Simple process for sharing R code via Docker

Ronald (Ryy) Glenn Thomas

2024-02-05

Table of contents

1 Introduction

Lets assume you have an rmarkdown `Rmd` file, say `peng.Rmd`, that you're written to analyze some data. You now 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 "rmd" file containing the code via the most convenient method (e.g. email/text/slack/discord/github/USB drive etc.)

The next step will be for the Joe to (attempt to) load and run the code. Typically he would do this with either using `Rstudio.app` to open the file and `knit` it, render it from the command line with the command:

```
> R -e "source('fig.R)"
```

Sometimes this approach works, and all is well. Joe can add comments or expand the code and reply to you. Frequently, however, this naive process will fail for any number of reasons. Ideally to facilitate reproducibility Joe will have as similar a computing environment as you, the original developer. This can be difficult to achieve, especially given the dynamic nature of open source software. For example Joe may have an outdated version of `R` installed on his workstation, or his `R` environment may be missing a necessary package. Additiional potential problems include: the required package may be present but its the wrong version, the program may need to source an additional file thats missing, or the program load some data that it can't find on Joe's machine.

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.

Assume a simple `Rmd` file like this:

```
zz.sum.max <- function(x, extra = F) {
  if (!is.numeric(x)) {
    cat("Error: Numeric arrays required.")
    return(1)
  }
  x <- x[!is.na(x)]
  N <- length(x)
  options(digits = 4)
  if (!N) {
    return(c(0))
  }
  Mean <- mean(x)
  V <- var(x)
  SD <- V^.5
  SE <- V^.5 / (N^.5)
  Min <- min(x)
  Median <- quantile(x, probs = c(.5), names = F)
  Max <- max(x)
```

```
  errmin2 <- Mean - 1.96 * SE
  errmax2 <- Mean + 1.96 * SE
  errmin1 <- Mean - SE
  errmax1 <- Mean + SE
  if (extra) {
    return(c(N = round(N), Mean = Mean, SE = SE, ERRMN1 = errmin1, ERRMX1 = errmax1, ERRMN2 = e
  } else {
    return(c(N = round(N), Mean = Mean, Median = Median, SE = SE, SD = SD, min = Min, max = Max
  }
}

dat1 <- read.csv("orth.csv")
out <- zz.sum.max(dat1$age)
print(out)
```

2 Share program code with Joe. Two approaches, Naive and Docker based.

Whats the best way to accomplish this?

We start by simply emailing the file to him (rgthomas4747@gmail.com) and asking him to collaborate.

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

with the command

```
> R -e "source('fig.R')"
```

Joe has a linux mint desktop

```
> mkdir fig_collaboration
> cd fig_collaboration
> R -e "source('fig.R')"
```

Linux can'f find R

Joe can fix this by installing `R`

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

Next `R` can not find the function `render`.

Joe determines that `render` is a function in the package `rmarkdown`

He endeavors to installs `rmarkdown` with

```
R -e "install.packages('rmarkdown')"
```

This fails due to inadequate permission on the directory `/usr/lib/R/library`

```
sudo apt install libssl-dev libcurl4-openssl-dev unixodbc-dev libxml2-dev\
libmariadb-dev libfontconfig1-dev libharfbuzz-dev libfribidi-dev\
libfreetype6-dev libpng-dev libtiff5-dev libjpeg-dev
```

Also latex is not available

One more try… and the latex engine notes the absence of the file

```
~/shr/preamble.tex
```

So, I need to relay the missing .tex file.

Also the .png (sudoku.png) logo file.

Finally! success.

3 Docker approach

Alternatively, consider the "Docker" approach.

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

- Prepare a work directory: penguins. We want to send Joe a container that has R and all the preliminaries taken care of so that all he has to do is

Here is the docker file

```
FROM rhub/r-minimal
RUN addgroup --system joe && adduser --system --ingroup joe joe
RUN chown joe:joe -R /home/joe
USER joe
WORKDIR /home/joe
RUN mkdir -p /home/joe/shr
RUN mkdir -p /home/joe/output
COPY fig.R /home/joe/shr
COPY orth.csv /home/joe/shr
CMD ["/bin/bash"]
```

run docker

```
docker build -t rgt47/penguin_review --platform=linux/amd64 .
docker push rgt47/peng_review
```

relay image to Joe

```
docker push rgt47/peng_review
```

or

```
docker save rgt47/peng_review | gzip > peng_review_trans.tgz
docker load -i peng_review_trans.tgz
```

```
> docker pull rgt47/penguin_review

> droot="$PWD"/output docker run -it --rm --platform linux/x86_64 \
-v $droot:/home/joe/output peng_review
> cd output
> library(rmarkdown); render('../shr/peng.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('peng.Rmd')"
```

and if he wants to edit peng.Rmd

```
> vim peng.Rmd
```

```
\usepackage[export]{adjustbox}
\usepackage{fancyhdr}
\usepackage{titling}

\pagestyle{fancy}

\pretitle{
\begin{flushright}
\includegraphics[width=3cm,valign=c]{sudoku.png}\\
\end{flushright}
\begin{flushleft} \LARGE }
\posttitle{\par\end{flushleft}\vskip 0.5em}
\predate{\begin{flushleft}\large}
    \postdate{\par\end{flushleft}}
    \preauthor{\begin{flushleft}\large}
    \postauthor{\par\end{flushleft}}
\fancyfoot[L]{\currfilename} %put date in header
\fancyfoot[R]{\includegraphics[width=.8cm]{sudoku.png}}
\fancyhead[L]{\today} %put current file in footer
```

4 REFERENCES

Running your R script in Docker