

RcppClassic: Deprecated Rcpp API

Dirk Eddelbuettel^a and Romain François^b

^a<http://dirk.eddelbuettel.com>; ^b<https://romain.rbind.io/>

This version was compiled on November 19, 2017

This vignette describes how to use the **RcppClassic** package. It has long been deprecated and superseded by the more powerful **Rcpp** package. All new R packages should use the **Rcpp** package instead.

R | packages | Rcpp | API | Deprecation

This document presents the **RcppClassic** package. This package has been factored out of **Rcpp** (Eddelbuettel *et al.*, 2017; Eddelbuettel, 2013; Eddelbuettel *et al.*, 2017; Eddelbuettel and Balamuta, 2017) and only contains code that is considered deprecated.

This package is released for the sole purpose of allowing package authors that are still using the classic **Rcpp** API to keep their package buildable. This document explains the changes needed in a package to use both the current and classic **Rcpp** APIs.

If you must use RcppClassic

A few changes are needed in packages that want to use the classic **Rcpp** API that is contained in **RcppClassic**. A sample package called **RcppClassicExample** is on CRAN and can be used as a template.

The DESCRIPTION file. The client package must declare that it depends on both **Rcpp** and **RcppClassic** in the Imports field and the LinkingTo field, so it must contain this:

```
Imports: RcppClassic, Rcpp
LinkingTo: RcppClassic, Rcpp
```

The NAMESPACE file. The client package should import both **Rcpp** and **RcppClassic**:

```
importFrom(Rcpp, evalCpp)
import(RcppClassic)
```

The Makevars file. The Makevars file must be updated so that user libraries for both **Rcpp** and **RcppClassic** are used. For **Rcpp** 0.11.0 we used

```
## This can be placed on one or two lines too
PKG_LIBS = \
  '${R_HOME}/bin/Rscript -e \
    "Rcpp:::LdFlags()" ' \
  '${R_HOME}/bin/Rscript -e \
    "RcppClassic:::LdFlags()" '
```

but starting with **Rcpp** version 0.11.0, the result of `Rcpp:::LdFlags()` is an empty string as **Rcpp** no longer provides a user-library. The above then reduces to

```
## This can be placed on one lines
PKG_LIBS = '${R_HOME}/bin/Rscript -e \
  "RcppClassic:::LdFlags()" '
```

which finds the required **RcppClassic** library.

The Makevars.win files. The `Makevars.win` must also be updated for the same reason, and in similar way. Use `${R_HOME}/bin/${R_ARCH_BIN}/Rscript.exe` instead of `${R_HOME}/bin/Rscript` to reflect both the bi-architecture nature of Windows builds and the differently names `Rscript` executable.

Include RcppClassic.h instead of Rcpp.h. Finally, all instances of this line :

```
#include <Rcpp.h>
```

need to be replaced by:

```
#include <RcppClassic.h>
```

You should not use RcppClassic

The previous section discusses the set of changes required to update a package so that it uses the classic API from **RcppClassic** since it has been removed from **Rcpp**.

We do, however, recommend that package authors stop using the classic API. It has been more than superseded by the current **Rcpp** API in terms of performance, design, maintainance, and ease of use.

References

- Eddelbuettel D (2013). *Seamless R and C++ Integration with Rcpp*. Use R! Springer, New York. ISBN 978-1-4614-6867-7.
- Eddelbuettel D, Balamuta JJ (2017). "Extending R with C++: A Brief Introduction to Rcpp." *PeerJ Preprints*, 5. URL [10.7287/peerj.preprints.3188v1](https://doi.org/10.7287/peerj.preprints.3188v1).
- Eddelbuettel D, François R, Allaire J, Ushey K, Kou Q, Russel N, Chambers J, Bates D (2017). *Rcpp: Seamless R and C++ Integration*. R package version 0.12.12, URL <http://CRAN.R-Project.org/package=Rcpp>.