Le {tidyverse} : {forcats} et {stringr}

Marie VAUGOYEAU

07/02/2023

Contents

Introduction

Historique

Le {tidyverse} s'appelait encore le hadleyverse il y a quelques année, c'est-à-dire l'univers de Hadley pour Hadley Wickham son génial créateur.
Le but de Hadley est de rendre l'analyse données plus facile, plus rapide et surtout **plus fun** et je trouve que cela transparaît dans ses packages !
Le {tidyverse} c'est l'ensemble des packages open-source développé par Hadley et son équipe (Hadley travaille maintenant pour RStudio en plus de plusieurs universités) qui partagent la même philosophie, la même structure de données (le fameux format tidy) et la même syntaxe.

Le format tidy

Le format tidyrepose sur la répétition des lignes des individus afin de limiter le nombre de colonnes.
Dans le plus stricte cas, le format tidy ne présente que 3 colonnes :

_ Identification de l'individu, *par exemple* : nom_du_pays, num_bague_identification,...
_ Variables mesurées, *par exemple* : variable peut prendre comme modalités superficie, taille_population, PIB ou masse, taille, longueur_du_bec...
_ Valeur de la mesure. **Attention**, le format tidy ne supporte par plusieurs type de données dans la même colonne !

La syntaxe `tidyverse`

Non détaillée ici, je vous invite à consulter le tidyverse style guide.

Les packages concernés

_ `ggplot2` : Visulisation des données
_ `dplyr` : Manipulation des données (filtrer, trier,...) à ne pas confondre avec `tidyr` qui manipule le format du jeu de données. Présenté le 7 février sur twitch.
_ `tidyr` : Modification du format du jeu de données pour en faire un jeu de donnée `tidy`. Présenté le 7 février sur twitch.
_ `readr` : Lecture rapide de fichiers de données format `csv` et autres. **Attention** : format `xslx` non pris en charge, il faut utiliser le package `readxl` qui fait partie du `tidyverse` au sens large mais qui n'est pas attaché par défaut quand on fait `library(tidyverse)`
_ `purrr` : Permet le remplacement d'un grand nombre de boucles *aujourd'hui*
_ `tibble` : Format des données `tidy`
_ `stringr` : Manipulation des chaînes de caractères. *Vu aujourd'hui*
_ `forcats` : Manipulation des variables facteurs `factors`. *Vu aujourd'hui*

```
library(tidyverse)
```

Projecteur sur :

`{forcats}`

Le package `{forcats}` permet de manipuler les facteurs en modifiant les modalités, en leur réordonnants... Quasiment toutes les fonctions de ce package commencent par `fct_` pour montrer qu'elles manipulent directement les vecteurs (contrairement aux fonctions du `tidyverse` présentées le mois dernier qui agissent sur un jeu de données `tibble` ou non).

Qu'est ce qu'un facteur ? C'est un vecteur avec des modalités ordonnées, c'est-à-dire que derrière les mots il y a un vecteur numérique.
Cette valeur numérique est accessible via la fonction `fct_anon()` qui permet aussi d'anonymiser.
Exercice : Utilisation du jeu de données `{starwars}`

```
starwars %>%
  glimpse()
```

```
## Rows: 87
## Columns: 14
## $ name       <chr> "Luke Skywalker", "C-3PO", "R2-D2", "Darth Vader", "Leia Or~
## $ height     <int> 172, 167, 96, 202, 150, 178, 165, 97, 183, 182, 188, 180, 2~
## $ mass       <dbl> 77.0, 75.0, 32.0, 136.0, 49.0, 120.0, 75.0, 32.0, 84.0, 77.~
## $ hair_color <chr> "blond", NA, NA, "none", "brown", "brown, grey", "brown", N~
## $ skin_color <chr> "fair", "gold", "white, blue", "white", "light", "light", "~
## $ eye_color  <chr> "blue", "yellow", "red", "yellow", "brown", "blue", "blue",~
## $ birth_year <dbl> 19.0, 112.0, 33.0, 41.9, 19.0, 52.0, 47.0, NA, 24.0, 57.0, ~
## $ sex        <chr> "male", "none", "none", "male", "female", "male", "female",~
## $ gender     <chr> "masculine", "masculine", "masculine", "masculine", "femini~
## $ homeworld  <chr> "Tatooine", "Tatooine", "Naboo", "Tatooine", "Alderaan", "T~
## $ species    <chr> "Human", "Droid", "Droid", "Human", "Human", "Human", "Huma~
```

```
## $ films      <list> <"The Empire Strikes Back", "Revenge of the Sith", "Return~
## $ vehicles   <list> <"Snowspeeder", "Imperial Speeder Bike">, <>, <>, <>, "Imp~
## $ starships  <list> <"X-wing", "Imperial shuttle">, <>, <>, "TIE Advanced x1",~
```

```
# Qu'est-ce qu'un facteur ?
starwars$name %>% fct_anon()
```

```
##  [1] 17 23 06 27 16 34 77 57 37 35 45 74 68 78 66 05 44 60 75 70 28 20 54 08 53
## [26] 38 55 71 25 81 01 42 26 29 15 82 52 30 21 80 31 76 18 84 12 72 40 64 19 22
## [51] 02 03 48 11 67 32 50 51 63 59 58 47 87 43 13 69 56 86 14 36 62 65 24 07 61
## [76] 73 04 39 10 83 33 41 49 46 85 79 09
## 87 Levels: 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 ... 87
```

```
starwars$hair_color %>% fct_anon()
```

```
##  [1] 07   <NA> <NA> 09   10   12   10   <NA> 01   11   07   08   10   10   <NA>
## [16] <NA> 10   10   06   05   01   09   09   01   09   09   04   10   10   09
## [31] 10   09   07   09   09   09   10   01   09   01   01   09   09   09   09
## [46] 09   09   09   06   09   01   09   09   09   09   09   01   10   10   09
## [61] 01   01   10   06   01   01   02   09   09   09   06   09   09   09   09
## [76] 09   09   10   10   09   09   01   10   10   09   03   10
## Levels: 01 02 03 04 05 06 07 08 09 10 11 12
```

Analyser les facteurs

```
# compter les modalités
starwars$eye_color %>% fct_count()
```

```
## # A tibble: 15 x 2
##    f                 n
##    <fct>         <int>
##  1 black            10
##  2 blue             19
##  3 blue-gray         1
##  4 brown            21
##  5 dark              1
##  6 gold              1
##  7 green, yellow     1
##  8 hazel             3
##  9 orange            8
## 10 pink              1
## 11 red               5
## 12 red, blue         1
## 13 unknown           3
## 14 white             1
## 15 yellow           11
```

```
starwars %>% count(eye_color)
```

```
## # A tibble: 15 x 2
##    eye_color         n
##    <chr>         <int>
##  1 black            10
##  2 blue             19
##  3 blue-gray         1
##  4 brown            21
##  5 dark              1
##  6 gold              1
##  7 green, yellow     1
##  8 hazel             3
##  9 orange            8
## 10 pink              1
## 11 red               5
## 12 red, blue         1
## 13 unknown           3
## 14 white             1
## 15 yellow           11
```

```
# afficher l'intégralité des modalité
starwars$species %>%  fct_unique()
```

```
##  [1] Aleena         Besalisk       Cerean         Chagrian       Clawdite
##  [6] Droid          Dug            Ewok           Geonosian      Gungan
## [11] Human          Hutt           Iktotchi       Kaleesh        Kaminoan
## [16] Kel Dor        Mirialan       Mon Calamari   Muun           Nautolan
## [21] Neimodian      Pau'an         Quermian       Rodian         Skakoan
## [26] Sullustan      Tholothian     Togruta        Toong          Toydarian
## [31] Trandoshan     Twi'lek        Vulptereen     Wookiee        Xexto
## [36] Yoda's species Zabrak         <NA>
## 37 Levels: Aleena Besalisk Cerean Chagrian Clawdite Droid Dug ... Zabrak
```

Modifier les facteurs

```
# combiner les colonnes
fct_cross(starwars$sex, starwars$gender, sep = "/")
```

```
##  [1] male/masculine           none/masculine          none/masculine
##  [4] male/masculine           female/feminine         male/masculine
##  [7] female/feminine          none/masculine          male/masculine
## [10] male/masculine           male/masculine          male/masculine
## [13] male/masculine           male/masculine          male/masculine
## [16] hermaphroditic/masculine male/masculine          male/masculine
## [19] male/masculine           male/masculine          male/masculine
## [22] none/masculine           male/masculine          male/masculine
## [25] male/masculine           male/masculine          female/feminine
## [28] male/masculine           male/masculine          male/masculine
## [31] male/masculine           male/masculine          male/masculine
## [34] male/masculine           male/masculine          male/masculine
## [37] <NA>                     male/masculine          male/masculine
## [40] <NA>                     female/feminine         male/masculine
```

```
## [43] male/masculine           female/feminine         male/masculine
## [46] male/masculine           male/masculine          male/masculine
## [49] male/masculine           male/masculine          male/masculine
## [52] female/feminine          male/masculine          male/masculine
## [55] male/masculine           male/masculine          male/masculine
## [58] female/feminine          male/masculine          male/masculine
## [61] female/feminine          female/feminine         female/feminine
## [64] male/masculine           male/masculine          male/masculine
## [67] female/feminine          male/masculine          male/masculine
## [70] female/feminine          female/feminine         male/masculine
## [73] none/feminine            male/masculine          male/masculine
## [76] female/feminine          male/masculine          male/masculine
## [79] male/masculine           <NA>                    male/masculine
## [82] male/masculine           female/feminine         male/masculine
## [85] none/masculine           <NA>                    female/feminine
## 5 Levels: female/feminine none/feminine ... none/masculine
```

```
# en dplyr avec la fonction glue() du package {glue}
mutate(
  starwars,
  sexgenre = glue::glue("{sex}/{gender}")
)
```

```
## # A tibble: 87 x 15
##    name        height  mass hair_~1 skin_~2 eye_c~3 birth~4 sex   gender homew~5
##    <chr>        <int> <dbl> <chr>   <chr>   <chr>     <dbl> <chr> <chr>  <chr>
##  1 Luke Skywa~    172    77 blond   fair    blue       19   male  mascu~ Tatooi~
##  2 C-3PO          167    75 <NA>    gold    yellow    112   none  mascu~ Tatooi~
##  3 R2-D2           96    32 <NA>    white,~ red        33   none  mascu~ Naboo
##  4 Darth Vader    202   136 none    white   yellow     41.9 male  mascu~ Tatooi~
##  5 Leia Organa    150    49 brown   light   brown      19   fema~ femin~ Aldera~
##  6 Owen Lars      178   120 brown,~ light   blue       52   male  mascu~ Tatooi~
##  7 Beru White~    165    75 brown   light   blue       47   fema~ femin~ Tatooi~
##  8 R5-D4           97    32 <NA>    white,~ red        NA   none  mascu~ Tatooi~
##  9 Biggs Dark~    183    84 black   light   brown      24   male  mascu~ Tatooi~
## 10 Obi-Wan Ke~    182    77 auburn~ fair    blue-g~    57   male  mascu~ Stewjon
## # ... with 77 more rows, 5 more variables: species <chr>, films <list>,
## #   vehicles <list>, starships <list>, sexgenre <glue>, and abbreviated
## #   variable names 1: hair_color, 2: skin_color, 3: eye_color, 4: birth_year,
## #   5: homeworld
```

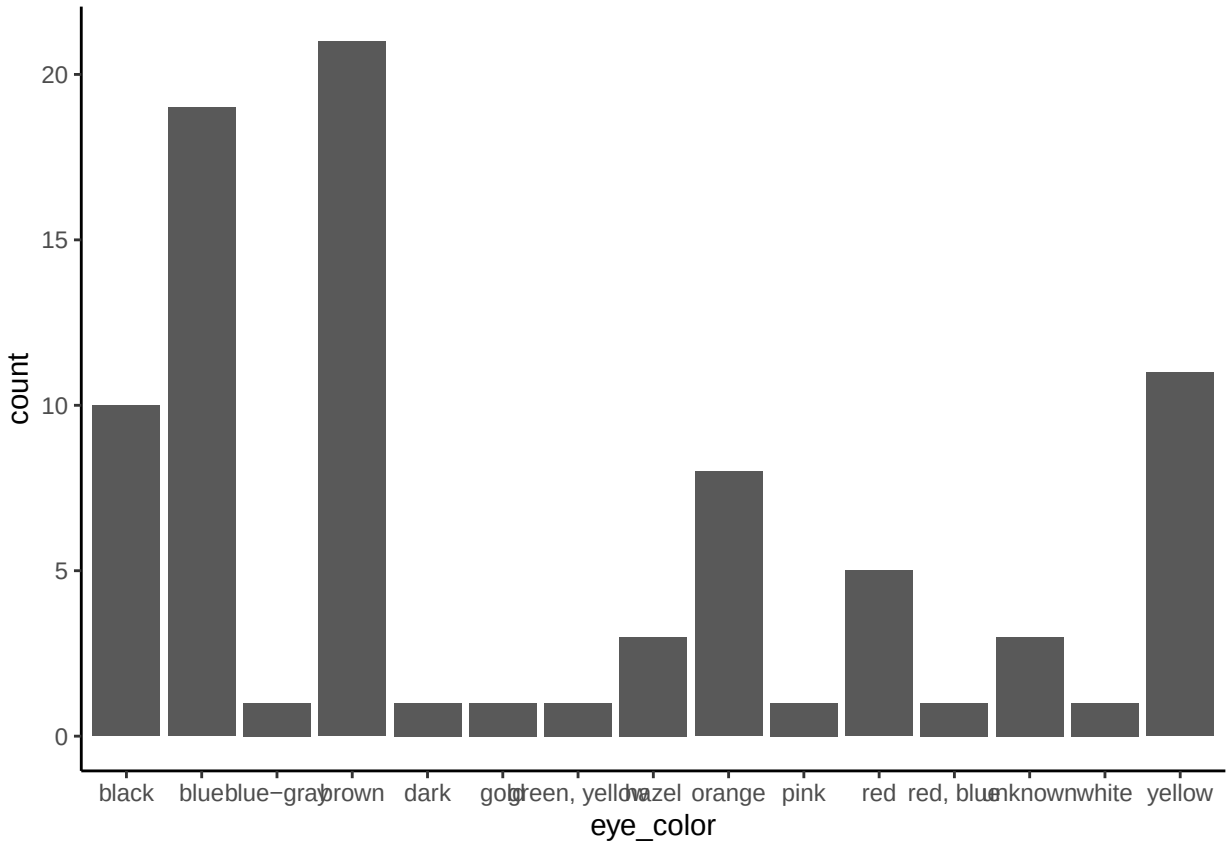
```
# changer les valeurs de gender -> sex
starwars_mod <- starwars %>%
  mutate(
    sexe = sex %>% as_factor(),
    genre = fct_recode(
      gender,
      male = "masculine",
      female = "feminine"
    )
  )

# homogénéiser les niveaux
fct_unify(list(starwars_mod$sexe, starwars_mod$genre))
```

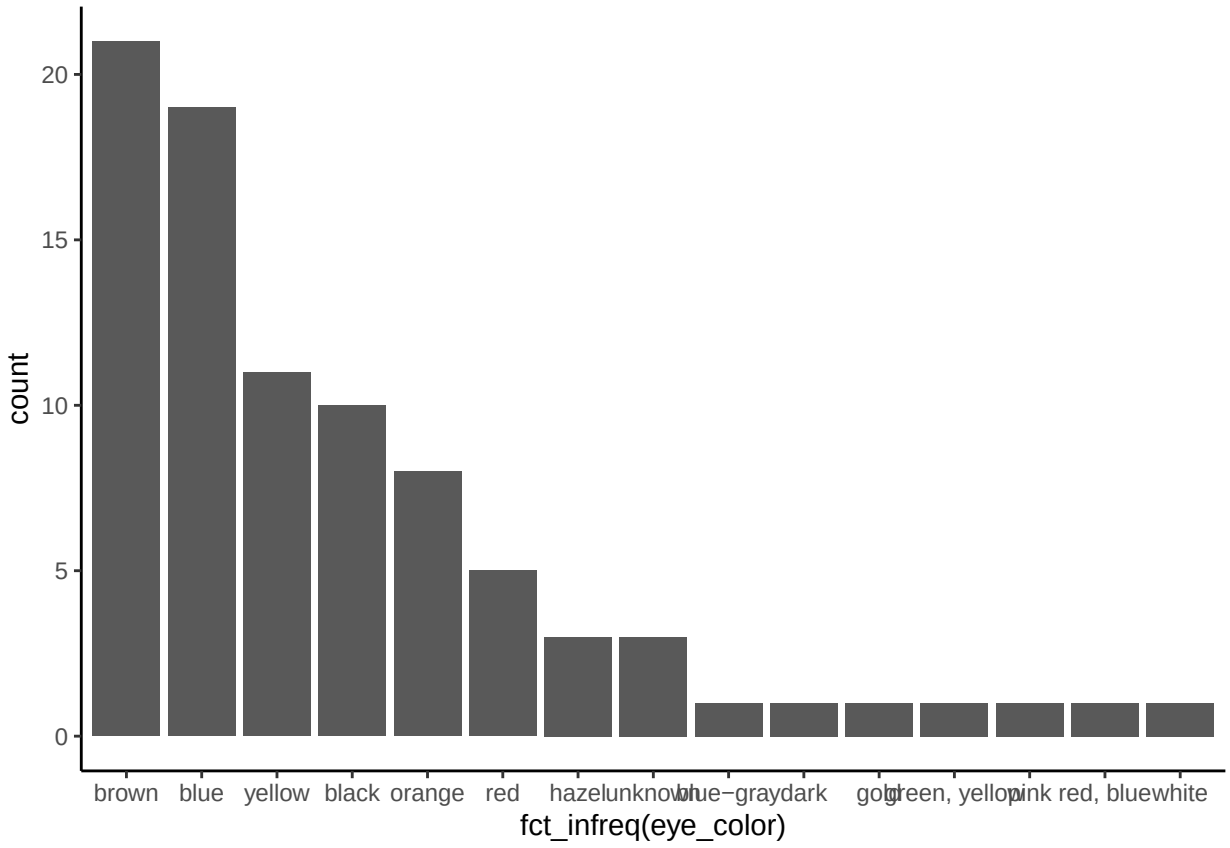
```
## [[1]]
##  [1] male           none           none           male           female
##  [6] male           female         none           male           male
## [11] male           male           male           male           male
## [16] hermaphroditic male           male           male           male
## [21] male           none           male           male           male
## [26] male           female         male           male           male
## [31] male           male           male           male           male
## [36] male           <NA>           male           male           <NA>
## [41] female         male           male           female         male
## [46] male           male           male           male           male
## [51] male           female         male           male           male
## [56] male           male           female         male           male
## [61] female         female         female         male           male
## [66] male           female         male           male           female
## [71] female         male           none           male           male
## [76] female         male           male           male           <NA>
## [81] male           male           female         male           none
## [86] <NA>           female
## Levels: male none female hermaphroditic
##
## [[2]]
##  [1] male   male   male   male   female male   female male   male   male
## [11] male   male   male   male   male   male   male   male   male   male
## [21] male   male   male   male   male   male   female male   male   male
## [31] male   male   male   male   male   male   <NA>   male   male   <NA>
## [41] female male   male   female male   male   male   male   male   male
## [51] male   female male   male   male   male   male   female male   male
## [61] female female female male   male   male   female male   male   female
## [71] female male   female male   male   female male   male   male   <NA>
## [81] male   male   female male   male   <NA>   female
## Levels: male none female hermaphroditic
```

Changer l'ordre des modalités

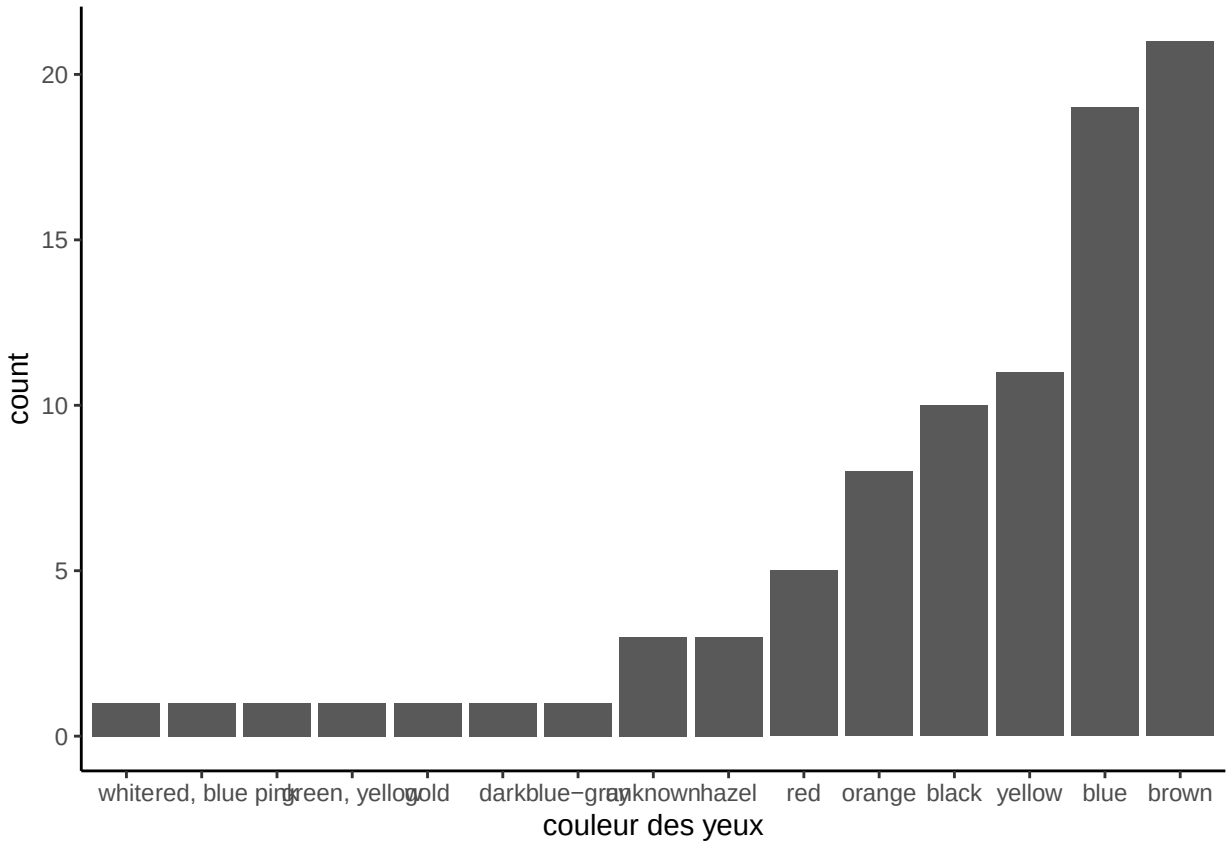
```
## origine
starwars %>%
  ggplot() +
  aes(x = eye_color) +
  geom_bar() +
  theme_classic()
```



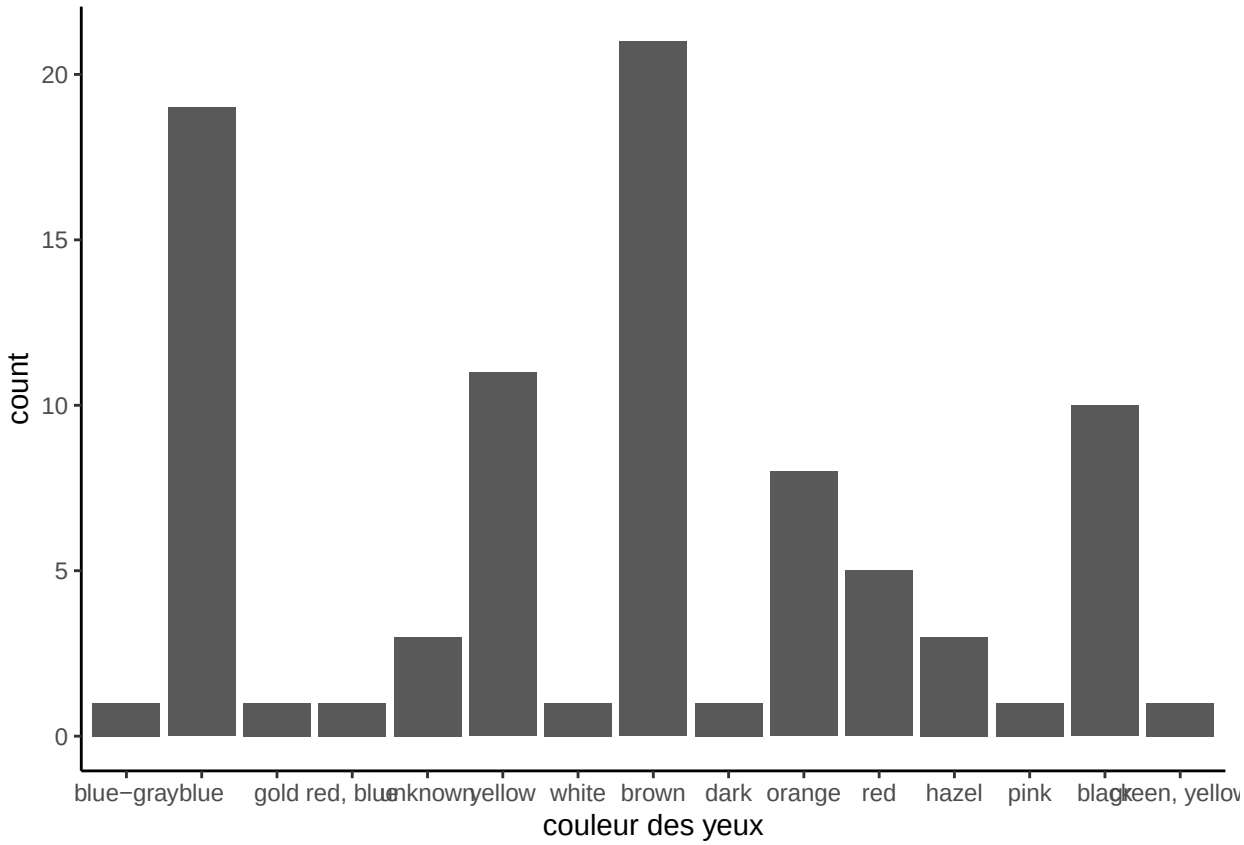

```
## en fonction de la frequence
starwars %>%
  ggplot() +
  aes(x = fct_infreq(eye_color)) +
  geom_bar() +
  theme_classic()
```




```
## descendant
starwars %>%
  ggplot() +
  aes(x = fct_infreq(eye_color) %>% fct_rev()) +
  geom_bar() +
  xlab("couleur des yeux") +
  theme_classic()
```




```
## ordre au hasard
starwars %>%
  ggplot() +
  aes(x = fct_infreq(eye_color) %>% fct_shuffle()) +
  geom_bar() +
  xlab("couleur des yeux") +
  theme_classic()
```



Ressources intéressantes :

_ le cheatsheet de `{forcats}`
_ la vignette de `{forcats}`

`{stringr}`

Le package `{stringr}` permet de manipuler facilement des chaînes de caractères.
Comme les fonctions du package `{forcats}`, les fonctions du package `{stringr}` agit sur un vecteur, non un jeu de données et commencent majoritairement par `str_`.

Manipulation de base

Création et manipulation de chaînes de caractères

```
# texte <- c('c'est', 'ceci')
```

```
texte <- c("c'est", "l'idée", "folle", "qu'une", "femme", "est", "une", "personne")

# calcul longeur
str_c(texte, collapse = " ") %>%
  str_length()
```

```
## [1] 48
```

```r
str_length(texte)
```

```
## [1] 5 6 5 6 5 3 3 8
```

```r
# extraire une partie
str_c(texte, collapse = " ") %>%
  str_sub(start = -8L)
```

```
## [1] "personne"
```

```r
str_sub(texte, start = 2, end = 5)
```

```
## [1] "'est" "'idé" "olle" "u'un" "emme" "st"   "ne"   "erso"
```

```r
# trier par ordre alphabétique
str_sort(texte)
```

```
## [1] "c'est"    "est"      "femme"    "folle"    "l'idée"   "personne" "qu'une"
## [8] "une"
```

```r
str_rank(texte)
```

```
## [1] 1 5 4 7 3 2 8 6
```

```r
# séparer selon un marqueur
str_split(texte, "'", simplify = TRUE)
```

```
##      [,1]       [,2]
## [1,] "c"        "est"
## [2,] "l"        "idée"
## [3,] "folle"    ""
## [4,] "qu"       "une"
## [5,] "femme"    ""
## [6,] "est"      ""
## [7,] "une"      ""
## [8,] "personne" ""
```

```r
texte %>%
  str_split("e")
```

```
## [[1]]
## [1] "c'" "st"
##
## [[2]]
## [1] "l'idé" ""
##
## [[3]]
## [1] "foll" ""
##
```

```
## [[4]]
## [1] "qu'un" ""
##
## [[5]]
## [1] "f"  "mm" ""
##
## [[6]]
## [1] ""   "st"
##
## [[7]]
## [1] "un" ""
##
## [[8]]
## [1] "p"     "rsonn" ""
```

Initiation au `regex`

Le `regex` est l'abréviation de `regular expression`, soit `expression régulière`.
L'expression régulière permet de détecter des structures particulières, des `pattern` dans du texte.
*Par exemple, en tant qu'être humain il est facile de savoir que `blabla@truc.much` a le format d'une adresse e-mail. L'ordinateur, lui a besoin qu'on le "guide" sur le format que l'on cherche.
En regex, une adresse e-mail se traduit par `[:alphanum:]+@[:alphanum:]+\\.[:alphanum:]{2,10}`.

```
str_detect(texte, "e")
str_extract(texte, "[aeiou]")
str_extract_all(texte, "[aeiou]", simplify = TRUE)

# attention au `.` qui remplace n'importe quel caractère
str_detect(texte, ".")
str_detect(autre_texte, "\.")
```

```
## Error: '\.' est un code escape non reconnu dans une chaîne de caractères débutant ""\."
```

```
str_detect(texte, "\\.")
```

```
## [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```
autre_texte <- "Le féminisme."
str_locate_all(autre_texte, "sm")
```

```
## [[1]]
##      start end
## [1,]    10  11
```

```
# détection de ponctuation
str_detect(texte, "[:punct:]")
```

```
## [1]  TRUE  TRUE FALSE  TRUE FALSE FALSE FALSE FALSE
```

```
str_detect(autre_texte, "[:punct:]")
```

```
## [1] TRUE
```

Modification du texte

```
# à la première itération
str_replace(texte, "une", "un")
```

```
## [1] "c'est"    "l'idée"   "folle"    "qu'un"    "femme"    "est"      "un"
## [8] "personne"
```

```
texte %>%
  str_c(collapse = " ") %>%
  str_replace("une", "un")
```

```
## [1] "c'est l'idée folle qu'un femme est une personne"
```

```
# ou à chaque fois
texte %>%
  str_c(collapse = " ") %>%
  str_replace_all("une", "un")
```

```
## [1] "c'est l'idée folle qu'un femme est un personne"
```

```
# passer en minuscule
autre_texte %>%
  str_to_lower()
```

```
## [1] "le féminisme."
```

```
# ou en majuscule
autre_texte %>%
  str_to_upper()
```

```
## [1] "LE FÉMINISME."
```

```
# lettre capitale pour chaque première lettre de mots
autre_texte %>%
  str_to_title()
```

```
## [1] "Le Féminisme."
```

```
# passer en minuscule les noms des colonnes d'un jeu de données
iris %>%
  rename_with(str_to_lower) %>%
  glimpse()
```

```
## Rows: 150
## Columns: 5
## $ sepal.length <dbl> 5.1, 4.9, 4.7, 4.6, 5.0, 5.4, 4.6, 5.0, 4.4, 4.9, 5.4, 4.~
## $ sepal.width  <dbl> 3.5, 3.0, 3.2, 3.1, 3.6, 3.9, 3.4, 3.4, 2.9, 3.1, 3.7, 3.~
## $ petal.length <dbl> 1.4, 1.4, 1.3, 1.5, 1.4, 1.7, 1.4, 1.5, 1.4, 1.5, 1.5, 1.~
## $ petal.width  <dbl> 0.2, 0.2, 0.2, 0.2, 0.2, 0.4, 0.3, 0.2, 0.2, 0.1, 0.2, 0.~
## $ species      <fct> setosa, setosa, setosa, setosa, setosa, setosa, setosa, s~
```

Ressources intéressantes :

_ le cheatsheet de `{stringr}`
_ la page de `{stringr}`