# How to publish with R Markdown in WordPress

*3.14a*

*Sunday, March 03, 2015*

## Contents



### Introcuction

With *R* and *Markdown* one can create reports for HTML, PDF or even Word (see RStudio Markdown,Markdown). In addition to the comfortable generation of Markdown reports (there is also a LaTeX based version) it is also possible to directly generate WordPress blog entries from R with R Markdown (.Rmd files). This is possible with the RWordPress package that allows to send parts of R Markdown to a WordPress blog.

**Download**: PublishBlogPosts.pdf

**Template**:
Use the template file to upload your R Markdown files to your WordPress site. First replace Replace in the *wordpress* chunk on the top of the template file with your WordPress login parameters and url. Then run the *wordpress* chunk on the top of the template file to publish the content of the template file on your WordPress website.

**Download:** RMarkdownTemplate.Rmd

### Publish with R Markdown in WordPress

In a first step on the WordPress website *remote publishing* needs to be enabled. In the WordPress admin area go to *Settings / Writing / Remote Publishing* and enable the *XML-RPC* option.

In the packages RWordPress and knitr provide the necessairy functions for publishing a .Rmd file on WordPress. To publish with R the *username* (typically 'admin'), *password* and the *url* of the WordPress site are required and set in `options`.

`knit2wp` converts the R Markdown to html and uploads the new entry to the WordPress blog. The function returns the postid, in case the user wants to update the blog post at a later point in time.

Supported options: categories, mt_keywords, mt_excerpt, mt_allow_comments ("open","closed"), mt_allow_pings ("open","closed"), wp_post_format ("standard",..). To include `custom_fields` use `custom_fields=list(value=list(id="444",key="myKey",value="myValue"))`(id is optional). For more information see: XML-RPC_MetaWeblog_API

```
# Install RWordpress
if (!require('RWordPress')){
  install.packages('RWordPress', repos = 'http://www.omegahat.org/R', type = 'source')}
library(RWordPress)
```

```
# Set login parameters (replace admin,password and blog_url!)
options(WordpressLogin = c(admin = 'password'), WordpressURL = 'blog_url/xmlrpc.php')

# Include toc (comment out if not needed)
library(markdown)
options(markdown.HTML.options =  c(markdownHTMLOptions(default = T),"toc"))

# Upload plots: set knitr options
opts_knit$set(upload.fun = function(file){library(RWordPress);uploadFile(file)$url;})

# Upload featured image / post thumbnail: option: wp_post_thumbnail=postThumbnail$id
postThumbnail <- RWordPress::uploadFile("figure/post_thumbnail.png",overwrite = TRUE)

# Post new entry to the wordpress blog and store the post id
library(knitr)
postid <- knit2wp('PublishBlogPosts.Rmd', title = 'How to publish with R Markdown in
  WordPress', categories=c('R'),mt_keywords = c('knitr', 'wordpress'),
  wp_post_thumbnail=postThumbnail$id,publish=FALSE)
```

**Include TOC**:
Include table of contents by adding the `toc` option in `markdown.HTML.options` in the Markdown package.

**Upload plots**:
To upload the plot files to WordPress set the `upload.fun` to use the RWordPress uploadFile function. The image names of the plots are derived from the chunk name `{r chunkname}` that generate the plot, if not provided the file are rather cryptic: *wpid-unnamed-chunk-8-1.png*. (The default *knitr* settings (option: standalone) for images directly encodes the image in the html `img` tag with base 64 which encoded e.g.: `<img src="...s=">`. For posting R Markdown in WordPress one rather prefers to upload the image files.)

**Upload featured image / post thumbnail**:
In WordPress a featured image can be associated with a post, which acts as the posts representative image. In order to link an uploaded image to the post, the *id* of the image is required. The `uploadFile` function returns the media id of the uploaded image along with the filename, url and file type. In `knit2wp` include the parameter `wp_post_thumbnail=postThumbnail$id` which associates the featured image to the post. **Note:** Featured images are only visible on your WordPress site if the current theme supports them.

**Alternative Plot upload options**
An alternative to upload the images/plots to WordPress are Imgur, flickr or dropbox.

- **Imgur**: The knitr package offers the function `imgur_upload()` to upload all generated images to imgur. Imgur upload
- **Flickr**: A similar function can be written to upload the image files to flickr. Flickr upload
- **Dropbox**: Another option is to store the files in your dropbox folder. E.g. create a *wp* folder inside your public dropbox folder and save the images to this location. Dropbox upload (3rd paragraph)

```
# Variant Imgur: upload all images to imgur.com
opts_knit$set(upload.fun = function(file) imgur_upload(file, key = "<your imgur key>"))

# Variant Dropbox: upload all images to dropbox
opts_knit$set(
  base.url = 'https://dl.dropbox.com/.../wp/', # online dropbox folder url
  base.dir = 'local/path/to/Dropbox/Public/wp/' # local dropbox folder
)
```

### Update existing post with R Markdown

To update an existing post, use the post id of the first upload or query the recent posts with `getRecentPostTitles` and retrieve the post id of the entry you want to update/overwrite.

**Note:** The action `editPost` seems to recreate the whole post entry and overwrites post title and further parameters such as categories or tags with default settings. Hence, it's best to get the complete post entry with `getPost` and re-upload the parameters with the `knit2wp` function.

```
# Get the post id from recent posts
posts <- getRecentPostTitles(num = 1, blogid = 0L, login = getOption("WordpressLogin",
  stop("need a login and password")))
postid <- as.character(posts[1,"postid"]) # assuming it's the last entry

# Get the post
post <- getPost(postid=postid, login = getOption("WordpressLogin",
  stop("need a login and password")))

# Edit the post (keep category,tags and title as before)
knit2wp('PublishBlogPosts.Rmd',postid=postid, action = c("editPost"),title=post$title,
  categories=post$categories,mt_keywords=post$mt_keywords,
  wp_post_thumbnail=post$wp_post_thumbnail,publish=FALSE)
```

### Upload additional files

While plot images are uploaded in a batch with the `knit2wp` function, further files need to be uploaded additionally. The following lines of code show how to upload a file to WordPress. Alternative options are to upload files to dropbox, github or other similar service provider and link to these files with Markdown.

**Example upload**: Create a datafile and upload it to WordPress. Note: For some reason the working uploadFile function triggers a rpc.serialize warning. To remove the warning include `warning=FALSE` in the code chunk option.

```
# create a testfile
d <- data.frame(a=1:10,b=rnorm(10))
write.table(d,file="dataset.csv",sep=";",row.names=FALSE,na = "",quote = FALSE)

# upload file to Wordpress
library(RWordPress)
fileUrl <- RWordPress::uploadFile("dataset.csv")$url
```

**Download:** dataset

```
# Provide a PDF of this article
library(rmarkdown)
render("RMarkdownWordpressTemplate.Rmd", "pdf_document")
pdfUrl <- RWordPress::uploadFile("PublishBlogPosts.pdf")$url
```

For security reasons not all filetypes are allowed for upload on WordPress. To add additional mime-types include the following function in `functions.php`.

```
/* Add Mime type support for additional files */
function custom_mime_types($mimes) {
  $mimes['rmd'] = 'text/plain';
  $mimes['r'] = 'text/plain';
  return $mimes;
}
add_filter('upload_mimes', 'custom_mime_types');
```

## Enable Math equations and Code highlighting in WordPress

In your WordPress admin area, go to Design -> Editor and select the file `functions.php` and include the following functions.

```
/* Add highlight.js support */
function highlight_load() {
 wp_enqueue_style( 'highlight',
   '//cdnjs.cloudflare.com/ajax/libs/highlight.js/8.4/styles/default.min.css');
 print "
    <script src='//cdnjs.cloudflare.com/ajax/libs/highlight.js/8.4/highlight.min.js'>
    </script>
    <script src='//cdnjs.cloudflare.com/ajax/libs/highlight.js/8.4/languages/r.min.js'>
    </script>
    <script>hljs.initHighlightingOnLoad();</script>";}
add_action('wp_footer','highlight_load');

/* Add MathJax support */
function mathjax_cdn() {
 wp_enqueue_script( 'mathjax-js',
    'http://cdn.mathjax.org/mathjax/latest/MathJax.js?config=TeX-AMS-MML_HTMLorMML',
    array('jquery'), '', true );}
add_action( 'wp_enqueue_scripts', 'mathjax_cdn' );

// Optional: Enable $-notation
function mathjax_config() {
 $config ="MathJax.Hub.Config(
    {tex2jax: {inlineMath: [['$','$'], ['\\\\(','\\\\)']], processEscapes: true}});";
 echo "<script type='text/x-mathjax-config'>" . $config . "</script>";
}
add_action('wp_head','mathjax_config',1);

/* WP editor */
// prevent removing code when changing between preview and raw text in the editor
function init_tinymce($init) {
  $init['extended_valid_elements'] .= ',pre[*],script[*],code[*],iframe[*]';
  return $init;}
add_filter('tiny_mce_before_init', 'init_tinymce');
```

**Code highlighting**:
The function `highlight_load()` includes highlight.js in the footer of your WordPress theme.

**Equations**:
R Markdown files with latex like math equations `$x=3$` are transformed to the MathJax syntax `\(x=3\)`. The function `mathjax_cdn()` add the rendering of equations to your WordPress site by adding the MathJax

Javascript library. MathJax is a JavaScript based rendering environment for mathematics. The function `mathjax_config()` enables the rendering of the LaTeX equation markup.

**Side note:** If you want to keep the LaTeX markup, the following global settings exclude the MathJax conversion in the `markdown::markdownToHTML` function used inside `knit2wp`. JetPack plugin provides support to automatically render the LaTeX math expressions.

```
# Set RMarkddown options (knit2wp uses markdown::markdownToHTML)
mdOpt <- markdownHTMLOptions(default = T)
options(markdown.HTML.options = mdOpt[mdOpt != "mathjax"])
mdExt <- markdownExtensions()
options(markdown.extensions = mdExt[!mdExt %in% c("latex_math","superscript")])
```

**WordPress editor issues**
The function `init_tinymce()` tries to circumvent the removal of code when changing between preview and raw text in the WordPress editor.

**Math equation rendering test:** Inline formula with $a = 3$ and $b = 5$

$$x = a + b$$

## Further resources

Useful resources used in this post

- RWordPress
- Yihuis': Knitr WordPress tests),Upload images
- Knitr with Flickr and WordPress
- Knitr and Imgur
- Blogging from R