Package 'scaffolder'

March 20, 2020

Type Package

Title Scaffolding Interfaces to Packages in Other Programming Languages

Version 0.0.1

Description Comprehensive set of tools for scaffolding R interfaces to modules, classes, functions, and documentations written in other programming languages, such as 'Python'.

License Apache License 2.0

URL https://github.com/terrytangyuan/scaffolder

BugReports https://github.com/terrytangyuan/scaffolder/issues

SystemRequirements Python (>= 2.7.0)

Encoding UTF-8

LazyData true

Depends R (>= 3.0)

Imports reticulate, utils

Suggests knitr, rmarkdown, testthat, stringr, tensorflow

RoxygenNote 7.0.2

VignetteBuilder knitr

NeedsCompilation no

Author Yuan Tang [aut, cre, cph] (<https://orcid.org/0000-0001-5243-233X>),
JJ Allaire [aut],
Kevin Ushey [aut],
RStudio [cph],
Navdeep Gill [ctb],
Erin LeDell [ctb]

Maintainer Yuan Tang <terrytangyuan@gmail.com>

Repository CRAN

Date/Publication 2020-03-20 10:10:02 UTC

R topics documented:

`custom_scaffold_py_function_wrapper`
Custom Scaffolding of R Wrappers for Python Functions

Description

This function can be used to generate R wrapper for a specified Python function while allowing to inject custom code for critical parts of the wrapper generation, such as process the any part of the docs obtained from `py_function_docs()` and append additional roxygen fields. The result from execution of `python_function` is assigned to a variable called `python_function_result` that can also be processed by `postprocess_fn` before writing the closing curly braces for the generated wrapper function.

Usage

```
custom_scaffold_py_function_wrapper(
  python_function,
  r_function = NULL,
  additional_roxygen_fields = NULL,
  process_docs_fn = function(docs) docs,
  process_param_fn = function(param, docs) param,
  process_param_doc_fn = function(param_doc, docs) param_doc,
  postprocess_fn = NULL,
  file_name = NULL
)
```

Arguments

`python_function`
: Fully qualified name of Python function or class constructor (e.g. `tf$nn$top_k`)

`r_function`
: Name of R function to generate (defaults to name of Python function if not specified)

`additional_roxygen_fields`
: A list of additional roxygen fields to write to the roxygen docs, e.g. `list(export = "",rdname = "generated-wrappers")`.

`process_docs_fn`
: A function to process docs obtained from `reticulate::py_function_docs(python_function)`.

`process_param_fn`
: A function to process each parameter needed for `python_funcion` before executing `python_funcion`.

process_param_doc_fn
: A function to process the roxygen docstring for each parameter.

postprocess_fn
: A function to inject any custom code in the form of a string before writing the closing curly braces for the generated wrapper function.

file_name
: The file name to write the generated wrapper function to. If NULL, the generated wrapper will only be printed out in the console.

Examples

```
library(tensorflow)
library(stringr)

# Example of a `process_param_fn` to cast parameters with default values
# that contains "L" to integers
process_int_param_fn <- function(param, docs) {
  # Extract the list of parameters that have integer values as default
  int_params <- gsub(
    " = [-]?[0-9]+L",
    "",
    str_extract_all(docs$signature, "[A-z]+ = [-]?[0-9]+L")[[1]])
  # Explicitly cast parameter in the list obtained above to integer
  if (param %in% int_params) {
    param <- paste0("as.integer(", param, ")")
  }
  param
}

# Note that since the default value of parameter `k` is `1L`. It is wrapped
# by `as.integer()` to ensure it's casted to integer before sending it to `tf$nn$top_k`
# for execution. We then print out the python function result.
custom_scaffold_py_function_wrapper(
  "tf$nn$top_k",
  r_function = "top_k",
  process_param_fn = process_int_param_fn,
  postprocess_fn = function() { "print(python_function_result)" })
```

---

scaffolder *Scaffolding Interfaces to Packages in Other Programming Languages*

Description

This package provides a comprehensive set of tools to scaffold interfaces to modules, classes, functions, and documentations written in other programming languages.

scaffold_py_function_wrapper *Scaffold R wrappers for Python functions*

Description

Scaffold R wrappers for Python functions

Usage

```
scaffold_py_function_wrapper(
  python_function,
  r_function = NULL,
  file_name = NULL
)
```

Arguments

python_function	Fully qualified name of Python function or class constructor (e.g. `tf$nn$top_k`)
r_function	Name of R function to generate (defaults to name of Python function if not specified)
file_name	The file name to write the generated wrapper function to. If `NULL`, the generated wrapper will only be printed out in the console.

Note

The generated wrapper will often require additional editing (e.g. to convert Python list literals in the docs to R lists, to massage R numeric values to Python integers via `as.integer` where required, etc.) so is really intended as an starting point for an R wrapper rather than a wrapper that can be used without modification.

Examples

```
library(scaffolder)
library(tensorflow)

scaffold_py_function_wrapper("tf$nn$top_k")
```

Index