

How to create an R package using R Studio

Creating a basic R package is quite simple. However, to keep it handy, you should use the roxygen2 package for handling the documentation (which is why the headers in the files below start with "#"). Here's a short overview on how to start:

1. Create a new empty package using File > New Project > New Directory > R Package and fill in the details
2. In the standard upper left window region, go to the Build tab and then on More > Configure Built. Mark "Generate documentation with Roxygen" and press the Configure button afterwards to additionally mark the line called "NAMESPACE file".
3. Delete the file called NAMESPACE in the root directory of your freshly created project (this file will be created by roxygen later).
4. Adapt your main R package file (which is called after the package). In the end it should look like the template below.
5. Add at least one additional function to your package and include a header like the one in the template below (the "@@" in the e-mail are mandatory!)
6. When everything is done, go to the Build tab again and click on More > Document to generate the documentation files. Click on Clean and Rebuild in the same drop down menu afterwards.

For more information have a look at these how-tos on [creating R Packages](#), [writing documentation](#) and [R packages](#) in general.

Template for main R Package file

```
#' One line information
#'
#' A view lines describing the package in more
#' detail.
#'
#' @name Name of the package
#' @aliases Short name of the package
#' @docType package
#' @title Title of the package
#' @author name of the author(s)\cr
#' \cr
#' \emph{Maintainer:} Maintainer of the package
#' \email{maintainer@@some-email.de}
#'
#' @keywords package
#'
NULL
```

Template file for function headers

```
#' One line description
#'
```

```
#' @description
#' More detailed, multi-line
#' description.
#'
#' @param example1 some parameter of the function
#' @param example2 some other parameter of the function
#'
#' @return what the function does return
#'
#' @export nameOfTheFunction
#'
#' @examples
#' not run:
#' nameOfTheFunction(example1= "X", example2 = 5)
nameOfTheFunction <- function(...){...}
```

From:

<https://geotraining.de/> - Summer School: GeoTraining

Permanent link:

<https://geotraining.de/doku.php?id=en:resources:howtos:rpackage>

Last update: **2025/03/14 12:52**

