read_datablock

reads data embedded in a block-comment in the edited or running script or in a given file

Syntax

```
data = read_datablock(n)
data = read_datablock(tag)

str = read_datablock(…, "-string")
str = read_datablock(…, "-string..")

… = read_datablock(filename, …)
```

Arguments

filename

single string: pathname of the text file the block-comment belongs to.

By default,

- the script being executed is considered.
- If no script is running, the currently edited script is considered. The script must be in an actual file, not some code not yet saved.

n

integer > 0: index of the block-comment to read in the file.

tag

single string to find in the title of the block-comment to select and read.

str

column of strings: the literal content of the block-comment, starting on the row after `/*`'s one, and ending on the row before `*/`'s one.

If the **`"-string.."`** option is used, any continuation mark `..` ending some lines are processed: full lines are built before returning.

data

actual data of the block-comment = result of `evstr(str)`.

Description

`read_datablock(…)` allows to work with any all-in-one script.sce including

- some Scilab's instructions, most often at the beginning of the script.sce file
- one or several multiline block-comments `/* ... */` containing blocks of data that you want to read and work with in the instructions.

This solution is very portable:

- Data do not need to be stored in any separate external files. Hence, the script.sce file can be distributed as is, without any external data to work. No need to build a zip archive bundling the script and data files together.

- There is no path of external data files to manage in the script. You don't even have to provide the path of your script.sce file to read the blocks content.

- Providing literal data in the script in a inline way is preferable only when data are short. With `read_datablock(…)`, you can separate clearly your code from the required data: For instance, put the bloc-comments embedding them after all instructions, at the end of the script.sce file. This avoids obfuscating the code with its input data.

- The script.sce file can be printed all together at once to document your work.

General rules:

- The actual content of a block-comment never includes the opening row containing `/*`, nor the closing row with `*/`.

- Each block can have a title written after the opening `/*` on the row. Any part of the title can be used as a tag to address the block.

- For syntaxes returning actual **`data`**, or with the `"-string.."` option, rows appended with some continuation marks `..` are processed: Full continuous lines are automatically rebuilt.
  ⚠ `..` must not be followed by any inline comment

- ⚠ In the script, any `/*` or `*/` sequences inside strings can make `read_datablock(…)`'s behavior unpredictable.

- A block-comment may include some passive `/*` sequences. Then they are normal parts of the content. Nested block-comments do not exist.

- When `read_datablock(…)` does not find any matching block-comment, `[]` is returned.

Extraction from a running script, or a script edited in Scinotes

`read_datablock(…)` will work when

- either it is called within lines of code selected in the script.sce currently edited in Scinotes ;
  ⚠ It won't work for a selection out of Scinotes, like in the console or in the examples below as only displayed in the help browser.
- or it is called from the whole script.sce being executed.

data = read_datablock(tag)

- reads in the executed or edited script the first block-comment whose title includes the given **`tag`** string,
- processes all continuation marks `..` of the literal content **`str`** of the block,
- returns the result of `data = evstr(str)`. `T` and `F` characters are interpreted as `%T` and `%F` values.

data = read_datablock(n) reads the n^{th} block-comment of the current script, processes its content as above, and returns actual **`data`**.

Addressing blocks according to a tag should be preferred. The order of block-comments in the file can be changed without jeopardizing their addressing.

str = read_datablock(tag, "-string") or **str = read_datablock(n, "-string")** returns the literal content **`str`** of the block-comment, as a column of strings. Leading and trailing spaces on lines are preserved. Continuation marks `..` are preserved as well.

str = read_datablock(tag, "-string..") or **str = read_datablock(n, "-string..")** returns the literal content **`str`** of the block-comment, as a column of strings, after having rebuilt continued rows.

Extraction from an external file

data = read_datablock(filename, n) and **data = read_datablock(filename, tag)** target and process the block-comment in the given file as `data = read_datablock(n)` and `data = read_datablock(tag)` do it for the current script.

The `"-string"` and `"-string.."` options work in the same way as when they are used for the current script.

Examples

Edit the following code in Scinotes ; save it as `read_datablock.sce` ; and execute the file.

```
v = read_datablock("booleans")  // T and F characters are converted into %T and %F values
disp(v, typeof(v))

v = read_datablock(3);  // comments and white rows are ignored
disp(v)

// evstr() can't convert the table in the block into actual data.
// We get its literal content and converts it afterward with msscanf():
v = read_datablock("Customers", "-string")
t = msscanf(-1, v, "%s %s %d %s")
disp(t(:,1), t(:,3))

// Continuation marks are supported:
v = read_datablock(2)
disp(v)

// Selection + exec:
// * Edit this set of examples in Scinote
// * Select one of the above read_datablock() instructions (with the mouse or keyboard)
// *¨Press CTRL + E to execute it
// ------------------------------------------------

/* Customers
John  SMITH     49  Canada
Elen  RACLIF    35  USA
Richard MICHEL  17  France
Katarina ORIANA 62  Brazil
*/

/* Misc data with continuation marks
 -4 -5 -6  1  9  ..
  9  6  2
  6 -1  7 ...
  1  2  7 -6 -2
*/

/*
  3  8  2 -2  0 -7  2  9   // first row

  8 -2  8 -8 -5 -2  7  6   // second row
  0 -2  7 -6 -5 -8  7 -9
*/

/* Some booleans
  F F T F F T F F F F
  F T F F T T T T F T
*/
```

```
--> exec('read_datablock.sce', -1)

  F  F  T  F  F  T  F  F  F  F
  F  T  F  F  T  T  T  T  F  T

  "boolean"

   3.    8.    2.  -2.    0.  -7.    2.    9.
   8.  -2.    8.  -8.  -5.  -2.    7.    6.
   0.  -2.    7.  -6.  -5.  -8.    7.  -9.

  "John  SMITH     49  Canada"
  "Elen  RACLIF    35  USA"
  "Richard MICHEL  17  France"
  "Katarina ORIANA 62  Brazil"

  "John"
  "Elen"
  "Richard"
  "Katarina"

   49.
   35.
   17.
   62.

  -4.  -5.  -6.    1.    9.    9.    6.    2.
   6.  -1.    7.    1.    2.    7.  -6.  -2.
```

From a given file:

```
[?, p] = libraryinfo("read_datablocklib");
File = fullfile(p, "read_datablock.sci")
scinotes(File, 11, "readonly")

read_datablock(File, 1, "-string")(1:3)

read_datablock(File, "booleans", "-string")
read_datablock(File, "booleans")

read_datablock(File, 3, "-string")
read_datablock(File, 3, "-string..")
read_datablock(File, 3)
```

```
--> File = fullfile(p, "read_datablock.sci")
 File  =
  "SCI\contrib\read_datablock\1.0\macros\read_datablock.sci"
--> scinotes(File, 11, "readonly")

--> read_datablock(File, 1, "-string")(1:3)
 ans  =
  "    read_datablock(n)"
  "    read_datablock(tag)"
  ""

--> read_datablock(File, "booleans", "-string")
 ans  =
  "  F  F  T  F  F  T  T"
  "  T  F  F  T  T  F  T"

--> read_datablock(File, "booleans")
 ans  =
  F  F  T  F  F  T  T
  T  F  F  T  T  F  T
```

```
--> read_datablock(File, 3, "-string")
 ans  =
  "  -9   9   5  -2  -7 ..."
  "  -3   4   5  -2  -4"
  ""
  "   7   4   5   3   4 .."
  "  -6  -2  -1  -7   6"

--> read_datablock(File, 3, "-string..")
 ans  =
  "  -9   9   5  -2  -7   -3   4   5  -2  -4"
  ""
  "   7   4   5   3   4   -6  -2  -1  -7   6"

--> read_datablock(File, 3)
 ans  =
  -9.   9.   5.  -2.  -7.  -3.   4.   5.  -2.  -4.
   7.   4.   5.   3.   4.  -6.  -2.  -1.  -7.   6.
```

See also

- comments
- mgetl
- evstr
- msscanf

Author

Samuel GOUGEON -- Le Mans Université

History

Version	Description
1.0	`2023-05-28: read_datablock()` introduced