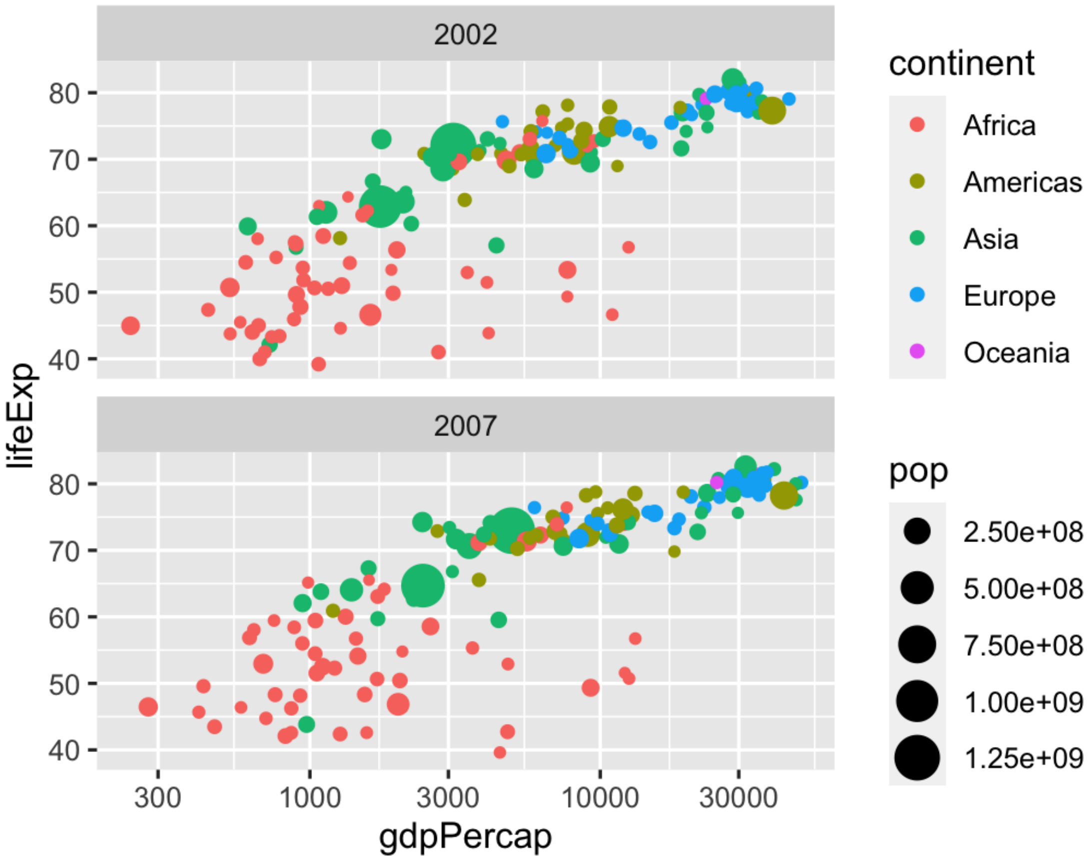
Geoms can be interchangable (e.g. switch `geom_violin()` to `geom_boxplot()`)



Describing graphs with the grammar

Map wealth to the x-axis, health to the y-axis, add points, color by continent, size by population, scale the y-axis with a log, and facet by year

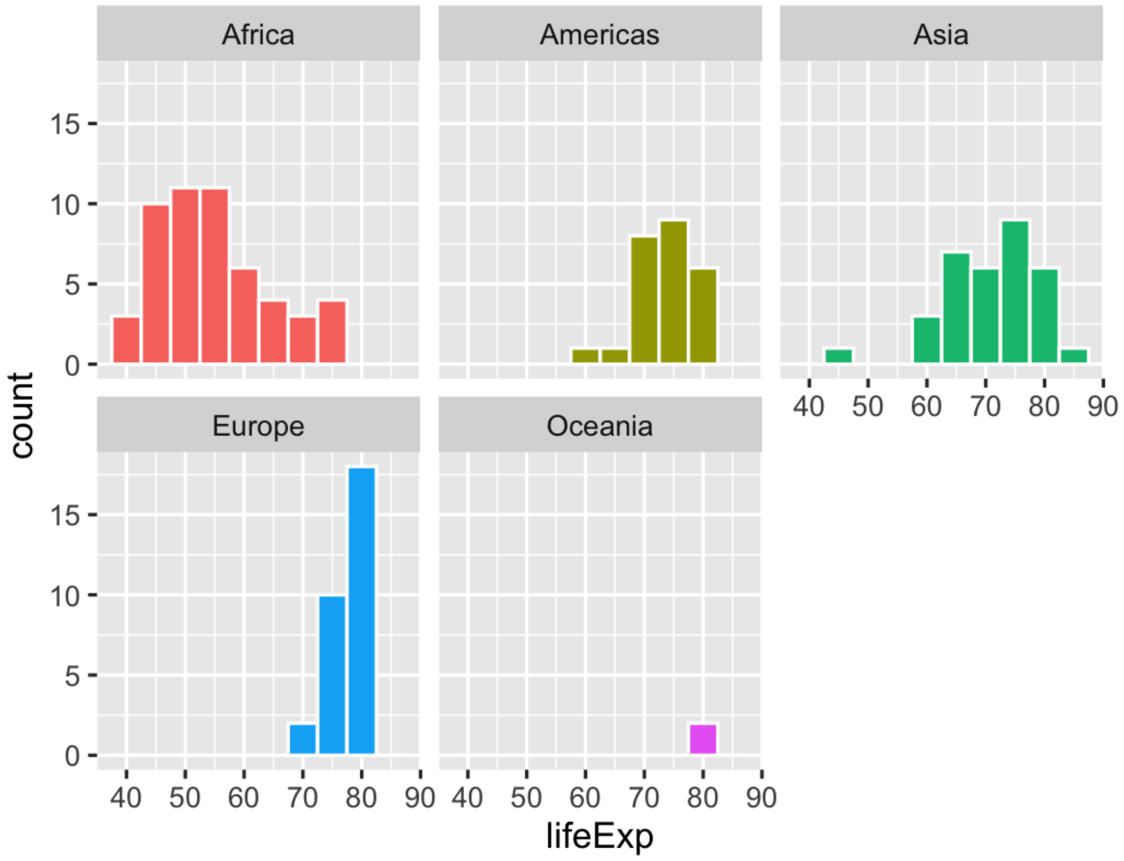
```
ggplot(filter(gapminder,
              year %in% c(2002, 2007)),
       aes(x = gdpPercap,
           y = lifeExp,
           color = continent,
           size = pop)) +
  geom_point() +
  scale_x_log10() +
  facet_wrap(vars(year), ncol = 1)
```



Describing graphs with the grammar

Map health to the x-axis, add a histogram with bins for every 5 years, fill and facet by continent

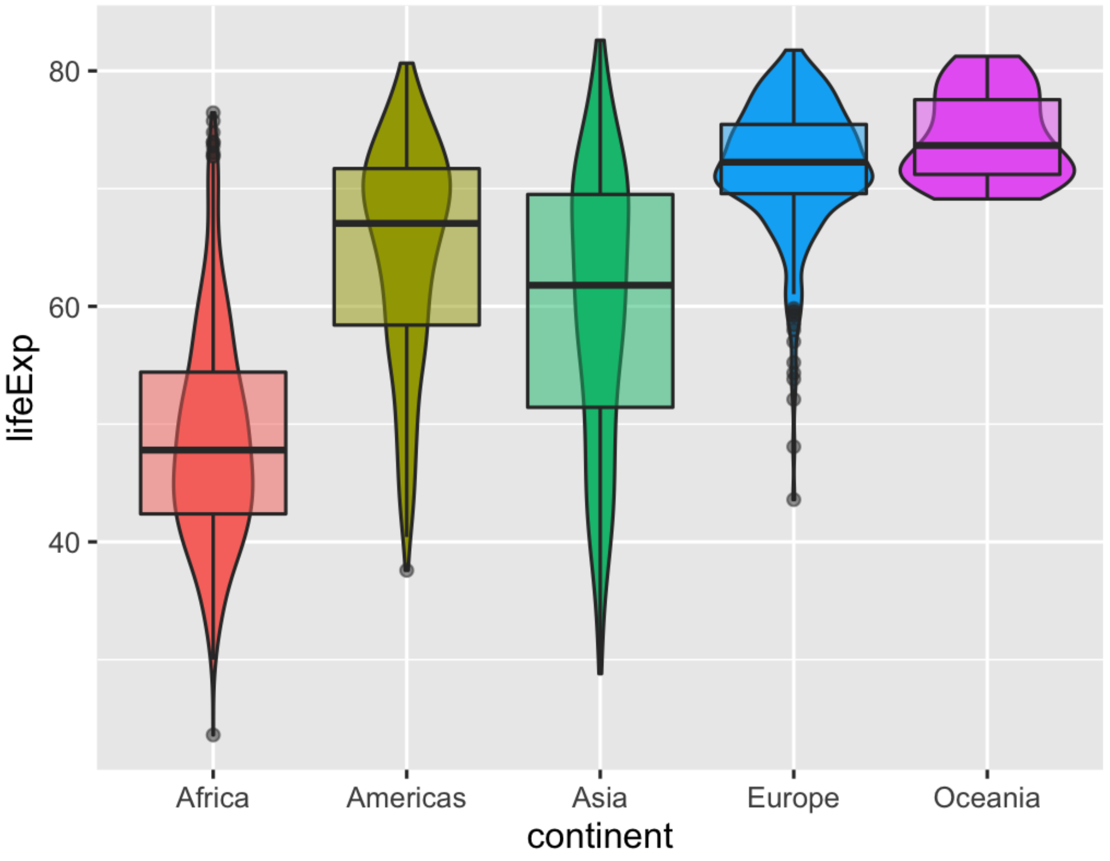
```
ggplot(gapminder_2007,
       aes(x = lifeExp,
           fill = continent)) +
  geom_histogram(binwidth = 5,
                 color = "white") +
  guides(fill = FALSE) +  # Turn off legend
  facet_wrap(vars(continent))
```



Describing graphs with the grammar

Map continent to the x-axis, health to the y-axis, add violin plots and semi-transparent boxplots, fill by continent

```
ggplot(gapminder,
       aes(x = continent,
           y = lifeExp,
           fill = continent)) +
  geom_violin() +
  geom_boxplot(alpha = 0.5) +
  guides(fill = FALSE)  # Turn off legend
```



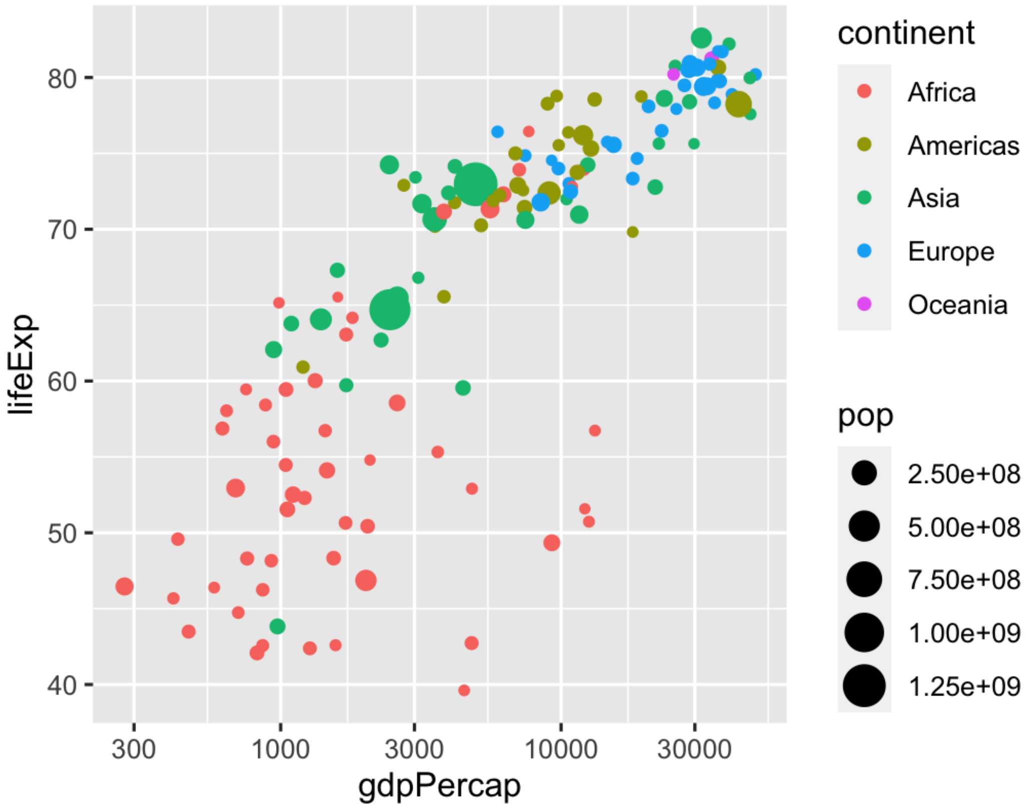
Scales

Scales change the properties of the variable mapping

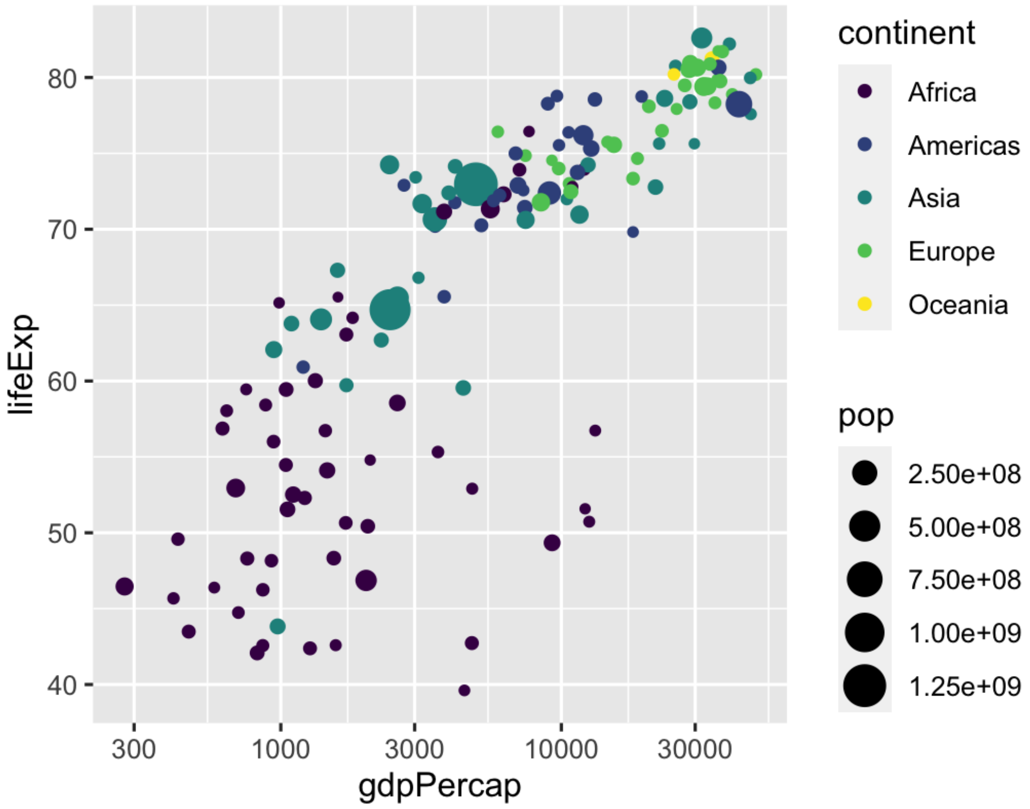
Example layer	What it does
`scale_x_continuous()`	Make the x-axis continuous
`scale_x_continuous(breaks = 1:5)`	Manually specify axis ticks
`scale_x_log10()`	Log the x-axis
`scale_color_gradient()`	Use a gradient
`scale_fill_viridis_d()`	Fill with discrete viridis colors

Scales

`scale_x_log10()`



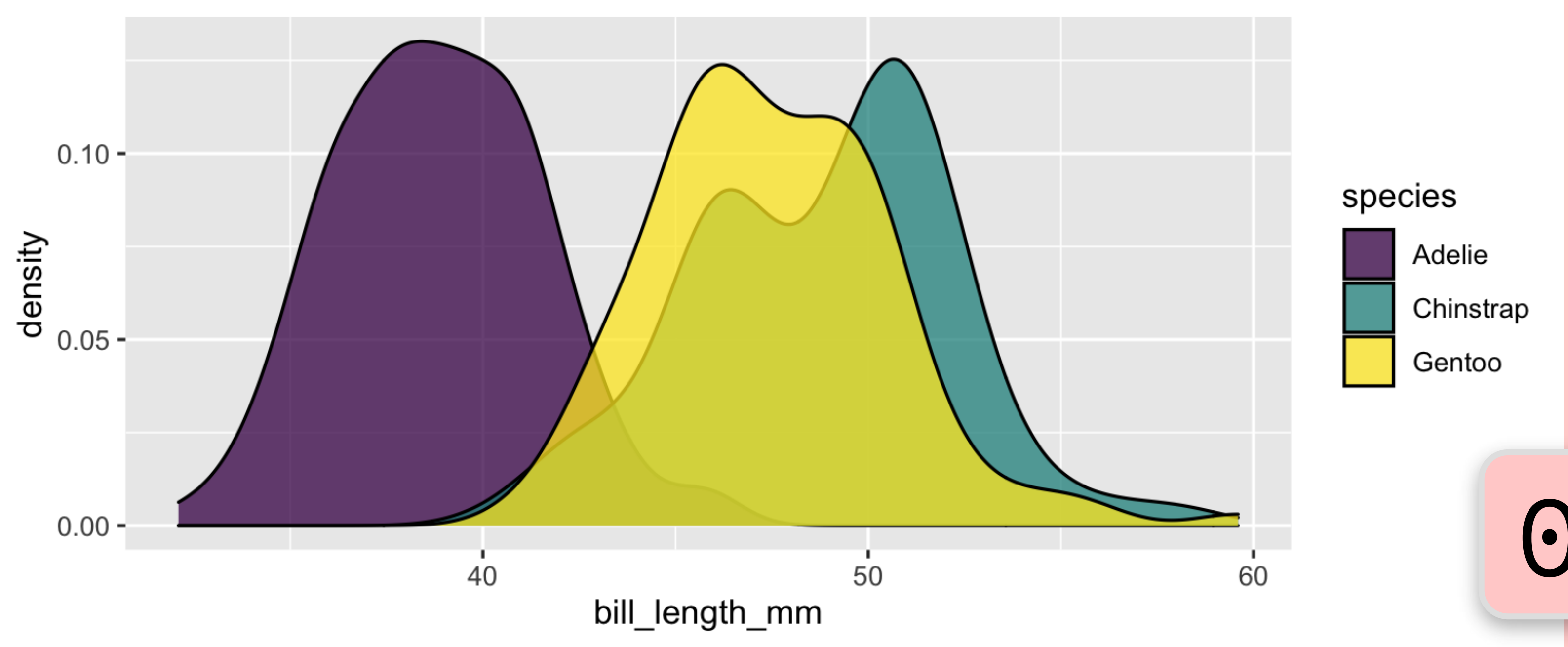
`scale_color_viridis_d()`



Your turn #7

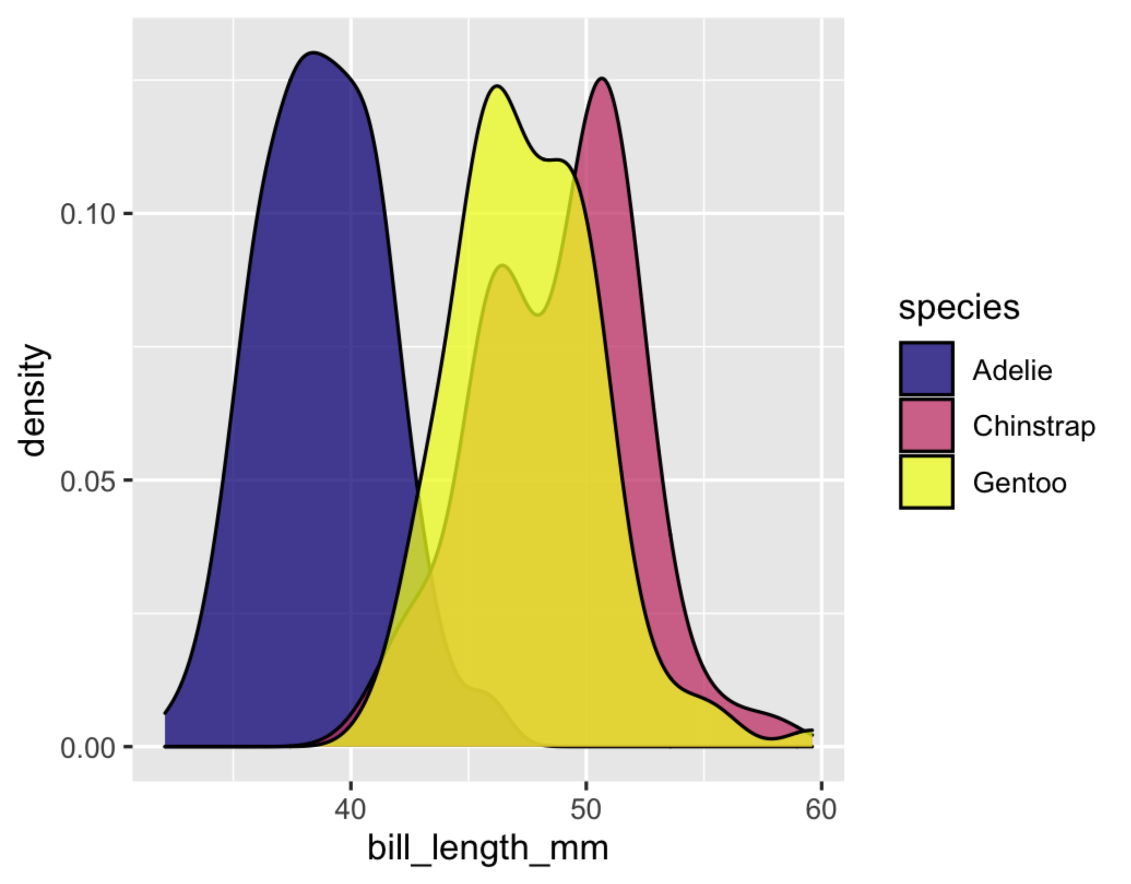
**Make this density plot of `bill_length_mm` filled by `species`.
Use the viridis fill scale.**

For bonus fun, try a different viridis option like `plasma` or `inferno`.



03:00

```r
ggplot(penguins,
       aes(x = bill_length_mm,
           fill = species)) +
  geom_density(alpha = 0.75) +
  scale_fill_viridis_d(option = "plasma")
```



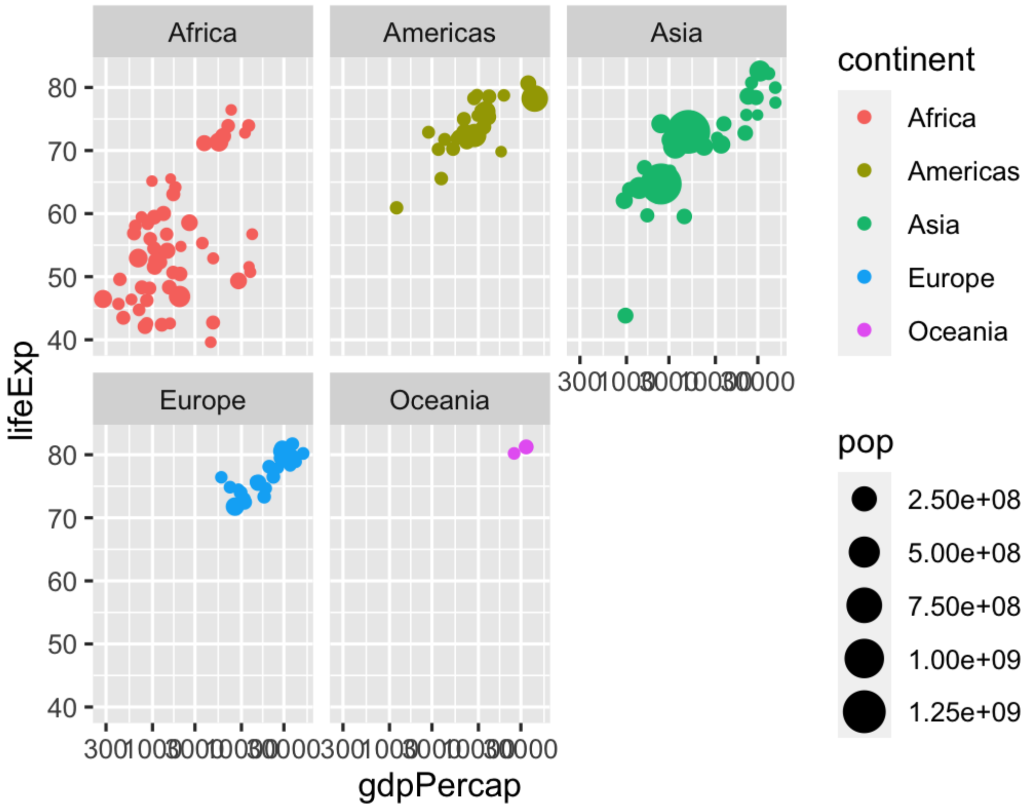
Facets

Facets show subplots for different subsets of data

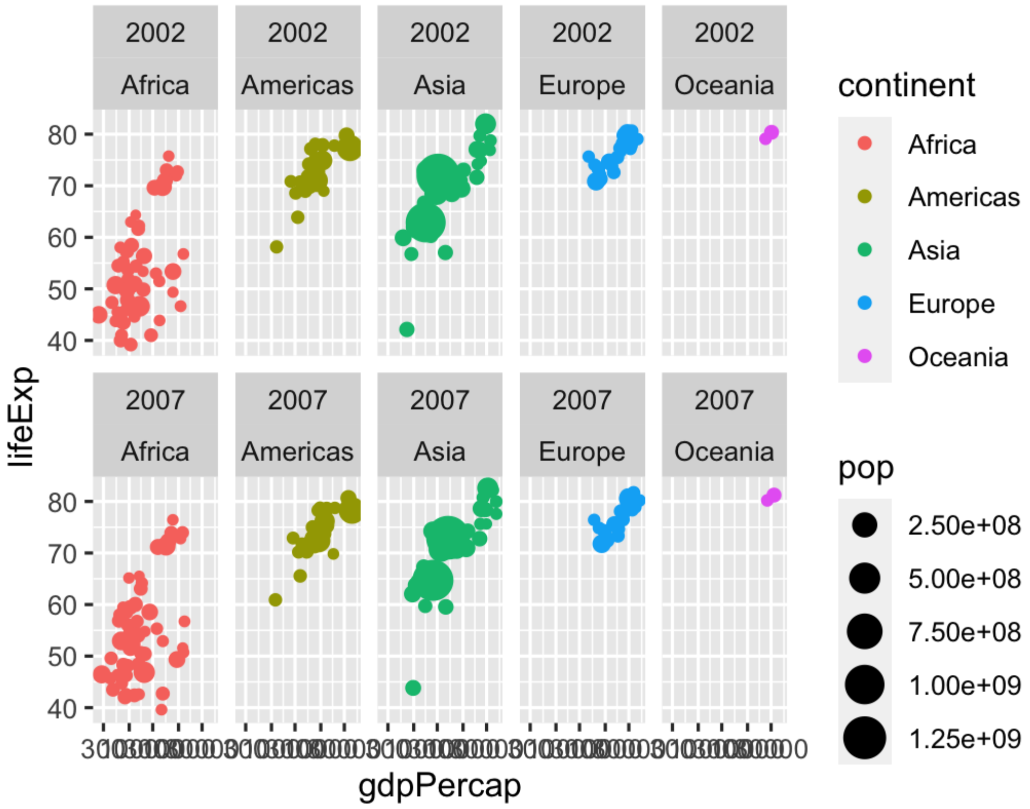
Example layer	What it does
`facet_wrap(vars(continent))`	Plot for each continent
`facet_wrap(vars(continent, year))`	Plot for each continent/year
`facet_wrap(..., ncol = 1)`	Put all facets in one column
`facet_wrap(..., nrow = 1)`	Put all facets in one row

Facets

`facet_wrap(vars(continent))`

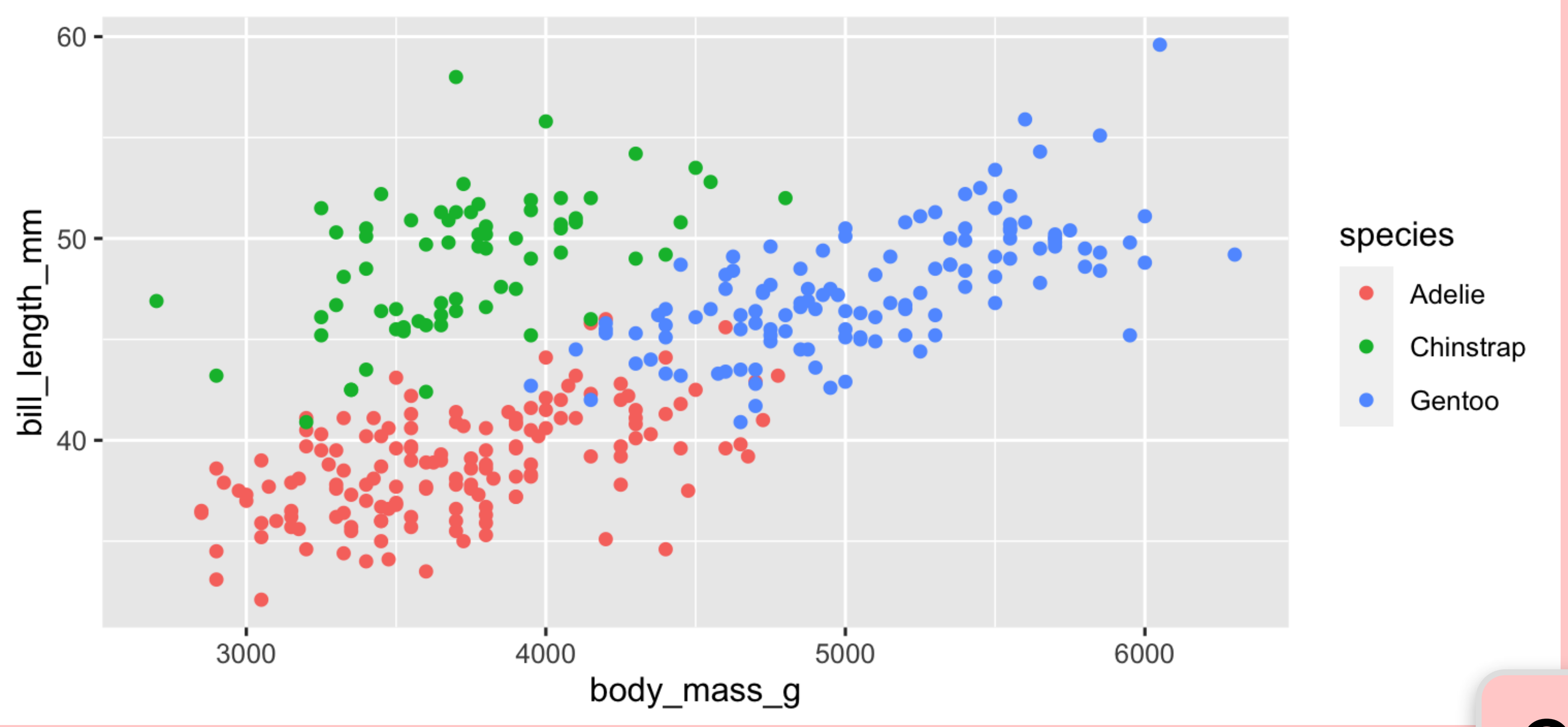


`facet_wrap(vars(continent, year))`



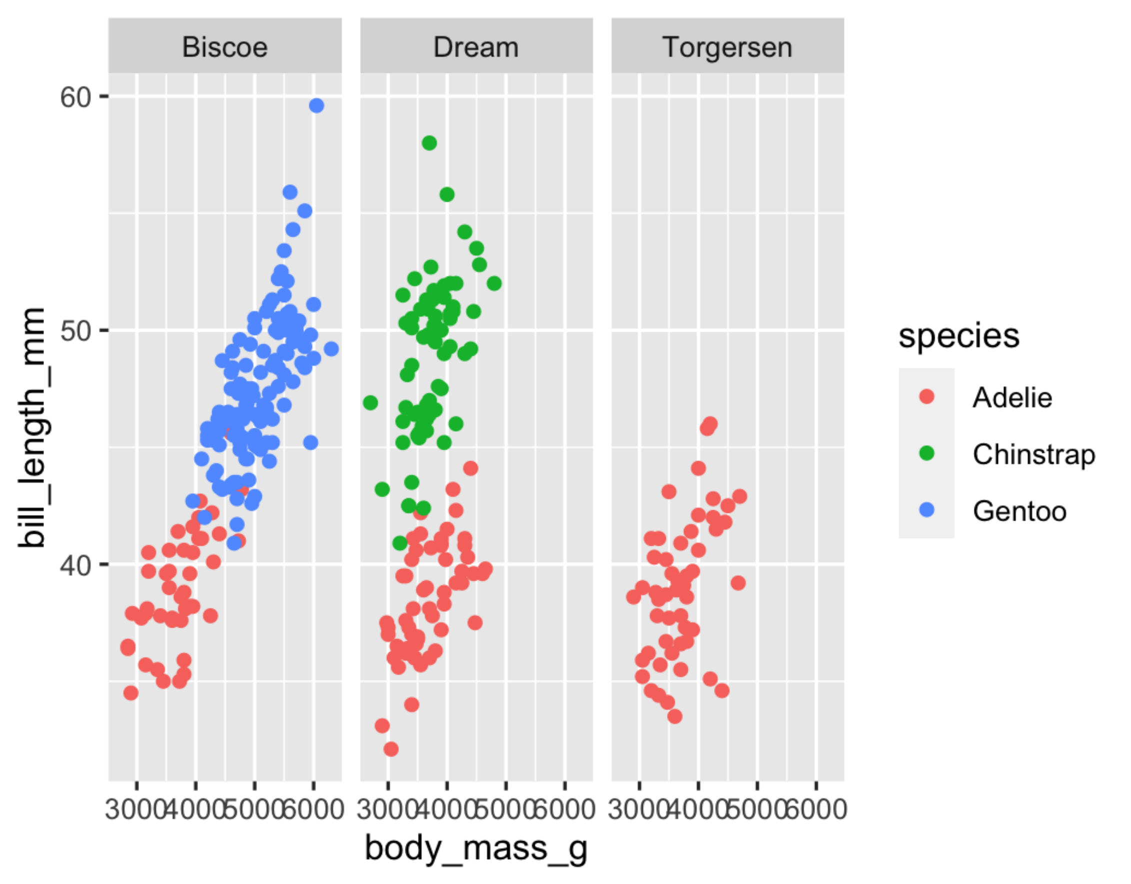
Your turn #8

Facet this scatterplot by `island`. Are there any interesting trends?



03:00

```
ggplot(penguins,
       aes(x = body_mass_g,
           y = bill_length_mm,
           color = species)) +
  geom_point() +
  facet_wrap(vars(island))
```



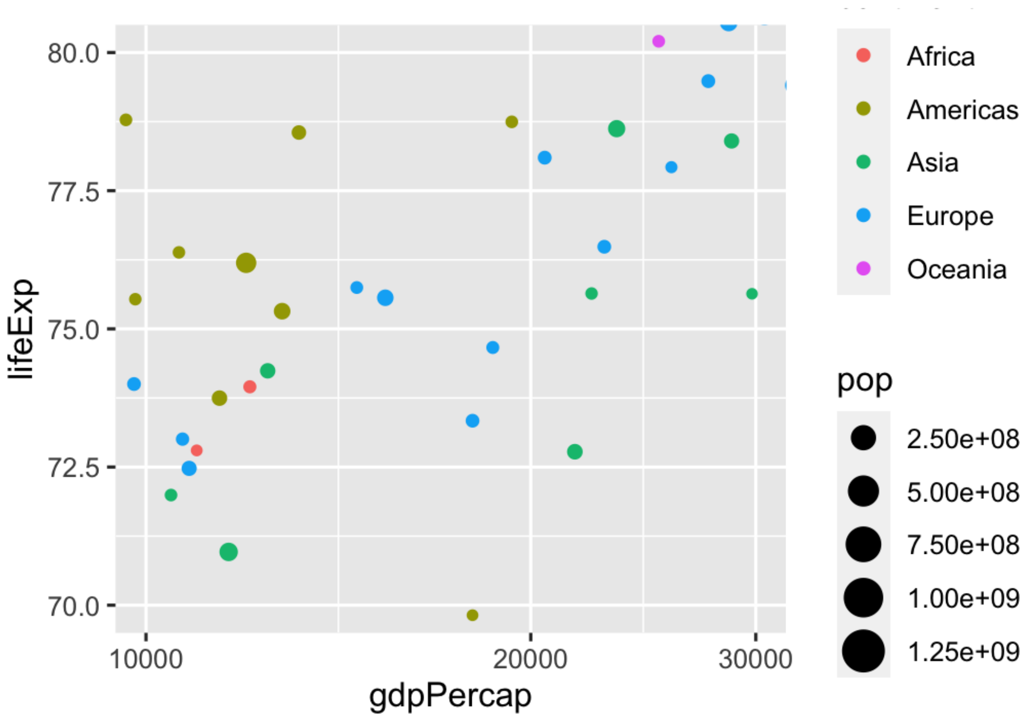
Coordinates

Change the coordinate system

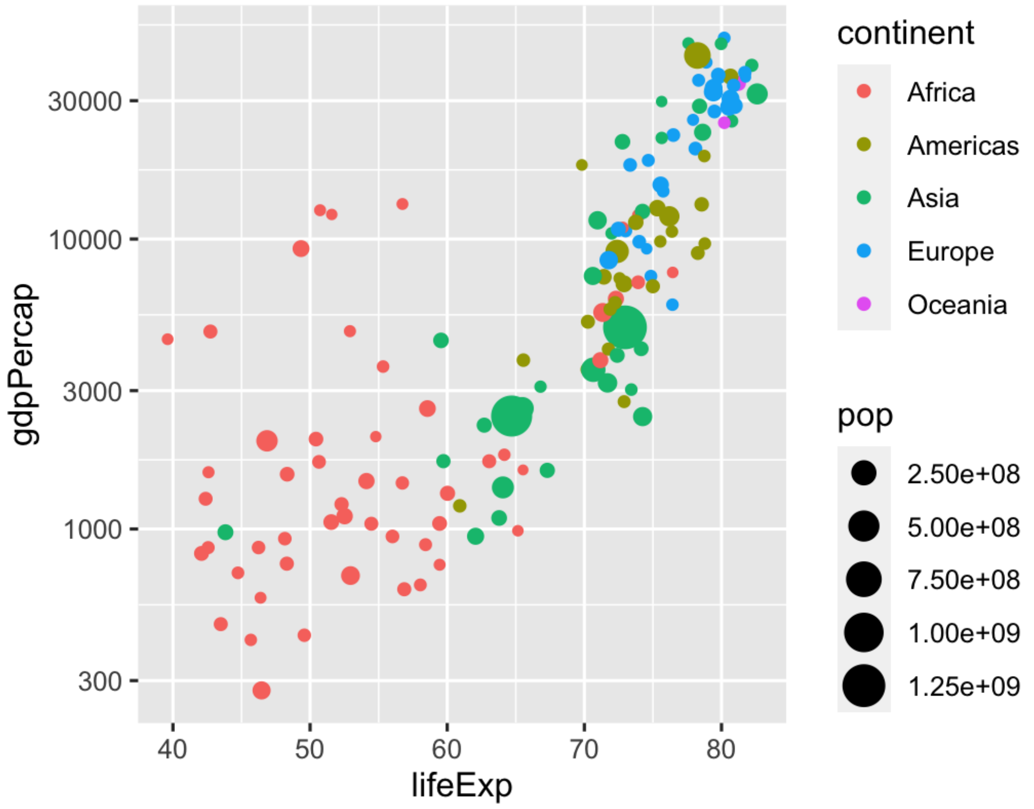
Example layer	What it does
`coord_cartesian()`	Standard x-y coordinate system
`coord_cartesian(ylim = c(1, 10))`	Zoom in where y is 1–10
`coord_flip()`	Switch x and y
`coord_polar()`	Use circular polar system

Coordinates

`coord_cartesian(ylim = c(70, 80), xlim = c(10000, 30000))`



`coord_flip()`



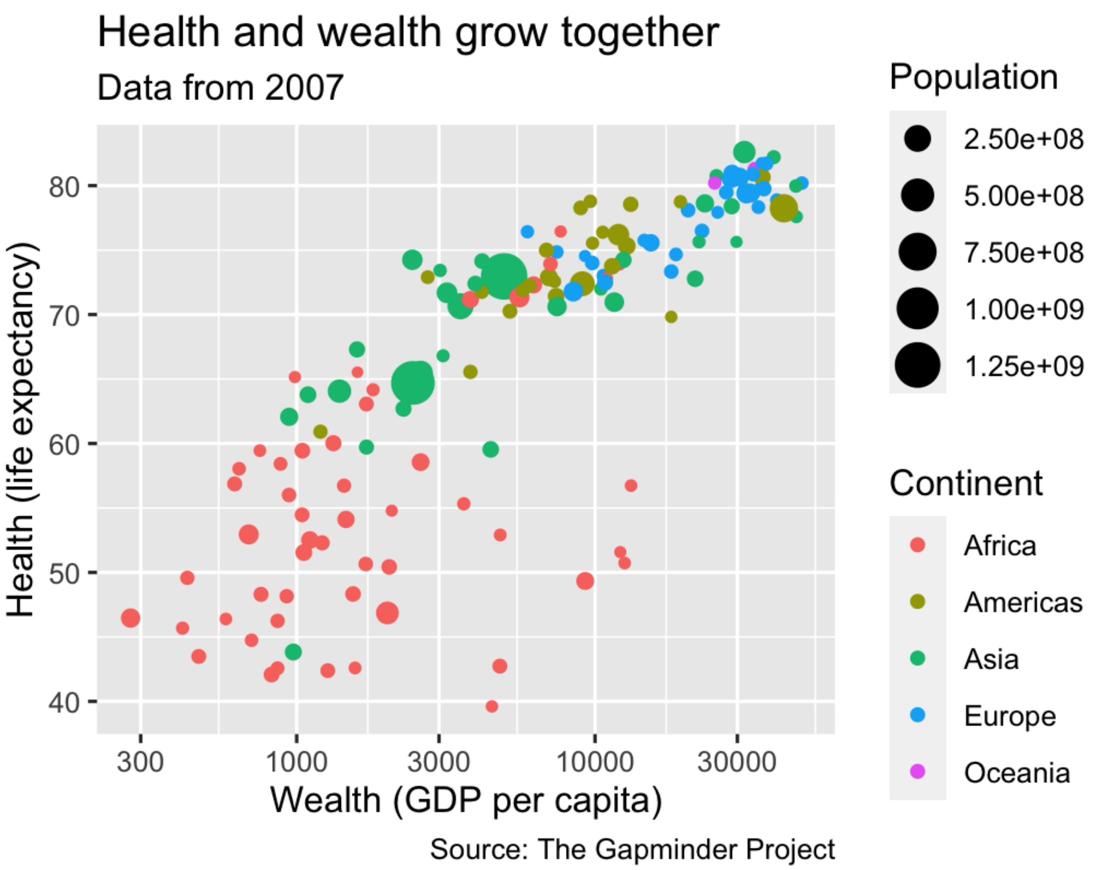
Labels

Add labels to the plot with a single `labs()` layer

Example layer	What it does
`labs(title = "Neat title")`	Title
`labs(caption = "Something")`	Caption
`labs(y = "Something")`	y-axis
`labs(size = "Population")`	Title of size legend

Labels

```
ggplot(gapminder_2007,
       aes(x = gdpPercap, y = lifeExp,
           color = continent, size = pop)) +
  geom_point() +
  scale_x_log10() +
  labs(title = "Health and wealth grow togeth
       subtitle = "Data from 2007",
       x = "Wealth (GDP per capita)",
       y = "Health (life expectancy)",
       color = "Continent",
       size = "Population",
       caption = "Source: The Gapminder Proje
```



Theme

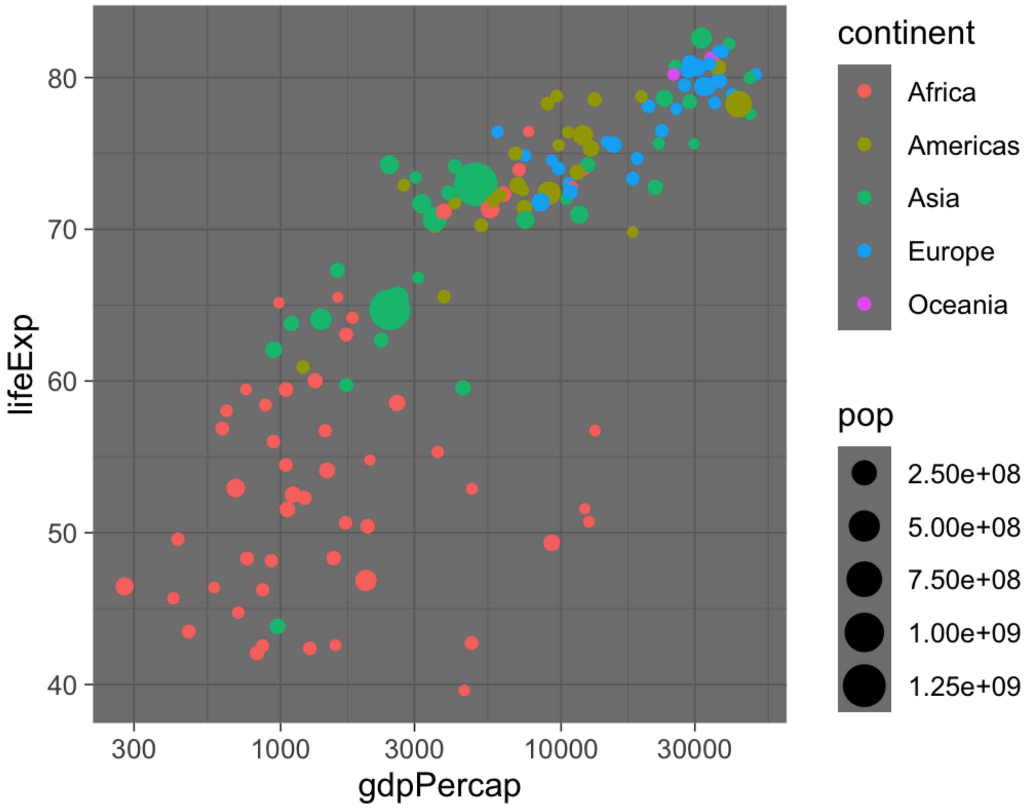
Change the appearance of anything in the plot

There are many built-in themes

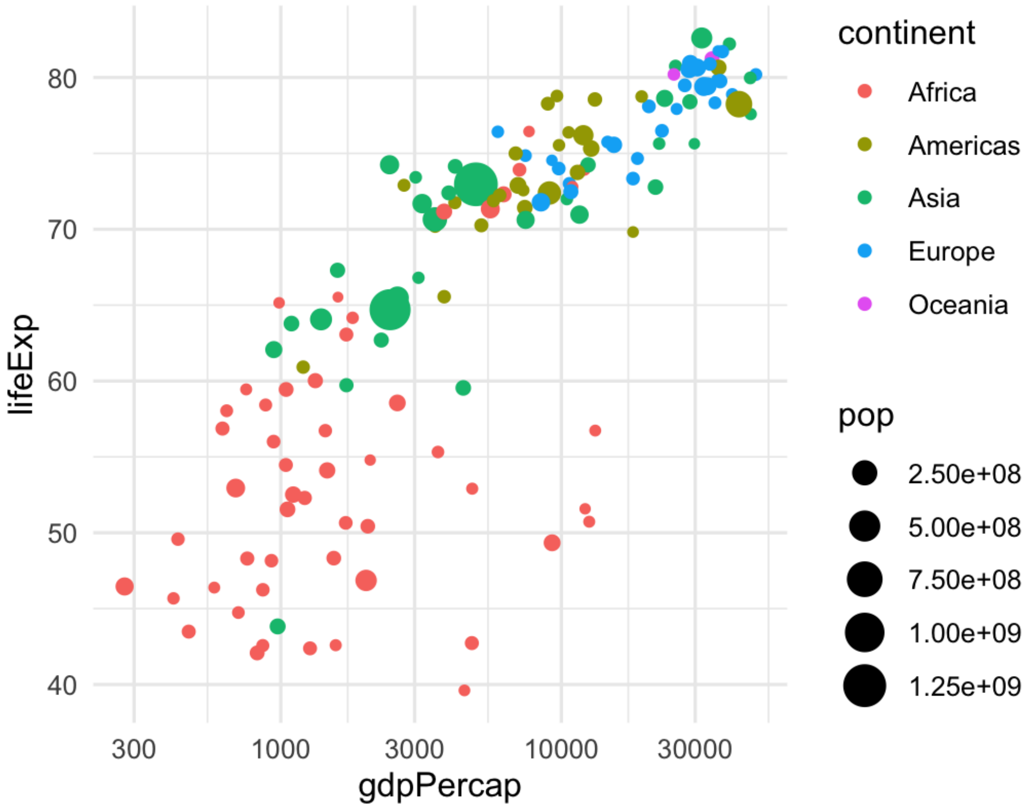
Example layer	What it does
`theme_grey()`	Default grey background
`theme_bw()`	Black and white
`theme_dark()`	Dark
`theme_minimal()`	Minimal

Theme

`theme_dark()`

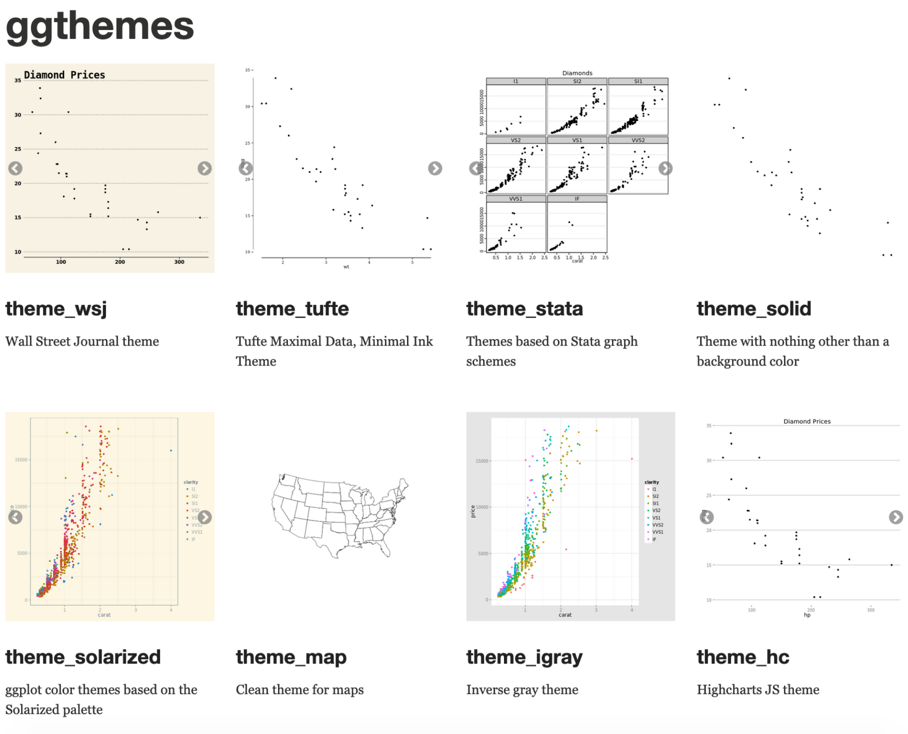


`theme_minimal()`



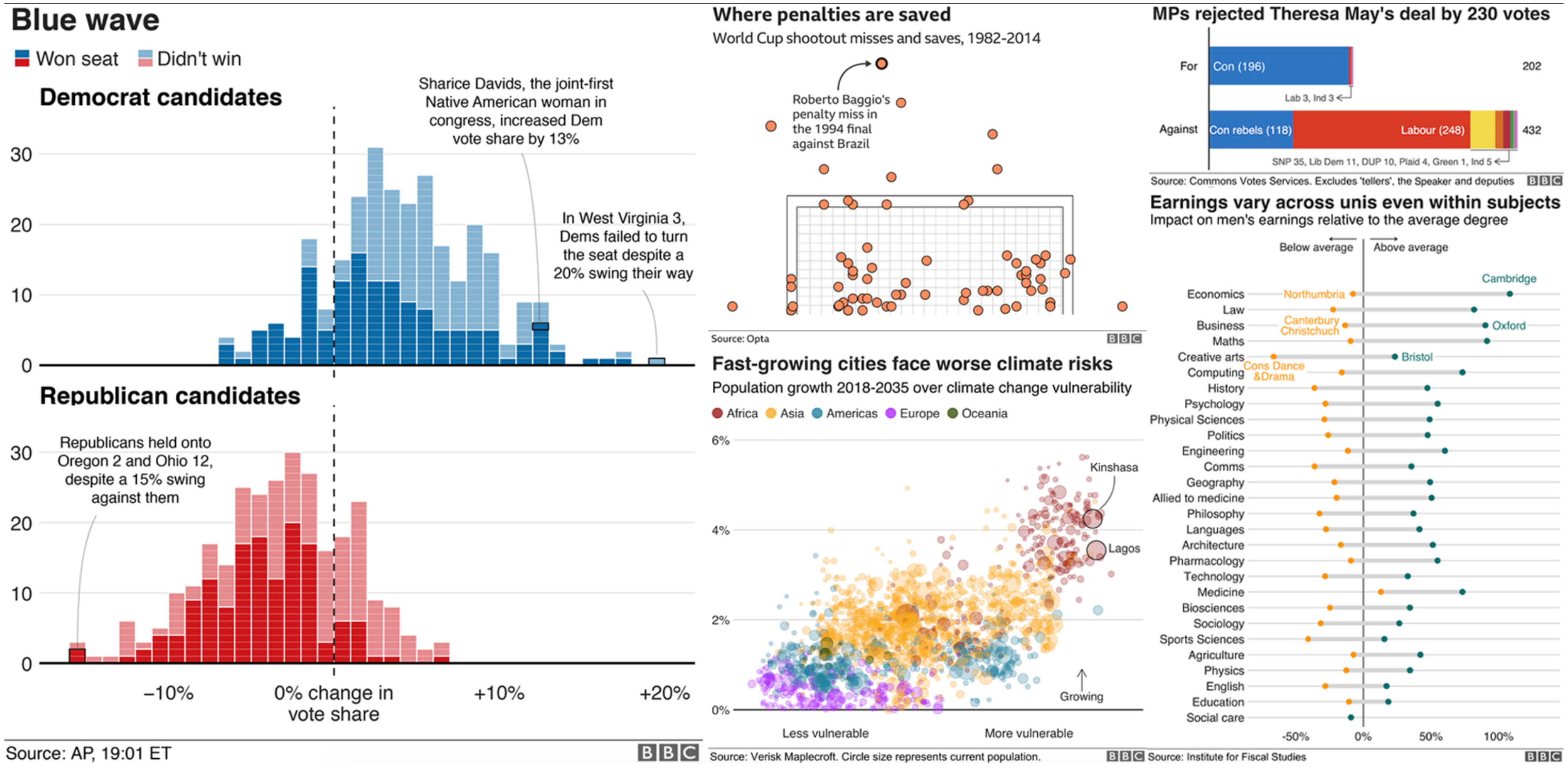
Theme

There are collections of pre-built themes online, like the ggthemes package



Theme

Organizations often make their own custom themes, like the BBC



Theme options

Make theme adjustments with `theme()`

There are a billion options here!

```
theme_bw() +
theme(legend.position = "bottom",
      plot.title = element_text(face = "bold"),
      panel.grid = element_blank(),
      axis.title.y = element_text(face = "italic"))
```

Saving graphs

Use `ggsave()` to save a plot to your computer

Store plot as an object, feed it to `ggsave()`

```
my_plot <- ggplot(...)

ggsave("plot_name.pdf", my_plot, width = 5, height = 3.5)
ggsave("plot_name.png", my_plot, width = 5, height = 3.5)
```