Package 'Ropj'

March 14, 2023

Type Package

Title Import Origin(R) Project Files

Version 0.3-4

Description Read the data from Origin(R) project files ('*.opj')
<https://www.originlab.com/doc/User-Guide/Origin-File-Types>.
No write support is planned.

License GPL (>= 3)

Depends R (>= 3.1)

Imports Rcpp (>= 0.12.9)

LinkingTo Rcpp

URL https://github.com/aitap/Ropj

BugReports https://github.com/aitap/Ropj/issues

NeedsCompilation yes

Author Miquel Garriga [aut, cph],
Stefan Gerlach [aut, cph],
Ion Vasilief [aut, cph],
Alex Kargovsky [aut, cph],
Knut Franke [ctb, cph],
Alexander Semke [ctb, cph],
Tilman Benkert [ctb, cph],
Kasper Peeters [ctb, cph],
Russell Standish [ctb, cph],
Ivan Krylov [cre, cph]

Maintainer Ivan Krylov <krylov.r00t@gmail.com>

Repository CRAN

Date/Publication 2023-03-14 12:00:02 UTC

R topics documented:

`Ropj-package` *Import Origin(R) Project Files*

Description

Read the data from Origin(R) project files ('*.opj') <https://www.originlab.com/doc/User-Guide/Origin-File-Types>. No write support is planned.

Details

This package exports a single function, `read.opj`, that tries to read an OPJ file and return a list of objects that it consists of.

No write support seems to be planned in the foreseeable future.

References

`https://sourceforge.net/projects/liborigin/`: the `liborigin` implementation used by this package. Seems to be maintained as of 2021.

Examples

```
## Not run: x <- read.opj('data.opj')
```

`read.opj` *Parse Origin(R) project file into a list of objects*

Description

This function parses an OPJ file into a list of objects it consists of. Items understood by `read.opj` include spreadsheets, matrices, and notes.

Usage

```
read.opj(file, encoding = 'latin1', tree = FALSE, ...)
```

Arguments

`file`	Path to the OPJ file to parse. Only file paths are supported, not R connections. Path is not expanded automatically.
`encoding`	Encoding of the strings inside the file being opened. This should correspond to the ANSI code page of Windows installation used to produce the file. The default of `'latin1'` is usually safe to use. See `iconvlist()` for the full list of encodings supported by your R installation.

`tree` Control the structure of the returned list. When `FALSE` (default), returned list is flat and its unique names correspond to the short names of the objects, while the `comment` attribute contains their long names and comments. When `TRUE`, the list itself becomes a recursive data structure containing the tree of the objects, making it possible to access objects by their paths in that tree.

`...` The rest of the arguments is passed to `as.data.frame` when converting spreadsheets from lists of columns, making it possible to set `stringsAsFactors` and other parameters as needed.

Value

A named `list` containing objects stored in the file.

- Spreadsheets are presented as `data.frame`s, with additional `attributes`:
  - `comment` contains the fields Long name, Units, and Comment, joined by `\r\n`, if present. Due to a possible bug in `liborigin`, these values are sometimes followed by `@` and some text that wasn't present in the original document. (In versions prior to v0.2-2 it was called `comments`, which should be still supported until v1.0.)
  - `commands` contains the formula that was used to create the values of the column (e.g. `col(A) * 2 + 1`).
- Multi-sheet spreadsheets are stored as named lists of `data.frame`s described above.
- Matrices are presented as `list`s of `matrix` objects containing numeric data. `dimnames` are also assigned. The list also has attributes:
  - `commands` contains the formula that was used to compute the values in the matrix.
- Notes are stored as plain strings.

When `tree = FALSE`, the list is flat, its names are short names of the objects, and the `comment` attribute of the list contains the long names of the objects stored in the file.

When `tree = TRUE`, the list names are long names (if present; short otherwise) and the list itself represents the folder structure of the project.

Note

While Origin(R) and its scripting language seem to rely on the *short* names of the objects being unique across a project, neither *long* names nor *folder* names are guaranteed to avoid collisions. Tree-like lists returned by `read.opj(..., tree = TRUE)` might be easier to navigate interactively but present problems if someone gives multiple folders or objects the same long name.

Examples

```
x <- read.opj(system.file('test.opj', package = 'Ropj'))
head(x$Book2, 7)
comment(x$Book2)
attr(x$Book2, 'commands')
with(x$Book1, head(Sheet2 - Sheet1))
x$MBook1$MSheet1[1:4,1:4]
x$Note1
```

```
if ('CP1251' %in% iconvlist()) {
  # encoding names aren't guaranteed to be supported across all platforms
  x <- read.opj(system.file('test.opj', package = 'Ropj'), 'CP1251')
  print(x$cyrillic)
}

str(read.opj(system.file('tree.opj', package = 'Ropj'), tree = TRUE))
```

Index