

Inserting figures and evaluated examples

Georgi N. Boshnakov

Abstract

Package **Rdpack** provides Rd macros for inserting evaluated examples and programmatically created figures. These macros are convenience wrappers around the native capabilities provided by the Rd parser. The macros work in Rd files and roxygen2 comments.

This vignette is part of package Rdpack, version 0.8-3.

Keywords: examples, graphics, figures, Rd, R.

1. Evaluated examples

The standard Rd macro `\Sexpr` caters for a number of possibilities to evaluate R code and insert the results and the code in the documentation. The Rd macro `\printExample` provided by package **Rdpack** builds on it to print a snippet of R code and the results of its evaluation, similarly to console output but the code is not prefixed and the results are prefixed with comment symbols. For example

```
\printExample{2+2; a <- 2*3; a}
```

gives

```
2 + 2
##: 4
a <- 2 * 3
a
##: 6
```

The argument of `\printExample` must be on a single line with released versions of R¹.

2. Creating and including graphs

Figures can be inserted with the help of the standard Rd markup command `\figure`. The Rd macro `\insertFig` provided by package **Rdpack** takes a snippet of R code, evaluates it and inserts the plot produced by it (using `\figure`). `\insertFig` takes three arguments: a filename, the package name and the code to evaluate to produce the figure. For example,

```
\insertFig{cars.png}{mypackage}{x <- cars$speed; y <- cars$dist; plot(x,y)}
```

will evaluate the code, save the graph in file "man/figures/cars.png" subdirectory of package "my-package", and include the figure using `\figure`. Subdirectory "figures" is created if it doesn't exist. Currently the graphs are saved in "png" format only. The code should be on a single line for the reasons explained in the discussion of `\printExample`.

¹This limitation has been lifted in R-devel starting from June 2018 but obviously this is currently of little help for production packages.

The sister macro `\makeFig` creates the graph in exactly the same way as `\insertFig` but does not insert it. This can be done with a separate `\figure` command. This can be used if additional options are desired for different output formats, see the description of `\figure` in "Writing R extensions".

2.1. A technical note

The above description should just work. This note is for users who wonder about technicalities.

The R documentation can be built in many ways and as a result the directory `"man/figures/"` does not necessarily refer to the developers source package. Indeed, when a package is built, R works on a modified and cleaned-up temporary copy of the source directory, so the figures are created in that copy and then included in the package tarball. Similarly during the package check. On the other hand, R CMD `Rd2pdf` and some other tools and R functions work directly on the source tree of the package and they will create the figures there.

The net effect is that a package tarball always contains freshly generated up-to-date graphs. Developers who never generate the documentation by other means may not even have the directory `man/figures`.

Affiliation:

Georgi N. Boshnakov
School of Mathematics
The University of Manchester
Oxford Road, Manchester M13 9PL, UK
URL: <http://www.maths.manchester.ac.uk/~gb/>