Skip to main content

__Back to top

__ `Ctrl`+`K`

[![conda 25.1.2.dev29 documentation -

Home](../../../../../_static/conda_logo_full.svg)](../../../../../index.html)

  * [Conda](https://docs.conda.io/projects/conda/)
  * [Conda-build](https://docs.conda.io/projects/conda-build)
  * [Miniconda](https://docs.anaconda.com/free/miniconda/)
  * conda.org

  * [__ GitHub](https://github.com/conda/conda "GitHub")
  * [![Element](../../../../../_static/element_logo.svg)](http://bit.ly/conda-chat-room "Element")
  * [__ Discourse](https://conda.discourse.group/ "Discourse")

  * [Conda](https://docs.conda.io/projects/conda/)
  * [Conda-build](https://docs.conda.io/projects/conda-build)
  * [Miniconda](https://docs.anaconda.com/free/miniconda/)
  * conda.org

  * [__ GitHub](https://github.com/conda/conda "GitHub")
  * [![Element](../../../../../_static/element_logo.svg)](http://bit.ly/conda-chat-room "Element")
  * [__ Discourse](https://conda.discourse.group/ "Discourse")

Navigation

* [Using non-standard certificates](../../../../../user-guide/configuration/non-standard-certs.html)

* [Using Custom Locations for Environment and Package Cache](../../../../../user-guide/configuration/custom-env-and-pkg-locations.html)

* [Improving interoperability with pip](../../../../../user-guide/configuration/pip-interoperability.html)

* [Using the free channel](../../../../../user-guide/configuration/free-channel.html)

* [Concepts](../../../../../user-guide/concepts/index.html) __

* [Commands](../../../../../user-guide/concepts/conda-commands.html)

* [Packages](../../../../../user-guide/concepts/packages.html)

* [Package specification](../../../../../user-guide/concepts/pkg-specs.html)

* [Package search and install specifications](../../../../../user-guide/concepts/pkg-search.html)

* [Channels](../../../../../user-guide/concepts/channels.html)

* [Environments](../../../../../user-guide/concepts/environments.html)

* [Installing with conda](../../../../../user-guide/concepts/installing-with-conda.html)

* [Performance](../../../../../user-guide/concepts/conda-performance.html)

* [Conda for data scientists](../../../../../user-guide/concepts/data-science.html)

* [Plugins](../../../../../user-guide/concepts/conda-plugins.html)

* [Troubleshooting](../../../../../user-guide/troubleshooting.html)

* [Cheatsheet](../../../../../user-guide/cheatsheet.html)

* [Configuration](../../../../../configuration.html)

* [Commands](../../../../../commands/index.html) __

* [` conda clean`](../../../../../commands/clean.html)

* [`conda compare`](../../../../../commands/compare.html)

* [`conda config`](../../../../../commands/config.html)

* [`conda create`](../../../../../commands/create.html)

* [`conda doctor`](../../../../../commands/doctor.html)

* [`conda env`](../../../../../commands/env/index.html) __

* [` conda env config`](../../../../../commands/env/config/index.html) __

        * [` conda env config vars`](../../../../../commands/env/config/vars/index.html) __

          * [` conda env config vars list`](../../../../../commands/env/config/vars/list.html)

          * [`conda env config vars set`](../../../../../commands/env/config/vars/set.html)

          * [`conda env config vars unset`](../../../../../commands/env/config/vars/unset.html)

      * [`conda env create`](../../../../../commands/env/create.html)

      * [`conda env export`](../../../../../commands/env/export.html)

      * [`conda env list`](../../../../../commands/env/list.html)

      * [`conda env remove`](../../../../../commands/env/remove.html)

      * [`conda env update`](../../../../../commands/env/update.html)

    * [`conda info`](../../../../../commands/info.html)

    * [`conda init`](../../../../../commands/init.html)

    * [`conda install`](../../../../../commands/install.html)

    * [`conda list`](../../../../../commands/list.html)

    * [`conda notices`](../../../../../commands/notices.html)

    * [`conda package`](../../../../../commands/package.html)

    * [`conda remove`](../../../../../commands/remove.html)

    * [`conda rename`](../../../../../commands/rename.html)

    * [`conda run`](../../../../../commands/run.html)

    * [`conda search`](../../../../../commands/search.html)

    * [`conda update`](../../../../../commands/update.html)

  * [Release notes](../../../../../release-notes.html)

  * [Glossary](../../../../../glossary.html)

  * [Developer guide](../../../../index.html) __

    * [Architecture](../../../../architecture.html)

    * [Contributing to conda](../../../../contributing.html)

    * [Development Environment](../../../../development-environment.html)

    * [Deep dives](../../../../deep-dives/index.html) __

  * [` conda install`](../../../../deep-dives/install.html)
  * [`conda init` and `conda activate`](../../../../deep-dives/activation.html)
  * [`conda config` and context](../../../../deep-dives/context.html)
  * [Solvers](../../../../deep-dives/solvers.html)
  * [Logging](../../../../deep-dives/logging.html)
* [Writing Tests](../../../../writing-tests/index.html) __
  * [Integration Tests](../../../../writing-tests/integration-tests.html)
* [Deprecations](../../../../deprecations.html)
* [Releasing](../../../../releasing.html)
* [Plugins](../../../../plugins/index.html) __
  * [Auth Handlers](../../../../plugins/auth_handlers.html)
  * [Health Checks](../../../../plugins/health_checks.html)
  * [Request Headers](../../../../plugins/request_headers.html)
  * [Post-commands](../../../../plugins/post_commands.html)
  * [Pre-commands](../../../../plugins/pre_commands.html)
  * [Reporter Backends](../../../../plugins/reporter_backends.html)
  * [Settings](../../../../plugins/settings.html)
  * [Solvers](../../../../plugins/solvers.html)
  * [Subcommands](../../../../plugins/subcommands.html)
  * [Virtual Packages](../../../../plugins/virtual_packages.html)
* [Specifications](../../../../specs/index.html) __
  * [Technical specification: solver state](../../../../specs/solver-state.html)
* [API](../../../../api.html) __
  * [` conda`](../../index.html) __
    * [` __main__`](../../__main__/index.html)
    * [`_vendor`](../../_vendor/index.html) __
      * [` frozendict`](../../_vendor/frozendict/index.html)

* [`_version`](../../_version/index.html)
* [`activate`](../../activate/index.html)
* [`api`](../../api/index.html)
* [`auxlib`](../../auxlib/index.html) __
  * [` collection`](../../auxlib/collection/index.html)
  * [`compat`](../../auxlib/compat/index.html)
  * [`decorators`](../../auxlib/decorators/index.html)
  * [`entity`](../../auxlib/entity/index.html)
  * [`exceptions`](../../auxlib/exceptions/index.html)
  * [`ish`](../../auxlib/ish/index.html)
  * [`logz`](../../auxlib/logz/index.html)
  * [`type_coercion`](../../auxlib/type_coercion/index.html)
* [`base`](../../base/index.html) __
  * [` constants`](../../base/constants/index.html)
  * [`context`](../../base/context/index.html)
* [`cli`](../../cli/index.html) __
  * [` actions`](../../cli/actions/index.html)
  * [`common`](../../cli/common/index.html)
  * [`conda_argparse`](../../cli/conda_argparse/index.html)
  * [`find_commands`](../../cli/find_commands/index.html)
  * [`helpers`](../../cli/helpers/index.html)
  * [`install`](../../cli/install/index.html)
  * [`main`](../../cli/main/index.html)
  * [`main_clean`](../../cli/main_clean/index.html)
  * [`main_commands`](../../cli/main_commands/index.html)
  * [`main_compare`](../../cli/main_compare/index.html)
  * [`main_config`](../../cli/main_config/index.html)

  * [`main_create`](../../cli/main_create/index.html)
  * [`main_env`](../../cli/main_env/index.html)
  * [`main_env_config`](../../cli/main_env_config/index.html)
  * [`main_env_create`](../../cli/main_env_create/index.html)
  * [`main_env_export`](../../cli/main_env_export/index.html)
  * [`main_env_list`](../../cli/main_env_list/index.html)
  * [`main_env_remove`](../../cli/main_env_remove/index.html)
  * [`main_env_update`](../../cli/main_env_update/index.html)
  * [`main_env_vars`](../../cli/main_env_vars/index.html)
  * [`main_export`](../../cli/main_export/index.html)
  * [`main_info`](../../cli/main_info/index.html)
  * [`main_init`](../../cli/main_init/index.html)
  * [`main_install`](../../cli/main_install/index.html)
  * [`main_list`](../../cli/main_list/index.html)
  * [`main_mock_activate`](../../cli/main_mock_activate/index.html)
  * [`main_mock_deactivate`](../../cli/main_mock_deactivate/index.html)
  * [`main_notices`](../../cli/main_notices/index.html)
  * [`main_package`](../../cli/main_package/index.html)
  * [`main_pip`](../../cli/main_pip/index.html)
  * [`main_remove`](../../cli/main_remove/index.html)
  * [`main_rename`](../../cli/main_rename/index.html)
  * [`main_run`](../../cli/main_run/index.html)
  * [`main_search`](../../cli/main_search/index.html)
  * [`main_update`](../../cli/main_update/index.html)
  * [`python_api`](../../cli/python_api/index.html)

* [`common`](../index.html) __
  * [` _logic`](../_logic/index.html)

* [`_os`](../_os/index.html) __
  * [` linux`](../_os/linux/index.html)
  * [`osx`](../_os/osx/index.html)
  * [`unix`](../_os/unix/index.html)
  * [`windows`](../_os/windows/index.html)
* [`compat`](../compat/index.html)
* [`configuration`](../configuration/index.html)
* [`constants`](../constants/index.html)
* [`disk`](../disk/index.html)
* `io`
* [`iterators`](../iterators/index.html)
* [`logic`](../logic/index.html)
* [`path`](../path/index.html) __
  * [` _cygpath`](../path/_cygpath/index.html)
  * [`directories`](../path/directories/index.html)
  * [`python`](../path/python/index.html)
  * [`windows`](../path/windows/index.html)
* [`pkg_formats`](../pkg_formats/index.html) __
  * [` python`](../pkg_formats/python/index.html)
* [`serialize`](../serialize/index.html)
* [`signals`](../signals/index.html)
* [`toposort`](../toposort/index.html)
* [`url`](../url/index.html)

* [`core`](../../core/index.html) __
  * [` envs_manager`](../../core/envs_manager/index.html)
  * [`index`](../../core/index/index.html)
  * [`initialize`](../../core/initialize/index.html)

  * [`link`](../../core/link/index.html)
  * [`package_cache_data`](../../core/package_cache_data/index.html)
  * [`path_actions`](../../core/path_actions/index.html)
  * [`portability`](../../core/portability/index.html)
  * [`prefix_data`](../../core/prefix_data/index.html)
  * [`solve`](../../core/solve/index.html)
  * [`subdir_data`](../../core/subdir_data/index.html)
* [`deprecations`](../../deprecations/index.html)
* [`env`](../../env/index.html) __
  * [` env`](../../env/env/index.html)
  * [`installers`](../../env/installers/index.html) __
    * [` base`](../../env/installers/base/index.html)
    * [`conda`](../../env/installers/conda/index.html)
    * [`pip`](../../env/installers/pip/index.html)
  * [`pip_util`](../../env/pip_util/index.html)
  * [`specs`](../../env/specs/index.html) __
    * [` binstar`](../../env/specs/binstar/index.html)
    * [`requirements`](../../env/specs/requirements/index.html)
    * [`yaml_file`](../../env/specs/yaml_file/index.html)
* [`exception_handler`](../../exception_handler/index.html)
* [`exceptions`](../../exceptions/index.html)
* [`exports`](../../exports/index.html)
* [`gateways`](../../gateways/index.html) __
  * [` anaconda_client`](../../gateways/anaconda_client/index.html)
  * [`connection`](../../gateways/connection/index.html) __
    * [` adapters`](../../gateways/connection/adapters/index.html) __
      * [` ftp`](../../gateways/connection/adapters/ftp/index.html)

        * [`http`](../../gateways/connection/adapters/http/index.html)
        * [`localfs`](../../gateways/connection/adapters/localfs/index.html)
        * [`s3`](../../gateways/connection/adapters/s3/index.html)
      * [`download`](../../gateways/connection/download/index.html)
      * [`session`](../../gateways/connection/session/index.html)
    * [`disk`](../../gateways/disk/index.html) __
      * [` create`](../../gateways/disk/create/index.html)
      * [`delete`](../../gateways/disk/delete/index.html)
      * [`link`](../../gateways/disk/link/index.html)
      * [`lock`](../../gateways/disk/lock/index.html)
      * [`permissions`](../../gateways/disk/permissions/index.html)
      * [`read`](../../gateways/disk/read/index.html)
      * [`test`](../../gateways/disk/test/index.html)
      * [`update`](../../gateways/disk/update/index.html)
    * [`logging`](../../gateways/logging/index.html)
    * [`repodata`](../../gateways/repodata/index.html) __
      * [` jlap`](../../gateways/repodata/jlap/index.html) __
        * [` core`](../../gateways/repodata/jlap/core/index.html)
        * [`fetch`](../../gateways/repodata/jlap/fetch/index.html)
        * [`interface`](../../gateways/repodata/jlap/interface/index.html)
      * [`lock`](../../gateways/repodata/lock/index.html)
    * [`subprocess`](../../gateways/subprocess/index.html)
  * [`history`](../../history/index.html)
  * [`instructions`](../../instructions/index.html)
  * [`misc`](../../misc/index.html)
  * [`models`](../../models/index.html) __
    * [` channel`](../../models/channel/index.html)

  * [`dist`](../../models/dist/index.html)
  * [`enums`](../../models/enums/index.html)
  * [`leased_path_entry`](../../models/leased_path_entry/index.html)
  * [`match_spec`](../../models/match_spec/index.html)
  * [`package_info`](../../models/package_info/index.html)
  * [`prefix_graph`](../../models/prefix_graph/index.html)
  * [`records`](../../models/records/index.html)
  * [`version`](../../models/version/index.html)
* [`notices`](../../notices/index.html) __
  * [` cache`](../../notices/cache/index.html)
  * [`core`](../../notices/core/index.html)
  * [`fetch`](../../notices/fetch/index.html)
  * [`types`](../../notices/types/index.html)
  * [`views`](../../notices/views/index.html)
* [`plan`](../../plan/index.html)
* [`plugins`](../../plugins/index.html) __
  * [` hookspec`](../../plugins/hookspec/index.html)
  * [`manager`](../../plugins/manager/index.html)
  * [`post_solves`](../../plugins/post_solves/index.html) __
    * [` signature_verification`](../../plugins/post_solves/signature_verification/index.html)
  * [`reporter_backends`](../../plugins/reporter_backends/index.html) __
    * [` console`](../../plugins/reporter_backends/console/index.html)
    * [`json`](../../plugins/reporter_backends/json/index.html)
  * [`solvers`](../../plugins/solvers/index.html)
  * [`subcommands`](../../plugins/subcommands/index.html) __
    * [` doctor`](../../plugins/subcommands/doctor/index.html) __
      * [` health_checks`](../../plugins/subcommands/doctor/health_checks/index.html)

    * [`types`](../../plugins/types/index.html)
    * [`virtual_packages`](../../plugins/virtual_packages/index.html) __
      * [` archspec`](../../plugins/virtual_packages/archspec/index.html)
      * [`conda`](../../plugins/virtual_packages/conda/index.html)
      * [`cuda`](../../plugins/virtual_packages/cuda/index.html)
      * [`freebsd`](../../plugins/virtual_packages/freebsd/index.html)
      * [`linux`](../../plugins/virtual_packages/linux/index.html)
      * [`osx`](../../plugins/virtual_packages/osx/index.html)
      * [`windows`](../../plugins/virtual_packages/windows/index.html)
  * [`reporters`](../../reporters/index.html)
  * [`resolve`](../../resolve/index.html)
  * [`testing`](../../testing/index.html) __
    * [` cases`](../../testing/cases/index.html)
    * [`fixtures`](../../testing/fixtures/index.html)
    * [`gateways`](../../testing/gateways/index.html) __
      * [` fixtures`](../../testing/gateways/fixtures/index.html)
    * [`helpers`](../../testing/helpers/index.html)
    * [`integration`](../../testing/integration/index.html)
    * [`notices`](../../testing/notices/index.html) __
      * [` fixtures`](../../testing/notices/fixtures/index.html)
      * [`helpers`](../../testing/notices/helpers/index.html)
    * [`solver_helpers`](../../testing/solver_helpers/index.html)
  * [`trust`](../../trust/index.html) __
    * [` constants`](../../trust/constants/index.html)
    * [`signature_verification`](../../trust/signature_verification/index.html)
  * [`utils`](../../utils/index.html)
* [`conda_env`](../../../conda_env/index.html) __

* [` cli`](../../../conda_env/cli/index.html)
* [`installers`](../../../conda_env/installers/index.html)

* [__](../../../../../index.html)
* [Developer guide](../../../../index.html)
* __
* [` common`](../index.html)
*

[`io`](https://docs.python.org/3/library/io.html#module-io "\(in Python v3.13\)")#

Common I/O utilities.

Classes#

`DeltaSecondsFormatter`	Logging formatter with additional attributes for run time logging.
`ContextDecorator`	Base class for a context manager class (implementing __enter__() and __exit__()) that also
`SwallowBrokenPipe`	Base class for a context manager class (implementing __enter__() and __exit__()) that also
`CaptureTarget`	Constants used for contextmanager captured.
`Spinner`
`ProgressBar`
`DummyExecutor`	This is an abstract base class for concrete asynchronous executors.
`ThreadLimitedThreadPoolExecutor`	This is an abstract base class for concrete asynchronous

executors.

`time_recorder` | Base class for a context manager class (implementing __enter__() and __exit__()) that also

Functions#

`dashlist`(iterable[, indent]) |

---|---

`env_vars`([var_map, callback, stack_callback]) |

`env_var`(name, value[, callback, stack_callback]) |

`env_unmodified`([callback]) |

`captured`([stdout, stderr]) | Capture outputs of sys.stdout and sys.stderr.

`argv`(args_list) |

`_logger_lock`() |

`disable_logger`(logger_name) |

`stderr_log_level`(level[, logger_name]) |

`attach_stderr_handler`([level, logger_name, propagate, ...]) | Attach a new stderr handler to the given logger and configure both.

`timeout`(timeout_secs, func, *args[, default_return]) | Enforce a maximum time for a callable to complete.

`get_instrumentation_record_file`() |

`print_instrumentation_data`() |

Attributes#

`IS_INTERACTIVE` |

---|---

`_FORMATTER` |

`swallow_broken_pipe` |

`as_completed` |

IS_INTERACTIVE#

_class _DeltaSecondsFormatter(_fmt =None_, _datefmt =None_)#

Bases:

[`logging.Formatter`](https://docs.python.org/3/library/logging.html#logging.Formatter

"\(in Python v3.13\)")

Logging formatter with additional attributes for run time logging.

`delta_secs`

Elapsed seconds since last log/format call (or creation of logger).

`relative_created_secs`

Like relativeCreated, time relative to the initialization of the logging
module but conveniently scaled to seconds as a float value.

Initialize the formatter with specified format strings.

Initialize the formatter either with the specified format string, or a default
as described above. Allow for specialized date formatting with the optional
datefmt argument. If datefmt is omitted, you get an ISO8601-like (or RFC
3339-like) format.

Use a style parameter of '%', '{' or '$' to specify that you want to use one
of %-formatting,
[`str.format()`](https://docs.python.org/3/library/stdtypes.html#str.format
"\(in Python v3.13\)") (`{}`) formatting or
[`string.Template`](https://docs.python.org/3/library/string.html#string.Template
"\(in Python v3.13\)") formatting in your format string.

Changed in version 3.2: Added the `style` parameter.

format(_record_)#

Format the specified record as text.

The record's attribute dictionary is used as the operand to a string

formatting operation which yields the returned string. Before formatting the dictionary, a couple of preparatory steps are carried out. The message attribute of the record is computed using LogRecord.getMessage(). If the formatting string uses the time (as determined by a call to usesTime(), formatTime() is called to format the event time. If there is exception information, it is formatted using formatException() and appended to the message.

_FORMATTER#

dashlist(_iterable_ , _indent =2_)#

_class _ContextDecorator#

Base class for a context manager class (implementing __enter__() and __exit__()) that also makes it a decorator.

__call__(_f_)#

_class _SwallowBrokenPipe#

Bases: `ContextDecorator`

Base class for a context manager class (implementing __enter__() and

__exit__()) that also makes it a decorator.

__enter__()#

__exit__(_exc_type_ , _exc_val_ , _exc_tb_)#

swallow_broken_pipe#

_class _CaptureTarget#

Bases: [`enum.Enum`](https://docs.python.org/3/library/enum.html#enum.Enum

"\(in Python v3.13\)")

Constants used for contextmanager captured.

Used similarly like the constants PIPE, STDOUT for stdlib's subprocess.Popen.

STRING#

STDOUT#

env_vars(_var_map =None_, _callback =None_, _stack_callback =None_)#

env_var(_name_ , _value_ , _callback =None_, _stack_callback =None_)#

env_unmodified(_callback =None_)#

captured(_stdout =CaptureTarget.STRING_, _stderr =CaptureTarget.STRING_)#

Capture outputs of sys.stdout and sys.stderr.

If stdout is STRING, capture sys.stdout as a string, if stdout is None, do not capture sys.stdout, leaving it untouched, otherwise redirect sys.stdout to the file-like object given by stdout.

Behave correspondingly for stderr with the exception that if stderr is STDOUT, redirect sys.stderr to stdout target and set stderr attribute of yielded object to None.

```
>>> from conda.common.io import captured
>>> with captured() as c:
...     print("hello world!")
...
>>> c.stdout
'hello world!\n'
```

Parameters:

* **stdout** \-- capture target for sys.stdout, one of STRING, None, or file-like object

* **stderr** \-- capture target for sys.stderr, one of STRING, STDOUT, None, or file-like object

Yields:

CapturedText \--

has attributes stdout, stderr which are either strings, None or the

corresponding file-like function argument.

argv(_args_list_)#

_logger_lock()#

disable_logger(_logger_name_)#

stderr_log_level(_level_ , _logger_name =None_)#

attach_stderr_handler(_level =WARN_, _logger_name =None_, _propagate =False_, _formatter =None_, _filters =None_)#

Attach a new stderr handler to the given logger and configure both.

This function creates a new StreamHandler that writes to stderr and attaches it to the logger given by logger_name (which maybe None, in which case the root logger is used). If the logger already has a handler by the name of stderr, it is removed first.

The given level is set **for the handler** , not for the logger; however, this function also sets the level of the given logger to the minimum of its current effective level and the new handler level, ensuring that the handler will receive the required log records, while minimizing the number of unnecessary log events. It also sets the loggers propagate property according to the propagate argument. The formatter argument can be used to set the formatter of the handler.

timeout(_timeout_secs_ , _func_ , _* args_, _default_return =None_, _** kwargs_)#

Enforce a maximum time for a callable to complete. Not yet implemented on

Windows.

_class _Spinner(_message_ , _enabled =True_, _json =False_, _fail_message

='failed\n'_)#

Parameters:

  * **message** ([_str_](https://docs.python.org/3/library/stdtypes.html#str "\(in Python v3.13\)")) -- A message to prefix the spinner with. The string ': ' is automatically appended.

  * **enabled** ([_bool_](https://docs.python.org/3/library/functions.html#bool "\(in Python v3.13\)")) -- If False, usage is a no-op.

  * **json** ([_bool_](https://docs.python.org/3/library/functions.html#bool "\(in Python v3.13\)")) -- If True, will not output non-json to stdout.

spinner_cycle#

start()#

stop()#

_start_spinning()#

__enter__()#

__exit__(_exc_type_ , _exc_val_ , _exc_tb_)#

_class _ProgressBar(_description_ , _enabled =True_, _json =False_, _position

=None_, _leave =True_)#

Parameters:

* **description** ([_str_](https://docs.python.org/3/library/stdtypes.html#str "\(in Python v3.13\)")) -- The name of the progress bar, shown on left side of output.

* **enabled** ([_bool_](https://docs.python.org/3/library/functions.html#bool "\(in Python v3.13\)")) -- If False, usage is a no-op.

* **json** ([_bool_](https://docs.python.org/3/library/functions.html#bool "\(in Python v3.13\)")) -- If true, outputs json progress to stdout rather than a progress bar. Currently, the json format assumes this is only used for "fetch", which maintains backward compatibility with conda 4.3 and earlier behavior.

_classmethod _get_lock()#

update_to(_fraction_)#

finish()#

refresh()#

Force refresh i.e. once 100% has been reached

close()#

_static __tqdm(_* args_, _** kwargs_)#

Deferred import so it doesn't hit the conda activate paths.

_class _DummyExecutor#

Bases:

[`concurrent.futures.Executor`](https://docs.python.org/3/library/concurrent.futures.html#concurrent.futures.Executor

"\(in Python v3.13\)")

This is an abstract base class for concrete asynchronous executors.

submit(_fn_ , _* args_, _** kwargs_)#

Submits a callable to be executed with the given arguments.

Schedules the callable to be executed as fn(*args, **kwargs) and returns a

Future instance representing the execution of the callable.

Returns:

A Future representing the given call.

map(_func_ , _* iterables_)#

Returns an iterator equivalent to map(fn, iter).

Parameters:

* **fn** \-- A callable that will take as many arguments as there are passed iterables.

* **timeout** \-- The maximum number of seconds to wait. If None, then there is no limit on the wait time.

* **chunksize** \-- The size of the chunks the iterable will be broken into before being passed to a child process. This argument is only used by ProcessPoolExecutor; it is ignored by ThreadPoolExecutor.

Returns:

map(func, *iterables) but the calls may be evaluated out-of-order.

Return type:

An iterator equivalent to

Raises:

* [**TimeoutError**](https://docs.python.org/3/library/exceptions.html#TimeoutError "\(in Python v3.13\)") \-- If the entire result iterator could not be generated before the given timeout.

* [**Exception**](https://docs.python.org/3/library/exceptions.html#Exception "\(in Python v3.13\)") \-- If fn(*args) raises for any values.

shutdown(_wait =True_)#

Clean-up the resources associated with the Executor.

It is safe to call this method several times. Otherwise, no other methods can
be called after this one.

Parameters:

* **wait** \-- If True then shutdown will not return until all running futures have finished executing and the resources used by the executor have been reclaimed.

* **cancel_futures** \-- If True then shutdown will cancel all pending futures. Futures that are completed or running will not be cancelled.

_class _ThreadLimitedThreadPoolExecutor(_max_workers =10_)#

Bases:
[`concurrent.futures.ThreadPoolExecutor`](https://docs.python.org/3/library/concurrent.futures.html#
concurrent.futures.ThreadPoolExecutor
"\(in Python v3.13\)")

This is an abstract base class for concrete asynchronous executors.

Initializes a new ThreadPoolExecutor instance.

Parameters:

* **max_workers** \-- The maximum number of threads that can be used to execute the given calls.

* **thread_name_prefix** \-- An optional name prefix to give our threads.

* **initializer** \-- A callable used to initialize worker threads.

* **initargs** \-- A tuple of arguments to pass to the initializer.

submit(_fn_ , _* args_, _** kwargs_)#

This is an exact reimplementation of the submit() method on the parent class,
except with an added try/except around self._adjust_thread_count(). So long as
there is at least one living thread, this thread pool will not throw an
exception if threads cannot be expanded to max_workers.

In the implementation, we use "protected" attributes from concurrent.futures

(_base and _WorkItem). Consider vendoring the whole concurrent.futures library

as an alternative to these protected imports.

[agronholm/pythonfutures](https://github.com/agronholm/pythonfutures/blob/3.2.0/concurrent/futures/thread.py#L121-L131)

NOQA

[python/cpython](https://github.com/python/cpython/blob/v3.6.4/Lib/concurrent/futures/thread.py#L114-L124)

as_completed#

get_instrumentation_record_file()#

_class _time_recorder(_entry_name =None_, _module_name =None_)#

Bases: `ContextDecorator`

Base class for a context manager class (implementing __enter__() and

__exit__()) that also makes it a decorator.

record_file#

start_time#

total_call_num#

total_run_time#

_set_entry_name(_f_)#

__call__(_f_)#

__enter__()#

__exit__(_exc_type_ , _exc_val_ , _exc_tb_)#

_classmethod _log_totals()#

_ensure_dir()#

print_instrumentation_data()#

__On this page

  * `ContextDecorator.__call__()`
* `SwallowBrokenPipe`
  * `SwallowBrokenPipe.__enter__()`
  * `SwallowBrokenPipe.__exit__()`
* `swallow_broken_pipe`
* `CaptureTarget`
  * `CaptureTarget.STRING`
  * `CaptureTarget.STDOUT`
* `env_vars()`
* `env_var()`
* `env_unmodified()`
* `captured()`
* `argv()`
* `_logger_lock()`
* `disable_logger()`
* `stderr_log_level()`
* `attach_stderr_handler()`
* `timeout()`
* `Spinner`
  * `Spinner.spinner_cycle`
  * `Spinner.start()`
  * `Spinner.stop()`
  * `Spinner._start_spinning()`
  * `Spinner.__enter__()`
  * `Spinner.__exit__()`
* `ProgressBar`
  * `ProgressBar.get_lock()`

    * `ProgressBar.update_to()`
    * `ProgressBar.finish()`
    * `ProgressBar.refresh()`
    * `ProgressBar.close()`
    * `ProgressBar._tqdm()`
  * `DummyExecutor`
    * `DummyExecutor.submit()`
    * `DummyExecutor.map()`
    * `DummyExecutor.shutdown()`
  * `ThreadLimitedThreadPoolExecutor`
    * `ThreadLimitedThreadPoolExecutor.submit()`
  * `as_completed`
  * `get_instrumentation_record_file()`
  * `time_recorder`
    * `time_recorder.record_file`
    * `time_recorder.start_time`
    * `time_recorder.total_call_num`
    * `time_recorder.total_run_time`
    * `time_recorder._set_entry_name()`
    * `time_recorder.__call__()`
    * `time_recorder.__enter__()`
    * `time_recorder.__exit__()`
    * `time_recorder.log_totals()`
    * `time_recorder._ensure_dir()`
  * `print_instrumentation_data()`

[__Edit on GitHub](https://github.com/conda/conda/edit/main/docs/source/dev-

guide/api/conda/common/io/index.rst)

[__Show Source](../../../../../_sources/dev-guide/api/conda/common/io/index.rst.txt)