

Package ‘RQDA’

September 20, 2009

Type Package

Title R-based Qualitative Data Analysis

Version 0.1-8

Date 2009-05-01

Author HUANG Ronggui

Maintainer HUANG Ronggui <ronggui.huang@gmail.com>

Depends R (>= 2.5.0), DBI, RSQLite, gWidgets (>= 0.0-31), gWidgetsRGtk2 (>= 0.0-36),igraph

Imports RGtk2 (>= 2.12.8), methods

Suggests tm, tcltk

Description R package for Qualitative Data Analysis. Current version only supports plain text.

License BSD

LazyLoad yes

URL <http://rqda.r-forge.r-project.org/>

Repository CRAN

Date/Publication 2009-09-20 20:22:39

R topics documented:

RQDA-package	2
CrossCode	3
Deletion	4
ExportCoding	5
GetAttr	6
GetCaseId	7
GetCodingTable	8
GetFileId	9

GetFileIdSets	10
gselect.list	11
Query	12
relation	13
retrieval	14
RQDA2tm	15
RQDAQuery	15
RQDATables	16
SearchFiles	19
SummaryCoding	21
write.FileList	22

Index	23
--------------	-----------

RQDA-package	<i>R-based Qualitative Data Analysis package</i>
--------------	--

Description

Qualitative Data Analysis based on R language. Current version only supports plain text.

Details

Package: RQDA
 Type: Package
 Version: 0.1-8
 Date: 2009-09-12
 Depends: R (>= 2.5.0), DBI, RSQLite, gWidgets (>= 0.0-31), gWidgetsRGtk2 (>= 0.0-36),igraph
 Imports: RGtk2 (>= 2.12.8), methods
 Suggests: tm, tcltk
 License: New style BSD License
 LazyLoad: yes
 URL: <http://rqda.r-forge.r-project.org/>

The workhorse function for end-user is the RQDA(), you can use RQDA() to start the GUI after library(RQDA). Please Refer to the documentation section of the project homepage for the usage of RQDA.

Author(s)

Huang Ronggui
 Maintainer: Huang <ronggui.huang@gmail.com>

Examples

```
## Not run:
library(RQDA)
```

```
RQDA()
## End(Not run)
```

CrossCode	<i>Inter-codes relationship</i>
-----------	---------------------------------

Description

Return a matrix, give a summary of inter-codes relationship.

Usage

```
CrossCode(relation=c("overlap", "inclusion", "exact", "proximity"), codeList=NULL, data=
CrossTwo(cid1, cid2, data, relation=c("overlap", "inclusion", "exact", "proximity"), ...)
```

Arguments

<code>relation</code>	The relation between codes
<code>codeList</code>	A character vector, the codes list on which the inter-code relationship is based
<code>data</code>	Data frame return by <code>GetCodingTable</code> , may be subset of the full coding table
<code>print</code>	When TRUE, print the results automatically
<code>cid1</code>	Length-1 code id. It is numeric.
<code>cid2</code>	Length-1 code id. It is numeric.
<code>...</code>	... is not used yet.

Details

The inter-codes relationship calculation is based on the relationship between the associated codings of the codes.

Giving the code name list (a character list), `CrossCode` returns the inter-relationship of 2 or more than 2 codes. `CrossCode` make heavy use of for loops, so it may takes a while to get the result when the coding table is large.

`CrossTwo` returns the summary of inter-codes relationship of two codes based on the code id (each code id is a length-1 integer vector).

Value

For `CrossCode`, it is a matrix. The upper matrix contains the number of codings fitting the relation between the respective two codes. the lower matrix is all NA. rownames of the matrix is the name of the codes, and the colnames of the matrix is the corresponding id of codes.

For `CrossCodes`, it is a numeric vector.

See Also

[relation](#)

Examples

```
## Not run:
CrossCodes()
## End(Not run)
```

Deletion

Functions for dealing with the temporarily deleted data

Description

`list.deleted` shows the temporarily deleted data (deleted by delete button, which is only tagged with deletion mark in the *.rqda file). `pdelete` permanently deletes them. `CleanProject` cleans the *.rqda file (call `pdelete` with every possible value for the `type` argument). `undelete` removes the temporarily deletion mark to reuse the temporarily deleted data.

Usage

```
list.deleted(type=c("file", "code", "case", "codecategory", "filecategory"))
pdelete(type=c("file", "code", "case", "codecategory", "filecategory", "coding"), ask=FALSE)
CleanProject(ask=FALSE)
undelete(type=c("file", "code", "case", "codecategory", "filecategory"), ask=TRUE)
```

Arguments

<code>type</code>	Types of elements in the *.rqda file. "file" is the name of file (in the Files Tab). "code" is the name of codes (in the Codes Tab). "case" is the name of case (in the Case Tab). "codecategory" is name of code category (in the C-Cat Tab). "filecategory" is name of file category (in the F-Cat Tab). "coding" is the text segment associated with specific code.
<code>ask</code>	You can choose which ones to be deleted when is TRUE. Otherwise, it will delete all with temporarily deletion mark.

Details

By GUI, you can delete file, code, case, code category and file category. When click the delete button, the status of related elements (e.g. for file, the elements includ file, related coding, related case category and file category) are set from 1 to 0. In this sense, deletion from GUI is temporary. After that, you can use `list.deleted` to show which ones are tagged as deleted. By `pdelete`, you can permanently delete those tagged with temporarily deletion mark. By `undelete`, you can undo the temporary deletion, the status of related elements are set back to 1.

When `ask` is FALSE, it will apply to all the propriate elements of specific type. When it is TRUE, you can choose the elements of the specific type which the action (`pdelet` or `undelete`) applies to.

Value

For `list.deleted`, a data frame if there are some records with temporarily deletion mark for the specified type. For `pdelete`, `CleanProject` and `undelete`, no value is return. These functions are used for the side-effects.

Note

In order to make the temporarily deletion of code and the associated coding can be undeleted again, RQDA differentiates the temporarily deletion of codings (which are deleted by deleting a code) from that produced by unmark button in the Coding Tab: the former with status = 0 while the latter with status = -1.

Author(s)

Ronggui HUANG

ExportCoding *Export codings to a HTML file.*

Description

To export retrieved codings to a HTML file.

Usage

```
ExportCoding(file = "Exported Codings.html", Fid = NULL, order = c("fname", "ftime"
```

Arguments

file	Length-one character vector, specify the name of exported file.
Fid	Integer vector of file id. The retrieved codings are from this subset of files. When is NULL, it means all the files.
order	Specify the order of retrieved codings. see details for the meanings.
append	Logical, when TRUE the exported codings are appended to the existing file (if exists); otherwise, it overwrites the existing file.

Details

"fname" means order the codings by file names, "ftime" by file imported time, and "ctime" by time of coding.

Value

A html file.

Author(s)

HUANG Ronggui

`GetAttr`*Attributes*

Description

Get the attributes of case or file.

Usage

```
GetAttr(type = c("case", "file"), attrs = svalue(.rqda$.AttrNamesWidget))
```

```
ShowSubset(x, ...)
```

Arguments

<code>type</code>	Type of attributes.
<code>attrs</code>	character vector, subset of attributes to retrieve.
<code>x</code>	an object from <code>GetAttr</code>
<code>...</code>	Not used currently.

Details

You can add and modify the attributes of cases or files. `GetAttr` returns this attributes as a data frame.

Sometimes, you only want to show a subset of files or cases according to their attributes. You can do the subset operation of the result from `GetAttr` and pass it to `ShowSubset`.

Value

For `GetAttr`, when `type` is "case", it is a data frame with class of "CaseAttr"; when `type` is "file", it is a data frame with class of "FileAttr". For `ShowSubset`, no value is returned, the side-effect is to change the file list or case list in respective widget.

Note

All the variables in the data frame is of class "character", you need to convert to suitable class when conduct statistical analysis.

Author(s)

HUANG Ronggui

Examples

```
## Not run:
attr <- GetAttr("case")
ShowSubset(subset(attr,attribute1==1)) ## assuming there is a variable
named attribute1 in attr.
## End(Not run)
```

GetCaseId

Get the Case ID and Case Name.

Description

GetCaseId returns the case IDs which a file belongs to given the file IDs. GetCaseName returns the case Names given the case IDs.

Usage

```
GetCaseId(fid = GetFileId(), nFiles = FALSE)

GetCaseName(caseId = GetCaseId(nFiles = FALSE))
```

Arguments

fid	numeric vector, the file IDs.
nFiles	logical, return the number of files that belong to a case.
caseId	numeric vector, the case IDs.

Value

GetCaseId returns a data frame of two columns when nFiles is TRUE, and a numeric vector when FALSE.

GetCaseName returns a character vector or NULL if no cases are associated with the file IDs.

Author(s)

HUANG Ronggui

See Also

See Also [GetFileId](#)

Examples

```
## Not run:
GetCaseName(GetCaseId(GetFileId("filecategory")))
## End(Not run)
```

GetCodingTable *Get the information of codings*

Description

Get the information of codings.

Usage

```
GetCodingTable ()
```

Details

Codings are stored in the coding table of *.rqda file. The coding table contains necessary information, but not informative to end-users. For example, it has id of code list and file list, but not the name of them, which are stored in freecode table and source table respectively. `GetCodingTable` joins information from the three tables, and returns more informative data. See value section on the the returned components.

Value

A data frame:

<code>cid</code>	Code id
<code>fid</code>	File id
<code>codename</code>	Code name, in accordance with <code>cid</code>
<code>filename</code>	File name, in accordance with <code>fid</code>
<code>CodingLength</code>	The number of characters in the coding
<code>index1</code>	beginning index of a coding
<code>index2</code>	end index of a coding

Author(s)

HUANG Ronggui

GetFileId *Get the ids or names of files list*

Description

Get the ids or names of files list.

Usage

```
GetFileId(condition = c("unconditional", "case", "filecategory"), type =  
c("all", "coded", "uncoded", "selected"))
```

```
GetFileName(fid = GetFileId())
```

Arguments

<code>condition</code>	Any one of "unconditional", "case" or "filecategory".
<code>type</code>	Any one of "all", "coded" or "uncoded", "selected".
<code>fid</code>	integer vector, the id of files.

Details

The imported files are stored in a data base table (called source) in the *.rqda file. Every file in the source table has a unique id. Besides, every file can be assigned to a case or file category.

Given that files meet the `condition`, the `type` argument "all" means all files, "coded" means the coded files, "uncoded" means the uncoded files and "selected" means the selected files; in "files" widget, "files of case" widget and "files of category" widget respectively.

`GetFileId` returns the id of files which fit the combined criterion of `condition` and `type`.

Value

Normally, it is a numeric vector of file id. If `condition` is "case" or "filecategory" but no case or file category is selected, it returns NULL.

Author(s)

HUANG Ronggui

See Also

[retrieval](#), [GetFileIdSets](#)

Examples

```
## Not run:
GetFileId() ## Id of all files
GetFileId("unconditional","coded") ## id of all coded files.
GetFileId("case","uncoded") ## id of uncoded files for the selected case.
GetFileId("filecategory","all") ## id of all files in the selected file category.
## End(Not run)
```

GetFileIdSets	<i>Get file id from sets.</i>
---------------	-------------------------------

Description

Get the file id from file-sets given the type of relation between sets. File-set is defined by the case or filecategory.

Usage

```
GetFileIdSets(set = c("case", "filecategory"), relation = c("union", "intersect"))
```

Arguments

set	type of set, either "case" or "filecategory".
relation	relation between sets. either "union" or "interset".

Details

File-set is deined by case or file category. files belonging to a case/filecategory are in a set. This function get file id from the selected sets. When multiple sets are selected, the relation between them can be define. When relation is union, file ids from either selected set are returned. When reltion is intersect, only file ids appear in all the selected sets are returned.

Value

A numeric vector or NULL if no file id is well-defined.

Author(s)

HUANG Ronggui

See Also

[retrieval](#), [GetFileId](#)

`gselect.list` *Select Items from a List*

Description

Select item(s) from a character vector.

Usage

```
gselect.list(list, multiple = TRUE, title = NULL, width = 200, height = 500, x=420, y
```

Arguments

<code>list</code>	character vector. A list of items.
<code>multiple</code>	logical: can more than one item be selected?
<code>title</code>	optional character string for window title.
<code>width</code>	integer. width of the widget.
<code>height</code>	integer. height of the widget.
<code>x</code>	
<code>y</code>	
<code>...</code>	Not used currently.

Details

GTK version of [select.list](#).

Value

A character vector of selected items with encoding of UTF-8. If no item was selected (or 'Cancel' was used), "" is returned.

Note

The license of this function is subject to interpretation of the first author.

Author(s)

John Verzani and Ronggui HUANG

See Also

[select.list](#)

Examples

```
## Not run:  
select.list(sort(.packages(all.available = TRUE)))  
## End(Not run)
```

Query

Retrieval of file names according to their codings.

Description

To retrieve file names according to their codings and the relationship of codings.

Usage

```
Query(or, and = NULL, not = NULL)
```

Arguments

or	code id.
and	code id.
not	code id.

Details

This function is experimental, and may be changed in future.

For arguments of 'or', 'and' and 'not', all of them have the same form of input. Examples are '(fid1 or fid2 or fid3)' or "(fid1, fid2, fid3)", where fid1, fid2 and fid3 are file IDs.

Value

A vector.

Author(s)

HUANG Ronggui

Examples

```
## Not run:
Query(or=(1,2)) ## files coded to code 1 or 2.

Query(or=(1), and=(2)) ## files coded to code 1 and 2.

Query(or=(1), not=(2,3)) ## files coded to code 1 but not either 2 or 2.
## End(Not run)
```

relation	<i>Relation between two codings</i>
----------	-------------------------------------

Description

To calculate the relation between two codings, given the coding indexes.

Usage

```
relation(index1, index2)
```

Arguments

index1	The first coding index, it is length-2 integer vector with the first element (index1[1]) less than the second element (index1[2]).
index2	The second coding index, it is length-2 integer vector with the first element (index2[1]) less than the second element (index2[2]).

Details

The relation between two codings can be any of inclusion, overlap, exact (special case of inclusion and overlap) and proximity (Neither overlap nor inclusion).

Value

A 6-element list:

Relation	Length-1 character, standing for the type of relation. It may be one of inclusion, overlap, exact or proximity.
OverlapIndex	Length-2 vector, the index of overlapping between two coding indexes. It is c(NA,NA) when relation is proximity.
UnionIndex	Length-2 vector, the index of union of the two coding indexes. It is c(NA,NA) when relation is proximity.
Distance	Distance of two coding indexes. It is NA when relation is not proximity.
WhichMin	Which argument (index1 or index2) has the minimum value. If both have the same minimum value, return NA.
WhichMax	Which argument (index1 or index2) has the maximum value. If both have the same maximum value, return NA.

Author(s)

HUANG Ronggui

Examples

```
## Not run:
relation(c(20,30),c(22,28)) # inclusion
relation(c(10,40),c(20,80)) # overlap
relation(c(10,20),c(30,50)) # proximity with distance of 10
relation(c(10,20),c(10,20)) # exact
relation(c(10,20),c(10,30)) # WhichMin is c(1,2)
## End(Not run)
```

retrieval

Retrieval of codings conditional on the file id.

Description

To retrieve the codings of a selected code from specific set of files.

Usage

```
retrieval(Fid = NULL, order = c("fname", "ftime", "ctime"),
          CodeNameWidget = .rqda$.codes_rqda)
```

Arguments

<code>Fid</code>	Numeric vector, the file id.
<code>order</code>	The method of sort of retrieved codings.
<code>CodeNameWidget</code>	The name of code list widget.

Details

This function retrieves the codings of a selected code from `CodeNameWidget`, given that all the codings are from a set of files which are determined by `Fid`.

Value

A `gtext` widget is open and all the codings are pushed into that widget.

Author(s)

HUANG Ronggui

See Also

[GetFileId](#)

`RQDA2tm`*RQDA to Corpus*

Description

Constructs a text document collection (corpus) from RQDA codings.

Usage

```
RQDA2tm(Code, language = "french")
```

Arguments

<code>Code</code>	character vector of length 1. name of a code in RQDA.
<code>language</code>	argument passed to Corpus

Details

To retrieve the codings of `Code`, and turn the codings into a corpus.

Value

An S4 object of class `Corpus`.

Author(s)

J-P Mueller and documented by HUANG Ronggui

See Also

[Corpus](#)

`RQDAQuery`*Execute a SQL statement on the open *.rqda file.*

Description

Submits and executes an arbitrary SQL statement on the open *.rqda file.

Usage

```
RQDAQuery(sql)
```

Arguments

<code>sql</code>	a character vector of length 1 with the SQL statement.
------------------	--

Details

It is a wrapped version of [dbSendQuery-methods](#), to make it more convenient to submit and execute a SQL statement.

Value

The same of [dbSendQuery-methods](#), possible NULL (for the side effects of sql on the *.rqda file) or a data.frame with the output (if any) of the query.

Author(s)

HUANG Ronggui

See Also

See Also as [dbSendQuery-methods](#)

Examples

```
## Not run:
RQDAQuery("select name from source where status=1")
## End(Not run)
```

RQDATables

Data Tables in rqda file

Description

The internal data table structures in rqda file, which is a SQLite data base.

Details

Table "attributes" contains information about the name list of attributes.

name:	name of attributes.
status:	.
date:	.
dateM:	.
owner:	.
memo:	.

Table "caseAttr" contains information about attributes of cases.

variable:	name of case attributes, corresponding to name in attributes table.
value:	.
caseID:	.
date:	.
dateM:	.

owner: .

Table "caselinkage" contains information about the relationship between case and files of case.

caseid: .
 fid: .
 selfirst: .
 selend: .
 status: .
 owner: .
 date: .
 memo: .

Table "cases" contains information about case list.

name: .
 memo: .
 owner: .
 date: .
 dateM: .
 id: .
 status: .

Table "codecat" contains information about upper-level of code list.

name: .
 cid: .
 catid: .
 owner: .
 date: .
 dateM: .
 memo: .
 status: .

Table "coding" contains information on codings.

cid :
 fid :
 seltext :
 selfirst :
 selend :
 status :
 owner :
 date :
 memo :

Table "fileAttr" contains information about attributes of files.

variable: .
 value: .
 fileID: .
 date: .
 dateM: .
 owner: .

Table "filecat" contains information on the file categorization.

name: name of the file category.
 fid: Not used.
 catid: id of file category.
 owner: creator of file-category.
 date:
 dateM:
 memo:
 status:

Table "freecode" contains information on the codes list.

name :
 memo :
 owner :
 date :
 dateM :
 id :
 status :

Table "journal" contains information about field work journal.

name: .
 journal: .
 date: .
 dateM: .
 owner: .
 status: .

Table "project" contains information about the project and *.rqda file.

encoding: .
 databaseversion: .
 date: .
 dateM: .
 memo: .
 BOM: .

Table "source" contains the content of files.

name: name of the file.
 id: id of the file.
 file: content of a file.
 memo: memo of the file.
 owner: creator the the file.
 date: the date of the file-import.
 dataM: not used now.
 status: 1 for standard status and 0 for temporarily deleted file.

The "treecode" table contains information on the codes categorization (relationship between codes and the codecat).

cid: .
 catid: .
 date: .
 dateM: .
 memo: .
 status: .

Table "treefile" contains information about file categorization (relation between source files and filecat).

fid: .
 catid: .
 date: .
 dateM: .
 memo: .
 status: .

Author(s)

HUANG Ronggui

SearchFiles

Search files

Description

Search files according to the pattern.

Usage

SearchFiles(pattern, content = FALSE, Fid = NULL, Widget = NULL, is.UTF8 = FALSE)

Arguments

pattern	The criterion of search, see examples section for examples.
content	When it is TRUE, the content of files fitting the pattern will be returned as well.
Fid	integer vector, the ids of subset of files to search.
Widget	Character, name of a gtable widget. If it is not NULL, the file names fitting the pattern will pushed to that gtable widget using <code>svalue</code> method. One useful value is ".fnames_rqda", so the file names will be pushed to the Files Tab of RQDA. Others are ".FileofCat" and ".FileofCase".
is.UTF8	If the coding of pattern is UTF-8. If you are not sure, always use FALSE.

Details

This function use select statment of sql to search files (from source database table). The pattern is the WHERE clause (without the keyword WHERE). For more information, please refer to the website of SQLite syntax. All data in *.rqda use UTF-8 encoding, so the encoding of pattern matters. It will be converted to UTF-8 if it is not (is.UTF8=FALSE).

Value

A data frame with variables:

id	The file id.
name	The file name.
file	The file content. Only return when content is TRUE.

Author(s)

HUANG Ronggui

References

http://www.sqlite.org/lang_expr.html

See Also

[gtable](#), [localeToCharset](#)

Examples

```
## Not run:
SearchFiles("file like '%keyword%'") ## search for files who contain the word of "keyword"
SearchFiles("file like 'keyword%'") ## search for files whose content begin with the word of
SearchFiles("name like '%keyword%'") ## search for files whose name end with the word of "key
SearchFiles("name like '%keyword one' and file like '%keyword tow%' ") ## combined condition
## End(Not run)
```

SummaryCoding	<i>Summary of coding</i>
---------------	--------------------------

Description

Give a summary of coding of current project.

Usage

```
SummaryCoding(byFile = FALSE, ...)  
## S3 method for class 'SummaryCoding':  
print
```

Arguments

<code>byFile</code>	When it is FALSE, return the summary of current project. When it is TRUE, return the summary of coding for each coded file.
<code>...</code>	Other possible arguments.

Value

A list:

<code>NumOfCoding</code>	Number of coding for each code.
<code>AvgLength</code>	Average number of characters in codings for each code.
<code>NumOfFile</code>	Number of files coded for each code
<code>CodingOfFile</code>	Number of codings for each file.NULL is byFile is FALSE.

Author(s)

HUANG Ronggui

See Also

[GetFileId](#) and [GetCodingTable](#)

Examples

```
## Not run:  
SummaryCoding()  
SummaryCoding(FALSE)  
## End(Not run)
```

```
write.FileList      Import a batch of files to the source table
```

Description

If import individual file to the project, you can do it by clicking import button in the Files Tab. Sometimes, you want to import a batch of files quickly, you can do it by command. This function is used to import a batch of files into the source table in the *.rqda file.

Usage

```
write.FileList(FileList, encoding = .rqda$encoding, con = .rqda$qdacon, ...)
```

Arguments

FileList	A list. Each element of the list is the file content, and the names (FileList) are the respective file name.
encoding	Don't change this argument.
con	Don't change this argument.
...	... is not used.

Details

The file content will converted to UTF-8 character before write to *.rqda. The original content can be in any suitable encoding, so you can inspect the content correctly; In other words, the better practices is to used the corresponding encoding (you can get a hint by localeToCharset function) to save the imported files.

Value

This function is used for the side-effects. No value is return.

Author(s)

Huang Ronggui

Examples

```
## Not run:
Files <- list("File name one"="content of first File.,"File name two"="content of the second File.")
write.FileList(Files) ## Please launch RQDA(), and open a project first.
## End(Not run)
```

Index

*Topic **package**

RQDA-package, [1](#)

*Topic **utilities**

Deletion, [3](#)

CleanProject (*Deletion*), [3](#)

Corpus, [14](#)

CrossCode, [2](#)

CrossTwo (*CrossCode*), [2](#)

dbSendQuery-methods, [15](#)

Deletion, [3](#)

ExportCoding, [4](#)

GetAttr, [5](#)

GetCaseId, [6](#)

GetCaseName (*GetCaseId*), [6](#)

GetCodingTable, [7, 20](#)

GetFileId, [7, 8, 9, 13, 20](#)

GetFileIdSets, [8, 9](#)

GetFileName (*GetFileId*), [8](#)

gselect.list, [10](#)

gtable, [19](#)

gtext, [13](#)

list.deleted (*Deletion*), [3](#)

localeToCharset, [19](#)

pdelete (*Deletion*), [3](#)

print.SummaryCoding
(*SummaryCoding*), [20](#)

Query, [11](#)

relation, [3, 12](#)

retrieval, [8, 9, 13](#)

RQDA (*RQDA-package*), [1](#)

RQDA-package, [1](#)

RQDA2tm, [14](#)

RQDAQuery, [14](#)

RQDATables, [15](#)

SearchFiles, [18](#)

select.list, [10](#)

ShowSubset (*GetAttr*), [5](#)

SummaryCoding, [20](#)

undelete (*Deletion*), [3](#)

write.FileList, [21](#)