Lecture 1

Penguins visualisation

Misja Mikkers and Gertjan Verhoeven

Introduction

The convention is to put the ticks " around R commands. In Lecture 5 you will find out why this is the convention.

Packages

For this notebook we need the tidyverse package (which should already be installed) and the `palmerpenguins` package.

If you want to use a specific package (e.g. `tidyverse`) you can use the command `library(tidyverse)`. `tidyverse` is the package we use for data wrangling (lectures 2 and 3) and visualization (this lecture).

To be able to visualize something, we also need data. For this notebook we will make some plots of penguins. The data are stored as well in a package.

If you type `library(palmerpenguins)`, you can access the data. If you run this command R may return the following error message

"Error in library(palmerpenguins) : there is no package called 'palmerpenguins' "

This error message means that you did not install the package `palmerpenguins`. You can install the package by using the tab "Packages" at the middle of the right side of the screen. Then push the "Install" tab and find the package `palmerpenguins`.

```
# Your code here
library(palmerpenguins)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.0.5     v dplyr   1.0.3
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Data

We can now access the data by typing the command: `data("penguins")`. At the top right of your screen under the tab "Environment" you will see 2 datasets

- penguins (with 344 observations of 8 variabbles)
- penguins_raw (with 344 observations of 17 variabbles)

We will only use the first data set (penguins). You can have a look at the dataset by double clicking on the penguins dataset in the Environment or by using the commands

- `head(penguins)` which gives you the top rows of the dataset
- `tail(penguins)` which gives you the bottom rows of the dataset
- `summary(penguins)` which gives you a summary table

```
data("penguins")
```

```
summary(penguins)
```

```
##       species          island    bill_length_mm  bill_depth_mm
##  Adelie   :152   Biscoe   :168   Min.   :32.10   Min.   :13.10
##  Chinstrap: 68   Dream    :124   1st Qu.:39.23   1st Qu.:15.60
##  Gentoo   :124   Torgersen: 52   Median :44.45   Median :17.30
##                                  Mean   :43.92   Mean   :17.15
##                                  3rd Qu.:48.50   3rd Qu.:18.70
##                                  Max.   :59.60   Max.   :21.50
##                                  NA's   :2       NA's   :2
##  flipper_length_mm  body_mass_g       sex           year
##  Min.   :172.0     Min.   :2700   female:165   Min.   :2007
##  1st Qu.:190.0     1st Qu.:3550   male  :168   1st Qu.:2007
##  Median :197.0     Median :4050   NA's  : 11   Median :2008
##  Mean   :200.9     Mean   :4202                Mean   :2008
##  3rd Qu.:213.0     3rd Qu.:4750                3rd Qu.:2009
##  Max.   :231.0     Max.   :6300                Max.   :2009
##  NA's   :2         NA's   :2
```

The dataset contains 8 variables about 344 individual penguins. If you want to know what all variables mean you can use the command `?penguins` and an description of this dataset will appear at the bottom right of your screen.

```
# Your code here to explore the penguins dataset
?penguins
```

```
## starting httpd help server ... done
```

There are some "NA"'s in the dataset by running the code

```
penguins <- penguins %>%   na.omit
```

we get rid of the NA's. You should recognize the "pipe operator" `%>%` from the datacamp course "Introduction to the Tidyverse".

```r
# Your code here
penguins <- penguins %>%
  na.omit
```

As you may notice in the top right of your screen the penguins dataset now contains 333 observations of 8 variables. We lost 11 penguins.

Introduction to plotting with ggplot

In 2005 Hadley Wickham created ggplot as part of his PhD. `ggplot` is a part of the tidyverse package and allows you to plot anything you can imagine.

Any plot in ggplot has (at least) the following parts

- the command `ggplot()` which tells R to plot.
- a `geom_something` that tells ggplot which type of graph should be created. Examples are
  - geom_point() for a scatterplot
  - geom_line() for a line graph
  - geom_bar() for a bar plot
  - geom_histogram() for a histogram
  - geom_boxplot()
  - etc
- We need to tell R where to look for the data.
- We need to specify what to do with the variables. `ggplot` makes it possible to link a certain *aesthetic* of a graph to a column in a dataframe. Examples of are *Aesthetics* :
  - x-coordinate
  - y-coordinate
  - size and shape of points
  - color fill
  - line color
  - line size
  - transparency (alpha)
  - line type
  - etc

In this notebook we will use the convention to specify the data and aesthetics in the `geom_something()` command. It is also possible to do this in the `ggplot()` command. Both work.

However, if you need 2 datasets for 2 different types of `geom_something()` (in other words you want to plot different layers), the convention we use here works better.

So our commands for plots look like:

```
ggplot() +  geom_something(data = some data, aes(x = some variable, y = some variable,
fill = some variable))
```

Scatterplots

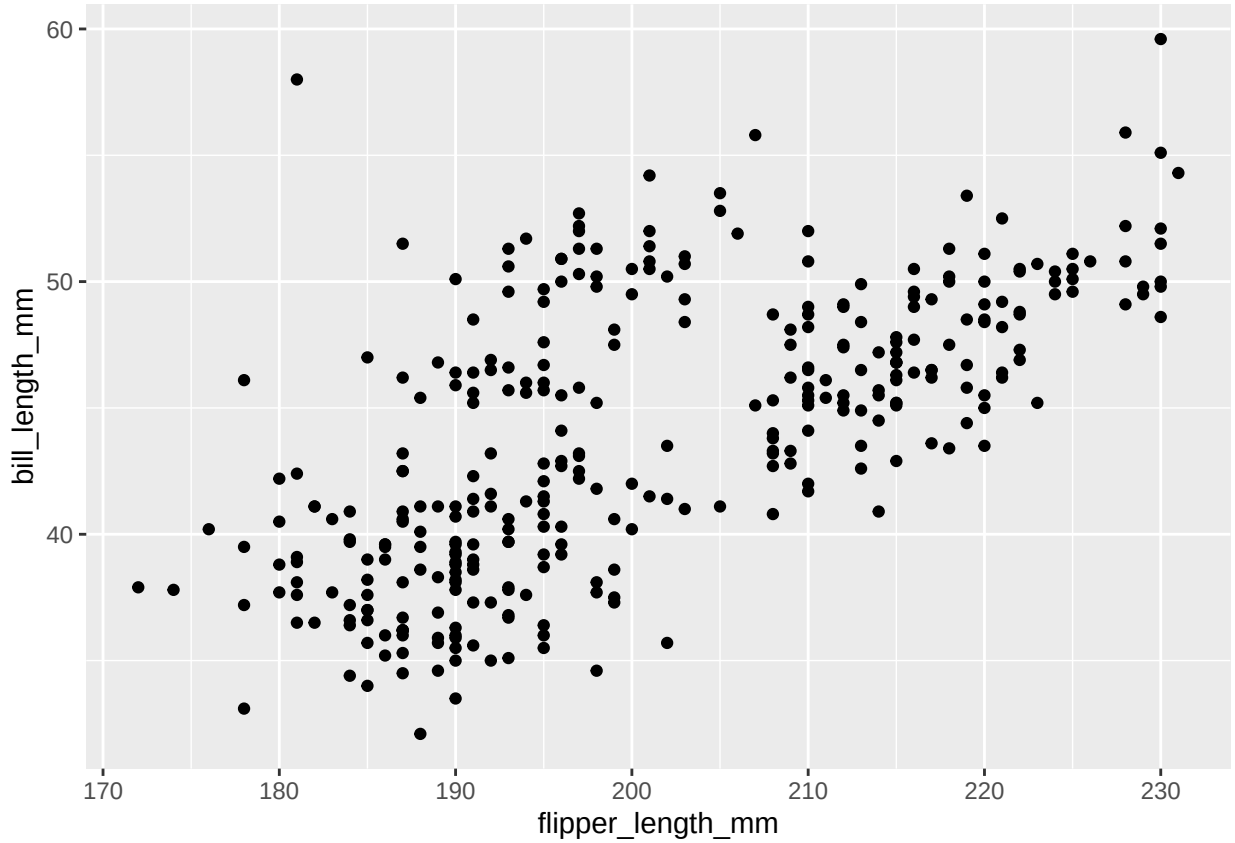
First plot

Suppose we are interested in the correlation between flipper length and bill length, because we assume that animals with a larger flipper will also have a larger bill.

In that case we can plot the data as a scatterplot (`geom_point()`)with

- data = penguins
- aesthetics (aes): x = flipper_length_mm and y = bill_length_mm

```
# Here your code
ggplot() +
  geom_point(data = penguins, aes(x = flipper_length_mm, y = bill_length_mm))
```



We have a already nice plot now.

Add some color

We could experiment a bit with adding colors and shapes to this plot by trying things like changing color, size and shape by for example adding `, color = "red", size = 3, shape = 1` after the `aes()` in `geom_point()`.

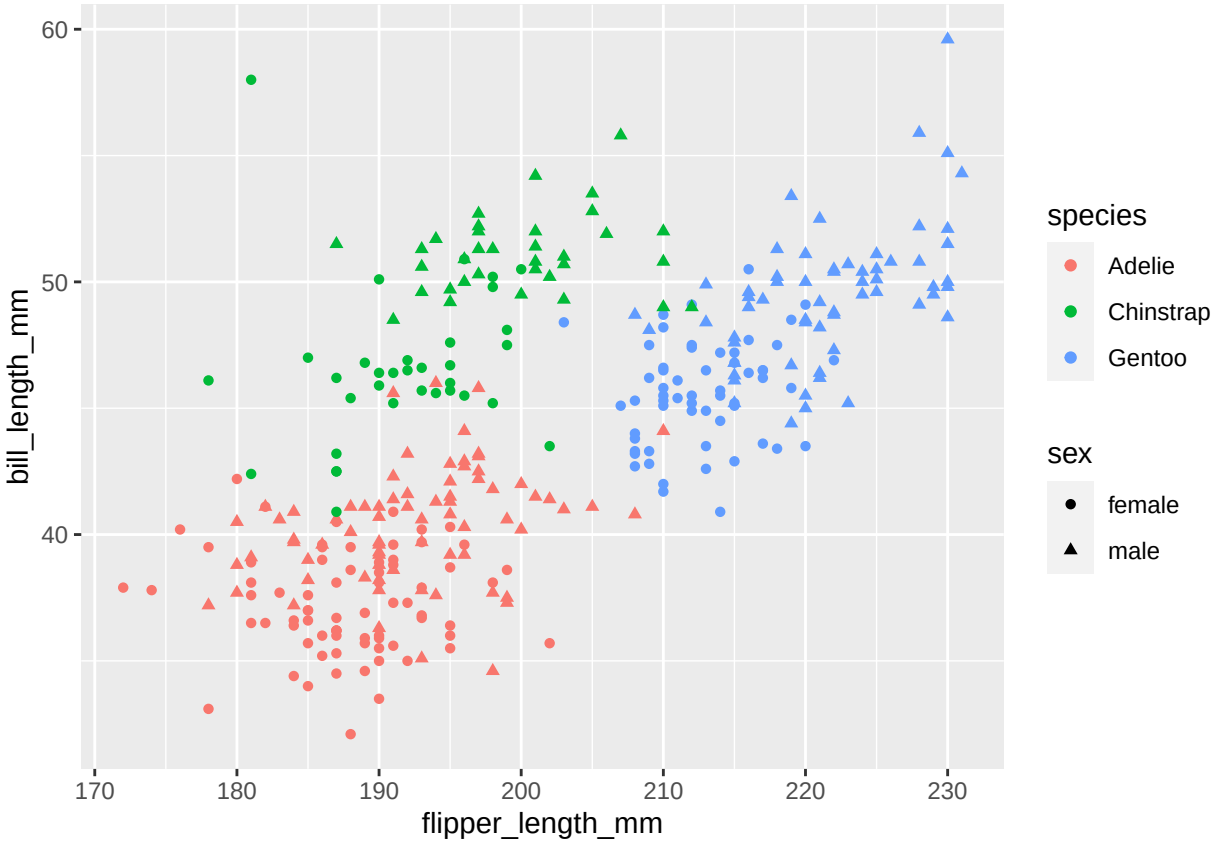
```
# Here your code
ggplot() +
  geom_point(data = penguins, aes(x = flipper_length_mm, y = bill_length_mm), color = "red", size = 3, s
```

Because we added color etc outside the `aes()` we apply the same color etc to all datapoints. Is does nothing with with variables.

Adding the dimensions of species and sex

It seems that there is a positive correlation between flipper length and bill length. However, we know that we have different species and sexes. In our next plot we will add an aesthetic "color" for species and "shape" for sex to our plot. We should do this inside the `aes()`, because now we want to make the color and the shape dependent on variables (species and sex.)

```
# Here your code
ggplot() +
  geom_point(data = penguins, aes(x = flipper_length_mm, y = bill_length_mm, color = species, shape = s
```



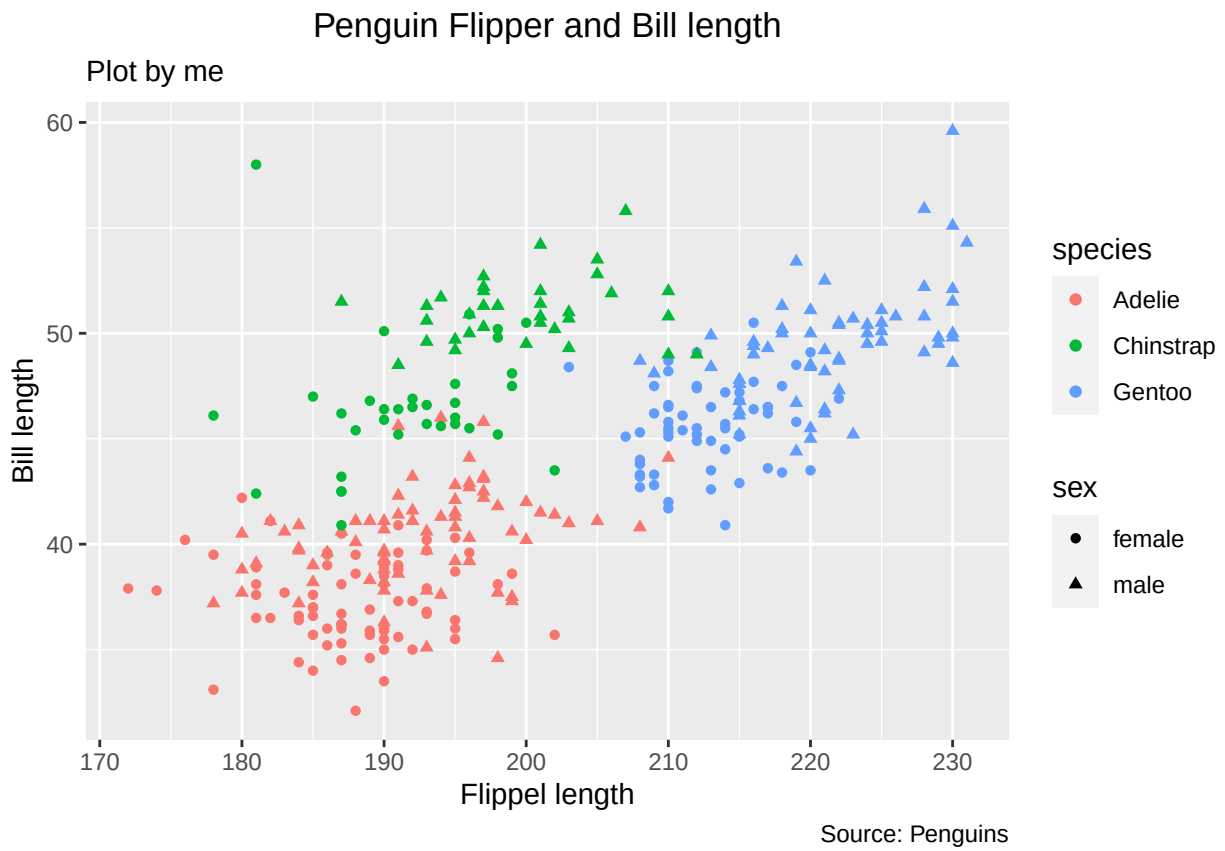
Make the plot nicer

We can make the plot nicer by adding labels (`labs()`) to the plot.

- adding a title (you can be creative)
- adding a subtitle (e.g. "plot by *your name*)
- adding a label to the x-axis (Flipper length)
- adding a label to the y-axis (Bill length)
- adding a caption (e.g. Source: Penguins)

Finally, we can add a theme (you can find some examples at https://ggplot2.tidyverse.org/reference/ggtheme.html). We will use `theme_bw()`

```
# Here the function theme was added to center the text. This was optional
# Here your code
ggplot() +
  geom_point(data = penguins, aes(x = flipper_length_mm, y = bill_length_mm, shape = sex, color = speci
  labs(title="Penguin Flipper and Bill length", subtitle = "Plot by me", caption = "Source: Penguins") +
  theme(plot.title = element_text(hjust = 0.5)) +
  xlab("Flippel length") +
  ylab("Bill length")
```



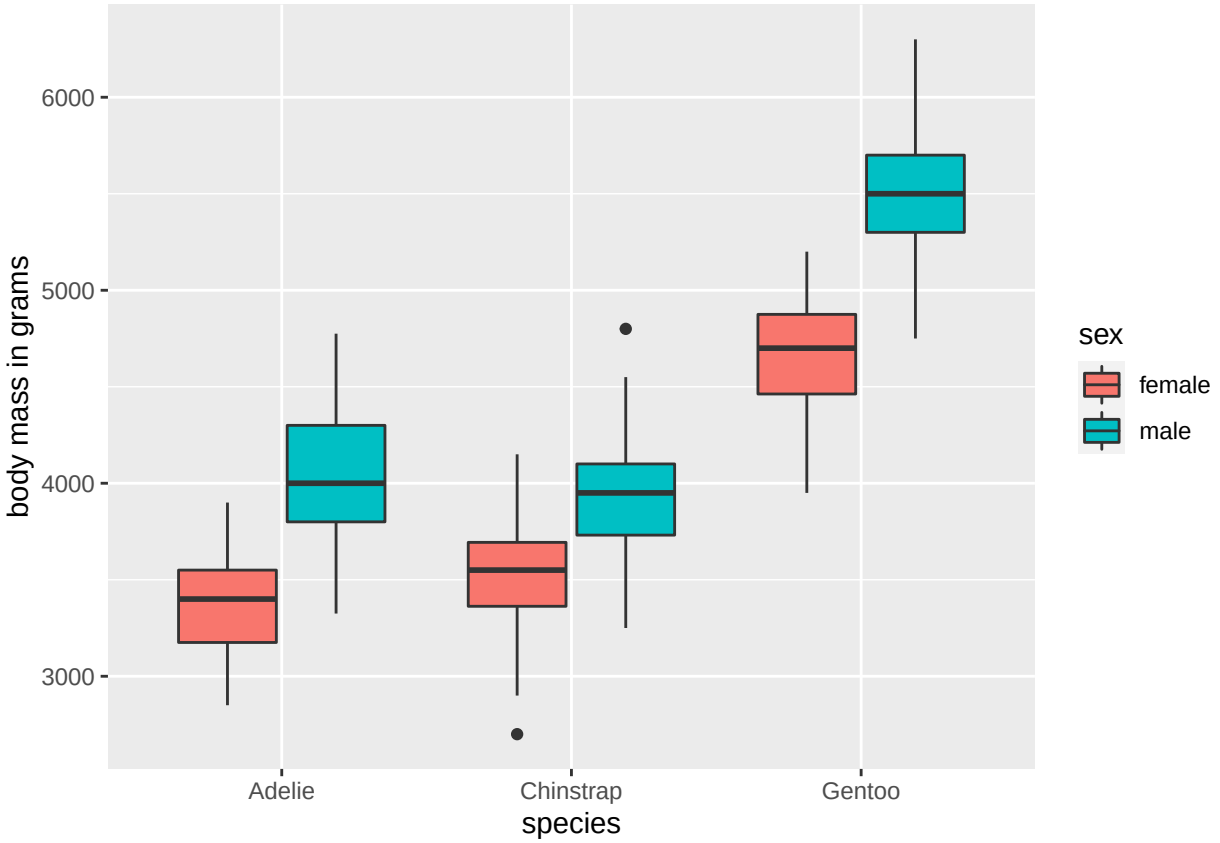
Boxplots

We could in more or less the same way make a boxplot. The assignment is:

To make a boxplot:

- with species on the x-axis
- with body_mass_g on the the y-axis
- with a fill for sex (so you will have different "boxes" for each sex)
- change the label on the y-axis to "body mass in grams")

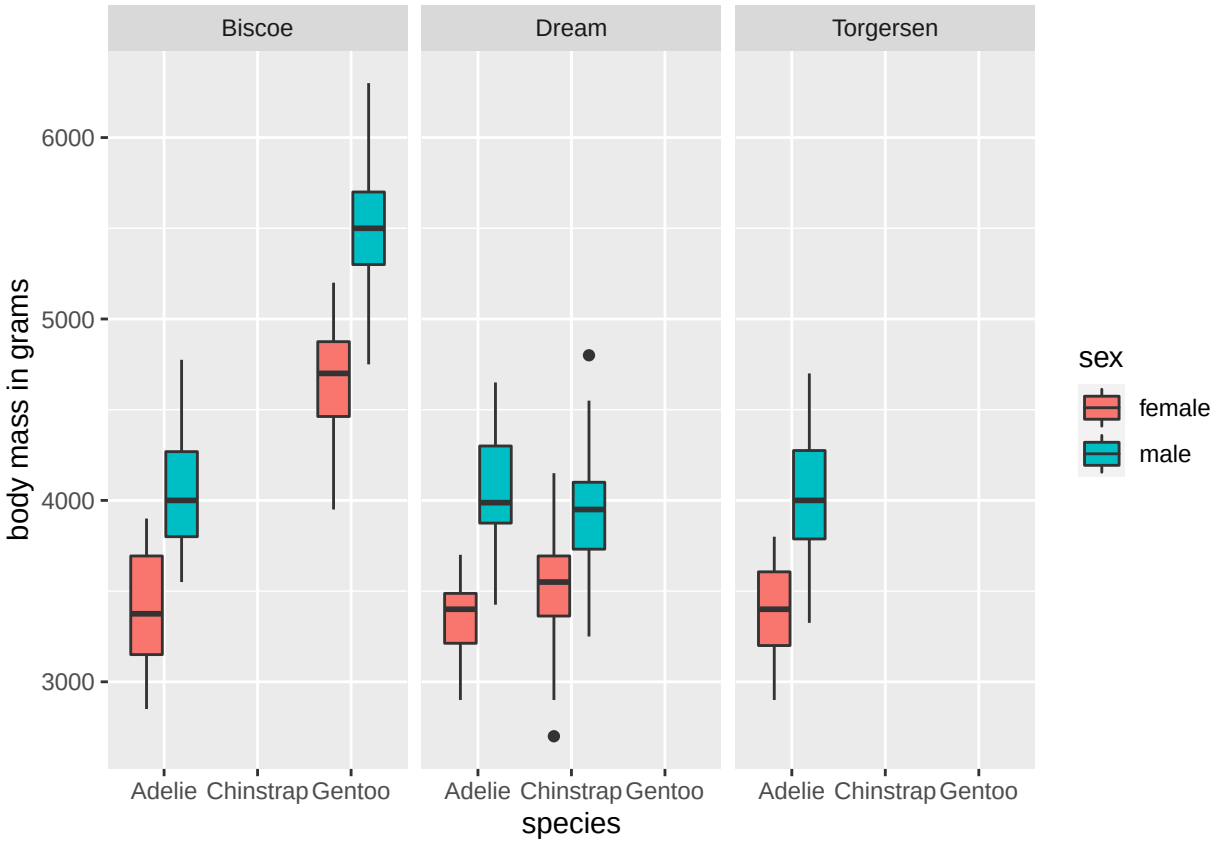
```
#Here your code for boxplot
ggplot() +
  geom_boxplot(data = penguins, aes(x = species, y = body_mass_g, fill = sex)) +
  ylab("body mass in grams")
```



We could also add another dimension by making a plot per island. With this plot we are able to see which species lives on which island and whether species living on different islands do have different body masses. We can split the plots per island by using for example

`facet_wrap(~ island)`.

```
# Here your code for separate graphs for islands
ggplot() +
  geom_boxplot(data = penguins, aes(x = species, y = body_mass_g, fill = sex)) +
  ylab("body mass in grams") +
  facet_wrap(~ island)
```



We can also change the colors of our variable sex.

Have a look at the following website

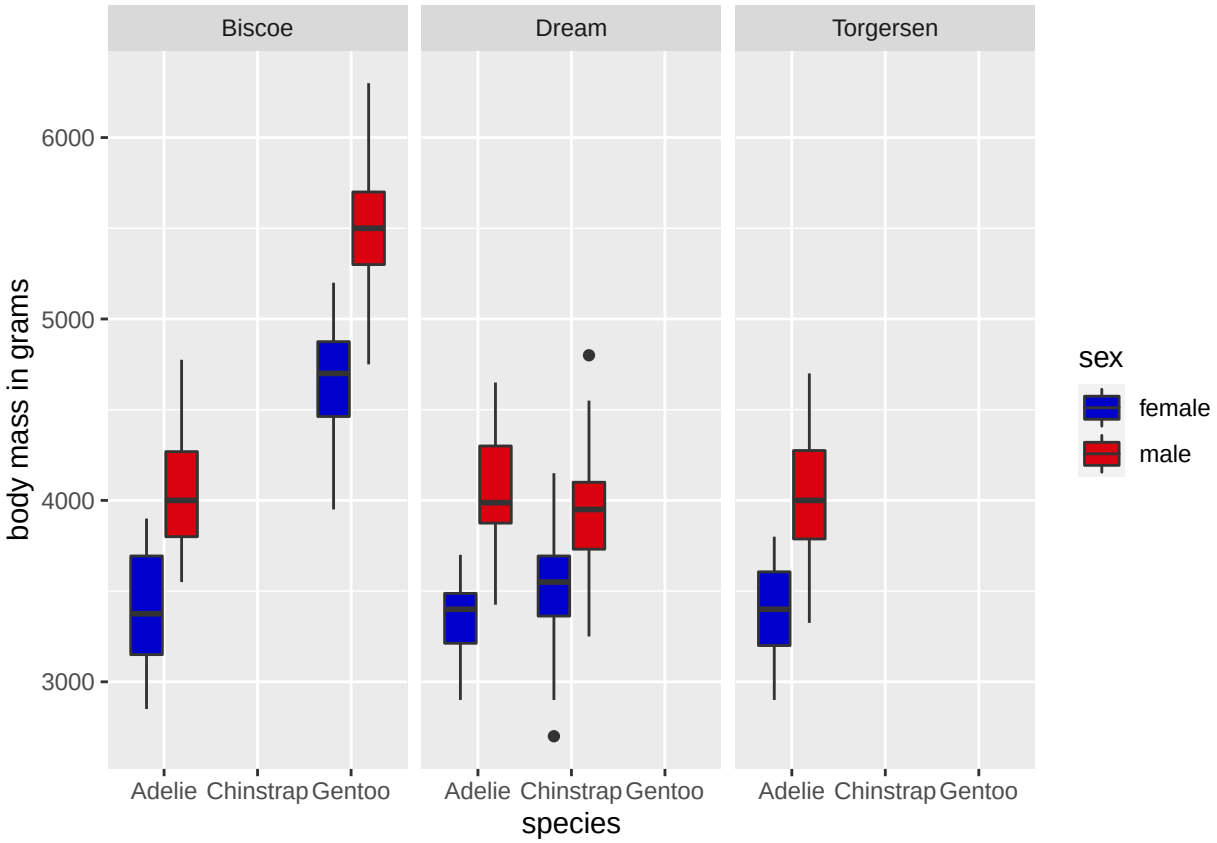
https://ggplot2.tidyverse.org/reference/scale_manual.html

and change the colors of sex to any color you would like by using so called "hex codes" (e.g. the hexcode for the red color of the football club Manchester United is #DA020E) or color names (see for example http://sape.inf.usi.ch/quick-reference/ggplot2/colour)

Try to change the colors of male penguins to Manchester United red and females to the color mediumblue.

hint: Why does scale_color_manual not work? –> Because we use a fill, not a colour command.

```
# Here your code
ggplot() +
  geom_boxplot(data = penguins, aes(x = species, y = body_mass_g, fill = sex)) +
  ylab("body mass in grams") +
  facet_wrap(~ island) +
  scale_fill_manual(values = c("female" = "mediumblue", "male" = "#DA020E" ))
```



End of Notebook