Package: plr (via r-universe)

October 10, 2024

Title Utility Functions for 'PrairieLearn' and R

Version 0.0.2.3

Date 2021-08-17

Author Dirk Eddelbuettel and Alton Barbehenn

Maintainer Dirk Eddelbuettel <edd@debian.org>

Description 'PrairieLearn' is an online problem-driven learning system for creating homeworks and tests. This package adds some helper functions for using it along with R as we are currently doing for <https://stat430.com>.

License GPL (>= 2)

OS_type unix

NeedsCompilation no

Encoding UTF-8

LazyData true

Imports unix, jsonlite

Suggests tinytest

RoxygenNote 6.0.1

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

RemoteUrl https://github.com/stat447/plr

RemoteRef HEAD

RemoteSha 21803deed3e216c414b153d89eb1a446b9f47bb1

Contents

get_question_details *Extract question name and score from header*

Description

This function is inspired by the roxygen2 decoration of source files with content used to create the manual and help files. Here we expect two tags `@title` with the displayed title of the question, and `@score` with the number of available points.

Usage

```
get_question_details(dir, pattern = "^test.*\\.[rR]$")
```

Arguments

`dir`	Directory containing the test files for a question
`pattern`	A regular expression identifying test files in the directory

Value

A data.frame object with colums name, file, and max_points

message_to_test_result *Helper function to format result object returned to PL*

Description

Helper function to format result object returned to PL

Usage

```
message_to_test_result(msg, max_pts = 100)
```

Arguments

`msg`	Character variable with the error or warning received
`max_pts`	Optional numeric variable with maximal attainable points, usuall 100

Value

A data.frame object with four elements as expected by PL

`source_and_eval_safe` *Wrapper to source a file and safely evaluate an expression*

Description

We assume all files surrounding the to be evaluated files have different user ids and file modes not allowing the supplied user id to read them. One way to do that is to just set all files within the evaluation directories to `root:root` removing group and others the rights to read (or write or execute). We therefore also `chmod` the supplied file back to mode "0644" ensuring that the file can be read so that the expression can be evaluated—but nothing else should be in reach.

Usage

```
source_and_eval_safe(file, expr, uid = NULL)

eval_safe_as(expr, uid = NULL)

source_and_eval_safe_with_hiding(file, expr, uid = NULL, path = NULL)
```

Arguments

`file`	A filename with an R file to be source, typically containing the student code to be evaluated safely.
`expr`	An expression to be evaluate by `eval_safe`, typically the name of the sane of the function containing the student code plus the argument supplied from the test runner.
`uid`	Optional numeric or character user id identifying the user id with (presumably lower) privileges as which the code is running; the numeric uid is obtained via `user_info` is a character is supplied. Note that using this argument requires being the 'root' user.
`path`	Optional path to a file that should be hidden before evaluation happens. It is then unhidden on exit.

Details

The `source_and_eval_safe_with_hiding` variant can *hide* a given file, for example containing a reference answer, but assigning it to a unique temporary name so that it cannot be sourced.

The `eval_safe_as` convenience function fetches the (numeric) user id before calling `unix::eval_safe`; it is equivalent to `source_and_eval_safe` but does not involve a file.

Note that you must run these functions as the 'root' user in order to set the uid.

Value

A value of the `expr` sourced from `file` and evaluated by `uid`, or NULL in case of error.

Examples

```
## Not run:
n <- sample(3:20, 1)         # random payload
res <- source_and_eval_safe("code/fib.R", fib(n), "ag")

## End(Not run)
```

`test_question` *Run a whole question and report aggregate results*

Description

This function is the equivalent of the `pltest.R` script which, given a directory runs the tests file therein and reports the results in a JSON file for PrairieLearn to consume.

Usage

```
test_question(tests_dir = "/grade/tests/tests",
  results_file = "results.json")
```

Arguments

`tests_dir`	Directory containing the test files for a question
`results_file`	JSON file into which results are written

Value

The results data.frame is returned, but the functions is invoked for its side-effect of creating the JSON file

Index