CuttlePool Documentation

Release 0.9.1

Spencer Mitchell

Sep 23, 2020

Contents:

CHAPTER 1

API

class cuttlepool.**CuttlePool**(*factory*, *capacity*, *overflow=0*, *timeout=None*, *resource_wrapper=None*, ***kwargs*)

A resource pool.

Parameters

- **factory** (*func*) – A factory that produces the desired resource.
- **capacity** (*int*) – Max number of resource instances in the pool.
- **overflow** (*int*) – The number of extra resource instances that can be made if the pool is exhausted. Defaults to `0`.
- **timeout** (*int*) – Time in seconds to wait for a resource. Defaults to `None`.
- **resource_wrapper** – A Resource subclass.
- ****kwargs** – Keyword arguments that are passed to `factory` when a resource instance is created.

Raises

- **ValueError** – If capacity <= 0 or overflow < 0 or timeout < 0.
- **TypeError** – If timeout is not int or `None`.

capacity
The maximum capacity the pool will hold under normal circumstances.

connection_arguments
For compatibility with older versions, will be removed in 1.0.

empty()
Return `True` if pool is empty.

factory_arguments
Return a copy of the factory arguments used to create a resource.

get_connection(*connection_wrapper=None*)
For compatibility with older versions, will be removed in 1.0.

get_resource(*resource_wrapper=None*)
Returns a `Resource` instance.

Parameters **resource_wrapper** – A Resource subclass.

Returns A `Resource` instance.

Raises ***PoolEmptyError*** – If attempt to get resource fails or times out.

maxsize
The maximum possible number of resource instances that can exist at any one time.

normalize_connection(*connection*)
For compatibility with older versions, will be removed in 1.0.

normalize_resource(*resource*)
A user implemented function that resets the properties of the resource instance that was created by *factory*. This prevents unwanted behavior from a resource retrieved from the pool as it could have been changed when previously used.

Parameters **resource** (*obj*) – A resource instance.

overflow
The number of additional resource instances the pool will create when it is at capacity.

ping(*resource*)
A user implemented function that ensures the `Resource` object is open.

Parameters **resource** (*obj*) – A `Resource` object.

Returns A bool indicating if the resource is open (`True`) or closed (`False`).

put_connection(*connection*)
For compatibility with older versions, will be removed in 1.0.

put_resource(*resource*)
Adds a resource back to the pool or discards it if the pool is full.

Parameters **resource** – A resource object.

Raises ***UnknownResourceError*** – If resource was not made by the pool.

size
The number of existing resource instances that have been made by the pool.

Note This is not the number of resources *in* the pool, but the number of existing resources. This includes resources in the pool and resources in use.

Warning: This is not threadsafe. `size` can change when context switches to another thread.

timeout
The duration to wait for a resource to be returned to the pool when the pool is depleted.

class `cuttlepool.`**Resource**(*resource*, *pool*)
A wrapper around a resource instance.

Parameters

- **resource** – A resource instance.
- **pool** – A resource pool.

close()
Returns the resource to the resource pool.

exception cuttlepool.**CuttlePoolError**
Base class for exceptions in this module.

exception cuttlepool.**PoolEmptyError**
Exception raised when pool timeouts.

exception cuttlepool.**PoolFullError**
Exception raised when there is no space to add a resource.

exception cuttlepool.**UnknownResourceError**
Exception raised when a resource is returned to the pool that was not made by the pool.

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

C

Index