teea_session_5

N.Paterno

7/30/2021

Working Directory

For the first bit of today we worked on checking/setting the working directory. This is where R files will be saved by default and where R will look for files we try to read in.

To check the current working directory:

```
getwd()
```

To set a new working directory (and then check that the switch worked)

```
setwd("D:/r_for_school/teea_R")
getwd()
```

Best practice is to use a project based workflow (discussed below) instead of using `setwd`. If you do use that command, it should only be run in the console and not in a script or markdown file.

Project Based Workflow

Your workflow is *how* you code/create. Your product is *what* you create. For a painter their environment (music, light, etc) is the workflow and the finished painting is the product. For a coder, you want to keep your workflow out of your product. Given the same paints, the painter should be able to create the same painting in a different room. Given the same script file, a coder should be able to recreate the same document on a different computer.

Your working directory is part of your workflow as a coder; this is why we should only run that in the console if at all. A better choice is to use a project (like the one we created called "Learning R"). No matter where that folder is located, the code inside of it will run on any computer that has that folder on it. So if I email you a project folder with code that I've written, you can download it onto your computer and run the code without issue.

When you make a project in RStudio, it creates a `.rproj` file in the folder. By clicking on that file, you will launch RStudio with the working directory automatically set to that folder (which is what we want to happen).

After this, we spent some time optimizing your file structure.

RMarkdown

We briefly went over RMarkdown the last bit of the session. RMarkdown files combine code and plain text into reports. We can choose if code appears in the resulting document - `echo = TRUE` - or if we want the code to appear but not the results - `echo = FALSE`- by setting options in our `setup` code chunk.

To create a new code chunk, you can press control-alt-i. For options that we want to apply to all code chunks, we would want this as the first code chunck and call it "setup".

```
knitr::opts_chunk$set(echo = TRUE)
```

RMarkdown documents compile or `knit` into html by default. You can change this to Word (not recommended) or pdf. To `knit` to pdf, you'll need to install the `tinytex` package by running the code below in your console.

```
install.packages("tinytex")
```

Resources and Next Session

Below are some links to a few videos that either I've created or that I feel would be useful. The last few are on version control using Git and GitHub. We'll go over this during our next session so only go over those if you have time.

The first two are playlists for videos created by DJ Navarro. Danielle is a computational coginitive scientist in Australia who also creates a lot of generative art with R.

- Playlist on Project Structure by DJ Navarro Video 2 in particular is about naming files.
- Playlist on RMarkdown by DJ Navarro This is a great intro to RMarkdown that assumes minimal experience with R.
- Using Git and GitHub via MIT opencourseware
- Setting up Git and GitHub Note: everything after the 5:25 mark is for a different project/not currently relevant to what we're doing.
- Setting up Git and GitHub (slides)