Package ‘ProjectTemplate’

February 19, 2015

Type Package

Title Automates the creation of new statistical analysis projects.

Version 0.6

Date 2014-10-05

Description ProjectTemplate provides functions to
automatically build a directory structure for a new R
project. Using this structure, ProjectTemplate
automates data loading, preprocessing, library
importing and unit testing.

License Artistic-2.0

LazyLoad yes

Depends R (>= 2.7)

Suggests foreign, reshape, plyr, stringr, ggplot2, lubridate, log4r
(>= 0.1-5), DBI, RMySQL, RSQLite, gdata, RODBC, RJDBC, xlsx,
tuneR, pixmap, data.table, RPostgreSQL, GetoptLong, whisker,
testthat

URL http://projecttemplate.net

BugReports https://github.com/johnmyleswhite/ProjectTemplate/issues

Collate 'xport.reader.R' 'xlsx.reader.R' 'xls.reader.R' 'wsv.reader.R'
'url.reader.R' 'tsv.reader.R' 'systat.reader.R'
'stata.reader.R' 'require.package.R' 'sql.reader.R'
'spss.reader.R' 'rdata.reader.R' 'r.reader.R' 'ppm.reader.R'
'octave.reader.R' 'mp3.reader.R' 'file.reader.R'
'epinio.reader.R' 'dbf.reader.R' 'db.reader.R' 'csv2.reader.R'
'csv.reader.R' 'arff.reader.R' 'preinstalled.readers.R'
'add.extension.R' 'cache.R' 'cache.name.R' 'cache.project.R'
clean.variable.name.R' 'create.project.R' 'translate.def.R'
default.config.R' 'get.project.R' 'help.R' 'load.project.R'
migrate.project.R' 'new.config.R' 'reload.project.R'
'run.project.R' 'show.project.R' 'stub.tests.R'
test.project.R'


Author  Aleksandar Blagotic [ctb],
        Diego Valle-Jones [ctb],
        Jeffrey Breen [ctb],
        Joakim Lundborg [ctb],
        John Myles White [aut, cph],
        Josh Bode [ctb],
        Kenton White [ctb],
        Kirill Mueller [ctb, cre],
        Matteo Redaelli [ctb],
        Noah Lorang [ctb],
        Patrick Schalk [ctb]

Maintainer  Kirill Mueller <krlmr+r@mailbox.org>

NeedsCompilation  no

Repository  CRAN

Date/Publication  2014-10-06 00:57:23

R topics documented:

.add.extension ............................................. 3
arff.reader .................................................. 4
cache ......................................................... 4
cache.name .................................................. 5
cache.project ............................................... 6
clean.variable.name ....................................... 6
create.project .............................................. 7
csv.reader .................................................. 8
csv2.reader .................................................. 9
db.reader .................................................... 9
dbf.reader ................................................... 10
default.config ............................................. 11
epiinfo.reader ............................................. 11
file.reader .................................................. 12
get.project .................................................. 13
load.project ............................................... 13
migrate.project ............................................ 14
mp3.reader ................................................... 15
mtp.reader ................................................... 15
new.config .................................................. 16
octave.reader .............................................. 16
ppm.reader ................................................... 17
preinstalled.readers ..................................... 18
ProjectTemplate ............................................ 19
r.reader ...................................................... 20
rdata.reader ............................................... 20
reload.project ............................................. 21
require.package ............................................ 22
.add.extension

run.project ........................................... 23
show.project .......................................... 23
spss.reader ........................................... 24
sql.reader ............................................. 25
stata.reader .......................................... 26
stub.tests ............................................ 27
systat.reader ......................................... 27
test.project .......................................... 28
translate.dcf ......................................... 28
tsv.reader ............................................. 29
url.reader ............................................. 30
wsv.reader ............................................. 30
xls.reader ............................................. 31
xlsx.reader ........................................... 32
xport.reader .......................................... 32

Index .................................................. 34

..add.extension

Associate a reader function with an extension.

Description

This function will associate an extension with a custom reader function.

Usage

.add.extension(extension, reader)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>extension</td>
<td>The extension of the new data file.</td>
</tr>
<tr>
<td>reader</td>
<td>The function to use when reading the data file. It should accept three arguments: data.file, filename and variable.name (in that order). The function should read the contents of the file filename, and save it into the workspace under the name variable.name. The data.file argument is just a relative file name and can be ignored.</td>
</tr>
</tbody>
</table>

Value

No value is returned; this function is called for its side effects.

Warning

This interface should not be considered as stable and is likely to be replaced by a different mechanism in a forthcoming version of this package.
Examples

```r
## Not run: Nadd.extension('foo', foo.reader)
```

---

**arff.reader**  
*Read the Weka file format.*

---

### Description

This function will load a data set stored in the Weka file format into the specified global variable binding.

### Usage

```r
arff.reader(data.file, filename, variable.name)
```

### Arguments

- `data.file`: The name of the data file to be read.
- `filename`: The path to the data set to be loaded.
- `variable.name`: The name to be assigned to in the global environment.

### Value

No value is returned; this function is called for its side effects.

### Examples

```r
## Not run: arff.reader('example.arff', 'data/example.arff', 'example')
```

---

**cache**  
*Cache a data set for faster loading.*

---

### Description

This function will store a copy of the named data set in the cache directory. This cached copy of the data set will then be given precedence at load time when calling `load.project`. Cached data sets are stored as `.RData` files.

### Usage

```r
cache(variable)
```

### Arguments

- `variable`: A character vector containing the name of the variable to be saved.
Value

No value is returned; this function is called for its side effects.

Examples

```r
class('ProjectTemplate')
## Not run: create.project('tmp-project')
cwd('tmp-project')
dataset1 <- 1:5
cache('dataset1')
cwd('..')
unlink('tmp-project')
## End(Not run)
```

---

**cache.name**

Translate a variable name into a file name for caching.

Description

This function will translate a variable name into a form that is suitable as a filename on most OS’s.

Usage

```r
cache.name(data.filename)
```

Arguments

- `data.filename`: The variable name to be translated into a filename.

Value

A translated variable name.

Examples

```r
library('ProjectTemplate')
## Not run: cache.name('example.1')
```
cache.project  
*Cache a project’s data sets in binary format.*

**Description**

This function will cache all of the data sets that were loaded by the `load.project` function in a binary format that is easier to load quickly. This is particularly useful for data sets that you’ve modified during a slow munging process that does not need to be repeated.

**Usage**

```r
cache.project()
```

**Value**

No value is returned; this function is called for its side effects.

**See Also**

`create.project`, `load.project`, `get.project`, `show.project`

**Examples**

```r
library('ProjectTemplate')
## Not run: load.project()

cache.project()
## End(Not run)
```

clean.variable.name  
*Translate a file name into a valid R variable name.*

**Description**

This function will translate a file name into a name that is a valid variable name in R. Non-alphabetic characters on the boundaries of the file name will be stripped; non-alphabetic characters inside of the file name will be replaced with dots.

**Usage**

```r
clean.variable.name(variable.name)
```

**Arguments**

- `variable.name`  
  A character vector containing a variable’s proposed name that should be standardized.
create.project

Value
A translated variable name.

Examples
library('ProjectTemplate')

## Not run: clean.variable.name('example_1')

create.project Create a new project.

Description
This function will create all of the scaffolding for a new project. It will set up all of the relevant directories and their initial contents. For those who only want the minimal functionality, the minimal argument can be set to TRUE to create a subset of ProjectTemplate’s default directories. For those who want to dump all of ProjectTemplate’s functionality into a directory for extensive customization, the dump argument can be set to TRUE.

Usage
create.project(project.name = "new-project", minimal = FALSE, dump = FALSE, merge.strategy = c("require.empty", "allow.non.conflict"))

Arguments
project.name A character vector containing the name for this new project. Must be a valid directory name for your file system.
minimal A boolean value indicating whether to create a minimal project or a full project. A minimal project contains only the directories strictly necessary to use ProjectTemplate and does not provide template code for profiling, unit testing or documenting your project.
dump A boolean value indicating whether the entire functionality of ProjectTemplate should be written out to flat files in the current project.
merge.strategy What should happen if the target directory exists and is not empty? If "force.empty", the target directory must be empty; if "allow.non.conflict", the method succeeds if no files or directories with the same name exist in the target directory.

Details
If the target directory does not exist, it is created. Otherwise, it can only contain files and directories allowed by the merge strategy.

Value
No value is returned; this function is called for its side effects.
See Also

load.project, get.project, cache.project, show.project

Examples

library('ProjectTemplate')

## not run: create.project('MyProject')

csv.reader

Read a comma separated values (.csv) file.

Description

This function will load a data set stored in the CSV file format into the specified global variable
binding.

Usage

csv.reader(data.file, filename, variable.name)

Arguments

data.file        The name of the data file to be read.
filename         The path to the data set to be loaded.
variable.name    The name to be assigned to in the global environment.

Value

No value is returned; this function is called for its side effects.

Examples

library('ProjectTemplate')

## not run: csv.reader('example.csv', 'data/example.csv', 'example')
csv2.reader  
Read a semicolon separated values (.csv2) file.

Description
This function will load a data set stored in the CSV2 file format into the specified global variable binding.

Usage
csv2.reader(data.file, filename, variable.name)

Arguments
- data.file: The name of the data file to be read.
- filename: The path to the data set to be loaded.
- variable.name: The name to be assigned to in the global environment.

Value
No value is returned; this function is called for its side effects.

Examples
library('ProjectTemplate')

## Not run: csv2.reader('example.csv2', 'data/example.csv2', 'example')

---

db.reader  
Read a SQLite3 database with a (.db) file extension.

Description
This function will load all of the data sets stored in the SQLite3 database into the global environment. If you want to specify a single table or query to execute against the database, move it elsewhere and use a .sql file interpreted by sql.reader.

Usage
db.reader(data.file, filename, variable.name)

Arguments
- data.file: The name of the data file to be read.
- filename: The path to the data set to be loaded.
- variable.name: The name to be assigned to in the global environment.
Value
No value is returned; this function is called for its side effects.

Examples

```r
library('ProjectTemplate')

## not run: dbf.reader('example.db', 'data/example.db', 'example')
```

---

**dbf.reader**

Read an XBASE file with a .dbf file extension.

Description
This function will load all of the data sets stored in the specified XBASE file into the global environment.

Usage

```
dbf.reader(data.file, filename, variable.name)
```

Arguments

- `data.file` The name of the data file to be read.
- `filename` The path to the data set to be loaded.
- `variable.name` The name to be assigned to in the global environment.

Value
No value is returned; this function is called for its side effects.

Examples

```r
library('ProjectTemplate')

## not run: dbf.reader('example.dbf', 'data/example.dbf', 'example')
```
**default.config**

**Default configuration**

**Description**

This list stores the configuration used for missing items in the configuration of the current project.

**Usage**

```
default.config
```

**Format**

List of 11

- `$ version` : chr "0.5"
- `$ data_loading` : chr "TRUE"
- `$ cache_loading` : chr "TRUE"
- `$ recursive_loading` : chr "FALSE"
- `$ munging` : chr "TRUE"
- `$ logging` : chr "FALSE"
- `$ load_libraries` : chr "FALSE"
- `$ libraries` : chr "reshape, plyr, ggplot2, stringr, lubridate"
- `$ as_factors` : chr "TRUE"
- `$ data_tables` : chr "FALSE"
- `$ attach_internal_libraries` : chr "TRUE"

---

**epiinfo.reader**

**Read an Epi Info file with a .rec file extension.**

**Description**

This function will load all of the data sets stored in the specified Epi Info file into the global environment.

**Usage**

```
epiinfo.reader(data.file, filename, variable.name)
```

**Arguments**

- **data.file** The name of the data file to be read.
- **filename** The path to the data set to be loaded.
- **variable.name** The name to be assigned to in the global environment.
file.reader

Read an arbitrary file described in a .file file.

Description
This function will load all of the data sets described in the specified .file file into the global environment. A .file file must contain DCF that specifies the path to the data set and which extension should be used from the dispatch table to load the data set.

Usage
file.reader(data.file, filename, variable.name)

Arguments
- data.file: The name of the data file to be read.
- filename: The path to the data set to be loaded.
- variable.name: The name to be assigned to in the global environment.

Details
Examples of the DCF format and settings used in a .file file are shown below:
- path: http://www.johnmyleswhite.com/ProjectTemplate/sample_data.csv
- extension: csv

Value
No value is returned; this function is called for its side effects.

Examples
library('ProjectTemplate')

## Not run: file.reader('example.file', 'data/example.file', 'example')
get.project

Show information about the current project.

Description

This function will return all of the information that ProjectTemplate has about the current project. This information is gathered when load.project is called. At present, ProjectTemplate keeps a record of the project’s configuration settings, all packages that were loaded automatically and all of the data sets that were loaded automatically. The information about autoloaded data sets is used by the cache.project function.

Usage

get.project()

Details

In previous releases this information has been available through the global variable project.info. Using this variable is now deprecated and will result in a warning.

Value

A named list.

See Also

create.project, load.project, cache.project, show.project

Examples

library('ProjectTemplate')

## Not run: load.project()

get.project()
## End(Not run)

load.project

Automatically load data and packages for a project.

Description

This function automatically load all of the data and packages used by the project from which it is called.
Usage

load.project(override.config = NULL)

Arguments

override.config
  Named list, allows overriding individual configuration items.

Value

No value is returned; this function is called for its side effects.

See Also

create.project, get.project, cache.project, show.project

Examples

library("projecttemplate")

## not run: load.project()

migrate.project

Migrates a project from a previous version of ProjectTemplate

Description

This function automatically performs all necessary steps to migrate an existing project so that it is compatible with this version of ProjectTemplate

Usage

migrate.project()

Value

No value is returned; this function is called for its side effects.

See Also

create.project

Examples

library("ProjectTemplate")

## Not run: migrate.project()
mp3.reader  

Read an MP3 file with a .mp3 file extension.

Description

This function will load the specified MP3 file into memory using the tuneR package. This is useful for working with music files as a data set.

Usage

mp3.reader(data.file, filename, variable.name)

Arguments

data.file The name of the data file to be read.
filename The path to the data set to be loaded.
variable.name The name to be assigned to in the global environment.

Value

No value is returned; this function is called for its side effects.

Examples

library('ProjectTemplate')

## Not run: mp3.reader('example.mp3', 'data/example.mp3', 'example')

mtp.reader  

Read a Minitab Portable Worksheet with an .mtp3 file extension.

Description

This function will load the specified Minitab Portable Worksheet into memory.

Usage

mtp.reader(data.file, filename, variable.name)

Arguments

data.file The name of the data file to be read.
filename The path to the data set to be loaded.
variable.name The name to be assigned to in the global environment.
Value
No value is returned; this function is called for its side effects.

Examples
library('ProjectTemplate')

## Not run: mtp.reader('example.mtp', 'data/example.mtp', 'example')

new.config  

Description
This list stores the configuration used for new projects.

Usage
new.config

Format
List of 11
$ version  : chr "0.6"
$ data_loading  : chr "TRUE"
$ cache_loading  : chr "TRUE"
$ recursive_loading  : chr "FALSE"
$ munging  : chr "TRUE"
$ logging  : chr "FALSE"
$ load_libraries  : chr "FALSE"
$ libraries  : chr "reshape, plyr, ggplot2, stringr, lubridate"
$ as_factors  : chr "TRUE"
$ data_tables  : chr "FALSE"
$ attach_internal_libraries: chr "FALSE"

octave.reader  

Description
This function will load the specified Octave file into memory.

Usage
octave.reader(data.file, filename, variable.name)
ppm.reader

Arguments

data.file  The name of the data file to be read.
filename   The path to the data set to be loaded.
variable.name  The name to be assigned to in the global environment.

Value

No value is returned; this function is called for its side effects.

Examples

library('ProjectTemplate')

## not run: octave.reader('example.m', 'data/example.m', 'example')

ppm.reader                      Read a PPM file with a .ppm file extension.

Description

This function will load the specified PPM file into memory using the pixamp package. This is useful for working with image files as a data set.

Usage

ppm.reader(data.file, filename, variable.name)

Arguments

data.file  The name of the data file to be read.
filename   The path to the data set to be loaded.
variable.name  The name to be assigned to in the global environment.

Value

No value is returned; this function is called for its side effects.

Examples

library('ProjectTemplate')

## Not run: ppm.reader('example.ppm', 'data/example.ppm', 'example')
preinstalled.readers  Maps file types to the reader functions used to autoload them.

Description

This list stores a mapping from regular expressions that match file extensions for the file types supported by ProjectTemplate to the reader functions that implement autoloading for those formats. Any new file type must be appended to this dispatch table.

Usage

preinstalled.readers

Format

List of 52
$ *.csv  : chr "csv.reader"
$ *.csv.bz2 : chr "csv.reader"
$ *.csv.zip : chr "csv.reader"
$ *.csv.gz : chr "csv.reader"
$ *.csv2  : chr "csv2.reader"
$ *.csv2.bz2 : chr "csv2.reader"
$ *.csv2.zip : chr "csv2.reader"
$ *.csv2.gz : chr "csv2.reader"
$ *.tsv  : chr "tsv.reader"
$ *.tsv.bz2 : chr "tsv.reader"
$ *.tsv.zip : chr "tsv.reader"
$ *.tsv.gz : chr "tsv.reader"
$ *.tab  : chr "tsv.reader"
$ *.tab.bz2 : chr "tsv.reader"
$ *.tab.zip : chr "tsv.reader"
$ *.tab.gz : chr "tsv.reader"
$ *.wsv  : chr "wsv.reader"
$ *.wsv.bz2 : chr "wsv.reader"
$ *.wsv.zip : chr "wsv.reader"
$ *.wsv.gz : chr "wsv.reader"
$ *.txt  : chr "wsv.reader"
$ *.txt.bz2 : chr "wsv.reader"
$ *.txt.zip : chr "wsv.reader"
$ *.txt.gz : chr "wsv.reader"
$ *.Rdata  : chr "rdata.reader"
$ *.rda  : chr "rdata.reader"
$ *.R  : chr "r.reader"
$ *.r  : chr "r.reader"
$ *.url  : chr "url.reader"
$ *.sql  : chr "sql.reader"
$ *.xls  : chr "xls.reader"
ProjectTemplate

Automates the creation of new statistical analysis projects.

Description

ProjectTemplate provides functions to automatically build a directory structure for a new R project. Using this structure, ProjectTemplate automates data loading, preprocessing, library importing and unit testing.

References

This code is inspired by the skeleton structure used by Ruby on Rails.

Examples

library('ProjectTemplate')

## Not run: create.project('project_name')

setwd('project_name')
load.project()
## End(Not run)
**r.reader**  
*Read an R source file with a .R file extension.*

**Description**
This function will call `source` on the specified R file, executing the code inside of it as a way of generating data sets dynamically, as in many Monte Carlo applications.

**Usage**
```r
r.reader(data.file, filename, variable.name)
```

**Arguments**
- `data.file` The name of the data file to be read.
- `filename` The path to the data set to be loaded.
- `variable.name` The name to be assigned to in the global environment.

**Value**
No value is returned; this function is called for its side effects.

**Examples**
```r
library('ProjectTemplate')

## Not run: r.reader('example.R', 'data/example.R', 'example')
```

---

**rdata.reader**  
*Read an RData file with a .rdata or .rda file extension.*

**Description**
This function will load the specified RData file into memory using the `load` function. This may generate many data sets simultaneously.

**Usage**
```r
rdata.reader(data.file, filename, variable.name)
```

**Arguments**
- `data.file` The name of the data file to be read.
- `filename` The path to the data set to be loaded.
- `variable.name` The name to be assigned to in the global environment.
reload.project

Value

No value is returned; this function is called for its side effects.

Examples

```r
library('ProjectTemplate')

## Not run: rdata.reader('example.RData', 'data/example.RData', 'example')
```

Description

This function will clear the global environment and reload a project from scratch. This is useful when you’ve updated your data sets or changed your preprocessing scripts.

Usage

```r
reload.project(override.config = NULL)
```

Arguments

- `override.config`
  
  Named list, allows overriding individual configuration items.

Value

No value is returned; this function is called for its side effects.

Examples

```r
library('ProjectTemplate')

## Not run: load.project()

reload.project()
## End(Not run)
```
require.package

Require a package for use in the project

Description

This functions will require the given package. If the package is not installed it will stop execution and print a message to the user instructing them which package to install and which function caused the error.

Usage

require.package(package.name, attach = TRUE)

.load.package(package.name)

Arguments

package.name A character vector containing the package name. Must be a valid package name installed on the system.

attach Should the package be attached to the search path (as with library) or not (as with loadNamespace)? Defaults to TRUE. (Internal code will use FALSE by default unless a compatibility switch is set, see below.)

Details

The function .require.package is called by internal code. It will attach the package to the search path (with a warning) only if the compatibility configuration attach_internal_libraries is set to TRUE. Normally, packages used for loading data are not needed on the search path, but not loading them might break existing code. In a forthcoming version this compatibility setting will be removed, and no packages will be attached to the search path by internal code.

Value

No value is returned; this function is called for its side effects.

Examples

library('ProjectTemplate')

## Not run: require.package('PackageName')
**run.project**

Run all of the analyses in the **src** directory.

**Description**

This function will run each of the analyses in the **src** directory in separate processes. At present, this is done serially, but future versions of this function will provide a means of running the analyses in parallel.

**Usage**

```r
run.project()
```

**Value**

No value is returned; this function is called for its side effects.

**Examples**

```r
library('ProjectTemplate')

## Not run: run.project()
```

**show.project**

Show information about the current project.

**Description**

This function will show the user all of the information that ProjectTemplate has about the current project. This information is gathered when **load.project** is called. At present, ProjectTemplate keeps a record of the project’s configuration settings, all packages that were loaded automatically and all of the data sets that were loaded automatically. The information about autoloaded data sets is used by the **cache.project** function.

**Usage**

```r
show.project()
```

**Value**

No value is returned; this function is called for its side effects.

**See Also**

`create.project`, `load.project`, `get.project`, `cache.project`
spss.reader

Examples

library('ProjectTemplate')

## Not run: load.project()

show.project()
## End(Not run)

spss.reader(data.file, filename, variable.name)

Arguments

data.file The name of the data file to be read.
filename The path to the data set to be loaded.
variable.name The name to be assigned to in the global environment.

Value

No value is returned; this function is called for its side effects.

Examples

library('ProjectTemplate')

## Not run: spss.reader('example.sav', 'data/example.sav', 'example')
**sql.reader**

*Read a database described in a .sql file.*

**Description**

This function will load data from a SQL database based on configuration information found in the specified .sql file. The .sql file must specify a database to be accessed. All tables from the database, one specific tables or one specific query against any set of tables may be executed to generate a data set.

**Usage**

```r
sql.reader(data.file, filename, variable.name)
```

**Arguments**

- `data.file`: The name of the data file to be read.
- `filename`: The path to the data set to be loaded.
- `variable.name`: The name to be assigned to in the global environment.

**Details**

Queries can support string interpolation to execute code snippets using mustache syntax (http://mustache.github.io). This is used to create queries that depend on data from other sources. Code delimited is `{{...}}`.

Example: `query: SELECT * FROM my_table WHERE id IN ({{ids}})`. Here `ids` is a vector previously loaded into the Global Environment through `ProjectTemplate`.

Examples of the DCF format and settings used in a .sql file are shown below:

- **Example 1**: `type: mysql user: sample_user password: sample_password host: localhost dbname: sample_database table: sample_table`
- **Example 2**: `type: mysql user: sample_user password: sample_password host: localhost port: 3306 socket: /Applications/MAMP/tmp/mysql/mysql.sock dbname: sample_database table: sample_table`
- **Example 3**: `type: sqlite dbname: /path/to/sample_database table: sample_table`
- **Example 4**: `type: sqlite dbname: /path/to/sample_database query: SELECT * FROM users WHERE user_active == 1`
- **Example 5**: `type: sqlite dbname: /path/to/sample_database table: *
- **Example 6**: `type: postgres user: sample_user password: sample_password host: localhost dbname: sample_database table: sample_table`
- **Example 7**: `type: odbc dsn: sample_dsn user: sample_user password: sample_password dbname: sample_database query: SELECT * FROM sample_table`
- **Example 8**: `type: oracle user: sample_user password: sample_password dbname: sample_database table: sample_table`
Example 10 type: heroku classpath: /path/to/jdbc4.jar (or set in CLASSPATH) user: scott password: tiger host: heroku.postgres.url port: 1234 dbname: herokudb query: select * from emp

Value

No value is returned; this function is called for its side effects.

Examples

library('ProjectTemplate')

## Not run: sql.reader('example.sql', 'data/example.sql', 'example')

---

stata.reader

Read a Stata file with a .stata file extension.

Description

This function will load the specified Stata file into memory.

Usage

stata.reader(data.file, filename, variable.name)

Arguments

data.file       The name of the data file to be read.
filename        The path to the data set to be loaded.
variable.name   The name to be assigned to in the global environment.

Value

No value is returned; this function is called for its side effects.

Examples

library('ProjectTemplate')

## Not run: stata.reader('example.stata', 'data/example.stata', 'example')
stub.tests  Generate unit tests for your helper functions.

Description
This function will parse all of the functions defined in files inside of the lib directory and will generate a trivial unit test for each function. The resulting tests are stored in the file tests/autogenerated.R. Every test is excepted to fail by default, so you should edit them before calling test.project.

Usage
stub.tests()

Value
No value is returned; this function is called for its side effects.

Examples
library('ProjectTemplate')

## Not run: stub.tests()

systat.reader  Read a Systat file with a .sys or .syd file extension.

Description
This function will load the specified Systat file into memory.

Usage
systat.reader(data.file, filename, variable.name)

Arguments
data.file The name of the data file to be read.
filename The path to the data set to be loaded.
variable.name The name to be assigned to in the global environment.

Value
No value is returned; this function is called for its side effects.
Example

```r
library('ProjectTemplate')

## Not run: systat.reader('example.sys', 'data/example.sys', 'example')
```

---

**test.project**  
*Run all unit tests for this project.*

---

**Description**

This function will run all of the testthat style unit tests for the current project that are defined inside of the `tests` directory. The tests will be run in the order defined by the filenames for the tests: it is recommended that each test begin with a number specifying its position in the sequence.

**Usage**

```r
test.project()
```

**Value**

No value is returned; this function is called for its side effects.

**Examples**

```r
library('ProjectTemplate')

## Not run: load.project()

test.project()
## End(Not run)
```

---

**translate.dcf**  
*Read a DCF file into an R list.*

---

**Description**

This function will read a DCF file and translate the resulting data frame into a list. The DCF format is used throughout ProjectTemplate for configuration settings and ad hoc file format specifications.

**Usage**

```r
translate.dcf(filename)
```

**Arguments**

- **filename**  
  A character vector specifying the DCF file to be translated.
Value

Returns a list containing the entries from the DCF file.

Examples

library('ProjectTemplate')

## Not run: translate.dcf(file.path('config', 'global.dcf'))

###

**tsv.reader**  
*Read a tab separated values (.tsv or .tab) file.*

Description

This function will load a data set stored in the TSV file format into the specified global variable binding.

Usage

```
hts.reader(data.file, filename, variable.name)
```

Arguments

- **data.file**  
The name of the data file to be read.
- **filename**  
The path to the data set to be loaded.
- **variable.name**  
The name to be assigned to in the global environment.

Value

No value is returned; this function is called for its side effects.

Examples

library('ProjectTemplate')

## Not run: tsv.reader('example.tsv', 'data/example.tsv', 'example')
url.reader  
*Read a remote file described in a .url file.*

**Description**

This function will load data from a remote source accessible through HTTP or FTP based on configuration information found in the specified .url file. The .url file must specify the URL of the remote data source and the type of data that is available remotely. Only one data source per .url file is supported currently.

**Usage**

```r
url.reader(data.file, filename, variable.name)
```

**Arguments**

- `data.file`  
The name of the data file to be read.
- `filename`  
The path to the data set to be loaded.
- `variable.name`  
The name to be assigned to in the global environment.

**Details**

Examples of the DCF format and settings used in a .url file are shown below:

Example 1 url: http://www.johnmyleswhite.com/ProjectTemplate/sample_data.csv separator: ,

**Value**

No value is returned; this function is called for its side effects.

**Examples**

```r
library('ProjectTemplate')

## Not run: url.reader('example.url', 'data/example.url', 'example')
```

wsv.reader  
*Read a whitespace separated values (.wsv or .txt) file.*

**Description**

This function will load a data set stored in the WSV file format into the specified global variable binding.

**Usage**

```r
wsv.reader(data.file, filename, variable.name)
```
**xls.reader**

**Arguments**
- `data.file` The name of the data file to be read.
- `filename` The path to the data set to be loaded.
- `variable.name` The name to be assigned to in the global environment.

**Value**
No value is returned; this function is called for its side effects.

**Examples**
```r
cat("This function will load the specified Excel file into memory using the gdata package. Each sheet of the Excel workbook will be read into a separate variable in the global environment.

Usage
```r
xls.reader(data.file, filename, workbook.name)
```r

**Arguments**
- `data.file` The name of the data file to be read.
- `filename` The path to the data set to be loaded.
- `workbook.name` The name to be assigned to in the global environment.

**Value**
No value is returned; this function is called for its side effects.

**Examples**
```r
cat("library('ProjectTemplate')

## Not run: xls.reader('example.xls', 'data/example.xls', 'example')
```
xlsx.reader

Description

This function will load the specified Excel file into memory using the xlsx package. Each sheet of the Excel workbook will be read into a separate variable in the global environment.

Usage

xlsx.reader(data.file, filename, workbook.name)

Arguments

data.file The name of the data file to be read.
filename The path to the data set to be loaded.
workbook.name The name to be assigned to in the global environment.

Value

No value is returned; this function is called for its side effects.

Examples

library('ProjectTemplate')

## Not run: xlsx.reader('example.xlsx', 'data/example.xlsx', 'example')

xport.reader

Description

This function will load the specified XPort file into memory.

Usage

xport.reader(data.file, filename, variable.name)

Arguments

data.file The name of the data file to be read.
filename The path to the data set to be loaded.
variable.name The name to be assigned to in the global environment.
Value
No value is returned; this function is called for its side effects.

Examples
library('ProjectTemplate')

## Not run: xport.reader('example.xport', 'data/example.xport', 'example')
Index

*Topic datasets
  default.config, 11
  new.config, 16
  preinstalled.readers, 18
  .add.extension, 3
  .require.package (require.package), 22

arff.reader, 4

cache, 4
  cache.name, 5
  cache.project, 6, 8, 13, 14, 23
  clean.variable.name, 6
  create.project, 6, 7, 13, 14, 23
  csv.reader, 8
  csv2.reader, 9

  db.reader, 9
  dbf.reader, 10
  default.config, 11

epiinfo.reader, 11

file.reader, 12

get.project, 6, 8, 13, 14, 23

library, 22

load, 20

  load.project, 4, 6, 8, 13, 13, 23
  loadNamespace, 22

migrate.project, 14

mp3.reader, 15

mtp.reader, 15

new.config, 16

octave.reader, 16

package-ProjectTemplate
  (ProjectTemplate), 19

ppm.reader, 17

preinstalled.readers, 18

ProjectTemplate, 19

ProjectTemplate-package
  (ProjectTemplate), 19

r.reader, 20

rdata.reader, 20

reload.project, 21

require.package, 22

run.project, 23

show.project, 6, 8, 13, 14, 23

spss.reader, 24

sql.reader, 9, 25

stata.reader, 26

stub.tests, 27

systat.reader, 27

test.project, 27, 28

translate.dcf, 28

tsv.reader, 29

url.reader, 30

wsv.reader, 30

xls.reader, 31

xlsx.reader, 32

xport.reader, 32