

# Simple process for sharing R code via Docker

Ronald (Ryy) Glenn Thomas

2024-05-30

## Table of contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Methods</b>	<b>2</b>
<b>3</b>	<b>Share program code with Joe.</b>	<b>3</b>
<b>4</b>	<b>Docker approach</b>	<b>4</b>
<b>5</b>	<b>REFERENCES</b>	<b>6</b>

## 1 Introduction

Its often the case that two research collaborators want to work on the same codebase. Unfortunately, its not as simple as one researcher sending the text file containing the code to their colleague.

Things that can differ between collaborators are: the version of R, the operating system, packages installed, and environment variables.

Lets assume you have an R code program, say `peng.R`, that you're written to analyze some 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 will be for Joe to ( attempt to ) run the code. Typically he could do this either using an IDE, e.g. `Rstudio.app`, to open the file and run it, or run it from the command line with the command:

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

Sometimes this approach works. Joe can add comments or expand the code and relay back to you, and all is well. Frequently, however, this simple approach will fail for any of several reasons. Even when it runs its not guaranteed that Joe will get the same results you did. Issues such as package version, R version, `.Rprofile` settings, even the compiler settings when R was compiled, can all impact results.

Ideally to facilitate reproducibility your colleague Joe will have a computing environment similar to yours. 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 or it may be the wrong version. Additional potential problems include: the program may need to source an additional file that's missing, or the program may load a dataset 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.

## 2 Methods

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

```
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 = c
  } 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)

```

### 3 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('peng.R')"
```

Joe has a linux mint desktop

```
> mkdir peng_collaboration
> cd peng_collaboration
> R -e "source('peng.R')"
```

Linux can't find R

Joe can fix this by installing R

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

## 4 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
ENV MRAN_BUILD_DATE=2024-02-01 # Install Basic Utility R Packages
RUN installr -r https://cran.microsoft.com/snapshot/${MRAN_BUILD_DATE} \
    --error -d -t "zlib-dev" shiny
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. That's 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

```

## 5 REFERENCES

[Running your R script in Docker](#)