The Magic of Markdown

Ted Laderas (laderast@ohsu.edu)

March 16, 2016

Introduction

- What is Markdown?
  - Simple Intro
- Pandoc
  - Basics of pandoc
  - Getting References to Work
- Rmarkdown
  - PDFs
  - Interactive Slides
- GitHub Pages
  - Jekyll

Where are These Slides?

`https://github.com/laderast/magic-of-markdown`

What is Markdown?

John Gruber (of Daring Fireball) originally created Markdown as a simple replacement for HTML for use in message board posts to allow for better formatting.

The wonderful thing about Markdown is that it doesn't get in your way - you can just get in and edit documents using any text editor. Once you learn it, you find that you can work very fast and not worry about how things look on the page until later.

You can worry about translating the markdown to other formats later, which is handled by an engine called Pandoc.

Markdown Variants

Markdown was originally made to be a substitute for HTML in forums. The original implementation doesn't really cover formatting such as tables.

GitHub also uses their own flavor of Markdown (called GitHub markdown) as the main format for their webpages, which makes maintaining them much easier than having to edit raw HTML. They extended markdown so you can add tables and better code formatting.

A variation of GitHub markdown is Rmarkdown, which is Markdown + R. Rmarkdown is really useful for reproducible analyses, and it can also be used in conjunction with Shiny to make interactive slides.

Follow Along!

Go to this Hackmd.io page (`http://bit.ly/2lZP9Ww`) to play around with a group edited markdown file (click on the pencil to edit).

Have fun, (but be SFW)!

Markdown Basics (Formatting)

When in doubt, look at this quick markdown cheat sheet that covers both plain markdown and GitHub Markdown.

```
*Italicise your text*
```

```
_Italicise your text_
```

Italicise your text

```
**Bold your text**
```

```
__Bold your text__
```

Bold your text

Markdown Basics (Bullets)

```
* Will add bullets
 * Another Depth
```

- Adding bullets
    - Another Depth

```
1.  Numbered Lists
2.  Another Number
```

1. Numbered Lists
2. Another Number

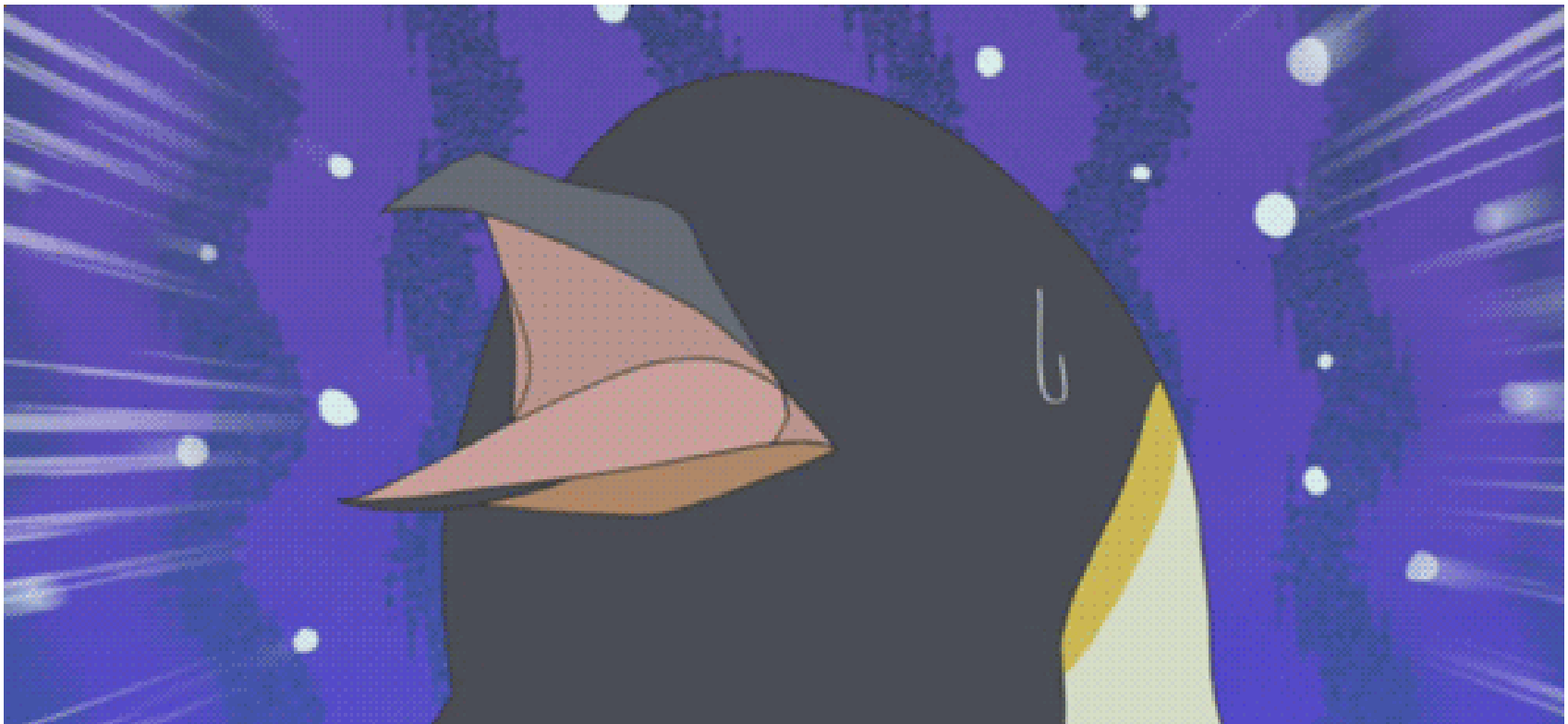
Markdown Basics (Links and Images)

```
Links are automatically generated for URLs:
http://yahoo.com http://yahoo.com
```

```
[Link to Yahoo](http://yahoo.com)
```

Link to Yahoo

```
![Add an Image](images/giphy.gif)
```



Markdown Basics (Code Formatting)

Use three backticks “‘ to enclose a code block:

```
Put Code Here
Put Code Here
for(i in 1:5){
  print(paste0("I said 'I said' ", 2*i, "times"))
}
```

```
Put Code Here
for(i in 1:5){
  print(paste0("I said 'I said' ", 2*i, "times"))
}
```

Use a single backtick ‘ to enclose `in-line code`

Markdown Basics (Blockquotes and Escape Characters)

Use > for blockquotes:

```
> This is a block quote
> continued on the next line
```

> *This is a block quote Continued on the next line*

The \ is an escape character and allows you to use *, > and other markdown characters as is.

`Example: \` * \#`

Example: ‘ * #

The dirty secret about Markdown

You can easily mix Markdown with HTML. For example, if you wanted to control the placement and size of an image, you can use the `<img>` tag to control the formatting.

Let's try it out!

Go to this Hackmd.io page (`http://bit.ly/2lZP9Ww`) to play around with a markdown file (click on the pencil to edit).

Have fun, (but be SFW)!

Pandoc

Pandoc is a wonderful piece of software. Essentially, it allows you to translate Markdown documents to a huge number of formats, including:

- HTML documents
- Word (.doc and .docx)
- PDF documents (via LaTEX)
- PDF presentations (via LaTeX)
- HTML presentations (ioslides, slidy)
- MediaWiki markup
- LibreOffice format
- Many more!

Using Pandoc

Invoking Pandoc is done on the command line, but it's easy to do:

```
pandoc -f markdown -t latex input.md -o output.pdf
```

- -f = "from" format
- -t = "to" format
- -o specify output file name. Here, we use "output.pdf"

Here we output an HTML file:

```
pandoc input.md -f markdown -o output.html
```

Note that the default output is html so we don't need to specify the -t flag here.

Here we output a PDF file via LaTeX (this won't work unless you have LaTeX installed)

```
pandoc input.md -f markdown -t latex -o output.pdf
```

Good Markdown Editors

You can edit markdown in any text editor. However, it's worth using a markdown capable editor that at least has syntax highlighting.

I currently use Atom, but Sublime and many other text editors support markdown highlighting and preview.

Examples include:

- Sublime Text
- Multimarkdown
- Writemonkey
- Texts

YAML

YAML* is another way of providing Pandoc the necessary metadata it needs (output format, location of BibTeX library, other executables, etc). For more info, consult the pandoc documentation.

You add YAML as a header to the document by specifiying three dashes: `---`

Here's the YAML that I used for this set of slides. I suggest that you use a YAML Linter such as YAMLLint to check that you formatted your YAML correctly, since it's a picky format (improperly placed colons can break it).

```
---
title: "The Magic of Markdown"
author: "Ted Laderas"
date: "March 16, 2017"
output: slidy_presentation
---
```

- 'YAML ain't markup language' - har har

Formatting Templates

If you're not satisfied with the default look of the output, you can customize the different documents using various templates.

Pandoc Templates

Markdown and LaTeX

What about equations and all the other jazz?

```
$$ \frac{1}{n} \sum_{i=1}^{n} x_{i} $$
```

$$\frac{1}{n} \sum_{i=1}^{n} x_i$$

You can integrate LaTeX equations into your code, no problem.

I don't have time to go through it, but here are some tutorials.

- Integrating Markdown and LaTeX
- How to Write a dissertation in LaTeX using markdown

Markdown and Citation Managers

Markdown works well with Zotero, using the pandoc-citeproc extension: `https://github.com/jgm/pandoc-citeproc`

I've written a tiny example with installation instructions here: Using Zotero with Pandoc

I think there is a workflow for Mendeley as well, but I haven't used it yet.

RMarkdown

Rmarkdown is an R-specific version of GitHub markdown (Technically it's based on sundown, but who cares.).

It's used a lot in making analyses and reports reproducible. It allows for Markdown formatting mixed with R analysis code. For this reason, it's ideal for sharing complex analyses with other people.

RMarkdown is translatable to PDF (via LaTeX), to PDF slides, HTML reports, and HTML slides within RStudio.

RStudio has pandoc built in for this purpose.

Before you start

Make sure that your R has the `knitr` package installed.

```
install.packages(knitr)
```

Rmarkdown Example

The key difference in Rmarkdown is in the codeblock, which actually executes code.

````
```{r, eval=TRUE}
data(iris)
plot(iris[,1], iris[,2])
```
````

Lots of codeblock options

We've already seen the `eval=TRUE` option for the R codeblock. But there are lots of others to help you customize your presentation of code and graphs:

- `echo=FALSE`
- `fig.size=5`
- `message=FALSE`
- `tidy=TRUE`
- `warning=FALSE`

Rstudio and Pandoc

RStudio actually has pandoc built in, with a limited set of options. If you want your markdown to execute code, you will have to use this version of pandoc. I usually just use Rstudio's "Knit" option to translate my documents.

If you have set your YAML correctly at the beginning of the file, pandoc will also process your references as well.

You can also use the following command to render a document. Make sure your YAML specifies the options you want (i.e. PDF, html_output). Note that to output PDF files, you will have to install LaTeX on your system.

```
rmarkdown::render("input.Rmd")
```

Rstudio/Markdown Example:

Open this file in RStudio: `http://github.com/laderast/magic-of-markdown/Rmarkdown-example.Rmd`

Shiny and Rmarkdown

What's especially cool is that you can mix Shiny and Rmarkdown to produce interactive slides.

You can embed Shiny applications into code blocks and then run the resulting code on a Shiny server, such as the one available on church.

I'm experimenting with this here: `http://church.ohsu.edu:3838/laderast/clusteringLecture/`

GitHub Pages

Another nice use of Markdown is for authoring websites. You can do this really easily with GitHub pages.

GitHub Pages uses an HTML translation engine called *Jekyll*, which is written in Ruby to translate Markdown into HTML. You don't really need to know about it, unless you want to use the templating in Jekyll to build something like a blog.

One benefit to using Jekyll is that it's a relatively lightweight way to manage a blog without a more complicated, database-using, content management system. Because of this, it's relatively portable and you can easily take your documents with you if you decide to migrate to another system.

Markdown and GitHub pages

Any markdown file that has an .markdown or .md extension will automatically be processed by GitHub and served as a html. For example: `https://github.com/laderast/magic-of-markdown/blob/master/magic-of-markdown.md`

If you want to make a webpage for your code, you need to create a branch called `gh-pages` and put the markdown files in there. I believe you can also just place html files as well.

Setting Up a Personal GitHub Page

If you have a GitHub account with USERNAME, you can serve a webpage directly from USERNAME.github.io. For example: `http://laderast.github.io`

You first need to create a repository with that name. Anything with index.md will be automatically translated and served as a webpage.

Using Jekyll, you can add header and footers and other templating features, standardizing the look of your pages as you like.

One thing to remember is when your output is html, you can embed HTML tags, which is especially useful for embedding images, since you can adjust image size directly within the tag.

Jekyll File Structure

If you're interested in setting up a blog or more complicated site, you'll have to learn a little bit about Jekyll and the liquid templating language.

You can download Jekyll by itself to see how it transforms markdown files, but to get started with a blog or pages, it's actually way easier to clone Poole, which gives you the basic file structure that you can use to serve the GitHub pages.

For More Info

- This document: `http://github.com/laderast/magic-of-markdown/`
- Pandoc user guide: `http://pandoc.org/README.html`
- Pandoc and Zotero: `pandoc-zotero/notes.md`
- Rmarkdown: `http://rmarkdown.rstudio.com`
- Rmarkdown and Shiny: `http://rmarkdown.rstudio.com/authoring_shiny_widgets.html`
- GitHub Pages: `https://pages.github.com`
- Bookdown: `https://bookdown.org/yihui/bookdown/`
- Markdown Template for PhD Thesis: `https://github.com/tompollard/phd_thesis_markdown`