

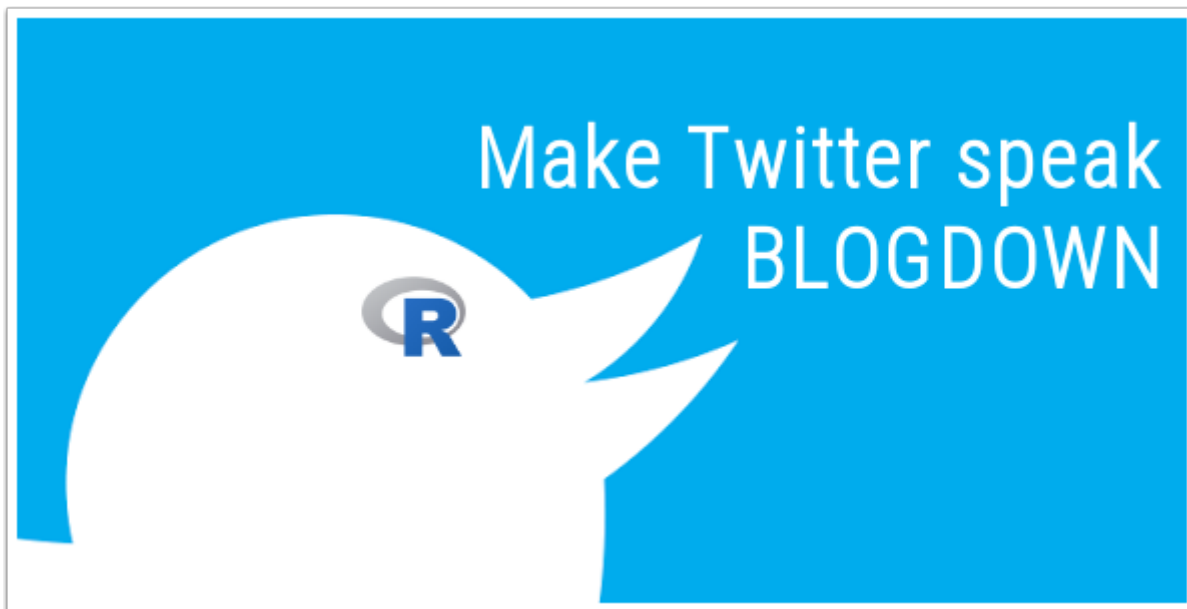
Socialize your blogdown

9 min read

2017/10/23

TL;DR

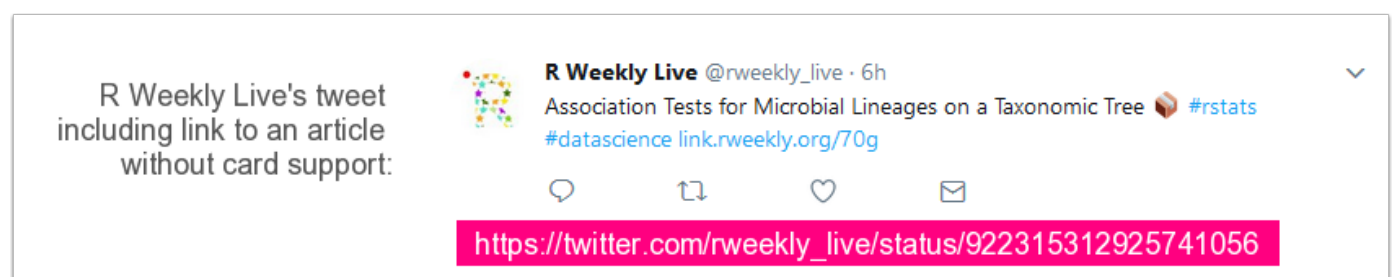
In this post, we show how to modify the default bookdown template so that rich cards are generated when links are shared on Twitter. We also try to learn a bit of Blogdown/Hugo templating along the way.



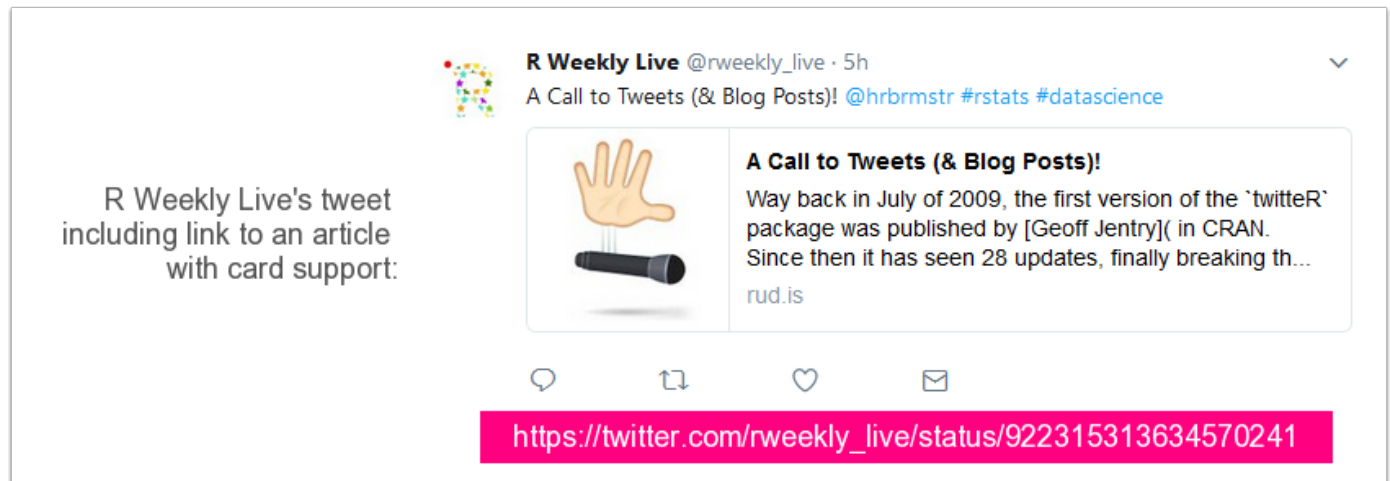
Tags and Twitter Tags

Social <meta> tags are HTML tags at the top of your pages that social network and search engine can parse to extract content and generate rich previews.

When the R Weekly twitter account share a link pointing to a site without tags for Twitter cards, it looks like a plain url.

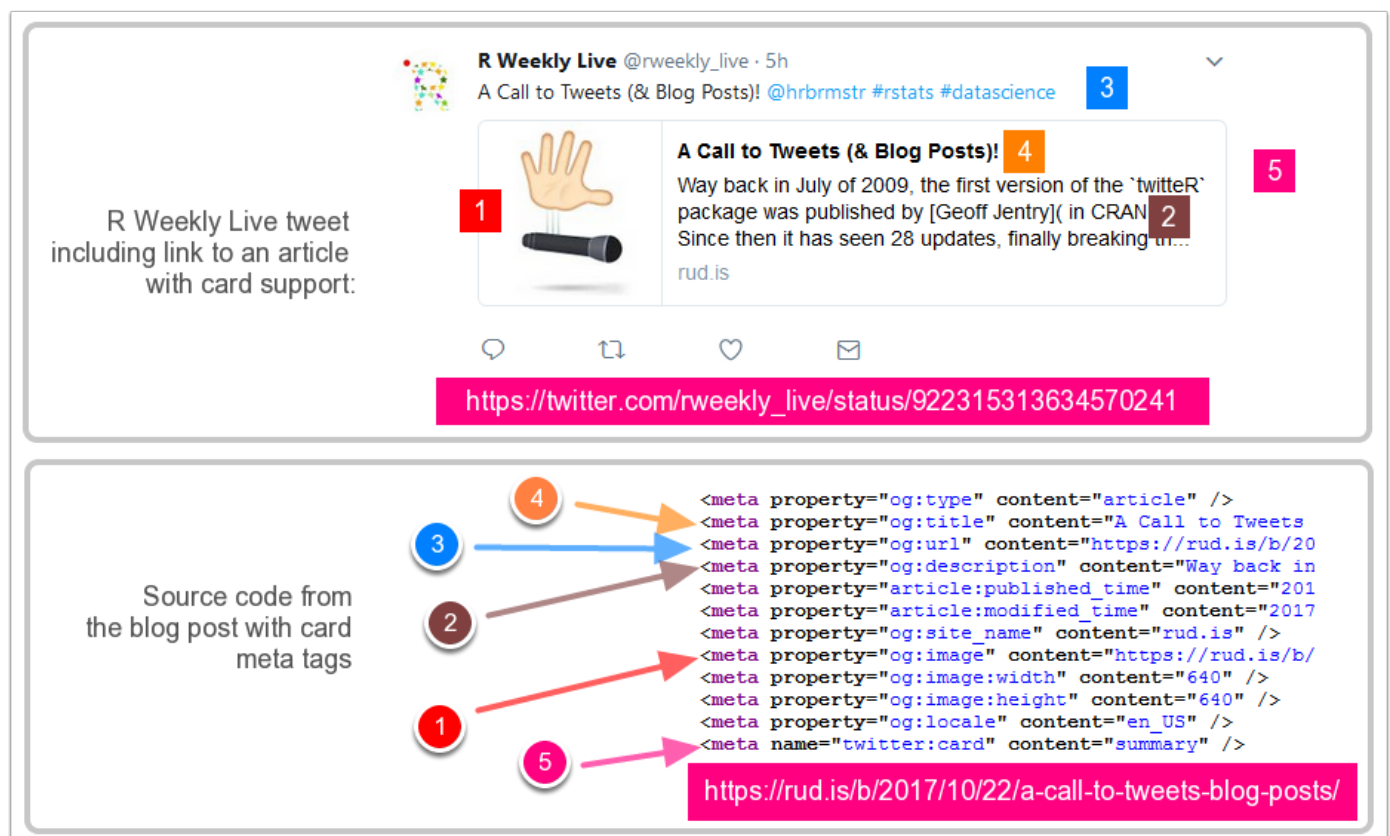


However, when they share a link pointing to a site with compatible meta tags, like Bob Rudis's blog, it doesn't look like a plain url, but rather like this:



The two big families of social graph meta tags are Twitter's `name=twitter:<tag-name>` and Facebook Open Graph's `property=og:<tag-name>`. Luckily, when Twitter doesn't find a tag, it can use the corresponding Open Graph tag in some cases, so we don't need to define everything twice (see "Twitter Cards and Open Graph" here).

If we look at the tags used in the example above, we see that a mix of `og:` and `twitter:` tags were used.



1. `og:image`: link to the image

2. og:description: short description text
3. og:url: actual reference url
4. og:title: title for the page
5. twitter:card: type of the card

Social graph tags and blogdown

Vanilla blogdown

This site is generated using the excellent [Yihui Xie's blogdown](#) package. It also use the default theme, a [fork of Jonathan Rutheiser's lithium-theme](#) for [Hugo](#).

The theme doesn't fully implement the <meta> tags necessary for creating cards on twitter. To see what tags are currently generated, we can check the template file head.html in themes/hugo-lithium-theme/layouts/partials/.

```
...
{{ if eq .URL "/" }}
<title>{{ .Site.Title }}</title>
<meta property="og:title" content="{{ .Site.Title }}">
<meta property="og:type" content="website">
{{ else }}
<title>{{ .Title }}{{ with .Params.subtitle }} - {{ . }} {{ end }} - {{
<meta property="og:title" content="{{ .Title }} - {{ .Site.Title }}">
{{ end }}

{{ if eq .URL "/" }}
<meta property="description" content="{{ .Site.Params.description }}">
{{ else }}
  {{ if .Description }}
    <meta property="description" content="{{ .Description }}">
  {{ end }}
{{ end }}
...
```

In addition to normal HTML, the template uses Hugo templating syntax elements to get dynamic html output (everything between double curly brackets):

1. Hugo variables to gather data about the site and the post, or to create your own reusable variables. There are different types of variables:
 - variables starting with `.` are defined in the top metadata of the page
 - variables starting with `.Site` are defined in the `config.toml` file
 - variables starting with `$` are defined in the template

I don't know enough about the details but it looks like the most common variable like `Title` and `Description` have "shortcuts" (i.e you access them simply with `.Title/.Site.Title` and `.Description`), whereas other variables need `.Params`.

```
<!-- insert the title of the site between a <title> tag -->
<title>{{ .Site.Title }}</title>
```

```
<!-- insert the title of the post between a <title> tag -->
<title>{{ .Title }}</title>
```

```
<!-- insert the title stored in variable mytitle -->
{{ $defaultTitle := "This is a title" }}
<title>{{ $defaultTitle }}</title>
```

2. if/else conditional to add different tags depending on a condition.

```
<!-- if the relative url of the page is "/", insert title "coconut" -->
{{ if eq .URL "/" }}
  <title>coconut</title>
{{ end }}
```

```
<!-- if there is a "description" metadata on the post, insert it -->
<!-- in a "description" meta tag, otherwise insert default text -->
{{ if .Description }}
  <meta property="description" content="{{ .Description }}">
{{ else }}
  <meta property="description" content="Default description">
{{ end }}
```

Let's deconstruct the Hugo templating syntax. I'll keep it brief so, if it doesn't make sense, read this introduction.

If the page is the homepage ({{ if eq .URL "/" }}):

```

{{ if eq .URL "/" }}
<title>{{ .Site.Title }}</title>
<meta property="og:title" content="{{ .Site.Title }}">
<meta property="og:type" content="website">
{{ else }}
...
{{ end }}

{{ if eq .URL "/" }}
<meta property="description" content="{{ .Site.Params.description }}">
{{ else }}
...
{{ end }}

```

1. Add a og:title tag with the name of site (.Site.Title refers to the title key in config.toml)
2. Add a og:type tag of type “website”.
3. Add a description tag with the description of the site (.Site.Params.description refers to the description key from the params section defined in config.toml)

If the page is not the homepage:

```

{{ if eq .URL "/" }}
...
{{ else }}
<title>{{ .Title }}{{ with .Params.subtitle }} - {{ . }} {{ end }} - {{
<meta property="og:title" content="{{ .Title }} - {{ .Site.Title }}">
{{ end }}

{{ if eq .URL "/" }}
...
{{ else }}
    {{ if .Description }}
        <meta property="description" content="{{ .Description }}">
    
```

```
{{ end }}
{{ end }}
```

1. Add a `og:title` tag with the name of the post (`.Title` refers to the `title` key at the top of the post) and the name of the site (`.Site.Title` refers to the `title` key in `config.toml`)
2. Only if a `description` key is defined on the post, add it as `description` key.

Adding tags

We do have `og:title`. We want to add:

- `og:description`: support for description, the current description tag isn't pulled as `og:description`.
- `twitter:creator` and `twitter:site`: twitter handles of the site/author
- `twitter:card` and `og:image`: the type of twitter card and image address

Post description with `og:description`

In the current template, a description tag is already added. Note that if there is no description for the post, no description tag is added.

We want (1) to add logic for `og:description` tag and (2) to add a default description for when no description was written for the post.

Add logic for `og:description`

Rather than using the `{{ if eq .URL "/" }}` twice we will wrap everything in the first `if` statement.

```
{{ if eq .URL "/" }}
<title>{{ .Site.Title }}</title>
<meta property="og:title" content="{{ .Site.Title }}">
<meta property="og:type" content="website">
<meta property="description" content="{{ .Site.Params.description }}">
<meta property="og:description" content="{{ .Site.Params.description }}"
{{ else }}
<title>{{ .Title }}{{ with .Params.subtitle }} - {{ . }} {{ end }} - {{
<meta property="og:title" content="{{ .Title }} - {{ .Site.Title }}">
```

```
<meta property="og:type" content="article">
<meta property="description" content="{{ .Description }}">
<meta property="og:description" content="{{ .Description }}">
{{ end }}
```

Add default for **og:description**

What happen if the post has no description? We could write an if statement to check first if there is a description key (`{{ if .Description }}`), but we can also provide a default, which goes like `{{ variable-name | default default-value }}`. To avoid writing the default-value twice (for tags description and og:description), we can store it in a Go template variable.

Below we create the `$defaultDescription` variable, defined as a string "Article posted by , on ". Then we add it as default for the tags og:description and description.

```
{{ $defaultDescription := printf "Article posted by %s, on %s" .Params.
<meta property="description" content="{{ .Description | default $defaultDescription }}">
<meta property="og:description" content="{{ .Description | default $defaultDescription }}">
```

and in context:

```
{{ if eq .URL "/" }}
<title>{{ .Site.Title }}</title>
<meta property="og:title" content="{{ .Site.Title }}">
<meta property="og:type" content="website">
<meta property="description" content="{{ .Site.Params.description }}">
<meta property="og:description" content="{{ .Site.Params.description }}">
{{ else }}
<title>{{ .Title }}{{ with .Params.subtitle }} - {{ . }} {{ end }} - {{ .Site.Title }}</title>
<meta property="og:title" content="{{ .Title }} - {{ .Site.Title }}">
<meta property="og:type" content="article">
```

```
{{ $defaultDescription := printf "Article posted by %s, on %s" .Params.
<meta property="description" content="{{ .Description | default $defaultDescription }}">
```

```
<meta property="og:description" content="{{ .Description | default $def
{{ end }}
```

Twitter creator and site

We could make twitter creator and site dependent of posts metadata if multiple authors were writing on your site. In this case, I just added two keys twitterAuthor and twitterSite into the params section of config.toml and used this for all posts (i.e outside of the if/else homepage conditional).

in config.toml

[params]

```
description = "A datascience journal."
twitterAuthor = "@xvrmdm"
twitterSite = "@invalid_input"
```

and in the template:

```
<meta name="twitter:creator" content="{{ .Site.Params.twitterAuthor }}"
<meta name="twitter:site" content="{{ .Site.Params.twitterSite }}">
```

and in context:

```
{{ if eq .URL "/" }}
<title>{{ .Site.Title }}</title>
<meta property="og:title" content="{{ .Site.Title }}">
<meta property="og:type" content="website">
<meta property="description" content="{{ .Site.Params.description }}">
<meta property="og:description" content="{{ .Site.Params.description }}"
{{ else }}
<title>{{ .Title }}{{ with .Params.subtitle }} - {{ . }} {{ end }} - {{
<meta property="og:title" content="{{ .Title }} - {{ .Site.Title }}">
<meta property="og:type" content="article">
```

```
{{ $defaultDescription := printf "Article posted by %s, on %s" .Params.
<meta property="description" content="{{ .Description | default $default
<meta property="og:description" content="{{ .Description | default $def
```



```
{{ end }}
```

```
<meta name="twitter:creator" content="{{ .Site.Params.twitterAuthor }}"
<meta name="twitter:site" content="{{ .Site.Params.twitterSite }}">
```

Twitter card type with `twitter:card`

Three scenarios:

- If the page is the homepage, we want to have a simple summary card (small image) with the site logo as image.
- If the page is a post and has no image specified in its metadata, again a simple summary card with the site logo as image.
- If the page is a post and has an image specified in its metadata, then let's do a `summary_large_image` card (image should be on a size ratio 2x1).

In the default template, the site logo is located in `https://<site-url>/images` with a name defined in `config.toml` (see the `params.logo` section) and accessible via `.Site.params.logo.url` variable.

For images, I place them all in a dir called `img` in `static`. So I refer them in posts with `/img/myimagefile.jpg`. At the top of the post, I can add a metadata key named `twitterImg` and then refer to it in the template with `.Params.twitterImg`.

A post metadata could look like this:

```
title: "Create maps from SITG files with sf and ggplot2"
author: "Xavier Adam"
date: 2017-09-15
twitterImg: /img/map-ggplot-sf-social.png
description: "Learn to create maps with sf and ggplot2, starting from S
categories: ["R"]
tags: ["R", "RMarkdown", "map", "shape", "sf", "ggplot2", "rmapshaper"]
```

The template syntax would be something like:

```
{{ if eq .URL "/" }}
<meta name="twitter:card" content="summary">
```

```

<meta name="twitter:image" content="http://xvrmd.github.io/images/{{
{{ else }}
{{ if .Params.twitterImg }}
  <meta name="twitter:card" content="summary_large_image">
  <meta name="twitter:image" content="http://xvrmd.github.io/{{ .Para
{{ else }}
  <meta name="twitter:card" content="summary">
  <meta name="twitter:image" content="http://xvrmd.github.io/images/{{
{{ end }}

```

and the final template:

```

{{ if eq .URL "/" }}
<title>{{ .Site.Title }}</title>
<meta property="og:title" content="{{ .Site.Title }}">
<meta property="og:type" content="website">
<meta property="description" content="{{ .Site.Params.description }}">
<meta property="og:description" content="{{ .Site.Params.description }}">
<meta name="twitter:card" content="summary">
<meta name="twitter:image" content="http://xvrmd.github.io/images/{{ .S
{{ else }}
<title>{{ .Title }}{{ with .Params.subtitle }} - {{ . }} {{ end }} - {{
<meta property="og:title" content="{{ .Title }} - {{ .Site.Title }}">
<meta property="og:type" content="article">

{{ if .Params.twitterImg }}
  <meta name="twitter:card" content="summary_large_image">
  <meta name="twitter:image" content="http://xvrmd.github.io/{{ .Params
{{ else }}
  <meta name="twitter:card" content="summary">
  <meta name="twitter:image" content="http://xvrmd.github.io/images/{{
{{ end }}

{{ $defaultDescription := printf "Article posted by %s, on %s" .Params.
<meta property="description" content="{{ .Description | default $default
<meta property="og:description" content="{{ .Description | default $def
{{ end }}

```

```
<meta name="twitter:creator" content="{ .Params.twitterAuthor }">
<meta name="twitter:site" content="{ .Params.twitterSite }">
```


Using the card validator, you can try your new cards:

Card URL


Preview card

*.xvrdm.github.io is whitelisted for summary_large_image card

Card preview

 **Xavier** @xvrdm

The card for your website will look a little something like this!




Socialize your blogdown - Invalid Input
Article posted by Xavier Adam, on Monday, October 23nd, 2017
xvrdm.github.io

Card URL


Preview card

*.xvrdm.github.io is whitelisted for summary_large_image card

Card preview

 **Xavier** @xvrdm

The card for your website will look a little something like this!



Geneva Land Use
showing communes, buildings, forests and fields

Create maps from SITG files with sf and ggplot2 - Invalid...
Article posted by Xavier Adam, on Friday, September 15nd, 2017
xvrdm.github.io