

# Package: R2DT (via r-universe)

September 2, 2024

**Type** Package

**Title** Translation of Base R-Like Functions for 'data.table' Objects

**Version** 0.2

**Author** Robin Van Oirbeek

**Maintainer** Robin Van Oirbeek <robin.vanoirbeek@gmail.com>

**Description** Some heavily used base R functions are reconstructed to also be compliant to data.table objects. Also, some general helper functions that could be of interest for working with data.table objects are included.

**License** GPL-3

**Encoding** UTF-8

**Depends** R (>= 3.3.0)

**Imports** data.table (>= 1.10.4), plyr (>= 1.8.4), devFunc

**LazyData** true

**RoxygenNote** 6.0.1

**NeedsCompilation** no

**Date/Publication** 2020-03-19 15:10:25 UTC

**Repository** <https://robin-vo.r-universe.dev>

**RemoteUrl** <https://github.com/cran/R2DT>

**RemoteRef** HEAD

**RemoteSha** 1dc763e5bfe8fdf81d00dab047e0a936667b9d70

## Contents

asCharacterDT . . . . .	2
asFactorDT . . . . .	3
asIntegerDT . . . . .	4
asLogicalDT . . . . .	5
asNumericDT . . . . .	6
checkDT . . . . .	7

detectWeirdLevelNamesDT . . . . .	8
extractLevelDT . . . . .	8
extractRefLevelDT . . . . .	9
isCharacterDT . . . . .	10
isFactorDT . . . . .	11
isIntegerDT . . . . .	12
isLogicalDT . . . . .	13
isNumericDT . . . . .	14
lowFreqLevel2MissingDT . . . . .	15
rbindDT . . . . .	16
removeEmptyLevelsDT . . . . .	16
setRefLevelDT . . . . .	17
sortByRowIndexDT . . . . .	18

<b>Index</b>	<b>20</b>
--------------	-----------

---

asCharacterDT	<i>Forcing the character/string data type on a selected set of columns of a data.table object</i>
---------------	---

---

### Description

Forcing the character/string data type on a selected set of columns of a data.table object

### Usage

```
asCharacterDT(inputDT, colNamesToBeTransformed = NULL)
```

### Arguments

inputDT	data.table object containing the data of interest. This is an obligatory argument, without default value.
colNamesToBeTransformed	Character vector containing potential column names of the 'inputDT' argument. The default value is NULL.

### Value

No value is returned. Note that a valid value needs to be supplied to the 'colNamesToBeTransformed' argument in order to make this function work.

### Examples

```
library(data.table)
inputDT <- as.data.table(data.frame(x = seq(1, 20, 2), y = LETTERS[1:10]))
asFactorDT(inputDT, c('y'))

asCharacterDT(inputDT)
asCharacterDT(inputDT, c('x', 'y'))
```

```
# First looking at the result, followed by testing if the transformation worked!

inputDT
isCharacterDT(inputDT, c('x', 'y'))
isFactorDT(inputDT, c('x', 'y'))
```

---

asFactorDT	<i>Forcing the character/string data type on a selected set of columns of a data.table object</i>
------------	---

---

### Description

Forcing the character/string data type on a selected set of columns of a data.table object

### Usage

```
asFactorDT(inputDT, colNamesToBeTransformed = NULL)
```

### Arguments

inputDT	data.table object containing the data of interest. This is an obligatory argument, without default value.
colNamesToBeTransformed	Character vector containing potential column names of the 'inputDT' argument. The default value is NULL.

### Value

No value is returned. Note that a valid value needs to be supplied to the 'colNamesToBeTransformed' argument in order to make this function work.

### Examples

```
library(data.table)
inputDT <- as.data.table(data.frame(x = seq(1, 20, 2), y = LETTERS[1:10]))
asFactorDT(inputDT, c('y'))

asCharacterDT(inputDT)
asCharacterDT(inputDT, c('x', 'y'))

# First looking at the result, followed by testing if the transformation worked!

inputDT
isCharacterDT(inputDT, c('x', 'y'))
isFactorDT(inputDT, c('x', 'y'))
```

---

asIntegerDT	<i>Forcing the integer data type on a selected set of columns of a data.table object</i>
-------------	--

---

### Description

Forcing the integer data type on a selected set of columns of a data.table object

### Usage

```
asIntegerDT(inputDT, colNamesToBeTransformed = NULL)
```

### Arguments

inputDT	data.table object containing the data of interest. This is an obligatory argument, without default value.
colNamesToBeTransformed	Character vector containing potential column names of the 'inputDT' argument. The default value is NULL.

### Value

No value is returned. Note that a valid value needs to be supplied to the 'colNamesToBeTransformed' argument in order to make this function work.

### Examples

```
library(data.table)
inputDT <- as.data.table(data.frame(x = seq(1, 20, 2), y = LETTERS[1:10]))
asFactorDT(inputDT, c('y'))

asIntegerDT(inputDT)
asIntegerDT(inputDT, c('x', 'y'))

# First looking at the result, followed by testing if the transformation worked!

inputDT
isIntegerDT(inputDT, c('x', 'y'))

# Note the following behavior that also holds for the as.integer() base R function.
isNumericDT(inputDT, c('x', 'y'))
```

---

asLogicalDT	<i>Forcing the logical/boolean data type on a selected set of columns of a data.table object</i>
-------------	--

---

### Description

Forcing the logical/boolean data type on a selected set of columns of a data.table object

### Usage

```
asLogicalDT(inputDT, colNamesToBeTransformed = NULL)
```

### Arguments

inputDT	data.table object containing the data of interest. This is an obligatory argument, without default value.
colNamesToBeTransformed	Character vector containing potential column names of the 'inputDT' argument. The default value is NULL.

### Value

No value is returned. Note that a valid value needs to be supplied to the 'colNamesToBeTransformed' argument in order to make this function work.

### Examples

```
library(data.table)
inputDT <- as.data.table(data.frame(x = seq(1, 20, 2), y = LETTERS[1:10]))
asFactorDT(inputDT, c('y'))

asLogicalDT(inputDT)
asLogicalDT(inputDT, c('x', 'y'))

# First looking at the result, followed by testing if the transformation worked!

inputDT
isLogicalDT(inputDT, c('x', 'y'))

# Notice the 'funny' side effect for the 'F' character value of column y...
# This behavior is also observed for the as.logical() base R function.
```

---

asNumericDT	<i>Forcing the numeric data type on a selected set of columns of a data.table object</i>
-------------	--

---

### Description

Forcing the numeric data type on a selected set of columns of a data.table object

### Usage

```
asNumericDT(inputDT, colNamesToBeTransformed = NULL)
```

### Arguments

inputDT            data.table object containing the data of interest. This is an obligatory argument, without default value.

colNamesToBeTransformed  
Character vector containing potential column names of the 'inputDT' argument. The default value is NULL.

### Value

No value is returned. Note that a valid value needs to be supplied to the 'colNamesToBeTransformed' argument in order to make this function work.

### Examples

```
library(data.table)
inputDT <- as.data.table(data.frame(x = seq(1, 20, 2), y = LETTERS[1:10]))

asNumericDT(inputDT)
asNumericDT(inputDT, c('x', 'y'))

# First looking at the result, followed by testing if the transformation worked!

inputDT
isNumericDT(inputDT, c('x', 'y'))
isIntegerDT(inputDT, c('x', 'y'))
```

---

checkDT	<i>Checking if an object is a data.table object and (optional) testing if some column names are valid for it</i>
---------	--

---

### Description

Checking if an object is a data.table object and (optional) testing if some column names are valid for it

### Usage

```
checkDT(inputDT, colNamesToBeChecked = NULL)
```

### Arguments

inputDT data.table object containing the data of interest. This is an obligatory argument, without default value.

colNamesToBeChecked Character vector containing potential column names of the 'inputDT' argument. The default value is NULL.

### Value

No value is returned if all elements in the 'colNamesToBeChecked' argument, are valid column names of the 'inputDT' argument. In the absence of a value for the 'colNamesToBeChecked' argument, it is only tested if the 'inputDT' argument is a data.table object (is tested irrespective of the value for the 'colNamesToBeChecked' argument).

### Examples

```
library(data.table)
inputDT <- as.data.table(data.frame(x = seq(1, 20, 2), y = LETTERS[1:10]))
asFactorDT(inputDT, c('y'))

checkDT(inputDT)
checkDT(inputDT, c('x', 'y'))

checkDT(inputDT, c('x', 'y1'))
checkDT(inputDT, c('x', 'y1', 'z1'))
checkDT(inputDT, c('x1', 'y1', 'z1'))
```

---

detectWeirdLevelNamesDT

*Detecting which levels of which factor of a data.table object contain non-alpha numeric characters (including whitespace) characters*

---

### Description

Detecting which levels of which factor of a data.table object contain non-alpha numeric characters (including whitespace) characters

### Usage

```
detectWeirdLevelNamesDT(inputDT)
```

### Arguments

inputDT            data.table object containing the data of interest. This is an obligatory argument, without default value.

### Value

No value is returned. Note that a valid value needs to be supplied to the 'colNamesToBeChecked' argument in order to make this function work.

### Examples

```
library(data.table)
inputDT <- as.data.table(data.frame(x = seq(1, 20, 2)))
detectWeirdLevelNamesDT(inputDT)

inputDT <- as.data.table(data.frame(x = seq(1, 20, 2), y = LETTERS[1:10]))
asFactorDT(inputDT, c('y'))
detectWeirdLevelNamesDT(inputDT)

inputDT <- as.data.table(data.frame(x = c(rep('test_', 5), rep('test@', 5)),
y = c(rep('test_', 5), rep('test@', 5))))
asFactorDT(inputDT, c('x', 'y'))
detectWeirdLevelNamesDT(inputDT)
```

---

extractLevelDT

*Extracting the levels of all or a selected set of the factor columns of a data.table object*

---

### Description

Extracting the levels of all or a selected set of the factor columns of a data.table object



**Usage**

```
extractLevelDT(inputDT, categoricalVar = NULL)
```

**Arguments**

`inputDT` data.table object containing the data of interest. This is an obligatory argument, without default value.

`categoricalVar` Character vector containing potential column names of the 'inputDT' argument. The default value is NULL.

**Value**

A named list is returned, with as names the different valid factor column names, either of the whole 'inputDT' argument, either of the factor variables of which the names are listed in 'categoricalVar' argument, containing a character vector with the different levels of the respective factor. In case that the 'categoricalVar' argument contains column names that aren't factors, a warning is thrown. An empty list is returned when no valid factors (with or without the 'categoricalVar' selection turned on) are found.

**Examples**

```
library(data.table)
inputDT <- as.data.table(data.frame(x = LETTERS[11:20], y = LETTERS[1:10]))
asFactorDT(inputDT, c('x', 'y'))

extractLevelDT(inputDT)
extractLevelDT(inputDT, c('x', 'y'))
extractLevelDT(inputDT, c('x', 'y1'))

inputDT <- as.data.table(data.frame(x = seq(1, 20, 2), y = LETTERS[1:10]))
asFactorDT(inputDT, c('y'))
extractLevelDT(inputDT)

inputDT <- as.data.table(data.frame(x = seq(1, 20, 2), y = seq(2, 21, 2)))
extractLevelDT(inputDT)
extractLevelDT(inputDT, c('x', 'y'))
```

---

extractRefLevelDT	<i>Extracting the reference level of all or a selected set of the factor columns of a data.table object</i>
-------------------	---

---

**Description**

Extracting the reference level of all or a selected set of the factor columns of a data.table object

**Usage**

```
extractRefLevelDT(inputDT, categoricalVar = NULL)
```

**Arguments**

- `inputDT` `data.table` object containing the data of interest. This is an obligatory argument, without default value.
- `categoricalVar` Character vector containing potential column names of the 'inputDT' argument. The default value is NULL.

**Value**

A named list is returned, with as names the different valid factor column names, either of the whole 'inputDT' argument, either of the factor variables of which the names are listed in 'categoricalVar' argument, containing a character vector of length 1 with the reference level of the respective factor. In case that the 'categoricalVar' argument contains column names that aren't factors, a warning is thrown. An empty is list is returned when no valid factors (with or without the 'categoricalVar' selection turned on) are found.

**Examples**

```
library(data.table)
inputDT <- as.data.table(data.frame(x = LETTERS[11:20], y = LETTERS[1:10]))
asFactorDT(inputDT, c('x', 'y'))

extractRefLevelDT(inputDT)
extractRefLevelDT(inputDT, c('x', 'y'))
extractRefLevelDT(inputDT, c('x', 'y1'))

inputDT <- as.data.table(data.frame(x = seq(1, 20, 2), y = LETTERS[1:10]))
asFactorDT(inputDT, c('y'))
extractRefLevelDT(inputDT)

inputDT <- as.data.table(data.frame(x = seq(1, 20, 2), y = seq(2, 21, 2)))
extractRefLevelDT(inputDT)
extractRefLevelDT(inputDT, c('x', 'y'))
```

---

<code>isCharacterDT</code>	<i>Testing if a set of columns of a data.table object corresponds to the character/string data type</i>
----------------------------	---

---

**Description**

Testing if a set of columns of a `data.table` object corresponds to the character/string data type

**Usage**

```
isCharacterDT(inputDT, colNamesToBeChecked = NULL, returnNames = FALSE)
```

**Arguments**

inputDT	data.table object containing the data of interest. This is an obligatory argument, without default value.
colNamesToBeChecked	Character vector containing potential column names of the 'inputDT' argument. The default value is NULL.
returnNames	Logical vector of length 1 indicating whether or not the column name of the selected strings should be returned. The default value is FALSE.

**Value**

A logical vector of length the size of the 'colNamesToBeChecked' argument, or in the absence of a value the number of columns of the 'inputDT' argument, that is TRUE if the corresponding column of the 'inputDT' argument is a string. If the 'returnNames' argument equals TRUE, then only those column names from the aforementioned selection of column of the 'inputDT' argument are returned that is a string.

**Examples**

```
library(data.table)
inputDT <- as.data.table(data.frame(x = rep(c(TRUE, FALSE), 5), y = LETTERS[1:10]))
asFactorDT(inputDT, c('y'))

isCharacterDT(inputDT)

inputDT2 <- as.data.table(data.frame(y = LETTERS[1:10]))

isCharacterDT(inputDT2)
isCharacterDT(inputDT2, c('x', 'y'))
isCharacterDT(inputDT2, returnNames = TRUE)
```

---

isFactorDT	<i>Testing if a set of columns of a data.table object corresponds to the factor data type</i>
------------	---

---

**Description**

Testing if a set of columns of a data.table object corresponds to the factor data type

**Usage**

```
isFactorDT(inputDT, colNamesToBeChecked = NULL, returnNames = FALSE)
```

**Arguments**

inputDT	data.table object containing the data of interest. This is an obligatory argument, without default value.
colNamesToBeChecked	Character vector containing potential column names of the 'inputDT' argument. The default value is NULL.
returnNames	Logical vector of length 1 indicating whether or not the column name of the selected factors should be returned. The default value is FALSE.

**Value**

A logical vector of length the size of the 'colNamesToBeChecked' argument, or in the absence of a value the number of columns of the 'inputDT' argument, that is TRUE if the corresponding column of the 'inputDT' argument is a factor. If the 'returnNames' argument equals TRUE, then only those column names from the aforementioned selection of column of the 'inputDT' argument are returned that are a factor.

**Examples**

```
library(data.table)
inputDT <- as.data.table(data.frame(x = seq(1, 20, 2), y = LETTERS[1:10]))
asFactorDT(inputDT, c('y'))

isFactorDT(inputDT)
isFactorDT(inputDT, c('x', 'y'))
isFactorDT(inputDT, returnNames = TRUE)

isFactorDT(inputDT, 'y')
isFactorDT(inputDT, c('x', 'y1'))
```

---

isIntegerDT	<i>Testing if a set of columns of a data.table object corresponds to the integer data type</i>
-------------	--

---

**Description**

Testing if a set of columns of a data.table object corresponds to the integer data type

**Usage**

```
isIntegerDT(inputDT, colNamesToBeChecked = NULL, returnNames = FALSE)
```

**Arguments**

inputDT	data.table object containing the data of interest. This is an obligatory argument, without default value.
---------	---

colNamesToBeChecked	Character vector containing potential column names of the 'inputDT' argument. The default value is NULL.
returnNames	Logical vector of length 1 indicating whether or not the column name of the selected integers should be returned. The default value is FALSE.

**Value**

A logical vector of length the size of the 'colNamesToBeChecked' argument, or in the absence of a value the number of columns of the 'inputDT' argument, that is TRUE if the corresponding column of the 'inputDT' argument is an integer. If the 'returnNames' argument equals TRUE, then only those column names from the aforementioned selection of column of the 'inputDT' argument are returned that are an integer.

**Examples**

```
library(data.table)
inputDT <- as.data.table(data.frame(x = seq(1L, 20L, 2L), y = LETTERS[1:10]))
asFactorDT(inputDT, c('y'))

isIntegerDT(inputDT)
isIntegerDT(inputDT, c('x', 'y'))
isIntegerDT(inputDT, returnNames = TRUE)

isIntegerDT(inputDT, 'x')
isIntegerDT(inputDT, c('x', 'y1'))
```

---

isLogicalDT	<i>Testing if a set of columns of a data.table object corresponds to the logical/boolean data type</i>
-------------	--

---

**Description**

Testing if a set of columns of a data.table object corresponds to the logical/boolean data type

**Usage**

```
isLogicalDT(inputDT, colNamesToBeChecked = NULL, returnNames = FALSE)
```

**Arguments**

inputDT	data.table object containing the data of interest. This is an obligatory argument, without default value.
colNamesToBeChecked	Character vector containing potential column names of the 'inputDT' argument. The default value is NULL.
returnNames	Logical vector of length 1 indicating whether or not the column name of the selected booleans should be returned. The default value is FALSE.

**Value**

A logical vector of length the size of the 'colNamesToBeChecked' argument, or in the absence of a value the number of columns of the 'inputDT' argument, that is TRUE if the corresponding column of the 'inputDT' argument is a boolean. If the 'returnNames' argument equals TRUE, then only those column names from the aforementioned selection of column of the 'inputDT' argument are returned that are a boolean.

**Examples**

```
library(data.table)
inputDT <- as.data.table(data.frame(x = rep(c(TRUE, FALSE), 5), y = LETTERS[1:10]))
asFactorDT(inputDT, c('y'))

isLogicalDT(inputDT)
isLogicalDT(inputDT, c('x', 'y'))
isLogicalDT(inputDT, returnNames = TRUE)

isLogicalDT(inputDT, 'x')
isLogicalDT(inputDT, c('x', 'y1'))
```

---

isNumericDT	<i>Testing if a set of columns of a data.table object corresponds to the numeric data type</i>
-------------	--

---

**Description**

Testing if a set of columns of a data.table object corresponds to the numeric data type

**Usage**

```
isNumericDT(inputDT, colNamesToBeChecked = NULL, returnNames = FALSE)
```

**Arguments**

inputDT	data.table object containing the data of interest. This is an obligatory argument, without default value.
colNamesToBeChecked	Character vector containing potential column names of the 'inputDT' argument. The default value is NULL.
returnNames	Logical vector of length 1 indicating whether or not the column name of the selected numerics should be returned. The default value is FALSE.

**Value**

A logical vector of length the size of the 'colNamesToBeChecked' argument, or in the absence of a value the number of columns of the 'inputDT' argument, that is TRUE if the corresponding column of the 'inputDT' argument is a numeric. If the 'returnNames' argument equals TRUE, then only those column names from the aforementioned selection of column of the 'inputDT' argument are returned that are a numeric.

**Examples**

```
library(data.table)
inputDT <- as.data.table(data.frame(x = seq(1, 20, 2), y = LETTERS[1:10]))
asFactorDT(inputDT, c('y'))

isNumericDT(inputDT)
isNumericDT(inputDT, c('x', 'y'))
isNumericDT(inputDT, returnNames = TRUE)

isNumericDT(inputDT, 'x')
isNumericDT(inputDT, c('x', 'y1'))
```

---

lowFreqLevel2MissingDT

*Transform levels of all the factor columns of a data.table object to missing if too little observations pertain to a given level of it.*

---

**Description**

Transform levels of all the factor columns of a data.table object to missing if too little observations pertain to a given level of it.

**Usage**

```
lowFreqLevel2MissingDT(inputDT, minNumberLevel = NULL)
```

**Arguments**

inputDT	data.table object containing the data of interest. This is an obligatory argument, without default value.
minNumberLevel	Numeric vector of length 1 that indicates the minimal number of observations of a given level that should be observed to avoid that that level will be deleted from the list of possible levels for that factor and the value of its observations will be turned into missing values.

**Value**

No value is returned. The level that was not underpopulated is also removed from the levels of the respective categorical variable.

**Examples**

```
library(data.table)
inputDT <- as.data.table(data.frame(x = seq(1, 20, 2), y = LETTERS[1:10]))
asFactorDT(inputDT, c('y'))
levels(inputDT$y)
lowFreqLevel2MissingDT(inputDT, 2)
levels(inputDT$y)
```

```
inputDT <- as.data.table(data.frame(x = seq(1, 40, 2),
y = c(LETTERS[1:10], LETTERS[1:10])))
asFactorDT(inputDT, c('y'))
levels(inputDT$y)
lowFreqLevel2MissingDT(inputDT, 1)
levels(inputDT$y)
```

---

rbindDT	<i>Glueing, not merging, two data.table objects together, by matching column names</i>
---------	--

---

### Description

Glueing, not merging, two data.table objects together, by matching column names

### Usage

```
rbindDT(topDT, bottomDT)
```

### Arguments

topDT	data.table object 1. Its values will be placed at the top of the returned data.table object. This is an obligatory argument, without default value.
bottomDT	data.table object 2. Its values will be placed at the bottom of the returned data.table object. This is an obligatory argument, without default value.

### Value

The glued data.table object. Matching column names of 'topDT' and 'bottomDT' will be identified and its values will be placed in one column in the returned data.table object, the values of the 'topDT' argument on top of the values of the 'bottomDT' argument. Non-matching columns will be have missing values for the rows in the returned data.table object that correspond to the input data.table object in which the column name was not found.

---

removeEmptyLevelsDT	<i>Remove empty levels from all the factor columns of a data.table object</i>
---------------------	---

---

### Description

Remove empty levels from all the factor columns of a data.table object

### Usage

```
removeEmptyLevelsDT(inputDT)
```



**Arguments**

`inputDT` `data.table` object containing the data of interest. This is an obligatory argument, without default value.

**Value**

No value is returned.

**Examples**

```
library(data.table)
inputDT <- as.data.table(data.frame(x = seq(1, 20, 2), y = LETTERS[1:10]))
asFactorDT(inputDT, c('y'))
levels(inputDT$y)
removeEmptyLevelsDT(inputDT)
levels(inputDT$y)
removeEmptyLevelsDT(inputDT[x < 10])
levels(inputDT$y)

# You need to define a new data.table object
# in order to make the 'removeEmptyLevelsDT' function work.
reducedDT <- inputDT[x < 10]
levels(reducedDT$y)
removeEmptyLevelsDT(reducedDT)
levels(reducedDT$y)
```

---

setRefLevelDT	<i>Setting the reference level of all or a selected set of the factor columns of a data.table object</i>
---------------	--

---

**Description**

Setting the reference level of all or a selected set of the factor columns of a `data.table` object

**Usage**

```
setRefLevelDT(inputDT, categoricalVar, referenceLevel)
```

**Arguments**

`inputDT` `data.table` object containing the data of interest. This is an obligatory argument, without default value.

`categoricalVar` Character vector containing potential column names of the 'inputDT' argument. This is an obligatory argument, without default value.

`referenceLevel` Character vector containing the new reference levels. This is an obligatory argument, without default value.

**Value**

No value is returned. Note that the 'categoricalVar' and 'referenceLevel' should match up, meaning that they should be of the same length and the ith element should refer to the same variable.

**Examples**

```
library(data.table)
inputDT <- as.data.table(data.frame(x = LETTERS[11:20], y = LETTERS[1:10]))
asFactorDT(inputDT, c('x', 'y'))

setRefLevelDT(inputDT)

levels(inputDT$x)[1]
levels(inputDT$y)[1]
setRefLevelDT(inputDT, c('x', 'y'), c('L', 'C'))
levels(inputDT$x)[1]
levels(inputDT$y)[1]

setRefLevelDT(inputDT, c('x', 'y'), c('bla', 'bla'))

inputDT <- as.data.table(data.frame(x = seq(1, 20, 2), y = LETTERS[1:10]))
asFactorDT(inputDT, c('y'))
levels(inputDT$y)[1]
setRefLevelDT(inputDT, 'y', 'E')
levels(inputDT$y)[1]
```

---

sortByRowIndexDT	<i>Order the rows of a data.table object by index</i>
------------------	---

---

**Description**

Order the rows of a data.table object by index

**Usage**

```
sortByRowIndexDT(inputDT, rowIndices)
```

**Arguments**

inputDT	data.table object containing the data of interest. This is an obligatory argument, without default value.
rowIndices	Integer vector that contains the row indices according to which the 'inputDT' object should be ordered. This is an obligatory argument, without default value.

**Value**

The 'inputDT' data.table object, ordered according to the 'rowIndices' argument. This function assumes that the length of the 'rowIndices' argument is correspond to the number of rows of the 'inputDT' argument. If the length of the 'rowIndices' argument is smaller than the number of rows of the 'inputDT' argument, the values of the 'rowIndices' argument are recycled until the as many indices as number of rows of the 'inputDT' argument is obtained.

**Examples**

```
library(data.table)
inputDT <- as.data.table(data.frame(x = 10:1, y = LETTERS[1:10]))
asFactorDT(inputDT, c('y'))
inputDT
sortByRowIndexDT(inputDT, 10:1)
inputDT
```

# Index

asCharacterDT, [2](#)  
asFactorDT, [3](#)  
asIntegerDT, [4](#)  
asLogicalDT, [5](#)  
asNumericDT, [6](#)

checkDT, [7](#)

detectWeirdLevelNamesDT, [8](#)

extractLevelDT, [8](#)  
extractRefLevelDT, [9](#)

isCharacterDT, [10](#)  
isFactorDT, [11](#)  
isIntegerDT, [12](#)  
isLogicalDT, [13](#)  
isNumericDT, [14](#)

lowFreqLevel2MissingDT, [15](#)

rbindDT, [16](#)  
removeEmptyLevelsDT, [16](#)

setRefLevelDT, [17](#)  
sortByRowIndexDT, [18](#)