Package: defer (via r-universe)

October 12, 2024

Title Defer Errors

Version 0.1.0

Description Create errors that can be deferred until later. With deferrable errors, collections of errors that are not immediately fatal can be collected and reported back at once.

URL https://github.com/mrc-ide/odin

BugReports https://github.com/mrc-ide/odin/issues

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Suggests testthat

RoxygenNote 6.1.1

Language en-GB

Repository https://mrc-ide.r-universe.dev

RemoteUrl https://github.com/reside-ic/defer

RemoteRef master

RemoteSha b9f933772c59974dc904de09cdcc4972f0812f0c

Contents

deferrable_error *Create a deferrable error*

Description

Create an error that will stop immediately, or can be continued from.

Usage

```
deferrable_error(message)
```

Arguments

`message`	The error message

Examples

```
# Deferrable errors will throw immediately if no suitable calling
# handlers are established:
tryCatch(
  deferrable_error("my error"),
  error = identity)

# Create a deferrable error and continue from it, using
# withCallingHandlers:
value <- withCallingHandlers({
   x <- 1
   defer::deferrable_error("a deferrable error")
   x * 2
 },
 deferrable_error = function(e)
   invokeRestart("continue_deferrable_error"))
```

deferred_errors_flush *Flush deferred errors*

Description

Within a `defer_errors` block, flush any deferred errors, turning them into realised errors. If no deferrable errors have occurred, this function has no effect.

Usage

```
deferred_errors_flush()
```

Examples

```
check_positive <- function(x) {
  if (x < 0) {
    deferrable_error(paste("got a negative number:", x))
  }
}
err <- tryCatch(
  defer::defer_errors({
    check_positive(-1)
    defer::deferred_errors_flush()
    check_positive(-2)
  }),
  error = identity)
err
```

`defer_errors` *Run a block of code, collecting deferrable errors*

Description

Run a block of code, collecting any `deferrable_error` calls that occur. Ordinary errors will be thrown immediately.

Usage

```
defer_errors(expr, handler = stop)
```

Arguments

`expr`	An expression to evaluate
`handler`	The final handler for the deferred errors. The default is `stop` which will raise the collected error. Alternatively, use `return` to return the error

Details

The error object will contain an element `errors` with the deferred errors, each of which will contain elements `message`, `call` (the call that *precedes* `deferrable_error` and `calls` which contains the "interesting" part of the stack trace (i.e., only calls below the `defer_errors` infrastructure).

Examples

```
check_positive <- function(x) {
  if (x < 0) {
    deferrable_error(paste("got a negative number:", x))
  }
}
err <- tryCatch(
  defer::defer_errors({
    check_positive(0)
    check_positive(-1)
    check_positive(-2)
  }),
  error = identity)
err

# Directly return the error:
err <- defer::defer_errors({
  check_positive(0)
  check_positive(-1)
  check_positive(-2)
}, handler = return)

# Stack traces are included to improve downstream reporting:
f <- function(x) {
  g(x)
}
g <- function(x) {
  check_positive(x)
}
err <- defer_errors({
  f(0)
  f(-1)
  f(-2)
}, handler = return)
err$errors[[1]]$calls
```

Index