





Getting to know ggplot

Рідлист

Про **ggplot2** специфічно:

- [ggplot2: Elegant Graphics for Data Analysis, 3e \(Hadley Wickham, Danielle Navarro, and Thomas Lin Pedersen\)](#)  — докладно про ggplot2 від власне розробника даної бібліотеки
- [R Graphics Cookbook, 2nd edition \(Winston Chang, 2024\)](#)  — короткі практичні “рецепти” для вирішення специфічних проблем при побудові графіків за використання **ggplot2** та базової графіки

Про датавіз загалом:

- [Fundamentals of Data Visualization \(Claus O. Wilke\)](#) 
- [Telling Stories with Data, With Applications in R and Python \(Rohan Alexander\)](#) 

Створення та збереження графіків

Об'єкт класу **gg** ініціюється через виклик команди **ggplot()**, нові шари до об'єкту додаються через оператор **+**

```
1 ggplot(data = penguins, aes(x = bill_length_mm, y = bill_depth_mm)) +  
2   geom_point()
```

Як і будь який інший об'єкт у R об'єкт класу **gg** може бути збережений у робочому середовищі через зв'язування з іменем

```
1 my_plot <- ggplot(penguins, aes(bill_length_mm, bill_depth_mm)) +  
2   geom_point()  
3 my_plot # при виклику об'єкту буде виконано рендеринг зображення у відповідному вікні
```

Отриманий графік можна зберегти на диск через виклик **ggsave()**, за дефолтного параметру аргументу **plot = last_plot()** буде збережено останній графік, рендер якого було виконано. Файл може бути збережено у розширенні **.png**, **.jpeg**, **.pdf**, **.svg**, **.tiff**, **.bmp**, **.eps**, **.ps** та **.wmf**. За одиницю виміру прийнято **in** (дюйм), проте також можуть бути використані **cm**, **mm** або **px**

```
1 ggsave("D:/work/plot.png", width = 4.1, height = 5.8) # розмір листу А6 у дюймах  
2 ggsave("D:/work/plot.jpeg", width = 105, height = 148, units = "mm") # розмір листу А6 у мм
```

Основні концепти

Граматика **ggplot2** ґрунтується на ідеях представлених Леландом Уїлкінсоном у "[The Grammar of Graphics](#)" (2005) та згодом розвинутих Хедлі Уїкхемом у публікації "[A Layered Grammar of Graphics](#)" (2009)

Графік складається із двох компонентів: даних — безпосередньо інформації, яку необхідно візуалізувати, та мапи — опису того, як змінні у даних співвідносяться з естетичними атрибутами. Графік бути описаний у вигляді ієрархії подібних мап. У **ggplot2** представлено п'ять компонентів для мапування:

- layers
- scales
- coord
- facet
- theme

Основні концепти

Пустий об'єкт класу **gg**

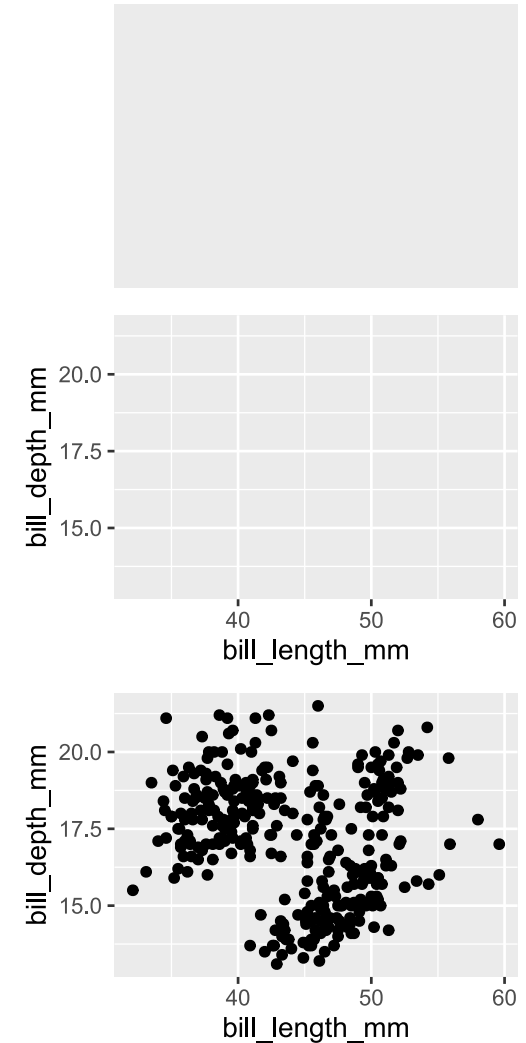
```
1 ggplot()
```

Об'єкт класу **gg**, що містить у собі інформацію про дані та картування естетичних атрибутів оХ та оY

```
1 ggplot(penguins, aes(x = bill_length_mm, y = bill_depth_mm))
```

Наступний шар у ієрархії, **geom_point()**, містить інформацію про геометрію (та статистичну трансформацію), яка має бути застосована для відображення значень. Інформація про дані та значення, які мають бути маповані на геометрію наслідуються від **ggplot()**

```
1 ggplot(penguins, aes(bill_length_mm, bill_depth_mm)) +  
2   geom_point()
```



Основні концепти

Зазвичай при створенні об'єкту одразу вказуються дані та осі, по яким буде будуватися графік, проте це не є обов'язковим — дані, осі та інші естетичні параметри можуть бути специфіковані у шарах нижчої ієрархії. До того ж ієрархічно нижчі шари можуть мати власну естетичну специфікацію, яка відрізняється від специфікації материнського шару створеного при ініціалізації графіку.

Усі наступні команди призводять до ідентичних за виглядом графіків, хоча *внутрішньо* їх структура є дещо відмінною (що можна перевірити зберігши об'єкти під певними іменами та уважно переглянувши структуру їх листів)

```
1 ggplot(penguins, aes(bill_length_mm, bill_depth_mm)) +  
2   geom_point()  
3  
4 ggplot(penguins) +  
5   geom_point(aes(bill_length_mm, bill_depth_mm))  
6  
7 ggplot() +  
8   geom_point(aes(bill_length_mm, bill_depth_mm), penguins)  
9  
10 # найбільш хворий варіант  
11 ggplot(mapping = aes(bill_length_mm, bill_depth_mm)) +  
12   geom_point(data = penguins)
```

Естетичні атрибути різних шарів

Функція `aes(x, y, ...)` слугує для конструкцій мап естетичних атрибутів, аргументи до функції подаються у вигляді пар `aesthetic = variable`, де `variable` зазвичай є колонкою певного кадру даних. Перші два аргументи можуть подаватися просто у вигляді `variable`, які у такому випадку будуть використані для конструкції осі X та Y відповідно

Змінну `species` маповано на естетичний атрибут `color`, мапування розповсюджується на усі ієрархічно нижчі рівні

```
1 penguins |>
2   ggplot(aes(bill_length_mm, bill_depth_mm, color = species)) +
3   geom_point() +
4   geom_smooth(method = lm) # lm - linear model
```

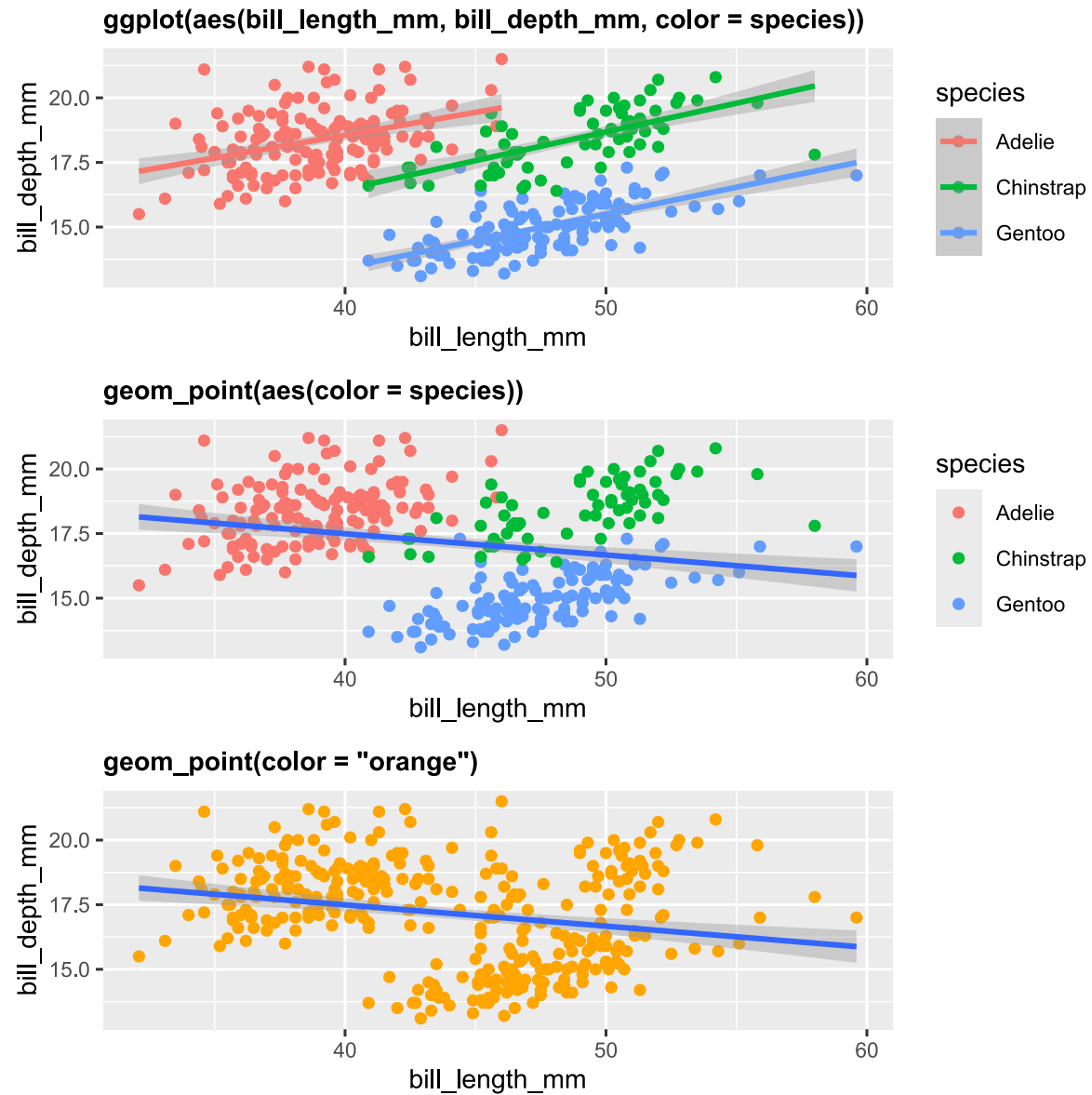
Теж саме, але мапування `species` на `color` розповсюджується лише на рівень шару `geom_point()`

```
1 penguins |>
2   ggplot(aes(bill_length_mm, bill_depth_mm)) +
3   geom_point(aes(color = species)) +
4   geom_smooth(method = lm)
```

Специфікація параметру `color` знаходиться поза межами функції `aes()` і тому визначає естетику використаної геометрії як такої, без зв'язку з даними

```
1 penguins |>
2   ggplot(aes(bill_length_mm, bill_depth_mm)) +
3   geom_point(color = "orange") +
4   geom_smooth(method = lm)
```

Естетичні атрибути різних шарів



Найчастіше використовувані естетичні атрибути

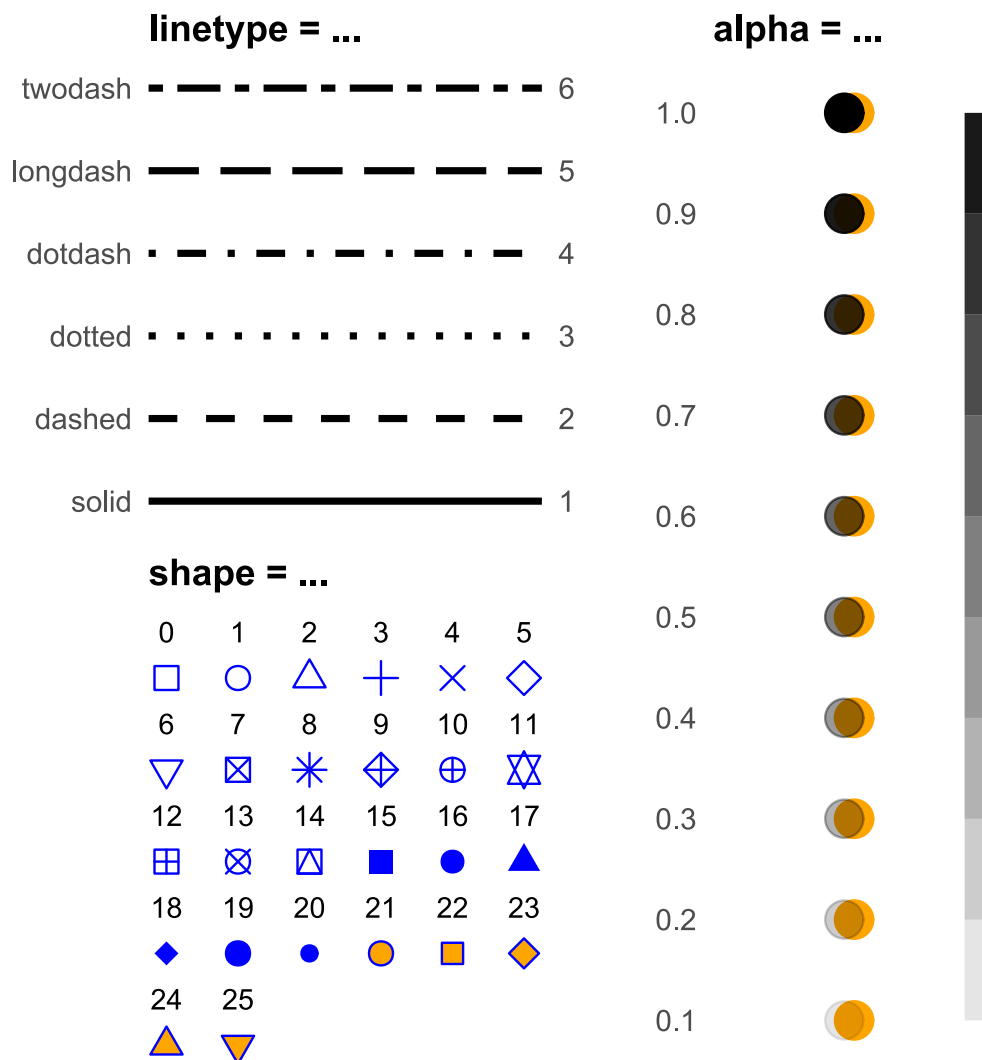
Доступні для всіх або більшості геометрій:

- **color** — колір ліній (контурів)
- **fill** — колір внутрішнього простору
- **alpha** — прозорість
- **linetype** — тип ліній
- **linewidth** — ширина лінії
- **linejoint** — тип з'єднання ліній
- **lineend** — тип кінця лінії
- **group** — група

Доступні для поїнтів:

- **size** — розмір
- **shape** — форма поїнту
- **stroke** — товщина контуру (заміняє **linewidth**)

Найчастіше використовувані естетичні атрибути



Тип лінії контролюється через текстове або чисельне значення (e.g. `linetype = "solid"` або `linetype = 1`).

Форма контролюється через чисельні значення від 0 до 25 або або кодову позначку, що відповідає певній формі (e.g. `shape = 3` або `shape = "+"`). Також можливо використання ASCII-елементів від 32 до 127.

Прозорість встановлюється чисельним значенням, що може приймати значення від 0 до 1

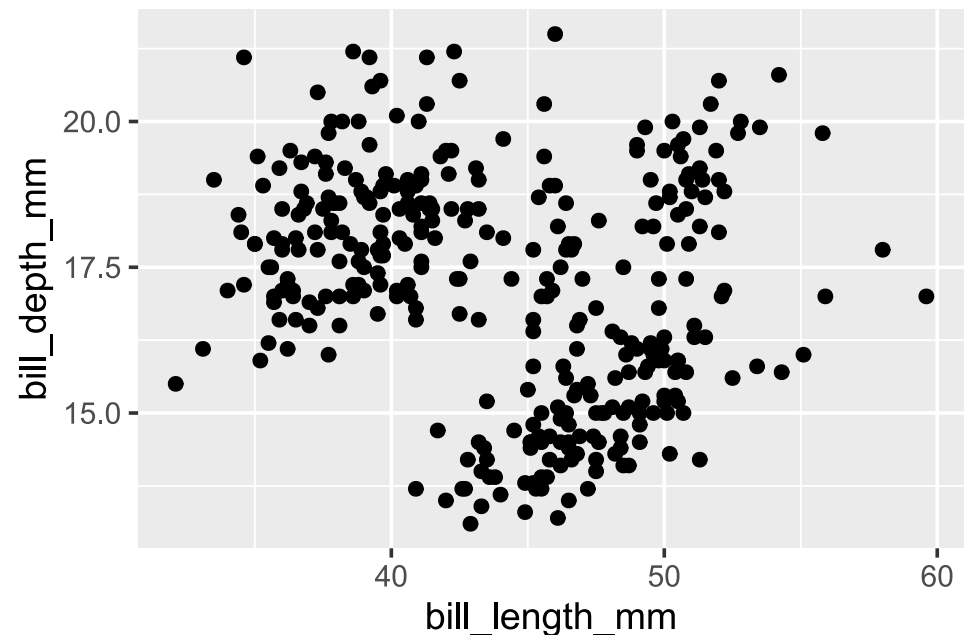
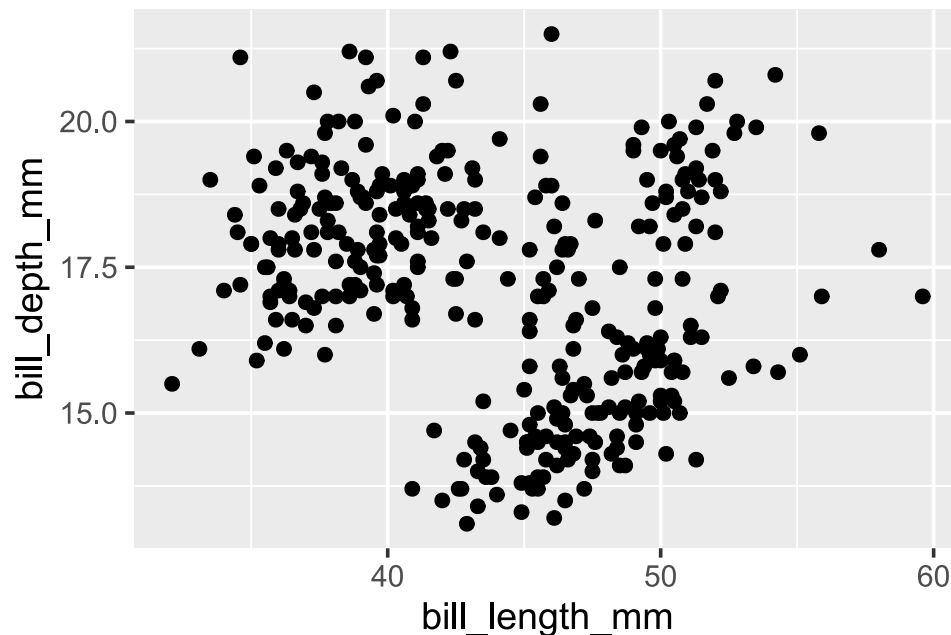
Layers

`layers` — колекція геометричних об'єктів, `geom_*()`, які власне репрезентують візуальну геометрію об'єкту на графіку (point, line, bar, polygon etc), та статистичних трансформації, `stat_*()`, які мають бути проведені над даними, аби представити їх на графіку (count, bin, smooth etc)

Якщо викликати допомогу на будь-яку з доступних геометрій (наприклад `?geom_point`) та подивитися доступні аргументи для функції, можна побачити аргумент `stat` з певним дефолтним значенням. Зворотно є дійсним і для наявних статистичних трансформацій, функції яких мають аргумент `geom` з певним дефолтним значенням (наприклад `?stat_identity`)

```
1 penguins |>  
2 ggplot(aes(bill_length_mm, bill_depth_mm)) +  
3 geom_point(stat = "identity")
```

```
1 penguins |>  
2 ggplot(aes(bill_length_mm, bill_depth_mm))  
3 stat_identity(geom = "point")
```

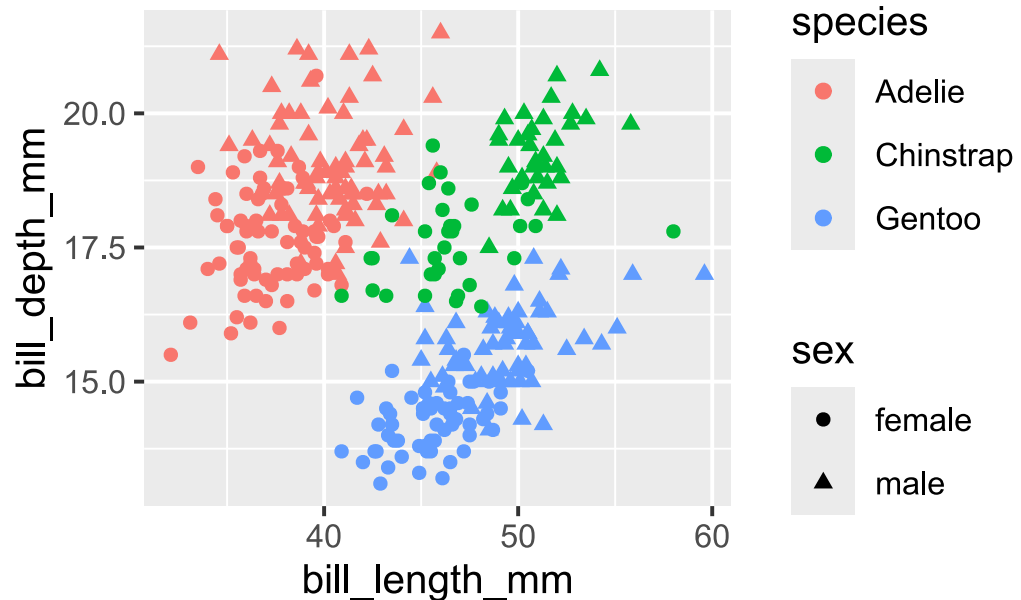


Scales

Scales — відповідають за мапування значень у просторі даних до значень у просторі естетичних атрибутів, а також за зворотне відображення цих зв'язків шляхом створення легенди та осей графіків. Функції для модифікації шкали мають вигляд типу `scale_*()` або `scale_*_*()`. Варіант функції `scale_` існує майже для кожного з можливих атрибутів, проте найчастіше використовуються:

- `scale_color_*()` — контроль кольору контурів геометрій та ліній
- `scale_fill_*()` — контроль кольору внутрішньої області геометрії
- `scale_shape_*()` — контроль форми для `geom_point`

```
1 plt_pengw <- penguins |>
2   ggplot(
3     aes(bill_length_mm,
4         bill_depth_mm)
5   ) +
6   geom_point(
7     aes(color = species,
8         shape = sex)
9   )
```

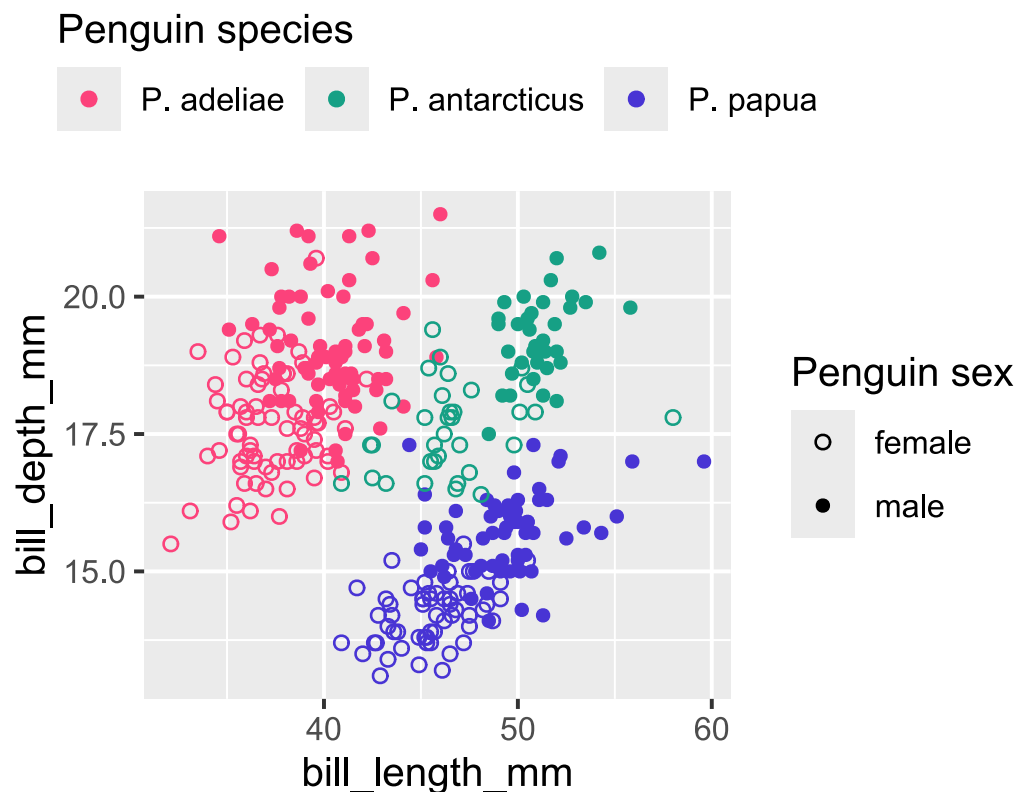


Scales (colors)

Суфікс `_manual()` вказує на те, що функція дозволяє вручну задати значення естетичному атрибуту через аргумент `values`, до якого подається вектор значень. Будь-який варіант `scale_` також одночасно має аргументи для контролю зовнішнього вигляду легенди графіку, що зворотно відображує зв'язок естетики з даними

```
1 pal <- c("#fc427b", "#16a085", "#4834d4") # колір може бути вказаний через ім'я, hex-код
2                                           # виклик функції rgb() або hsv()
3 lab <- c("P. adeliae", "P. antarcticus", "P. papua")
```

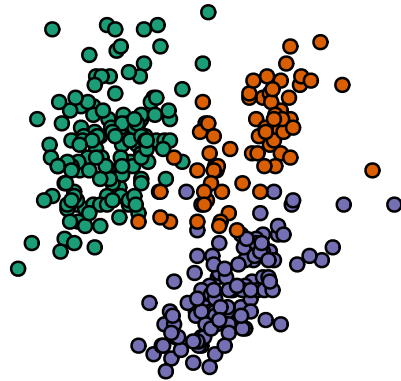
```
1 plt_pengw +
2   scale_color_manual(
3     values = pal,
4     labels = lab,
5     name = "Penguin species",
6     guide = guide_legend(
7       position = "top",
8       title.position = "top"
9     )
10 ) +
11 scale_shape_manual(
12   values = c(1, 21),
13   name = "Penguin sex"
14 )
```



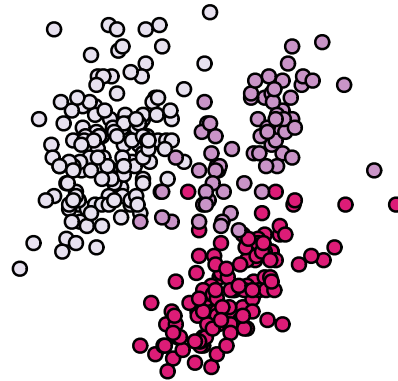
Scales (colors)

Окрім `color/fill_manual` доступним є набір кольорових пресетів з палітр [ColorBrewer](#) та [Viridis](#)

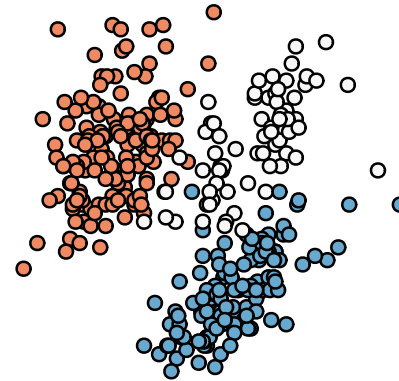
`scale_fill_brewer(
palette = "Dark2")`



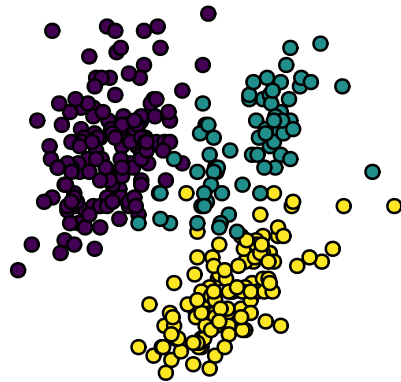
`scale_fill_brewer(
palette = "PuRd")`



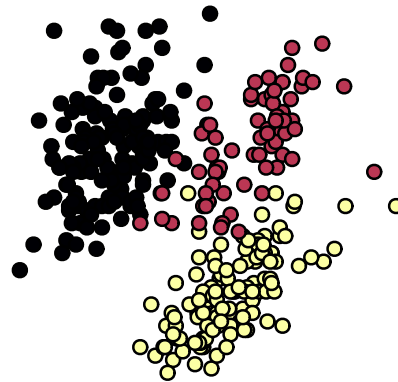
`scale_fill_brewer(
palette = "RdBu")`



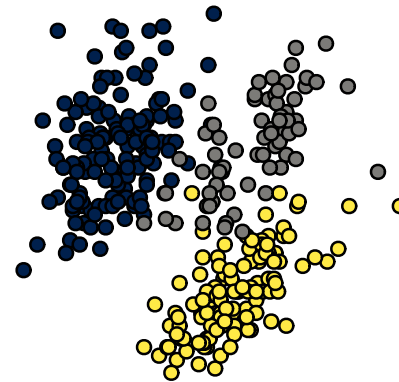
`scale_fill_viridis_d(
option = "viridis")`



`scale_fill_viridis_d(
option = "inferno")`



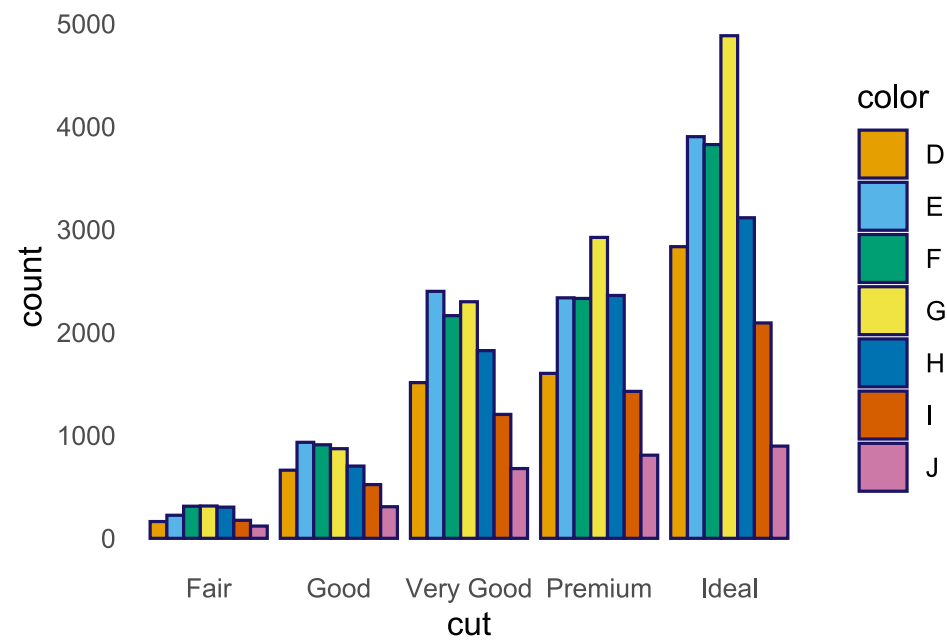
`scale_fill_viridis_d(
option = "cividis")`



Scales (colors)

ColorBrewer першочергово розроблявся для застосування у картографії і має три типи палітр — qualitative, sequential та diverging. Viridis та його похідні розроблялись як заміна кольорової мапи “Jet” [↗](#), їх особливістю є перцептивна однорідність по ходу градієнту. І ті, і інші набори пресетів розроблялись з ідеєю доступності для людей з найбільш розповсюдженими типами колірної сліпоти.

Інший широко застосований кольоровий пресет, що є інклюзивним щодо людей з порушеннями кольорового зору є палітра Окабе-Іто, значення якої продемонстровано нижче, на прикладі датасету diamonds

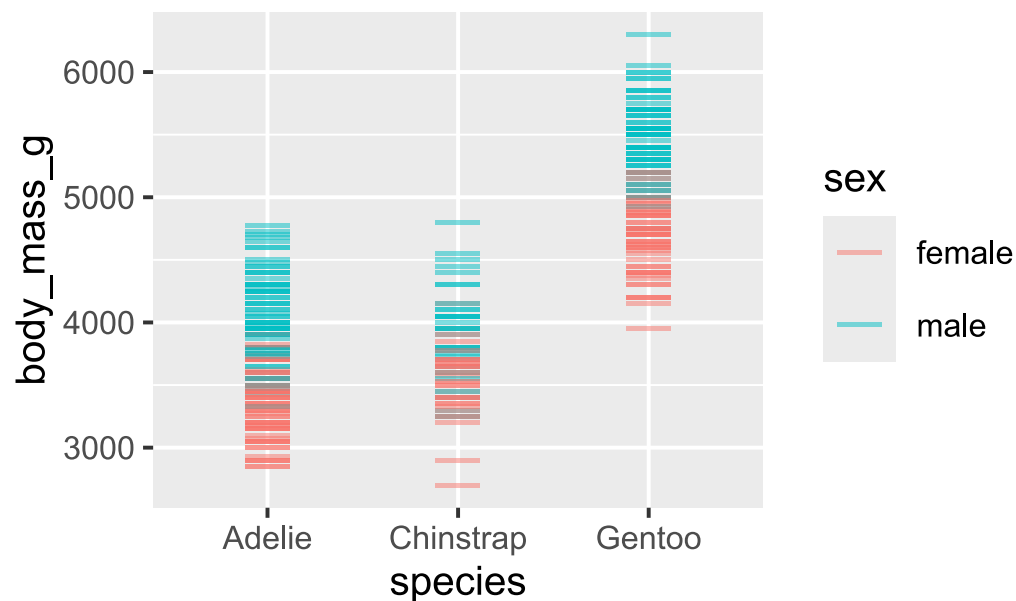


Scales (axis)

`scale_x_*` та `scale_y_*` дозволяють налаштовувати зовнішній вигляд осі X та Y відповідно до залежності від типу даних та їх трансформації, можливими варіантами є:

- **discrete** — для дискретних даних
- **continuous** — для неперервних даних
- **binned** — для неперервних даних, що розбито на біни
- **date**, **time** та **datetime** — для даних у форматі дати та часу
- **log10**, **sqr**, **reverse** — шорткати для лог10, квадратного коріню та реверс-трансформації осі

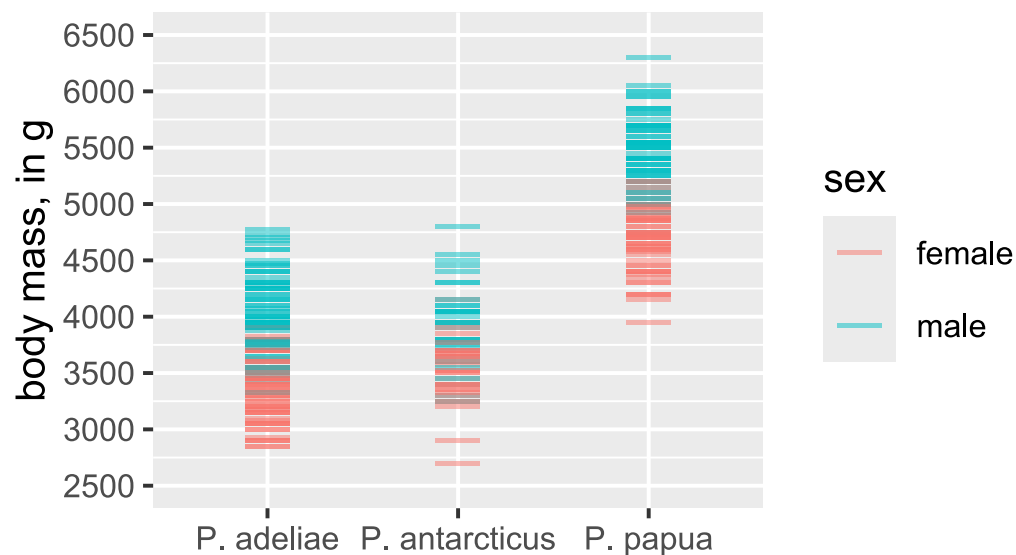
```
1 pengw_mass <- penguins |>  
2   ggplot(  
3     aes(species,  
4         body_mass_g,  
5         color = sex)  
6   ) +  
7   geom_point(  
8     shape = 95,  
9     size = 7,  
10    alpha = .5  
11  )
```



Scales (axis)

`scale_x/y_*` дозволяють контролювати назву осі, назви та кількість окремих засічок на осі, її максимальне та мінімальне значення, її позицію по відношенню до тіла графіку, її статистичну трансформації та наявність додаткової протилежної осі

```
1 pengw_mass +  
2   scale_x_discrete(  
3     name = "",  
4     label = c("P. adeliae",  
5               "P. antarcticus",  
6               "P. papua")  
7   ) +  
8   scale_y_continuous(  
9     name = "body mass, in g",  
10    limits = c(2500, 6500),  
11    breaks = seq(2000, 6500, 500)  
12  )
```

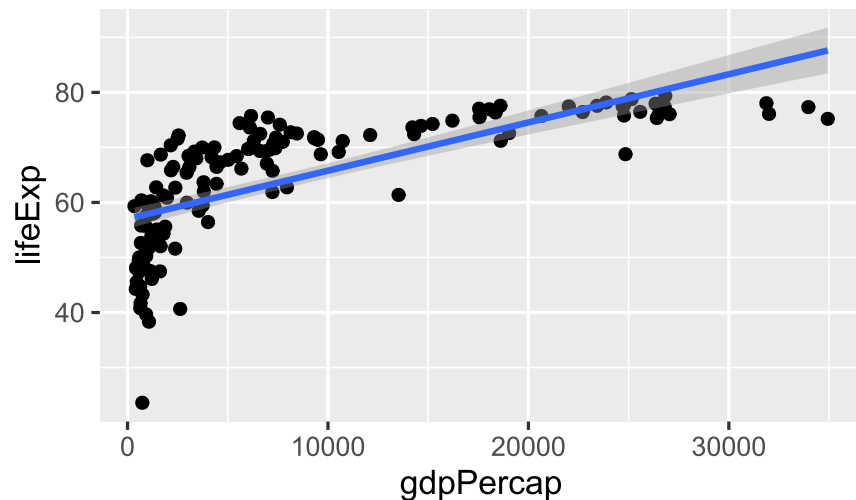


Scales (axis)

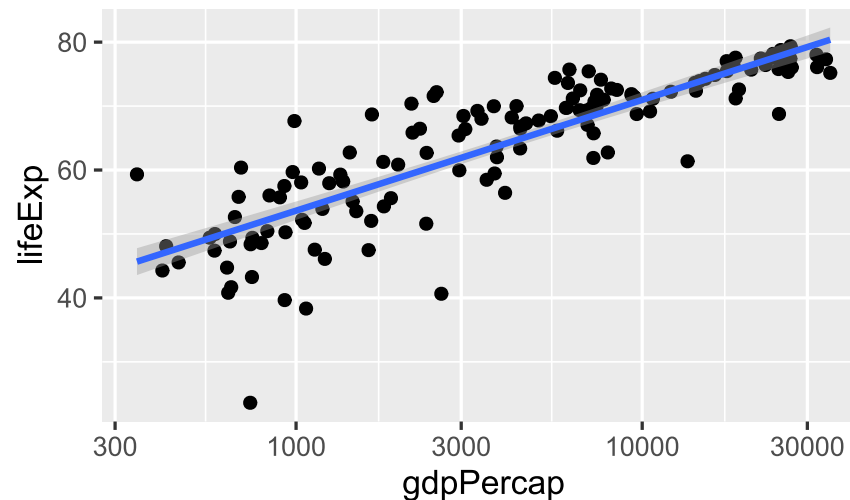
У випадку необхідності трансформації даних для їх кращого відображення, трансформація може бути виконана безпосередньо при передачі даних до функції `aes()`, через `coord_trans()`, через аргумент `transform` у `scale_x/y_*` або, у окремих випадках, через виклик відповідних `scale_x/y_*`

```
1 # дані про ВВП різних країн та очікувану тривалість життя за 1992 рік
2 gdp <- gapminder::gapminder |> filter(year == 1992) |>
3   ggplot(aes(gdpPercap, lifeExp)) +
4   geom_point() +
5   geom_smooth(method = lm)
```

```
1 gdp
```



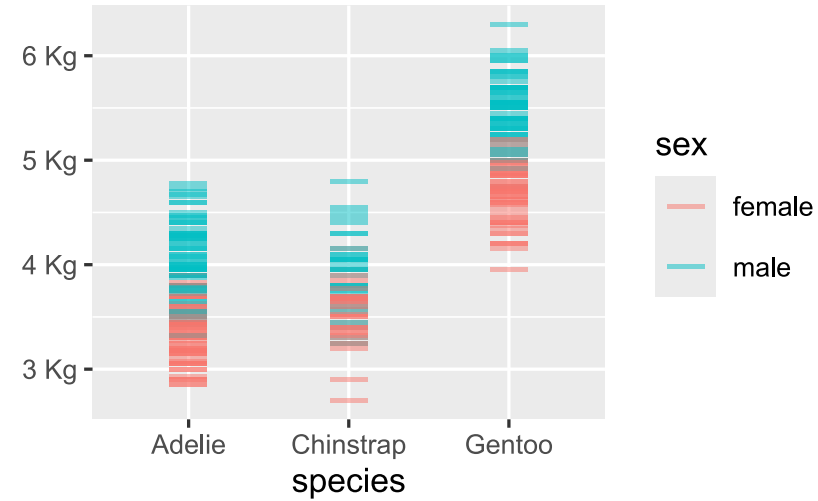
```
1 gdp + scale_x_log10()
```



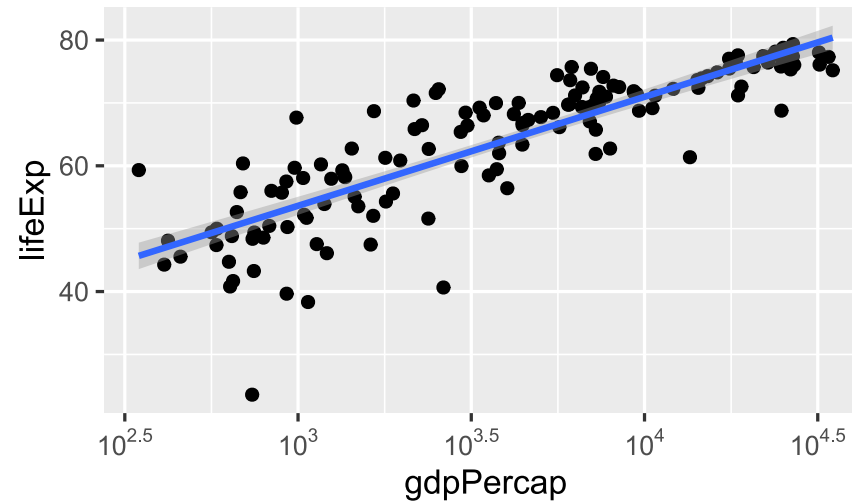
scales::

Додатковий пакет **scales** дозволяє більш тонко контролювати позначки на осях

```
1 pengw_mass +  
2   scale_y_continuous(  
3     name = "",  
4     labels = scales::label_number(  
5       scale = 0.001,  
6       suffix = " Kg"  
7     )  
8   )
```



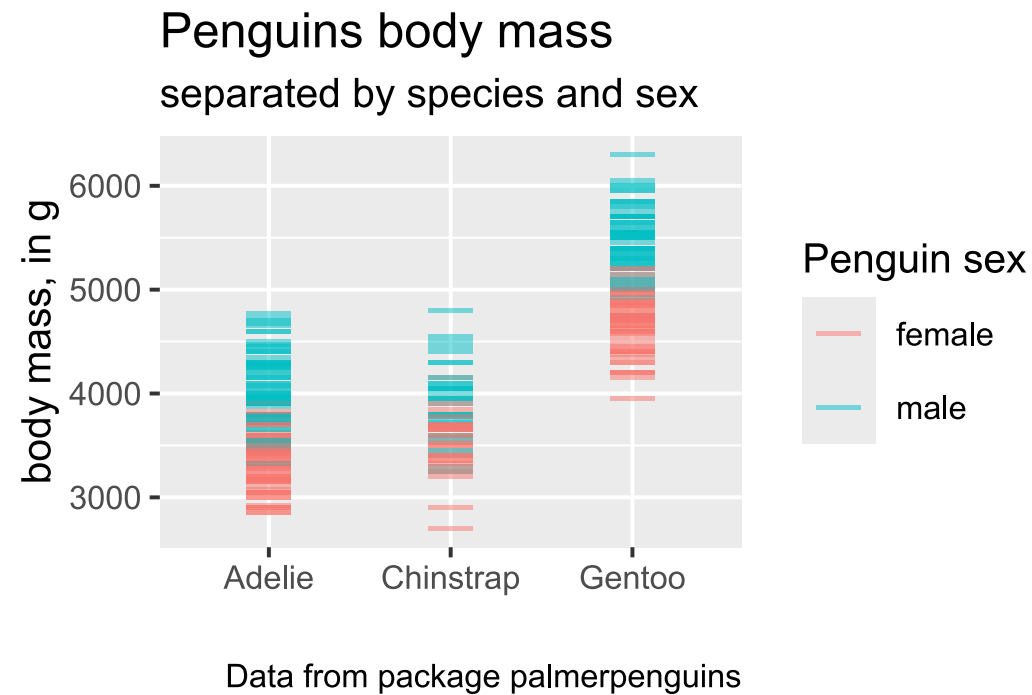
```
1 gdp +  
2   scale_x_log10(  
3     label = scales::label_log(),  
4     breaks = scales::trans_breaks(  
5       "log10", function(x) 10^x  
6     )  
7   )
```



labs()

Функція `labs()` слугує шорткатом для призначення імен елементам графіку

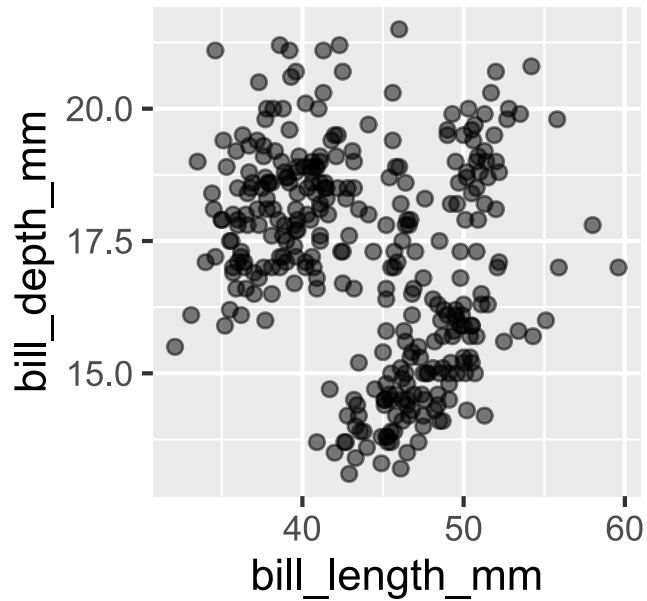
```
1 pengw_mass +  
2   labs(  
3     x = "",  
4     y = "body mass, in g",  
5     title = "Penguins body mass",  
6     color = "Penguin sex",  
7     subtitle = "separated by species and sex",  
8     caption = "Data from package palmerpenguins"  
9   )
```



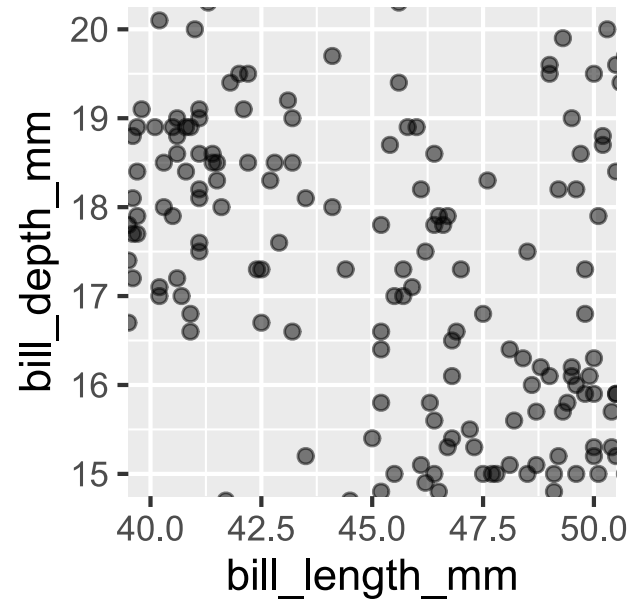
Coord

`coord` визначають, як координати даних відображаються на площині зображення

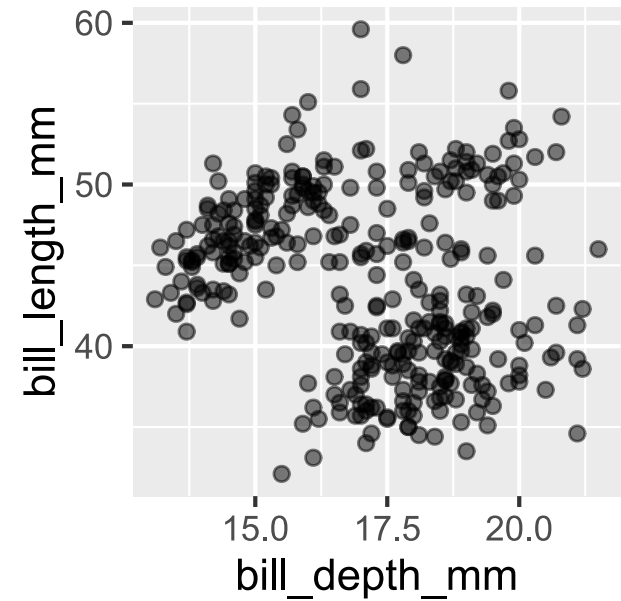
cartesian



**coord_cartesian(
xlim = c(40, 50),
ylim = c(15, 20))**



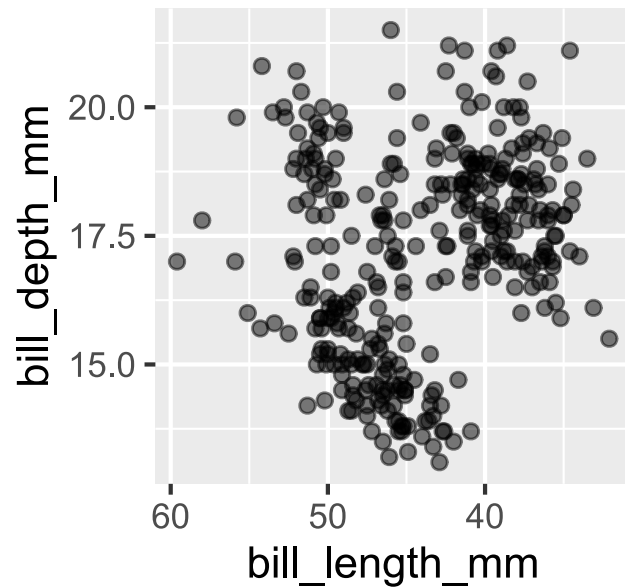
coord_flip



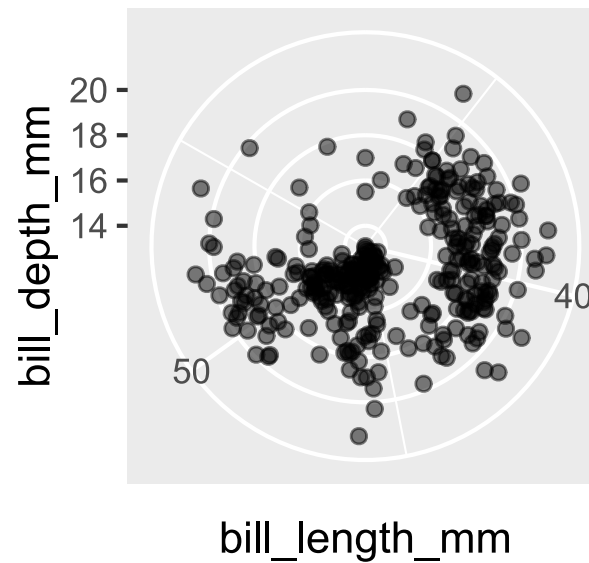
Coord

Окрім зображених нижче, також є `coord_sf()`, що використовується для базового зображення даних накладених на географічну мапу

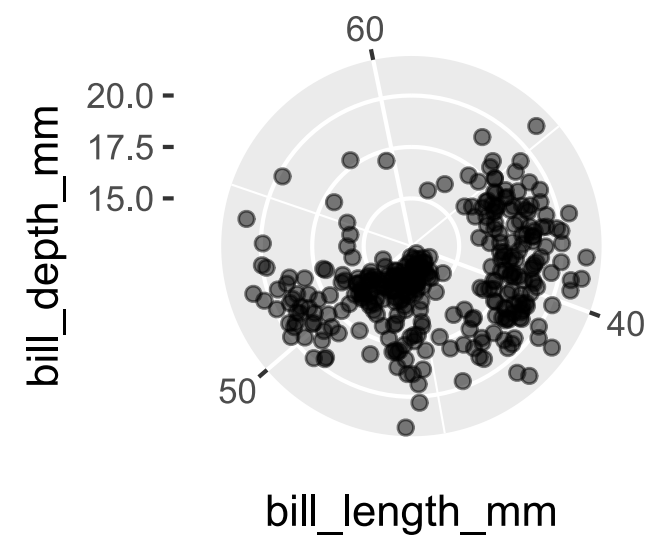
**coord_trans(
x = "reverse")**



coord_polar



coord_radial

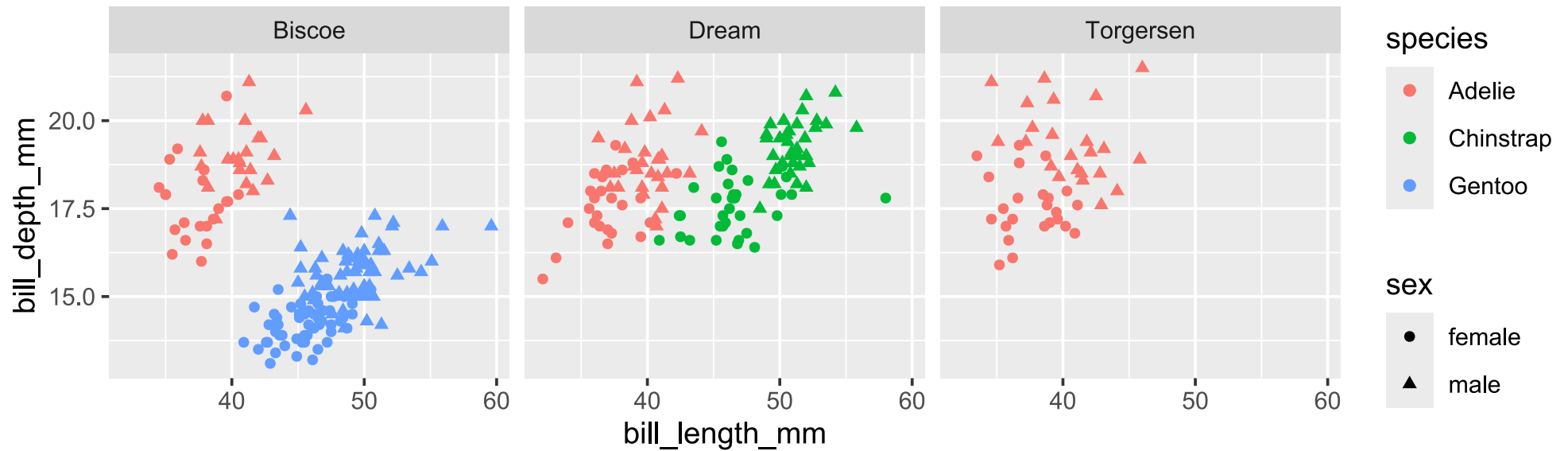


Facet

facet дозволяє розбити набір даних на окремі сеті на основі рівнів певного фактору і зобразити їх у вигляді матриці

- **facet_wrap** — для одного фактору

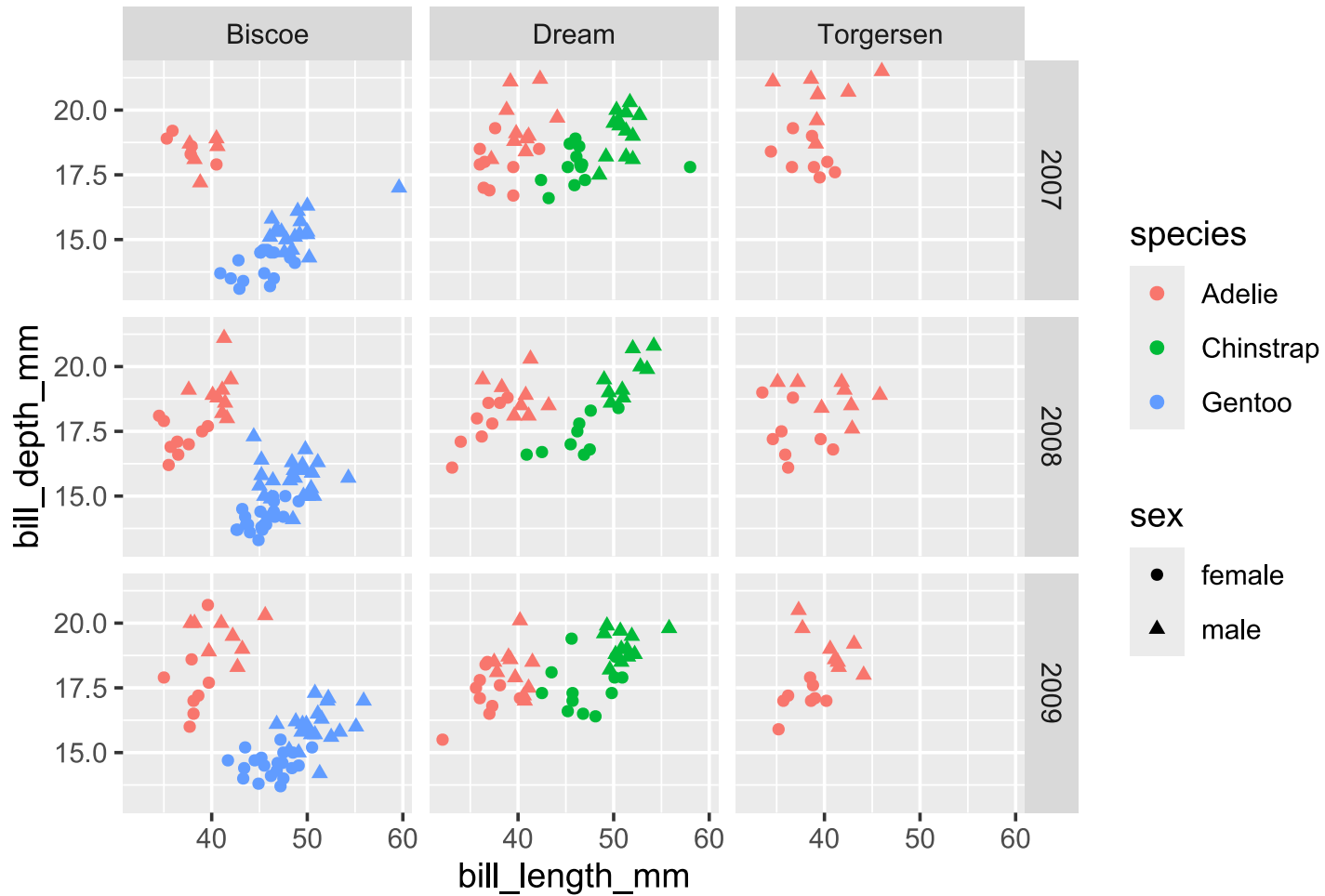
```
1 plt_pengw +  
2   facet_wrap(~island)
```



Facet

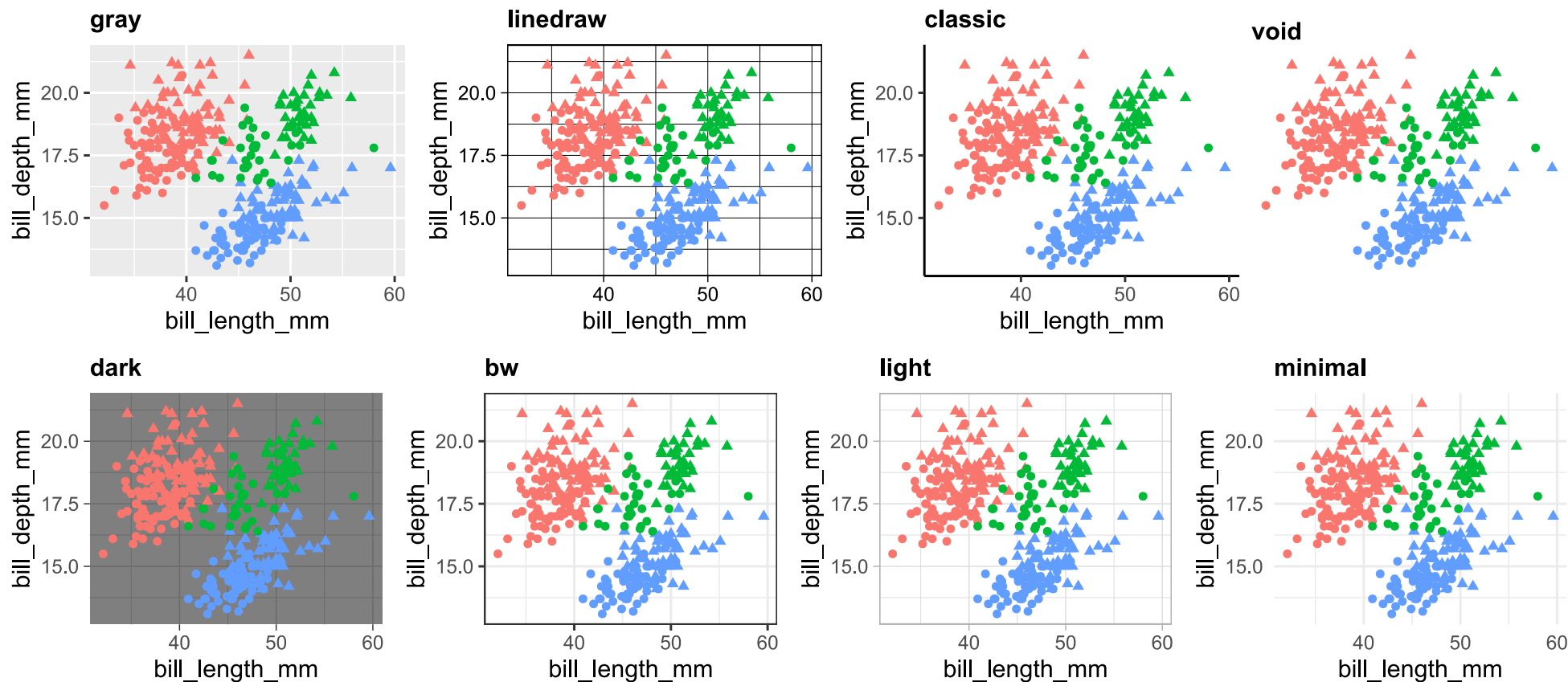
- `facet_grid` — для комбінації рівнів двох факторів

```
1 plt_penguin +  
2   facet_grid(year ~ island)
```



Themes

theme слугує для тонкого налаштування усіх інших візуальних атрибутів графіку, від розміру шрифту до кольору заднього фону. У **ggplot2** існує 8 готових пресетів



Можливо завантажити додаткові готові пресети у вигляді бібліотек, наприклад [ggthemes](#) або [ggdark](#)

Themes

Глобально призначити тему можливо через `theme_set()`, глобально змінити певний елемент теми можливо через `theme_update()`

Мануально налаштувати кожен окремий компонент теми можливо через звернення до функції `theme()`. Аргументи легенди мають доволі дескриптивні назви і також мають ієрархію наслідування. Так аргумент `text` розповсюджується на усі текстові елементи графіку, `axis.title` на підпис осей, `axis.title.x` лише на підпис осі X, `axis.title.x.bottom` лише на підпис осі X, що розташована знизу від тіла графіку.

Абсолютна більшість аргументів утворює пару з однією із наступних функцій, усередині якої вже встановлюються значення атрибутів елемента (колір, розмір, товщина, тип шрифту, відносне положення і т.д.):

- `element_text()`
- `element_line()`
- `element_rect()`
- `element_blank()`

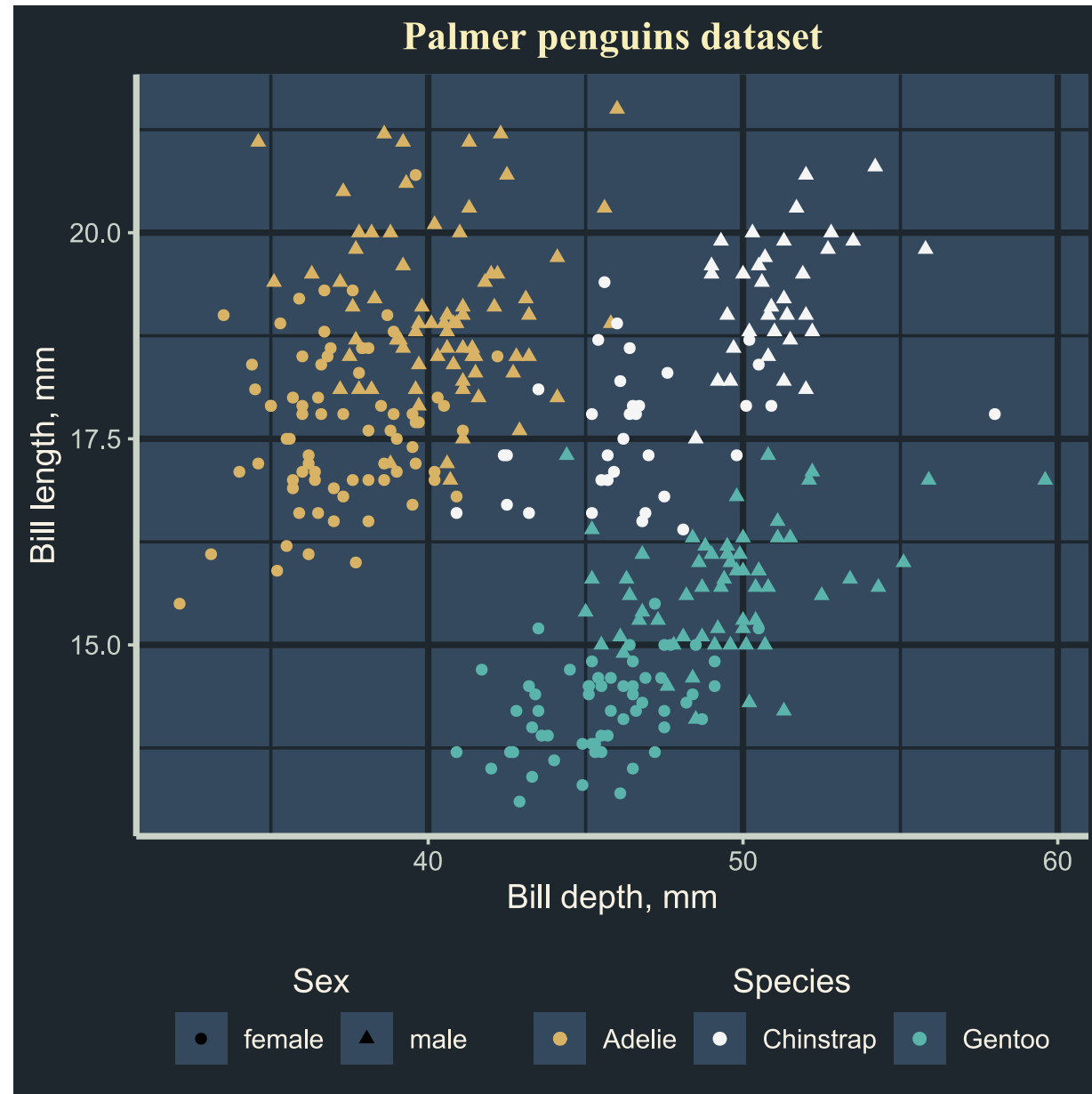
Комбінація певного аргумент з `element_blank()` прибирає елемент з графіку

Themes

Приклад кастомізації теми графіку

```
1 plt_penguin +  
2   labs(x = "Bill depth, mm",  
3       y = "Bill length, mm",  
4       color = "Species",  
5       shape = "Sex",  
6       title = "Palmer penguins dataset") +  
7   scale_color_brewer(palette = "BrBG") +  
8   theme(  
9     text = element_text(color = "#f7f1e3"),  
10    panel.background = element_rect(fill = "#34495e"),  
11    panel.grid = element_line(color = "#1e272e", linewidth = 1),  
12    axis.text = element_text(color = "#CAD3C8"),  
13    axis.line = element_line(color = "#CAD3C8", linewidth = 1),  
14    axis.ticks = element_line(color = "#CAD3C8"),  
15    legend.background = element_rect(fill = "#1e272e"),  
16    legend.position = "bottom",  
17    legend.title.position = "top",  
18    legend.title = element_text(hjust = .5),  
19    plot.background = element_rect(fill = "#1e272e", color = "#1e272e"),  
20    plot.title = element_text(color = "#F8EFBA",  
21                             hjust = .5,  
22                             face = "bold",  
23                             family = "serif")  
24  )
```

Themes



Додаткові посилання

Щодо R:

- [ggplot2 extensions gallery](#) — галерея бібліотек-розширень до **ggplot2**
- [The R Graph Gallery](#) — колекція прикладів графіків створених за допомогою R, по категоріям
- [R Charts](#) — інша колекція прикладів графіків у R, по категоріям

Інші каталоги/галереї/архіви графіків:

- [PolicyViz Data Visualization Catalog](#)
- [The Data Visualisation Catalogue \(Severino Ribecca\)](#)
- [Data Viz Project \(ferdio agency\)](#)

Щодо вибору кольорів:

- [Paul Tol's Notes. INTRODUCTION TO COLOUR SCHEMES](#)
- [DaltonLens. Online colorblindness simulator](#) — можливість перевірити вашу кольорову палітру на читабельність, вичерпний список алгоритмів для симуляції кольорової сліпоты
- [ColorHexa](#) — непоганий онлайн-тул для підбору кольорів

Створення графіків у R по найкращім заповітам Едварда Тафті:

- [Tufte in R \(Lukasz Piwek\)](#)