



# Finanzas en R

Good practices


Sebastián Egaña Santibáñez 


Nicolás Leiva Díaz 

---

## Enlaces del profesor

 <https://semana.netlify.app>

 <https://github.com/sebaegana>

 <https://www.linkedin.com/in/sebastian-egana-santibanez/>

---

## Instalación de paquetes en R

### ¿Cómo se instalan paquetes en R?

Los paquetes amplían las capacidades de R y se descargan desde CRAN.

```
 install.packages("dplyr")
```

### Cargar un paquete

Instalar un paquete no es lo mismo que cargarlo:

```
library(dplyr)
```

Buenas prácticas al instalar paquetes

- Instala paquetes solo una vez por entorno
- Usa `library()` dentro de scripts o notebooks
- En proyectos, utiliza `renv` para controlar versiones
- Evita instalar paquetes dentro de funciones

## Renv

### ¿Qué es `renv`?

`renv` es una herramienta para gestionar entornos de proyectos en R. Permite aislar las versiones de paquetes usados en un proyecto, asegurando que el análisis sea reproducible incluso si cambian las versiones globales de R o de los paquetes.

### ¿Por qué usar `renv`?

- Asegura que otros puedan ejecutar tu código con los mismos paquetes y versiones.
- Facilita colaborar en equipo.
- Útil para compartir proyectos o generar informes reproducibles.

## Paso a paso

1. Instalar `renv`

```
install.packages("renv")
```

2. Inicializar en tu proyecto

Esto crea un entorno `renv` en tu carpeta actual.

```
renv::init()
```

Esto hace:

- Crea una carpeta `renv/` donde se instalarán los paquetes del proyecto.
- Crea un archivo `renv.lock` que guarda las versiones exactas de los paquetes usados.

### 3. Instalar paquetes normalmente

Una vez iniciado renv, puedes instalar paquetes como siempre:

```
install.packages("dplyr")  
install.packages("ggplot2")
```

renv registrará automáticamente estas instalaciones en el archivo renv.lock

### 4. Guardar el estado del entorno Si ya tenías paquetes instalados y quieres capturar el entorno actual:

```
renv::snapshot()
```

Esto actualiza el archivo renv.lock con los paquetes y versiones actuales.

### 5. Restaurar un entorno en otra máquina Cuando otra persona descarga tu proyecto (por ejemplo, desde GitHub):

```
renv::restore()
```

## Archivos importantes

- renv.lock: archivo de texto con las versiones de los paquetes.
- renv/: carpeta con los paquetes instalados para este proyecto.

No incluyas renv/library/ en tu repositorio si usas Git (puedes agregarlo a .gitignore).

## Buenas prácticas

- Siempre correr renv::snapshot() luego de instalar o actualizar paquetes.
- Incluir el archivo renv.lock en tu repositorio para asegurar la reproducibilidad.
- Usar renv::restore() al comenzar a trabajar en un proyecto de otra persona.

## Referencias

- <https://cran.r-project.org/web/packages/tesseract/vignettes/intro.html>
- [https://blog.djnavarro.net/posts/2023-06-16\\_tabulizer/](https://blog.djnavarro.net/posts/2023-06-16_tabulizer/)
- <https://crimebythenumbers.com/scrape-table.html>

# Instalación de Git

## ¿Qué es Git?

Git es un sistema de control de versiones que permite:

- Registrar cambios en el código
- Volver a versiones anteriores
- Trabajar en equipo sin sobrescribir trabajo

Es la base para trabajar con GitHub y flujos de CI/CD.

---

## Verificar si Git ya está instalado

Antes de instalar, revisa si Git ya está disponible en tu sistema:

```
git --version
```

Si aparece un número de versión, Git ya está instalado.

## Instalación según sistema operativo

### Windows

1. Descarga Git desde <https://git-scm.com/download/win>
2. Ejecuta el instalador
3. Acepta las opciones por defecto
4. Reinicia RStudio si estaba abierto

### macOS

Opción recomendada usando Homebrew:

```
brew install git
```

## Connect to Github

### Prerrequisitos

- Una cuenta en GitHub
- Git instalado en tu equipo (verificalo con `git --version`)
- RStudio (opcional, pero recomendado)
- Un proyecto de R ya inicializado (idealmente con `renv::init()`)

### Paso a paso

#### 1. Paquetes

Asegúrate de tener los paquetes necesarios

```
install.packages(c("usethis", "renv", "gh"))
```

#### 2. Inicializa Git en el proyecto

```
usethis::use_git()
```

Esto:


- Inicializa Git en la carpeta del proyecto
- Crea un `.gitignore` con entradas útiles (incluye `renv/library/`)
- Realiza un primer commit

#### 3. Crea el repositorio en GitHub desde R

```
usethis::use_github()
```

Esto:

- Crea el repositorio remoto en GitHub (usando tu cuenta)
- Conecta tu proyecto local al remoto (`git remote add origin`)
- Realiza push al repositorio

 Asegúrate de haber iniciado sesión en GitHub con un Personal Access Token (PAT). Si no lo tienes, puedes crearlo e instalarlo con:

```
usethis::create_github_token()
usethis::edit_r_environ()
```

Pega el token como:

GITHUB\_TOKEN=ghp\_tu\_token\_aquí

Luego reinicia R y corre `usethis::use_github()` de nuevo.

## Referencias:

- <https://rpubs.com/RonaldoAnticono/818156>
- <https://rfortherestofus.com/2021/02/how-to-use-git-github-with-r>
- <https://happygitwithr.com/rstudio-git-github.html>