Simple process for sharing R code via Docker

Ronald (Ryy) Glenn Thomas

2025-03-03

Table of contents

1 Motivation

We have code we want to send to a colleague to run and/or edit. In doing this we need to avoid the problem of potentially different `R` and `R`-package versions. We also need to avoid needing to install `pandoc` and `latex` as well as supplemental files, e.g. `bibtex`, `latex` preamble, logo graphics files.

In sum, we want out colleague to simply have to download a `github` repository and pull a `docker` image from `dockerhub` and then run the command:

```
docker run  -v "$(pwd):/home/joe" -v "$(pwd)/output:/home/joe/output" rgt47/penguins1
```

inside the repo to get access to the identical R environment. And

```
vim peng1.Rmd
```

to edit

and

```
R -e "library(rmarkdown); render('peng1.Rmd')"
```

to process code through `R`, `pandoc`, and `latex`.

2 Introduction

Its often the case that two or more research collaborators want to work on the `R` codebase for the same project. To accomplish this, unfortunately, its often not as simple as one researcher sending a set of text files containing the code to the other colleagues.

For example, a number of elements of the computing environment can differ between collaborators' computers, for example, the version of `R`, the operating system (macos vs. windows vs linux), the `R` packages installed, the versions of these `R` packages, `R` startup files (e.g. `.Rprofile`, and `.Renviron`). Any of these components of a researchers computing environment can cause code working for one reseracher to break when run by a second.

3 Methods

Lets assume you have some `R` program code in a file, say `peng.R`, that you're written to analyze some "Palmer Penguin" data. You want to share the code with a colleague, we'll call him Joe. How to proceed?

The simplest option is simply to send Joe the `R` file containing the code via the most convenient method (e.g. email)

The next step is for for Joe to (attempt to) run the code with the idea of expanding the analysis. Typically he would do this either using an IDE, such as `Rstudio`, or run the code from the command line in the terminal with the command:

```
> R -e "source('peng1.Rmd')"
```

Sometimes this approach works. When it does, Joe can add comments or expand the code and relay it back to you, "rince and repeat" and all is well. Frequently, however, this simple approach will fail for any of several reasons. Even when it runs, unless care is taken, its not guaranteed that Joe will get the same results you do.

Ideally to facilitate reproducibility your colleague Joe will have a computing environment as similar to yours as possible. This can be difficult to achieve, especially given the dynamic nature of open source software. For example, Joe may have an older version of `R` installed on his workstation, or his `R` environment may be missing a necessary package to run the code or it may be the wrong version. Additiional potential problems include: the program may need to source an additional file or load data that's missing.

All of these problems go away if instead of sending the program as a standalone text file you send it as a docker image. In this post we'll walk through the process of dockerizing the R code.

4 Methods

Assume we have a simple `R` file that we want to share with Joe
such as the following:

```
---
title: "\\includegraphics[width=2cm, right]{sudoku_apple.pdf}\\newline
  New Penguin plot"
author: "R.G. Thomas"
date: "`r format(Sys.time(), '%B %d, %Y')`"
fontsize: 11pt
geometry: "left=3cm,right=5cm,top=2cm,bottom=2cm"
output:
  pdf_document:
    keep_tex: true
    includes:
header-includes:
   - \usepackage{lipsum}
   - \usepackage[export]{adjustbox}
   - \usepackage{currfile}
   - \usepackage{fancyhdr}
   - \pagestyle{fancy}
   - \setlength{\headheight}{14pt}
   - \fancyfoot[L]{\currfilename} %put date in header
   - \fancyfoot[R]{\includegraphics[width=.8cm]{sudoku_apple.pdf}}
   - \fancyhead[L]{\today} %put current file in footer
   - \fancyhead[R]{Penguins RGT Lab report}
latex_engine: xelatex
---

```{r  echo=F}
# clear env: objects and packages
library(pacman)
p_load(palmerpenguins, rmarkdown, knitr)

opts_chunk$set(
  warning = FALSE, message = FALSE, echo = FALSE, fig.width = 5.2,
  fig.height = 3, results = "asis", dev = "pdf"
)
```
```

```
# Introduction

\lipsum[1-5]

Draft report examining Penguin characteristics. Data from Alison Horst article
in the `R journal`
@m.horstPalmerArchipelagoPenguins2022

```{r }
attach(penguins)
plot(flipper_length_mm, bill_length_mm, col = sex)
legend("topleft", legend = levels(sex), pch = 16, col = unique(sex))
```

\lipsum[1-5]
\bibliography{penguins.bib}
\bibliographystyle{plain}
\nocite{*}
```

5 Share program code with Joe.

Joe downloads the attachment. Opens a working directory and attempts to run the Rmd file

with the command

```
> R -e "source('peng1.Rmd')"
```

Joe has a linux mint desktop

```
> mkdir peng1
> cd peng1
> R -e "render('peng1.Rmd')"
```

Linux can't find R

Joe can fix this by installing `R`

```
> sudo apt install r-base-core
```

So install R…. to fix

> sudo apt install r-base-core Try again…
>
> R -e "render('peng1.Rmd')" Result: R loads but gives error: could not find function "render"

Search google "R render"

Result: https://pkgs.rstudio.com/rmarkdown/reference/render.html

Looks like render is a function inside rmarkdown package. so install package

```
R> install.packages("rmarkdown")
```

Try again.

```
> R -e "library(rmarkdown); render('peng1.Rmd')"
```

Result: error. pandoc version 1.12.3 or higher required.

Now try to install pandoc Try again. > R -e "library(rmarkdown); render('peng1.Rmd')" Result: error. there is no called pacman

so install pacman R> install.packages("pacman") Try again…

> R -e "library(rmarkdown); render('peng1.Rmd')" result: failed pandoc could not find /Users/zenn/shr/preamble.tex

This makes sense we forgot to send the preamble.tex file to researcher 2. Lets send it now. Also we need to adjust its location from a macos style file name to a linux based one.

To edit peng1.Rmd we need vim > sudo apt install vim

Try again Result: pandoc error "pdflatex not found" Lets install tinytex. First in R and then in mint R> install.packages("tinytex") R> tinytex::install_tinytex() # to uninstall TinyTeX, run # tinytex::uninstall_tinytex()

Try again:

> R -e "library(rmarkdown); render('peng1.Rmd')"
> result: pandoc error: file sudoku_apple.pdf not found.

This makes sense we forgot to send the logo file. Lets send the file and try again.

> R -e "library(rmarkdown); render('peng1.Rmd')"
> Result: error no bibliography file found.

This makes sense we forgot to send the bib file. Lets send the file and try again. Also need to edit the location of bib file.

Try again:

> R -e "library(rmarkdown); render('peng1.Rmd')"

results: minor error. Some packages weren't loading via pacman.

Try removing janitor, kableExtra, tidyverse, readxl and add ggplot2.

Try again:

> R -e "library(rmarkdown); render('peng1.Rmd')"

Success!

Lesson learned: The program peng1.Rmd needs access to three external files: 1. preamble.tex 2. penguins.bib 3. sudoku_apple.pdf

6 Docker approach

Alternatively, consider the "Docker" approach.

Before sending peng.Rmd to Joe we'll dockerize it.

- Prepare a work directory: peng1. We want to prepare a repo and a docker image that has a standardized computing environment set up including R, the R packages and all the preliminaries taken care of, so that all Joe has to do is clone the repo and run the image from dockerhub.

Steps for Joe:

- install docker on his machine.
- download docker image from dockerhub
- run image

Here is the docker file

```
FROM rocker/r-devel
RUN apt-get update -qq && apt-get -y --no-install-recommends install \
    libxml2 \
    libxml2-dev \
    libcairo2-dev \
    libsqlite3-dev \
    libpq-dev \
    libssh2-1-dev \
    unixodbc-dev \
    r-cran-v8 \
    libv8-dev \
    net-tools \
    libprotobuf-dev \
    protobuf-compiler \
    libjq-dev \
    libudunits2-0 \
    libudunits2-dev \
    libgdal-dev \
    libssl-dev
## update system libraries
RUN apt-get update && \
    apt-get upgrade -y && \
    apt-get clean
RUN apt-get install pandoc -y
RUN groupadd --system joe
RUN useradd --system --gid joe -m joe
RUN chown joe:joe -R /home/joe
```

```
USER joe
WORKDIR /home/joe
COPY renv.lock renv.lock
RUN R -e "install.packages('renv', repos = c(CRAN = 'https://cloud.r-project.org'))"
COPY renv/settings.json renv/settings.json
RUN R -e "renv::restore()"
CMD ["/bin/bash"]
```

run docker

```
docker build -t rgt47/penguins1 --platform=linux/amd64 .
```

relay image to Joe via dockerhub.

```
docker push rgt47/penguins1
```

or

```
docker save rgt47/penguins1 | gzip > penguins1_transfer.tgz
docker load -i penguins1_transfer.tgz
```

```
> docker pull rgt47/penguins1

docker run -it --rm \
-v "$(pwd):/home/joe" -v "$(pwd)/output:/home/joe/output" rgt47/penguins1


> cd output
> library(rmarkdown); render('peng1.Rmd')
```

Important to include the association between the /home/joe/output directory in the container with the output directory on the local workstation. Thats where the results of the analysis will be saved.

```
> R -e "library(rmarkdown); render('peng1.Rmd')"
```

and if he wants to edit peng.Rmd

```
> vim peng1.Rmd
```

7 REFERENCES

Running your R script in Docker

```
docker run --rm -p 8787:8787 -e PASSWORD=z rocker/rstudio

j share
cd peng1

docker build -t rgt47/penguin1 .
docker run --rm -it rgt47/penguin1

droot="${PWD}"

docker run -it --rm -v $droot:/home/joe/output rgt47/penguin1

Problem
  docker run -it --rm -v $droot:/home/joe rgt47/penguins1

Unable to find image 'rgt47/penguins1:latest' locally
docker: Error response from daemon: manifest for rgt47/penguins1:latest not found: manifest unk
See 'docker run --help'.
share_R_code_via_docker/peng1   echo $droot
/Users/zenn/prj/qblog/posts/share_R_code_via_docker/peng1
share_R_code_via_docker/peng1   mkdir output
share_R_code_via_docker/peng1   f

Need a base R that includes dev tools

Try rocker/r-devel

docker build -t rgt47/penguin2 . --platform linux/amd64


docker run --rm \
  -e DISABLE_AUTH=true -e ROOT=true \
  -v "$(pwd):/home/rstudio/project:rw" \
```

```
  -p 8787:8787 \
  --name rstudio rocker/verse:3.6.3

docker run -it --rm -v "$(pwd):/home/joe" rgt47/penguins1
```

Dockerfile with R code

```
FROM rocker/r-devel
RUN apt-get update && \
    apt-get upgrade -y && \
    apt-get clean
RUN apt-get install terminator tree eza ssh zsh git fzf ripgrep vim curl \
autojump  zsh-autosuggestions sudo pandoc -y
RUN Rscript -e 'install.packages("renv")'
COPY renv.lock renv.lock
RUN  Rscript -e 'renv::restore()'
RUN groupadd --system joe
RUN useradd --system --gid joe -m joe
RUN usermod -aG sudo joe
RUN echo joe:z | chpasswd
RUN chown joe:joe -R /home/joe
RUN chown joe:joe -R /usr/local/lib/R/site-library
RUN chsh -s $(which zsh) joe
WORKDIR /home/joe/
COPY .vimrc .vimrc
COPY .zshrc .zshrc
RUN chown joe:joe -R /home/joe
RUN mkdir -p  /home/joe/output
USER joe
RUN curl -fLo ~/.vim/autoload/plug.vim --create-dirs \
    https://raw.githubusercontent.com/junegunn/vim-plug/master/plug.vim
CMD ["/bin/zsh"]
```

```
docker build -t rgt47/penguins1 . --platform linux/amd64
docker pull rgt47/penguins
docker run -it --rm -v "$(pwd):/home/joe/peng2" -v "$(pwd)/out:/home/joe/output" rgt47/penguins
```