Package 'Dasst'

January 20, 2025

Version 0.3.4

Date 2022-04-09

Title Tools for Reading, Processing and Writing 'DSSAT' Files

Author Homero Lozza [aut, cre]

Maintainer Homero Lozza <homerolozza@gmail.com>

Depends R (>= 2.14.0)

Description Provides methods for reading, displaying, processing and writing files originally arranged for the 'DSSAT-CSM' fixed width format. The 'DSSAT-CSM' cropping system model is described at J.W. Jones, G. Hoogenboomb, C.H. Porter, K.J. Boote, W.D. Batchelor, L.A. Hunt, P.W. Wilkens, U. Singh, A.J. Gijsman, J.T. Ritchie (2003) <doi:10.1016/S1161-0301(02)00107-7>.

License GPL (>= 2)

Imports methods

Collate 'Class-Dasst.R' 'Dasst-methods.R' 'Dasst.R' 'dasstOptions.R' 'extractData.R' 'extractHeader.R' 'buildContents.R' 'data.R' 'gatherTables.R' 'read.dssat.R' 'stackTables.R' 'write.dssat.R'

NeedsCompilation no

Repository CRAN

Date/Publication 2022-04-13 22:32:28 UTC

Contents

`Dasst-package` *Dasst: a package for reading, processing and writing DSSAT-style files*

Description

The **Dasst** (DSSAT-style AS Simple Tables) package provides methods for reading, processing and writing files originally formatted for the 'DSSAT-CSM' crop simulation models.

Details

The **Dasst** package defines the `Dasst` class which enables to store the data encoded for the 'DSSAT-CSM' crop simulation models as an S4 object, both for the inputs and the outputs. Several methods are available.

Author(s)

Homero Lozza, <homerolozza@gmail.com>

See Also

Dasst for class definition, `show`, `summary`, `[[ [` methods for content description, and `read.dssat` function for reading files. For other available methods see INDEX file.

An object example is available at `plantGrowth`.

addDate<- *Add date class to objects of class Dasst*

Description

addDate<- adds a column of class date to tables of the object of class Dasst.

Arguments

x	An object of class Dasst.
...	Other parameters: format, character vector encoding the date format;
value	A formula, numeric vector or character vector. Order of the column fields from where dates can be composed.

Details

This method adds a column of class date to tables of the object of class Dasst. Dates expressed as string or integers may be converted and stored as date objects in a new column whose name begins with "date_" and follows with the names of column fields involved in the date extraction.

So far, the new column will not be saved if the write method is invoked.

Value

The actual object.

Examples

```
data(plantGrowth)
addDate(plantGrowth) <- ~ YEAR + DOY

# or
addDate(plantGrowth) <- c("YEAR", "DOY")

# or
addDate(plantGrowth) <- c(1, 2)

# Only one tables 1 and specifying date format
addDate(plantGrowth, index=c(1,2), format="%Y%j") <- ~ YEAR + DOY
```

as *As forces an object of class Dasst to belong to class list*

Description

Coerces an object of class `Dasst` to an object of class `list`.

Details

This function enables the function `as` to coerce objects of class `Dasst` to belong to class `list`.

Examples

```
data(plantGrowth)
lplantgro <- as(plantGrowth, "list")
class(lplantgro)
```

buildContents *Build contents for an object of class Dasst.*

Description

`buildContents` generates contents that are inserted into an object of class `Dasst`.

Usage

```
buildContents(fileName, section, headerLine, dataLine,
  table)
```

Arguments

`fileName`	A character string. The file name and path corresponding to the generated contents.
`section`	A character string. The section title corresponding to the generated contents.
`headerLine`	A character string. The header names for the table of data included in the generated contents.
`dataLine`	A character string. A typically formatted line of data.
`table`	A data.frame. The records for the table of data included in the generated contents are inserted as a `data.frame`

Details

This function builds the contents that are inserted into an object of class `Dasst` from a `data.frame` and ancillary character strings.

The `data.frame` contains the actual data that is stored as a table within the object of class `Dasst`.

Value

An object of class Dasst.

Examples

```
mydf <- data.frame(a=c(1,2,3), b=c("one","two","three"),c=c(1.1,2.2,3.3))
myObj <- Dasst()
myObj[1] <- buildContents("MyTest.OUT","*TestSec",
 "@ID  NAME VALUE","  1   one 1.100", mydf)
```

---

compute<- *Compute within columns from an object of class* Dasst.

Description

compute<- computes an expression using the columns of the object of class Dasst.

Arguments

x	An object of class Dasst.
cocol	A character string. The name of the new column field.
value	A character string. An expression to compute within column fields.

Details

This method computes an expression taking the values recorded on each column field used in the expression belonging to the object of class Dasst. The result is stored as a new column table.

So far, the new column will not be saved if the write method is invoked.

Value

The actual object of class Dasst.

Examples

```
data(plantGrowth)
compute(plantGrowth, "date_YEAR_DOY") <-
 "as.Date(paste(YEAR, DOY, sep=\"\"), format=\"%Y%j\")"
```

Dasst *A constructor for an empty object of class Dasst*

Description

Dasst is a constructor for the user that returns an empty object of class Dasst.

Usage

```
Dasst()
```

Details

This function constructs an empty object of class Dasst. No arguments are required.

Value

An empty object of class Dasst.

See Also

Dasst for class definition.

Examples

```
myObj <- Dasst()
myObj
class(myObj)
length(myObj)
```

Dasst-class *Dasst class description*

Description

An S4 class that stores information recorded on DSSAT-style files.

Slots

fileNames: A character vector containing the names and the paths to the original data files.

sections: A character vector containing the names for each section within the DSSAT-style format specification and structure.

fields: A list of data.frame containing the names, modes, and widths in characters for each data field. The number of decimal digits are also stored in each numeric field.

tables: A list of data.frame containing the actual values retrieved from the original file.

See Also

show, summary, [[[methods for content description, and read.dssat function for reading files. For other available methods see INDEX file.

An object example is available at plantGrowth.

---

gatherTables *Gather tables of an object of class Dasst.*

Description

gatherTables gathers the result of performing a certain operation over the tables of an object of class Dasst.

Usage

```
gatherTables(object, coCol, opCol, operation, ...)
```

Arguments

object	Object of class Dasst.
coCol	A character vector. The field names of those columns that will be copied identically into the result.
opCol	A character vector. The field names of those columns that will be gather by means of applying the required operation.
operation	A function. The function name for the required operation. i.e. mean, sum, etc.
...	Other parameters for the apply function as additional arguments for the operation.

Details

This function gathers the result of performing a certain operation over the tables of an object of class Dasst. The result is given as a data.frame.

Value

A data.frame with the values gathered after the application of the operator to the required columns.

Examples

```
data(plantGrowth)
plantgro12 <- gatherTables(plantGrowth[1:10], c("DAP"),
   c("SWAD","LWAD","GWAD"), mean)
```

getAncillary *Get ancillary data from an object of class* Dasst

Description

getAncillary gets ancillary data from an object of class Dasst connected to the selected table orders.

Arguments

x	An object of class Dasst.
i	An optional integer vector. Orders where to retrieve ancillary data. The default action is to retrieve all the available ancillary data.

Details

This method gets ancillary data from an object of class Dasst connected to the selected table orders. Values are arranged in tables, and the order is the number assigned successively to each of them after the data have been stored within the Dasst object. getAncillary provides ancillary data such as the file name which was originally read, and the section and the header which introduced the values within the file.

Value

An object of class Ancillary which contains the retrieved ancillary data for the selected table orders.

Examples

```
data(plantGrowth)
getAncillary(plantGrowth, c(1,3,5))
```

length.Dasst *Length of an object of class* Dasst.

Description

length.Dasst computes the length of an object of class Dasst.

Usage

```
## S3 method for class 'Dasst'
length(x)
```

Arguments

x	Object of class Dasst.

Details

This function extends the S3 `length` generic function. It computes the length of an object of class `Dasst`. The length equals the quantity of stored tables. The empty object has length `0`.

Value

An integer representing the length of the object.

See Also

`length<-.Dasst`

Examples

```
data(plantGrowth)
length(plantGrowth)
```

---

length<-.Dasst *Set the length of an object of class* Dasst.

Description

This sets the length of an object of class `Dasst`.

Usage

```
  ## S3 replacement method for class 'Dasst'
length(x) <- value
```

Arguments

x	Object of class `Dasst`.
value	Integer value. Sets the new length of the `Dasst` object.

Details

`length<-.Dasst` function extends the S3 `length` generic function. It sets the length of an object of class `Dasst`. The object can be shrinked or extended adding `NULL` or `NA` contents.

Value

An integer value corresponding to the actual length of the object.

See Also

`length.Dasst`

Examples

```
data(plantGrowth)
length(plantGrowth)
length(plantGrowth) <- 8
length(plantGrowth)
```

`plantGrowth` *An example of a Dasst object*

Description

An example of a `Dasst` object based on file PlantGro.OUT. The originally data set can be found within package directory as '`extdata/PlantGro.OUT`'.

Details

`plantGrowth` contains the daily plant growth for 3 treatments repeated on 10 years. A summary of its contents follows

Table 1 45 fields and 131 records

Table 2 45 fields and 112 records

Table 3 45 fields and 124 records

Table ... ...

Table 30 45 fields and 123 records

Tables ranging from 1 to 10 have plant output data for treatment 1 which simulates maize growth from 1970 to 1979 (south hemisphere). Treatment 2 adds fertilization with 50kg/ha of urea, and treatment 3 adds fertilization with 100kg/ha of urea.

See Also

`Dasst` for class definition and the example at `read.dssat`.

Examples

```
data(plantGrowth)
plot(plantGrowth[[1]][,"DAP"],plantGrowth[[1]][,"LAID"])

# Or

plot(plantGrowth[[1]][,c("DAP","LAID")])
```

print.Ancillary *Print object of class* Ancillary

Description

print.Ancillary prints the contents of an object of class Ancillary.

Usage

```
## S3 method for class 'Ancillary'
print(x, ...)
```

Arguments

x	Object of class Ancillary.
...	Arguments that may be passed to other functions.

Details

This function extends the S3 print generic function. It prints the contents of an object of class Ancillary.

Value

An invisible object.

Examples

```
data(plantGrowth)
getAncillary(plantGrowth, 1:5)
```

print.Dasst *Print object of class Dasst*

Description

print.Dasst prints the contents of an object of class Dasst.

Usage

```
## S3 method for class 'Dasst'
print(x, ix = 1, ...)
```

Arguments

x	Object of class Dasst.
ix	An integer number. The contents of the first table are print by default. Others table contents can be display setting this parameter in the range form 1 to length(x).
...	Other parameters for the print.data.frame function that specify how tables should look.

Details

This function extends the S3 print generic function. It prints the contents of an object of class Dasst.

Value

An invisible object.

Examples

```
data(plantGrowth)
print(plantGrowth)
```

---

print.summary.Dasst *Print object of class* summary.Dasst

Description

print.summary.Dasst prints the contents of an object of class summary.Dasst.

Usage

```
## S3 method for class 'summary.Dasst'
print(x, ...)
```

Arguments

x	Object of class summary.Dasst.
...	Arguments that may be passed to other functions.

Details

This function extends the S3 print generic function. It prints the contents of an object of class summary.Dasst.

Value

An invisible object.

Examples

```
data(plantGrowth)
summary(plantGrowth)
```

`read.dssat` *Read a DSSAT-style file into an object of class Dasst*

Description

`read.dssat` reads the contents of a file or group of files and stores the contents into an object of class `Dasst`.

Usage

```
read.dssat(fileVec, fieldVec = character(),
  keyVec = character())
```

Arguments

`fileVec`	A character vector. The names including the paths to the files that will be read.
`fieldVec`	A character vector. An optional parameter. If it is not specified, all column fields are retrieved. Else, the names subset are the only retrieved column fields.
`keyVec`	A character vector. An optional parameter. If it is not specified, only sections satisfying this condition will be retrieved.

Details

This function reads the contents of a file or group of files and stores the contents into an object of class `Dasst`.

Value

A `Dasst` object with the structure and information originally saved in the file using DSSAT-style format specifications.

Examples

```
dssatfile <- system.file("extdata","PlantGro.OUT",package="Dasst")
dssatfile
plantgro <- read.dssat(dssatfile)
summary(plantgro)
```

searchAncillary *Search for ancillary data within the* Dasst *object*

Description

searchAncillary looks for ancillary data that satisfies the search criteria and gives the table orders in the Dasst object for successful results.

Arguments

x	An object of class Dasst.
fileKey	A character string. Search for this pattern within the "filename" slot.
secKey	A character string. Search for this pattern within the "section" slot.
colKey	A character string. Search for this pattern within the tables column names.
...	Other parameters than may be passed to grepl.

Details

This method searches for character strings or regular expressions in the ancillary data of the Dasst object. Patterns are sought into "fileNames" and "sections" slots, and table column names. The corresponding table orders whose ancillary data satisfied the search criteria are gathered in a vector.

Value

An integer representing the table orders whose ancillary data satisfied the search criteria.

Examples

```
data(plantGrowth)
searchAncillary(plantGrowth, secKey="run[[:space:]]*1")
searchAncillary(plantGrowth, secKey="run[[:space:]]*1", ignore.case=TRUE)
```

show *Show method for class* Dasst

Description

show shows a few contents of an object of class Dasst.

Details

This method shows the contents of the first table stored in an object of class Dasst. It displays values limited to a few records. Use the print function for more options.

Examples

```
data(plantGrowth)
plantGrowth
```

`stackTables` *Stack the tables of an object of class Dasst*

Description

`stackTables` stacks the tables of an object of class `Dasst`.

Usage

```
stackTables(object)
```

Arguments

`object`	Object of class `Dasst`.

Details

This function stacks the tables of an object of class `Dasst`. The result is given as a `data.frame`.

Value

A `data.frame` composed of the stacked tables.

Examples

```
data(plantGrowth)
nrow(plantGrowth[[1]])
nrow(plantGrowth[[2]])
plantgro12 <- stackTables(plantGrowth[1:2])
nrow(plantgro12)
```

summary *Summary method for class* Dasst

Description

summary summarizes the contents of an object of class Dasst.

Usage

```
## S3 method for class 'Dasst'
summary(object, ...)
```

Arguments

object	An object of class Dasst.
...	Arguments that may be passed to other functions.

Details

This method summarizes the contents of the object of object of class Dasst. After reading a DSSAT file, summary can give an idea of the volume of information stored in that file.

Value

An object of class summary.Dasst

Examples

```
data(plantGrowth)
summary(plantGrowth)
```

write.dssat *Write to a DSSAT-style file from an object of class Dasst*

Description

write.dssat writes to a file the contents of an object of class Dasst.

Usage

```
write.dssat(object, fnames)
```

Arguments

object	An object of class Dasst.
fnames	A character vector. The paths to the files where the contents of the object of class Dasst will be stored.

Details

This function writes to a file the contents of an object of class Dasst striving to maintain compatibility with the DSSAT-style format specifications.

The fnames vector specifies the paths to the files where data will be stored. Each table of the Dasst object may be saved in an individual file. If the length of fnames vector is shorter than the length of the object, then the paths will be recycled as necessary.

If paths contain file names that already exist, first the original files are saved appending a '.bak' extension. Then, the Dasst object is saved using these paths.

Examples

```
data(plantGrowth)
length(plantGrowth) <- 1
ffn <- paste(tempdir(), "PlantGro.OUT", sep="/")
write.dssat(plantGrowth, ffn)
```

[*"[" method for class Dasst*

Description

"[" gets a subset of an object of class Dasst.

Arguments

x	An object of class Dasst.
i	An integer or logical vector. This is the subset that will be retrieved from the whole object.

Details

This method gets a subset of an object of class Dasst. Shorter objects in the expression are recycled as often as need be until they match the length of the longest object.

Value

A new object of class Dasst that comprises the elements from the selected subset.

See Also

[<-

Examples

```
data(plantGrowth)
length(plantGrowth)
plantgro1 <- plantGrowth[1:10]
length(plantgro1)
class(plantgro1)

# Drop contents corresponding to selected orders
summary(plantGrowth)
plantgro2 <- plantGrowth[-1]
summary(plantgro2)
```

[<- *"[<-" method for class* Dasst

Description

"[<-" sets to a subset of an object of class Dasst an other object of the same class

Arguments

x	An object of class Dasst.
i	An integer or logical vector. This is the subset that will be updated from the whole object.
value	An object of class Dasst that will be stored at the given subset.

Details

This method sets to a subset of an object of class Dasst an other object of the same class. Shorter objects in the expression are recycled as often as need be until they match the length of the longest object.

Value

The actual object of class Dasst that comprises the elements updated from the selected subset.

See Also

[

Examples

```
# Replace position 1 with the contents of position 30.
data(plantGrowth)
plantGrowth[[1]][1:10, 1:15]
plantGrowth[1] <- plantGrowth[30]
plantGrowth[[1]][1:10, 1:15]
```

```
# Add a copy of the first order at the end extending the object length
rmax <- length(plantGrowth)
rmax
plantGrowth[rmax+1] <- plantGrowth[1]
length(plantGrowth)

# Copy position 2 into position 31, moving the former position 31 to the 32.
plantgro31 <- plantGrowth[31]
plantGrowth[31] <- plantGrowth[2]
plantGrowth[32] <- plantgro31
```

[[*"[[" method for class Dasst*

Description

"[[" gets the contents of a table from an object of class Dasst.

Arguments

x	An object of class Dasst.
i	An integer value. Position where values will be retrieved.

Details

This method gets the contents of the selected table stored in an object of class Dasst. Tables are internally stored and retrieved as data.frame. Rules for subset can be applied.

Value

The values retrieved from the table at position i as data.frame.

See Also

[[<-

Examples

```
data(plantGrowth)
class(plantGrowth[[1]])
plantGrowth[[1]]
plantGrowth[[1]][1:10,]
```

[[<- *"[[<-" method for class* Dasst

Description

"[[<-" sets the contents of a table from an object of class Dasst.

Arguments

x	An object of class Dasst.
i	An integer value. Position where values will be updated.
value	Any Values to be stored at the given position.

Details

This method sets the contents of the selected table stored in an object of class Dasst. Tables are internally stored and retrieved as data.frame. Rules for subset can be applied.

Value

The actual object of class Dasst.

See Also

[[

Examples

```
# Add a row of NA at the end of the table 1
data(plantGrowth)
rmax <- nrow(plantGrowth[[1]])
plantGrowth[[1]][rmax + 1, ] <- NA

# Edit a subset
plantGrowth[[1]][131:132,2:4]
plantGrowth[[1]][131:132,2:4] <- matrix(rep(100,6),nrow=2)
plantGrowth[[1]][131:132,2:4]

# Remove the last rows
# No need to subset left hand side. Dimension are automatically adjusted.
tail(plantGrowth[[1]])
plantGrowth[[1]] <- plantGrowth[[1]][c(-131,-132), ]
tail(plantGrowth[[1]])

# Column names are also valid
plantGrowth[[1]][129:130,"SNW1C"]
plantGrowth[[1]][129:130,"SNW1C"] <- 1100:1101
plantGrowth[[1]][129:130,"SNW1C"]
```

Index