XLConnect

A platform-independent interface to Excel

The **XLConnect** Package

http://mirai-solutions.com

http://cran.r-project.org/web/packages/XLConnect/index.html

Author of the **XLConnect** package:
Martin Studer,
Mirai Solutions GmbH

Author of this Vignette:
Anna Maria Ksiezopolska,
Mirai Solutions GmbH

Version 0.1-7 October, 2011

Contents

1 Introduction

1.1 Scope and purpose of this document

This document is a user manual for the **XLConnect** R package. It is meant to be a top-level introduction and some of the more advanced features of **XLConnect** are not presented here. For such details, please refer to the Reference Manual.

1.2 Introduction to XLConnect

XLConnect is a package that allows for reading, writing and manipulating Microsoft Excel files from within R. It uses the Apache POI API[1] as the underlying interface.

XLConnect allows you to produce formatted Excel reports, including graphics, straight from within R. This enables automation of manual formatting and reporting processes. Reading and writing named ranges enables you to process complex inputs and outputs in an efficient way.

XLConnect's Main Features

- Reading & writing of **Excel worksheets** (via data.frames)
- Reading & writing of **named ranges** (via data.frames)
- Creating, removing, renaming and cloning worksheets
- Adding **graphics**
- Specifying **cellstyles**: data formats, borders, back- and foreground fill color, fill pattern, text wrapping
- Controlling **sheet visibility**
- Defining **column width** and **row height**
- **Merging/unmerging** cells
- Setting/getting **cell formulas**
- Defining **formula recalculation** behavior (when workbooks are opened)
- Setting **auto-filters**
- **Style actions**: controlling application of cell styles when writing (e.g. when using templates)
- Defining behavior when **error cells** are encountered

2 Installation

2.1 Software Requirements

XLConnect is completely cross-platform and as such runs under Windows, Unix/Linux and Mac (32- and 64-bit). It does **not** require an installation of Microsoft Excel, or any special drivers.

All you need to use **XLConnect** are the following:

- R, version 2.10.0 or higher
- Java Runtime Environment (JRE), version 5.0 or higher

[1] For more information on the Apache POI API, see the `http://poi.apache.org/` webpage.

2.2 Package Installation

The **XLConnect** package is part of the Comprehensive R Archive Network (CRAN). It can be easily installed by using the `install.packages()` command in your R session:

```
install.packages("XLConnect")
```

3 Usage and Examples

3.1 Getting Started

To load the package, use the `library()` or `require()` command in your R session:

```
library(XLConnect)
```

Now, you are ready to use **XLConnect**!

The package includes a User Manual (this document), which you can view by entering the following command:

```
vignette("XLConnect")
```

The Reference Manual, containing help pages for each function within the package, can be opened by using the `help()` command.

```
help(XLConnect)
```

3.2 Basic XLConnect functions

3.2.1 Loading/creating an Excel workbook

The `loadWorkbook()` function loads a Microsoft Excel workbook, so that it can then be further manipulated. Setting the `create` argument to `TRUE` will ensure the file will be created, if it does not exist yet.
Both .xls and .xlsx file formats can be used.

Load an Excel workbook (create if not existing)

```
loadWorkbook(filename, create = TRUE)
```

3.2.2 Creating a sheet within an Excel file

The `createSheet()` function creates a sheet of a chosen `name` into the workbook specified as the `object` argument.

Create a worksheet of a chosen name within a workbook

```
createSheet(object, name)
```

3.2.3 Creating names

The `createName()` function creates a name of a chosen `name` for a specified `formula` into a workbook. The `overwrite` argument lets you define the behaviour if the name already exists. If set to `TRUE`, the existing name will be removed first, before creating a new one. If set to `FALSE` (default setting), an exception will be thrown.

Create a name for a specified formula within a workbook

```
createName(object, name, formula, overwrite)
```

3.2.4 Writing named regions

The `writeNamedRegion()` method writes a named range into a workbook. The `data` argument is assumed to be a `data.frame`. The `header` argument allows you to specify whether column names should be written. The arguments are vectorized, which allows for writing multiple named regions with one call. In such a case, `data` is assumed to be a list of `data.frames`.

Write a named range into a workbook

```
writeNamedRegion(object, data, name, header)
```

3.2.5 Saving an Excel file to disk

The `saveWorkbook()` function saves a workbook to the corresponding Excel file and actually writes the workbook object to disk.

Save a workbook to a chosen Excel file

```
saveWorkbook(object)
```

3.2.6 Example using basic functions

Let's see how the basic functions introduced in this section can be used to create and save an Excel file. We will use the in-built **mtcars** dataset for this simple example.

The code below first loads the "XLConnect.xlsx" workbook, using `loadWorkbook()`. If the workbook does not exist yet, the function creates it.

Then, via `createSheet()`, a sheet named "mtcars" is created witin the workbook.

We then use `createName()` to create the ”name.mtcars” name in the workbook and specify its location in the mtcars sheet, with the C5 cell as the top-left corner.

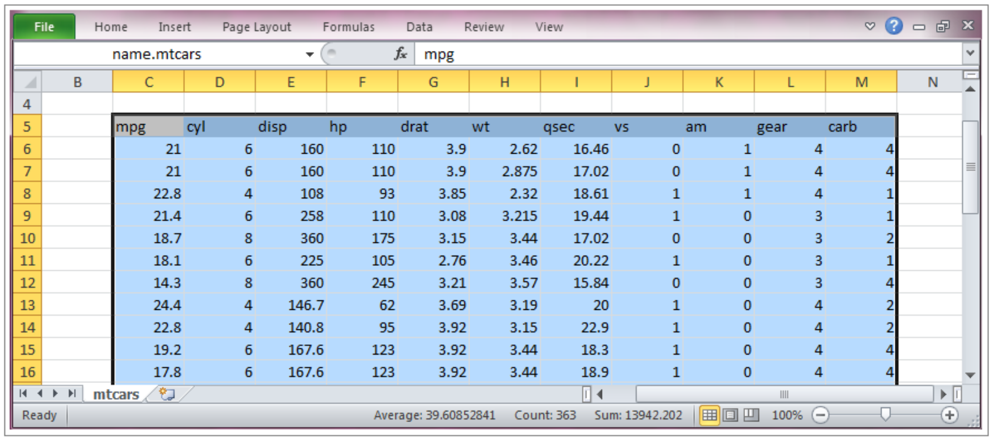
We then call `writeNamedRegion()` to write a named range to the workbook, using the mtcars dataset and the name.mtcars name.

At the end, we use `saveWorkbook()` to save the XLConnect.xlsx file.

```
> require(XLConnect)
> wb = loadWorkbook("XLConnect.xlsx", create = TRUE)
> createSheet(wb, name = "mtcars")
> createName(wb, name = "name.mtcars", formula = "mtcars!$C$5",
+     overwrite = TRUE)
> writeNamedRegion(wb, mtcars, name = "name.mtcars")
> saveWorkbook(wb)
```

We have now got the mtcars data written to an Excel file! The figure below illustrates the result.

Figure 1: name.mtcars named region written into the XLConnect.xlsx file



name.mtcars | mpg

	B	C	D	E	F	G	H	I	J	K	L	M	N
4													
5		mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb	
6		21	6	160	110	3.9	2.62	16.46	0	1	4	4	
7		21	6	160	110	3.9	2.875	17.02	0	1	4	4	
8		22.8	4	108	93	3.85	2.32	18.61	1	1	4	1	
9		21.4	6	258	110	3.08	3.215	19.44	1	0	3	1	
10		18.7	8	360	175	3.15	3.44	17.02	0	0	3	2	
11		18.1	6	225	105	2.76	3.46	20.22	1	0	3	1	
12		14.3	8	360	245	3.21	3.57	15.84	0	0	3	4	
13		24.4	4	146.7	62	3.69	3.19	20	1	0	4	2	
14		22.8	4	140.8	95	3.92	3.15	22.9	1	0	4	2	
15		19.2	6	167.6	123	3.92	3.44	18.3	1	0	4	4	
16		17.8	6	167.6	123	3.92	3.44	18.9	1	0	4	4	

mtcars

Ready Average: 39.60852841 Count: 363 Sum: 13942.202 100%

Please note that only at the point when we call `saveWorkbook()`, the Excel file is written to disk. All the previous operations are performed in-memory, which has great performance advantages.

Thanks for giving XLConnect a try!

Please note, that this is the very first version of the Vignette for **XLConnect**. We are planning to regularly add more descriptions of features and examples of usage, so please look out for updates!