Project Management

Matthew Leonawicz

January 9, 2015

1 Introduction

This is a project management project. The aim of this project is the development of convenient **R**-related project management tools.

1.1 Motivation

I am working on these tools to enhance my own workflow across multiple **R** projects.

1.2 Details

R code for the project will be compiled into an **R** package, `rpm` for easy use. This is a personal package and will not be available anywhere but my Github repository, but you are welcome to explore the package and functions. It is unlikely that you would manage your **R** projects in the same manner that I do, but if you do, or just want some ideas, feel free to explore.

1.2.1 Capabilities

The `rpm` package offers a collection of functions ranging from creating new projects to compiling project websites. With `rmp` you can:

- Create a new project - Bulk generate Rmd files associated with project **R** scripts - Update Rmd yaml front-matter contents - Append Rmd files with new code chunk identifiers after code is added to project **R** scripts - Bidirectional conversion between Rmd and Rnw files - Bulk file organization after generating a large number of final and intermediary project documentation files of various formats - Functions for generating a project-specific website, great for hosting on the `gh-pages` branch of your Github project repo - Functions for generating a Github user account website - Multi-project hierarchy diagram - Project-specific code flow diagram

1.2.2 Limitations

The project management code is not yet in package form. Additional features are yet to be incorporated. Below are specific limitations related to the above highlighted capabilities.

Files While `rpm` assists with project documentation, this mainly takes the form of file generation and appending. Documentation is unique to every project of course. Every script is different. The most that is possible with Rmd generation and updating is to auto-fill commonly used code chunk names and metadata. Each document must be written individually by the author, but when a project has many **R** scripts requiring documentation, it is nice to not have to create all the corresponding Rmd files by hand and copy and paste generic contents.

Conversion Conversion between Rmd and Rnw files also cannot be perfectly general. I have written this code mainly to account for the limited types of content and formatting which I tend to include in my own Rmd and Rnw files. The documentation on the conversion code (which you can find in html or pdf!) lists some [rules and assumptions](func$_c onvert.html"rulesandassumptions")Imakeaboutmyowndocumentation, whichdefinesthecontextinu

R, Rstudio, knitr, rmarkdown, MikTeX, Pandoc, Windows, Linux,... OH MY! And I'm not particularly interested in thinking about Mac. Anyway, this is my own limitation, an area where my current knowledge

of cross-system use and third party software integration is not strong. When using `knit` and/or `rmakrdown`, I tend to have more success avoiding crashes while knitting documents if I do all the file generation in the same directory as the source files. This is why `rpm` has code for moving output files after knitting them. I don't know why this problem exists, but for context, it is convenient for me to do the knitting with **R** on my Windows pc. I have `MikTeX` and `Pandoc` for Windows installed. However, I never bothered fussing with environment variables or whatever I might have to do in order to get pdf generation to work on Windows using the standard **R** GUI. Instead, I find it easier to use RStudio, in which case `Pandoc` is accessible and everything works fine.

So I can do all of this with RStudio, but not all of it with standard **R** — which is what I otherwise use exclusively, either on Windows or Linux. Suffice it to say that, for now at least, I must use RStudio specificallly for pdf generation. I cannot use standalone **R** on Windows. I know this is not a real prohibition, but due to my lack of having looked into it and figured it out yet. If I use Linux, pdf files sometimes come out looking a little different. This is all to say that I am not currently in a position with this project to produce something that will work completely and identically across multiple systems. In an ideal scenario, my full application of `knitr` and `rmarkdown` would work similarly in standard **R**, whether in interactive or non-interactive mode, and whether on Linux or Windows. I wouldn't worry about whether it worked seamlessly in RStudio as well. But for now I have the opposite situation.

Websites Regarding the website-building functions, the limitations are that certain libraries are expected to be in place, only particular CSS themes and code highlighting are permitted, and the files, projects, and local Git repositories these functions parse the contents of are all required to follow my specific styles of organization.