

# Package ‘RcmdrPlugin.Export’

November 17, 2009

**Version** 0.3-0

**Date** 2009-11-17

**Title** Graphically export output to LaTeX or HTML

**Author** Liviu Andronic

**Maintainer** Liviu Andronic <landronimirc@gmail.com>

**Depends** Rcmdr (>= 1.3-12), xtable, Hmisc

**Description** The package helps users to graphically export Rcmdr output to LaTeX or HTML code, via `xtable()` or `Hmisc::latex()`. The plug-in was originally intended to facilitate exporting Rcmdr output to formats other than ASCII text and to provide R novices with an easy-to-use, easy-to-access reference on exporting R objects to formats suited for printed output. The package documentation contains several pointers on creating reports, either by using conventional word processors or LaTeX/LyX.

**License** GPL (>= 2)

**Repository** CRAN

**Date/Publication** 2009-11-17 08:48:44

## R topics documented:

RcmdrPlugin.Export-package . . . . .	2
formatdfExport . . . . .	3
latexExport . . . . .	4
Utilities . . . . .	5
xtableExport . . . . .	6
<b>Index</b>	<b>8</b>

---

RcmdrPlugin.Export-package

*Graphically export objects to LaTeX or HTML*

---

## Description

The package provides facilities to graphically export Rcmdr output to LaTeX or HTML code.

## Details

The package is designed to assist in exporting Rcmdr output (or other objects printed in Rcmdr) to formats other than ASCII text, and provide R novices with an easy-to-use, easy-to-access reference on exporting R objects to formats suitable for publication. At the moment, it is a graphical front-end to `xtable` and `latex` and can export various R objects to LaTeX or HTML.

In the case of LaTeX, the printed output can be used in two ways. Taking a static approach, one can copy the generated LaTeX code and paste directly into a LaTeX document, or within an ERT inset of a LyX [1] document. If the code is exported to a `.tex` file, one may insert it as an external file in LyX.

A more dynamic approach consists in creating `Sweave` documents, which combine LaTeX and R code. In this case, one may copy the generated R code and paste into code chunks in the `Sweave` document. Learning `Sweave` is considerably trivial compared to learning R, so don't let yourself intimidated! Although standard usage of `Sweave` requires knowledge of LaTeX, one can readily use LyX to create `Sweave` documents [2][3][4], thus avoiding the burden of (properly) learning LaTeX. For this particular usage, the plug-in can be seen as a reference guide for retrieving (or learning) the correct syntax to exporting objects.

Inserting the exported HTML code into conventional word-processing programmes (MS Word, OpenOffice Writer, AbiWord, etc.) can get trickier, but several approaches are possible. The HTML code can, of course, be readily copied and pasted into the source of an HTML document (for example, in Mozilla SeaMonkey Composer). This also works for an HTML document developed in OpenOffice Writer/Web, which subsequently can be *exported* to `.odt`.

It is possible export the R object to an `.html` file. Then, one could *insert* the *file* in an OpenOffice Writer or MS Word document, with a resulting table that could be easily formatted. When working with OpenOffice, MS Word, AbiWord or Gnumeric, one can also *open* the `.html` document, then copy and paste the table into the desired document. This approach combined with the `append` argument is particularly useful when exporting many objects.

For more information concerning exporting R output please see this page [5] by Frank Harrell.

Developing the plug-in became possible when John Fox, the author of Rcmdr, implemented the `popOutput()` function. The function keeps a stack of the last several objects printed in Rcmdr, and allows to retrieve them following a "First In, First Out" paradigm. Thus, the last printed object would also be the first one retrieved. This also means that objects available for export can be displayed only one at a time; one should reinitialise the dialogue for each subsequent object. Once the dialogue is initialised, pressing "Cancel" removes the object from the stack. The length of the stack can be controlled via Rcmdr options. Please note that Rcmdr.Export is limited to handling objects printed within Rcmdr; consequently, the plug-in will not retrieve objects printed in the underlying R console.

On a final note, please contact me should you spot a bug, want to request a feature or know how to work around the issues listed in LIMITATIONS. I will also be happy to receive feedback on the present documentation.

### Author(s)

Liviu Andronic <landronimirc@gmail.com>

### References

- [1] <http://www.lyx.org/> – A GUI for LaTeX, similar to conventional word-processing programmes
- [2] [http://cran.r-project.org/doc/Rnews/Rnews\\_2008-1.pdf](http://cran.r-project.org/doc/Rnews/Rnews_2008-1.pdf) – *Using Sweave with LyX*, Rnews article by Gregor Gorjanc
- [3] <http://gregor.gorjanc.googlepages.com/lyx-sweave> – Files needed to configure Sweave and LyX
- [4] <http://n2.nabble.com/Moving-graphics-from-R-into-LyX-best-format-tp477568p477571.html> – Hackish instructions to configure Sweave and LyX (mostly for Linux users)
- [5] <http://biostat.mc.vanderbilt.edu/SweaveConvert> – Converting Documents Produced by Sweave

### See Also

[Rcmdr](#), [xtable](#), [latex](#), [Sweave](#)

### Examples

```
## Not run:  
## start R  
library(RcmdrPlugin.Export) ## loads Rcmdr and the plug-in  
  
## End(Not run)
```

---

formatdfExport

*Format a data frame or a matrix for export*

---

### Description

Rcmdr interface to format matrices and data frames for export to LaTeX or HTML code.

### Usage

```
formatdfExport ()
```

### Details

The graphical interface to `format.df` allows to format any matrix or data.frame from the current workspace. The printed object can subsequently be retrieved for exporting to LaTeX or HTML.

**Value**

NULL

**Author(s)**

Liviu Andronic &lt;landronimirc@gmail.com&gt;

**See Also**[format.df](#)

---

`latexExport`*Export objects using Hmisc:latex*

---

**Description**

Rcmdr interface to export objects to LaTeX code.

**Usage**

```
latexExport()
```

**Details**

The graphical interface is limited to exporting objects supported by [latex](#). For some objects, however, the plug-in attempts to work around the limitations, either by converting the object to a supported class or by considering only the relevant element from its structure. Objects of certain classes are simply ignored and are not displayed in the dialogue.

If the retrieved object is on the ignored list, initialising the dialogue will fail "silently", display a warning message and attempt re-initialize. If the stack is empty, or contains only objects that cannot be exported, initializing the dialogue will fail "silently" and display an error message.

Depending on the class of the retrieved object, the plug-in will attempt to propose a vector of appropriate length for the `digits` argument. By default, the vector will set all columns to "2", a double-digit precision.

The `size printing` option should be any valid LaTeX size (see [1]).

The `file` option lets you specify an output file. Only the name of the file should be entered; the plug-in will add the extension automatically depending on the chosen export format. This behaviour is intended to prevent the user from making carelesss errors, such as exporting LaTeX code to an `.html` file.

Unlike in `xtable`, the `append printing` option defaults to `TRUE`. The conservative approach was chosen since by default `xtable` overwrites existing files; in case of a name clash, it could be easier to recover the files. The option is ignored when the `file` input field is empty, and doesn't affect the process of outputting to new files.

The plug-in supports previewing the [latex](#) code using a `.dvi` viewer. However, unlike in `latex`, by default the code will not be previewed; the user must check the relevant option. Also, the LaTeX code will be printed on screen prior to calling the viewer. To choose the programme used for previewing, for example, set options (`xdviCmd='evince'`) to use Evince (see [Startup](#)).

**Value**

NULL

**Author(s)**

Liviu Andronic <landronimirc@gmail.com>

**References**

[1] [http://en.wikibooks.org/wiki/LaTeX/Formatting#Font\\_Styles\\_and\\_size](http://en.wikibooks.org/wiki/LaTeX/Formatting#Font_Styles_and_size)

**See Also**

[latex](#)

---

Utilities

*Rcmdr.Export utility functions*

---

**Description**

Rcmdr.Export utility functions

**Usage**

```
listMatrixObjects(envir=.GlobalEnv, ...)
```

**Arguments**

<code>envir</code>	the environment to be searched; should generally be left at the default.
<code>...</code>	currently ignored.

**Details**

This section lists the functions used to support the graphical dialogues.

**Value**

NULL

---

`xtableExport`*Export objects using xtable*

---

## Description

Rcmdr interface to export objects to LaTeX or HTML code.

## Usage

```
xtableExport ()
```

## Details

The graphical interface is limited to exporting objects supported by `xtable`. For some objects, however, the plug-in attempts to work around the limitations, either by converting the object to a supported class or by considering only the relevant element from its structure. Objects of certain classes are simply ignored and are not displayed in the dialogue.

If the retrieved object is on the ignored list, initialising the dialogue will fail "silently", display a warning message and attempt re-initialize. If the stack is empty, or contains only objects that cannot be exported, initializing the dialogue will fail "silently" and display an error message.

Depending on the class of the retrieved object, the plug-in will attempt to propose a vector of appropriate length for the `digits` argument. By default, the vector will set all columns to "2", a double-digit precision. If the user does not modify the vector, the plug-in will generate the export command by ignoring the vector and resorting to the `xtable` defaults.

The `size` printing option should be any valid LaTeX size (see [1]).

The `file` option lets you specify an output file. Only the name of the file should be entered; the plug-in will add the extension automatically depending on the chosen export format. This behaviour is intended to prevent the user from making careless errors, such as exporting LaTeX code to an `.html` file.

Unlike in `xtable`, the `append` printing option defaults to `TRUE`. The conservative approach was chosen since by default `xtable` overwrites existing files; in case of a name clash, it could be easier to recover the files. The option is ignored when the `file` input field is empty, and doesn't affect the process of outputting to new files.

## Value

NULL

## Author(s)

Liviu Andronic <[landronimirc@gmail.com](mailto:landronimirc@gmail.com)>

## References

[1] [http://en.wikibooks.org/wiki/LaTeX/Formatting#Font\\_Styles\\_and\\_size](http://en.wikibooks.org/wiki/LaTeX/Formatting#Font_Styles_and_size)

*xtableExport*

7

**See Also**

[xtable](#)

# Index

\*Topic **misc**

Utilities, 5

\*Topic **package**

RcmdrPlugin.Export-package, 1

\*Topic **print**

formatdfExport, 3

latexExport, 4

xtableExport, 6

format.df, 3, 4

formatdfExport, 3

latex, 2-5

latexExport, 4

listMatrixObjects (*Utilities*), 5

Rcmdr, 3

RcmdrPlugin.Export-package, 1

Startup, 4

Sweave, 2, 3

Utilities, 5

xtable, 2, 3, 6, 7

xtableExport, 6