Package 'QuickJSR'

January 8, 2025

Title Interface for the 'QuickJS' Lightweight 'JavaScript' Engine

Version 1.5.1

Description An 'R' interface to the 'QuickJS' portable 'JavaScript' engine. The engine and all 'R' to 'JavaScript' interoperability is bundled within the package, requiring no dependencies beyond a 'C' compiler.

License MIT + file LICENSE

URL `https://github.com/andrjohns/QuickJSR`, `https://bellard.org/quickjs/`

BugReports `https://github.com/andrjohns/QuickJSR/issues`

Suggests knitr, rmarkdown, tinytest

Encoding UTF-8

Language en-AU

NeedsCompilation yes

RoxygenNote 7.3.1

SystemRequirements GNU make

VignetteBuilder knitr

Author Andrew R. Johnson [aut, cre] (<`https://orcid.org/0000-0001-7000-8065`>), Fabrice Bellard [cph] (Author of QuickJS sources and headers), Charlie Gordon [cph] (Author of QuickJS sources and headers), QuickJS-NG Authors [cph] (QuickJS-NG sources and headers)

Maintainer Andrew R. Johnson <`andrew.johnson@arjohnsonau.com`>

Repository CRAN

Date/Publication 2025-01-08 09:00:10 UTC

Contents

`QuickJSR-package` *The QuickJSR package.*

Description

An interface to the QuickJS lightweight Javascript engine

Author(s)

Maintainer: Andrew R. Johnson <`andrew.johnson@arjohnsonau.com`> (ORCID)

Other contributors:

- Fabrice Bellard (Author of QuickJS sources and headers) [copyright holder]
- Charlie Gordon (Author of QuickJS sources and headers) [copyright holder]

See Also

Useful links:

- `https://github.com/andrjohns/QuickJSR`
- `https://bellard.org/quickjs/`
- Report bugs at `https://github.com/andrjohns/QuickJSR/issues`

`cxxflags` *cxxflags*

Description

Function for returning the C/C++ flags needed for compilation using the package's headers

Usage

```
cxxflags(to_console = FALSE)
```

Arguments

`to_console`	Whether the result should be returned as a string

Value

Character string of CXX flags, or print flags to console and invisibly return NULL (for use in package Makevars or similar)

`from_json` *from_json*

Description

Use the QuickJS C API to convert a JSON string to an R object

Usage

```
from_json(json)
```

Arguments

`json`	JSON string to convert to an R object

Value

R object

`JSContext` *JSContext object*

Description

An initialised context within which to evaluate Javascript scripts or commands.

Usage

```
JSContext
```

Format

An object of class `list` of length 1.

Value

A JSContext object containing an initialised JavaScript context for evaluating scripts/commands

`JSContext-method-assign` *Assign a value to a variable in the current context*

Description

Assign a value to a variable in the current context

Usage

```
assign(var_name, value)
```

Arguments

`var_name`	The name of the variable to assign
`value`	The value to assign to the variable

Format

An object of class `NULL` of length 0.

Value

No return value, called for side effects

Examples

```
## Not run:
ctx <- JSContext$new()
ctx$assign("a", 1)
ctx$get("a")

## End(Not run)
```

`JSContext-method-call` *Call a JS function in the current context*

Description

Call a specified function in the JavaScript context with the provided arguments.

Usage

```
call(function_name, ...)
```

Arguments

`function_name`	The function to be called
`...`	The arguments to be passed to the function

Format

An object of class `NULL` of length 0.

Value

The result of calling the specified function

Examples

```
## Not run:
ctx <- JSContext$new()
ctx$source(code = "function add(a, b) { return a + b; }")
ctx$call("add", 1, 2)

## End(Not run)
```

`JSContext-method-get` *Get a variable from the current context*

Description

Get the value of a variable from the current context

Usage

```
get(var_name)
```

Arguments

`var_name`	The name of the variable to retrieve

Format

An object of class `NULL` of length 0.

Value

The value of the variable

Examples

```
## Not run:
ctx <- JSContext$new()
ctx$source(code = "var a = 1;")
ctx$get("a")

## End(Not run)
```

`JSContext-method-source` *Evaluate JS string or file in the current context*

Description

Evaluate a provided JavaScript file or string within the initialised context. Note that this method should only be used for initialising functions or values within the context, no values are returned from this function. See the `$call()` method for returning values.

Usage

```
source(file = NULL, code = NULL)
```

Arguments

`file`	A path to the JavaScript file to load
`code`	A single string of JavaScript to evaluate

Format

An object of class `NULL` of length 0.

Value

No return value, called for side effects

Examples

```
## Not run:
ctx <- JSContext$new()
ctx$source(file = "path/to/file.js")
ctx$source(code = "1 + 2")

## End(Not run)
```

`JSContext-method-validate`

Assess validity of JS code without evaluating

Description

Checks whether JS code string is valid code in the current context

Usage

```
validate(code_string)
```

Arguments

`code_string`	The JS code to check

Format

An object of class `NULL` of length 0.

Value

A boolean indicating whether code is valid

Examples

```
## Not run:
ctx <- JSContext$new()
ctx$validate("1 + 2")

## End(Not run)
```

`ldflags` *ldflags*

Description

Function for returning the flags needed for linking to the package

Usage

```
ldflags(to_console = FALSE)
```

Arguments

`to_console`	Whether the result should be returned as a string

Value

Character string of linker flags, or print flags to console and invisibly return NULL (for use in package Makevars or similar)

`qjs_eval` *qjs_eval*

Description

Evaluate a single Javascript expression.

Usage

```
qjs_eval(eval_string)
```

Arguments

`eval_string`	A single string of the expression to evaluate

Value

The result of the provided expression

Examples

```
# Return the sum of two numbers:
qjs_eval("1 + 2")

# Concatenate strings:
qjs_eval("'1' + '2'")

# Create lists from objects:
qjs_eval("var t = {'a' : 1, 'b' : 2}; t")
```

`quickjs_version` *Get the version of the bundled QuickJS library*

Description

Get the version of the bundled QuickJS library

Usage

```
quickjs_version()
```

Value

Character string of the version of the bundled QuickJS library

`to_json` *to_json*

Description

Use the QuickJS C API to convert an R object to a JSON string

Usage

```
to_json(arg, auto_unbox = FALSE)
```

Arguments

`arg`	Argument to convert to JSON
`auto_unbox`	Automatically unbox single element vectors

Value

JSON string

Index