**What is a vignette? Where does it live?**

In this section we shall go over basics of package vignettes.

**Vignette 101**

In the “R packages” book by Hadley Wickham and Jenny Bryan, the [vignettes chapter](https://r-pkgs.org/vignettes.html) starts with *“A vignette is a long-form guide to your package. Function documentation is great if you know the name of the function you need, but it’s useless otherwise."*[1](https://blog.r-hub.io/2020/06/03/vignettes/#fn:1) In [“Writing R Extensions”](https://cran.r-project.org/doc/manuals/r-release/R-exts.html#Writing-package-vignettes), vignettes are defined as *“documents in PDF or HTML format obtained from plain-text literate source files from which R knows how to extract R code and create output (in PDF/HTML or intermediate LaTeX)."*.

In practice, if your package contains one or several vignette(s), an user could

* find them using the vignette() or browseVignettes() function, for instance they could type vignette(package = "rhub") or browseVignettes(package = "rhub") to access the list of installed vignettes for the rhub package[2](https://blog.r-hub.io/2020/06/03/vignettes/#fn:2)

vignette(package = "rhub")

| **Item** | **Title** |
| --- | --- |
| rhub | get-started (source, html) |
| local-debugging | Local Linux checks with Docker (source, html) |

browseVignettes("rhub")

| **Vignette** | **Title** |
| --- | --- |
| rhub.html | get-started |
| local-debugging.html | Local Linux checks with Docker |

Note that if the user installs your package from GitHub using devtools, [they will need to explicitly ask for installing vignettes](https://community.rstudio.com/t/vignettes-suddenly-stopped-installing/18391/2).

* see them from the [CRAN page of the package](https://cran.r-project.org/web/packages/rhub/index.html), and its [pkgdown website](https://r-hub.github.io/rhub/) if there’s one.

As a package author you could be fine only knowing about [usethis::use\_vignette()](https://usethis.r-lib.org/reference/use_vignette.html) for creating a vignette, and that packages used in the vignette need to be listed in DESCRIPTION (under Suggests if they’re only used in the vignette[3](https://blog.r-hub.io/2020/06/03/vignettes/#fn:3)). Still, it’s useful to know about vignettes for debugging problems or finding workarounds for issues you might encounter.

**Infrastructure & dependencies for vignettes**

The building of package vignettes can either use the default Sweave vignette engine, or [a vignette engine provided by a CRAN package](https://cran.r-project.org/doc/manuals/r-release/R-exts.html#Non_002dSweave-vignettes) like [knitr by Yihui Xie](https://bookdown.org/yihui/rmarkdown-cookbook/package-vignette.html). [knitr::rmarkdown vignette engine](https://community.rstudio.com/t/question-about-usethis-vignette-template/32048) is the one recommended in the R packages book, and usethis. It allows writing vignettes in R Markdown.

[See the source of rhub main vignette](https://github.com/r-hub/rhub/blob/master/vignettes/rhub.Rmd). It has YAML metadata at the top, some non-executed code chunks, some executed code chunks. To allow for that vignette to be built, a [field in DESCRIPTION](https://github.com/r-hub/rhub/blob/6ae6f35e958f3beab1e2c8e6f704affa23c8ce29/DESCRIPTION#L47) mentions the vignette engine[4](https://blog.r-hub.io/2020/06/03/vignettes/#fn:4):

VignetteBuilder: knitr, rmarkdown

And these two packages are declared as dependencies under Suggests as well.

The creation of a boilerplate Rmd under a new vignettes folder, and the dependencies declaration in DESCRIPTION, are what usethis::use\_vignette() would handle for you. Then you can write as you would a standard R Markdown document, knitting for previewing it.

Other vignette builders include [R.rsp](https://cran.r-project.org/web/packages/R.rsp/index.html) that we’ll mention again later, [noweb](https://cran.r-project.org/web/packages/noweb/index.html) to use the [noweb literate programming tool](https://en.wikipedia.org/wiki/Noweb) (which actually looks a lot like sweave?), [rasciidocs](https://cran.r-project.org/web/packages/rasciidoc/index.html) that was recently archived at the time of writing. It is unlikely you’ll want to write your own vignette engine.

How many packages use a non-Sweave vignette? One way to assess that is to look for packages that have a VignetteBuilder field in DESCRIPTION with R-hub’s own [pkgsearch](http://r-hub.github.io/pkgsearch/).[5](https://blog.r-hub.io/2020/06/03/vignettes/#fn:5)

results <- pkgsearch::advanced\_search("\_exists\_" = "VignetteBuilder")

attr(results, "metadata")$total

[1] 4969

knitr <- pkgsearch::advanced\_search(VignetteBuilder = "knitr")

attr(knitr, "metadata")$total

[1] 4739

# for comparison

nrow(available.packages())

[1] 15694

Quite a lot, about 32% of CRAN pages use a non Sweave vignette engine and about 30% use knitr for at least one vignette[6](https://blog.r-hub.io/2020/06/03/vignettes/#fn:6) Other packages might have *Sweave* vignettes, and some CRAN packages don’t have vignettes, whereas having a vignette is compulsory for Bioconductor packages.

**Overview of vignettes states**

Following the [neat diagram of the R packages book](https://r-pkgs.org/package-structure-state.html),

* You write your vignette(s) in the vignettes/ folder. (See e.g. [rhub source](https://github.com/r-hub/rhub" \t "_blank)).
* During building vignettes are [built](https://github.com/wch/r-source/blob/1d4f7aa1dac427ea2213d1f7cd7b5c16e896af22/src/library/tools/R/build.R#L320) and then vignettes sources, outputs, and anything written in [install\_extras](https://cran.r-project.org/doc/manuals/r-release/R-exts.html#index-_002einstall_005fextras-file) (a friend of [.Rbuildignore and .Rinstignore](https://blog.r-hub.io/2020/05/20/rbuildignore/) except it shows what to *keep* not *discard*!) gets moved to inst/doc/. ( See e.g. [rhub contents on CRAN](https://github.com/cran/rhub" \t "_blank)).

If your vignette shows an external image not generated by the build process, you also need to include it in install\_extras,

* During installation the content of inst/doc/ get copied to doc/. (See e.g. rhub content in my local library:)

fs::dir\_tree(find.package("rhub"))

/home/maelle/R/x86\_64-pc-linux-gnu-library/3.6/rhub

├── DESCRIPTION

├── INDEX

├── LICENSE

├── Meta

│ ├── Rd.rds

│ ├── features.rds

│ ├── hsearch.rds

│ ├── links.rds

│ ├── nsInfo.rds

│ ├── package.rds

│ └── vignette.rds

├── NAMESPACE

├── NEWS.md

├── R

│ ├── rhub

│ ├── rhub.rdb

│ └── rhub.rdx

├── bin

│ ├── rhub-linux-docker.sh

│ └── rhub-linux.sh

├── doc

│ ├── index.html

│ ├── local-debugging.R

│ ├── local-debugging.Rmd

│ ├── local-debugging.html

│ ├── rhub.R

│ ├── rhub.Rmd

│ └── rhub.html

├── help

│ ├── AnIndex

│ ├── aliases.rds

│ ├── figures

│ │ └── logo.png

│ ├── paths.rds

│ ├── rhub.rdb

│ └── rhub.rdx

└── html

├── 00Index.html

└── R.css

**Your vignette for R CMD check**

So, sometimes R CMD check[7](https://blog.r-hub.io/2020/06/03/vignettes/#fn:7) will throw errors related to vignette building. How to deal with them?

There is [good troubleshooting advice in the R packages book](https://r-pkgs.org/vignettes.html#vignette-cran).

Vignette metadata is important. A non place-holder title in [VignetteIndexEntry](https://www.mail-archive.com/r-package-devel@r-project.org/msg02902.html) is compulsory! Vignettes with a place-holder title are even [called bad\_vignettes in R source](https://github.com/wch/r-source/blob/95864f9a791189d3332b501f7544253a946e776f/src/library/tools/R/check.R#L4277). 

Based on what we said in the previous subsection, R CMD build builds vignettes from vignettes/ whereas R CMD check checks they can be rebuilt from inst/doc/. So if there were data in vignettes/, given it’s not copied to inst/doc/… R CMD check will error!

It’s also useful to know that there are options related to vignette building and checking in R CMD build and R CMD check. Of course you don’t control these options for CRAN but you do control them when sending your packages to R-hub package builder, and when setting up continuous integration. See for instance [this great tip by John Blischak](https://community.rstudio.com/t/compute-intensive-vignettes-devtools-and-travis-ci/45865/6), *“checking the package while ignoring the vignettes can be done with the following steps:"*

R CMD build --no-build-vignettes --no-manual .

R CMD check --no-manual --ignore-vignettes --as-cran \*. tar.gz

For R-hub package builder,

* To tweak the build you need to build your package yourself (from the command line or with devtools::build()) and indicate the path to the tarball, as opposed to the package source, in your call to [rhub::check()](https://r-hub.github.io/rhub/reference/check.html)
* You can tweak the R CMD check by using the check\_args argument.

**Workaround workflows for vignettes**

In this section we’ll go over workarounds for some common vignette “problems”.

**How to include my pre-print / cheatsheet as a PDF vignette?**

Sometimes you might want to include a PDF as a vignette, without wanting to deal with missing LaTeX dependencies; or because the PDF is not knit from R (a cheatsheet); or the computations are too long. In that case there are two alternatives:

* Following the process described in [a blog post by Mark van der Loo, entitled *“Add a static pdf vignette to an R package”*](http://www.markvanderloo.eu/yaRb/2019/01/11/add-a-static-pdf-vignette-to-an-r-package/);
* Using the [R.rsp package](https://cran.r-project.org/web/packages/R.rsp/index.html) by Henrik Bengtsson.

As an example of R.rsp usage, the [treeBUGS package](https://cran.r-project.org/web/packages/TreeBUGS/) has [HTML vignettes](https://cran.r-project.org/web/packages/TreeBUGS/vignettes/TreeBUGS_1_intro.html), and a [PDF vignette corresponding to a pre-print](https://cran.r-project.org/web/packages/TreeBUGS/vignettes/Heck_2018_BRM.pdf). In its [DESCRIPTION](https://github.com/danheck/TreeBUGS/blob/9983dd0597717950557f3dc4ccaa7b118b24d864/DESCRIPTION#L35) it indicates R.rsp as one of the vignette engines.

VignetteBuilder:

knitr,

R.rsp

In the vignettes/ folder of its source one sees [a file called Heck\_2018\_BRM.pdf.asis](https://github.com/danheck/TreeBUGS/blob/master/vignettes/Heck_2018_BRM.pdf.asis)

%\VignetteIndexEntry{Heck, Arnold, & Arnold (2018): TreeBUGS paper (Behavior Research Methods)}

%\VignetteEngine{R.rsp::asis}

%\VignetteKeyword{PDF}

%\VignetteKeyword{HTML}

%\VignetteKeyword{vignette}

%\VignetteKeyword{package}

%\VignetteKeyword{TreeBUGS}

Slightly related is this [workaround by Iñaki Úcar for building a vignette with a different output format based on the pandoc version available.](https://www.mail-archive.com/r-package-devel@r-project.org/msg02921.html)

**How to include a compute-intensive / authentication-dependent vignette?**

A very similar problem can happen with HTML vignettes, when their computations are too long, or depend on a system dependency or authentication token absent from CRAN machines – hence R CMD check would fail for sure. So, what can you do?

* You could [pre-compute vignettes following the approach described by Jeroen Ooms in an rOpenSci tech note](https://ropensci.org/technotes/2019/12/08/precompute-vignettes/). The gist is to call actual Rmd vignette source something like .Rmd.orig and to knit them to .Rmd. The .Rmd fake vignette sources have already executed R code. Therefore it can be used in the R CMD build/check process without creating errors, it can be knit rapidly.
* You could indeed use purl & eval, as [global knitr options](https://community.rstudio.com/t/precompiling-vignette-with-devtools/1583/6). See for instance [this GitHub thread](https://github.com/hrecht/censusapi/issues/32). A chunk could be line [the one from googlesheet](https://github.com/jennybc/googlesheets/blob/master/vignettes/managing-auth-tokens.Rmd#L15-L23)

```{r, echo = FALSE}

NOT\_CRAN <- identical(tolower(Sys.getenv("NOT\_CRAN")), "true")

knitr::opts\_chunk$set(

collapse = TRUE,

comment = "#>",

purl = NOT\_CRAN,

eval = NOT\_CRAN

)

```

* You could skip having vignettes and make them “articles” instead, that are present in a pkgdown site but not on CRAN/Bioconductor. [googledrive setup](https://github.com/tidyverse/googledrive/tree/master/vignettes), and [these explanations around tokens](https://gargle.r-lib.org/articles/articles/managing-tokens-securely.html). Articles can be created by [usethis::use\_article()](https://usethis.r-lib.org/reference/use_vignette.html). Of course it means vignettes are not available for offline consumption. In the case of a package interacting with an online service users are quite stuck when offline anyway. 

**Hey what about testing? And reproducibility?**

In the two previous subsections we recommended pre-building stuff, which might make some people cringe, but we like this [quote by Henrik Bengtsson in R-package-devel](https://www.mail-archive.com/r-package-devel@r-project.org/msg00812.html).

Some may argue that your package is not fully tested this way, but that depends on how well your package tests/ are written. I tend to look at examples() and vignettes as demos, and tests/ as actually tests. All should of course pass R CMD check and run, but the tests/ are what really test the package.

He also makes the point,

For reproducibility, I would include the root/source vignette in the package as well, e.g. in inst/full-vignettes/ with instructions and/or a function on how to rebuild it.

**User-friendly vignettes**

In this section we’ll give some tips for making vignettes easier to navigate.

**Pretty vignettes**

You might want to tweak layout and aspect of your vignette a bit to make people even more likely to read them, maybe with [custom CSS](https://bookdown.org/yihui/rmarkdown/r-package-vignette.html)[8](https://blog.r-hub.io/2020/06/03/vignettes/#fn:8). Using a [disappointingly unspecific GitHub code search on R-hub mirror of CRAN](https://github.com/search?l=&o=desc&q=css+user%3Acran++extension%3ARmd+path%3Ainst%2Fdoc&s=indexed&type=Code) we found the example of [idiogramFISH](https://gitlab.com/ferroao/idiogramFISH/-/tree/master) that [defines](https://gitlab.com/ferroao/idiogramFISH/-/tree/master/vignettes/css) and [uses](https://gitlab.com/ferroao/idiogramFISH/-/blob/master/vignettes/AplotIdiogramsVig.Rmd#L33) custom stylesheets for its vignette, that makes the vignette look very modern [on its CRAN page](https://cran.r-project.org/web/packages/idiogramFISH/vignettes/AplotIdiogramsVig.html)! Note that it also uses some JavaScript for the table of content and “return to top” links, definitely not light-weight styling.

Now, an even better way to tweak your vignettes is to invest some time in creating a pkgdown website that will feature both manual pages, vignettes, changelogs, etc. It’s actually little work. It’s worth it reading how vignettes are built in [pkgdown docs](https://pkgdown.r-lib.org/reference/build_articles.html), in particular

* A vignette called packagename.Rmd will appear under “Get started” in the navbar;
* You can tweak the navbar.

Once you’ve created the website, do not forget to indicate its [URL in DESCRIPTION](https://blog.r-hub.io/2019/12/10/urls/). 

Some further thoughts around vignettes and pkgdown. Since vignettes look better and are more integrated with other docs in the pkgdown website than locally, should your local vignettes contain a link to the pkgdown version to be sure that users that look at an offline vignette but have an internet connection can get a better user experience? And regarding the offline experience, would it make sense to also generate a PDF version of HTML vignettes, maybe with paged.js[9](https://blog.r-hub.io/2020/06/03/vignettes/#fn:9)?

**Cross-references**

Vignettes and manual pages serve [different roles](https://twitter.com/JennyBryan/status/1048634586274529281) and complement each other.

In places other than the vignettes you could tell the user to type vignette("vignette-name"). In pkgdown websites, using that function [will create a link the vignette page](https://www.mail-archive.com/r-package-devel@r-project.org/msg03203.html).

To link a vignette from another vignette, the [R packages book mentions](https://r-pkgs.org/vignettes.html#organisation) *“Although it’s a slight hack, you can link various vignettes by taking advantage of how files are stored on disk: to link to vignette abc.Rmd, just make a link to abc.html."* Again, this is supported in pkgdown websites, where functions are furthermore automatically linked to their manual page.

If you have many vignettes, you might want to use the ultimate R Markdown machinery for having cross-references, [bookdown](https://github.com/rstudio/bookdown), i.e. writing a book instead of a pkgdown website! See [how drake website links to a “Full manual” in its navbar](https://docs.ropensci.org/drake/). This process is currently separate from your usual a vignettes/pkgdown workflow, but might [not always be](https://github.com/r-lib/pkgdown/issues/853).

**Repeat yourself**

Even better than cross-references, or complementary to them is the idea to repeat yourself. As a quick reminder from [our post about READMEs](https://blog.r-hub.io/2019/12/03/readmes/#tools-for-writing-and-re-using-content), and as [explained very well by Garrick Aden-Buie](https://www.garrickadenbuie.com/blog/dry-vignette-and-readme/), you can re-use Rmd fragments in your package README, vignettes and manual pages without actually needing to copy-paste content!

**Conclusion**

In this post we offered a quite detailed, but probably not exhaustive, guide around R package vignettes. We haven’t discussed [*content* of vignettes](https://r-pkgs.org/vignettes.html#vignette-advice), how to best assess their usefulness (surveys? traffic data in pkgdown websites?), or their use as a way to [encapsulate analyses in a package structure or “research compendium”](https://annakrystalli.me/rrresearchACCE20/creating-a-research-compendium-with-rrtools.html).