02 - Pandoc Markdown Syntax.Rmd

Pandoc Markdown Syntax

Pandoc supports a large number of input file formats for processing (including markdown, reStructuredText, textile, HTML, DocBook, LaTeX, MediaWiki markup, TWiki markup, OPML, Emacs Org-Mode, Txt2Tags, Microsoft Word docx, LibreOffice ODT, EPUB, or Haddock markup) into a vast number of output formats. For our purposes in this workshop we will focus on working with the extended markdown syntax supported by Pandoc as a language for creating content and generating multiple output representations.

For complete documentation of Pandoc's syntax refer to the Pandoc User's Guide.

Title Block and Additional Metadata

Pandoc provides an option for providing document metadata in two forms: as a simple `title block` at the beginning of a document, and also as a `YAML` metadata block that you can provide anywhere in the document, or even as an externally referenced document.

The `title block` provides basic title, author, and date information about the document that is used by many of the default Pandoc templates to automatically fill in specific document elements. For example, the `title` provided in the title block is used to set the `title` meta element in a generated HTML file and for a `title` block in a LaTeX document. The authorship and date information is similarly used to automatically generate content for some output document types. The `title block` has a simple three part structure, each of which is required if a `title block` if a subsequent element is used. The format of a title block is:

```
% title
% author(s) (separated by semicolons)
% date (treated as a text string)
```

These elements (and any others that you would like to create and use) may also be placed in a `YAML` metadata block:

```
---
title: title
author: author(s) (separated by semicolons)
date: date (treated as a text string)
...
```

Outline Structure

The section of the Pandoc User's Guide relating to headers can be found here.

A key element of any document that you create is the hierarchical structure representing the document outline. This hierarchy is used to deliniate content sections and the styles used to present the content of those sections. Examples of the use of the outline structure include:

- mapping of the hierarchy levels into corresponding HTML header elements - `<h1>`, `<h2>`,`<h3>`, etc.
- defining chapters and sections of generated PDF files
- defining title slides, slides and slide content for slide shows

When defining the outline structure of your documents focus on the structure and not intended representation of your content. Other elements of the markdown syntax can be used to create effects like emphasys or strong

emphasys, and additional options are available through very granular control that you can exercise through customized templates (e.g. for HTML and LaTeX generated PDF files) and CSS (for HTML and EPubs) stylesheets.

There are a couple of options for specifying outline levels:

```
# Level One
## Level Two
### Level Three
...
###### Level Six
```

and

```
A Level One Header
==================

A Level Two Header
------------------
```

How youre selected header levels are rendered will depend upon the template and format used when translating the markdown into a specific representation.

Note: while standard markdown allows for having no empty line before a header, Pandoc required a blank line before a header.

Optional Header Attributes

You can also specific optional header attributes as part of your document. These attributes can be used to assign `id` and `class` values to headers that can be used when creating custom HTML styles, provide labels used in automatically generated Table of Contents content, and other output-format-specific elements. The attributes for a header are specified using the following syntax:

```
{#identifier .class .class key=value key=value}
```

which is added following the end of the header text.

Blocks of Content

Block of content fall into a number of categories including *paragraphs*, *block quotes*, a variety of *lists*, *verbatim* and *line blocks*. Each of these content block types has its own syntax for specifying the content.

Paragraphs

The section of the Pandoc User's Guide relating to paragraphs can be found here.

The syntax for paragraphs is very simple - just blocks of text separated from each other by blank lines. By default line breaks will be ignored so you can type your paragraph as multiple (contiguous) lines and expect it to be rendered as a single paragraph in the output representation. Paragraph breaks are automatically created when a blank line is encountered at the end of a block of contiguous lines of text.

Block Quotes

The section of the Pandoc User's Guide relating to block quotes can be found here.

Block quotes are defined through the use of the `>` character at the beginning of the text that is intended for inclusion in the block quote. The `>` may appear at the start of each line, or a *lazy* may be used in which only the first line of a block of contiguous text lines is preceeded by a `>`. For example:

```
> this is a
> multiline quote
```

> this is a multiline quote

which is treated similarly to

```
> this is another
multiline quote
```

> this is another multiline quote

In either case, the quote will be styled according to the specifications of the output file format.

Lists

The section of the Pandoc User's Guide relating to lists can be found here.

A variety of list types are supported by Pandoc, including nesting of lists of different types. Nesting is defined through the use of four spaces or one tab to indent the list item. Individual list items may contain multiple paragraphs (each separated by a blank line) and preceeded by *four spaces*. Because of the indentation rule for multi-paragraph list items, `code blocks` included in a list item must be indented *eight spaces*. These lists include:

Unnumbered Lists List items in unnumbered (bulleted) lists are preceeded by a `*` or `+` character followed by a space.

```
* List Item 1
* List Item 2
    * List Item 2a
    * List Item 2b
* List Item 3
```

Which will be rendered like:

- List Item 1
- List Item 2
  - List Item 2a
  - List Item 2b
- List Item 3

Numbered/lettered Lists Numbered and lettered lists may be preceeded by numbers or lower- and upper-case letters, any of which may be enclosed in parentheses, or followed by a closing parenthesis or period followed by a space (or two-spaces if the preceeding character is an upper-case. The specific characters or numbers used aren't used (except for the first one in the list which defines the beginning of the sequence), but instead are used to define the desired auto-generated list item prefix values. To obtain the default list numbers you can preceed each item with a `#.` instead of a character or number.

```
1. List Item 1
1. List Item 2
    a. List Item 2a
    a. List Item 2b
1. List Item 3
```

Even when not indented, a new list will be created when the list type changes.

Definition Lists Lists that are specifically designed for providing terms and their definitions are also supported by Pandoc's mardown syntax. An example (others are available) of a compact syntax for a defintion list is provided here:

```
Term 1
  ~ Definition 1
```

```
Term 2
  ~ Definition 2a
  ~ Definition 2b
```

Example Lists Example lists may be created within a document. The elements of an example list may be created throughout the document with the examples being numbered sequentially throughout the document. Example list elements may be given labels that may be used to cross-reference examples elsewhere in the document.

```
(@)  My first example will be numbered (1).
(@)  My second example will be numbered (2).

Explanation of examples.

(@)  My third example will be numbered (3).

(@good)  This is a good example.

As (@good) illustrates, ...
```

Code and Other Verbatim Text

The section of the Pandoc User's Guide relating to verbatim text can be found here.

There are times when you want to display text exactly as entered instead of having line breaks or other modifications made to the text structure. While this is commonly used for displaying code, it is also helpful when you want to present a block of text where there is a specific line structure that needs to be maintained.

Code Blocks Code blocks can be defined by either indenting each line by four spaces or a tab, or by creating a *fenced code block* by placing the text between lines with three or more tilde characters (~) and the closing set being at least as long as the opening set.

```
    pandoc -s -S \
    --normalize \
    --filter pandoc-citeproc \
    --csl ./science.csl \
    --natbib \
    --template=poster.tex \
    -f markdown+raw_tex \
    -o 2016-12_AGUPoster.tex \
    2016-12_AGUPoster.md
```

This is an example of a *fenced code block*

```
~~~
pandoc -s -S \
--normalize \
--filter pandoc-citeproc \
--csl ./science.csl \
--natbib \
--template=poster.tex \
-f markdown+raw_tex \
-o 2016-12_AGUPoster.tex \
2016-12_AGUPoster.md
~~~
```

Some renderers support syntax coloring for different programming languages. Pandoc allows for the specification of attributes for a fenced code block to provide hints for renderers as to what language is being presented and whether line numbers should be provided in the output:

```
~~~~ {#mycode .bash .numberLines startFrom="15"}
pandoc -s -S \
--normalize \
--filter pandoc-citeproc \
--csl ./science.csl \
--natbib \
--template=poster.tex \
-f markdown+raw_tex \
-o 2016-12_AGUPoster.tex \
2016-12_AGUPoster.md
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

Line Blocks An alternative method for controlling the flow of text without the semantic implication of `code` is to use *line blocks* to define blocks of text for which line breaks and included indentation is maintained.

```
| Karl Benedict
| MSC05 3020
| 1 University of New Mexico
| Albuquerque, NM
|              98131
```

Inline Text Formatting

The section of the Pandoc User's Guide relating to inline text formatting can be found here.

Pandoc markdown supports a number of methods for altering the appearance of text within heading, paragraph, block quote, and list blocks. These methods enable some of the standard text modifications you are accustomed to using in word processors, though with an emphasys on the *semantic* specification (for example *emphasys* or **strong emphasys**) of text modifications instead of directly specifying (*italic* or **bold**). While a bit of a nuanced point, it is important to think about this difference as these *semantic* directives get translated into different representations when output files are generated. You can specify the following inline text formats:

- *Emphasys*

  ```
  Here is an emphasized *word*

  This is another way to define an emphasized _word_
  ```

  Here is an emphasized *word*

  This is another way to define an emphasized *word*

- **Strong Emphasys**

  ```
  Here is a strongly emphasized **word**

  This is another way to define a strongly emphasized __word__
  ```

  Here is a strongly emphasized **word**

  This is another way to define a strongly emphasized **word**

- Superscript and subscript text

  Superscript text is surrounded by ^ characters:

`e=mc^2^`

Subscript text is surrounded by `~` characters:

`H~2~O`

- Strikeout text

  Strikeout text is surrounded by `~~` (double tilde) characters:

  `I meant this instead of ~~that~~`

  I meant this instead of ~~that~~

- Inline verbatim text

  Verbatim text is surrounded by "`" (back-tick) characters:

  `You should enter `pandoc` on the command line to start a document processing command.`

  You should enter `pandoc` on the command line to start a document processing command.

  ## Footnotes

The section of the Pandoc User's Guide relating to footnotes can be found here.

Footnotes are supported by Pandoc through both *inline* footnotes and two-part footnote definitions. Inline footnotes are very convenient for defining a relatively short bit of text (or other content) within the flow of a text block while two-part footnotes may be used to define footnote content that is larger or more complex than you can place within the flow of a text block. Both inline and two-part footnotes may be mixed within a single document. The specific rendering of the footnotes would need to be modified through updates to the template used to generate the output.

Inline Footnotes

The syntax for inline footnotes consists of a combination of a `^` followed by a pair of square brackets (`[]`) that contain the content of the footnote.

```
The Google^[http://www.google.com] search engine may be used to ...
```

where the contents of the inline footnote - `http://www.google.com` - would be added to the generated output using a representation appropriate for the specific output format.

Two-part Footnotes

Two-part footnotes consist of an inline identifier that is placed within square brackets (`[]`) that starts with a `^` symbol followed by the remainder of the identifier text (which cannot include spaces). Elsewhere in the document you can place the actual footnote text on its own line. The footnotes will be automatically numbered based on the order of the references within the document.

```
This my text with a two-part footnote[^1]. The actual actual label text used
doesn't matter, except that it can't contain spaces, tabs or newlines[^ref].

[^1]: this is the text for the first footnote
[^ref]: this is the text for the secondd footnote.
```

Tables

The section of the Pandoc User's Guide relating to tables can be found here.

Pandoc supports a number of methods for defining the structure and content of tabular content within documents. A review of the different syntaxes for specifying tables (here) is recommended if you need to

provide more than the most simple tables as part of your output. A basic table production example is provided here:

```
  Right     Left     Center     Default
-------     ------ ----------   -------
     12     12        12            12
    123     123       123          123
      1     1          1             1
```

This simple table format illustrates the use of column headings (which are optional), a row of dashed lines that define the top of the table and provide a reference for how the content of the table rows should be aligned within each column, and the content of each table row - note that the position of the header row columns determines the alignment of each column. If there are no headers the alignment of the first row of values determines the column alignment.

An Example :

Following is an example of these various content elements within a markdown document.

```
% title
% author(s) (separated by semicolons)
% date (treated as a text string)

# Level One

Fatback labore cupidatat meatball quis. Consequat kevin commodo ipsum laborum ham hock aute jerky pastra

## Level Two

Chuck pork consequat, biltong pork loin meatball pancetta brisket commodo anim. Cupidatat capicola panc

### Level Three

Officia alcatra anim fugiat. Laborum bresaola shoulder beef doner pork belly et burgdoggen. Commodo swi

#### Level Four

Cillum turducken consectetur ut tri-tip short ribs t-bone meatloaf venison cupidatat labore. Aliqua tai

##### Level Five

Capicola hamburger duis minim. Andouille sed dolore sunt voluptate exercitation bacon anim eu capicola

###### Level Six

Labore eiusmod magna chuck occaecat ribeye pig sed. Occaecat nisi id ut deserunt anim drumstick pastram

Level One
=========

Fatback labore cupidatat meatball quis. Consequat kevin commodo ipsum laborum ham hock aute jerky pastra

Level Two
---------

Chuck pork consequat, biltong pork loin meatball pancetta brisket commodo anim. Cupidatat capicola panc
```

```
-----------------------------------------------



> this is a
> multiline quote

> this is another
multiline quote

* List Item 1
* List Item 2
    * List Item 2a
    * List Item 2b
* List Item 3

Term 1
  ~ Definition 1

Term 2
  ~ Definition 2a
  ~ Definition 2b

(@)  My first example will be numbered (1).
(@)  My second example will be numbered (2).

Explanation of examples.

(@)  My third example will be numbered (3).

(@good)  This is a good example.

As (@good) illustrates, ...

    pandoc -s -S \
    --normalize \
    --filter pandoc-citeproc \
    --csl ./science.csl \
    --natbib \
    --template=poster.tex \
    -f markdown+raw_tex \
    -o 2016-12_AGUPoster.tex \
    2016-12_AGUPoster.md

~~~
pandoc -s -S \
--normalize \
--filter pandoc-citeproc \
--csl ./science.csl \
--natbib \
--template=poster.tex \
-f markdown+raw_tex \
-o 2016-12_AGUPoster.tex \
```

````
2016-12_AGUPoster.md
~~~

~~~~ {#mycode .bash .numberLines startFrom="15"}
pandoc -s -S \
--normalize \
--filter pandoc-citeproc \
--csl ./science.csl \
--natbib \
--template=poster.tex \
-f markdown+raw_tex \
-o 2016-12_AGUPoster.tex \
2016-12_AGUPoster.md
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


| Karl Benedict
| MSC05 3020
| 1 University of New Mexico
| Albuquerque, NM
|              98131

-----------------------------------------------



Here is an emphasized *word*

This is another way to define an emphasized _word_

Here is a strongly emphasized **word**

This is another way to define a strongly emphasized __word__

e=mc^2^

H~2~O

I meant this instead of ~~that~~

You should enter `pandoc` on the command line to start a document processing command.

-----------------------------------------------

The Google^[http://www.google.com] search engine may be used to ...

This my text with a two-part footnote[^1]. The actual actual label text used
doesn't matter, except that it can't contain spaces, tabs or newlines[^ref].

[^1]: this is the text for the first footnote
[^ref]: this is the text for the secondd footnote.

-----------------------------------------------
````

```
  Right     Left     Center     Default
-------     ------ ----------   -------
     12     12        12            12
    123     123       123          123
      1     1          1             1
```

and rendered in an RMarkdown document

==========

% title % author(s) (separated by semicolons) % date (treated as a text string)

Level One

Fatback labore cupidatat meatball quis. Consequat kevin commodo ipsum laborum ham hock aute jerky pastrami deserunt cillum non turducken. In irure bresaola cupidatat laborum alcatra. In flank do labore kevin, filet mignon in. Dolore chicken beef ribs ribeye cow. Commodo cupidatat shankle laboris, exercitation pastrami magna porchetta.

Level Two

Chuck pork consequat, biltong pork loin meatball pancetta brisket commodo anim. Cupidatat capicola pancetta, excepteur ribeye ex hamburger prosciutto elit filet mignon. Aliquip voluptate est occaecat pancetta meatloaf sunt commodo laborum spare ribs. Cupidatat commodo shank, culpa fugiat veniam dolore dolor consequat. Beef shank landjaeger short loin. Nisi flank ad, alcatra prosciutto consequat ullamco.

Level Three

Officia alcatra anim fugiat. Laborum bresaola shoulder beef doner pork belly et burgdoggen. Commodo swine culpa ad shank voluptate cow kevin elit strip steak minim. Swine ullamco burgdoggen chuck est occaecat dolore meatball reprehenderit deserunt jerky adipisicing. Sed venison aliquip officia short ribs. Irure drumstick tempor reprehenderit kielbasa brisket. Ea swine kielbasa pork chop picanha pancetta.

Level Four Cillum turducken consectetur ut tri-tip short ribs t-bone meatloaf venison cupidatat labore. Aliqua tail esse, filet mignon cupim drumstick ut nostrud. Ribeye laborum aliquip, ad voluptate aliqua biltong commodo pig burgdoggen nulla. In boudin chuck aute labore ad alcatra pig deserunt strip steak picanha mollit shank nostrud burgdoggen. Commodo tongue deserunt, brisket ball tip voluptate magna turducken.

Level Five Capicola hamburger duis minim. Andouille sed dolore sunt voluptate exercitation bacon anim eu capicola sausage burgdoggen brisket. T-bone pastrami lorem, in short loin dolore sed pork chop incididunt turducken exercitation. Quis ribeye boudin, dolore turkey shoulder do rump aliquip picanha adipisicing. Aute enim ex, ad corned beef aliqua in eiusmod culpa incididunt.

Level Six

Labore eiusmod magna chuck occaecat ribeye pig sed. Occaecat nisi id ut deserunt anim drumstick pastrami cow. Meatloaf culpa pork loin swine nisi, dolore tri-tip sirloin andouille nostrud salami tongue lorem porchetta. Pastrami kielbasa landjaeger tenderloin.

Level One

Fatback labore cupidatat meatball quis. Consequat kevin commodo ipsum laborum ham hock aute jerky pastrami deserunt cillum non turducken. In irure bresaola cupidatat laborum alcatra. In flank do labore kevin, filet mignon in. Dolore chicken beef ribs ribeye cow. Commodo cupidatat shankle laboris, exercitation pastrami magna porchetta.

Level Two

Chuck pork consequat, biltong pork loin meatball pancetta brisket commodo anim. Cupidatat capicola pancetta, excepteur ribeye ex hamburger prosciutto elit filet mignon. Aliquip voluptate est occaecat pancetta meatloaf sunt commodo laborum spare ribs. Cupidatat commodo shank, culpa fugiat veniam dolore dolor consequat. Beef shank landjaeger short loin. Nisi flank ad, alcatra prosciutto consequat ullamco.

> this is a multiline quote
>
> this is another multiline quote

- List Item 1
- List Item 2
  - List Item 2a
  - List Item 2b
- List Item 3

Term 1 Definition 1
Term 2 Definition 2a
Definition 2b

(1) My first example will be numbered (1).
(2) My second example will be numbered (2).

Explanation of examples.

(3) My third example will be numbered (3).

(4) This is a good example.

As (4) illustrates, ...

```
pandoc -s -S \
--normalize \
--filter pandoc-citeproc \
--csl ./science.csl \
--natbib \
--template=poster.tex \
-f markdown+raw_tex \
-o 2016-12_AGUPoster.tex \
2016-12_AGUPoster.md
```


```
pandoc -s -S \
--normalize \
--filter pandoc-citeproc \
--csl ./science.csl \
--natbib \
--template=poster.tex \
-f markdown+raw_tex \
-o 2016-12_AGUPoster.tex \
2016-12_AGUPoster.md
```


```
pandoc -s -S \
--normalize \
--filter pandoc-citeproc \
--csl ./science.csl \
--natbib \
--template=poster.tex \
-f markdown+raw_tex \
```

```
-o 2016-12_AGUPoster.tex \
2016-12_AGUPoster.md
```

Karl Benedict
MSC05 3020
1 University of New Mexico
Albuquerque, NM
98131

Here is an emphasized *word*

This is another way to define an emphasized *word*

Here is a strongly emphasized **word**

This is another way to define a strongly emphasized **word**

e=mc^2

H_2O

I meant this instead of ~~that~~

You should enter `pandoc` on the command line to start a document processing command.

The Google[1] search engine may be used to ...

This my text with a two-part footnote[2]. The actual actual label text used doesn't matter, except that it can't contain spaces, tabs or newlines[3].

Right	Left	Center	Default
12	12	12	12
123	123	123	123
1	1	1	1

[1] http://www.google.com
[2] this is the text for the first footnote
[3] this is the text for the secondd footnote.