More on packages

Naomi Tague

January, 2023

Creating Packages

- creating and link with github
- edit your DESCRIPTION file to add title, descripition of your package
- populate with R functions
- run *document()* to create help files
- repeat as needed adding functions as you go *build* and *load* as you go
- add package dependencies with * **usethis::use_package("name")** * which updates your DESCRIPITION file

Next

- add data
- add testing
- share your package

Data stored with packages

Reminder: To access a dataset that has been stored with a package

data(name)

There are numerous data sets stored with the base R package try

`data(CO2)'

`help(CO2)'

This is a data set on carbon dioxide uptake in plants - note that it always has help documentation that is similar to functions

You can then refer to that dataset as an object in your workspace

Storing your own data

- have the dataset (e.g *fish*) in your workspace (e.g you've already read it in)
- save the data to the data subdirectory of your package using:
  - *use_data(name, pkg="packagename")*
  - *eg. use_data(fish, pkg=ESM262Examples)*
- dataset will be stored as *data/name.rda* in your package

Try this

Documenting Data Sets for Packages

You can also create help pages for your dataset, the format is similar to what you use to document functions, what differs is where you put the documentation code

Here's an example

Data Documentation example

Example of documentation for data

```
#'  Climate Data from Santa Barbara
#'
#'  Data from SB-LTER met station, daily meterologic data
#'  including temperature, precipitation, solar radidatiaon and
#'  variables derived from RHESSys model estimates
#'
#' @format A data frame with 53940 rows and 10 variables:
#' \itemize{
#'   \item data Year-Month-Day
#'   \item day
#'   \item month
#'   \item year
#'   \item wy wateryear
#'   \item tavg average daily temperature (C)
#'   \item vpd vapor pressure deficit (MPa)
#'   \item rnet (kJ/day)
#'   \item precip (mm/day)
#' }
#' @source \url{http://sbc.lternet.edu/data/}
#'
"metdata"
```

Steps for documenting data sets

- store documentation as a file (dataname.R) (e.g **R/clim.R**) save that file in the *R* subdirectory of your package
- this file will contain ONLY the documentation for the data set
- when you rebuild or run document - documentation will be created with roxygen2
- format is similar to documentation for function but you will use some different identifiers for documentation including:
  - *title*
  - *descripition*
  - *@format*
  - *@source*
  - *\itemize - itemize allows you to describe the data (see example above)*
  - *end the file with the name of your dataset (in data directory) in quotes*
  - *set your working directory to your project directory*
  - *once you've generated the documentation - run document(dataname) and try help(dataname) to test*

Generate some data and store it with your package

Testing

For all of your functions; you should write at least one test

Put that test into the *tests* directory *test_functionname.R*

Run tests_dir() frequently

Giving the package to others

To give the package to others - (who will simply be using it)

- build a source package (see build directory)
- this will create an *.gz file that you can distribute
- the user will then **install** this package (see installation menu, go to install from *.gz files) in order to have access to functions and data in your package

Alternatively if your users are used to *github* you can use

devtools::install_github("yourname/packagename")

This can be a good way to share the package with yourself :)

Final Assignment

Create a package that includes at least 1 function and 1 data set, make sure you create help pages for both. You can use functions that you created for the last assignment or generate new ones.

Work with the same partner.

Turn in link to package repository.

Grading

Creation of a package

- has at least one function (5pts)
- has a least one data set (5pts)
- package works (e.g you can run library(packagename)) (5pts)

Documentation

- help pages for function
  - *clearly describes inputs/output with units (5pts)*
  - *explains what the function does (5pts)*
- help pages for data set
  - *clearly describes the dataset (5pts)*
  - *clearly describes variables (5pts)*
- documentation in the functions themselves (5pts)

Total 40 pts

Vignettes

Vignettes are additional documentation on how to use your packages (and/or functions within the package) that provide example uses * usually written in *Rmarkdown*

You can see available vignettes for all currently loaded packages with

vignette()

or for a particular package

vignette(package="dplyr")

To actually see the vignette (comes up in help window) **vignette("two-table")**

Basic outline for making a vignette for your package

- write with *R markdown*
- store in a **vignettes** subdirectory of the package structure
- run **build_vignettes()**

Running **build_vignettes()** will create some additional subdirectories **doc** and **Meta** after build you need to reinstall with build vignette flag on

install(buld_vignettes=TRUE) library(package_name)

and re load your library!

A few technical details first

- add the following to the top of the my-vignette.Rmd (and edit **my_vignette** with your names)

```
---
title: "my-vignette"
output: rmarkdown::html_vignette
vignette: >
  %\VignetteIndexEntry{my-vignette}
  %\VignetteEngine{knitr::rmarkdown}
  %\VignetteEncoding{UTF-8}
---
```

- note that you can have more than one vignette in a package
- you may need to add the following lines to **DESCRIPTION**

```
Suggests: knitr, rmarkdown
VignetteBuilder: knitr
```

See **esm237examplesS18/vignettes/using_climate_summaries.Rmd**

Try to make a vignette for your function