

Package: entratar (via r-universe)

October 16, 2026

Title What the Package Does (One Line, Title Case)

Version 0.0.0.9000

Description What the package does (one paragraph).

License MIT + file LICENSE

URL <https://github.com/jimbrig/entratar>,
<http://docs.jimbrig.com/entratar/>

BugReports <https://github.com/jimbrig/entratar/issues>

Imports cli, crayon, desc, rlang ($\geq 1.1.0$), tools, utils, yyjsonr

Suggests httptest2, httr2, knitr, rmarkdown, testthat ($\geq 3.0.0$)

VignetteBuilder knitr

Config/roxygen2/markdown TRUE

Config/roxygen2/version 8.1.1

Config/testthat/edition 3

Encoding UTF-8

Language en-US

Roxygen list(markdown = TRUE)

Repository <https://jimbrig.r-universe.dev>

Date/Publication 2026-10-16 20:53:05 UTC

RemoteUrl <https://github.com/jimbrig/entratar>

RemoteRef HEAD

RemoteSha 14746ab1e7bc2ce8564acf1e8290aeb172c403d5

Contents

check_inherits	2
check_not_empty	3
check_string	4
entrata_abort_check	4
entrata_active_config	7
entratar-conditions	8

check_inherits	<i>Class Inheritance Checks</i>
----------------	---------------------------------

Description

These functions perform checks that assert the underlying class of objects passed to them.

- `check_inherits()`: checks that object `x` is of class `class` using `base::inherits()`
- `check_inherits2()`: checks that object `x` is of class `class` using `base::.class2()`
- `check_inherits_any()`: checks that object `x` is at least one of the provided classes via `rlang::inherits_any()`
- `check_inherits_all()`: checks that object `x` is all of the provided classes via `rlang::inherits_all()`

If validation fails for any of these functions, an error is thrown via `check_abort()` displaying a friendly error message.

Usage

```
check_inherits(  
  x,  
  class,  
  arg = rlang::caller_arg(x),  
  call = rlang::caller_env()  
)
```

```
check_inherits2(  
  x,  
  class,  
  arg = rlang::caller_arg(x),  
  call = rlang::caller_env()  
)
```

```
check_inherits_any(  
  x,  
  classes,  
  arg = rlang::caller_arg(x),  
  call = rlang::caller_env()  
)
```

```
check_inherits_all(  
  x,  
  classes,  
  arg = rlang::caller_arg(x),  
  call = rlang::caller_env()  
)
```

Arguments

x	The object to check.
class, classes	The name of the class or classes to use during checking.
arg	An argument name as a string. This argument will be mentioned in error messages as the input that is at the origin of a problem.
call	The execution environment of a currently running function, e.g. <code>caller_env()</code> . The function will be mentioned in error messages as the source of the error. See the <code>call</code> argument of <code>abort()</code> for more information.

Value

If checks pass, invisibly returns the provided `x` object. If checks fail, a condition error is thrown.

<code>check_not_empty</code>	<i>Check Not Empty</i>
------------------------------	------------------------

Description

Checks the provided `x` is not "empty" via `rlang::is_empty()`.

Usage

```
check_not_empty(x, arg = rlang::caller_arg(x), call = rlang::caller_env())
```

Arguments

x	The object to check.
arg	An argument name as a string. This argument will be mentioned in error messages as the input that is at the origin of a problem.
call	The execution environment of a currently running function, e.g. <code>caller_env()</code> . The function will be mentioned in error messages as the source of the error. See the <code>call</code> argument of <code>abort()</code> for more information.

Value

If checks pass, invisibly returns the provided `x` object. If checks fail, a condition error is thrown.

<code>check_string</code>	<i>Check String (Scalar)</i>
---------------------------	------------------------------

Description

Checks the provided `x` is a scalar string via `rlang::check_string()`.

Usage

```
check_string(
  x,
  allow_empty = FALSE,
  allow_na = FALSE,
  allow_null = FALSE,
  arg = rlang::caller_arg(x),
  call = rlang::caller_env()
)
```

Arguments

<code>x</code>	The argument to check.
<code>allow_empty</code>	Whether the empty string "" is allowed (<code>check_string()</code> only).
<code>allow_na</code>	Whether NA values are allowed.
<code>allow_null</code>	Whether NULL is allowed.
<code>arg</code>	An argument name as a string. This argument will be mentioned in error messages as the input that is at the origin of a problem.
<code>call</code>	The execution environment of a currently running function, e.g. <code>caller_env()</code> . The function will be mentioned in error messages as the source of the error. See the <code>call</code> argument of <code>abort()</code> for more information.

Value

If `x` is not a scalar string, an error is thrown. Otherwise, `x` is invisibly returned.

<code>entrata_abort_check</code>	<i>Check Conditions</i>
----------------------------------	-------------------------

Description

These functions are specialized versions of the `entratar` condition functions that are intended for use in the package's internal checking functions. They create conditions with specific classes that can be used to identify and handle errors, warnings, and messages.

Usage

```

entrata_abort_check(
  msg,
  ...,
  parent = NULL,
  call = rlang::caller_env(),
  .envir = parent.frame()
)

entrata_warn_check(
  msg,
  ...,
  call = rlang::caller_env(),
  .envir = parent.frame()
)

entrata_inform_check(
  msg,
  ...,
  call = rlang::caller_env(),
  .envir = parent.frame()
)

check_abort(
  msg,
  ...,
  parent = NULL,
  call = rlang::caller_env(),
  .envir = parent.frame()
)

check_warn(msg, ..., call = rlang::caller_env(), .envir = parent.frame())

check_inform(msg, ..., call = rlang::caller_env(), .envir = parent.frame())

```

Arguments

<code>msg</code>	Character string or vector containing the message to be displayed. This gets passed to <code>cli::cli_bullets()</code> allowing for <code>cli</code> 's custom formatting, glue interpolation, inline markup syntax, and other features.
<code>...</code>	Passed to <code>rlang::abort()</code> , <code>rlang::warn()</code> or <code>rlang::inform()</code> .
<code>call</code>	The execution environment of a currently running function, e.g. <code>call = caller_env()</code>. The corresponding function call is retrieved and mentioned in error messages as the source of the error. You only need to supply <code>call</code> when throwing a condition from a helper function which wouldn't be relevant to mention in the message. Can also be <code>NULL</code> or a defused function call to respectively not display any call or hard-code a code to display.

For more information about error calls, see [Including function calls in error messages](#).

`.envir` Environment to evaluate the glue expressions in.

Details

The condition class hierarchy is structured like so:

```
<entrata_condition>
  <entrata_error>
    <entrata_{custom}_error>
  <entrata_warning>
    <entrata_{custom}_warning>
  <entrata_message>
    <entrata_{custom}_message>
```

where `<entrata_condition>` is the base class for all `entratar` conditions and sits on top of the existing `rlang` and base R condition classes.

For example, an `entrata` error thrown without any additional custom class added to it using `entrata_abort()` will have the following classes:

```
[1] "entrata_error" "entrata_condition" "rlang_error" "error" "condition"
```

This allows for filtering and handling of Entrata-specific conditions while still being compatible with the broader R condition system.

Value

These functions do not return values, but are used for their side effects, namely creating and signaling conditions. The `entrata_abort()` function will stop execution, while `entrata_warn()` & `entrata_inform()` will allow execution to proceed.

Functions

- `entrata_abort_check()`: creates an `entrata_error` condition for checks.
- `entrata_warn_check()`: creates an `entrata_warning` condition for checks.
- `entrata_inform_check()`: creates an `entrata_message` condition for checks.

Aliases

- `check_abort()`: alias for `entrata_abort_check()`
- `check_warn()`: alias for `entrata_warn_check()`
- `check_inform()`: alias for `entrata_inform_check()`

See Also

Utility Functions: [entrata_active_config](#)

Conditions: [entratar-conditions](#)

Examples

```
## Not run:
entrata_abort("This is a entratar error.")

# catch a condition and check its classes
err <- rlang::catch_cnd(entrata_abort("Error Message"))
class(err)

# catch a warning and check its classes
wrn <- rlang::catch_cnd(entrata_warn("Warning Message"))
class(wrn)

# catch a message and check its classes
msg <- rlang::catch_cnd(entrata_inform("Message"))
class(msg)

## End(Not run)
```

entrata_active_config

Manage Active Configuration

Description

Functions for quickly accessing or setting the active configuration file and environment for `config::get()`.

Get:

- `get_active_config_env()`: Returns the active configuration run-time environment (ENTRATA_CONFIG_ACTIVE, R_CONFIG_ACTIVE).
- `get_active_config_file()`: Returns the active configuration file (ENTRATA_CONFIG_FILE, R_CONFIG_FILE).

Set:

- `set_active_config_env(env)`: Sets the active configuration environment
- `set_active_config_file(file)`: Sets the active configuration file

Usage

```
get_active_config_env(
  default = c("default", "development", "production", "testing")
)

set_active_config_env(env)

get_active_config_file(default = pkg_sys_config("config.yml"))

set_active_config_file(file)
```

Arguments

default	Default environment to return if no active environment is set. Defaults to "default". Possible values are "default", "development", "production", "testing".
env	The configuration environment to set as active.
file	The configuration file path to set as active.

Value

- `get_active_config_env()`: The active configuration environment as a string.
- `get_active_config_file()`: The active configuration file as a string.
- `set_active_config_env(env)`: Returns the previous active configuration environment.
- `set_active_config_file(file)`: Returns the previous active configuration file.

See Also

`config::get()`

Utility Functions: `entrata_abort_check()`

`entratar-conditions` `entratar` *Custom Conditions*

Description

Functions for creating and handling custom Entrata-specific conditions, including errors, warnings, and messages.

- `entrata_abort()`: creates a condition representing an error and when thrown, aborts execution.
- `entrata_warn()`: creates a condition representing a warning condition
- `entrata_inform()`: creates a condition representing a message condition

These functions wrap `cli::cli_abort()`, `cli::cli_warn()`, and `cli::cli_inform()` respectively, adding `entratar`-specific context and class names (`entrata_*`) to the condition objects for downstream handling and identification.

Usage

```
entrata_abort(
  msg = NULL,
  cls = NULL,
  ...,
  parent = NULL,
  call = rlang::caller_env(),
```

```

    .envir = parent.frame()
  )

  entrata_warn(msg = NULL, cls = NULL, ..., .envir = parent.frame())

  entrata_inform(msg = NULL, cls = NULL, ..., .envir = parent.frame())

```

Arguments

<code>msg</code>	Character string or vector containing the message to be displayed. This gets passed to <code>cli::cli_bullets()</code> allowing for <code>cli</code> 's custom formatting, glue interpolation, inline markup syntax, and other features.
<code>cls</code>	Optional character vector of additional classes to assign to the condition.
<code>...</code>	Passed to <code>rlang::abort()</code> , <code>rlang::warn()</code> or <code>rlang::inform()</code> .
<code>call</code>	The execution environment of a currently running function, e.g. <code>call = caller_env()</code>. The corresponding function call is retrieved and mentioned in error messages as the source of the error. You only need to supply <code>call</code> when throwing a condition from a helper function which wouldn't be relevant to mention in the message. Can also be <code>NULL</code> or a defused function call to respectively not display any call or hard-code a code to display. For more information about error calls, see Including function calls in error messages.
<code>.envir</code>	Environment to evaluate the glue expressions in.

Details

The condition class hierarchy is structured like so:

```

<entrata_condition>
  <entrata_error>
    <entrata_{custom}_error>
  <entrata_warning>
    <entrata_{custom}_warning>
  <entrata_message>
    <entrata_{custom}_message>

```

where `<entrata_condition>` is the base class for all `entratar` conditions and sits on top of the existing `rlang` and base R condition classes.

For example, an `entrata` error thrown without any additional custom class added to it using `entrata_abort()` will have the following classes:

```
[1] "entrata_error" "entrata_condition" "rlang_error" "error" "condition"
```

This allows for filtering and handling of Entrata-specific conditions while still being compatible with the broader R condition system.

Value

These functions do not return values, but are used for their side effects, namely creating and signaling conditions. The `entrata_abort()` function will stop execution, while `entrata_warn()` & `entrata_inform()` will allow execution to proceed.

See Also

Conditions: [entrata_abort_check\(\)](#)

Examples

```
## Not run:
entrata_abort("This is a entratar error.")

# catch a condition and check its classes
err <- rlang::catch_cnd(entrata_abort("Error Message"))
class(err)

# catch a warning and check its classes
wrn <- rlang::catch_cnd(entrata_warn("Warning Message"))
class(wrn)

# catch a message and check its classes
msg <- rlang::catch_cnd(entrata_inform("Message"))
class(msg)

## End(Not run)
```

Index

- * checks
 - entrata_abort_check, 4
- * conditions
 - entrata_abort_check, 4
 - entratar-conditions, 8
- * config
 - entrata_active_config, 7
- * package
 - entratar-conditions, 8
- * utils
 - entrata_abort_check, 4
 - entrata_active_config, 7

abort(), 3, 4

active_config

- (entrata_active_config), 7

base::class2(), 2

base::inherits(), 2

check_abort (entrata_abort_check), 4

check_inform (entrata_abort_check), 4

check_inherits, 2

check_inherits2 (check_inherits), 2

check_inherits_all (check_inherits), 2

check_inherits_any (check_inherits), 2

check_not_empty, 3

check_string, 4

check_warn (entrata_abort_check), 4

cli::cli_abort(), 8

cli::cli_bullets(), 5, 9

cli::cli_inform(), 8

cli::cli_warn(), 8

conditions (entratar-conditions), 8

config::get(), 7, 8

defused function call, 5, 9

entrata_abort (entratar-conditions), 8

entrata_abort_check, 4

entrata_abort_check(), 8, 10

entrata_active_config, 6, 7

entrata_inform (entratar-conditions), 8

entrata_inform_check

- (entrata_abort_check), 4

entrata_warn (entratar-conditions), 8

entrata_warn_check

- (entrata_abort_check), 4

entratar-conditions, 8

get_active_config_env

- (entrata_active_config), 7

get_active_config_file

- (entrata_active_config), 7

Including function calls in error messages, 6, 9

rlang::abort(), 5, 9

rlang::check_string(), 4

rlang::inform(), 5, 9

rlang::inherits_all(), 2

rlang::inherits_any(), 2

rlang::is_empty(), 3

rlang::warn(), 5, 9

set_active_config_env

- (entrata_active_config), 7

set_active_config_file

- (entrata_active_config), 7