Package: repboxEvaluate (via r-universe)

August 30, 2024

Type Package

Title Adaption of evaluate package version 0.23.01 to repbox

Version 0.1.0

Description Parsing and evaluation tools that make it easy to recreate
the command line behaviour of R.

License MIT + file LICENSE

Depends restorepoint, R (>= 3.0.2)

Imports methods

RoxygenNote 7.2.3

Encoding UTF-8

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

RemoteUrl https://github.com/repboxr/repboxEvaluate

RemoteRef main

RemoteSha 0a30aa43fce082ae8b7674ea2e90e24a7231de97

Contents

---

flush_console   *An emulation of flush.console() in evaluate()*

Description

When [evaluate()] is evaluating code, the text output is diverted into an internal connection, and there is no way to flush that connection. This function provides a way to "flush" the connection so that any text output can be immediately written out, and more importantly, the 'text' handler (specified in the 'output_handler' argument of 'evaluate()') will be called, which makes it possible for users to know it when the code produces text output using the handler.

Usage

```
flush_console()
```

Note

This function is supposed to be called inside 'evaluate()' (e.g. either a direct 'evaluate()' call or in **knitr** code chunks).

---

new_output_handler   *Custom output handlers.*

Description

An 'output_handler' handles the results of [evaluate()], including the values, graphics, conditions. Each type of output is handled by a particular function in the handler object.

Usage

```
new_output_handler(
  source = identity,
  text = identity,
  graphics = identity,
  message = identity,
  warning = identity,
  error = identity,
  value = render,
  calling_handlers = list()
)
```

Arguments

source	Function to handle the echoed source code under evaluation.
text	Function to handle any textual console output.
graphics	Function to handle graphics, as returned by [recordPlot()].
message	Function to handle [message()] output.
warning	Function to handle [warning()] output.
error	Function to handle [stop()] output.
value	Function to handle the values returned from evaluation. If it only has one argument, only visible values are handled; if it has more arguments, the second argument indicates whether the value is visible.
calling_handlers	
	List of [calling handlers][withCallingHandlers]. These handlers have precedence over the exiting handler installed by [evaluate()] when 'stop_on_error' is set to 0.

Details

The handler functions should accept an output object as their first argument. The return value of the handlers is ignored, except in the case of the 'value' handler, where a visible return value is saved in the output list.

Calling the constructor with no arguments results in the default handler, which mimics the behavior of the console by printing visible values.

Note that recursion is common: for example, if 'value' does any printing, then the 'text' or 'graphics' handlers may be called.

Value

A new 'output_handler' object

---

parse_all	*Parse, retaining comments.*

Description

Works very similarly to parse, but also keeps original formatting and comments.

Usage

```
parse_all(x, filename = NULL, allow_error = FALSE)
```

Arguments

x	object to parse. Can be a string, a file connection, or a function. If a connection, will be opened and closed only if it was closed initially.
filename	string overriding the file name
allow_error	whether to allow syntax errors in 'x'

Value

A data.frame with columns 'src', the source code, and 'expr'. If there are syntax errors in 'x' and 'allow_error = TRUE', the data frame has an attribute 'PARSE_ERROR' that stores the error object.

---

repbox_evaluate   *Evaluate input and return all details of evaluation.*

Description

Compare to [eval()], 'evaluate' captures all of the information necessary to recreate the output as if you had copied and pasted the code into a R terminal. It captures messages, warnings, errors and output, all correctly interleaved in the order in which they occured. It stores the final result, whether or not it should be visible, and the contents of the current graphics device.

Usage

```
repbox_evaluate(
  input,
  envir = parent.frame(),
  enclos = NULL,
  debug = FALSE,
  stop_on_error = 0L,
  keep_warning = TRUE,
  keep_message = TRUE,
  log_echo = FALSE,
  log_warning = FALSE,
  new_device = TRUE,
  output_handler = default_output_handler,
  filename = NULL,
  include_timing = FALSE,
  remove_src_from_out = TRUE
)
```

Arguments

input	input object to be parsed and evaluated. May be a string, file connection or function. Passed on to [parse_all()].
envir	environment in which to evaluate expressions.
enclos	when 'envir' is a list or data frame, this is treated as the parent environment to 'envir'.
debug	if 'TRUE', displays information useful for debugging, including all output that evaluate captures.
stop_on_error	if '2', evaluation will halt on first error and you will get no results back. If '1', evaluation will stop on first error without signaling the error, and you will get back all results up to that point. If '0' will continue running all code, just as if you'd pasted the code into the command line.

keep_warning, keep_message

>   whether to record warnings and messages; if 'FALSE', messages will be suppressed; if 'NA', they will not be captured (normally they will be sent to the console). Note that if the environment variable 'R_EVALUATE_BYPASS_MESSAGES' is set to true, these arguments will always be set to 'NA', meaning that messages will not be captured by this function.

log_echo, log_warning

>   If 'TRUE', will immediately log code and warnings (respectively) to 'stderr'.

new_device   if 'TRUE', will open a new graphics device and automatically close it after completion. This prevents evaluation from interfering with your existing graphics environment.

output_handler   an instance of [output_handler()] that processes the output from the evaluation. The default simply prints the visible return values.

filename   string overrriding the [base::srcfile()] filename.

include_timing   if 'TRUE', evaluate will wrap each input expression in 'system.time()', which will be accessed by following 'replay()' call to produce timing information for each evaluated command.

| replay | *Replay a list of evaluated results.* |

Description

Replay a list of evaluated results, as if you'd run them in an R terminal.

Usage

```
replay(x)
```

Arguments

x   result from [evaluate()]

Examples

```
samples <- system.file("tests", "testthat", package = "evaluate")
if (file_test("-d", samples)) {
    replay(evaluate(file(file.path(samples, "order.R"))))
    replay(evaluate(file(file.path(samples, "plot.R"))))
    replay(evaluate(file(file.path(samples, "data.R"))))
}
```

Index