
Pipecat Documentation

Release 0.1.0-dev

Timothy M. Shead

December 02, 2016

1	API Reference	3
1.1	pipecat module	3
1.2	pipecat.device module	3
1.3	pipecat.device.gps module	3
1.4	pipecat.limit module	3
1.5	pipecat.queue module	4
1.6	pipecat.record module	4
1.7	pipecat.source module	4
1.8	pipecat.source.charger module	4
1.9	pipecat.source.time module	5
1.10	pipecat.storage module	5
1.11	pipecat.udp module	5
2	Indices and tables	7
	Python Module Index	9

Contents:



Contents:

1.1 pipecat module

Data logging library.

1.2 pipecat.device module

1.3 pipecat.device.gps module

Functions for working with UDP messages.

`pipecat.device.gps.nmea` (*source*)
Parse NMEA messages from raw strings.

1.4 pipecat.limit module

Options to constrain the information that is logged.

`pipecat.limit.count` (*source, count*)
Limits the number of records returned from another source.

`pipecat.limit.duration` (*source, duration, timeout=<Quantity(0.1, 'second')>*)
Return records from another source until a fixed time duration has expired.

`pipecat.limit.timeout` (*source, timeout, initial=<Quantity(1, 'hour')>*)
Return records from another source until they stop arriving.

Parameters

- **source** (*generator, required*) – A source of records.
- **timeout** (*quantity, required*) – Maximum time to wait for the next record.
- **initial** (*quantity, optional*) – Maximum time to wait for the first record.

1.5 pipecat.queue module

Communication with Queue.Queue.

`pipecat.queue.receive(queue)`
Receive records from a queue.

`pipecat.queue.send(source, queue)`
Send records from a source to a queue.

1.6 pipecat.record module

`pipecat.record.add_field(record, key, value)`
Add a key-value pair to a record.

Parameters

- **record** (*dict, required*) – Dictionary of key-value pairs that constitute a record.
- **key** (*string or tuple of strings, required*) – Record key to be overwritten.
- **value** (*object*) – New record value.

`pipecat.record.dump(record, fobj=<open file '<stdout>', mode 'w'>)`
Dump a human-readable text representation of a record to a file-like object.

Parameters

- **record** (*dict, required*) – Dictionary of key-value pairs to be written-out.
- **fobj** (*file-like object, optional*)

1.7 pipecat.source module

Data sources from which information can be logged.

`pipecat.source.add(source, key, value)`
Adds a key-value pair to every record returned from another source.

`pipecat.source.concatenate(sources)`
Concatenate records from multiple sources.

`pipecat.source.multiplex(*sources)`
Interleave records from multiple sources.

`pipecat.source.trace(source)`
Log records for debugging.

1.8 pipecat.source.charger module

Data sources that retrieve information from battery chargers.

`pipecat.source.charger.icharger208b(fobj)`
Read data from an iCharger 208B battery charger.

Logs data events emitted by the charger during charge, discharge, etc.

Parameters `fobj` (*file-like object, typically an instance of `serial.Stream`*)

1.9 pipecat.source.time module

Data sources that provide information related to time.

`pipecat.source.time.metronome` (*rate=<Quantity(1.0, 'second')>*)
Generate an empty record at fixed time intervals.

`pipecat.source.time.timestamp` (*source, key='timestamp'*)
Add a timestamp to every record returned from another source.

1.10 pipecat.storage module

Functions for storing data.

class `pipecat.storage.Cache` (*source*)
Bases: `object`

Cache records in memory for column-oriented access.

next ()

table

class `pipecat.storage.Table`
Bases: `object`

append (*record*)

items ()

keys ()

values ()

`pipecat.storage.cache` (*source*)
Create an in-memory cache for records.

Parameters `source` (*generator, required*) – A source of records to be cached.

Returns `cache`

Return type instance of `pipecat.storage.Cache`.

`pipecat.storage.csv` (*source, fobj*)
Append records to a CSV file.

1.11 pipecat.udp module

Functions for working with UDP messages.

`pipecat.udp.receive` (*address, maxsize*)
Receive messages from a UDP socket.

Indices and tables

- `genindex`
- `modindex`
- `search`

p

- pipecat, 3
- pipecat.device, 3
- pipecat.device.gps, 3
- pipecat.limit, 3
- pipecat.queue, 4
- pipecat.record, 4
- pipecat.source, 4
- pipecat.source.charger, 4
- pipecat.source.time, 5
- pipecat.storage, 5
- pipecat.udp, 5

A

add() (in module `pipecat.source`), 4
add_field() (in module `pipecat.record`), 4
append() (`pipecat.storage.Table` method), 5

C

Cache (class in `pipecat.storage`), 5
cache() (in module `pipecat.storage`), 5
concatenate() (in module `pipecat.source`), 4
count() (in module `pipecat.limit`), 3
csv() (in module `pipecat.storage`), 5

D

dump() (in module `pipecat.record`), 4
duration() (in module `pipecat.limit`), 3

I

icharger208b() (in module `pipecat.source.charger`), 4
items() (`pipecat.storage.Table` method), 5

K

keys() (`pipecat.storage.Table` method), 5

M

metronome() (in module `pipecat.source.time`), 5
multiplex() (in module `pipecat.source`), 4

N

next() (`pipecat.storage.Cache` method), 5
nmea() (in module `pipecat.device.gps`), 3

P

pipecat (module), 3
pipecat.device (module), 3
pipecat.device.gps (module), 3
pipecat.limit (module), 3
pipecat.queue (module), 4
pipecat.record (module), 4
pipecat.source (module), 4

pipecat.source.charger (module), 4
pipecat.source.time (module), 5
pipecat.storage (module), 5
pipecat.udp (module), 5

R

receive() (in module `pipecat.queue`), 4
receive() (in module `pipecat.udp`), 5

S

send() (in module `pipecat.queue`), 4

T

Table (class in `pipecat.storage`), 5
table (`pipecat.storage.Cache` attribute), 5
timeout() (in module `pipecat.limit`), 3
timestamp() (in module `pipecat.source.time`), 5
trace() (in module `pipecat.source`), 4

V

values() (`pipecat.storage.Table` method), 5