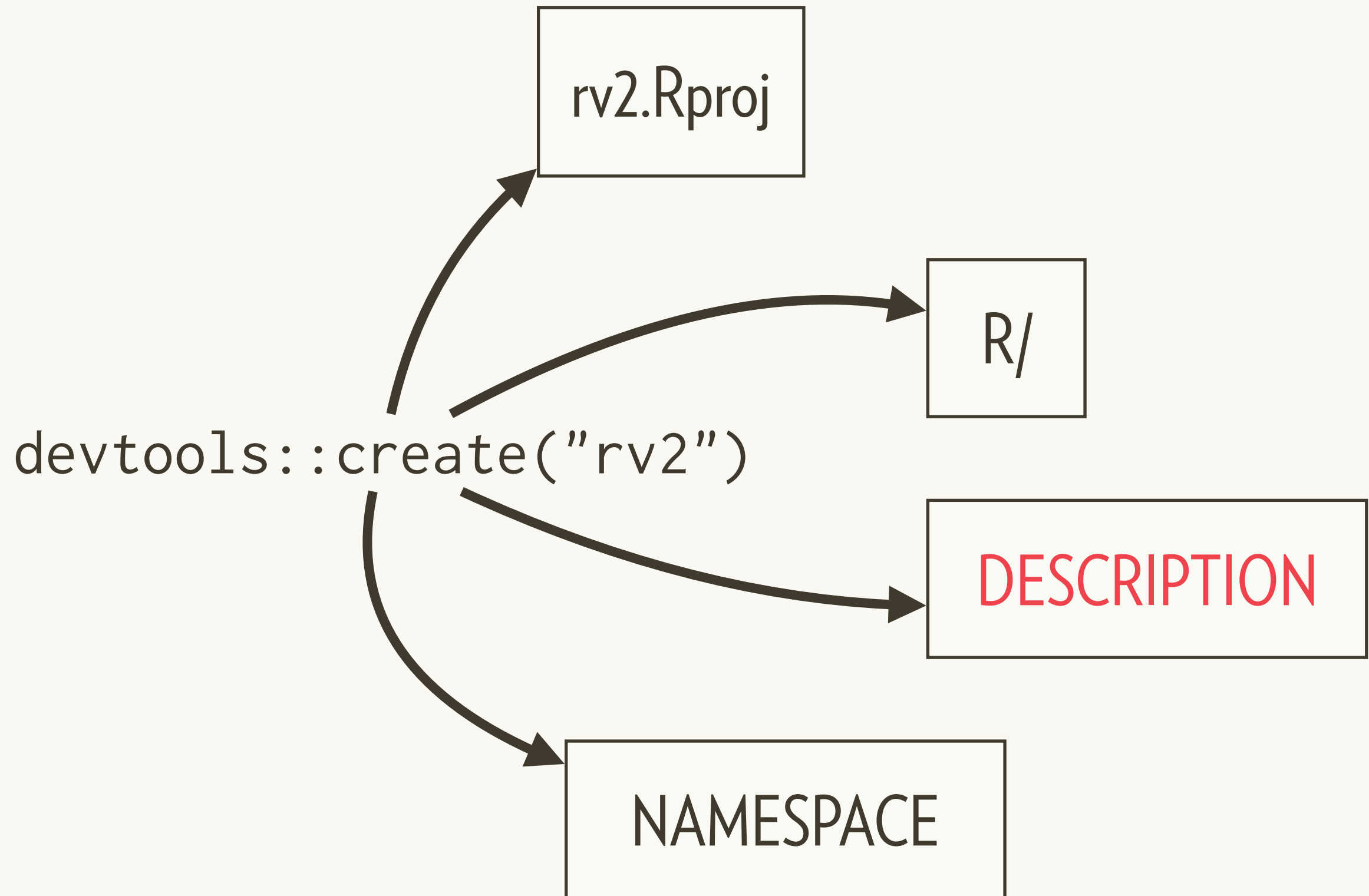


DESCRIPTION

September 2016

Hadley Wickham
[@hadleywickham](#)
Chief Scientist, RStudio

What happens we run create?



The default uses a template to get you started

Package: rv2

Version: 0.0.0.9000

Title: What The Package Does (one line, title case)

Description: What the package does (one paragraph)

Authors@R:

```
person("First", "Last", , "first.last@example.com", c("aut", "cre"))
```

License: What license is it under?

LazyData: true

Depends: R (>= 3.1.0)

If keyboard shortcuts don't work in RStudio, probably because your package lacks a DESCRIPTION

License

There are three main open source licenses

MIT

Free for anyone to
do anything with

CC0

“public domain”, best
for data packages

GPL

Changes and
bundles must also
be GPL

These are gross simplifications!

DESCRIPTION:

License: file LICENSE

LICENSE:

Proprietary: do not distribute outside of Widgets
Incorporated.

Requirements

I need you!

```
version (3.0.2) # optional version spec
```

Imports:

```
  stringr,  
  lubridate
```

Suggests:

I like having you around

There are three types of dependency

Imports: package is required to for your package to work. Installed automatically.

Suggests: nice to have.

Not installed automatically.

Depends: Basically deprecated; don't use.

(Correct uses exist, but beyond the scope of this class)

Use `::` to access functions in imported packages

```
# In DESCRIPTION
```

```
Imports: foo
```

```
# In bar.R
```

```
new_function <- function(x, y, z) {  
  foo::bar(x, y) + z  
}
```

Must check for presence of suggested packages

```
# In DESCRIPTION
```

```
Suggests: foo
```

```
# In bar.R
```

```
new_function <- function(x, y, z) {  
  if (!requireNamespace("foo", quietly = TRUE)) {  
    stop("Need foo! Use install.packages('foo').")  
  }  
  foo::bar(x, y) + z  
}
```

Never do this

```
new_function <- function(x, y, z) {  
  if (!requireNamespace("foo", quietly = TRUE)) {  
    install.packages("foo")  
  }  
  foo::bar(x, y) + z  
}
```

This is a side effect!

Reasons to use suggests instead of imports

Needed for development
(e.g. `testthat`)

Package has heavy dependencies
and only needed in special
situations (see `ggplot2`)

Used in vignettes
or examples

Reasons to use depends instead of imports

This page has been intentionally left blank

```
# use_package() will modify the DESCRIPTION  
# and remind you how to use the function.  
devtools::use_package("ggplot2")  
devtools::use_package("ggplot2", "suggests")
```

```
# load_all() will fail if imported packages are  
# not installed. Up to you to ensure suggested  
# packages have own checks.
```

```
# Installs all dependencies (including suggested)  
devtools::install_deps(deps = TRUE)
```



library(xyz)
require(xyz)

Error

Returns FALSE

Affects search
path

library()

require()

Doesn't affect
search path

loadNamespace()

requireNamespace()

This work is licensed under the
Creative Commons Attribution-Noncommercial 3.0
United States License.

To view a copy of this license, visit
<http://creativecommons.org/licenses/by-nc/3.0/us/>