Reproducing package environments

using renv to lock packages

2021-10-12

Once upon a time (6 months ago), a hero (you) wrote some code

```
add_rownames(data, var = "rowname")
```

But then!

Updates were made to the dplyr package:

```
Warning message:
`add_rownames()` was deprecated in dplyr 1.0.0.
Please use `tibble::rownames_to_column()` instead.
```

renv

renv is designed to improve **project-level reproducibility**

records and **restores** the packages used in a project

Successor to **packrat**

How does renv help?

1. **Each project gets it's own library (isolated)**
2. **The project library can be shipped with a self-contained lockfile,** renv.lock **(portable)**
3. renv.lock **can be restored with** renv::restore() **(reproducible)**

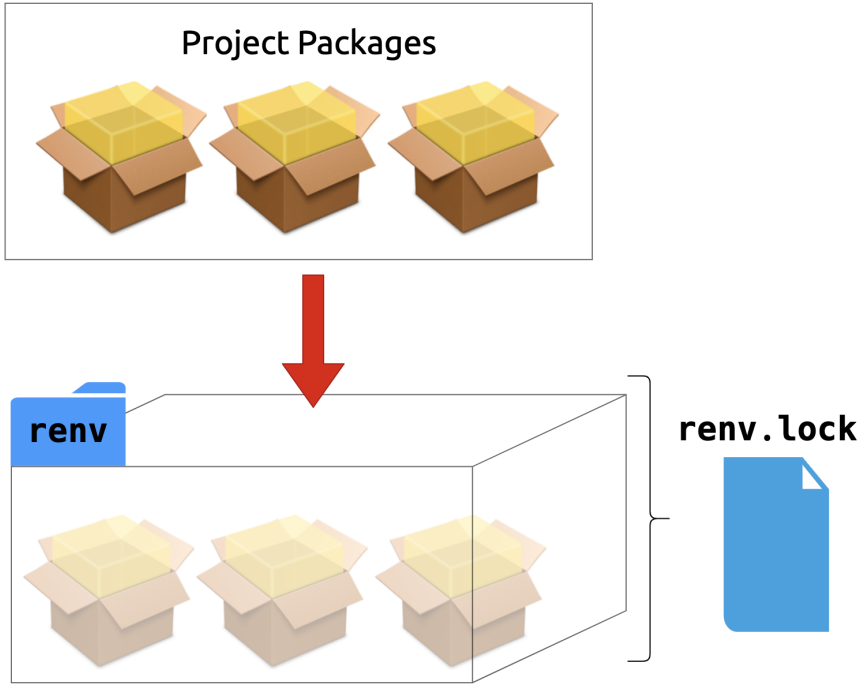
renv::init()

Creates local renv environment; caches packages.

renv::init()

Creates local renv environment; caches packages.

Documents packages in renv.lock



renv::dependencies()

```
library(ggplot2)

targets::tar_target()

require(dplyr)

requireNamespace("devtools")
```

```
renv::dependencies()
```

```
library(ggplot2)

targets::tar_target()

require(dplyr)

requireNamespace("devtools")
```

Your Turn 1

Create a new project

Create a new file called plot.R **In that file, load** ggplot2 **and create a plot with** quickplot(mpg$displ)**.**

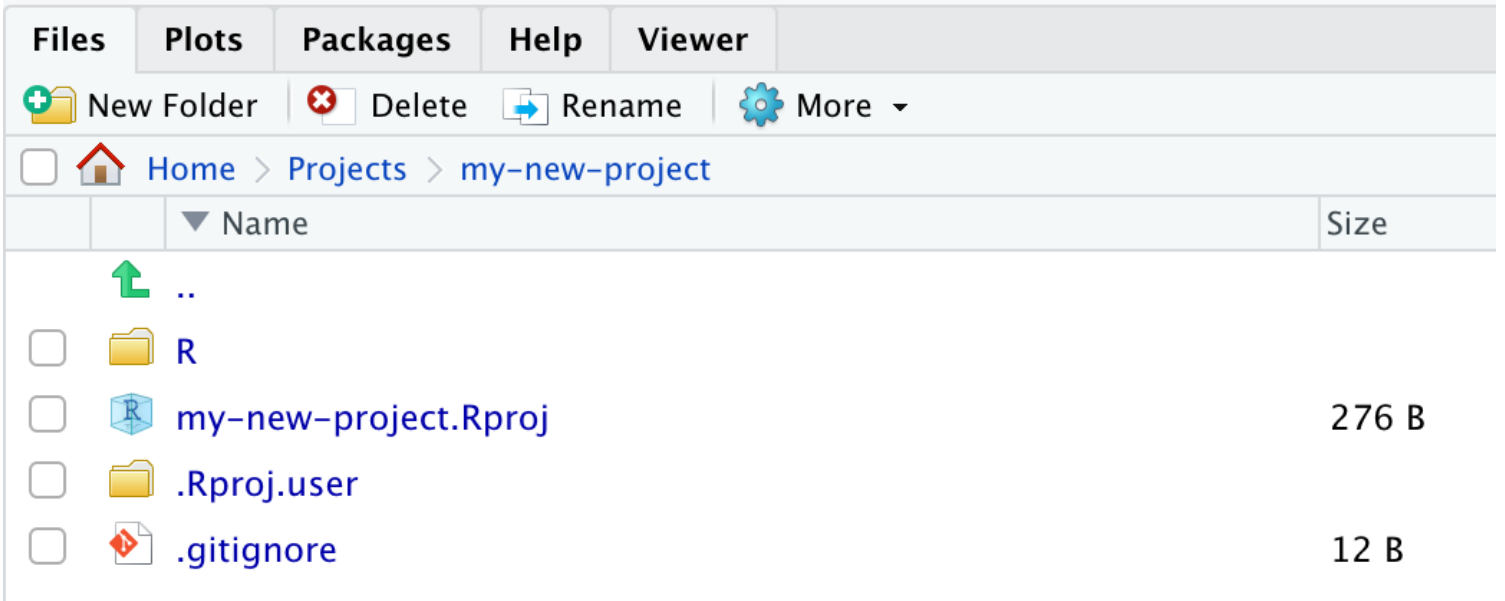
What packages does your project depend on? Make a prediction, then run renv::dependencies() **in the console (not in** plot.R**!) to see if you were right.**

In the console, initiate a renv **environment with** renv::init()

Open the file called renv.lock**. What is this information?**

Your Turn 1

In RStudio: File > New Project > 'my-new-project'

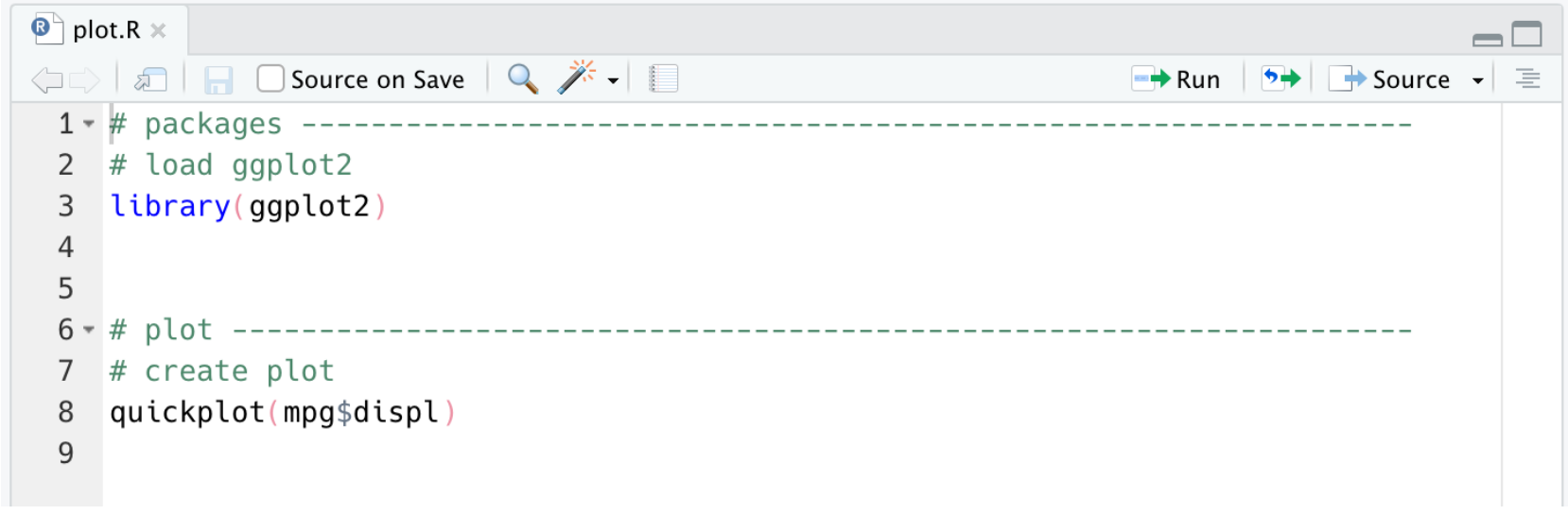


Your Turn 1

Or with usethis

```
usethis::create_project("~/Projects/my-new-project")

✓ Creating '~/Projects/my-new-project/'
✓ Setting active project to '~/Projects/my-new-project'
✓ Creating 'R/'
✓ Writing 'my-new-project.Rproj'
✓ Adding '.Rproj.user' to '.gitignore'
✓ Opening '~/Projects/my-new-project/' in new RStudio session
✓ Setting active project to 'my-last-project'
```

Your Turn 1


plot.R
Source on Save
Run
Source
1 # packages ----------
2 # load ggplot2
3 library(ggplot2)
4
5
6 # plot ----------
7 # create plot
8 quickplot(mpg$displ)
9

Your Turn 1

```
> renv::dependencies()
Finding R package dependencies ... Done!
                                    Source Package Require Version   Dev
1 /Users/Projects/my-new-project/plot.R ggplot2                 FALSE
```

Your Turn 1

```
> renv::init()
* Initializing project ...
* Discovering package dependencies ... Done!
* Copying packages into the cache ... Done!
The following package(s) will be updated in the lockfile:
```

Your Turn 1

Packages in project library

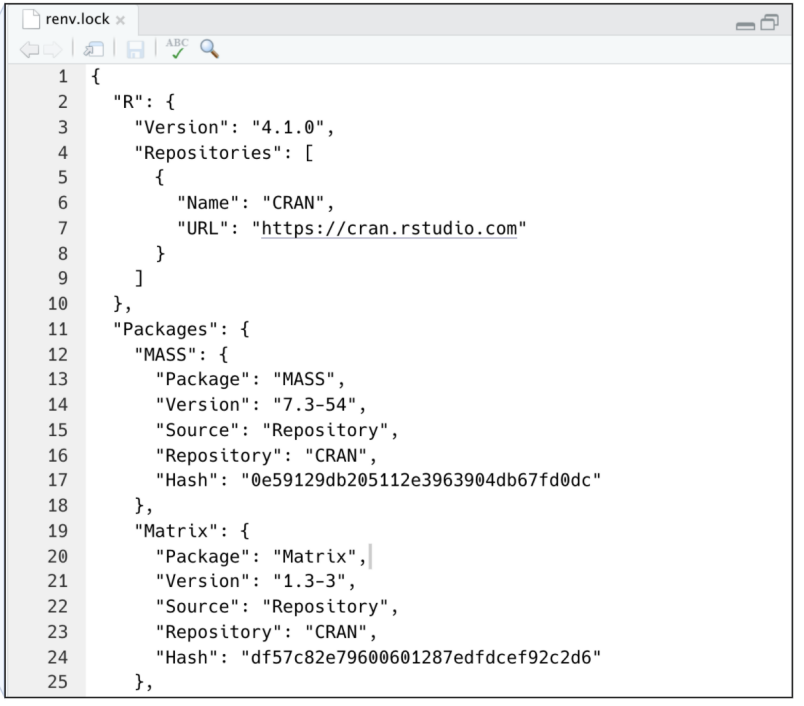
```
# CRAN ================================
- MASS           [* -> 7.3-54]
- Matrix         [* -> 1.3-3]
- R6             [* -> 2.5.0]
- RColorBrewer   [* -> 1.1-2]
- cli            [* -> 2.5.0]
- colorspace     [* -> 2.0-1]
- crayon         [* -> 1.4.1]
            ...<omitted>...
```

Your Turn 1

```
* Lockfile written to '~/Projects/my-new-project/renv.lock'.

Restarting R session...

* Project '~/Projects/my-new-project' loaded. [renv 0.13.2]
```

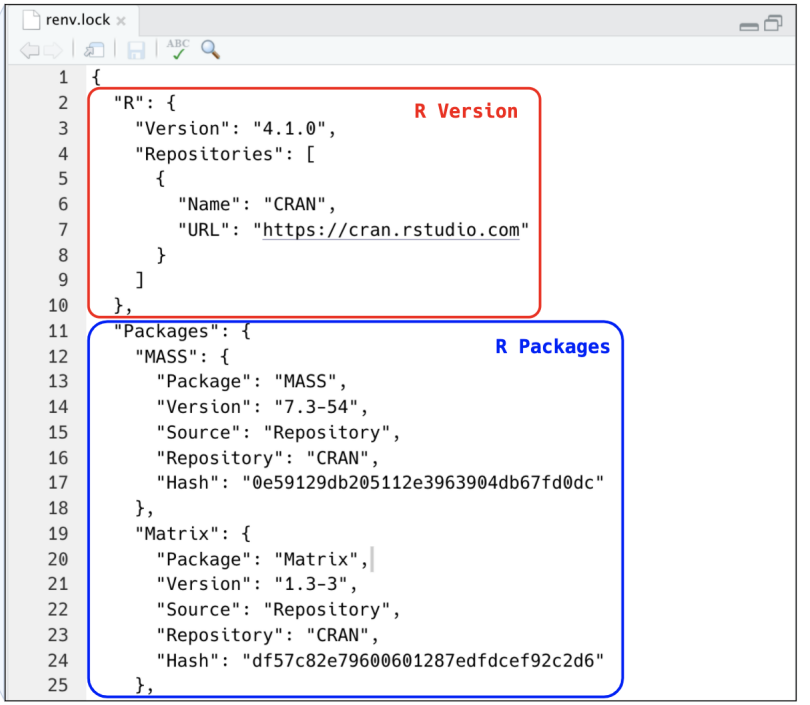
Your Turn 1

```
* Lockfile written to '~/Projects/my-new-project/renv.lock'.

Restarting R session...

* Project '~/Projects/my-new-project' loaded. [renv 0.13.2]
```

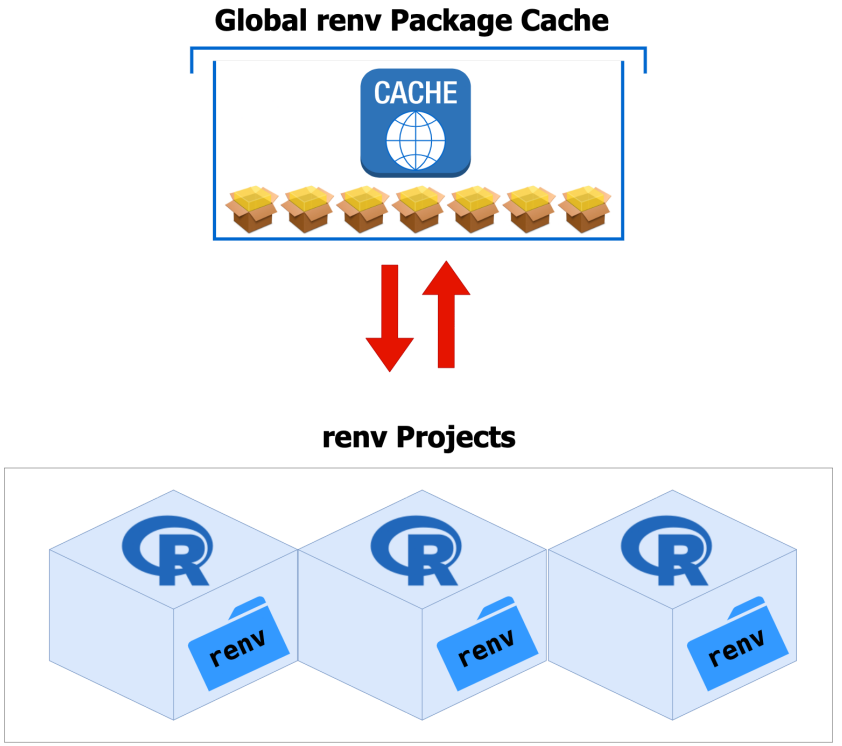


Your Turn 1

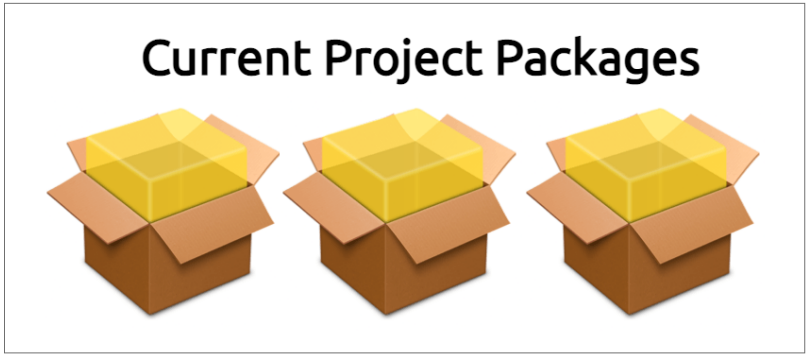
```
* Lockfile written to '~/Projects/my-new-project/renv.lock'.

Restarting R session...

* Project '~/Projects/my-new-project' loaded. [renv 0.13.2]
```

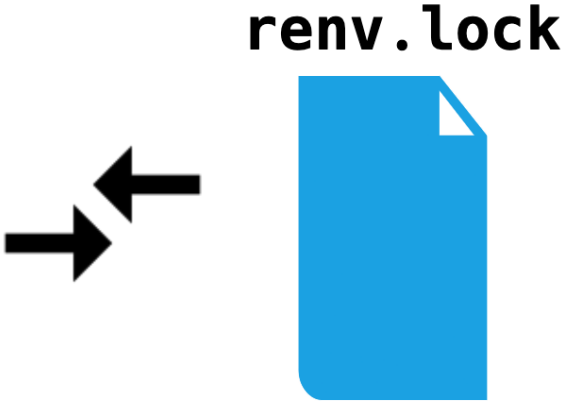


How renv stores packages



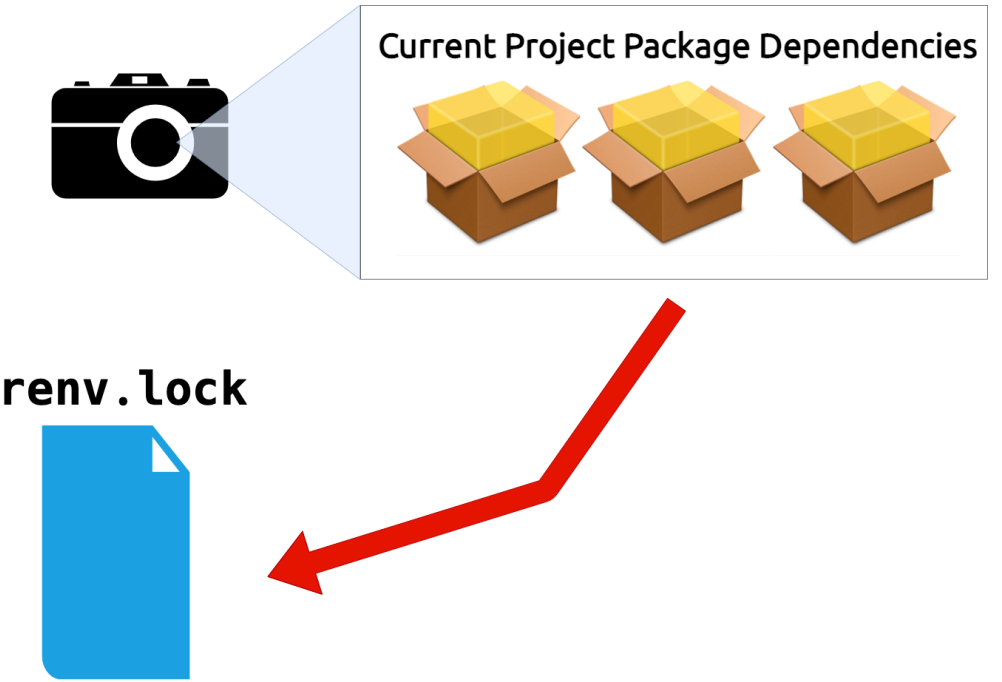
renv::status()





Checks for differences between the renv.lock and the current project's packages

renv::snapshot()



Your Turn 2

In plot.R**, modify your code to load the** patchwork **library, then replace your previous plot with**
quickplot(mpg$displ) +
quickplot(as.factor(mpg$cyl))

Run renv::status() **in the console**

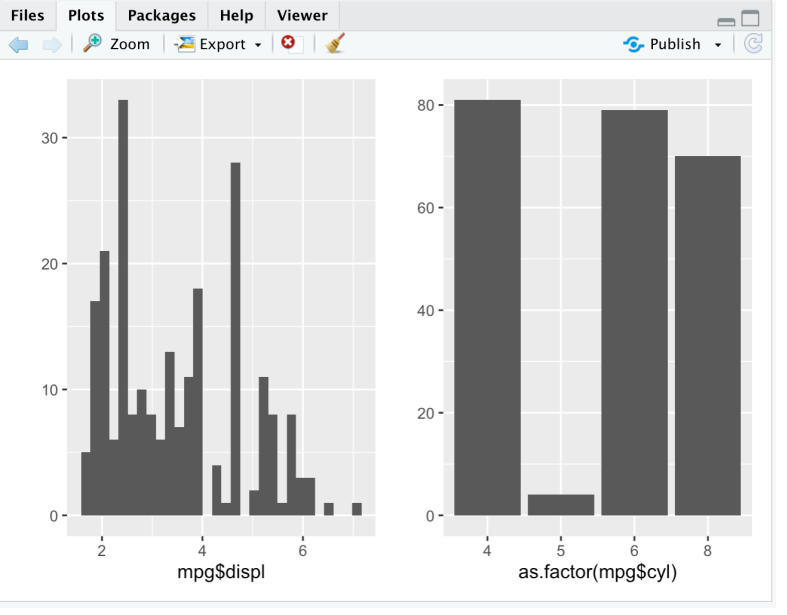
Run renv::snapshot()**. What's changed?**

Your Turn 2

plot.R

Source on Save | Run | Source

```
1  # packages ------------------------------------------------------------------
2  # install packages
3  install.packages(c("ggplot2", "patchwork"))
4  # load packages
5  library(ggplot2)
6  library(patchwork)
7
8
9  # plot ----------------------------------------------------------------------
10 # adjust plot
11 quickplot(mpg$displ) + quickplot(as.factor(mpg$cyl))
12
```



Your Turn 2

```
> renv::status()
The following package(s) are installed but not recorded in the lockfile:

  patchwork [1.1.1]

Use `renv::snapshot()` to add these packages to your lockfile.
```

Your Turn 2

```
> renv::snapshot()
The following package(s) will be updated in the lockfile:

# CRAN ================================
- patchwork   [* -> 1.1.1]

Do you want to proceed? [y/N]:
```

```
Do you want to proceed? [y/N]: y
* Lockfile written to '~/Projects/my-new-project/renv.lock'.
```

renv workflow

1. **Create a project**
2. renv::init()
3. **Write code**
4. renv::snapshot()
5. **Iterate**

Restoring project states

1. **Copy project**
2. renv::restore()
3. **That's it!**

Using renv with targets

targets encourages

tar_option_set(packages = "...")

Using renv with targets

targets encourages
tar_option_set(packages = "...")

tar_option_set() is **efficient** but **works differently** than other ways of loading packages

Using renv with targets

```
library(ggplot2)

targets::tar_target()

require(dplyr)

requireNamespace("devtools")
```

Recent updates to renv also support `tar_option_set()`!

Using renv with targets

tar_renv()

Still helpful with recent updates because of interactive and suggested packages

Your Turn 3

Create a new project

Load targets and run tar_script() **in the console. Read** _targets.R **and predict what your dependencies are.**

Run renv::dependencies()

Run tar_renv() **and look at** _targets_packages.R **What did targets do here? Why is that important? Run** renv::dependencies() **again and confirm that we can detect all of our dependencies.**

Run renv::init()

Your Turn 3

```
> usethis::create_project("~/Projects/my-newest-project")
✓ Creating '/Users/mjfrigaard/Projects/my-newest-project/'
✓ Setting active project to '/Users/mjfrigaard/Projects/my-newest-project'
✓ Creating 'R/'
✓ Writing 'my-newest-project.Rproj'
✓ Adding '.Rproj.user' to '.gitignore'
✓ Opening '/Users/mjfrigaard/Projects/my-newest-project/' in new RStudio session
✓ Setting active project to '<no active project>'
```

Your Turn 3

tar_script(): _targets.R

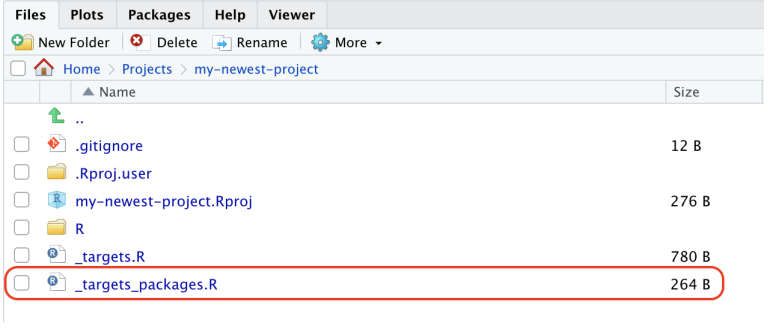
```
library(targets)
# This is an example _targets.R file. Every {targets} pipeline
# needs one. Use tar_script() to create _targets.R and
# tar_edit() to open it again for editing. Then, run tar_make()
# to run the pipeline and tar_read(summary) to view the results.
# Define custom functions and other global objects. This is where
# you write source(\"R/functions.R\") if you keep your functions
# in external scripts.
summ <- function(dataset) {
  summarize(dataset, mean_x = mean(x))
}
# Set target-specific options such as packages.
tar_option_set(packages = "dplyr")
# End this file with a list of target objects.
list(
  tar_target(data, data.frame(
    x = sample.int(100),
    y = sample.int(100))),
  # Call your custom functions as needed.
  tar_target(summary, summ(data))
)
```

Your Turn 3

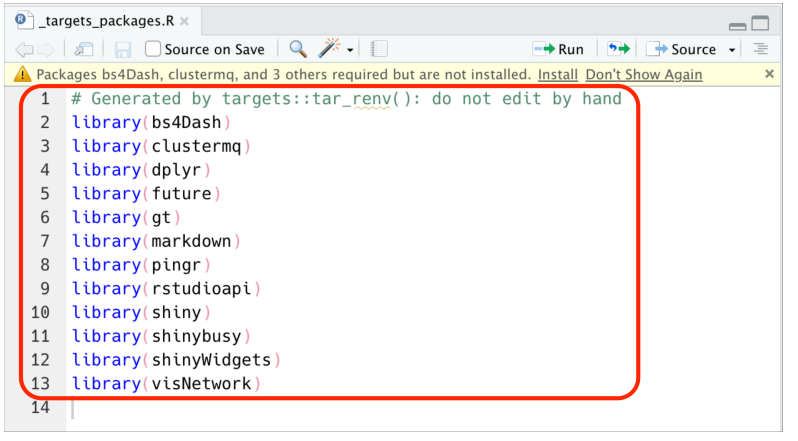
```
> renv::dependencies()
Finding R package dependencies ... Done!
                                                  Source Package Require Version   Dev
1 /Users/mjfrigaard/Projects/my-newest-project/_targets.R   dplyr                 FALSE
2 /Users/mjfrigaard/Projects/my-newest-project/_targets.R targets                 FALSE
```

Your Turn 3

tar_renv()



_targets_packages.R



Your Turn 3

```
> renv::dependencies()
Finding R package dependencies ... Done!
                                                    Source      Package Require Version   Dev
1  /Users/Projects/my-newest-project/_targets_packages.R      bs4Dash                   FALSE
2  /Users/Projects/my-newest-project/_targets_packages.R    clustermq                   FALSE
3  /Users/Projects/my-newest-project/_targets_packages.R        dplyr                   FALSE
4  /Users/Projects/my-newest-project/_targets_packages.R       future                   FALSE
5  /Users/Projects/my-newest-project/_targets_packages.R           gt                   FALSE
6  /Users/Projects/my-newest-project/_targets_packages.R     markdown                   FALSE
7  /Users/Projects/my-newest-project/_targets_packages.R        pingr                   FALSE
8  /Users/Projects/my-newest-project/_targets_packages.R   rstudioapi                   FALSE
9  /Users/Projects/my-newest-project/_targets_packages.R        shiny                   FALSE
10 /Users/Projects/my-newest-project/_targets_packages.R    shinybusy                   FALSE
11 /Users/Projects/my-newest-project/_targets_packages.R shinyWidgets                   FALSE
12 /Users/Projects/my-newest-project/_targets_packages.R   visNetwork                   FALSE
13          /Users/Projects/my-newest-project/_targets.R        dplyr                   FALSE
14          /Users/Projects/my-newest-project/_targets.R      targets                   FALSE
```

Your Turn 3

```
> renv::init()
* Initializing project ...
* Discovering package dependencies ... Done!
* Copying packages into the cache ... [73/73] Done!
* Resolving missing dependencies  ...
```

Your Turn 3

```
* Lockfile written to '~/Projects/my-newest-project/renv.lock'.

Restarting R session...

* Project '~/Projects/my-newest-project' loaded. [renv 0.13.2]
```

Resources

renv: Getting started: A brief introduction to renv and its workflows

renv: Project Environments for R (blog): An RStudio blog post introducing renv

renv: Project Environments for R (talk): A talk on renv from rstudio::conf() 2020