

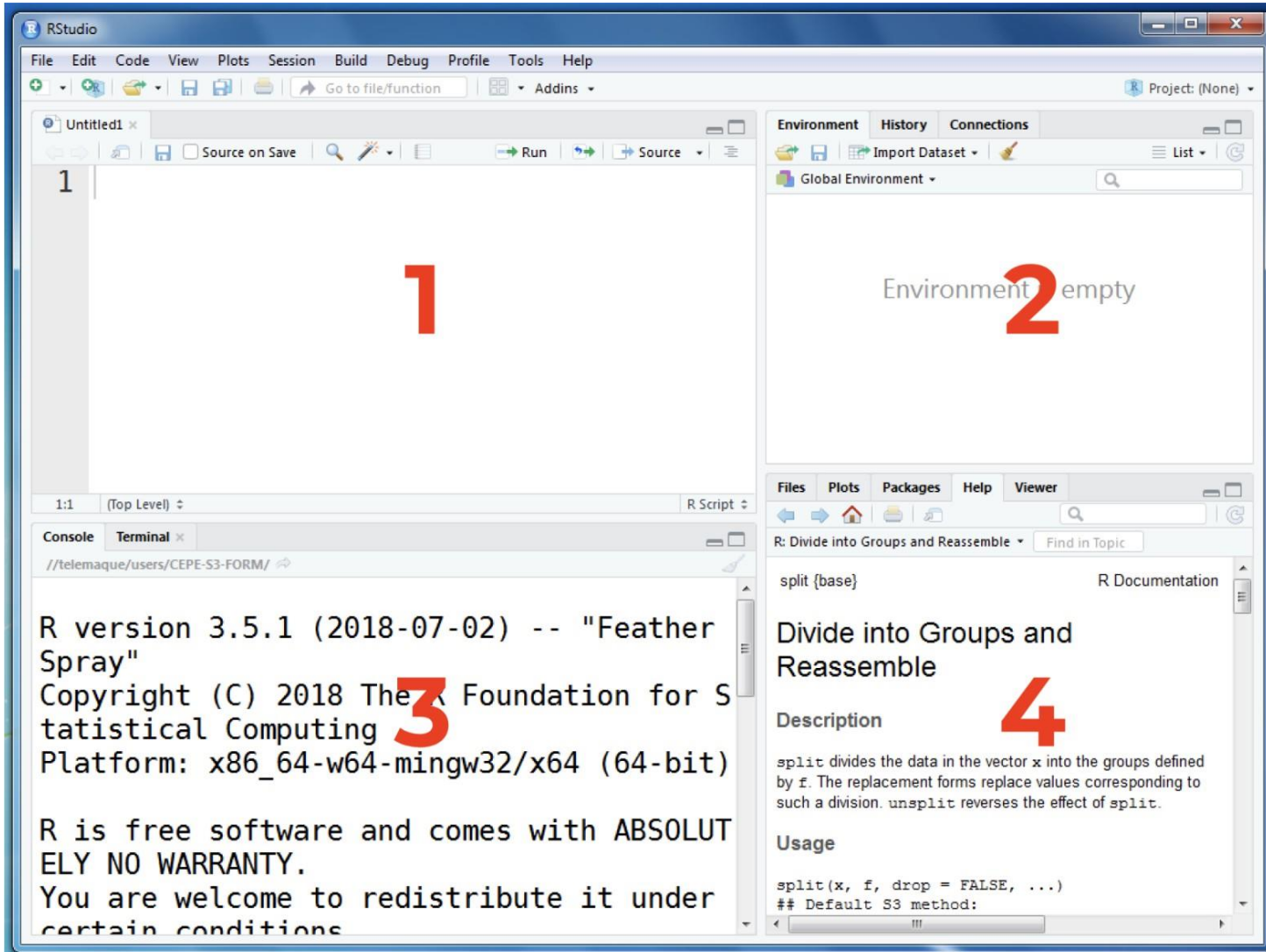
# Introduction à R

# logiciel R : gratuit et « open-source »

- **Utilisations principales**
  - Analyse statistique
  - Visualisation de données
  - Développement de modèles prédictifs
- **Communauté et extensions**
  - Communauté active et collaborative
  - Plus de 15,000 packages disponibles sur le CRAN (Comprehensive R Archive Network)
  - Extensions pour des domaines spécifiques (bio-informatique, économétrie, etc.)
- **Interfaçage et intégration**
  - Compatible avec d'autres langages (C, C++, Python)
  - Peut être utilisé dans divers IDEs (RStudio, Jupyter Notebook)
- **Visualisation de données**
  - Création de graphiques avancés
  - Support pour ggplot2, plotly et d'autres bibliothèques graphiques
- **Recherche et enseignement**
  - Largement adopté dans la recherche académique
  - Ressources pédagogiques abondantes pour l'apprentissage
- **Réplicabilité et transparence**
  - Code source ouvert pour une science reproductible
  - Documentation exhaustive et tutoriels



# L'interface de RStudio

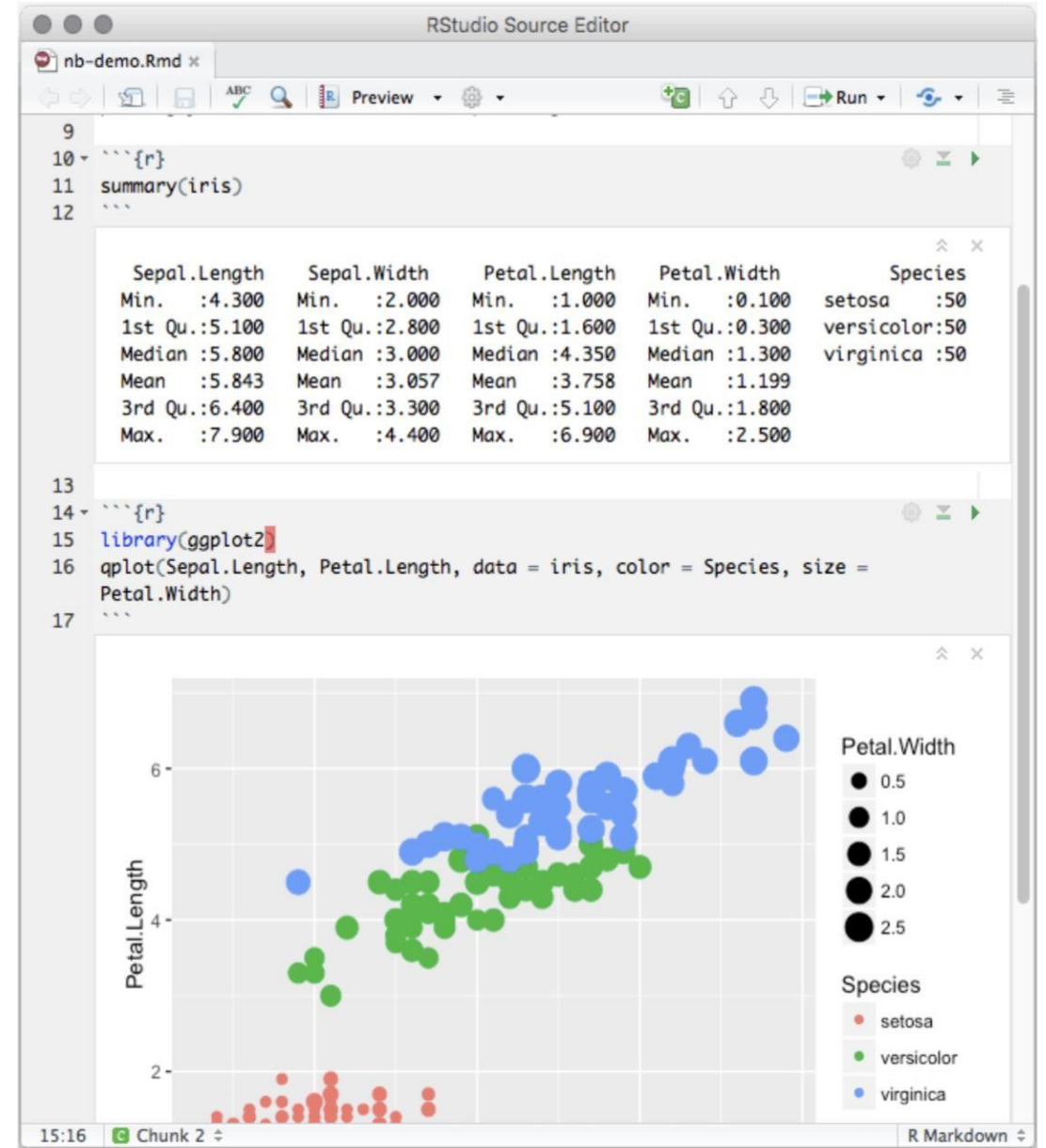
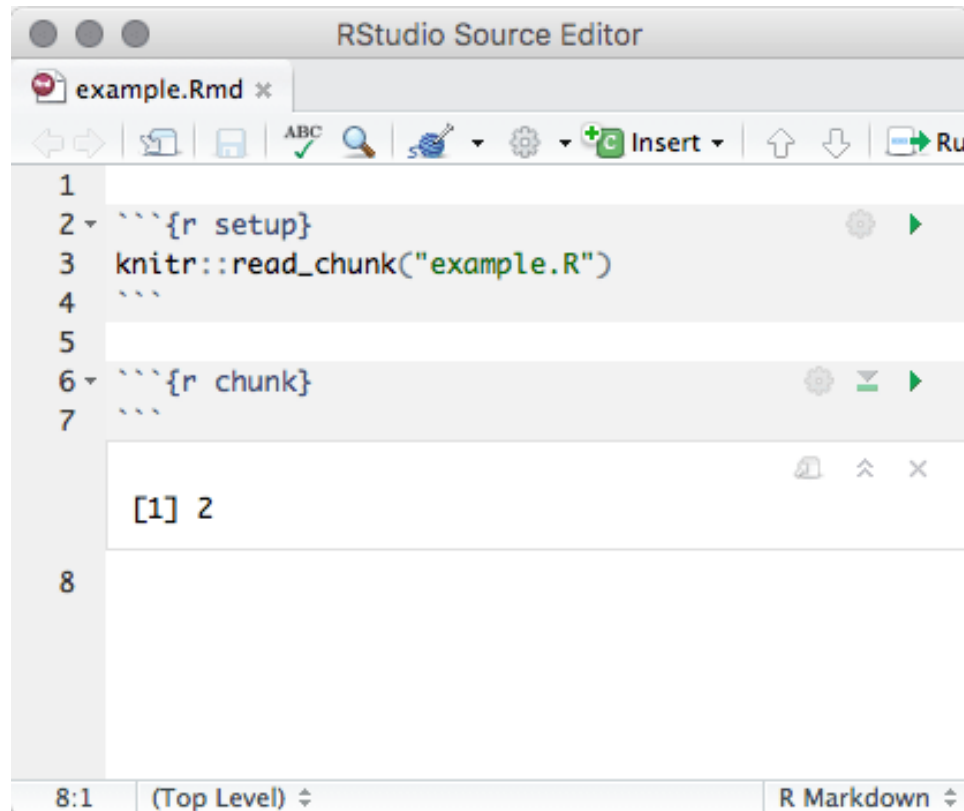


**RStudio** est divisé en **4 quadrants** :

1. Éditeur de texte, de codes, etc.
2. Espace de travail, historique, importation, etc.
3. Console
4. Visualisations, aide

# R Notebook

- Document R Markdown interactif
- Contient des blocs de code exécutables
- Exécution indépendante et interactive des codes
- Résultats immédiatement affichés sous le code



# Notebook

## A: Anatomy of a Notebook / R Markdown document

```
---
title: "R Notebook"
output: html_notebook
---
```

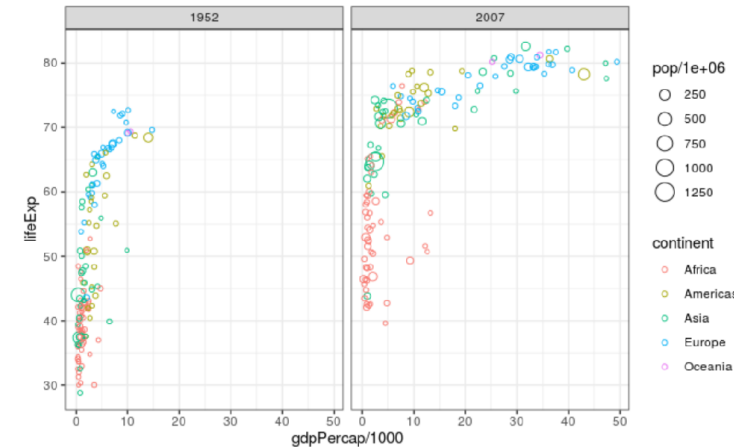
```
## Aim
To compare global life expectancy between 1952 and 2007.
```

```
## Data
"Gapminder" project data.
```

```
## Plot the data
This plot nicely shows the distribution of the data. I added a filter to exclude
`r gapminder %>% filter(year == 1952 & gdpPercap > 15000) %>% pull(country)`,
which is an extreme outlier. See here:
```

```
--- {r}
gapminder %>%
  filter(year == 1952 & gdpPercap > 15000)
---
```

```
--- {r}
library(gapminder)
library(dplyr)
gapminder %>%
  filter(year == 2007 | year == 1952) %>%
  filter(gdpPercap < 15000) %>%
  ggplot(aes(x = gdpPercap/1000, #divide by 1000 to tidy the x-axis
             y = lifeExp,
             colour = continent,
             size = pop/1e6)) +
  geom_point(shape = 1) +
  theme_bw() +
  facet_wrap(~year)
---
```



Chunk 2 R Markdown

YAML header

Headings (h2)

In-line code

Code chunk

- begin {r}
- end }
- options
- run chunk
- run all chunks to here

Output

R Markdown document type

## B: Rendered preview

R Notebook

Aim

To compare global life expectancy between 1952 and 2007.

Data

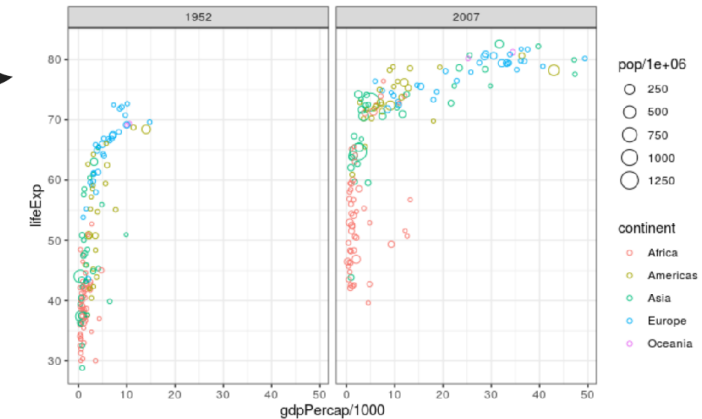
Gapminder project data.

Plot the data

This plot nicely shows the distribution of the data. I added a filter to exclude Kuwait, which is an extreme outlier. See here:

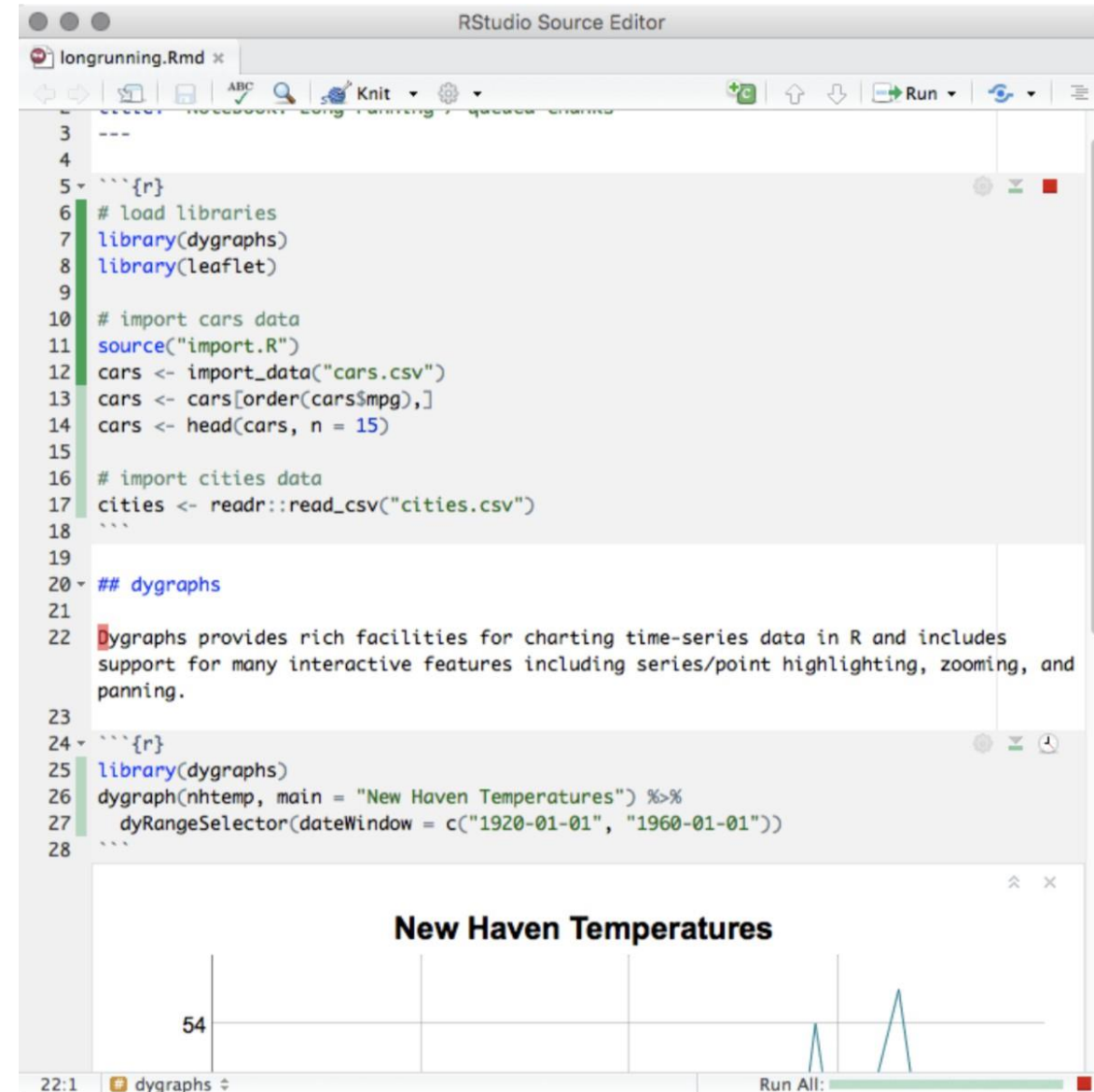
```
gapminder %>%
  filter(year == 1952 & gdpPercap > 15000)
```

```
library(gapminder)
library(dplyr)
gapminder %>%
  filter(year == 2007 | year == 1952) %>%
  filter(gdpPercap < 15000) %>%
  ggplot(aes(x = gdpPercap/1000, #divide by 1000 to tidy the x-axis
             y = lifeExp,
             colour = continent,
             size = pop/1e6)) +
  geom_point(shape = 1) +
  theme_bw() +
  facet_wrap(~year)
```



# Les chunks

- **Définition** : Un chunk, ou bloc de code, est une portion de code R encadrée par des marqueurs spécifiques dans un document R Markdown ou R Notebook.
- **Fonctionnalité** : Permet l'exécution de code R directement dans le document, offrant ainsi des résultats interactifs et immédiats.
- **Structure** : Délimité par des signes `{r}` au début et des signes `}` à la fin. Les options de chunk peuvent être ajoutées entre les accolades pour contrôler le comportement de l'exécution, comme l'affichage des résultats, des messages, des avertissements, etc.
- **Interactivité** : Les chunks permettent aux utilisateurs de tester des variations de code rapidement sans avoir à recompiler l'ensemble du document, facilitant ainsi le développement itératif et l'exploration de données.
- **Intégration** : Les résultats de l'exécution, qu'ils soient des graphiques, des tableaux ou du texte simple, sont affichés juste après le chunk de code, intégrant ainsi analyses et conclusions dans un flux de travail fluide et documenté.



The screenshot shows the RStudio Source Editor with a file named 'longrunning.Rmd'. The code is organized into two R chunks, each enclosed in `{r}` markers. The first chunk (lines 5-18) loads the 'dygraphs' and 'leaflet' libraries, imports 'cars' data from 'import.R', and reads 'cities' data from 'cities.csv'. The second chunk (lines 24-28) loads 'dygraphs' and creates a time-series plot titled 'New Haven Temperatures' using 'dygraph()' and 'dyRangeSelector()'. The plot is displayed below the second chunk, showing a line graph with a y-axis value of 54 and a date range from 1920-01-01 to 1960-01-01. The status bar at the bottom indicates '22:1' and 'dygraphs'.

```
3 ---
4
5 {r}
6 # load libraries
7 library(dygraphs)
8 library(leaflet)
9
10 # import cars data
11 source("import.R")
12 cars <- import_data("cars.csv")
13 cars <- cars[order(cars$mpg),]
14 cars <- head(cars, n = 15)
15
16 # import cities data
17 cities <- readr::read_csv("cities.csv")
18
19
20 ## dygraphs
21
22 Dygraphs provides rich facilities for charting time-series data in R and includes
23 support for many interactive features including series/point highlighting, zooming, and
24 panning.
25
26 {r}
27 library(dygraphs)
28 dygraph(nhtemp, main = "New Haven Temperatures") %>%
29   dyRangeSelector(dateWindow = c("1920-01-01", "1960-01-01"))
30
31 
```

New Haven Temperatures

54

22:1 dygraphs Run All:

# Les packages

- **Définition** : Un package en R est une collection de fonctions, de données et de code compilé qui sont étendus à partir du logiciel R de base, permettant aux utilisateurs d'effectuer des tâches spécialisées et complexes.
- **Installation** : Les packages peuvent être installés depuis le Comprehensive R Archive Network (CRAN), GitHub, ou d'autres sources, en utilisant des commandes telles que **install.packages("nom\_du\_package")**.
- **Utilisation** : Après installation, un package peut être chargé dans l'environnement R avec la commande **library(nom\_du\_package)**.
- **Exemples courants** :
  - **ggplot2** pour la visualisation de données avancée.
  - **dplyr** pour la manipulation de données.
- **Mise à jour** : Les packages peuvent être mis à jour avec la commande **update.packages()** pour assurer la compatibilité et l'accès aux dernières fonctionnalités.
- **Communauté** : Le développement de packages est soutenu par une large communauté de développeurs et de scientifiques qui contribuent constamment à l'enrichissement de l'écosystème R.