Package: jsutils (via r-universe)

May 17, 2026

Title 'JavaScript' Utilities for 'R'

Version 0.3.0

Description A collection of popular/useful JavaScript utilities, including the terser minifier, sass compiler, typescript transpiler, and more.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Imports QuickJSR (>= 1.8.1)

URL https://github.com/andrjohns/jsutils, https://andrjohns.github.io/jsutils/

Repository https://andrjohns.r-universe.dev

Date/Publication 2026-05-17 05:33:03 UTC

RemoteUrl https://github.com/andrjohns/jsutils

RemoteRef HEAD

RemoteSha 30042a5fe1023cf02c5ffeae5a4b8946567f5872

Contents

esprima *Parse or Tokenize JavaScript code using Esprima*

Description

Use the Esprima library to parse or tokenize JavaScript code. Note that the first time this function is called, it will load the esprima library into the JavaScript context, which may take a few seconds. Subsequent calls will be faster.

Usage

```
esprima_parse(input, options = NULL, type = "script")

esprima_tokenize(input, options = list())
```

Arguments

input	Either a path to a file or a character string containing the JavaScript code to be parsed or tokenized.
options	A list of configuration options to pass to the Esprima parser or tokenizer.
type	A character string specifying the type of code to parse: "script" (default) or "module".

Value

For esprima_parse, a list representing the Abstract Syntax Tree (AST) of the parsed code. For esprima_tokenize, a list of tokens extracted from the input code.

Examples

```
js_code <- 'const answer = 42;'
esprima_parse(js_code)
esprima_tokenize(js_code)
```

js_versions *Get versions of bundled JavaScript libraries*

Description

This function returns the versions of the bundled JavaScript libraries used in the package.

Usage

```
js_versions()
```

Value

A named list with the versions of Esprima, sass.js, Terser, and TypeScript.

Examples

```
js_versions()
```

`sass` *Compile SASS/SCSS to CSS using the sass.js library*

Description

This function uses the sass.js library to compile SASS/SCSS code into CSS. It leverages the QuickJSR package to run JavaScript code within R. Note that the first time this function is called, it will load the sass.js library into the JavaScript context, which may take a few seconds. Subsequent calls will be faster.

Usage

```
sass(input, options = NULL)
```

Arguments

`input`	Either a path to a file or a character string containing the SASS/SCSS code to be compiled.
`options`	A list of options to pass to the sass.js compiler.

Value

A list containing the compiled CSS code and any warnings or errors.

Examples

```
scss_code <- "h1 { font-size: 40px; code { font-face: Roboto Mono; } }"
sass(scss_code, list(style = "compressed"))
```

terser *Minify JavaScript code using Terser*

Description

Uses the Terser JavaScript library to minify JavaScript code. Note that the first time this function is called, it will load the terser library into the JavaScript context, which may take a few seconds. Subsequent calls will be faster.

Usage

```
terser(input, options = NULL)
```

Arguments

input	Either a path to a file or a character string containing the JavaScript code to be minified.
options	A list of options to pass to Terser for minification. See the Terser documentation for available options.

Value

A list containing the minified code and any warnings or errors.

Examples

```
js_code <- "function add(a, b) { return a + b; }"
terser(js_code, list(sourceMap = TRUE))
```

typescript *Transpile TypeScript code to JavaScript*

Description

This function uses the TypeScript compiler to transpile TypeScript code into JavaScript. Note that the first time this function is called, it will load the TypeScript library into the JavaScript context, which may take a few seconds. Subsequent calls will be faster.

Usage

```
typescript(input, options = NULL)
```

Arguments

`input`	Either a path to a file or a character string containing the TypeScript code to be transpiled.
`options`	A list of options to pass to the TypeScript transpiler. See the TypeScript documentation for available options.

Value

A list containing the transpiled JavaScript code and any diagnostics.

Examples

```
ts_code <- "const greet = (name: string): string => `Hello, ${name}!`;"
typescript(ts_code, list(compilerOptions = list(target = "ES5")))
```

Index