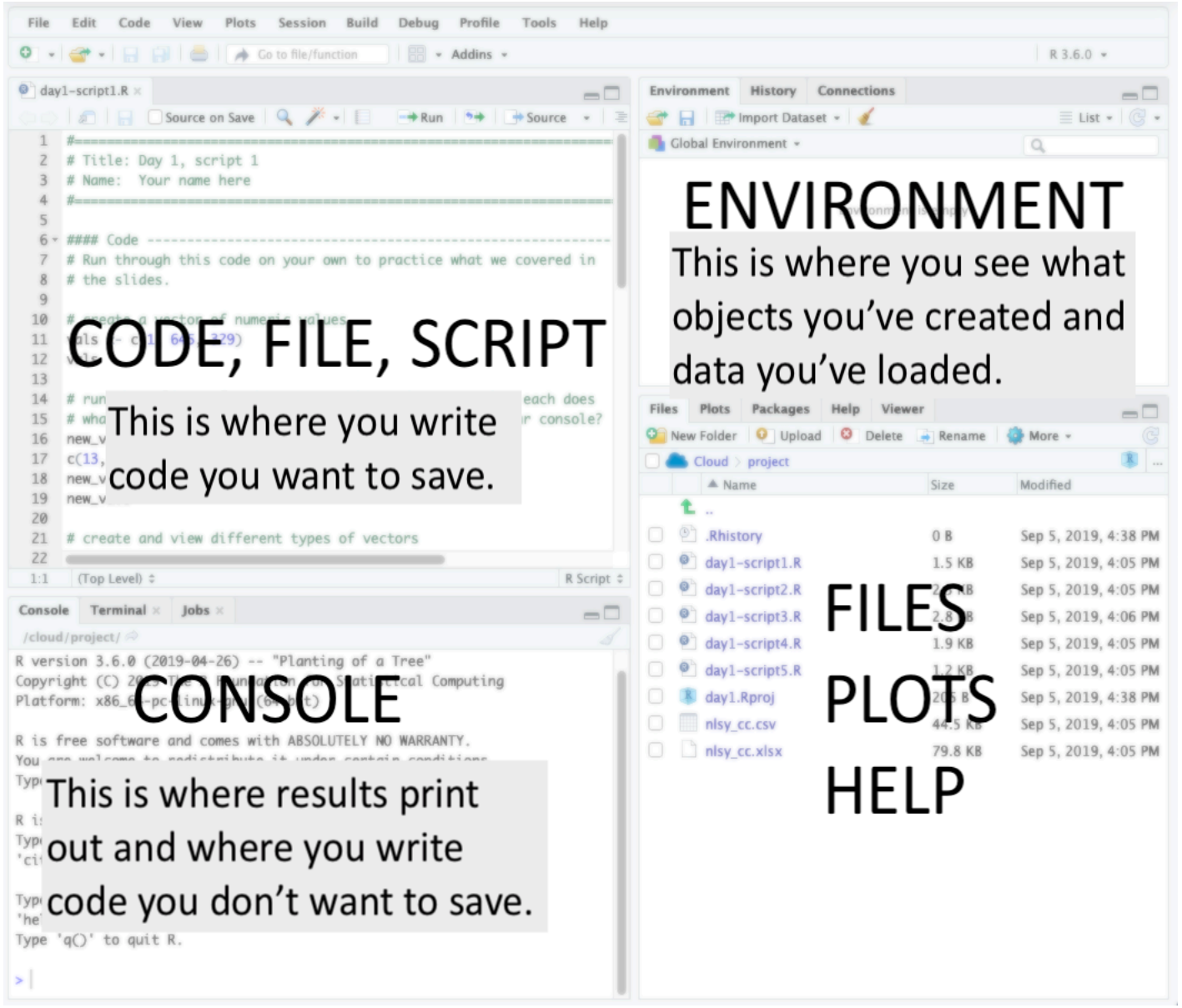
Introduction to R

Using R Studio

An IDE for R

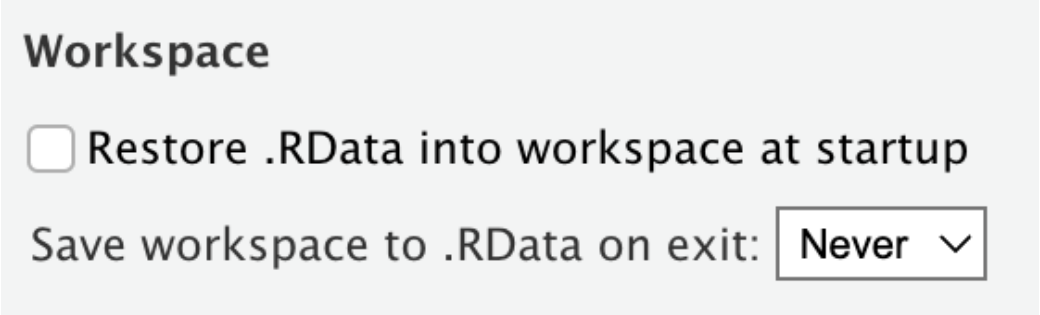
An *integrated development environment* is software that makes coding easier

- see objects you've imported and created
- autocomplete
- syntax highlighting
- run part or all of your code


CODE, FILE, SCRIPT
This is where you write code you want to save.
ENVIRONMENT
This is where you see what objects you've created and data you've loaded.
CONSOLE
This is where results print out and where you write code you don't want to save.
FILES
PLOTS
HELP

Start fresh

- If you have used R previously, an old workspace may still be active when you open RStudio
- You always want to start with a fresh session
- Go to Tools -> Global Options, and under General, change these settings:



Tip

Now, you can just quit and restart RStudio if something goes wrong! You can also go to Session -> Restart R to clear your session.

Demo

Packages

- Some functions are built into R
  - `mean()`, `lm()`, `table()`, etc.
- They actually come from built-in packages
  - `base`, `stats`, `graphics`, etc.
- Anyone (yes, *anyone*) build their own package to add to the functionality of R
  - `{ggplot2}`, `{dplyr}`, `{data.table}`, `{survival}`, etc.

[1]

1. Image from Zhi Yang

Packages

- You have to **install** a package once[1]

```
1 install.packages("survival")
```

- You then have to **load** the package every time you want to use it

```
1 library(survival)
```

1. Actually, with every new major R release, but we won't worry about that.

Packages

"You only have to buy the book once, but you have to go get it out of the bookshelf every time you want to read it."

```
1 install.packages("survival")
2 library(survival)
3 survfit(...)
```

Several days later...

```
1 library(survival)
2 coxph(...)
```

Package details

- When you use `install.packages`, packages are downloaded from CRAN (The Comprehensive R Archive Network)
  - This is also where you downloaded R
- Packages can be hosted lots of other places, such as Bioconductor (for bioinformatics), and Github (for personal projects or while still developing)
- The folks at CRAN check to make things “work” in some sense, but don’t check on the statistical methods...
  - But because R is open-source, you can always read the code yourself
- Two functions from different packages can have the same name... if you load them both, you may have some trouble

Demo

Using R

The biggest difference between R and Stata is that R can have many different objects in its environment

- datasets, numbers, figures, etc.
- you have to be explicit about storing and retrieving objects
  - e.g., what dataset a variable belongs to

R uses `<-` to store objects in the environment

I call this the “assignment arrow”

Now `vals` holds those values

> ⚠ **Warning**
>
> No assignment arrow means that the object will be printed to the console (and lost forever!)

Objects

We can retrieve those values by running just the name of the object

```
[1]   1 645 329
```

We can also perform operations on them using functions like `mean()`

```
[1] 325
```

If we want to keep the result of that operation, we need to use `<-` again

Types of data (*classes*)

We could also create a character *vector*:

```
[1] "dog"   "cat"   "rhino"
```

Or a *logical* vector:

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

> **Tip**
>
> We'll see more options as we go along!

Types of objects

We created *vectors* with the `c()` function (`c` stands for concatenate)

We could also create a *matrix* of values with the `matrix()` function:

```
     [,1] [,2] [,3]
[1,]  234   12  183
[2,] 7456  654  753
```

Indices

The numbers in square brackets are *indices*, which we can use to pull out values:

```
[1] "cat"
```

We can pull out rows or columns from matrices:

```
[1] 7456  654  753
```

```
[1]  234 7456
```

Demo