Package 'datamart'

October 13, 2014

Type Package

Title Unified access to your data sources

Version 0.5.2

Date 2014-10-12

Author Karsten Weinert

Maintainer Karsten Weinert <k.weinert@gmx.net>

Description Provides an S4 infrastructure for unified handling
of internal datasets and web based data sources. The package is
currently in beta; things may break, change or go away without warning.

License GPL (>= 3)

Imports RJSONIO, XML, RCurl, base64, gsubfn, methods, markdown

LazyLoad yes

NeedsCompilation no

Repository CRAN

Date/Publication 2014-10-13 08:05:09

R topics documented:

`datamart-package` *Unified interface to your data sources.*

Description

This package provides several S4 classes that make it easier to collect and work with datasets. The package is inspired by the scraperwiki project, which provides a webbased service for data collection. Also inspiring are Mathematica's xxxData functions, which provide in-built parametrizable datasets.

Details

You can specify web resources with the `urldata` and the `xsparql` functions. For working with locally saved data, see the `internalData` and the `csvdata` function. The objects instantiated with these functions can than be passed to the generic `query` along with some parameters to get to the data.

You can combine several resources with the `datamart` function.

Besides parameterized queries ("read" operations), the package also aims to support "write" operations. For this purpose, some functions (currently `mdreport`, `swvreport`) for defining targets as well as some functions (currently `blogger` and `dirloc`) for defining locations are provided. The generic `put` then builds the target and puts it at the defined location.

Some examples aim to proof the concept, for instance `dbpedia`, `sourceforge`, `expenditures`, and `city_coords`.

The package is highly experimental, and likely to change heavily without backward compatiblity.

References

Karsten Weinert, factbased blogspot.

address_lookup *Reverse Geocoding: Adresse from Longitude and Latitude*

Description

The UrlData object provides a resource 'AddressLookup' which takes two parameters `lat` and `lon` and returns an approximate adress for this coordinates.

Usage

```
address_lookup()
```

Details

Based on Stackoverflow solution by Jochem Donkers.

Value

an object of class UrlData

References

Stackoverflow

allensbach *Election polls for Germany*

Description

This function exposes an interface to the IfD/Allensbach website's poll data.

Usage

```
allensbach(resource = "allensbach")
```

Arguments

`resource`	the resource name (character)

Details

It serves as a proof of concept for the `urldata` function.

See Also

`urldata`

as.character *String representation of an Xdata object*

Description

The `as.character` method is extended to various Xdata objects in order to give a more detailled description of the object instance.

Usage

```
## S4 method for signature 'DirectoryLocation'
as.character(x)

## S4 method for signature 'FileTarget'
as.character(x)

## S4 method for signature 'SftpLocation'
as.character(x)

## S4 method for signature 'WebLocation'
as.character(x)
```

Arguments

`x`	an Xdata object

Details

Inherited classes should override this method if necessary.

Blogger-class *Location Class for Google's Blogger service*

Description

This class implements a small subset of the blogger API v3. The `meta` method provides information on the submitted blogposts. The `put` method accepts a `BlogPostTarget` that can be transfered to Blogger.

Instantiates an object and authenticates with google. If the provided `oauthfile` parameter points to an existing file, the authentication information is loaded by `read.google.oauth2`, and the client_id and client_secret information are ignored. If `oauthfile` is missing, an initial authentication (directing the user to an website) is performed by `google.oauth2`.

Usage

```
blogger(oauthfile = getOption("blogger.oauthfile"),
  client_id = getOption("datamart.client_id"),
  client_secret = getOption("datamart.client_secret"),
  blogurl = getOption("blogger.blog"), clss = "Blogger")
```

Arguments

oauthfile	filename of previously saved authentication information.
client_id	client_id. See google.oauth2
client_secret	client_secret. See google.oauth2
blogurl	URL of the (existing) blog. Defaults to getOption("blogger.blog").
clss	name of the class for convenient inheritance. Defaults to "Blogger".

Value

Blogger

References

Blogger

See Also

blogger, mdreport

Examples

```
getSlots("Blogger")
```

BlogPostTarget-class *A Target representing a blog post*

Description

This is an internal class representing a blog post. Use MdReport instead.

For internal use only

Usage

```
blogpost(name, subject, content, label = "", draft = TRUE,
  overwrite = TRUE, clss = "BlogPostTarget")
```

Arguments

name	short (file) name of the blogpost
subject	title of the blogpost
content	content of the post
label	character vector of keywords
draft	draft or not? default=TRUE
overwrite	overwrite or not? overwrite=TRUE
clss	class of the object, default 'BlogPostTarget'

Value

BlogPostTarget

See Also

blogpost

Examples

```
getSlots("BlogPostTarget")
```

---

city_coords *Longitude and Latitude for Cities*

Description

The UrlData object provides a resource 'CityCoordinates' that takes a city parameter and returns a two-valued vector with latitude and longitude.

Usage

```
city_coords(country = "DE")
```

Arguments

country	two-character country code, default 'DE'

Details

Based on Stackoverflow solution by Jochem Donkers.

Value

an object of class UrlData

References

Stackoverflow

csvdata *Create CsvData object from data.frame*

Description

This function takes a data.frame, its metadata and a location. The function writes the data to the location and returns a CsvData object. The data.frame itself is not stored in this object.

Usage

```
csvdata(resource, dat, location, name = resource, dset.meta = NULL,
  cols.meta = NULL, update.fct = function(csv) return(data.frame()),
  verbose = TRUE, clss = "CsvData")
```

Arguments

resource	the name of the resource. Required.
dat	data.frame to convert to CsvData object. Required.
location	either a Location object, or a character pointing to a local directory. See details.
name	the physical name of the resource. Defaults to resource.
dset.meta	A list of metadata on the dataset.
cols.meta	A data.frame of metadata on the columns, rows are column names, columns are type, name, format
update.fct	function for updating the data. the update method and update interval is specified as meta data. Default is a function that returns an empty data.frame.
verbose	print diagnostic messages (default=TRUE)
clss	class to construct. Defaults to CsvData.

Details

If metadata is missing, it is set with sensible default values. The default values for the dataset metadata are title=<resource name>, modified=<current timestamp>, encoding='UTF-8', type='csv2', all other entries missing. The default values for column metadata are name=<column name of the data.frame>, type=<class(column)>, format='

No updating takes place here.

CsvData-class *CsvData – tabular data and its metadata*

Description

This class provides the infrastructure to read and write tabular data, including metadata on the dataset as a whole and metadata on the columns of the dataset.

Details

Metadata on the dataset that are not interpreted are title, description, license, publisher, keywords. Metadata on the dataset that is interpreted are update.scheme (always, never, and everything that can be passed to seq.POSIXt as by parameter), update.method (lastmod, fullreplace), encoding (an element of iconvlist()), type (csv, csv2), modified (automatically set, YYYY-mm-dd HH:MM:SS UTC format).

Metadata on a dataset column that are currently interpreted are name, type (logical, integer, numeric, complex, character, raw, factor, Date, POSIXct), format (for Date and POSIXct, format string for strptime).

See Also

csvdata, read.csvdata

Examples

```
getSlots("CsvData")
```

Dbpedia-class *A class for querying Dbpedia.org*

Description

This class defines some resources at dbpedia. See `queries(dbpedia())` for a list of resources.

Constructor for Dbpedia objects

Usage

```
dbpedia(lang = "")
```

Arguments

lang	two-character language code for the dbpedia, default ”

Value

a Dbpedia object, inherited from Xsparql

See Also

dbpedia, xsparql

Examples

```
## Not run:
  dbp <- dbpedia()
  queries(dbpedia)
  query(dbp, "Nuts1")

## End(Not run)
```

---

dependencies *Dependencies of a Xdata object*

Description

The dependencies method returns a list of character or list elements that define the resources the object depends on.

Usage

```
dependencies(self)

## S4 method for signature 'Xdata'
dependencies(self)

## S4 method for signature 'ResFunc'
dependencies(self)
```

Arguments

self	an Xdata object

Details

By default, NULL is returned.

Inherited classes should override this method if necessary.

DirectoryLocation-class *Directory location*

Description

The show method for the DirectoryLocation class has been adapted to display the path.

The dirloc function creates a DirectoryLocation object.

Usage

```
dirloc(path, clss = "DirectoryLocation")
```

Arguments

path	character, pointing to an existing directory. Required.
clss	character, optional class name. Default is "DirectoryLocation".

Details

The as.character method for the DirectoryLocation class returns the path to the directory it represents.

The meta method for the DirectoryLocation class returns the output of file.info of the folder.

See Also

dirloc

Examples

```
getSlots("DirectoryLocation")
```

enipedia *Interface to the Enipedia Project*

Description

Enipedia is an active exploration into the applications of wikis and the semantic web for energy and industry issues. Through this we seek to create a collaborative environment for discussion, while also providing the tools that allow for data from different sources to be connected, queried, and visualized from different perspectives.

Usage

```
enipedia()
```

Details

This function creates an datamart with selected queries to the SPARQL endpoint of the enipedia project.

Value

a Mashup object

See Also

enipedia, xsparql

---

enipedia_countries *Translation map between Enipedia country codes and ISO Codes*

Description

This dataset maps the 2-digit ISO 3166-1 alpha-2 country code to the country name used by enipedia.tudelf.nl

Usage

```
enipedia_countries
```

Format

CSV file (western european style)

Source

Wikipedia for ISO country codes, Enipedia for Enipedia names

---

evs2008.lvl2 *German Income and Expenditure Survey 2008 on private spendings, differentiated by household type and household income.*

Description

Germany's Sample survey of income and expenditure (Einkommens- und Verbrauchsstichprobe, EVS) is conducted by the Federal Statistical Office. The data provided here is processed and is not identical with der Federal Office' data. Some information is lost by the processing. If you want more and/or more accurate data, contact the Federal Statistical Office.

Usage

```
evs2008.lvl2
```

Format

CSV file (western european style)

Source

Statistisches Bundesamt: Wirtschaftsrechnungen. Einkommens- und Verbrauchsstichprobe. Einnahmen und Ausgaben privater Haushalte. Fachserie 15 Heft 4.

---

expenditures *Elasticity of Private Households*

Description

This is a proof of concept for a use case of `datamart` to be used for organising internal datasets as well as some calculations and graphics on these datasets.

Usage

```
expenditures()
```

Details

The example uses data from the Germany's Sample survey of income and expenditure (Einkommens- und Verbrauchsstichprobe, EVS).

References

Statistisches Bundesamt: Wirtschaftsrechnungen. Einkommens- und Verbrauchsstichprobe. Einnahmen und Ausgaben privater Haushalte. Fachserie 15 Heft 4.

See Also

`internalData`, `datamart`

Examples

```
xp <- expenditures()
queries(xp)
query(xp, "categories")
query(xp, "elasticity", categ="05")
```

FileTarget-class *FileTarget*

Description

This class is a decorator for a physical existing file. A common workflow for other targets is to create a temporary file and then call put again with a FileTarget.

see class FileTarget for details.

Usage

```
filetarget(name, filename, clss = "FileTarget")
```

Arguments

name	name of the Report, default ”
filename	name of original file
clss	class name, default 'FileTarget'

Value

generic

See Also

filetarget

Examples

```
getSlots("FileTarget")
```

gapminder *Gapminder data source.*

Description

Gapminder describes itself as a "fact tank" that promotes a fact based world view. On their website they provide a service that allows to create animated charts for various indicators, differentiated by country. They also provide the underlying datasets for download. This S3 class serves as a wrapper for easy access to a subset of these data.

Usage

```
gapminder()
```

Details

Please note that neither Gapminder nor the package developer/maintainer are the data provider, except for a few cases. Therefore you will have to go to the source to find out the terms of use for the specific indicator.

This class defines some resources of the Gapminder Project. See `queries(gapminder())` for a list of resources.

This is a proof of concept for the `urldata` function.

Value

Mashup

References

http://www.gapminder.org

See Also

urldata

Examples

```
## Not run:
  gm <- gapminder()
  queries(gm)
  query(gm, "ReligionAndBabies")

## End(Not run)
```

---

google.oauth2 *Google OAuth2 initial authentication*

Description

This function redirects the user to a consent screen where the user can grant or revoke access for the provided scope.

Usage

```
google.oauth2(scope, client_id = getOption("datamart.client_id"),
  client_secret = getOption("datamart.client_secret"), name = "",
  clss = "GoogleOAuth2", curl = RCurl::getCurlHandle(), verbose = TRUE)
```

Arguments

scope	either shortname ("blogger") or URL of the scope
client_id	client id. Defaults to getOption("datamart.client_id")
client_secret	client secret. Defaults to getOption("datamart.client_secret")
name	currently ignored
clss	the class to create. Must be derived from GoogleOAuth2
curl	curl.handle object
verbose	prints diagnostic messages on the way

Value

GoogleOAuth2 object

See Also

read.google.oauth2, google.oauth2

GoogleOAuth2-class *A class for managing Google OAuth2 workflow for installed apps*

Description

This class is aimed to manage the process where the user grants Google account access to your R application.

Details

There are two basic ways to create an GoogleOAuth2 object. First, you can use google.oauth2 to initially register your application with google. You may store this authentication information using put. Second, you can use read.google.oauth2 to load authentication information from a previously written file.

References

Google

See Also

google.oauth2,

InternalData-class *A class for querying data()sets*

Description

This class allows to query datasets that can be loaded with data(). Only read-only access.

Constructor for InternalData objects

Usage

```
internalData(name, package, clss = "InternalData")
```

Arguments

name	name of the dataset. Required.
package	name of the package where the dataset is located. Default NULL.
clss	name of the class to create. Default InternalData, must be inherited from this class.

See Also

internalData

Examples

```
getSlots("InternalData")
```

iwv_online *website traffic from IWV*

Description

website traffic as tracked by iwv online. Use yyyymm for query() as resource. This is an example for the urldata function.

Usage

```
iwv_online()
```

Value

UrlData object

References

Wikipedia

See Also

urldata

`localappdir` *Platform-independent local app folder*

Description

determines local app folder based on Sys.info()["sysname"].

Usage

```
localappdir(appname = "R", create = TRUE)
```

Arguments

`appname`	subdir in local app folder, default "R"
`create`	logical, default TRUE, create folder if non-existent

Value

character

`Location-class` *S4 base class to represent output Locations*

Description

The `Location` class is an abstract class that represents a place where to put something.

Mashup-class *Combine resource into a mashup object*

Description

The Mashup class administers a list of Xdata objects. This can be data objects representing different data sources such as internal data or web data. It can also be calculated data resources, so-called resource functions of class ResFunc.

Constructor for Mashup objects

Usage

```
datamart(..., clss = "Mashup")
```

Arguments

...	named arguments of Xdata objects
clss	name of the class to create. Default Mashup, must be inherited from this class.

Details

In a way, this class can be viewed as a make-like tool for data. The resource functions can declare dependencies. When a resource is requested by the query method, the Mashup class takes care of the build order.

See Also

datamart

Examples

```
getSlots("Mashup")
```

MdFigure-class *MdFigure*

Description

Internal Class representing figures in MdReport

Internal function to create an MdFigure object.

Usage

```
mdfigure(name, xdata, resource = name, clss = "MdFigure")
```

Arguments

name	name of the figure, default ”
xdata	call for creating the figure
resource	name of the resource
clss	class name, default 'MdFigure'

Value

MdFigure

See Also

mdfigure

Examples

```
getSlots("MdReport")
```

MdReport-class *Buildable markdown report*

Description

This class provides the basis for building dynamic reports using markdown text and resource from a datamart. You can create a report with mdreport, which takes a template file name, and a list of variables as arguments. The generic method put can then be used to actually produce the report at various locations (directory, memory, blogging site).

see class MdReport for details.

Usage

```
mdreport(tpl, name = "", xdata = NULL, clss = "MdReport",
  verbose = getOption("verbose"), ...)
```

Arguments

tpl	path to markdown template file
name	name of the Report, default ”
xdata	instance of Xdata class
clss	class of the object, default 'MdReport'
verbose	diagnostic messages T/F
...	number of targets

Details

strsubst is a simple templating mechanism inspired from Python (PEP-0292). Variables in the template are marked by a preceeding dollar sign and get replaced with the value of the corresponding variables passed to strsubst.

Value

generic

See Also

mdreport, swvreport

Examples

```
getSlots("MdReport")
```

---

mem.info *Information on objects in R environment*

Description

This function creates a list of objects that are currently in a given R environment (default the global workspace). Hence it is an extended version of ls.

Usage

```
mem.info(envir = .GlobalEnv, sortBy = "Size")
```

Arguments

envir	the environment to inspect, default is .GlobalEnv
sortBy	the result will be decreasingly sorted by this column. Possible values "Type", "Size" (default), "Rows", "Columns"

Value

data.frame with object information

Author(s)

Petr Pikal, David Hinds and Dirk Eddelbuettel

References

Stackoverflow

MemoryLocation-class *S4 base class to represent output in Memory*

Description

The MemoryLocation class represents the place in the memory of the current R process.

Examples

```
getSlots("MemoryLocation")
```

MemoryLocation2-class *S4 base class to represent output in Memory*

Description

The MemoryLocation2 class represents the place in the memory of the current R process.

Constructor for MemoryLocation2 objects

Usage

```
memloc(clss = "MemoryLocation2", ...)
```

Arguments

clss	class to construct. Defaults to MemoryLocation2.
...	initial objects that are contained in the location

See Also

memloc

Examples

```
getSlots("MemoryLocation2")
```

meta *Verbose list of resources*

Description

The meta method returns a data.frame with meta information entities available at the location. By default, zero rows are returned.

Usage

```
meta(self, ...)

## S4 method for signature 'Xdata'
meta(self, ...)

## S4 method for signature 'Xsparql'
meta(self)

## S4 method for signature 'InternalData'
meta(self, ...)

## S4 method for signature 'DirectoryLocation'
meta(self)

## S4 method for signature 'MemoryLocation2'
meta(self, ...)

## S4 method for signature 'SftpLocation'
meta(self)

## S4 method for signature 'Pastebin'
meta(self)

## S4 method for signature 'WebLocation'
meta(self)

## S4 method for signature 'CsvData'
meta(self, ...)

## S4 method for signature 'Blogger'
meta(self)
```

Arguments

self	an Location object
...	additional parameters

Details

Inherited classes should override this method if necessary.

`netConnectGermany` *Basic Price Information on natural Gas in Germany*

Description

This function exposes an interface to the Netconnect Germany website and allows the download of price data on natural gas.

Usage

```
netConnectGermany(clss = "UrlData")
```

Arguments

`clss`	Class name for the object, default UrlData2

Value

UrlData2

References

NCG

`openei` *Interface to the OpenEI platform*

Description

OpenEI is growing into a global leader in the energy data realm - specifically analyses on renewable energy and energy efficiency. The platform is a wiki, similar to Wikipedia's Wiki, and offers an SPARQL endpoint.

Usage

```
openei()
```

Details

The `openei` function provides an datamart with predefined queries on this SPARQL endpoint.

References

Blogpost by Chris Davis, OpenEI website

See Also

openei, xsparql

Pastebin-class *Pastebin*

Description

This class exposes partially the Web API to the pastebin service.

see Pastebin class for more information

Usage

```
pastebin(api_dev_key = getOption("pastebin.api_dev_key"),
  api_user_name = getOption("pastebin.api_user_name"),
  api_user_password = getOption("pastebin.api_user_password"),
  clss = "Pastebin")
```

Arguments

api_dev_key	API Dev Key, default getOption("pastebin.api_dev_key")
api_user_name	API User Name, default getOption("pastebin.api_user_name")
api_user_password	API User password, default getOption("pastebin.api_user_password")
clss	Class name to initiate, default "Pastebin"

See Also

pastebin

Examples

```
getSlots("Pastebin")
```

put *Put a target*

Description

This generic creates the target at the given location.

Usage

```
put(target, where, ...)

## S4 method for signature 'Target,character'
put(target, where, ...)

## S4 method for signature 'Target,MemoryLocation2'
put(target, where, ...)

## S4 method for signature 'FileTarget,DirectoryLocation'
put(target, where, overwrite = TRUE)

## S4 method for signature 'FileTarget,SftpLocation'
put(target, where, ...)

## S4 method for signature 'character,SftpLocation'
put(target, where, ...)

## S4 method for signature 'BlogPostTarget,DirectoryLocation'
put(target, where, ...)

## S4 method for signature 'BlogPostTarget,SftpLocation'
put(target, where, ...)

## S4 method for signature 'MdFigure,DirectoryLocation'
put(target, where, ...)

## S4 method for signature 'MdFigure,MemoryLocation'
put(target, where)

## S4 method for signature 'MdReport,Location'
put(target, where, draft = TRUE,
  overwrite = TRUE, ...)

## S4 method for signature 'SweaveReport,DirectoryLocation'
put(target, where, verbose = TRUE,
  ...)

## S4 method for signature 'SweaveReport,missing'
```

```
put(target, where, verbose = TRUE, ...)

## S4 method for signature 'CsvData,DirectoryLocation'
put(target, where, verbose = TRUE, ...)

## S4 method for signature 'CsvData,SftpLocation'
put(target, where, verbose = TRUE, ...)

## S4 method for signature 'GoogleOAuth2,character'
put(target, where, ...)

## S4 method for signature 'BlogPostTarget,Blogger'
put(target, where, verbose = TRUE, ...)
```

Arguments

target	an object of class Target or derived
where	an object of class Location or derived
...	additional parameters
overwrite	parameter for FileTarget/MdReport – overwrite existing files? Default TRUE.
draft	Parameter to MdReport, is draft? Logical.
verbose	print diagnostic messages

---

queries *List resources*

Description

The queries method returns a character vector of all defined resources for the given data object.

Usage

```
queries(self)

## S4 method for signature 'Xdata'
queries(self)

## S4 method for signature 'Xsparql'
queries(self)

## S4 method for signature 'InternalData'
queries(self)

## S4 method for signature 'UrlData'
queries(self)
```

```
## S4 method for signature 'Mashup'
queries(self)

## S4 method for signature 'Location'
queries(self)

## S4 method for signature 'MemoryLocation2'
queries(self)

## S4 method for signature 'ResFunc'
queries(self)

## S4 method for signature 'CsvData'
queries(self)
```

Arguments

self	an Xdata object

Details

The default (XData) implementation inspects definitions of the query method. Inherited classes should override this method if necessary.

---

query *Request data from data source*

Description

This generic function is the main interface to the data behind the Xdata layer. The first argument is the data object, the second argument is an identifier (type character), of the resource requested.

Usage

```
query(self, resource, ...)

## S4 method for signature 'Xdata,character'
query(self, resource, ...)

## S4 method for signature 'Xsparql,character'
query(self, resource, maxrows = NULL,
  interactive = FALSE, typeconv = TRUE, verbose = getOption("verbose"),
  ...)

## S4 method for signature 'InternalData,character'
query(self, resource, ...)
```

```
## S4 method for signature 'UrlData,character'
query(self, resource,
  verbose = getOption("verbose"), ...)

## S4 method for signature 'Mashup,character'
query(self, resource, verbose = TRUE, ...)

## S4 method for signature 'DirectoryLocation,character'
query(self, resource,
  verbose = getOption("verbose"), extract.fct = readLines, ...)

## S4 method for signature 'MemoryLocation2,character'
query(self, resource, ...)

## S4 method for signature 'SftpLocation,character'
query(self, resource,
  verbose = getOption("verbose"), ...)

## S4 method for signature 'Pastebin,character'
query(self, resource,
  verbose = getOption("verbose"), ...)

## S4 method for signature 'ResFunc,character'
query(self, resource,
  verbose = getOption("verbose"), ...)

## S4 method for signature 'WebLocation,character'
query(self, resource,
  verbose = getOption("verbose"), extract.fct = readLines, ...)

## S4 method for signature 'CsvData,character'
query(self, resource,
  verbose = getOption("verbose"), for.update = FALSE, ...)

## S4 method for signature 'GoogleOAuth2,character'
query(self, resource,
  curl = RCurl::getCurlHandle(), ...)
```

Arguments

self	an Xdata object
resource	an identifier of the resource requested. End-user usually provide character, developer use resource and dispatch on the type.
...	additional parameter
maxrows	(Xsparql) limit of lines to return (default NULL)
interactive	(Xsparql) if TRUE, display result in chunks (default FALSE)
typeconv	(Xsparql) if TRUE (default), convert numbers and dates

verbose	print diagnostic messages, default=FALSE
extract.fct	(DirectoryLocation) which function to use to read file (default readLines)
for.update	(CsvData) update before loading (default FALSE)
curl	(GoogleOAuth2) curl handle

Details

Depending on the data object, additional parameter can be provided.

---

read.csvdata *Read CSV from local directory or Internet*

Description

It is lazy, i.e. it does not read the csv in until requested. It does read in the metadata, though. If according to the dataset meta data an update is due, an update process is performed.

Usage

```
read.csvdata(resource, location = NULL, name = resource,
  update.fct = function(csv) return(data.frame()), verbose = TRUE,
  clss = "CsvData")
```

Arguments

resource	the name of the resource. Required.
location	either a Location object, or a character pointing to a local directory or an url. See details.
name	the physical name of the resource. Defaults to resource.
update.fct	function for updating the data. the update method and update interval is specified as meta data. Default is a function that returns an empty data.frame.
verbose	Print diagnostic messages, default is TRUE.
clss	class to construct. Defaults to CsvData.

read.google.oauth2 *Refreshed Google OAuth2 authentication*

Description

This function reads a file previously created via `put(oauth2, filename)`.

Usage

```
read.google.oauth2(fn)
```

Arguments

`fn`	data filename

Value

GoogleOAuth2 object

See Also

`google.oauth2`

resalias *Phony ResFunc Objects*

Description

This function creates a ResFunc object that returns an already existing resource.

Usage

```
resalias(resource, alias_for, clss = "ResFunc")
```

Arguments

`resource`	the name of the resource. Required.
`alias_for`	name of the resource this object is a proxy for.
`clss`	name of the class to create. Default ResFunc, must be inherited from this class.

See Also

`resfunc`

resfunc *Constructor for ResFunc objects*

Description

This function creates an ResFunc object. When queried, it returns the result of an function call.

Usage

```
resfunc(resource, fun, depends = list(), clss = "ResFunc", ...)
```

Arguments

resource	the name of the resource. Required.
fun	a function that matches the signature function(self, resource, ...)
depends	the names of the resources this function depends on.
clss	name of the class to create. Default ResFunc, must be inherited from this class.
...	additional parameters passed to the function when the resource is queried.

ResFunc-class *ResFunc – A class representing a "calculated" resource*

Description

While the UrlData and InternalData classes are examples for accessing actual data, the ResFunc class represents resources that are not physical data but for example simulated data or data derived from physical data.

Details

This class is intended to be used with the Mashup class.

See Also

resfunc, resalias, datamart

SftpLocation-class *SFTP location*

Description

This class represents a directory on a remote server that is accessed via the SFTP protocol. The only SSH authentication currently supported the public RSA key method.

The sftpdir function creates a SftpLocation object.

Usage

```
sftpdir(uri, pubkey, privatekey, keypasswd = "", clss = "SftpLocation")
```

Arguments

uri	character, the address of the remote host in the form sftp://username@the.host.com:port/remote/dir/
pubkey	character, pointing to the public key file. Required.
privatekey	character, pointing to the private key file. Required.
keypasswd	character, optional key password
clss	character, optional class name. Default is "SftpLocation".

Details

The show and as.character methods for this class have been adapted to reveal the URL of the directory it represents.

The meta method returns a data.frame with filenames and an is.dir attribute. It currently returns wrong values when the remote directory contains files with spaces in their file name.

See Also

sftpdir

Examples

```
getSlots("SftpLocation")
```

show *Show Method for Xdata classes*

Description

The show method for the Xdata class has been adapted to display the class name. Some inherited classes such DirectoryLocation or Blogger override this default definition.

Usage

```
## S4 method for signature 'Xdata'
show(object)

## S4 method for signature 'DirectoryLocation'
show(object)

## S4 method for signature 'Target'
show(object)

## S4 method for signature 'FileTarget'
show(object)

## S4 method for signature 'SftpLocation'
show(object)

## S4 method for signature 'Pastebin'
show(object)

## S4 method for signature 'WebLocation'
show(object)

## S4 method for signature 'Blogger'
show(object)
```

Arguments

object	Xdata object

sourceforge *SourceForge – query stats for sourceforge projects*

Description

This exposes part of the API of the sourceforge.net website. The function is a proof of concept for the urldata function.

Usage

```
sourceforge(proj, from = "2008-01-01", clss = "UrlData")
```

Arguments

proj	name of the project
from	when did the project start? Default "2008-01-01".
clss	which clss to instantiate, default "UrlData"

References

SourceForge

See Also

urldata

---

strcap *Capitalize a string*

Description

The first character is uppercased, the other lowercased.

Usage

```
strcap(s)
```

Arguments

s	character or character vector

Value

character

strdecrypt *Obfuscate string*

Description

reverts the action of strencrypt (not safe across machines!)

Usage

```
strdecrypt(message)
```

Arguments

message	character or character vector

Value

character

strdehtml *Decode HTML entities*

Description

Removes HTML named entities like & and numbered entities like ' by replacing it with the corresponding character.

Usage

```
strdehtml(s)
```

Arguments

s	character or character vector

Value

character or character vector

strencrypt *Obfuscate string*

Description

a fancy method to make the string unreadable use strdecypt to revert (not safe across machines!)

Usage

```
strencrypt(message)
```

Arguments

message	character or character vector

Value

character

strhead *Get the first n letters*

Description

if n>0, return the first n letters of x if n<0, return all but the last abs(n) letters of x

Usage

```
strhead(s, n = 1)
```

Arguments

s	character or character vector
n	numeric, default 1

Value

character

strparse *Parse named patterns*

Description

code based on examples for regexpr()

Usage

```
strparse(pat, x)
```

Arguments

pat	named pattern
x	character or character vector

Value

named character vector or matrix

strrecode *Pattern-based recoding*

Description

Pattern-based recoding

Usage

```
strrecode(pats, repls, x, ...)
```

Arguments

pats	vector of patterns
repls	vector of replacements
x	character or character vector
...	additional parameter, passed to grepl

Value

replaced vector

strsubst *Named substitution in strings*

Description

Simple template mechanism inspired by PEP-0292. Use lists or named character vectors (vectors not tested) as a mapping for substitution.

Usage

```
strsubst(template, map, verbose = getOption("verbose"))
```

Arguments

template	character with $(VARS)
map	object with [functionality e.g. a vector. Should return values that can be coerced to character
verbose	print debugging messages when TRUE, default is getOption("verbose")

Details

Substitutions are marked by $(NAME).

Value

character

References

Python-PEP-292 Stackoverflow

strtail *Get the last n letters*

Description

if n>0, return the last n letters of x if n<0, return all but the first abs(n) letters of x

Usage

```
strtail(s, n = 1)
```

Arguments

s	character or character vector
n	numeric, default 1

Value

character

`SweaveReport-class` *Wrapper for Sweave and pdf*

Description

This class provides the basis to build dynamic reports using LaTeX and resource from a datamart. You can create a report with `swvreport`, which takes a sweave file name. The generic method `put` can then be used to actually produce the report in pdf format.

see class SweaveReport for details.

Usage

```
swvreport(tpl, name = NULL, clss = "SweaveReport",
  verbose = getOption("verbose"), ...)
```

Arguments

`tpl`	path to markdown template file
`name`	name of the Report, default ”
`clss`	class of the constructed object, default 'SweaveReport'
`verbose`	diagnostic messages T/F
`...`	additional arguments, currently unused.

Value

generic

Author(s)

Karsten Weinert <k.weinert@gmx.net>

See Also

`swvreport`

Examples

```
getSlots("SweaveReport")
```

Target-class *Buildable target*

Description

This is an abstract class for defining buildable targets.

uconv *Convert between numerical units*

Description

This function converts between numerical units. It works similar to the `iconv` function: You provide vector `x` and a `from` and a `to` unit name and the function converts.

This is an internal class. It administers the unitsets used by the `uconv` method. One instance, usually the only one, is created at startup.

Internal function to create an UnitSetManager object.

Usage

```
uconv(x, from, to, uset = NULL)

unitsetmanager()
```

Arguments

`x`	numerical vector
`from`	character, unit to convert from.
`to`	character, unit to convert to
`uset`	optional, character, unit set to use.

Details

Additionally, you may provide a unitset name. Here, the analogy to `iconv` ceases. Think of unitset as a dimension of units, or a context for units. Predefined unitsets are "Length", "Mass", "Energy", and "Temperature". It is recommened to provide the unitset name. A list of available unitsets and the units defined by them can be obtained with `uconvlist()`.

This is a proof of concept for the `datamart` function.

See Also

`datamart`

Examples

```
uconv(1, "horse length", "m", "Length")
uconv(1:10, "TWh", "PJ", "Energy")
getSlots("UnitSetManager2")
```

`uconvlist` *List unitsets and their units*

Description

The function lists the currently available unitsets and the units supported by them.

Usage

```
uconvlist()
```

Value

named list, names=Unitsets, values=Units in these Unitsets

`urldata` *Constructor for UrlData objects*

Description

This function creates a web resource. As this, it allows to map an Web API into an R S4 class.

Usage

```
urldata(resource, template, extract.fct = readLines,
  transform.fct = identity, clss = "UrlData", ...)
```

Arguments

`resource`	the name of the resource. Required.
`template`	a pattern for the url. Must contain %s for substitution. Required.
`extract.fct`	a function that takes an URI and returns the raw data. Default readLines.
`transform.fct`	a function that takes the raw data and returns the cleaned/transformed data. Default identity.
`clss`	name of the class to create. Default UrlData, must be inherited from this class.
`...`	parameters for the query. Must be named arguments, values can be characters (for defaults), NULL, or functions.

UrlData-class *UrlData – unified access to WWW resources*

Description

This class provides the infrastructure to scrape the web with a Extract, Transform, Load (ETL) approach.

Details

In most cases, it is not necessary to subclass `UrlData`. The slots can be set by the `urldata` function and allow to customize each step of the process.

See Also

`urldata`

Examples

```
getSlots("UrlData")
```

WebLocation-class *Web location*

Description

Read-only web folder

This function returns a WebLocation object that can be used as a read-only folder.

Usage

```
webloc(path, clss = "WebLocation")
```

Arguments

`path`	character, pointing to an existing web directory. Required.
`clss`	character, optional class name. Default is "WebLocation".

Details

The queries and meta methods do not work for this class.

See Also

`webloc`

Examples

```
getSlots("WebLocation")
```

`Xdata-class` *Xdata – A class representing a data source*

Description

Most methods of the class are abstract, however the `show`, `print`, `queries` methods will usually not need to be redefined.

Details

The `query` method is defined for character resource arguments. It is tried to transform the argument to an object; if that succeeds, `query` is called again. Derived methods that are also interpreting resource as character should first call this method via `callNextMethod`.

`xsparql` *Constructor for Xsparql*

Description

The function `xsparql` constructs a Xsparql object.

Usage

```
xsparql(resource, url, statement, nspace = "", clss = "Xsparql", ...)
```

Arguments

`resource`	the resource the query represents
`url`	sparql end point
`statement`	the SPARQL statement the query stands for
`nspace`	character vector with short name / namespace expansions
`clss`	class to create, default Xsparql
`...`	named parameters for resources

Value

a Xsparql object

Xsparql-class *A class for querying SPARQL end points*

Description

This class allows to run SELECT statement on SPARQL endpoints. The resource parameter is interpreted as SPARQL statement.

See Also

xsparql, dbpedia, enipedia, openei

Examples

```
getSlots("Xsparql")
```

Index