The Note for the RStudio Presenter

Shuai Wang

Sunday, May 17, 2015

This note is edited from rStudioPresent index.MD file from the repo of this course.

RStudio Presentation

- RStudio created a presentation authoring tool within their development environment.
- If you are familiar with slidify, you will also be familiar with this tool
  - Code is authored in a generalized markdown format that allows for R code chunks
  - The output is an html5 presentation
  - The file index for the presenter file is .Rpres, which gets converted to an .md file and then to an html file if desired
  - There's a preview tool in RStudio and **GUIs** for publishing to Rpubs or viewing/creating an html file

Authoring content

- This is a fairly complete guide
  - http://www.rstudio.com/ide/docs/presentations/overview
- Quick start is
  - `file` then `New File` then `R Presentation`
  - (`alt-f` then `f` then `p` if you want key strokes)
  - Use `===` (three equal sign) to separate slides (of course you can use more equal signs) with slide title before the line with three equal sign
  - Use basically the same R markdown format for authoring as slidify/knitr (they have all these things that will help walk you through it)
    - Single quotes for inline code
    - Tripple qutoes for block code
    - Same options for code evaluation, caching, hiding etcetera

GUI

On the presentation sheet (preview sheet), click arrow to turn slide pages, click magnifier to maximize the presentation window, click edit (the notepad icon) to be brought to the location of the slide you want to edit, and there are more options in the `More` button. Click `?` and `Markdown Quick Reference` to see help pages.

Compiling and tools

- R Studio auto formats and runs the code when you save the document (if you want to make sure, click refresh button)
- Mathjax JS library is loaded by default so that `$x^2$` yields x^2
- Slide navigation button on the preview; clicking on the notepad icon takes you to that slide in the deck (you don't get nice html effect on the preview pane, and you have to view it in the browser)

- Clicking on `more` yields options for
  - Clearning the knitr cache
  - Viewing in a browser (creates a temporay html file in `AppData/local/temp` for the teacher, does not create a html file to save where you want)
  - Create a html file to save where you want
- A refresh button
- A zoom button that brings up a full window

Visuals

- R Studio has made it easy to get some cool html5 effects, like cube transitions with simple options in YAML-like code after the first slide such as `transition: rotate` (the option specified after the first slide becomes the default)
- You can specify it in a slide-by-slide basis
- Just put `transition: linear` right after the slide creation (the line after three equal signs or more in a row)
- Tansition options
  - http://www.rstudio.com/ide/docs/presentations/slide_transitions_and_navigation

Hierarchical organization

- If you want a hierarchical organization structure, just add a `type: typename` option (e.g. `type: section`, `type: subsection`) after the slide (the line after the three equal sign)
- This changes the default appearance
  - http://www.rstudio.com/ide/docs/presentations/slide_transitions_and_navigation

Two columns

- Do whatever for column one
- Then put `***` on a line by itself with blank lines before and after
- Then do whatever for column two

Changing the slide font

- Add a `font-family: fontname` option after the slide (it is exactly the same way you would specify the font family in a html document, e.g. `font-family: 'Risque'`)
  - http://www.rstudio.com/ide/docs/presentations/customizing_fonts_and_appearance
- Specified in the same way as css font families (so you can go there to find out the fonts to be used)
  - http://www.w3schools.com/cssref/css_websafe_fonts.asp
- Use `font-import: url` to import fonts
- Important caveats
  - Fonts must be present on the system that you're presenting on, or it will go to a fallback font
  - You have to be connected to the internet to use an imported font (so don't rely on this for offline presentations)
- The url to be imported for font family `Risque`
  - http://fonts.googleapis.com/css?family=Risque

Really changing things

- If you know html5 and CSS well, then you can basically change whatever you want
- A css file with the same names as your presentation (in the same directory) will be autoimported
- You can use `css: file.css` to import a css file
- You have to create named classes and then use `class: classname` to get slide-specific style control from your css
  - (Or you can apply then within a `<span>` to control a subset of your slides)
- Ultimately, you have an html file, that you can edit as you wish
  - This should be viewed as a last resort, as the whole point is to have reproducible presentations, but may be the easiest way to get the exact style control you want for a final product

Slidify versus R Studio Presenter

Slidify

- Flexible control from the RMD file
- Under rapid ongoing development
- Large user base (there are lots of questions answered on Stack Exchange)
- Lots and lots of styles and options
- Steeper learning curve
- More command-line oriented

R Studio Presenter

- Embedded in R Studio
- More GUI oriented
- Very easy to get started
- Smaller set of easy styles and options
- Default styles look very nice
- Ultimately as flexible as slidify with a little CSS and HTML knowledge

The big step is deciding that you want a reproducible presentation with R embedded in your presentation building software, and then either of these solutions will work quite well.

Publish in Github Manually Using gh-pages

You can host your presentation directly off of Github. You can do it via publish command, but if you want to do it manually, you need create a Github branch named `gh-pages` and use some Git commands. In the following commands, suppose you have already pushed your presentation html file (testPres.html for example here) and its related files to your master repository.

`git branch gh-pages` (to create the gh-pages branch)

`git branch` (to view the branch under your master)

`git checkout gh-pages` (to switch to the branch in the remote Github)

`git push origin gh-pages` (to push the same things in your master to the branch)

`touch .nojekyll` (This file tells Github just do straight html without fancy stuff. After creating it locally, you need add, commit, and push it to the Github branch.)

Now open a browser, go to the URL with format `http://USERNAME.github.io/REPONAME/HTMLFILENAME` (in our case it is `bcaffo.github.io/testPres/testPres.html#/`), and press F11 to switch to full screeen. This way you can host your presentation directly off of Github.

This method works for both slidify and RStudio presenter. In fact, it is exactly what publish command does for slidify. But if Windows cannot find the Git path on your computer (may be configured incorrectly), publish command will fail, but you can do it manually like this.