Package ‘qgisprocess’

August 17, 2023

Title Use 'QGIS' Processing Algorithms

Version 0.1.0

Description Provides seamless access to the 'QGIS'
(https://qgis.org/en/site/) processing toolbox using the standalone
'qgis_process' command-line utility. Both native and third-party
(plugin) processing providers are supported. Beside referring data
sources from file, also common objects from 'sf', 'terra' and 'stars'
are supported. The native processing algorithms are documented by QGIS.org

License GPL (>= 3)

URL https://r-spatial.github.io/qgisprocess/,
https://github.com/r-spatial/qgisprocess

BugReports https://github.com/r-spatial/qgisprocess/issues

Depends R (>= 3.6.0)

Imports assertthat, glue, jsonlite, processx (>= 3.5.2), rappdirs,
rlang, stats, stringr, tibble, vctrs, withr

Suggests dplyr, knitr, mapview, raster, rmarkdown, rprojroot, sf,
spDataLarge, stars, stringi, terra, testthat, tidyrr

VignetteBuilder knitr

Additional_repositories https://geocompr.r-universe.dev

Config/checklist/communities inbo

Config/checklist/keywords R; package: QGIS

Encoding UTF-8

Language en-GB

RoxygenNote 7.2.3

SystemRequirements 'QGIS' latest or long-term release (>= 3.28). Older
'QGIS' releases are not officially supported, but may work
since 'QGIS' 3.16.

NeedsCompilation no

Maintainer  Floris Vanderhaeghe <floris.vanderhaeghe@inbo.be>

Repository  CRAN

Date/Publication  2023-08-17 15:10:02 UTC

R topics documented:

has_qgis .......................................................... 3
qgis_algorithms .................................................. 4
qgis_as_raster .................................................... 5
qgis_as_terra ..................................................... 6
qgis_clean_result ................................................ 7
qgis_configure .................................................... 8
qgis_detect_windows_paths ..................................... 9
qgis_enable_plugins ............................................. 10
qgis_extract_output ............................................ 11
qgis_function .................................................... 13
qgis_list_input .................................................. 14
qgis_path ......................................................... 15
qgis_result_status .............................................. 16
qgis_run .......................................................... 17
qgis_run_algorithm ............................................. 18
qgis_run_algorithm_p ........................................... 19
qgis_search_algorithms ......................................... 20
qgis_show_help .................................................. 21
qgis_tmp_file .................................................... 22
qgis_unconfigure ................................................ 23
qgis_using_json_input ......................................... 24
st_as_sf ........................................................ 25
st_as_stars ....................................................... 26

Index  28
has_qgis

Check availability of QGIS, a plugin, a provider or an algorithm

Description

has_qgis() checks whether the loaded qgisprocess cache is populated, which means that a QGIS installation was accessible and responsive when loading the package. qgis_has_plugin(), qgis_has_provider() and qgis_has_algorithm() check for the availability of one or several plugins, processing providers and algorithms, respectively. They are vectorized.

Usage

has_qgis()

qgis_has_plugin(plugin, query = FALSE, quiet = TRUE)

qgis_has_provider(provider, query = FALSE, quiet = TRUE)

qgis_has_algorithm(algorithm, query = FALSE, quiet = TRUE)

Arguments

plugin A plugin name (e.g., "native"). Can be a vector of names.
query Use TRUE to refresh the cached value.
quiet Use FALSE to display more information, possibly useful for debugging.
provider A provider name (e.g., "native"). Can be a vector of names.
algorithm A qualified algorithm name (e.g., "native:buffer"). Can be a vector of names.

Value

A logical, with length 1 in case of has_qgis().

Note

Only plugins that implement processing providers are supported.

See Also

Other topics about reporting the QGIS state: qgis_algorithms(), qgis_path(), qgis_using_json_input()

Examples

has_qgis()
if (has_qgis()) qgis_has_algorithm("native:filedownloader")
if (has_qgis()) qgis_has_provider("native")
if (has_qgis()) qgis_has_plugin(c("grassprovider", "processing_saga_nextgen"))
qgis_algorithms  List algorithms, processing providers or plugins

Description

Functions that return metadata about the installed and enabled algorithms or processing providers, or about the installed plugins that implement processing providers. See the QGIS docs for a detailed description of the algorithms provided 'out of the box' on QGIS.

Usage

```r
qgis_algorithms(query = FALSE, quiet = TRUE)
qgis_providers(query = FALSE, quiet = TRUE)
qgis_plugins(which = "all", query = FALSE, quiet = TRUE, ...)
```

Arguments

- `query` Use TRUE to refresh the cached value.
- `quiet` Use FALSE to display more information, possibly useful for debugging.
- `which` String defining which plugins to select, based on their status in QGIS (enabled or disabled). Must be one of: "all", "enabled", "disabled".
- `...` Only used by other functions calling this function.

Value

A tibble of algorithms, processing providers or plugins, with metadata.

See Also

- `qgis_enable_plugins()`, `qgis_disable_plugins()`
- Other topics about information on algorithms & processing providers: `qgis_search_algorithms()`, `qgis_show_help()`
- Other topics about reporting the QGIS state: `has_qgis()`, `qgis_path()`, `qgis_using_json_input()`

Examples

```r
qgis_algorithms()
qgis_providers()
qgis_plugins(quiet = FALSE)
qgis_plugins(which = "disabled")
```
qgis_as_raster

Convert a qgis_result object or one of its elements to a raster object

Description

Convert a qgis_result object or one of its elements to a raster object

Usage

qgis_as_raster(x, ...)
qgis_as_brick(x, ...)

## S3 method for class 'qgis_outputRaster'
qgis_as_raster(x, ...)

## S3 method for class 'qgis_outputRaster'
qgis_as_brick(x, ...)

## S3 method for class 'qgis_outputLayer'
qgis_as_raster(x, ...)

## S3 method for class 'qgis_outputLayer'
qgis_as_brick(x, ...)

## S3 method for class 'qgis_result'
qgis_as_raster(x, ...)

## S3 method for class 'qgis_result'
qgis_as_brick(x, ...)

Arguments

x A qgis_result object from qgis_run_algorithm() or a qgis_output object from one of the qgis_extract_output() functions.
...
Arguments passed to raster::raster() or raster::brick().

Value

A RasterLayer or a RasterBrick object.

See Also

Other topics about coercing processing output: qgis_as_terra(), st_as_sf, st_as_stars
Other topics about accessing or managing processing results: qgis_as_terra(), qgis_clean_result(), qgis_extract_output(), qgis_result_status(), st_as_sf, st_as_stars
Examples

```r
# not running below examples in R CMD check to save time
result <- qgis_run_algorithm(
  "native:slope",
  INPUT = system.file("longlake/longlake_depth.tif", package = "qgisprocess")
)

# most direct approach, autoselecting a `qgis_outputRaster` type
# output from the `result` object and reading as RasterLayer:
qgis_as_raster(result)

# if you need more control, extract the needed output element first:
output_raster <- qgis_extract_output(result, "OUTPUT")
qgis_as_raster(output_raster)
```

---

**qgis_as_terra**

Convert a `qgis_result` object or one of its elements to a terra object

**Description**

Convert a `qgis_result` object or one of its elements to a terra object

**Usage**

```r
qgis_as_terra(x, ...)
```

## S3 method for class 'qgis_outputRaster'
```r
qgis_as_terra(x, ...)
```

## S3 method for class 'qgis_outputLayer'
```r
qgis_as_terra(x, ...)
```

## S3 method for class 'qgis_result'
```r
qgis_as_terra(x, ...)
```

**Arguments**

- `x` A `qgis_result` object from `qgis_run_algorithm()` or a `qgis_output*` object from one of the `qgis_extract_output()` functions.
- `...` Arguments passed to `terra::rast()`.

**Value**

A `SpatRaster` or a `SpatVector` object.
qgis_clean_result  

Clean processing results

Description

Deletes any temporary files that are defined in a qgis_result object. These may comprise both input and output files.

Usage

qgis_clean_result(x)

Arguments

x A qgis_result object returned by qgis_run_algorithm().

Value

The qgis_result object passed to the function is returned invisibly.

See Also

Other topics about accessing or managing processing results: qgis_as_raster(), qgis_as_terra(), qgis_extract_output(), qgis_result_status(), st_as_sf, st_as_stars
Examples

```r
result <- qgis_run_algorithm(
  "native:buffer",
  INPUT = system.file("longlake/longlake_depth.gpkg", package = "qgisprocess"),
  DISTANCE = 10
)

file.exists(qgis_extract_output(result))
qgis_clean_result(result)
file.exists(qgis_extract_output(result))
```

---

**qgis_configure**  
Configure qgisprocess

**Description**

Run `qgis_configure()` to bring the package configuration in line with QGIS and to save this configuration to a persistent cache. See the Details section for more information about setting the path of the 'qgis_process' command line tool.

**Usage**

```r
qgis_configure(quiet = FALSE, use_cached_data = FALSE)
```

**Arguments**

- `quiet`  
  Use `FALSE` to display more information, possibly useful for debugging.

- `use_cached_data`  
  Use the cached algorithm list and path found when configuring qgisprocess during the last session. This saves some time loading the package.

**Details**

The qgisprocess package is a wrapper around the 'qgis_process' command line tool distributed with QGIS (>=3.14). Several functions use heuristics to detect the location of the 'qgis_process' executable.

When loading the package, the configuration is automatically read from the cache with `qgis_configure(use_cached_data = TRUE, quiet = TRUE)` in order to save time. Run `qgis_configure(use_cached_data = TRUE)` manually to get more details.

Use `qgis_algorithms()`, `qgis_providers()`, `qgis_plugins()`, `qgis_using_json_output()`, `qgis_path()` and `qgis_version()` to inspect cache contents.

If the configuration fails or you have more than one QGIS installation, you can set `options(qgisprocess.path = "path/to/qgis_process")` or the `R_QGISPROCESS_PATH` environment variable (useful on CI).

On Linux the 'qgis_process' executable is generally available on the user's PATH, on MacOS...
the executable is within the QGIS*.app/Contents/MacOS/bin folder, and on Windows the executable is named qgis_process-qgis.bat or qgis_process-qgis-dev.bat and is located in Program Files/QGIS*/bin or OSGeo4W(64)/bin.

Value

The result of `processx::run()`.

See Also

`qgis_unconfigure()`, `qgis_path()`, `qgis_version()`

Other topics about configuring QGIS and qgisprocess: `qgis_enable_plugins()`, `qgis_run()`

Examples

```r
# not running in R CMD check to save time
qgis_configure(use_cached_data = TRUE)

## Not run:
# package reconfiguration
# (not run in example() as it rewrites the package cache file)
qgis_configure()

## End(Not run)
```

---

**qgis_detect_windows_paths**

Detect QGIS installations that provide the ‘qgis_process’ command

Description

Discovers existing ‘qgis_process’ executables on the system and returns their filepath. Only available for Windows and macOS systems.

Usage

```r
qgis_detect_windows_paths(drive_letter = strsplit(R.home(), ":")[[1]][1])

qgis_detect_macos_paths()
```

Arguments

- `drive_letter` The drive letter on which to search. By default, this is the same drive letter as the R executable.
Value

A character vector of possible paths to the ‘qgis_process’ executable.

Note

These functions do not verify whether the discovered ‘qgis_process’ executables successfully run. You can run `qgis_path(query = TRUE, quiet = FALSE)` to discover and cache the first ‘qgis_process’ in the list that works.

See Also

`qgis_configure()`, `qgis_path()`

Examples

```r
if (.Platform$OS.type == "windows") qgis_detect_windows_paths()
if (Sys.info()["sysname"] == "Darwin") qgis_detect_macos_paths()
```
plugins are to be disabled. Note that the 'processing' plugin is ignored, because it is always available to 'qgis_process' (not QGIS though).

Value

A tibble of plugins, invisibly.

Note

Only plugins that implement processing providers are supported. Installing or removing plugins is not supported.

See Also

qgis_plugins()

Other topics about configuring QGIS and qgisprocess: qgis_configure(), qgis_run()

Examples

qgis_enable_plugins("name_of_plugin")

qgis_extract_output

Access processing output

Description

These functions extract one output element from the result of qgis_run_algorithm(), potentially more than one in the case of qgis_extract_output_by_class(). An output element can be extracted based on its name, its position in the printed qgis_result object returned by qgis_run_algorithm(), or its class.

qgis_extract_output() is an alias to qgis_extract_output_by_name().

Usage

qgis_extract_output_by_name(x, name = "OUTPUT", first = TRUE)
qgis_extract_output(x, name = "OUTPUT", first = TRUE)
qgis_extract_output_by_position(x, which)
qgis_extract_output_by_class(x, class, single = TRUE)
Arguments

x  A qgis_result object returned by qgis_run_algorithm().
name  The name of an output.
first  Logical. Should qgis_extract_output_by_name() fall back to the first output element if the default OUTPUT or output element is not available? Only takes effect if name is equal to OUTPUT or output, but not found.
which  The index of an output.
class  Character vector of classes. At least one class must be inherited by an element of x for that element to be selected.
single  Logical. Ensures the selection of a single output in qgis_extract_output_by_class(). The OUTPUT or output element is taken if available and on condition that it inherits a specified class; otherwise falls back to the first element that inherits a specified class.

Value

A qgis_output* object.

See Also

Other topics about accessing or managing processing results: qgis_as_raster(), qgis_as_terra(), qgis_clean_result(), qgis_result_status(), st_as_sf, st_as_stars

Examples

result <- qgis_run_algorithm(
  "native:buffer",
  INPUT = system.file("longlake/longlake_depth.gpkg", package = "qgisprocess"),
  DISTANCE = 10
)

# the print() method of a qgis_result only prints its output elements:
result

# nevertheless, more elements are included:
length(result)
names(result)

# extract the output element 'OUTPUT':
qgis_extract_output(result)
qgis_function

Create a wrapper function that runs one algorithm

Description

As opposed to `qgis_run_algorithm()`, `qgis_function()` creates a callable function based on the argument metadata provided by `qgis_get_argument_specs()`.

Usage

```r
qgis_function(algorithm, ...)
```

Arguments

- `algorithm` A qualified algorithm name (e.g., "native:buffer").
- `...` Algorithm arguments. These values are evaluated once and immediately, so you shouldn’t call `qgis_tmp_file()` here.

Details

The logic of `qgis_function()` has been implemented in R package `qgis`. This package also provides the QGIS documentation of each processing algorithm as corresponding R function documentation.

Value

A function.

Examples

```r
qgis_buffer <- qgis_function("native:buffer")
qgis_buffer(
  system.file(
    "longlake/longlake_depth.gpkg",
    package = "qgisprocess"
  ),
  DISTANCE = 10
)
```
qgis_list_input

Prepare a compound input argument

Description

Some algorithm arguments require a compound object, consisting of several layers or elements. These functions apply strict validation rules when generating this object and are recommended.

Usage

qgis_list_input(…)

qgis_dict_input(…)

Arguments

... Named values for qgis_dict_input() or unnamed values for qgis_list_input().

Details

qgis_list_input() generates an unnamed list of class qgis_list_input. The use of qgis_list_input() instead of list() is required for compound arguments in case of no-JSON input (see qgis_using_json_input()). Since it applies strict validation rules, it is recommended in all cases though.

qgis_dict_input() generates a named list of class qgis_dict_input. qgis_dict_input() is only supported when the JSON input method applies (see qgis_using_json_input()), where it can be interchanged with a named list(). It can only be used for arguments requiring named lists. Since it applies strict validation rules, it is recommended above list().

Value

• qgis_list_input(): An object of class 'qgis_list_input'

• qgis_dict_input(): An object of class 'qgis_dict_input'

Examples

qgis_list_input(1, 2, 3)
qgis_dict_input(a = 1, b = 2, c = 3)
Description

`qgis_path()` returns the filepath of the 'qgis_process' command, while `qgis_version()` returns the QGIS version.

Usage

```r
qgis_path(query = FALSE, quiet = TRUE)
qgis_version(query = FALSE, quiet = TRUE, full = TRUE, debug = FALSE)
```

Arguments

- `query` Use TRUE to refresh the cached value.
- `quiet` Use FALSE to display more information, possibly useful for debugging.
- `full` Logical. If FALSE, only return the "x.y.z" version string instead of the full version string that includes the name. Defaults to TRUE; ignored if `debug = TRUE`.
- `debug` Logical. If TRUE, also output the version of QGIS, the operating system and all relevant libraries, as reported by the 'qgis_process' command.

Value

A string.

See Also

`qgis_configure()`

Other topics about reporting the QGIS state: `has_qgis()`, `qgis_algorithms()`, `qgis_using_json_input()`

Examples

```r
qgis_path()
qgis_path(quiet = FALSE)
qgis_version()
qgis_version(full = FALSE)
qgis_version(debug = TRUE)
```
qgis_result_status

Access processing results: extra tools

Description

A qgis_result object is a list that, next to the output elements, also contains other elements that can be useful in scripting. Several of these can be extracted with convenience functions: the exit status of the process, standard output and standard error of 'qgis_process', arguments passed to 'qgis_process'.

Usage

qgis_result_status(x)
qgis_result_stdout(x)
qgis_result_stderr(x)
qgis_result_args(x)

Arguments

x

A qgis_result object returned by qgis_run_algorithm().

Value

• A number in case of qgis_result_status().
• A string in case of qgis_result_stdout() and qgis_result_stderr().
• A list in case of qgis_result_args().

See Also

Other topics about programming or debugging utilities: qgis_run(), qgis_tmp_file(), qgis_unconfigure(), qgis_using_json_input()

Other topics about accessing or managing processing results: qgis_as_raster(), qgis_as_terra(), qgis_clean_result(), qgis_extract_output(), st_as_sf, st_as_stars

Examples

result <- qgis_run_algorithm(
  "native:buffer",
  INPUT = system.file("longlake/longlake_depth.gpkg", package = "qgisprocess"),
  DISTANCE = 10
)
qgis_result_status(result)
qgis_run

stdout <- qgis_result_stdout(result)
cat(substr(stdout, 1, 335))
qgis_result_stderr(result)
qgis_result_args(result)

---

qgis_run

**Call the 'qgis_process' command directly**

---

**Description**

qgis_run() offers full access to 'qgis_process'. Run cat(qgis_run("--help")$stdout) to get the command's help.

**Usage**

qgis_run(args = character(), ..., env = qgis_env(), path = qgis_path())

**Arguments**

args Command-line arguments

... Passed to processx::run().

env A list() of environment variables. Defaults to getOption("qgisprocess.env", list(QT_QPA_PLATFORM = "offscreen").

path A path to the 'qgis_process' executable. Defaults to qgis_path().

**Value**

A processx::run() return value, i.e. a list with status, stdout, stderr and timeout elements.

**See Also**

Other topics about programming or debugging utilities: qgis_result_status(), qgis_tmp_file(), qgis_unconfigure(), qgis_using_json_input()

Other topics about configuring QGIS and qgisprocess: qgis_configure(), qgis_enable_plugins()

**Examples**

processx_list <- qgis_run(args = "--help")
cat(processx_list$stdout)
qgis_run_algorithm

Run an algorithm using 'qgis_process'

Description

Runs an algorithm using 'qgis_process'. See the QGIS docs for a detailed description of the algorithms provided 'out of the box' on QGIS.

Usage

```r
qgis_run_algorithm(
  algorithm,
  ...,
  PROJECT_PATH = NULL,
  ELLIPSOID = NULL,
  .raw_json_input = NULL,
  .quiet = TRUE
)
```

Arguments

- `algorithm`: A qualified algorithm name (e.g., "native:buffer") or a path to a QGIS model file.
- `...`: Named key-value pairs as arguments for the algorithm. Features of `rlang::list2()` are supported. These arguments are converted to strings using `as_qgis_argument()`.
- `PROJECT_PATH`, `ELLIPSOID`: Global values for QGIS project file and ellipsoid name for distance calculations.
- `.raw_json_input`: The raw JSON to use as input in place of ....
- `.quiet`: Use FALSE to get extra output from 'qgis_process'. This can be useful in debugging.

Value

A `qgis_result` object.

See Also

Other functions to run one geoprocessing algorithm: `qgis_run_algorithm_p()`

Examples

```r
qgis_run_algorithm(
  "native:buffer",
  INPUT = system.file("longlake/longlake_depth.gpkg", package = "qgisprocess"),
  DISTANCE = 10
)```
### qgis_run_algorithm_p

Run an algorithm using 'qgis_process': pipe-friendly wrapper

#### Description

`qgis_run_algorithm_p()` wraps `qgis_run_algorithm()`, passing its first argument to the first argument of the QGIS algorithm. This makes it more convenient in a pipeline (hence '_p' in the name).

#### Usage

```r
qgis_run_algorithm_p(
  .data,  
  algorithm,  
  ...,  
  .select = "OUTPUT",  
  .clean = TRUE,  
  .quiet = TRUE
)
```

#### Arguments

- **.data**: Passed to the first input of `algorithm`. If `.data` is a `qgis_result` (the result of a previous processing step), `.data[[.select]]` is passed instead.
- **algorithm**: A qualified algorithm name (e.g., "native:buffer").
- **...**: Other algorithm arguments. These values are evaluated once and immediately, so you shouldn’t call `qgis_tmp_file()` here.
- **.select**: String. The name of the element to select from `.data` if the latter is a `qgis_result`. Defaults to "OUTPUT".
- **.clean**: Logical. Should an incoming `qgis_result` be cleaned (using `qgis_clean_result()`) after processing?
- **.quiet**: Use `FALSE` to get extra output from 'qgis_process'. This can be useful in debugging.

#### Details

Uses `qgis_function()` under the hood.

#### Value

A `qgis_result` object.
See Also

Other functions to run one geoprocessing algorithm: `qgis_run_algorithm()`

Examples

```r
system.file(
  "longlake/longlake_depth.gpkg",
  package = "qgisprocess"
) |> 
qgis_run_algorithm_p(
  "native:buffer",
  DISTANCE = 10
)
```

---

### `qgis_search_algorithms`

**Search geoprocessing algorithms**

#### Description

Searches for algorithms using a regular expression. In its simplest form that is just a string that must match part of a character value.

#### Usage

```r
qgis_search_algorithms(algorithm = NULL, provider = NULL, group = NULL)
```

#### Arguments

- **algorithm**: Regular expression to match the algorithm or algorithm_title value from the output of `qgis_algorithms()`.
- **provider**: Regular expression to match the provider or provider_title value from the output of `qgis_algorithms()`.
- **group**: Regular expression to match the group value from the output of `qgis_algorithms()`.

#### Details

When using multiple arguments in combination, only the algorithms are returned that fulfill all conditions.

All regular expressions that `stringr::str_detect()` can handle, are accepted. Have a look at `stringi::search_regex()` to get a nice overview.

#### Value

A tibble.
See Also

Other topics about information on algorithms & processing providers: `qgis_algorithms()`, `qgis_show_help()`

Examples

```r
qgis_search_algorithms(
  algorithm = "point.*line",
  provider = "*native$"
)
```

---

`qgis_show_help` | Get detailed information about one algorithm

Description

Get detailed information about one algorithm

Usage

```r
qgis_show_help(algorithm)
qgis_get_description(algorithm)
qgis_get_argument_specs(algorithm)
qgis_get_output_specs(algorithm)
```

Arguments

`algorithm` | A qualified algorithm name (e.g., "native:buffer").

Value

- `qgis_get_description()`: a string.
- `qgis_get_argument_specs()`, `qgis_get_output_specs()`: a tibble.
- `qgis_show_help()`: the algorithm name, invisibly.

See Also

Other topics about information on algorithms & processing providers: `qgis_algorithms()`, `qgis_search_algorithms()`
qgis_tmp_file

Manage temporary files

Examples

qgis_get_description("native:filedownloader")

# not running below examples in R CMD check to save time
qgis_get_argument_specs("native:filedownloader")
qgis_get_output_specs("native:filedownloader")
qgis_show_help("native:filedownloader")

Description

These functions create temporary files that can be used in calls to qgis_run_algorithm() or elsewhere. These files are created in a special temporary directory (qgis_tmp_base()) that should be periodically cleaned up using qgis_clean_tmp(). You can set your preferred vector and/or raster file extension using options(qgisprocess.tmp_vector_ext = "...") and/or options(qgisprocess.tmp_raster_ext = "..."), respectively.

Usage

qgis_tmp_file(ext)

qgis_tmp_folder()

qgis_tmp_vector(ext = getOption("qgisprocess.tmp_vector_ext", ".gpkg"))

qgis_tmp_raster(ext = getOption("qgisprocess.tmp_raster_ext", ",tif"))

qgis_tmp_base()

qgis_clean_tmp()

Arguments

ext The file extension to be used.

Value

A character vector indicating the location of a (not yet created) temporary file.

See Also

Other topics about programming or debugging utilities: qgis_result_status(), qgis_run(), qgis_unconfigure(), qgis_using_json_input()
qgis_unconfigure

Examples

qgis_tmp_base()
qgis_tmp_file(".csv")
qgis_tmp_vector()
qgis_tmp_raster()

qgis_unconfigure  Clean the package cache

Description

Empties the qgisprocess cache environment.

Usage

qgis_unconfigure()

Value

NULL, invisibly.

See Also

Other topics about programming or debugging utilities: qgis_result_status(), qgis_run(), qgis_tmp_file(), qgis_using_json_input()

Examples

## Not run:
# not running this function in example() as it clears the cache environment.
qgis_unconfigure()

## End(Not run)

# undoing qgis_unconfigure() by repopulating the cache environment from file:

# not running in R CMD check to save time
qgis_configure(use Cached_data = TRUE)
qgis_using_json_input  Report if JSON objects are used for input to and output from `qgis_process`

Description

Returns a logical that reveals whether the JSON input and output methods are used, respectively.

Usage

qgis_using_json_input()

qgis_using_json_output(query = FALSE, quiet = TRUE)

Arguments

query   Use TRUE to refresh the cached value.
quiet   Use FALSE to display more information, possibly useful for debugging.

Details

Since QGIS 3.24 the JSON input method of `qgis_process` is used by default when calling the command. It allows for more complex input argument types in certain algorithms that require a more complex input argument, e.g. a list of lists (see qgis_list_input()). Likewise, JSON output is the default output format requested from `qgis_process`.

The settings can be overruled with the options qgisprocess.use_json_input or qgisprocess.use_json_output, and with the environment variables R_QGISPROCESS_USE_JSON_INPUT or R_QGISPROCESS_USE_JSON_OUTPUT.

Since the JSON output method is cached by the package, qgis_using_json_output(query = TRUE) is needed for these settings to take effect if the package was loaded already.

Value

A logical of length 1.

See Also

Other topics about programming or debugging utilities: qgis_result_status(), qgis_run(), qgis_tmp_file(), qgis_unconfigure()

Other topics about reporting the QGIS state: has_qgis(), qgis_algorithms(), qgis_path()

Examples

qgis_using_json_input()
Convert a qgis_result object or one of its elements to an sf object

Description
Convert a qgis_result object or one of its elements to an sf object

Usage
st_as_sf.qgis_result(x, ...)
st_as_sf.qgis_outputVector(x, ...)
st_as_sf.qgis_outputLayer(x, ...)

Arguments
x A qgis_result object from qgis_run_algorithm() or a qgis_output* object from one of the qgis_extract_output() functions.

Details
The sf package must be loaded explicitly to use these methods.

Value
An sf object.

Note
Just use st_as_sf() in R scripts, it will use the correct method.

See Also
Other topics about coercing processing output: qgis_as_raster(), qgis_as_terra(), st_as_stars
Other topics about accessing or managing processing results: qgis_as_raster(), qgis_as_terra(), qgis_clean_result(), qgis_extract_output(), qgis_result_status(), st_as_stars

Examples

# not running below examples in R CMD check to save time
result <- qgis_run_algorithm(
  "native:buffer",
  INPUT = system.file("longlake/longlake_depth.gpkg", package = "qgisprocess"),
  DISTANCE = 10
st_as_stars

Convert a qgis_result object or one of its elements to a stars object

Description
Convert a qgis_result object or one of its elements to a stars object

Usage
st_as_stars.qgis_outputRaster(x, ...)
st_as_stars.qgis_outputLayer(x, ...)
st_as_stars.qgis_result(x, ...)

Arguments
x A qgis_result object from qgis_run_algorithm() or a qgis_output* object from one of the qgis_extract_output() functions.
... Arguments passed to stars::read_stars().

Details
The stars package must be loaded explicitly to use these methods.

Value
A stars or a stars_proxy object.

Note
Just use st_as_stars() in R scripts, it will use the correct method.

See Also
Other topics about coercing processing output: qgis_as_raster(), qgis_as_terra(), st_as_sf
Other topics about accessing or managing processing results: qgis_as_raster(), qgis_as_terra(), qgis_clean_result(), qgis_extract_output(), qgis_result_status(), st_as_sf
Examples

# not running below examples in R CMD check to save time
result <- qgis_run_algorithm(
    "native:slope",
    INPUT = system.file("longlake/longlake_depth.tif", package = "qgisprocess")
)

# most direct approach, autoselecting a `qgis_outputRaster` type
# output from the `result` object and reading as stars or stars_proxy:
stars::st_as_stars(result)
stars::st_as_stars(result, proxy = TRUE)

# if you need more control, extract the needed output element first:
output_raster <- qgis_extract_output(result, "OUTPUT")
stars::st_as_stars(output_raster)
Index

* functions to manage and explore QGIS and qgisprocess
  has_qgis, 3
  qgis_algorithms, 4
  qgis_configure, 8
  qgis_detect_windows_paths, 9
  qgis_enable_plugins, 10
  qgis_path, 15
  qgis_search_algorithms, 20
  qgis_using_json_input, 24
* functions to run one geoprocessing algorithm
  qgis_run_algorithm, 18
  qgis_run_algorithm_p, 19
* main functions to access or manage processing results
  qgis_clean_result, 7
  qgis_extract_output, 11
* topics about accessing or managing processing results
  qgis_as_raster, 5
  qgis_as_terra, 6
  qgis_clean_result, 7
  qgis_extract_output, 11
  qgis_result_status, 16
  st_as_sf, 25
  st_as_stars, 26
* topics about coercing processing output
  qgis_as_raster, 5
  qgis_as_terra, 6
  st_as_sf, 25
  st_as_stars, 26
* topics about configuring QGIS and qgisprocess
  qgis_configure, 8
  qgis_enable_plugins, 10
  qgis_run, 17
* topics about information on algorithms & processing providers
  qgis_algorithms, 4
  qgis_search_algorithms, 20
  qgis_show_help, 21
* topics about preparing input values
  qgis_list_input, 14
* topics about programming or debugging utilities
  qgis_result_status, 16
  qgis_run, 17
  qgis_tmp_file, 22
  qgis_unconfigure, 23
  qgis_using_json_input, 24
* topics about reporting the QGIS state
  has_qgis, 3
  qgis_algorithms, 4
  qgis_path, 15
  qgis_using_json_input, 24

as_qgis_argument(), 18
has_qgis, 3, 4, 15, 24
list(), 17
processx::run(), 9, 17
qgis_algorithms, 3, 4, 15, 21, 24
qgis_algorithms(), 20
qgis_as_brick(qgis_as_raster), 5
qgis_as_raster, 5, 7, 12, 16, 25, 26
qgis_as_terra, 5, 6, 7, 12, 16, 25, 26
qgis_clean_result, 5, 7, 7, 12, 16, 25, 26
qgis_clean_result(), 19
qgis_clean_tmp(qgis_tmp_file), 22
qgis_clean_tmp(), 22
qgis_configure, 8, 11, 17
qgis_configure(), 10, 15
qgis_detect_macos_paths
(qgis_detect_windows_paths), 9
qgis_detect_windows_paths, 9
qgis_dict_input (qgis_list_input, 14
qgis_disable_plugins
   (qgis_enable_plugins), 10
qgis_disable_plugins(), 4
qgis_enable_plugins, 9, 10, 17
qgis_enable_plugins(), 4
qgis_extract_output, 5, 7, 11, 16, 25, 26
qgis_extract_output(), 5, 6, 25, 26
qgis_extract_output_by_class
   (qgis_extract_output), 11
qgis_extract_output_by_name
   (qgis_extract_output), 11
qgis_extract_output_by_position
   (qgis_extract_output), 11
qgis_function, 13
qgis_function(), 13, 19
qgis_get_argument_specs
   (qgis_show_help), 21
qgis_get_argument_specs(), 13
qgis_get_description (qgis_show_help), 21
qgis_get_output_specs (qgis_show_help), 21
qgis_has_algorithm (has_qgis), 3
qgis_has_plugin (has_qgis), 3
qgis_has_provider (has_qgis), 3
qgis_list_input, 14
qgis_list_input(), 24
qgis_path, 3, 4, 15, 24
qgis_path(), 9, 10, 17
qgis_plugins (qgis_algorithms), 4
qgis_plugins(), 10, 11
qgis_providers (qgis_algorithms), 4
qgis_result_args (qgis_result_status), 16
qgis_result_status, 5, 7, 12, 16, 17, 22–26
qgis_result_stderr
   (qgis_result_status), 16
qgis_result_stdout
   (qgis_result_status), 16
qgis_run, 9, 11, 16, 17, 22–24
qgis_run_algorithm, 18, 20
qgis_run_algorithm(), 5–7, 11–13, 16, 19, 22, 25, 26
qgis_run_algorithm_p, 18, 19
qgis_run_algorithm_p(), 19
qgis_search_algorithms, 4, 20, 21
qgis_show_help, 4, 21, 21
qgis_tmp_base (qgis_tmp_file), 22
qgis_tmp_base(), 22
qgis_tmp_file, 16, 17, 22, 23, 24
qgis_tmp_file(), 13, 19
qgis_tmp_folder (qgis_tmp_file), 22
qgis_tmp_raster (qgis_tmp_file), 22
qgis_tmp_vector (qgis_tmp_file), 22
qgis_unconfigure, 16, 17, 22, 23, 24
qgis_unconfigure(), 9
qgis_using_json_input, 3, 4, 15–17, 22, 23, 24
qgis_using_json_input(), 14
qgis_using_json_output
   (qgis_using_json_input), 24
qgis_version (qgis_path), 15
qgis_version(), 9
raster::brick(), 5
raster::raster(), 5
rlang::list2(), 18
sf::read_sf(), 25
st_as_sf, 5, 7, 12, 16, 25, 26
st_as_stars, 5, 7, 12, 16, 25, 26
stars::read_stars(), 26
stringi::search_regex(), 20
stringr::str_detect(), 20
terra::rast(), 6