

Package: ajv (via r-universe)

August 25, 2024

Title Another JSON Schema Validator

Version 1.0.0

Maintainer Jason Thorpe <jdthorpe@gmail.com>

Description A thin wrapper around the 'ajv' JSON validation package for JavaScript. See <<http://epoberezkin.github.io/ajv/>> for details.

License GPL-2

LazyData true

URL <https://github.com/jdthorpe/ajvr>

BugReports <https://github.com/jdthorpe/ajvr/issues>

Imports V8, yaml, RJSONIO

Suggests knitr, rmarkdown, testthat

RoxygenNote 5.0.1

VignetteBuilder knitr

Repository <https://jdthorpe.r-universe.dev>

RemoteUrl <https://github.com/jdthorpe/ajvr>

RemoteRef HEAD

RemoteSha 754e5e882d80bccb8678c9fde861361459a5d022

Contents

Ajv	2
ajv.addFormat	2
ajv.addKeyword	3
ajv.addSchema	4
ajv.compile	5
ajv.errorsText	5
ajv.getSchema	6
ajv.keyword	6
ajv.removeSchema	7
ajv.validate	8
ajv.validateSchema	8

Index **10**

<code>Ajv</code>	<i>Create an Ajv instnace.</i>
------------------	--------------------------------

Description

Create an Ajv instnace (the equivalent of calling `new Ajv()` in javascript)

Usage

```
Ajv(options = NULL)
```

Arguments

<code>options</code>	Optional; see the ajv github page for details
----------------------	---

<code>ajv.addFormat</code>	<i>A wrapper for the Ajv.addFormat method</i>
----------------------------	---

Description

Add a string format to an Ajv instance.

Usage

```
ajv.addFormat(this, key, format)
```

Arguments

<code>this</code>	An AJV instance, provided implicitly when called via <code>my_instance\$addFormat(...)</code>
<code>key</code>	String; the name with format to add.
<code>format</code>	the format to be added. Note that JavaScript object literals should be enclosed in a call to <code>JS</code> . (i.e. <code>ajv\$addFormat("numbers", JS("/^\d+\$/"))</code>)

Value

```
invisible(NULL)
```

See Also

Other `AJV.Instance.Methods`: [ajv.addKeyword](#), [ajv.addSchema](#), [ajv.errorsText](#), [ajv.keyword](#), [ajv.removeSchema](#), [ajv.validateSchema](#), [ajv.validate](#)

Examples

```
## Not run:
my_ajv_instance = Ajv()
my_ajv_instance$keyword(key,object)

## End(Not run)
```

ajv.addKeyword	<i>A wrapper for the Ajv.addKeyword method</i>
----------------	--

Description

The add a schema to an Ajv instance

Usage

```
ajv.addKeyword(this, name, definition)
```

Arguments

this	An AJV instance, provided implicitly when called via my_instance\$addSchema(...)
name	The name of the keyword to be added.
definition	A string encoding of a javascript object to be used as to define the keyword.

Value

```
invisible(NULL)
```

See Also

Other `AJV.Instance.Methods`: [ajv.addFormat](#), [ajv.addSchema](#), [ajv.errorsText](#), [ajv.keyword](#), [ajv.removeSchema](#), [ajv.validateSchema](#), [ajv.validate](#)

Examples

```
## Not run:
my_ajv_instance = Ajv()
my_ajv_instance$addSchema

## End(Not run)
```

ajv.addSchema	<i>A wrapper for the Ajv.addSchema method</i>
---------------	---

Description

The add a schema to an Ajv instance

Usage

```
ajv.addSchema(this, schema, key)
```

Arguments

this	An AJV instance, provided implicitly when called via my_instance\$addSchema(...)
schema	The schema to be added. schema may be a valid JSON string, an R object (i.e. 'list(...)'), a connection to a JSON file, or the name of a JSON or YAML file. YAML files are parsed via [js-yaml](https://www.npmjs.com/package/js-yaml)'s 'safeLoad()' method.
key	String; the name with which to store the schema

Value

invisible(NULL)

See Also

Other `AJV.Instance.Methods`: [ajv.addFormat](#), [ajv.addKeyword](#), [ajv.errorsText](#), [ajv.keyword](#), [ajv.removeSchema](#), [ajv.validateSchema](#), [ajv.validate](#)

Examples

```
## Not run:  
my_ajv_instance = Ajv()  
my_ajv_instance$addSchema  
  
## End(Not run)
```

ajv.compile	<i>The Ajv.compile method</i>
-------------	-------------------------------

Description

Create an Ajv validator function from a schema

Usage

```
ajv.compile(this, schema)
```

Arguments

this	An AJV instance, provided implicitly when called via <code>my_instance\$compile(...)</code>
schema	The Schema with which to validate the data. <code>schema</code> may be a valid JSON string, an R object (i.e. <code>'list(...)</code> '), a connection to a JSON file, or the name of a JSON or YAML file. YAML files are parsed via <code>[js-yaml](https://www.npmjs.com/package/js-yaml)</code> 's <code>'safeLoad()'</code> method.

Value

an AJV validation function

ajv.errorsText	<i>A wrapper for the Ajv.errorsText method</i>
----------------	--

Description

Extracts the errors object from

Usage

```
ajv.errorsText(this)
```

Arguments

this	An AJV instance, provided implicitly when called via <code>my_instance\$errorsText(...)</code>
------	--

Value

JSON encoded object containing the error message (if any), with class "AJV-errorsText" for pretty printing via `print.errorsText`

See Also

Other `AJV.Instance.Methods`: [ajv.addFormat](#), [ajv.addKeyword](#), [ajv.addSchema](#), [ajv.keyword](#), [ajv.removeSchema](#), [ajv.validateSchema](#), [ajv.validate](#)

Examples

```
## Not run:
my_ajv_instance = Ajv()
my_ajv_instance$errorsText

## End(Not run)
```

ajv.getSchema	<i>The Ajv.compile method</i>
---------------	-------------------------------

Description

Create an Ajv validator function from a schema

Usage

```
ajv.getSchema(this, key)
```

Arguments

this	An AJV instance, provided implicitly when called via my_instance\$getSchema(...)
key	String; the name of the schema to fetch from the Ajv instanced.

Value

an AJV validation function

ajv.keyword	<i>A wrapper for the Ajv.addFormat method</i>
-------------	---

Description

Add a string format to an Ajv instance.

Usage

```
ajv.keyword(this, key, object)
```

Arguments

this	An AJV instance, provided implicitly when called via my_instance\$keyword(...)
key	String; the name with keyword to add.
object	the format to be added. Must be enclosed in a call to JS .

Value

invisible(NULL)

See Also

Other AJV.Instance.Methods: [ajv.addFormat](#), [ajv.addKeyword](#), [ajv.addSchema](#), [ajv.errorsText](#), [ajv.removeSchema](#), [ajv.validateSchema](#), [ajv.validate](#)

Examples

```
## Not run:
my_ajv_instance = Ajv()
my_ajv_instance$keyword(key, object)

## End(Not run)
```

ajv.removeSchema *A wrapper for the Ajv.removeSchema method*

Description

The remove a schema from an Ajv instance

Usage

```
ajv.removeSchema(this, key)
```

Arguments

this	An AJV instance, provided implicitly when called via my_instance\$removeSchema(...)
key	String; the name with schema to remove

Value

invisible(NULL)

See Also

Other AJV.Instance.Methods: [ajv.addFormat](#), [ajv.addKeyword](#), [ajv.addSchema](#), [ajv.errorsText](#), [ajv.keyword](#), [ajv.validateSchema](#), [ajv.validate](#)

Examples

```
## Not run:
my_ajv_instance = Ajv()
my_ajv_instance$removeSchema

## End(Not run)
```

ajv.validate *A wrapper for the Ajv.validate method*

Description

The equivalent of calling `var ajv = new Ajv(); ajv.validate(...)` in javascript.

Usage

```
ajv.validate(this, schema, data)
```

Arguments

<code>this</code>	An AJV instance, provided implicitly when called via <code>my_instance\$validate(...)</code>
<code>schema</code>	The Schema with which to validate the data. <code>schema</code> may be a valid JSON string, an R object (i.e. <code>'list(...)'</code>), a connection to a JSON file, or the name of a JSON or YAML file. YAML files are parsed via <code>[js-yaml](https://www.npmjs.com/package/js-yaml)</code> 's <code>'safeLoad()'</code> method.
<code>data</code>	The data to be validated. may be any of the above foremat.

See Also

Other `AJV.Instance.Methods`: [ajv.addFormat](#), [ajv.addKeyword](#), [ajv.addSchema](#), [ajv.errorsText](#), [ajv.keyword](#), [ajv.removeSchema](#), [ajv.validateSchema](#)

Examples

```
## Not run:
my_ajv_instance = Ajv()
my_ajv_instance$validate

## End(Not run)
```

ajv.validateSchema *A wrapper for the Ajv.validateSchema method*

Description

The validate a json schema

Usage

```
ajv.validateSchema(this, schema)
```


Arguments

<code>this</code>	An AJV instance, provided implicitly when called via <code>my_instance\$ajv(...)</code>
<code>schema</code>	The Schema to be validated. <code>schema</code> may be a valid JSON string, an R object (i.e. <code>'list(...)'</code>), a connection to a JSON file, or the name of a JSON or YAML file. YAML files are parsed via <code>[js-yaml](https://www.npmjs.com/package/js-yaml)</code> 's <code>'safeLoad()'</code> method.

Value

boolean

See Also

Other `AJV.Instance.Methods`: [ajv.addFormat](#), [ajv.addKeyword](#), [ajv.addSchema](#), [ajv.errorsText](#), [ajv.keyword](#), [ajv.removeSchema](#), [ajv.validate](#)

Examples

```
## Not run:  
my_ajv_instance = Ajv()  
my_ajv_instance$validateSchema  
  
## End(Not run)
```

Index

Ajv, [2](#)
ajv.addFormat, [2](#), [3–5](#), [7–9](#)
ajv.addKeyword, [2](#), [3](#), [4](#), [5](#), [7–9](#)
ajv.addSchema, [2](#), [3](#), [4](#), [5](#), [7–9](#)
ajv.compile, [5](#)
ajv.errorsText, [2–4](#), [5](#), [7–9](#)
ajv.getSchema, [6](#)
ajv.keyword, [2–5](#), [6](#), [7–9](#)
ajv.removeSchema, [2–5](#), [7](#), [7](#), [8](#), [9](#)
ajv.validate, [2–5](#), [7](#), [8](#), [9](#)
ajv.validateSchema, [2–5](#), [7](#), [8](#), [8](#)

JS, [2](#), [6](#)