

Package ‘rjson’

July 23, 2025

Version 0.2.23

Title JSON for R

Author Alex Couture-Beil [aut, cre]

Maintainer Alex Couture-Beil <rjson_pkg@mofo.ca>

Depends R (>= 4.0.0)

Description Converts R object into JSON objects and vice-versa.

URL <https://github.com/alexcb/rjson>

License GPL-2

Repository CRAN

NeedsCompilation yes

Date/Publication 2024-09-16 18:20:02 UTC

Contents

fromJSON	1
newJSONParser	2
rjson	4
toJSON	4

Index	6
--------------	----------

fromJSON	<i>Convert JSON To R</i>
----------	--------------------------

Description

Convert a JSON object into an R object.

Usage

```
fromJSON( json_str, file, method = "C", unexpected.escape = "error", simplify = TRUE )
```

Arguments

<code>json_str</code>	a JSON object to convert
<code>file</code>	the name of a file to read the <code>json_str</code> from; this can also be a URL. Only one of <code>json_str</code> or <code>file</code> must be supplied.
<code>method</code>	use the C implementation, or the older slower (and one day to be deprecated) R implementation
<code>unexpected.escape</code>	changed handling of unexpected escaped characters. Handling value should be one of "error", "skip", or "keep"; on unexpected characters issue an error, skip the character, or keep the character
<code>simplify</code>	If TRUE, attempt to convert json-encoded lists into vectors where appropriate. If FALSE, all json-encoded lists will be wrapped in a list even if they are all of the same data type.

Value

R object that corresponds to the JSON object

See Also

[toJSON](#)

Examples

```
fromJSON('[1,2,3]', simplify=TRUE)
# returns c(1,2,3)
fromJSON('[1,2,3]', simplify=FALSE)
# returns list(1,2,3)

#As a result, this will output "1"
toJSON(fromJSON('[1]', simplify=TRUE))
#Compared with this which will output "[1]" as expected
toJSON(fromJSON('[1]', simplify=FALSE))

#R vs C execution time
x <- toJSON( iris )
system.time( y <- fromJSON(x) )
system.time( y2 <- fromJSON(x,method = "R") )
```

newJSONParser

Convert buffered JSON objects To R

Description

Convert a collection of JSON objects into R objects.

Usage

```
newJSONParser(method = "R")
```

Arguments

method use the C implementation, or the slower original R implementation

Value

A list of functions used for parsing objects

See Also

[toJSON](#)

Examples

```
sample_json <- '{
{
"breakfast" : [ "milk", "fruit loops", "juice" ],
"lunch" : [ "left over sushi" ]
}
}'

parser <- newJSONParser()

parser$addData( sample_json )
food <- parser$getObject()
print( food )

#This is equivalent to using FromJSON( sample_json )
#However, sample_json can be split into several parts:

### EXAMPLE 2:

part_1 <- '{ "breakfast" : [ "milk", "fruit loops", "juice" ], '
part_2 <- '"lunch" : [ "left over sushi" ]'
# close off the first object, and create a 2nd JSON object, which is simply an
# array
part_3 <- '}' [1,2,3,4,5]'

parser <- newJSONParser()
parser$addData( part_1 )
parser$getObject() #returns NULL - since part_1 isn't complete
parser$addData( part_2 )
parser$getObject() #returns NULL - since part_2 still isn't complete
parser$addData( part_3 )
parser$getObject() #returns the first food object
parser$getObject() #returns the second array
```

`rjson`*JSON for R*

Description

JSON (JavaScript Object Notation) is a lightweight data-interchange format. This package converts JSON objects into R objects and vice-versa. See json.org for an overview of JSON. Unicode is unfortunately not supported at the moment.

`toJSON`*Convert R To JSON*

Description

Convert an R object into a corresponding JSON object.

Lists with unnamed components are not currently supported

Usage

```
toJSON( x, indent=0, method="C" )
```

Arguments

<code>x</code>	a vector or list to convert into a JSON object
<code>indent</code>	an integer specifying how much indentation to use when formatting the JSON object; if 0, no pretty-formatting is used
<code>method</code>	use the C implementation, or the older slower (and one day to be deprecated) R implementation

Value

a string containing the JSON object

See Also

[fromJSON](#)

Examples

```
x <- list( alpha = 1:5, beta = "Bravo",
           gamma = list(a=1:3, b=NULL),
           delta = c(TRUE, FALSE) )
json <- toJSON( x )
fromJSON( json )
```

```
#named vectors are treated as JSON objects (lists)
toJSON(islands[1:4])
```

```
#data.frames must be converted into a list before converting into JSON
plot(cars, pch=2)
json_cars <- toJSON(as.list(cars))
points( data.frame( fromJSON( json_cars ) ), col="red", pch=3 )
```

```
#special R types are encoded as strings
testString <- c(1,2,3,4,NA,NaN,Inf,8,9);
toJSON(testString);
```

Index

- * **interface**

- fromJSON, [1](#)

- newJSONParser, [2](#)

- toJSON, [4](#)

- * **package**

- rjson, [4](#)

fromJSON, [1](#), [4](#)

newJSONParser, [2](#)

rjson, [4](#)

rjson-package (rjson), [4](#)

toJSON, [2](#), [3](#), [4](#)