

# RR 2024

## esquisse

Une application pour visualiser ses données avec ggplot2



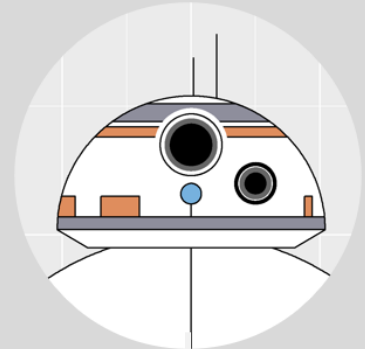
# Qui suis-je

**Victor**



Tavaille à dreamRs

Développe  
principalement des  
packages autour de  
{shiny}



Twitter:

[https://twitter.com/\\_pvictor](https://twitter.com/_pvictor)  
[or](#)

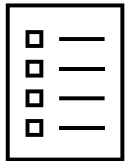
GitHub:

<https://github.com/pvictor>

# esquisse



**Moins tu sais coder,  
plus tu vas rigoler !**



- Présentation du paquet
- Nouveautés de la version 2.0.0
- Application web pour utiliser esquisse
- Construction d'appels à des fonctions avec `{rlang}`
- Utiliser esquisse dans d'autres langues
- Futurs développements

# esquisse



## Présentation du paquet

# esquisse



## C'est quoi ? Pour qui ?



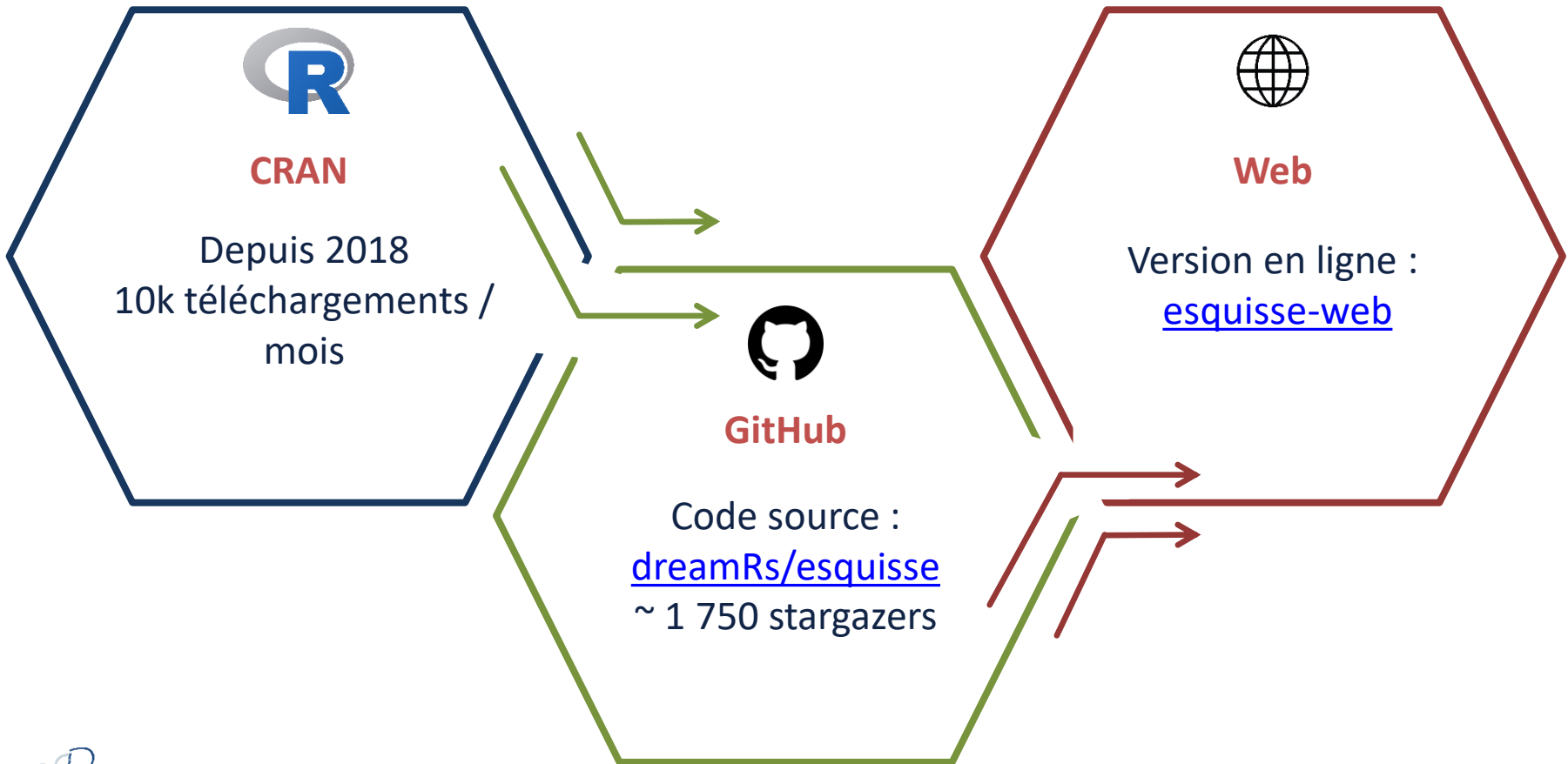
Une interface graphique pour créer des graphiques avec `{ggplot2}` et voir le code associé



- Débutants en ggplot2
- Utilisateurs occasionnels
- Pour explorer graphiquement des données
- Développeurs d'application pouvant utiliser esquisse en mode module

# esquisse

## Où le trouver ?



# esquisse

## Comment l'utiliser ?



En mode « addin » :

*# Lancer l'application :*

```
esquisse::esquisser()
```

*# En spécifiant un jeu de données :*

```
esquisse::esquisser(mes_donnees)
```

>\_

# esquisse

## Comment l'utiliser ?



Dans votre propre application, en utilisant les modules :

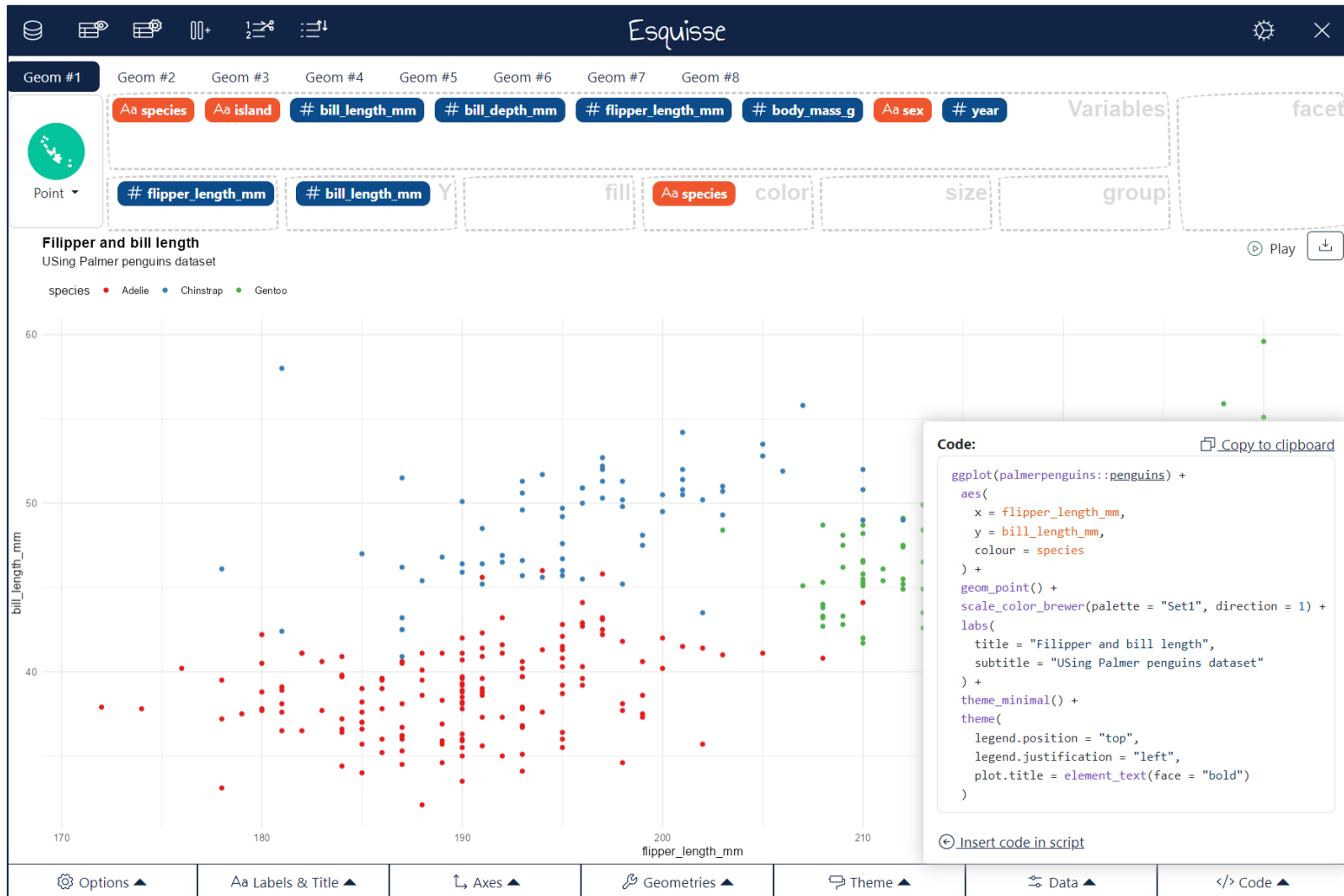
```
ui <- fluidPage(  
  theme = bs_theme_esquisse(),  
  esquisse_ui(  
    id = "esquisse",  
    container = esquisse_container(height = "700px")  
  )  
)  
  
server <- function(input, output, session) {  
  
  esquisse_server(  
    id = "esquisse",  
    data_rv= reactiveValues(data = mtcars, name = "mtcars")  
  )  
  
}
```





# esquisse

## Interface graphique



# esquisse

## Principales fonctionnalités



Outils  
d'interactions avec  
les données

Variables / noms  
de colonne

Variables à  
utiliser dans le  
graphique

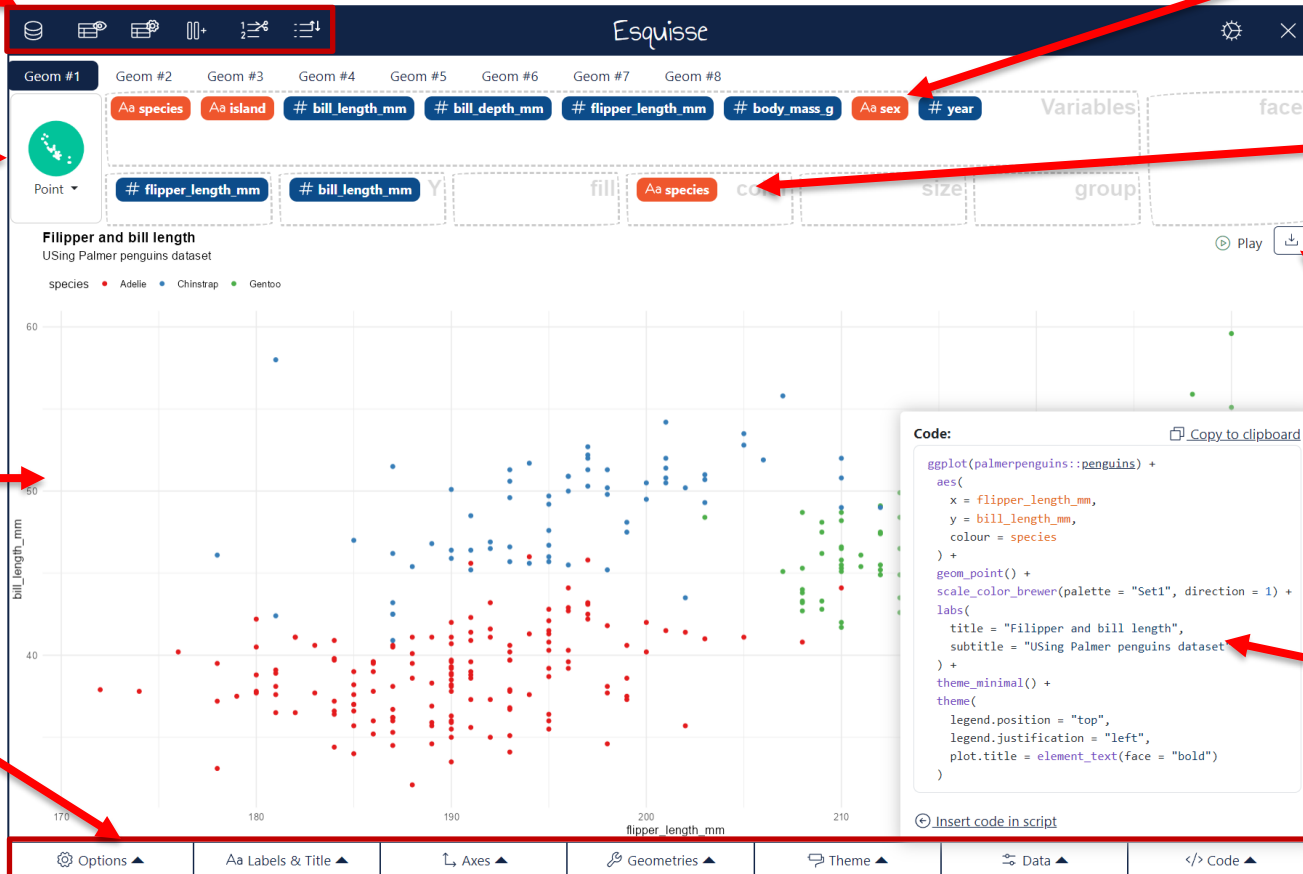
Menu pour  
exporter le  
graphique

Code généré pour  
produire le  
graphique

Type de  
graphique

Graphique

Paramètres  
pour modifier le  
graphique et  
pour obtenir le  
code



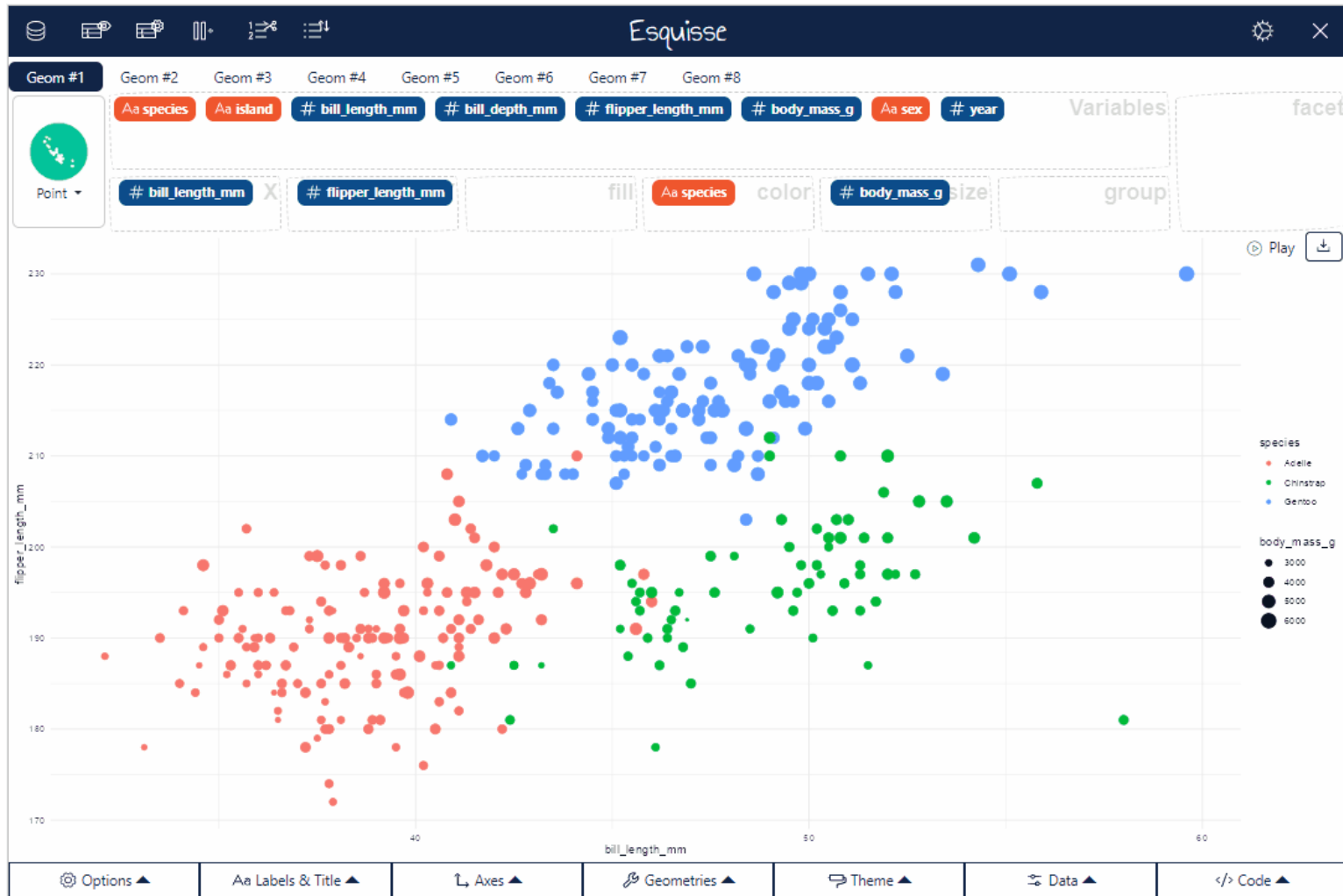
# esquisse



## **Nouveautés de la version 2.0.0**

# esquisse

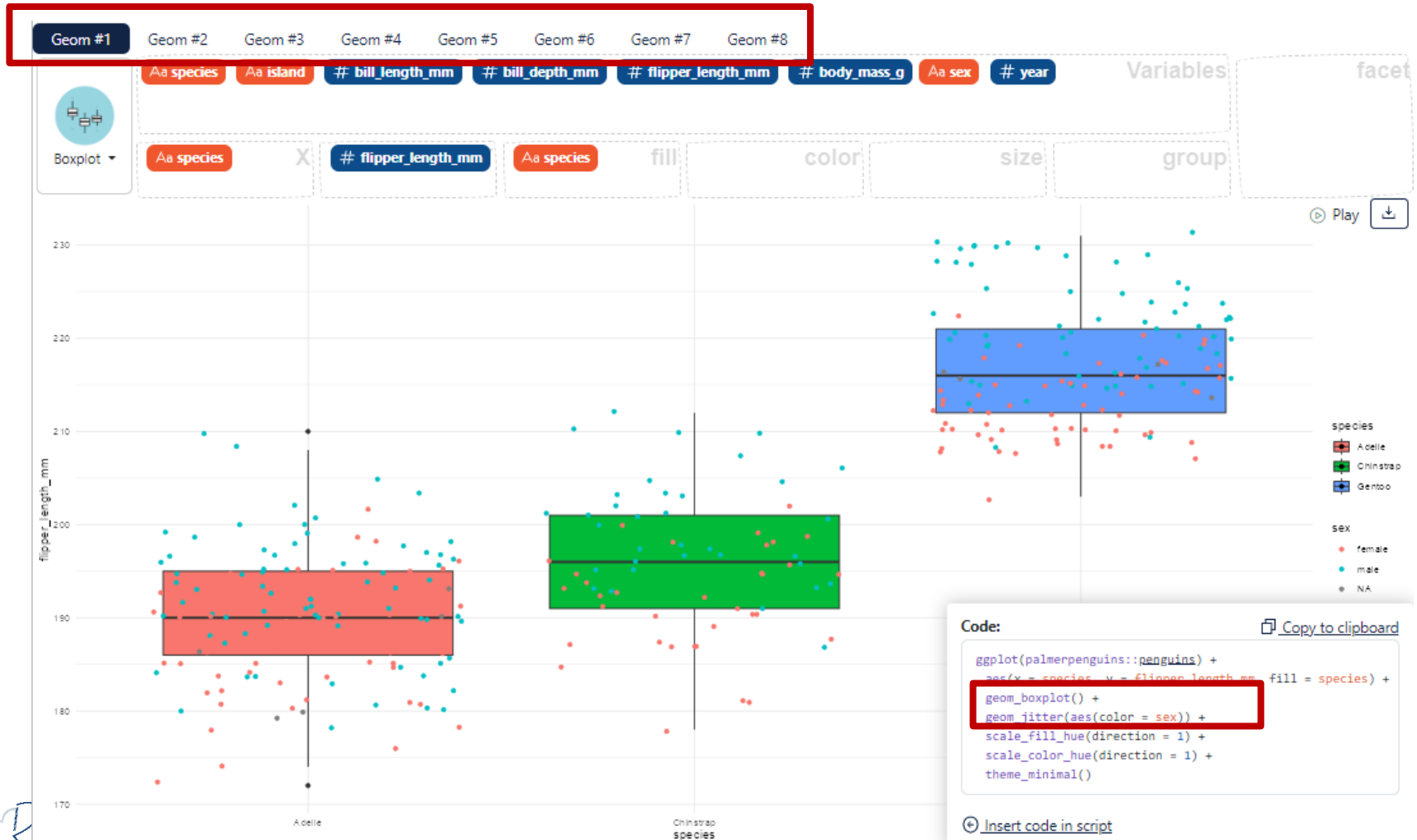
## Interactivité avec {plotly}



# esquisse



## Multiples fonctions « geom\_ »



# esquisse

## Modifications des données



Ajouter une nouvelle variable



Découper une variable continue en classes



Trier les niveaux d'une variable de type « factor »

# esquisse

## Ajouter une variable



### Create a new column

✕

New column name:

Group calculation by:

Select

▼

Enter an expression to define new column:

bill\_length\_mm / mean(bill\_depth\_mm)

❗ Click on a column name to add it to the expression:

Aa species

Aa island

# bill\_length\_mm


# bill\_depth\_mm


# flipper\_length\_mm


# body\_mass\_g

Aa sex

# year

❓ Choose a name for the column to be created or modified, then enter an expression before clicking on the button above to validate or on  to delete it.

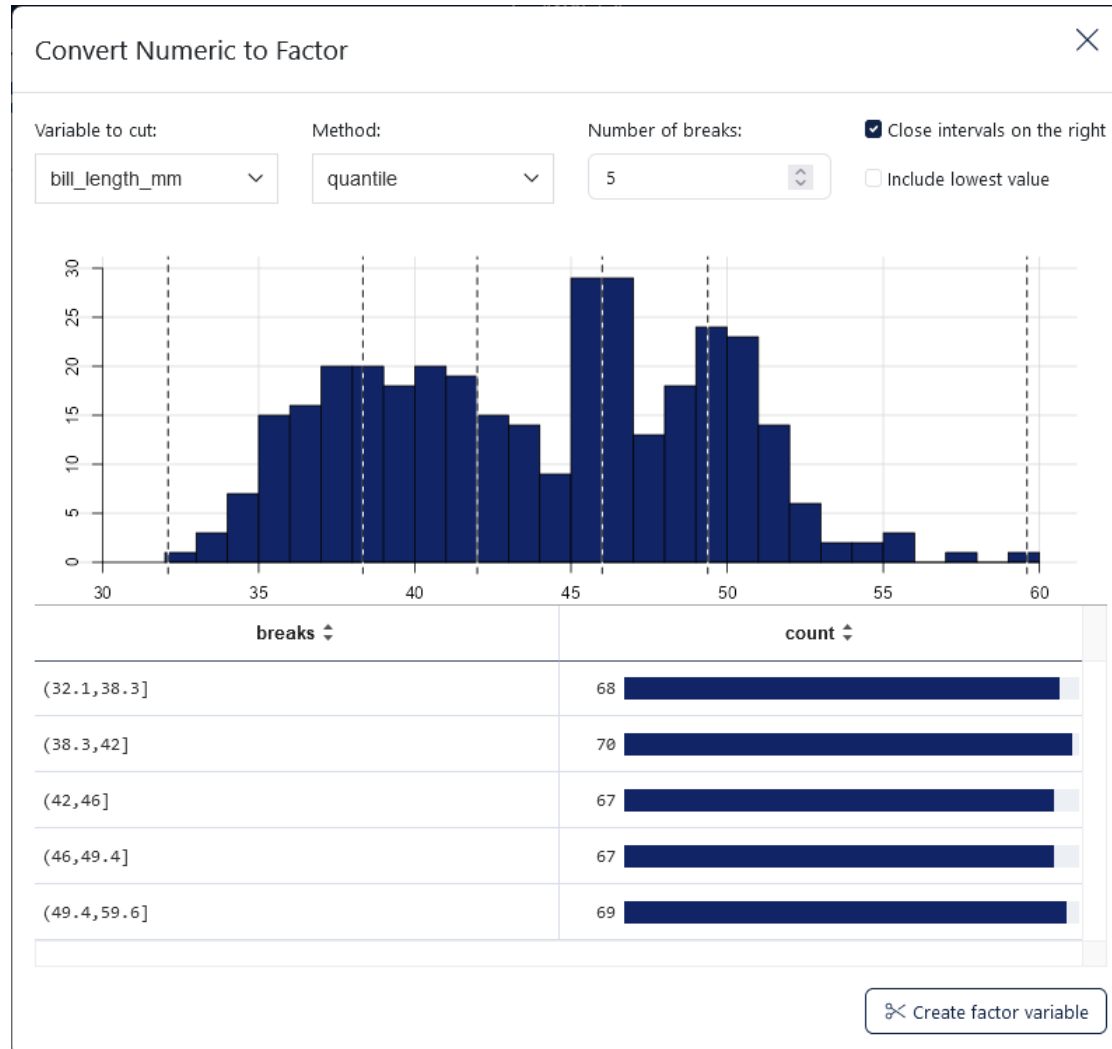
 Create column



# esquisse



## Découper une variable continue





# esquisse

## Trier un « factor »



Update levels of a factor

Factor variable to reorder:

species

Sort by levels

Sort by count

	Levels	Count
Adelie	152	<div></div>
Chinstrap	68	<div></div>
Gentoo	124	<div></div>

☐ Create a new variable (otherwise replaces the one selected)

Update factor variable

# esquisse



**Application web pour  
utiliser esquisse**

# esquisse

## Version web



Welcome to Esquisse

by [dreamRs](#)

{esquisse} is an R package, available on [CRAN](#), for creating graphs with {ggplot2}, this app allow you to use esquisse directly online, without having to install the package. Before creating a graph, you must first import data to be used to create a graph, or use a demo dataset.

[See the code on GitHub](#) 

Upload a file

Copy/Paste data

Import a GoogleSheet

Read from URL

Or use demo dataset

Upload a file:

Browse...

No file selected

Rows to skip before reading data:

n = 0

0.00

Decimal separator:

NA ,NA

UTF-8

Missing values character(s):

Encoding:

ⓘ

If several use a comma (,) to separate them

No file selected: You can import .csv, .txt, .xls, .xlsx, .rds, .fst, .sas7bdat, .sav files

×

No data.



# esquisse

## Version web



Possibilité d'utiliser un jeu de données de démonstration :

Upload a file   Copy/Paste data   Import a GoogleSheet   Read from URL

Or use demo dataset

Select a demo dataset:

### palmerpenguins: Palmer Archipelago (Antarctica) Penguin Data

Size measurements, clutch observations, and blood isotope ratios for adult foraging Adélie, Chinstrap, and Gentoo penguins observed on islands in the Palmer Archipelago near Palmer Station, Antarctica. Data were collected and made available by Dr. Kristen Gorman and the Palmer Station Long Term Ecological Research (LTER) Program.

👍 **Good for:** histogram, scatter plot, bar plot, boxplot

🔗 **Source:** <https://allisonhorst.github.io/palmerpenguins/>

➔ Select this dataset

### Fuel economy data from 1999 to 2008 for 38 popular models of cars

This dataset contains a subset of the fuel economy data that the EPA makes available on <https://fuelconomy.gov/>. It contains only models which had a new release every year between 1999 and 2008 - this was used as a proxy for the popularity of the car.

👍 **Good for:** histogram, scatter plot, bar plot, boxplot

🔗 **Source:** <https://ggplot2.tidyverse.org/reference/mpg.html>

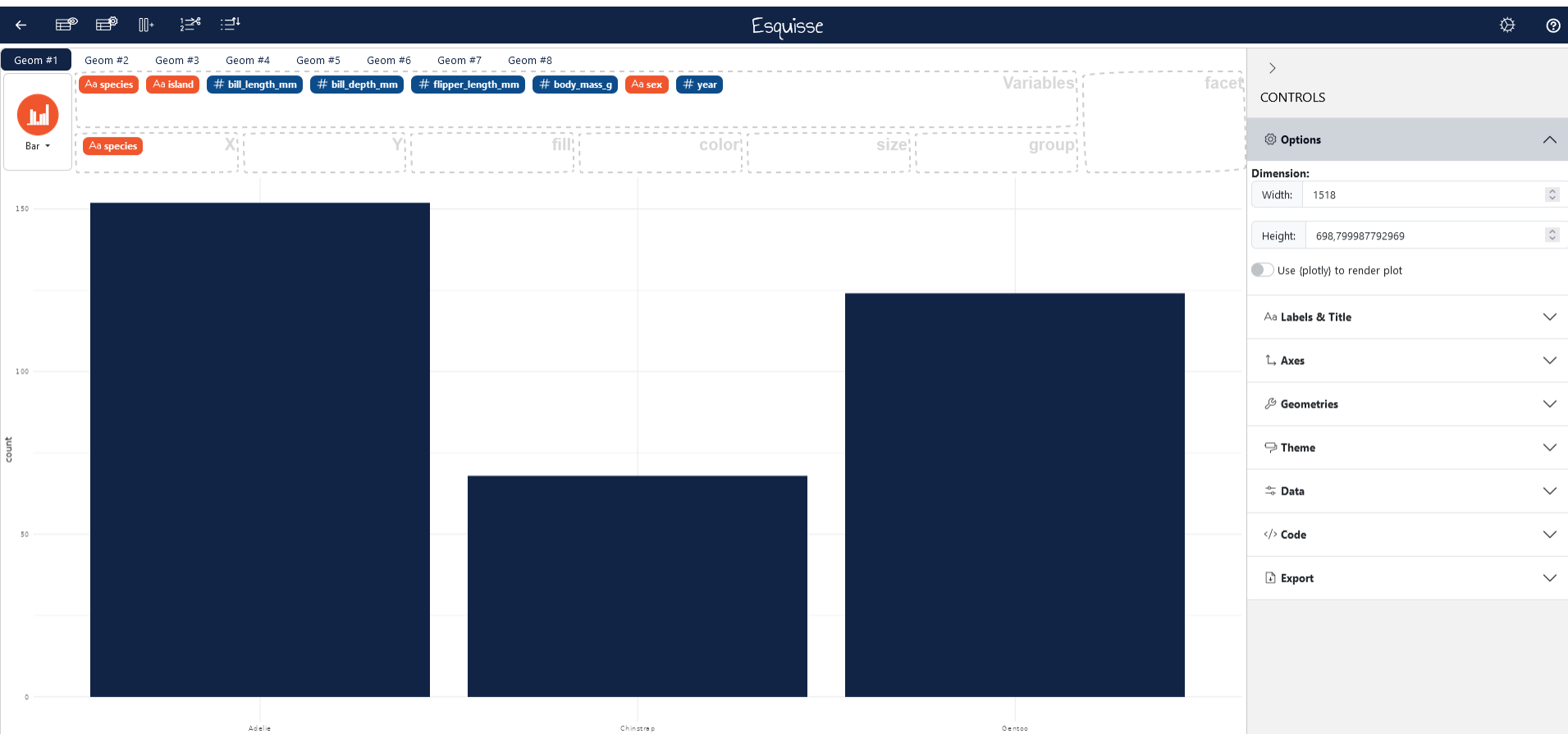
➔ Select this dataset

# esquisse

## Version web



Version en ligne avec les paramètres dans des menus en accordéons :



# esquisse

## Version web



Lien : <https://dreamrs.shinyapps.io/esquisse/>



# esquisse



## Génération de code avec {rlang}

# rlang

## Construire une expression

Ce que l'on veut et ce que l'on a

*# Code pour produire un graphique*

```
ggplot(penguins) +  
  aes(x = bill_length_mm, flipper_length_mm, color = species) +  
  geom_point(color = "#112466")
```

*# Paramètres dans des vecteurs ou listes*

```
data <- "penguins"  
mapping <- list("bill_length_mm", "flipper_length_mm", color = "species"  
")  
geom_fun <- "geom_point"  
geom_args <- list(color = "#112466")
```

>\_



# rlang

## Construire une expression

Définir une expression avec {rlang}

```
expr(ggplot(data))  
## ggplot(data)  
# "data" ne fait pas partie de l'expression, on veut la valeur  
expr(ggplot(!!data))  
## ggplot("penguins")  
# on ne veut pas data en tant que caractère mais en tant que symbole  
expr(ggplot(!!sym(data)))  
## ggplot(penguins)
```

>\_

# rlang

## Construire une expression

On procède de la même manière pour la mapping

```
expr(aes(mapping))  
## aes(mapping)  
expr(aes(!!mapping))  
## aes(list("bill_length_mm", "flipper_length_mm", color = "species"))  
expr(aes(!!!mapping))  
## aes("bill_length_mm", "flipper_length_mm", color = "species")  
expr(aes(!!!syms(mapping)))  
## aes(bill_length_mm, flipper_length_mm, color = species)
```

>\_

# rlang

## Construire une expression

On réunit nos deux expressions :

```
gg_aes_call <- call2(  
  "+",  
  expr(ggplot(!!sym(data))),  
  expr(aes(!!!syms(mapping)))  
)  
gg_aes_call
```

```
## ggplot(penguins) + aes(bill_length_mm, flipper_length_mm, color = sp  
ecies)
```

>\_

# rlang

## Construire une expression

Et on ajoute le geom :

```
geom_fun <- "geom_point"
geom_args <- list(color = "#112466")

gg_aes_geom_call <- call2(
  "+",
  gg_aes_call,
  call2(geom_fun, !!!geom_args)
)
gg_aes_geom_call

## ggplot(penguins) +
##   aes(bill_length_mm, flipper_length_mm, color = species) +
##   geom_point(color = "#112466")
```

>\_

# rlang

## Construire une expression

On peut afficher l'arbre de syntaxe abstraite (AST) de notre expression :

```
lobstr::ast(!!gg_aes_geom_call)
```

```
## ── `+`  
## ── ── `+`  
## ── ── ── ggplot  
## ── ── ── ── penguins  
## ── ── ── aes  
## ── ── ── ── bill_length_mm  
## ── ── ── ── flipper_length_mm  
## ── ── ── ── color = species  
## ── ── ── geom_point  
## ── ── ── ── color = "#112466"
```

>\_

# rlang

## Construire une expression

Il est possible d'extraire des éléments de l'expression et de les modifier :

```
gg_aes_geom_call[[3]]  
## geom_point(color = "#112466")  
gg_aes_geom_call[[3]]$size <- 4  
  
gg_aes_geom_call  
## ggplot(penguins) + aes(bill_length_mm, flipper_length_mm, color = species) +  
##      geom_point(color = "#112466", size = 4)
```

>\_

# rlang

## évaluer une expression

L'expression finale peut être évaluée pour obtenir le graphique :

```
eval(gg_aes_geom_call)
eval(
  gg_aes_geom_call,
  env(penguins = subset(penguins, species == "Adelie"))
)
```

>\_

# esquisse

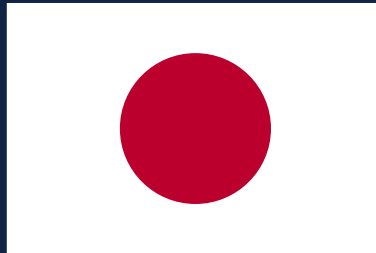
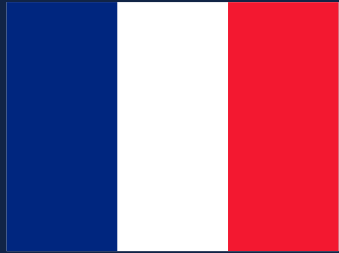


## Internationalisation



# esquisse

## 14 langues disponibles



# Changer la langue

On utilise la fonction `set_i18n()` :

```
library(esquisse)

# Choix de La Langue (code ISO 3166-2 du pays)
set_i18n("kr")

# Puis on lance esquisse comme d'habitude
esquisser()
```



Il faut déclarer la langue à utiliser avant de lancer l'application



Une évolution possible sera la possibilité de changer la langue directement dans l'interface (comme dans la version web)

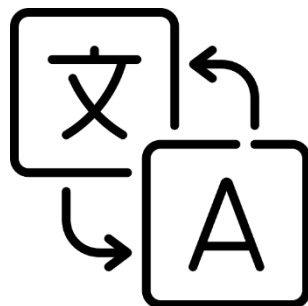
# Changer la langue

Comment cela fonctionne ?

- Une fonction `i18n()` qui prend en argument un libellé à traduire
- Un dictionnaire (un par langue) avec la correspondance entre le libellé (en anglaise) et la traduction à utiliser

Exemple avec le fichier de traduction français :

<https://github.com/dreamRs/esquisse/blob/master/inst/i18n/fr.csv>



**Contributions bienvenues !**  
(nouvelles traductions ou revue  
de celles déjà présentes)

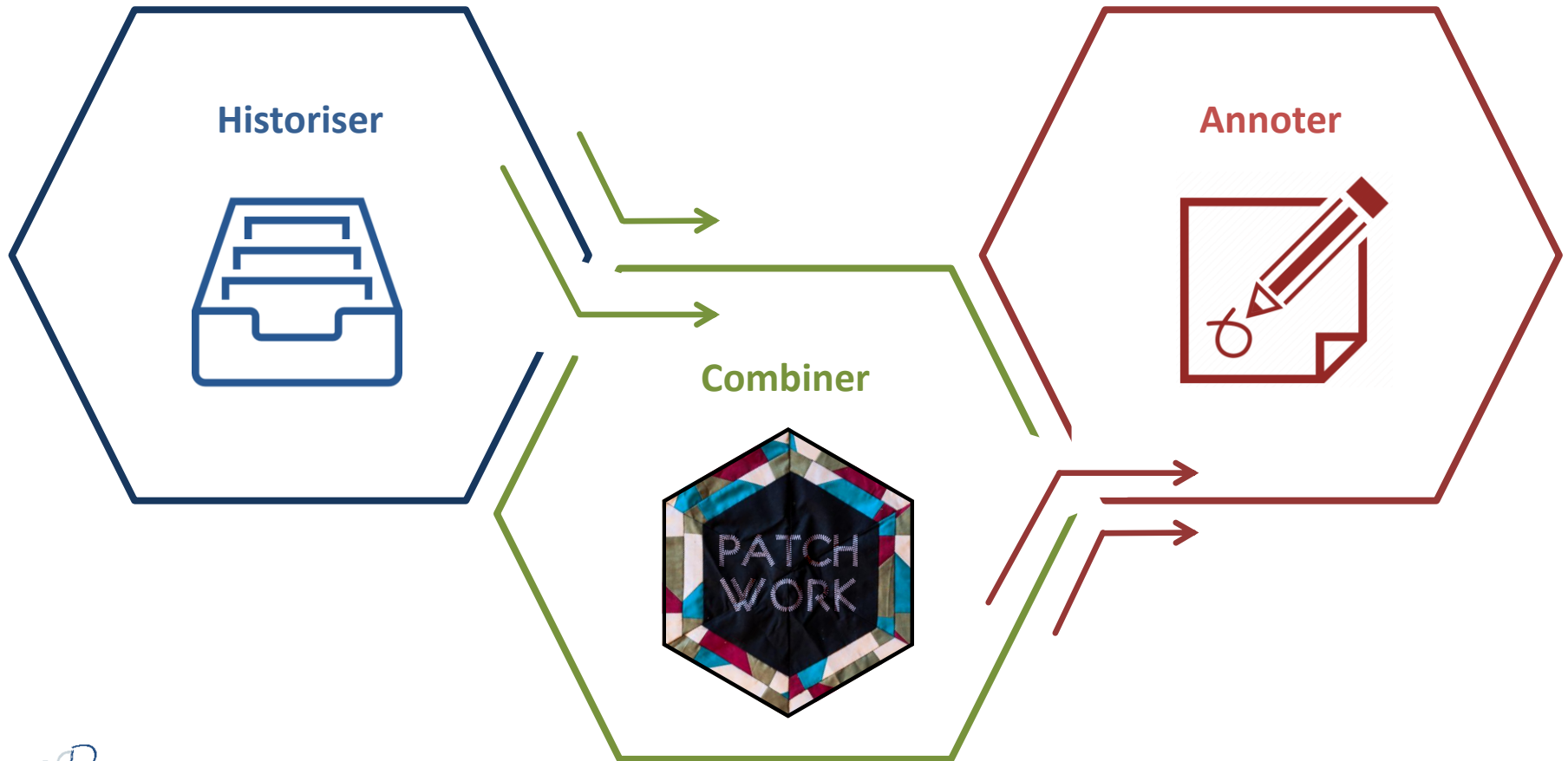
# esquisse



## Futurs développements

# esquisse

## The future



# esquisse

## Historiser



Enregistrer des graphiques réalisés lors d'une même session

```
RStudio Source Editor
Untitled9*
Source on Save
Run
Source

1
2
3
4 # Plot 1 -----
5
6 ggplot(palmerpenguins::penguins) +
7   aes(x = bill_length_mm, fill = species) +
8   geom_histogram()
9
10
11 # Plot 2 -----
12
13 ggplot(palmerpenguins::penguins) +
14   aes(x = species) +
15   geom_bar(fill = "#0d1c37")
16
17
18 # Plot 3 -----
19
20 ggplot(palmerpenguins::penguins) +
21   aes(x = bill_length_mm, flipper_length_mm, color = species) +
22   geom_point() +
23   labs(
24     title = "Title"
25   )
26
27
28 # Plot 4 -----
29
30 ggplot(palmerpenguins::penguins) +
31   aes(x = bill_length_mm, fill = species) +
32   geom_histogram()
```



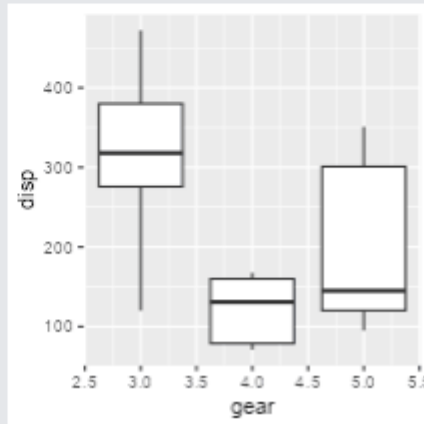
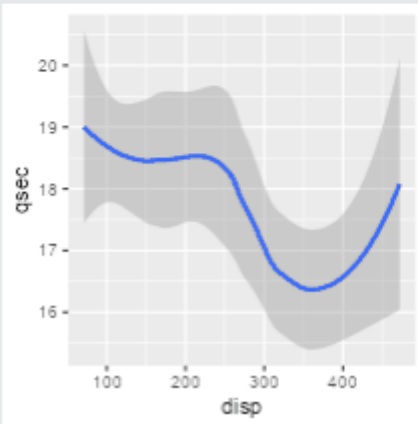
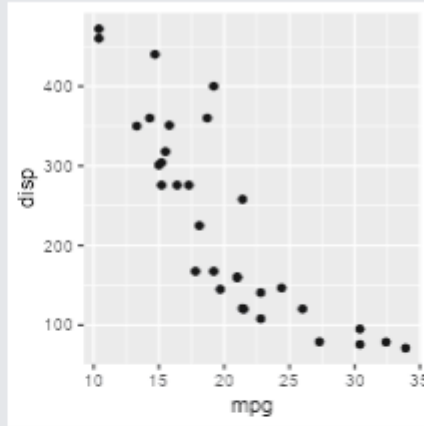
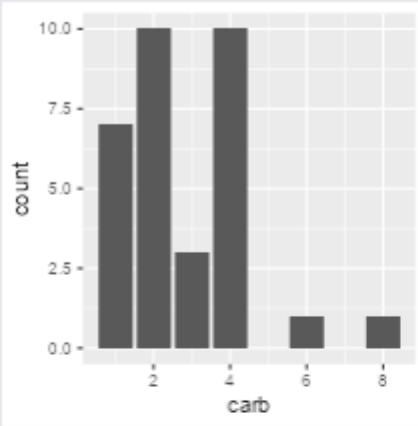
Export d'un script R contenant  
l'ensemble des codes pour générer  
les graphiques



Export des graphiques au format  
PNG (zip) ou PDF ...

# esquisse

## Combiner

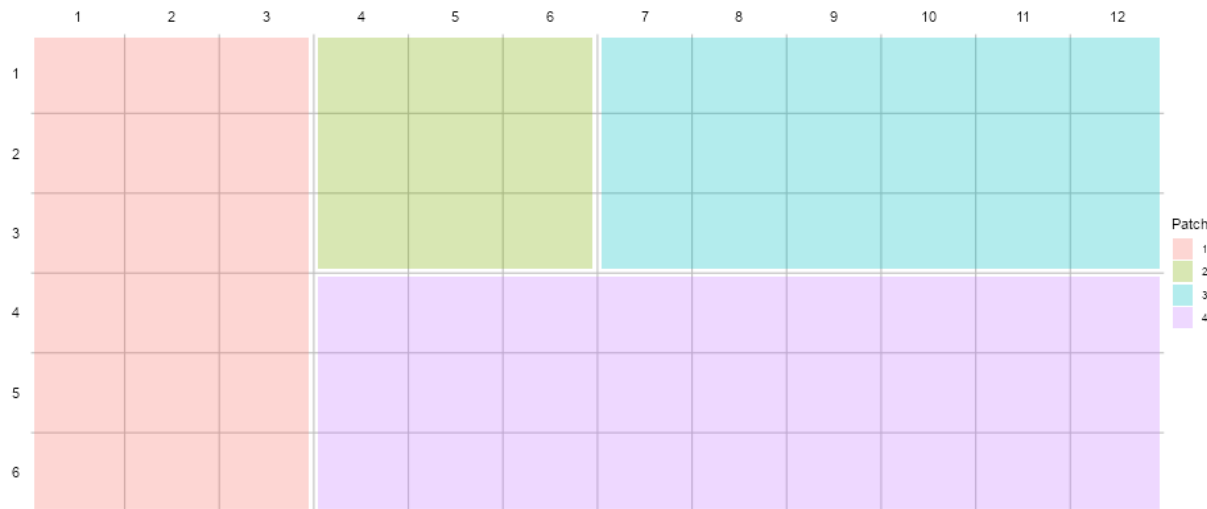


# esquisse

## Combiner



Pour obtenir un « design » utilisable avec le package {patchwork} :



Voir <https://patchwork.data-imaginist.com/reference/area.html>

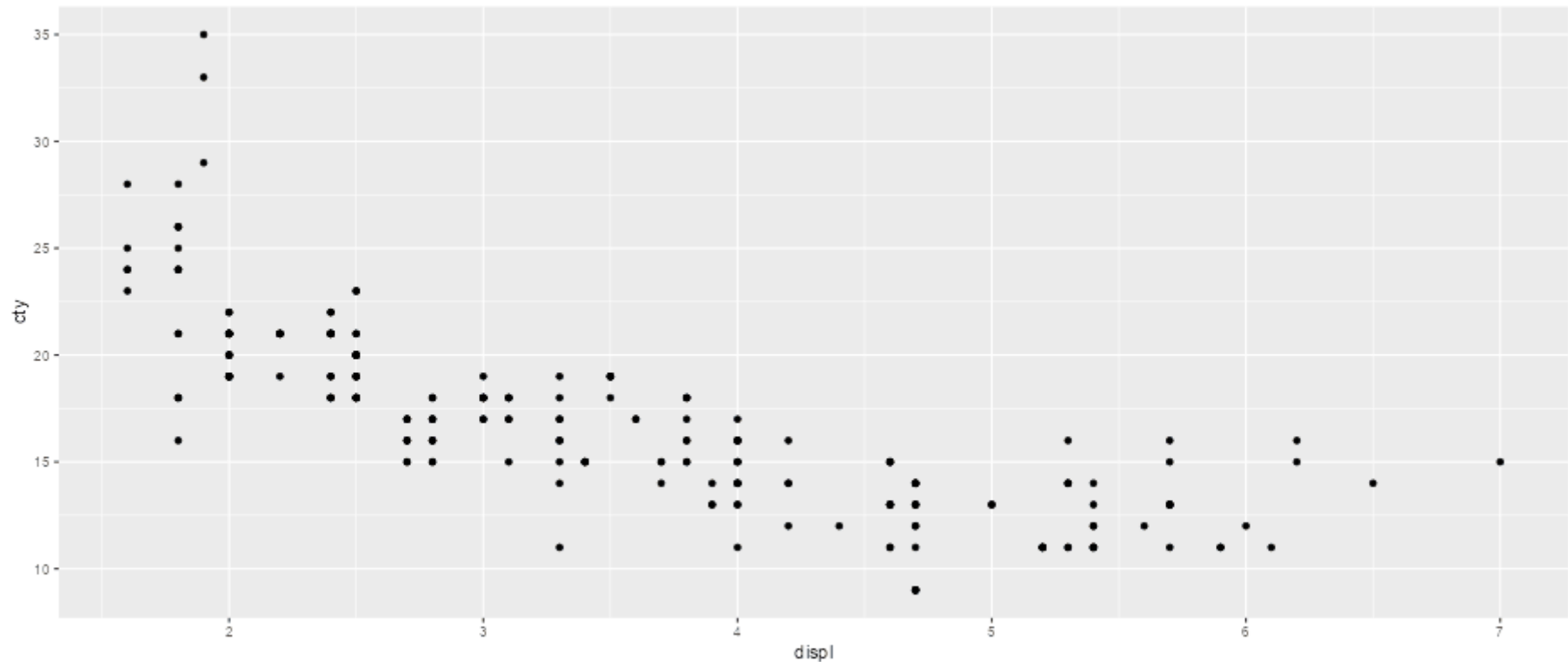


# esquisse

## Annoter



Add an annotation to a plot



Add annotation

# Merci



## Contact

Twitter : [@dreamrs\\_fr](https://twitter.com/dreamrs_fr)

Galerie Shiny : <http://shinyapps.dreamrs.fr/>

Site web : <https://www.dreamrs.fr/>

GitHub : <https://github.com/dreamRs>