
TwinDB Backup Documentation

Release 2.18.0

TwinDB Development Team

Apr 18, 2019

Contents

1	TwinDB Backup	3
1.1	Features	3
1.2	Credits	4
2	Installation	5
2.1	Requirements	5
2.2	Stable release	5
2.3	From sources	5
3	Usage	7
3.1	Configuration	7
3.2	Compression	8
3.3	Email notification	10
4	twindb_backup	11
4.1	twindb_backup package	11
5	Contributing	41
5.1	Types of Contributions	41
5.2	Get Started!	42
5.3	Pull Request Guidelines	43
6	Credits	45
6.1	Development Lead	45
6.2	Contributors	45
7	History	47
8	Indices and tables	49
	Python Module Index	51

Contents:

TwinDB Backup is a multipurpose tool for backing up MySQL and file system. It can store backup copies on a remote SSH server, Amazon S3 or Google Cloud Storage.

The tool can easily restore the backup copies.

Read full documentation on <https://twindb-backup.readthedocs.io>.

1.1 Features

twindb-backup key features:

- Files/directories backups
- MySQL backups
- Incremental MySQL backups
- Encrypting backup copies

twindb-backup store backups on:

- Remote SSH server
- Amazon S3
- Google Cloud Storage
- Optionally save local copy

Other features:


- Retention policy defines how many hourly/daily/weekly/monthly/yearly copies to keep

- Separate retention policy for remote and local backup copies
- Supports non-impacting Percona XtraDB Cluster backups
- Email notifications
- cron configuration comes with a package


1.1.1 How do I get set up?

twindb-backup is distributed via package repositories. See installation instruction on <https://packagecloud.io/twindb/main/install>. Once the repository for your operating system is configured, install the `twindb-backup` package.

On CentOS and RedHat

```
# curl -s https://packagecloud.io/install/repositories/twindb/main/script.rpm.sh |   
↪ sudo bash  
# yum install twindb-backup
```

On Debian and Ubuntu

```
# curl -s https://packagecloud.io/install/repositories/twindb/main/script.deb.sh |   
↪ sudo bash  
# apt-get install twindb-backup
```

1.1.2 Configuration

Configuration is stored in `/etc/twindb/twindb-backup.cfg`. See <http://twindb-backup.readthedocs.io/en/master/usage.html> for more details.

1.2 Credits

This package was created with [Cookiecutter](#) and the [audreyr/cookiecutter-pypackage](#) project template.

2.1 Requirements

TwinDB Backup package will pull all necessary dependencies except `aws` tool. We recommend to install it from PyPi.

```
# pip install awscli
```

2.2 Stable release

To install TwinDB Backup, run this command in your terminal:

```
# yum install https://twindb.com/twindb-release-latest.noarch.rpm
# yum install twindb-backup
```

This is the preferred method to install TwinDB Backup, as it will always install the most recent stable release.

2.3 From sources

The sources for TwinDB Backup can be downloaded from the [Github repo](#).

You can either clone the public repository:

```
$ git clone git://github.com/twindb/backup
```

Or download the tarball:

```
$ curl -OL https://github.com/twindb/backup/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ make install
```

3.1 Configuration

Once the `twindb-backup` package is installed you need to edit configuration file `/etc/twindb/twindb-backup.cfg`.

Let's review each configuration section.

3.1.1 Source

[`source`] section defines what to backup.

`twindb-backup` supports backing up local directories and MySQL database.

In `backup_dirs` you specify which directories to backup. Each directory is separated by a white space.

`backup_mysql` tells whether the tool backups MySQL or not.

```
[source]
backup_dirs=/etc /root /home
backup_mysql=no
```

3.1.2 Destination

[`destination`] specifies where to store backup copies.

`backup_destination` can be either `ssh` (if you want to store backups on a remote SSH server) or `s3` (if you want to store backups in Amazon S3).

In the optional `keep_local_path` you can specify a local path where the tool will store a local copy of the backup. It's useful if you want to stream a MySQL backup to S3 and would like to keep a local copy as well.

```
[destination]
backup_destination=ssh
keep_local_path=/var/backup/local
```

3.2 Compression

In `[compression]` section you can specify compression program such as `gzip`, `pigz`, `bzip2` and `lbzip2`. You can use parallel compression by using `pigz` or `lbzip2`, and specify number of threads to use in parallel. Number of threads defaults to number of cores minus one, if not specified. Level specifies the compression level from 1 to 9

```
[compression]
program=pigz
#threads=4
#level=9
```

3.2.1 Amazon S3

In `[s3]` section you specify Amazon credentials as well as an S3 bucket where to store backups.

```
[s3]
# S3 destination settings

AWS_ACCESS_KEY_ID=XXXXX
AWS_SECRET_ACCESS_KEY=YYYYY
AWS_DEFAULT_REGION=us-east-1
BUCKET=twindb-backups
```

3.2.2 Google Cloud Storage

In `[gcs]` section you specify Google credentials as well as cloud storage bucket where to store backups.

```
[gcs]
# GCS destination settings

GC_CREDENTIALS_FILE=XXXXX
GC_ENCRYPTION_KEY=
BUCKET=twindb-backups
```

3.2.3 SSH Settings

If your backup destination is an SSH server, you specify `ssh` parameters in `[ssh]` section. It is assumed you configured SSH keys authentication. It will not work if you need to enter a password to login to `backup_host`.

```
[ssh]

backup_host=127.0.0.1
backup_dir=/tmp/backup
```

(continues on next page)

(continued from previous page)

```
ssh_user=root
ssh_key=/root/.ssh/id_rsa
ssh_port=22
```

3.2.4 MySQL

XtraBackup needs to connect to MySQL. In `[mysql]` section you specify a defaults file with user and password. Besides this, You can specify period for MySQL full backup.

The `expire_log_days` options specifies for how many days the binlog should be kept. By default it's seven days.

```
[mysql]
mysql_defaults_file=/etc/twindb/my.cnf
full_backup=daily
expire_log_days=7
```

3.2.5 Encryption

The tool uses GPG for encrypting/decrypting backup copies. If you want to enable encryption add `[gpg]` section to the configuration file. It's your responsibility to generate and manage the encryption key.

```
[gpg]
keyring = /root/.gnupg/pubring.gpg
secret_keyring = /root/.gnupg/secring.gpg
recipient = backupuser@yourdomain.com
```

3.2.6 Retention Policy

In `[retention]` section you specify how many copies you want to keep on the remote storage (s3 or ssh).

```
[retention]
hourly_copies=24
daily_copies=7
weekly_copies=4
monthly_copies=12
yearly_copies=3
```

3.2.7 Local Retention Policy

if `keep_local_path` is defined in *Destination* the tool will apply `[retention_local]` on the local copies.

```
[retention_local]
hourly_copies=1
daily_copies=1
weekly_copies=0
monthly_copies=0
yearly_copies=0
```

3.2.8 Running Intervals

By default **twindb-backup** will run *hourly*, *daily*, *weekly*, *monthly* and *yearly*. If you would like to skip some runs [intervals] section is the right place to do so.

```
[intervals]
run_hourly=yes
run_daily=yes
run_weekly=yes
run_monthly=yes
run_yearly=yes
```

3.3 Email notification

The RPM package installs a cron job. If a backup job fails it will send standard error output to the specified email. The email address is specified in the cron configuration file `/etc/cron.d/twindb-backup`.

```
MAILTO=nagios@twindb.com
@hourly root twindb-backup backup hourly
@daily root twindb-backup backup daily
@weekly root twindb-backup backup weekly
@monthly root twindb-backup backup monthly
@yearly root twindb-backup backup yearly
```

4.1 twindb_backup package

4.1.1 Subpackages

twindb_backup.cache package

Submodules

twindb_backup.cache.cache module

Backup copy cache

class twindb_backup.cache.cache.Cache (*path*)

Bases: object

Class implements local cache to save full backup copies

add (*path*, *key=None*)

Add directory to cache. The directory may be a full or relative path with backup copy. The directory name must match with a file name of the backup copy. If backup copy is `/path/to/backups/master1/daily/mysql/mysql-2017-05-13_22_04_06.xbstream.gz`, then the directory can be something like `/var/tmp/mysql-2017-05-13_22_04_06.xbstream.gz/`.

Let's say we want to add `/var/tmp/mysql-2017-05-13_22_04_06.xbstream.gz/` to the cache in `/var/tmp/cache`. Then this method will create directory `/var/tmp/cache/mysql-2017-05-13_22_04_06.xbstream.gz/`.

If you want to save directory `/var/tmp/foo` in cache under a key name `mysql-2017-05-13_22_04_06.xbstream.gz` you need to specify the key e.g. `add('/var/tmp/cache', 'mysql-2017-05-13_22_04_06.xbstream.gz')`

Parameters

- **path** (*str*) – full or relative path

- **key** – if specified the directory will be added as this key name in the cache

Raise CacheException if errors

purge ()

Remove all entries from the cache

restore_in (*item*, *path*)

Restore backup copy item in path.

Parameters

- **item** (*str*) – directory in the cache
- **path** (*str*) – directory where to restore item

exception `twindb_backup.cache.cache.CacheException`

Bases: `exceptions.Exception`

Cache errors

Module contents

`twindb_backup.copy` package

Submodules

`twindb_backup.copy.base_copy` module

Base class for a backup copy

class `twindb_backup.copy.base_copy.BaseCopy` (*host*, *name*)

Bases: `object`

Base class for a backup copy in status

Parameters

- **host** (*str*) – Hostname where the backup was taken from.
- **name** (*str*) – Base name of the backup copy file as it's stored on the destination.

key

It's a relative path backup copy. It's relative to the remote path as from the twindb config. It's also a key in status, hence the name.

Returns Path to file

Return type `str`

Raises `UnknownSourceType` – If source type is not defined

`twindb_backup.copy.binlog_copy` module

Class to describe Binlog backup copy

class `twindb_backup.copy.binlog_copy.BinlogCopy` (*host*, *name*, *created_at*)

Bases: `twindb_backup.copy.base_copy.BaseCopy`

Instantiate a Binlog copy in status

Parameters

- **host** (*str*) – Hostname where the backup was taken from.
- **name** (*str*) – Base name of the backup copy file as it's stored on the destination.
- **created_at** (*int*) – Time when copy created

created_at

Time of created copy

name

Binlog copy name as in SHOW BINARY LOGS.

twindb_backup.copy.exceptions module

Module for Backup copy exception classes

exception `twindb_backup.copy.exceptions.BackupCopyError`

Bases: `twindb_backup.exceptions.TwinDBBackupError`

General backup copy error

exception `twindb_backup.copy.exceptions.UnknownSourceType`

Bases: `twindb_backup.copy.exceptions.BackupCopyError`

Raises when source type is not set

exception `twindb_backup.copy.exceptions.WrongInputData`

Bases: `twindb_backup.copy.exceptions.BackupCopyError`

Raises when incorrent inputs are used

twindb_backup.copy.mysql_copy module

Class to describe MySQL backup copy

class `twindb_backup.copy.mysql_copy.MySQLCopy(*args, **kwargs)`

Bases: `twindb_backup.copy.periodic_copy.PeriodicCopy`

Instantiate a MySQL copy.

Parameters

- **host** (*str*) – Hostname where the backup was taken from.
- **run_type** (*str*) – Run type when the backup was taken: daily, weekly, etc.
- **name** (*str*) – Base name of the backup copy file as it's stored on the destination.

Raises `WrongInputData` – if type is neither full or incremental, if name is not a basename.

as_dict()

Return representation of the class instance for output purposes.

backup_finished

Timestamp when the backup job finished.

backup_started

Timestamp when the backup job started.

binlog

File name of the binlog.

config

Dictionary of configs and their content.

created_at

Timestamp when the backup job started.

duration

Time in seconds it took to take the backup.

galera

True if the backup was taken from Galera.

lsn

LSN of the backup.

parent

For incremental backup it is a base full copy name.

position

Binlog position of the backup copy.

serialize ()

Prepare the status for storing as a string.

type

Full or incremental.

wsrep_provider_version

If it was Galera, value of wsrep_provider_version

twindb_backup.copy.periodic_copy module

Interval class for a backup copy

class twindb_backup.copy.periodic_copy.**PeriodicCopy** (*args, **kwargs)

Bases: *twindb_backup.copy.base_copy.BaseCopy*

Interval class for a periodic backup copy in status

Parameters

- **host** (*str*) – Hostname where the backup was taken from.
- **run_type** (*str*) – Run type when the backup was taken: daily, weekly, etc.
- **name** (*str*) – Base name of the backup copy file as it's stored on the destination.

host

Host where the backup was taken

name

Name of the backup. It's basename w/o directory part.

run_type

What run type was when the backup copy was taken.

Module contents

twindb_backup.destination package

Submodules

twindb_backup.destination.base_destination module

Module defines Base destination class and destination exception(s).

class twindb_backup.destination.base_destination.**BaseDestination** (*remote_path*)

Bases: object

Base destination class

delete (*path*)

Delete object from the destination

Parameters *path* – Relative path to the file to delete

get_stream (*copy*)

Get a PIPE handler with content of the backup copy streamed from the destination.

Parameters *copy* (*BaseCopy*) – Backup copy

Returns Standard output.

list_files (*prefix=None, recursive=False, pattern=None, files_only=False*)

Get list of file by prefix.

Parameters

- **prefix** (*str*) – Path
- **recursive** (*bool*) – Recursive return list of files
- **pattern** (*str*) – files must match with this regexp if specified
- **files_only** (*bool*) – If True don't list directories

Returns List of files

Return type list

read (*filepath*)

Read content of a file path from destination.

Parameters *filepath* (*str*) – Relative path to file.

Returns Content of the file.

Return type str

save (*handler, filepath*)

Save a stream given as handler to filepath.

Parameters

- **handler** (*file*) – Incoming stream.
- **filepath** (*str*) – Save stream as this name.

write (*content, filepath*)

Write string to a file.

Parameters

- **content** (*str*) – String.
- **filepath** (*str*) – Relative path to file.

twindb_backup.destination.exceptions module

Module for destination exceptions

exception `twindb_backup.destination.exceptions.DestinationError`

Bases: `twindb_backup.exceptions.TwinDBBackupError`

General destination error

exception `twindb_backup.destination.exceptions.FileNotFound`

Bases: `twindb_backup.destination.exceptions.DestinationError`

File doesn't exist on destination

exception `twindb_backup.destination.exceptions.GCSDestinationError`

Bases: `twindb_backup.destination.exceptions.DestinationError`

GCS destination errors

exception `twindb_backup.destination.exceptions.S3DestinationError`

Bases: `twindb_backup.destination.exceptions.DestinationError`

S3 destination errors

exception `twindb_backup.destination.exceptions.SshDestinationError`

Bases: `twindb_backup.destination.exceptions.DestinationError`

SSH destination errors

twindb_backup.destination.local module

Module defines Local destination.

class `twindb_backup.destination.local.Local` (*path=None*)

Bases: `twindb_backup.destination.base_destination.BaseDestination`

Local destination class.

delete (*path*)

Delete object from the destination

Parameters *path* – Relative path to the file to delete

get_stream (*copy*)

Get a PIPE handler with content of the backup copy streamed from the destination

Parameters *copy* (`BaseCopy`) – Backup copy

Returns

path

Root path on local file system where local backup copies are stored.

read (*filepath*)

Read content of a file path from destination.

Parameters *filepath* (*str*) – Relative path to file.

Returns Content of the file.

Return type `str`

save (*handler, filepath*)

Read from handler and save it on local storage

Parameters

- **filepath** – store backup copy as this name
- **handler** – Input stream

write (*content*, *filepath*)

Write string to a file.

Parameters

- **content** (*str*) – String.
- **filepath** (*str*) – Relative path to file.

twindb_backup.destination.s3 module

Module for S3 destination.

class `twindb_backup.destination.s3.S3` (***kwargs*)

Bases: `twindb_backup.destination.base_destination.BaseDestination`

S3 destination class.

Parameters **kwargs** – Keyword arguments.

- **bucket** - S3 bucket name.
- **aws_access_key_id** - AWS key id.
- **aws_secret_access_key** - AWS secret key.
- **aws_default_region** - AWS default region.
- **hostname** - Hostname of a host where a backup is taken from.

bucket

S3 bucket name.

create_bucket ()

Creates the bucket in s3 that will store the backups.

Raises `S3DestinationError` – if failed to create the bucket.

delete (*path*)

Deletes an S3 object.

Parameters **path** (*str*) – Key of S3 object.

Raises `S3DestinationError` – if failed to delete object.

delete_all_objects ()

Delete all objects from S3 bucket.

Raises `S3DestinationError` – if failed to delete objects from the bucket.

delete_bucket (*force=False*)

Delete the bucket in s3 that was storing the backups.

Parameters **force** (*bool*) – If the bucket is non-empty then delete the objects before deleting the bucket.

Raises `S3DestinationError` – if failed to delete the bucket.

get_stream (***kws*)

Get a PIPE handler with content of the backup copy streamed from the destination.

Parameters **copy** (*BaseCopy*) – Backup copy

Returns Stream with backup copy

Return type generator

Raises *S3DestinationError* – if failed to stream a backup copy.

static get_transfer_config ()

Build Transfer config

Returns Transfer config

Return type boto3.s3.transfer.TransferConfig

list_files (*prefix=None, recursive=False, pattern=None, files_only=False*)

List files in the destination that have common prefix.

Parameters

- **prefix** (*str*) – Common prefix. May include the bucket name. (e.g. `s3://my_bucket/foo/`) or simply a prefix in the bucket (e.g. `foo/`).
- **recursive** – Does nothing for this class.
- **pattern** (*str*) – files must match with this regexp if specified.
- **files_only** – Does nothing for this class.

Returns sorted list of file names.

Returns Full S3 url in form `s3://bucket/path/to/file`.

Return type list(str)

Raises *S3DestinationError* – if failed to list files.

read (*filepath*)

Read content of filepath and return it as a string.

Parameters **filepath** – Path in S3 bucket.

Returns Content of the file.

Return type str

Raises *FileNotFound* – If filepath doesn't exist.

save (*handler, filepath*)

Read from handler and save it to Amazon S3

Parameters

- **filepath** – save backup copy in a file with this name
- **handler** – stdout handler from backup source

static setup_s3_client ()

Creates an authenticated s3 client.

Returns S3 client instance.

Return type botocore.client.BaseClient

share (*s3_url*)

Share S3 file and return public link

Parameters `s3_url` (*str*) – S3 url

Returns Public url

Return type `str`

Raises `S3DestinationError` – if failed to share object.

static `validate_client_response` (*response*)

Validates the response returned by the client. Raises an exception if the response code is not 200 or 204

Parameters `response` (*dict*) – The response that needs to be validated.

Raises `S3DestinationError` – if response from S3 is invalid.

write (*content*, *filepath*)

Write string to a file.

Parameters

- **content** (*str*) – String.
- **filepath** (*str*) – Relative path to file.

class `twindb_backup.destination.s3.S3FileAccess`

Bases: `object`

Access modes for S3 files

private = 'private'

public_read = 'public-read'

twindb_backup.destination.ssh module

Module for SSH destination.

class `twindb_backup.destination.ssh.Ssh` (*remote_path*, ***kwargs*)

Bases: `twindb_backup.destination.base_destination.BaseDestination`

SSH destination class

Parameters

- **remote_path** – Path to store backup
- **kwargs** – Keyword arguments. See below
- **kwargs** – dict

****hostname**(str)hostname**(str)** Hostname of the host where backup is taken from.

****ssh_host**(str)ssh_host**(str)** Hostname for SSH connection. Default '127.0.0.1'.

****ssh_user**(str)ssh_user**(str)** Username for SSH connection. Default 'root'.

****ssh_port**(int)ssh_port**(int)** TCP port for SSH connection. Default 22.

****ssh_key**(str)ssh_key**(str)** File with an rsa/dsa key for SSH authentication. Default '/root/.ssh/id_rsa'.

client

Return client

delete (*path*)

Delete file by path

Parameters **path** – path to a remote file.

ensure_tcp_port_listening (*port*, *wait_timeout=10*)

Check that tcp port is open and ready to accept connections. Keep checking up to *wait_timeout* seconds.

Parameters

- **port** (*int*) – TCP port that is supposed to be listening.
- **wait_timeout** (*int*) – wait this many seconds until the port is ready.

Returns True if the TCP port is listening.

Return type bool

execute_command (*cmd*, *quiet=False*, *background=False*)

Execute ssh command

Parameters

- **cmd** (*str*) – Command for execution
- **quiet** – If True don't print errors
- **background** (*bool*) – Don't wait until the command exits.

Returns Handlers of stdin, stdout and stderr

Return type tuple

get_stream (***kws*)

Get a PIPE handler with content of the backup copy streamed from the destination

Parameters **copy** (*BaseCopy*) – Backup copy

Returns Standard output.

host

IP address of the destination.

netcat (*command*, *port=9990*)

Run netcat on the destination pipe it to a given command.

port

TCP port of the destination.

read (*filepath*)

Read content of a file path from destination.

Parameters **filepath** (*str*) – Relative path to file.

Returns Content of the file.

Return type str

save (*handler*, *filepath*)

Read from handler and save it on remote ssh server

Parameters

- **filepath** – relative path to a file to store the backup copy.
- **handler** – stream with content of the backup.

user
SSH user.

write (*content*, *filepath*)
Write string to a file.

Parameters

- **content** (*str*) – String.
- **filepath** (*str*) – Relative path to file.

Module contents

twindb_backup.exporter package

Submodules

twindb_backup.exporter.base_exporter module

Module defines base exporter class.

class `twindb_backup.exporter.base_exporter.BaseExporter`
Bases: object

Base exporter class

export (*category*, *measure_type*, *data*)
Send data to server

class `twindb_backup.exporter.base_exporter.ExportCategory`
Bases: object

Category of export data: files or mysql

files = 0

mysql = 1

class `twindb_backup.exporter.base_exporter.ExportMeasureType`
Bases: object

Type of measure time: backup or restore

backup = 0

restore = 1

twindb_backup.exporter.datadog_exporter module

Module defines DataDog exporter class.

class `twindb_backup.exporter.datadog_exporter.DataDogExporter` (*app_key*,
api_key)

Bases: `twindb_backup.exporter.base_exporter.BaseExporter`

DataDog exporter class

export (*category, measure_type, data*)

Export data to DataDog :param category: Data meant :param measure_type: Type of measure :param data: Data to posting :raise: DataDogExporterError if data is invalid

twindb_backup.exporter.exceptions module

Module for exporters exceptions

exception `twindb_backup.exporter.exceptions.BaseExporterError`

Bases: `exceptions.Exception`

General exporters error

exception `twindb_backup.exporter.exceptions.DataDogExporterError`

Bases: `twindb_backup.exporter.exceptions.BaseExporterError`

DataDog exporters error

Module contents

twindb_backup.modifiers package

Submodules

twindb_backup.modifiers.base module

Module defines Modifier() base class and its errors.

class `twindb_backup.modifiers.base.Modifier` (*input_stream*)

Bases: `object`

Base Modifier class

callback (***kwargs*)

Method that will be called after the stream ends

get_stream (***kws*)

Compress the input stream and return it as the output stream

Returns output stream handle

input

Returns Input stream to be modified

revert_stream (***kws*)

Un-Apply modifier and return output stream. The Base modifier does nothing, so it will return the input stream without modifications

Returns output stream handle

twindb_backup.modifiers.gpg module

Module defines modifier that implements asymmetric encryption with gpg

```
class twindb_backup.modifiers.gpg.Gpg (input_stream, recipient, keyring, secret_keyring=None)
    Bases: twindb_backup.modifiers.base.Modifier
    Asymmetric encryption
```

twindb_backup.modifiers.gzip module

Module defines modifier that compresses a stream with gzip

```
class twindb_backup.modifiers.gzip.Gzip (input_stream, level=9)
    Bases: twindb_backup.modifiers.base.Modifier
    Modifier that compresses the input_stream with gzip.
    suffix = '.gz'
```

twindb_backup.modifiers.keeplocal module

Module defines modifier that save a stream on the local file system

```
class twindb_backup.modifiers.keeplocal.KeepLocal (input_stream, local_path)
    Bases: twindb_backup.modifiers.base.Modifier
    KeepLocal() class saves a copy of the stream on the local file system. It doesn't alter the stream.
    callback (**kwargs)
        Method that will be called after the stream ends
```

Module contents

Modifiers module.

Modifier take a stream as input, do something with it (compress, encrypt, etc) and return the modified stream for a next modifier or backup destination.

Modifiers also do reverse operation - i.e. decompress, decrypt.

twindb_backup.source package

Submodules

twindb_backup.source.base_source module

Module defines base source class.

```
class twindb_backup.source.base_source.BaseSource (run_type)
    Bases: object
    Base source for backup
    basename
        Return file name (w/o directory part) of the backup.
    get_name ()
        Name that will be used to store backup copy from this source.
```

get_prefix()
Get prefix of the backup copy. It includes hostname and run type.
Returns Backup name prefix like 'db-10/daily'

get_stream()
Get backup stream in a handler

host
Return host where the backup is being taken from.

run_type = None

suffix
Backup file name suffix

twindb_backup.source.exceptions module

Module for backup source exceptions

exception `twindb_backup.source.exceptions.BinlogSourceError`
Bases: `twindb_backup.source.exceptions.SourceError`
Exceptions in Binlog source

exception `twindb_backup.source.exceptions.MySQLSourceError`
Bases: `twindb_backup.source.exceptions.SourceError`
Exceptions in MySQL source

exception `twindb_backup.source.exceptions.RemoteMySQLSourceError`
Bases: `twindb_backup.source.exceptions.MySQLSourceError`
Exceptions in remote MySQL source

exception `twindb_backup.source.exceptions.SourceError`
Bases: `twindb_backup.exceptions.TwinDBBackupError`
General source error

twindb_backup.source.file_source module

Module defines File source class for backing up local directories.

class `twindb_backup.source.file_source.FileSource` (*path, run_type*)
Bases: `twindb_backup.source.base_source.BaseSource`

FileSource class

apply_retention_policy (*dst, config, run_type*)
Apply retention policy

get_name ()
Generate relative destination file name

Returns file name

get_stream (***kws*)
Get a PIPE handler with content of the source

Returns

media_type

Get media type. Media type is a general term that describes what you backup. For directories `media_type` is 'file'.

Returns 'file'

twindb_backup.source.mysql_source module

Module defines MySQL source class for backing up local MySQL.

```
class twindb_backup.source.mysql_source.MySQLClient (defaults_file,           con-
                                                    nect_timeout=10,           host-
                                                    name='127.0.0.1')
```

Bases: object

Class to send queries to MySQL

cursor (***kws*)

MySQL cursor for connection to local MySQL instance.

get_connection (***kws*)

Connect to MySQL host and yield a connection.

Returns MySQL connection

Raises *MySQLSourceError* – if can't connect to server

variable (*varname*)

Read MySQL variable and return its value

```
class twindb_backup.source.mysql_source.MySQLConnectInfo (defaults_file,           con-
                                                    nect_timeout=10,           con-
                                                    cursor=<class
                                                    'pymysql.cursors.DictCursor'>,
                                                    hostname='127.0.0.1')
```

Bases: object

MySQL connection details

```
class twindb_backup.source.mysql_source.MySQLMasterInfo (host, port, user, password,
                                                    binlog, binlog_pos)
```

Bases: object

MySQL master details

```
class twindb_backup.source.mysql_source.MySQLSource (mysql_connect_info, run_type,
                                                    backup_type, **kwargs)
```

Bases: *twindb_backup.source.base_source.BaseSource*

MySQLSource class

apply_retention_policy (*dst, config, run_type, status*)

Delete old backup copies.

Parameters

- **dst** (*BaseDestination*) – Destination where the backups are stored.
- **config** (*TwinDBBackupConfig*) – Tool configuration
- **run_type** (*str*) – Run type.
- **status** (*Status*) – Backups status.

Returns Updated status.

Return type Status

binlog_coordinate

Binary log coordinate up to that backup is taken

Returns file name and position

Return type tuple

datadir

Return datadir path on MySQL server

disable_wsrep_desync ()

Wait till `wsrep_local_recv_queue` is zero and disable `wsrep_local_recv_queue` then

enable_wsrep_desync ()

Try to enable `wsrep_desync`

Returns True if `wsrep_desync` was enabled. False if not supported

full

Check if the backup copy is a full copy.

Returns True if it's a full copy.

Return type bool

galera

Check if local MySQL instance is a Galera cluster

Returns True if it's a Galera.

Return type bool

static get_binlog_coordinates (*err_log_path*)

Parse innobackupex log and return binary log coordinate

Parameters `err_log_path` (*str*) – path to the innobackupex log

Returns Binlog coordinate.

Return type tuple

get_connection (***kws*)

Connect to MySQL host and yield a connection.

Returns MySQL connection

Raises `MySQLSourceError` – if can't connect to server

get_name ()

Generate relative destination file name

Returns file name

get_stream (***kws*)

Get a PIPE handler with content of the source :return:

incremental

Check if the backup copy is an incremental copy.

Returns True if it's an incremental copy.

Return type bool

is_galera ()
Check if local MySQL instance is a Galera cluster

Returns True if it's a Galera.

Return type bool

lsn
The latest LSN of the taken backup :return: LSN :rtype: int

status
Backup status on a destination

Returns Backups status

Return type dict

type
Get backup copy type - full or incremental

Returns 'full' or 'incremental'

Return type str

wsrep_provider_version
Parse Galera version from wsrep_provider_version.

Returns Galera version

Return type str

twindb_backup.source.remote_mysql_source module

Module defines MySQL source class for backing up remote MySQL.

class `twindb_backup.source.remote_mysql_source.RemoteMySQLSource` (*kwargs*)
Bases: `twindb_backup.source.mysql_source.MySQLSource`

Remote MySQLSource class

apply_backup (*datadir*)
Apply backup of destination server

Parameters *datadir* – Path to datadir

Returns Binlog file name and position

Return type tuple

Raises `RemoteMySQLSourceError` – if any error.

clone (*dest_host, port, compress=False*)
Send backup to destination host

Parameters

- **dest_host** (*str*) – Destination host
- **port** (*int*) – Port to sending backup
- **compress** (*bool*) – If True compress stream

Raises `RemoteMySQLSourceError` – if any error

clone_config (*dst*)
Clone config to destination server

Parameters `dst` (`Ssh`) – Destination server

`get_stream` (***kws*)

Get a PIPE handler with content of the source :return:

`setup_slave` (*master_info*)

Change master

Parameters `master_info` (`MySQLMasterInfo`) – Master details.

Module contents

twindb_backup.ssh package

Submodules

twindb_backup.ssh.client module

Module that implements SSH client.

class `twindb_backup.ssh.client.SshClient` (*host='127.0.0.1', port=22, key='/root/.id_rsa', user='root'*)

Bases: `object`

SSH client class. Allows to connect to a remote SSH server and execute commands on it.

Parameters

- **host** (*str*) – Destination host to connect to. Defaults to '127.0.0.1'.
- **port** (*int*) – Destination port to connect to. Default is 22.
- **key** (*str*) – SSH client key for passwordless authentication. Default is '/root/.id_rsa'.
- **user** (*str*) – SSH client username. Default is 'root'.

execute (*cmd, quiet=False, background=False*)

Execute a command on a remote SSH server.

Parameters

- **cmd** (*str*) – Command for execution.
- **quiet** – if quiet is True don't print error messages
- **background** (*bool*) – Don't wait until the command exits.

Returns Strings with stdout and stderr. If command is executed in background the method will return None.

Return type tuple

Raises `SshClientException` – if any error or non-zero exit code

get_remote_handlers (***kws*)

Get remote stdin, stdout and stderr handler

Parameters `cmd` (*str*) – Command for execution

Returns Remote stdin, stdout and stderr handler

Return type tuple(generator, generator, generator)

Raises `SshDestinationError` – if any error

get_text_content (*path*)

Get text content of file by path

Parameters **path** (*str*) – File path

Returns File content

Return type str

host

Remote SSH host

list_files (*path, recursive=False, files_only=False*)

Get list of file by prefix

Parameters

- **path** (*str*) – Path
- **recursive** (*bool*) – Recursive return list of files
- **files_only** (*bool*) – Don't list directories if True. Default is False.

Returns List of files

Return type list

port

TCP port for SSH connection

session (***kws*)

Get SSH session

Return type generator

Returns SSH session

user

User for SSH connection

write_config (*path, cfg*)

Write config to file

Parameters

- **path** – Path to file
- **cfg** – Instance of ConfigParser

write_content (*path, content*)

Write content to path

Parameters

- **path** – Path to file
- **content** – Content

twindb_backup.ssh.exceptions module

SSH Client Exceptions.

exception twindb_backup.ssh.exceptions.SshClientException

Bases: exceptions.Exception

Exception in SshClient

Module contents

twindb_backup.status package

Submodules

twindb_backup.status.base_status module

Base status is a class for a general purpose status.

```
class twindb_backup.status.base_status.BaseStatus (content=None, dst=None, status_directory=None)
```

Bases: object

Base class for status. It can be instantiated either from a string with status content or from a destination instance. If the destination is given then the status will be read from a status file on the destination.

Parameters

- **content** (*str*) – if passed it will initialize a status from this string.
- **dst** (*BaseDestination*) – Destination instance.
- **status_directory** (*str*) – Relative path to a directory where the status file is stored. Usually, it's a hostname where backup was taken from.

Raises *CorruptedStatus* – If the content string is not a valid status or empty string.

add (*backup_copy*)

Add entry to status.

Parameters **backup_copy** (*BaseCopy*) – Instance of backup copy

basename

Returns file name without a directory path where the status is stored in the destination.

latest_backup

Find the latest backup copy.

Returns backup copy or None if status is empty.

Return type *BaseCopy*

md5

Returns MD5 checksum of the status. It is calculated as a md5 of output of

```
self._status_serialize().:type: str
```

remove (*key*)

Remove key from the status.

Parameters **key** (*str*) – A copy key in the status.

save (*dst*)

Write status file to the destination.

Parameters **dst** (*BasicDestination*) – Destination instance.

serialize ()

Return a string that represents current state

status_path

Return relative path where status is stored.

Returns relative to the destination path where the status is stored.

Return type str

version

Version of status file. Originally status file didn't have any versions, but in future the version will be used to work with new features.

twindb_backup.status.exceptions module

Status exceptions

exception `twindb_backup.status.exceptions.CorruptedStatus`

Bases: `twindb_backup.status.exceptions.StatusError`

Status file is corrupt

exception `twindb_backup.status.exceptions.StatusError`

Bases: `twindb_backup.exceptions.TwinDBBackupError`

General status error

exception `twindb_backup.status.exceptions.StatusKeyNotFound`

Bases: `twindb_backup.status.exceptions.StatusError`

Accessing a key that doesn't exist

twindb_backup.status.mysql_status module

Class to store and work with status file

class `twindb_backup.status.mysql_status.MySQLStatus` (*content=None, dst=None, status_directory=None*)

Bases: `twindb_backup.status.periodic_status.PeriodicStatus`

Class that stores status file and implements operations on it.

basename

candidate_parent (*run_type*)

Find a backup copy that can be a parent

Parameters *run_type* – See `get_backup_type()`.

Returns Backup copy or None

Return type *MySQLCopy*

full_copy_exists (*run_type*)

Check whether there is a full copy.

Parameters *run_type* – See `get_backup_type()`.

Returns True if there is a full copy. False if there is no an eligible full copy.

Return type bool

next_backup_type (*full_backup, run_type*)

Return backup type to take. If `full_backup=daily` then for hourly backups it will be incremental, for all other - full

Parameters

- **full_backup** – when to take full backup according to config.
- **run_type** – what kind of backup run it is.

Returns “full” or “incremental”

Return type str

Module contents

4.1.2 Submodules

4.1.3 `twindb_backup.backup` module

Module that parses config file, builds a modifiers chain and fires backup jobs.

`twindb_backup.backup.backup_binlogs` (*run_type*, *config*)
Copy MySQL binlog files to the backup destination.

Parameters

- **run_type** (*str*) – Run type
- **config** (`TwinDBBackupConfig`) – Tool configuration

`twindb_backup.backup.backup_everything` (*run_type*, *twindb_config*, *binlogs_only=False*)
Run backup job

Parameters

- **run_type** (*str*) – hourly, daily, etc
- **twindb_config** (`TwinDBBackupConfig`) – ConfigParser instance
- **binlogs_only** (*bool*) – If True copy only MySQL binary logs.

`twindb_backup.backup.backup_files` (*run_type*, *config*)
Backup local directories

Parameters

- **run_type** (*str*) – Run type
- **config** (`TwinDBBackupConfig`) – Configuration

`twindb_backup.backup.backup_mysql` (*run_type*, *config*)
Take backup of local MySQL instance

Parameters

- **run_type** (*str*) – Run type
- **config** (`TwinDBBackupConfig`) – Tool configuration

`twindb_backup.backup.binlogs_to_backup` (*cursor*, *last_binlog=None*)

Finds list of binlogs to copy. It will return the binlogs from the last to the current one (excluding it). If binlog are not enabled in the server the function will return empty list.

Parameters

- **cursor** – MySQL cursor
- **last_binlog** – Name of the last copied binlog.

Returns list of binlogs to backup.

Return type list

`twindb_backup.backup.run_backup_job` (*twindb_config*, *run_type*, *lock_file*='var/run/twindb-backup.lock', *binlogs_only*=False)

Grab a lock waiting up to allowed timeout and start backup jobs

Parameters

- **twindb_config** (*TwinDBBackupConfig*) – Tool configuration
- **run_type** (*str*) – Run type
- **lock_file** (*str*) – File used as a lock
- **binlogs_only** (*bool*) – If True copy only binlogs.

`twindb_backup.backup.set_open_files_limit` ()

Detect maximum supported number of open file and set it

`twindb_backup.backup.timeout` (**args*, ***kws*)

Implement timeout

Parameters **seconds** (*int*) – timeout in seconds

4.1.4 twindb_backup.cli module

Entry points for twindb-backup tool

4.1.5 twindb_backup.clone module

Module defines clone feature

`twindb_backup.clone.clone_mysql` (*cfg*, *source*, *destination*, *replication_user*, *replication_password*, *netcat_port*=9990, *compress*=False)

Clone mysql backup of remote machine and stream it to slave

Parameters **cfg** (*TwinDBBackupConfig*) – TwinDB Backup tool config

4.1.6 twindb_backup.configuration module

Module to process configuration file.

class `twindb_backup.configuration.TwinDBBackupConfig` (*config_file*='etc/twindb/twindb-backup.cfg')

Bases: object

Class represents TwinDB Backup configuration

backup_dirs

Directories to backup

backup_mysql

FLag to backup MySQL or not

compression

Returns Compression configuration

Return type CompressionConfig

destination (*backup_source*='build-8932059-project-262906-twindb-backup')

Parameters `backup_source` (*str*) – Hostname of the host where backup is taken from.

Returns Backup destination instance

Return type *BaseDestination*

exporter

Read config and return export transport instance

Returns Instance of export transport, if it is set

Return type *BaseExporter*

Raise `ConfigurationError`, if transport isn't implemented

gcs

Google Cloud Storage configuration

gpg

GPG configuration.

keep_local_path

If specified a local path where the tool will keep an additional local backup copy.

mysql

Returns Local MySQL source configuration.

Return type *MySQLConfig*

retention

Returns Remote retention policy.

Return type *RetentionPolicy*

retention_local

Returns Local retention policy.

Return type *RetentionPolicy*

run_intervals

Run intervals config. When to run or not the backup.

Returns Configuration with data on whether to run the backup tool now.

Return type *RunIntervals*

s3

Amazon S3 configuration

ssh

Returns Remote SSH configuration.

Return type *SSHConfig*

4.1.7 `twindb_backup.exceptions` module

Module that describes exceptions of `twindb_backup` module

exception `twindb_backup.exceptions.LockWaitTimeoutError`

Bases: `twindb_backup.exceptions.TwinDBBackupError`

Class that describes exception of lock wait timeout

exception `twindb_backup.exceptions.OperationError`
 Bases: `twindb_backup.exceptions.TwinDBBackupError`

High level exceptions of `twindb_backup` package

exception `twindb_backup.exceptions.TwinDBBackupError`
 Bases: `exceptions.Exception`

Catch-all exceptions

exception `twindb_backup.exceptions.TwinDBBackupInternalError`
 Bases: `twindb_backup.exceptions.TwinDBBackupError`

Internal errors in the tool itself

4.1.8 `twindb_backup.export` module

Module to process export

`twindb_backup.export.export_info` (*cfg, data, category, measure_type*)
 Export data to service

Parameters

- **cfg** (`TwinDBBackupConfig`) – Config file
- **data** – Data
- **category** – Category of data
- **measure_type** – Type of measure
- **category** – Category

4.1.9 `twindb_backup.ls` module

Module that works with list of backup copies

`twindb_backup.ls.list_available_backups` (*twindb_config, copy_type=None*)
 Print known backup copies on a destination specified in the configuration.

Parameters

- **twindb_config** (`TwinDBBackupConfig`) – tool configuration
- **copy_type** (*files|mysql*) – Limit list to specific type of backups.

4.1.10 `twindb_backup.restore` module

Module that restores backup copies.

`twindb_backup.restore.gen_grastate` (*path, version, uuid, seqno*)
 Generate and save grastate file.

Parameters

- **path** – Path to grastate file.
- **version** – Galera version from grastate.dat.
- **uuid** – UUID from grastate.dat.

- **seqno** – seqno from grastate.dat.

`twindb_backup.restore.get_my_cnf(status, key)`

Get MySQL config from the status.

Parameters

- **status** (`MySQLStatus`) – Backup status.
- **key** (`str`) – Backup name.

Returns Content of my.cnf or None if not found

Return type `str`

`twindb_backup.restore.restore_from_file(twindb_config, copy, dst_dir)`

Restore a directory from a backup copy in the directory

Parameters

- **twindb_config** (`TwinDBBackupConfig`) – tool configuration
- **copy** (`BaseCopy`) – Instance of BaseCopy or and inheriting classes.
- **dst_dir** (`str`) – Path to destination directory. Must exist and be empty.

`twindb_backup.restore.restore_from_mysql(twindb_config, copy, dst_dir, tmp_dir=None, cache=None, hostname=None)`

Restore MySQL datadir in a given directory

Parameters

- **twindb_config** (`TwinDBBackupConfig`) – tool configuration
- **copy** (`MySQLCopy`) – Backup copy instance.
- **dst_dir** (`str`) – Destination directory. Must exist and be empty.
- **tmp_dir** (`str`) – Path to temp directory
- **cache** (`Cache`) – Local cache object.
- **hostname** (`str`) – Hostname

`twindb_backup.restore.restore_from_mysql_full(stream, dst_dir, config, redo_only=False, xtrabackup_binary='/opt/twindb-backup/embedded/bin/xtrabackup', xbstream_binary='/opt/twindb-backup/embedded/bin/xbstream')`

Restore MySQL datadir from a backup copy

Parameters

- **stream** – Generator that provides backup copy
- **dst_dir** (`str`) – Path to destination directory. Must exist and be empty.
- **config** (`TwinDBBackupConfig`) – Tool configuration.
- **redo_only** (`bool`) – True if the function has to do final apply of the redo log. For example, if you restore backup from a full copy it should be False. If you restore from incremental copy and you restore base full copy redo_only should be True.
- **xtrabackup_binary** – path to xtrabackup binary.
- **xbstream_binary** – Path to xbstream binary

Returns If success, return True

Return type bool

```
twindb_backup.restore.restore_from_mysql_incremental (stream, dst_dir, config, tmp_dir=None,
xtrabackup_binary='/opt/twindb-backup/embedded/bin/xtrabackup',
xstream_binary='/opt/twindb-backup/embedded/bin/xstream')
```

Restore MySQL datadir from an incremental copy.

Parameters

- **stream** – Generator that provides backup copy
- **dst_dir** (*str*) – Path to destination directory. Must exist and be empty.
- **config** (*TwinDBBackupConfig*) – Tool configuration.
- **tmp_dir** (*str*) – Path to temp dir
- **xtrabackup_binary** – Path to xtrabackup binary.
- **xstream_binary** – Path to xstream binary

Returns If success, return True

Return type bool

```
twindb_backup.restore.update_grastate (dst_dir, status, key)
```

If xtrabackup_galera_info exists in the destination directory then parse it and generate grastate.dat file.

Parameters

- **dst_dir** (*str*) – Path to destination directory.
- **status** (*dict*) – Backup status
- **key** (*str*) – Backup name

4.1.11 twindb_backup.share module

Module that works with sharing backups

```
twindb_backup.share.share (twindb_config, s3_url)
```

Function for generate make public file and get public url

Parameters

- **twindb_config** (*TwinDBBackupConfig*) – tool configuration
- **s3_url** (*str*) – S3 url to file

Raise TwinDBBackupError

4.1.12 twindb_backup.util module

Module with helper functions

```
twindb_backup.util.empty_dir (path)
```

Remove all files and directories in path

Parameters **path** (*str*) – Path to directory to be emptied.

`twindb_backup.util.ensure_empty(path)`

Check if a given directory is empty and exit if not.

Parameters `path` (*str*) – path to directory

`twindb_backup.util.kill_children()`

Kill child process

`twindb_backup.util.mkdir_p(path, mode=511)`

Emulate `mkdir -p`. Create a directory named `path` with numeric mode `mode`. The default mode is `0777` (octal)

Parameters

- **path** (*str*) – Directory path.
- **mode** (*int*) – Directory permissions. The default mode is `0777` (octal)

`twindb_backup.util.my_cnfs(common_paths=None)`

Start reading a root `my.cnf` file given in `common_paths` and parse included files.

Parameters `common_paths` (*list*) – list of `my.cnf` files to start parsing from.

Returns list of all included `my.cnf` files

Return type `list`

`twindb_backup.util.normalize_b64_data(coding)`

Normalize base64 key. See <http://bit.ly/2vxIANC> for details.

Parameters `coding` – Encoded data

Returns Normalized encoded data

`twindb_backup.util.run_command(*args, **kwds)`

Run shell command locally

Parameters

- **command** (*list*) – Command to run
- **ok_non_zero** (*bool*) – Don't consider non-zero exit code as an error.

Returns file object with `stdout` as generator to use with `with`

`twindb_backup.util.split_host_port(host_port)`

Splits a string of host and port separated by a semicolon.

Parameters `host_port` – host or `host:port`. Allowed values are like `10.20.31.1:3306` or just `10.20.31.1`

Returns a tuple with host and port. If only address is specified it'll return (address, None). If `host_port` is None it will return (None, None)

Return type `tuple`

4.1.13 `twindb_backup.verify` module

Module that verify backup copies.

`twindb_backup.verify.edit_backup_my_cnf(dst_path)`

Removed options from config(besides MySQL 5.7.8)

`twindb_backup.verify.verify_mysql_backup(twindb_config, dst_path, backup_file, host-name=None)`

Restore mysql backup and measure time

Parameters

- **hostname** –
- **backup_file** –
- **dst_path** –
- **twindb_config** (`TwinDBBackupConfig`) – tool configuration

4.1.14 Module contents

TwinDB Backup module.

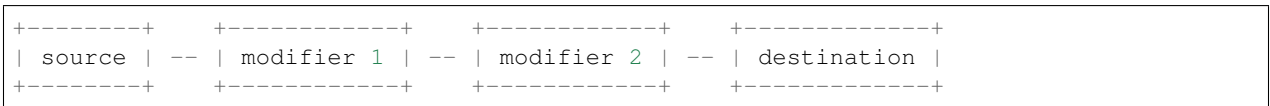
The module is a core of twindb-backup tool. It includes backup and restore functionality. The module takes a backup from something defined in a source class and saves the backup copy in something defined in a destination class.

The source class inherits from `BaseSource()` from `twindb_backup.source.base_source.py`. The source class must define `get_stream()` method that yields a file object that is used for next classes. Typical classes are `FileSource()` to backup files and directories, `MySQLSource()` to backup MySQL.

The destination class inherits from `BaseDestination()`. This is where you store backups. The destination class must define `save()` method that takes an input stream and saves it somewhere. Examples of the destination class are `S3()`, `Ssh()`.

There are modifier classes. The modifier class sits in the middle between the source and the destination and does something with a stream before the stream is saved. The modifier class may save a local copy (`KeepLocal()`) or encrypt the stream or else. The modifier class inherits `Modifier()`

The backup process may be depicted as a chain of modifiers with the source in the head and the destination in the tail.



```
class twindb_backup.LessThanFilter (exclusive_maximum, name="")
```

```
    Bases: logging.Filter
```

```
    Filters out log messages of a lower level.
```

```
    filter (record)
```

```
        Determine if the specified record is to be logged.
```

```
        Is the specified record to be logged? Returns 0 for no, nonzero for yes. If deemed appropriate, the record may be modified in-place.
```

```
twindb_backup.delete_local_files (dir_backups, keep_copies)
```

```
    Deletes local backup copies based on given retention number.
```

Parameters

- **dir_backups** (*str*) – directory with backup copies
- **keep_copies** (*int*) – how many to keep

```
    Returns None
```

```
twindb_backup.get_files_to_delete (all_files, keep_copies)
```

```
    If you give it a list of files and number of how many you'd like to keep the function will return files that need to be deleted
```

Parameters

- **all_files** (*list*) – list of strings
- **keep_copies** (*int*) – number of copied to keep

Returns list of strings (files) to delete

Return type list

`twindb_backup.get_timeout` (*run_type*)

Get timeout for a each run type - daily, hourly etc

Parameters **run_type** (*str*) – Run type

Returns Number of seconds the tool allowed to wait until other instances finish

Return type int

`twindb_backup.save_measures` (*start_time*, *end_time*, *log_path*='/var/log/twindb-backup-measures.log')

Save backup measures to log file

`twindb_backup.setup_logging` (*logger*, *debug=False*)

Configures logging for the module

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given. You can contribute in many ways:

5.1 Types of Contributions

5.1.1 Report Bugs

Report bugs at <https://github.com/twindb/backup/issues>.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

5.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” and “help wanted” is open to whoever wants to implement it.

5.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “enhancement” and “help wanted” is open to whoever wants to implement it.

5.1.4 Write Documentation

TwinDB Backup could always use more documentation, whether as part of the official TwinDB Backup docs, in docstrings, or even on the web in blog posts, articles, and such.

5.1.5 Submit Feedback

The best way to send feedback is to file an issue at <https://github.com/twindb/backup/issues>.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

5.2 Get Started!

Ready to contribute? Here's how to set up TwinDB Backup for local development.

1. Fork the TwinDB Backup repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/backup.git twindb_backup
```

3. Install your local copy into a virtualenv:

```
$ make virtualenv
$ source env/bin/activate
$ make bootstrap
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ make test-all
```

To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

5.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

- The pull request should include tests.
- If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
- The pull request should work for Python 2.6 and 2.7.
- Check https://travis-ci.org/twindb/backup/pull_requests and make sure that the tests pass for all supported Python versions.

6.1 Development Lead

- TwinDB Development Team <dev@twindb.com>

6.2 Contributors

None yet. Why not be the first?

CHAPTER 7

History

See [GitHub commits history](#).

CHAPTER 8

Indices and tables

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

t

- twindb_backup, 39
- twindb_backup.backup, 32
- twindb_backup.cache, 12
- twindb_backup.cache.cache, 11
- twindb_backup.cli, 33
- twindb_backup.clone, 33
- twindb_backup.configuration, 33
- twindb_backup.copy, 14
 - twindb_backup.copy.base_copy, 12
 - twindb_backup.copy.binlog_copy, 12
 - twindb_backup.copy.exceptions, 13
 - twindb_backup.copy.mysql_copy, 13
 - twindb_backup.copy.periodic_copy, 14
- twindb_backup.destination, 21
- twindb_backup.destination.base_destination, 15
- twindb_backup.destination.exceptions, 16
- twindb_backup.destination.local, 16
- twindb_backup.destination.s3, 17
- twindb_backup.destination.ssh, 19
- twindb_backup.exceptions, 34
- twindb_backup.export, 35
- twindb_backup.exporter, 22
 - twindb_backup.exporter.base_exporter, 21
 - twindb_backup.exporter.datadog_exporter, 21
 - twindb_backup.exporter.exceptions, 22
- twindb_backup.ls, 35
- twindb_backup.modifiers, 23
 - twindb_backup.modifiers.base, 22
 - twindb_backup.modifiers.gpg, 22
 - twindb_backup.modifiers.gzip, 23
 - twindb_backup.modifiers.keeplocal, 23
- twindb_backup.restore, 35
- twindb_backup.share, 37
- twindb_backup.source, 28
 - twindb_backup.source.base_source, 23
 - twindb_backup.source.exceptions, 24
 - twindb_backup.source.file_source, 24
 - twindb_backup.source.mysql_source, 25
 - twindb_backup.source.remote_mysql_source, 27
 - twindb_backup.ssh, 30
 - twindb_backup.ssh.client, 28
 - twindb_backup.ssh.exceptions, 29
 - twindb_backup.status, 32
 - twindb_backup.status.base_status, 30
 - twindb_backup.status.exceptions, 31
 - twindb_backup.status.mysql_status, 31
 - twindb_backup.util, 37
 - twindb_backup.verify, 38

A

add() (*twindb_backup.cache.cache.Cache* method), 11
 add() (*twindb_backup.status.base_status.BaseStatus* method), 30
 apply_backup() (*twindb_backup.source.remote_mysql_source.RemoteMySQLSource* method), 27
 apply_retention_policy() (*twindb_backup.source.file_source.FileSource* method), 24
 apply_retention_policy() (*twindb_backup.source.mysql_source.MySQLSource* method), 25
 as_dict() (*twindb_backup.copy.mysql_copy.MySQLCopy* method), 13

B

backup (*twindb_backup.exporter.base_exporter.ExportMeasureType* attribute), 21
 backup_binlogs() (in module *twindb_backup.backup*), 32
 backup_dirs (*twindb_backup.configuration.TwinDBBackupConfig* attribute), 33
 backup_everything() (in module *twindb_backup.backup*), 32
 backup_files() (in module *twindb_backup.backup*), 32
 backup_finished (*twindb_backup.copy.mysql_copy.MySQLCopy* attribute), 13
 backup_mysql (*twindb_backup.configuration.TwinDBBackupConfig* attribute), 33
 backup_mysql() (in module *twindb_backup.backup*), 32
 backup_started (*twindb_backup.copy.mysql_copy.MySQLCopy* attribute), 13
 BackupCopyError, 13
 BaseCopy (class in *twindb_backup.copy.base_copy*), 12
 BaseDestination (class in *twindb_backup.destination.base_destination*), 15

BaseExporter (class in *twindb_backup.exporter.base_exporter*), 21
 BaseExporterError, 22
 basename (*twindb_backup.source.base_source.BaseSource* attribute), 23
 basename (*twindb_backup.status.base_status.BaseStatus* attribute), 30
 basename (*twindb_backup.status.mysql_status.MySQLStatus* attribute), 31
 BaseSource (class in *twindb_backup.source.base_source*), 23
 BaseStatus (class in *twindb_backup.status.base_status*), 30
 binlog (*twindb_backup.copy.mysql_copy.MySQLCopy* attribute), 13
 binlog_coordinate (*twindb_backup.source.mysql_source.MySQLSource* attribute), 26
 BinlogCopy (class in *twindb_backup.copy.binlog_copy*), 12
 binlogs_to_backup() (in module *twindb_backup.backup*), 32
 BinlogSourceError, 24
 bucket (*twindb_backup.destination.s3.S3* attribute), 17

C

Cache (class in *twindb_backup.cache.cache*), 11
 CacheException, 12
 callback() (*twindb_backup.modifiers.base.Modifier* method), 22
 callback() (*twindb_backup.modifiers.keeplocal.KeepLocal* method), 23
 candidate_parent() (*twindb_backup.status.mysql_status.MySQLStatus* method), 31
 client (*twindb_backup.destination.ssh.Ssh* attribute), 19
 clone() (*twindb_backup.source.remote_mysql_source.RemoteMySQLSource* method), 27

clone_config() (*twindb_backup.source.remote_mysql_source.RemoteMySQLSource* module *twindb_backup.util*), 37
 method), 27
 clone_mysql() (*in module twindb_backup.clone*), 33
 ensure_tcp_port_listening() (*twindb_backup.destination.ssh.Ssh* method),
 compression (*twindb_backup.configuration.TwinDBBackupConfig* 20
 attribute), 33
 execute() (*twindb_backup.ssh.client.SshClient*
 method), 28
 config (*twindb_backup.copy.mysql_copy.MySQLCopy*
 attribute), 13
 execute_command() (*twindb_backup.destination.ssh.Ssh* method),
 CorruptedStatus, 31
 create_bucket() (*twindb_backup.destination.s3.S3*
 method), 17
 export() (*twindb_backup.exporter.base_exporter.BaseExporter*
 method), 21
 created_at (*twindb_backup.copy.binlog_copy.BinlogCopy*
 attribute), 13
 export() (*twindb_backup.exporter.datadog_exporter.DataDogExporter*
 method), 21
 created_at (*twindb_backup.copy.mysql_copy.MySQLCopy*
 attribute), 14
 export_info() (*in module twindb_backup.export*),
 cursor() (*twindb_backup.source.mysql_source.MySQLClient*
 method), 25
 35
 ExportCategory (class *in*
twindb_backup.exporter.base_exporter),
 21
D
 datadir (*twindb_backup.source.mysql_source.MySQLSource*
 attribute), 26
 exporter (*twindb_backup.configuration.TwinDBBackupConfig*
 attribute), 34
 DataDogExporter (class *in* *twindb_backup.exporter.datadog_exporter*), 21
 ExportMeasureType (class *in*
twindb_backup.exporter.base_exporter),
 21
 DataDogExporterError, 22
 delete() (*twindb_backup.destination.base_destination.BaseDestination*
 method), 15
F
 delete() (*twindb_backup.destination.local.Local*
 method), 16
 FileNotFound, 16
 files (*twindb_backup.exporter.base_exporter.ExportCategory*
 attribute), 21
 delete() (*twindb_backup.destination.s3.S3* method),
 17
 FileSource (class *in*
twindb_backup.source.file_source), 24
 delete() (*twindb_backup.destination.ssh.Ssh*
 method), 19
 filter() (*twindb_backup.LessThanFilter* method), 39
 delete_all_objects() (*twindb_backup.destination.s3.S3* method),
 17
 full (*twindb_backup.source.mysql_source.MySQLSource*
 attribute), 26
 delete_bucket() (*twindb_backup.destination.s3.S3*
 method), 17
 full_copy_exists() (*twindb_backup.status.mysql_status.MySQLStatus*
 method), 31
 delete_local_files() (*in module*
twindb_backup), 39
G
 destination() (*twindb_backup.configuration.TwinDBBackupConfig* *twindb_backup.copy.mysql_copy.MySQLCopy*
 attribute), 14
 DestinationError, 16
 galera (*twindb_backup.source.mysql_source.MySQLSource*
 attribute), 26
 disable_wsrep_desync() (*twindb_backup.source.mysql_source.MySQLSource*
 method), 26
 ges (*twindb_backup.configuration.TwinDBBackupConfig*
 attribute), 34
 duration (*twindb_backup.copy.mysql_copy.MySQLCopy*
 attribute), 14
 GCSDestinationError, 16
E
 gen_grastate() (*in module twindb_backup.restore*),
 35
 edit_backup_my_cnf() (*in module*
twindb_backup.verify), 38
 get_binlog_coordinates() (*twindb_backup.source.mysql_source.MySQLSource*
 static method), 26
 empty_dir() (*in module twindb_backup.util*), 37
 get_connection() (*twindb_backup.source.mysql_source.MySQLClient*
 method), 25
 enable_wsrep_desync() (*twindb_backup.source.mysql_source.MySQLSource*
 method), 26
 get_connection() (*twindb_backup.source.mysql_source.MySQLSource*
 method), 26

[get_files_to_delete\(\)](#) (in module `twindb_backup`), 39
[get_my_cnf\(\)](#) (in module `twindb_backup.restore`), 36
[get_name\(\)](#) (`twindb_backup.source.base_source.BaseSource` method), 23
[get_name\(\)](#) (`twindb_backup.source.file_source.FileSource` method), 24
[get_name\(\)](#) (`twindb_backup.source.mysql_source.MySQLSource` method), 26
[get_name\(\)](#) (`twindb_backup.source.mysql_source.MySQLSource` method), 26
[get_prefix\(\)](#) (`twindb_backup.source.base_source.BaseSource` method), 23
[get_remote_handlers\(\)](#) (`twindb_backup.ssh.client.SshClient` method), 28
[get_stream\(\)](#) (`twindb_backup.destination.base_destination.BaseDestination` method), 15
[get_stream\(\)](#) (`twindb_backup.destination.local.Local` method), 16
[get_stream\(\)](#) (`twindb_backup.destination.s3.S3` method), 17
[get_stream\(\)](#) (`twindb_backup.destination.ssh.Ssh` method), 20
[get_stream\(\)](#) (`twindb_backup.modifiers.base.Modifier` method), 22
[get_stream\(\)](#) (`twindb_backup.source.base_source.BaseSource` method), 24
[get_stream\(\)](#) (`twindb_backup.source.file_source.FileSource` method), 24
[get_stream\(\)](#) (`twindb_backup.source.mysql_source.MySQLSource` method), 26
[get_stream\(\)](#) (`twindb_backup.source.mysql_source.MySQLSource` method), 26
[get_stream\(\)](#) (`twindb_backup.source.remote_mysql_source.RemoteMySQLSource` method), 28
[get_text_content\(\)](#) (`twindb_backup.ssh.client.SshClient` method), 28
[get_timeout\(\)](#) (in module `twindb_backup`), 40
[get_transfer_config\(\)](#) (`twindb_backup.destination.s3.S3` static method), 18
[Gpg](#) (class in `twindb_backup.modifiers.gpg`), 22
[gpg](#) (`twindb_backup.configuration.TwinDBBackupConfig` attribute), 34
[Gzip](#) (class in `twindb_backup.modifiers.gzip`), 23

H

[host](#) (`twindb_backup.copy.periodic_copy.PeriodicCopy` attribute), 14
[host](#) (`twindb_backup.destination.ssh.Ssh` attribute), 20
[host](#) (`twindb_backup.source.base_source.BaseSource` attribute), 24
[host](#) (`twindb_backup.ssh.client.SshClient` attribute), 29

I

[incremental](#) (`twindb_backup.source.mysql_source.MySQLSource`

[input](#) (`twindb_backup.modifiers.base.Modifier` attribute), 22
[is_galera\(\)](#) (`twindb_backup.source.mysql_source.MySQLSource` method), 26
[keep_local_path](#) (`twindb_backup.configuration.TwinDBBackupConfig` attribute), 34
[keep_local](#) (class in `twindb_backup.modifiers.keeplocal`), 23
[key](#) (`twindb_backup.copy.base_copy.BaseCopy` attribute), 12
[kill_children\(\)](#) (in module `twindb_backup.util`), 28

K

L

[latest_backup](#) (`twindb_backup.status.base_status.BaseStatus` attribute), 30
[LessThanFilter](#) (class in `twindb_backup`), 39
[list_available_backups\(\)](#) (in module `twindb_backup.ls`), 35
[list_files\(\)](#) (`twindb_backup.destination.base_destination.BaseDestination` method), 15
[list_files\(\)](#) (`twindb_backup.destination.s3.S3` method), 18
[list_files\(\)](#) (`twindb_backup.ssh.client.SshClient` method), 29
[Local](#) (class in `twindb_backup.destination.local`), 16
[LockWaitTimeoutError](#), 34
[lsn](#) (`twindb_backup.copy.mysql_copy.MySQLCopy` attribute), 14
[lsn](#) (`twindb_backup.source.mysql_source.MySQLSource` attribute), 27

M

[md5](#) (`twindb_backup.status.base_status.BaseStatus` attribute), 30
[media_type](#) (`twindb_backup.source.file_source.FileSource` attribute), 24
[mkdir_p\(\)](#) (in module `twindb_backup.util`), 38
[Modifier](#) (class in `twindb_backup.modifiers.base`), 22
[my_cnfs\(\)](#) (in module `twindb_backup.util`), 38
[mysql](#) (`twindb_backup.configuration.TwinDBBackupConfig` attribute), 34
[mysql](#) (`twindb_backup.exporter.base_exporter.ExportCategory` attribute), 21
[MySQLClient](#) (class in `twindb_backup.source.mysql_source`), 25
[MySQLConnectInfo](#) (class in `twindb_backup.source.mysql_source`), 25
[MySQLCopy](#) (class in `twindb_backup.copy.mysql_copy`), 13

MySQLMasterInfo (class in *twindb_backup.source.mysql_source*), 25
 MySQLSource (class in *twindb_backup.source.mysql_source*), 25
 MySQLSourceError, 24
 MySQLStatus (class in *twindb_backup.status.mysql_status*), 31

N

name (*twindb_backup.copy.binlog_copy.BinlogCopy* attribute), 13
 name (*twindb_backup.copy.periodic_copy.PeriodicCopy* attribute), 14
 netcat() (*twindb_backup.destination.ssh.Ssh* method), 20
 next_backup_type() (*twindb_backup.status.mysql_status.MySQLStatus* method), 31
 normalize_b64_data() (in module *twindb_backup.util*), 38

O

OperationError, 34

P

parent (*twindb_backup.copy.mysql_copy.MySQLCopy* attribute), 14
 path (*twindb_backup.destination.local.Local* attribute), 16
 PeriodicCopy (class in *twindb_backup.copy.periodic_copy*), 14
 port (*twindb_backup.destination.ssh.Ssh* attribute), 20
 port (*twindb_backup.ssh.client.SshClient* attribute), 29
 position (*twindb_backup.copy.mysql_copy.MySQLCopy* attribute), 14
 private (*twindb_backup.destination.s3.S3FileAccess* attribute), 19
 public_read (*twindb_backup.destination.s3.S3FileAccess* attribute), 19
 purge() (*twindb_backup.cache.cache.Cache* method), 12

R

read() (*twindb_backup.destination.base_destination.BaseDestination* method), 15
 read() (*twindb_backup.destination.local.Local* method), 16
 read() (*twindb_backup.destination.s3.S3* method), 18
 read() (*twindb_backup.destination.ssh.Ssh* method), 20
 RemoteMySQLSource (class in *twindb_backup.source.remote_mysql_source*), 27
 RemoteMySQLSourceError, 24

in remove() (*twindb_backup.status.base_status.BaseStatus* method), 30
 in restore (*twindb_backup.exporter.base_exporter.ExportMeasureType* attribute), 21
 restore_from_file() (in module *twindb_backup.restore*), 36
 restore_from_mysql() (in module *twindb_backup.restore*), 36
 restore_from_mysql_full() (in module *twindb_backup.restore*), 36
 restore_from_mysql_incremental() (in module *twindb_backup.restore*), 37
 restore_in() (*twindb_backup.cache.cache.Cache* method), 12
 retention (*twindb_backup.configuration.TwinDBBackupConfig* attribute), 34
 retention_local (*twindb_backup.configuration.TwinDBBackupConfig* attribute), 34
 revert_stream() (*twindb_backup.modifiers.base.Modifier* method), 22
 run_backup_job() (in module *twindb_backup.backup*), 33
 run_command() (in module *twindb_backup.util*), 38
 run_intervals (*twindb_backup.configuration.TwinDBBackupConfig* attribute), 34
 run_type (*twindb_backup.copy.periodic_copy.PeriodicCopy* attribute), 14
 run_type (*twindb_backup.source.base_source.BaseSource* attribute), 24

S

S3 (class in *twindb_backup.destination.s3*), 17
 s3 (*twindb_backup.configuration.TwinDBBackupConfig* attribute), 34
 S3DestinationError, 16
 S3FileAccess (class in *twindb_backup.destination.s3*), 19
 save() (*twindb_backup.destination.base_destination.BaseDestination* method), 15
 save() (*twindb_backup.destination.local.Local* method), 16
 save() (*twindb_backup.destination.s3.S3* method), 18
 save() (*twindb_backup.destination.ssh.Ssh* method), 20
 save() (*twindb_backup.status.base_status.BaseStatus* method), 30
 save_measures() (in module *twindb_backup*), 40
 serialize() (*twindb_backup.copy.mysql_copy.MySQLCopy* method), 14
 serialize() (*twindb_backup.status.base_status.BaseStatus* method), 30
 session() (*twindb_backup.ssh.client.SshClient* method), 29

- set_open_files_limit() (in module `twindb_backup.backup`), 33
 setup_logging() (in module `twindb_backup`), 40
 setup_s3_client() (in module `twindb_backup.destination.s3.S3` static method), 18
 setup_slave() (in module `twindb_backup.source.remote_mysql_source.RemoteMySQLSource` method), 28
 share() (in module `twindb_backup.share`), 37
 share() (in module `twindb_backup.destination.s3.S3` method), 18
 SourceError, 24
 split_host_port() (in module `twindb_backup.util`), 38
 Ssh (class in `twindb_backup.destination.ssh`), 19
 ssh (in module `twindb_backup.configuration.TwinDBBackupConfig` attribute), 34
 SshClient (class in `twindb_backup.ssh.client`), 28
 SshClientException, 29
 SshDestinationError, 16
 status (in module `twindb_backup.source.mysql_source.MySQLSource` attribute), 27
 status_path (in module `twindb_backup.status.base_status.BaseStatus` attribute), 30
 StatusError, 31
 StatusKeyNotFound, 31
 suffix (in module `twindb_backup.modifiers.gzip.Gzip` attribute), 23
 suffix (in module `twindb_backup.source.base_source.BaseSource` attribute), 24
- ## T
- timeout() (in module `twindb_backup.backup`), 33
 twindb_backup (module), 39
 twindb_backup.backup (module), 32
 twindb_backup.cache (module), 12
 twindb_backup.cache.cache (module), 11
 twindb_backup.cli (module), 33
 twindb_backup.clone (module), 33
 twindb_backup.configuration (module), 33
 twindb_backup.copy (module), 14
 twindb_backup.copy.base_copy (module), 12
 twindb_backup.copy.binlog_copy (module), 12
 twindb_backup.copy.exceptions (module), 13
 twindb_backup.copy.mysql_copy (module), 13
 twindb_backup.copy.periodic_copy (module), 14
 twindb_backup.destination (module), 21
 twindb_backup.destination.base_destination (module), 15
 twindb_backup.destination.exceptions (module), 16
 twindb_backup.destination.local (module), 16
- twindb_backup.destination.s3 (module), 17
 twindb_backup.destination.ssh (module), 19
 twindb_backup.exceptions (module), 34
 twindb_backup.export (module), 35
 twindb_backup.exporter (module), 22
 twindb_backup.exporter.base_exporter (module), 21
 twindb_backup.exporter.datadog_exporter (module), 21
 twindb_backup.exporter.exceptions (module), 22
 twindb_backup.ls (module), 35
 twindb_backup.modifiers (module), 23
 twindb_backup.modifiers.base (module), 22
 twindb_backup.modifiers.gpg (module), 22
 twindb_backup.modifiers.gzip (module), 23
 twindb_backup.modifiers.keeplocal (module), 23
 twindb_backup.restore (module), 35
 twindb_backup.share (module), 37
 twindb_backup.source (module), 28
 twindb_backup.source.base_source (module), 23
 twindb_backup.source.exceptions (module), 24
 twindb_backup.source.file_source (module), 24
 twindb_backup.source.mysql_source (module), 25
 twindb_backup.source.remote_mysql_source (module), 27
 twindb_backup.ssh (module), 30
 twindb_backup.ssh.client (module), 28
 twindb_backup.ssh.exceptions (module), 29
 twindb_backup.status (module), 32
 twindb_backup.status.base_status (module), 30
 twindb_backup.status.exceptions (module), 31
 twindb_backup.status.mysql_status (module), 31
 twindb_backup.util (module), 37
 twindb_backup.verify (module), 38
 TwinDBBackupConfig (class in `twindb_backup.configuration`), 33
 TwinDBBackupError, 35
 TwinDBBackupInternalError, 35
 type (in module `twindb_backup.copy.mysql_copy.MySQLCopy` attribute), 14
 type (in module `twindb_backup.source.mysql_source.MySQLSource` attribute), 27
- ## U
- UnknownSourceType, 13

`update_grastate()` (in module `twindb_backup.restore`), 37
`user` (`twindb_backup.destination.ssh.Ssh` attribute), 20
`user` (`twindb_backup.ssh.client.SshClient` attribute), 29

V

`validate_client_response()` (`twindb_backup.destination.s3.S3` static method), 19
`variable()` (`twindb_backup.source.mysql_source.MySQLClient` method), 25
`verify_mysql_backup()` (in module `twindb_backup.verify`), 38
`version` (`twindb_backup.status.base_status.BaseStatus` attribute), 31

W

`write()` (`twindb_backup.destination.base_destination.BaseDestination` method), 15
`write()` (`twindb_backup.destination.local.Local` method), 17
`write()` (`twindb_backup.destination.s3.S3` method), 19
`write()` (`twindb_backup.destination.ssh.Ssh` method), 21
`write_config()` (`twindb_backup.ssh.client.SshClient` method), 29
`write_content()` (`twindb_backup.ssh.client.SshClient` method), 29
`WrongInputData`, 13
`wsrep_provider_version` (`twindb_backup.copy.mysql_copy.MySQLCopy` attribute), 14
`wsrep_provider_version` (`twindb_backup.source.mysql_source.MySQLSource` attribute), 27