Whalesong Documentation

Release 0.9.1

alfred82santa

May 29, 2019

Contents

Whalesong is an asyncio python library to manage WebApps remotely. Currently WhatsappWeb is implemented

Warning: NEW VERSION 0.7.0

With new version some new requirements have been defined and some API change have been committed.

Version 0.7.0 introduces new driver for Chromium browser. It has required some changes on base driver's API. It should not affect anyone. But be aware.

Warning: NEW VERSION 0.7.

Stream model now uses enumerations. It means that if you check stream states you must be aware that they are not a string anymore, now they are enumerations. Look at *documentation*.

CHAPTER 1

Binary Requirements

- Python 3.6+

1.1 Using Firefox

- Geckodriver
- Firefox 50+

1.2 Using Chromium

- Execute this after installation:

```
$ pyppeteer-install
```

1.3 Development requirements

- node (only for development)
- npm (only for development)
- make (only for development)

CHAPTER 2

Installation

Starting with version *0.7.0* Whalesong introduces new browsers backends. It means you must decide which browser backend you want to use. So, depending on it:

Tip: You could install both with no problems.

2.1 Using Firefox

```
$ pip3 install whalesong[firefox]
```

2.2 Using Chronium

```
$ pip3 install whalesong[chromium]
```

If you choose Chromium backend, you should execute this after installation:

```
$ pyppeteer-install
```

It will download a patched Chromium distribution needed for that backend.

CHAPTER 3

Table of contents

3.1 Features

- Non blocking driver. It can do more than one thing at same time.
- Monitor feature. Python code could react to Javascript event.
- Iterator feature. Large item list are passed to Python as async iterators.
- Persistent browser profile.
- Easy way to build new features.
- AmpersandJS/BackboneJS models and collection monitor.
- AmpersandJS/BackboneJS field monitor.
- Monitor localStorage.
- Take screenshots (page and css elements).
- Powered by AsyncIO.
- Firefox backend.
- Chromium backend.

3.1.1 Whatsapp features

- Monitor connection state.
- Monitor QR changes.
- Monitor stream state.
- It's able to refresh QR.
- It's able to take over session.

- List chats.
- List contacts.
- List messages
- Monitor new messages.
- Monitor unread messages.
- Monitor message acks.
- Monitor new contacts.
- Monitor new chats.
- Send text messages.
- Reply messages.
- Send VCard messages.
- Send Media (image/video/pdf) messages.
- Send seen to chats.
- Create groups.
- Manage groups (add/kick/promote/demote people).
- Auto load link previews.
- Allow to check whether a contact is registered on Whatsapp.
- Load and send stickers (even with a quoted message).
- Presence monitor.
- Profile and group's picture management.
- (Un)Pin and (un)archive chats.
- Report spam.
- (Un)Star messages.
- Status management.
- Pushname management.
- Display information management.
- Live location management.
- Mutes management.
- It's able to revoke messages (delete messages for others).
- List and manage WhatsApp Status version 3 (Stories).

3.2 Browser backends

Whalesong uses a browser backend in order to execute a WebApp (currently only WhatsAppWeb). All backends have an interface to manage webviews and that is what Whalesong uses to manage applications. That interface changes depending on browser, but there is a standard interface called WebDriver. Firsts Whalesong versions used to use a Selenium library in order to communicate with Firefox browser. This backend is the default one for now, **but it will be deprecated in next versions and removed in version 1.0**.

3.2.1 Firefox backend

It was the first backend developed. It uses Selenium library and Geckodriver to communicate with Firefox process. It is the most tested (the most, but not well).

3.2.1.1 Pros

- Tested (more or less).
- Use Firefox (I prefer it in front Chromium).

3.2.1.2 Contras

- Selenium is a huge library. It is wonderful for what it was created, but not for Whalesong.
- Selenium is a synchronous library. It is a problem, because Whalesong is an asynchronous library. It means, Whalesong creates a thread pool to communicate with Selenium.
- We need Geckodriver. Firefox does not implement Webdriver protocol by itself. Firefox has its own protocol called Marionette. So Geckodriver is used as proxy between Webdriver protocol and Marionette protocol.
- As Webdriver is a synchronous protocol. Whalesong must poll continuously to Firefox in order to get new events. There is no way to make Firefox notify Whalesong proactively. It means, Whalesong is polling for new results continuously, with an interval (by default 0.5 seconds).

Note: There is only one way for the Firefox backend to survive:

Drop Selenium, drop Geckodriver, implement Marionette protocol directly and implement a notification system (I'm not sure it is possible in Marionette, currently).

3.2.1.3 How to use

Currently Firefox backend is the default one. But it will change on next versions. So, in order to ensure you use Firefox backend you must instantiate Whalesong with proper driver.

```
from whalesong import Whalesong
from whalesong.driver_firefox import WhalesongDriver

driver = WhalesongDriver(profile='/path/to/your/firefox/profile')
whaleapp = Whalesong(driver=driver)
```

3.2.2 Chromium backend

It is the new one. It is implemented using Pyppetter which is inspired on Puppetter (a *node* library to control Chromium headless, mainly, for testing). It uses Devtools protocol in order to communicate with the browser. It is an asynchronous protocol over websocket.

3.2.2.1 Pros

- No more polling! When a result is produced it will send proactively to Whalesong. No more *sleeps*, no more waitings.

- Small footprint (at least, it looks like, even being Chromium).
- No extra processes (No more Geckodriver).
- Application mode. No tabs, no URL field.
- No huge libraries (No more Selenium).

3.2.2.2 Contras

- It is Chromium. It uses Blink: over-vitaminized Webkit render. A memory eater.
- Currently Pypperter has a bug. It makes to loose connection after 20 seconds. It is resolved in miyakogi/pyppeteer/#160 but is not approved yet (some test errors).
- It uses a patched Chromium version from Puppetter. Whalesong needs this patch because it uses *Runtime.addBinding* command. It is not available in regular stable version. So, you must download it before to use the backend.
- Poorly tested.
- There is a bug in Chromium under Wayland. It makes impossible to get WhastappWeb QR when Chromium is executed with window (no headless).

3.2.2.3 How to use

In order to use Chromium backend you must inject Chromium driver to Whalesong service constructor.

```
from whalesong import Whalesong
from whalesong.driver_chromium import WhalesongDriver

driver = WhalesongDriver(profile='/path/to/your/chromium/profile')
whaleapp = Whalesong(driver=driver)
```

3.2.3 Other backends

No, there are no other backends. But I'm thinking about other possibilities:

- Create a small footprint webview using Webkit directly (GTK or QT ways are not an option).
- Create a small footprint webview using Servo directly (I want to learn rust language).

Of course, any contribution will be welcome (so much).

3.3 Examples

3.3.1 Raw driver examples

3.3.1.1 State monitor

It monitors Stream state, Connection state and localStorage. It prints any change on them. It takes a page screenshot on each stream state change.

It tries to own WhatsappWeb session, it means that it will restore session if you open a new session in other browser.

On the other hand, if session is not started, it will renew QR automatically when it expires. It will save QR image each time it changes.

```
$ PYTHONPATH=.:$PYTHONPATH python3 examples/driver/statemonitor.py
```

3.3.1.2 Get contacts

It prints contact list.

```
$ PYTHONPATH=.:$PYTHONPATH python3 examples/driver/getcontacts.py
```

3.3.1.3 Get chats

It prints chat list.

```
$ PYTHONPATH=.:$PYTHONPATH python3 examples/driver/getchats.py
```

3.3.1.4 Get messages

It prints message list and monitors it. So, if new messages are received it will print them. It monitors message acknowledgments and prints them, as well.

```
$ PYTHONPATH=.:$PYTHONPATH python3 examples/driver/getmessages.py
```

3.3.2 Whatsapp driver examples

3.3.2.1 State monitor

It monitors Stream state, Connection state and localStorage. It prints any change on them. It takes a page screenshot on each stream state change.

It tries to own WhatsappWeb session, it means that it will restore session if you open a new session in other browser.

On the other hand, if session is not started, it will renew QR automatically when it expires. It will save QR image each time it changes.

Firefox backend

```
$ PYTHONPATH=.:$PYTHONPATH python3 examples/statemonitor.py
```

Chromium backend

```
$ PYTHONPATH=.:$PYTHONPATH python3 examples/statemonitor-chromium.py
```

3.3.2.2 Get contacts

It prints contact list.

```
$ PYTHONPATH=.:$PYTHONPATH python3 examples/getcontacts.py
```

3.3.2.3 Get chats

It prints chat list.

```
$ PYTHONPATH=.:$PYTHONPATH python3 examples/getchats.py
```

3.3.2.4 Get messages

It prints message list and monitors it. So, if new messages are received it will print them. It monitors message acknowledgments and prints them, as well.

It stores files and thumbnails from media messages.

Firefox backend

```
$ PYTHONPATH=.:$PYTHONPATH python3 examples/getmessages.py
```

Chromium backend

```
$ PYTHONPATH=.:$PYTHONPATH python3 examples/getmessages-chromium.py
```

3.3.2.5 Minibot

Mini bot to test features.

Firefox backend

```
$ PYTHONPATH=.:$PYTHONPATH python3 examples/minibot.py
```

Chromium backend

```
$ PYTHONPATH=.:$PYTHONPATH python3 examples/minibot-chromimum.py
```

Minibot implements some test features:

Echo

When a contact sends */echo [text]* it replies with *[text]*.

Example

```
/echo Hello!
```

Contact

When a contact sends */contact [contactID]* it replies with the contact in VCard format.

Example

```
/contact 495555555555
```

Download

When a contact sends */download [url]* it replies with content pointed by URL (image, pdf, video).

Example

```
/download http://example.com/image.jpg
```

Send

When a contact sends */send [contactId] [text]* it will send *[text]* to *[contactId]*. *[contactId]* must be a phone number with country prefix: 495555555555 where *49* is Germany prefix.

Example

```
/send 495555555555 Hello!
```

Link

When a contact sends */link [text]* it replies with *[text]*. It's very similar to */echo*, but never quote original message and if there was a link it will try to get link preview and attach it.

Example

```
/link https://www.google.com
```

Exist

When a contact sends */exist [contactId]*, it will return whether a phone number is registered on Whatsapp.

Example

```
/exist 495555555555
```

List sticker packs

When a contact sends */sticker list*, it will send all sticker pack names with main image attached.

Example

```
/sticker list
```

Send random sticker

When a contact sends */sticker [stickerPackName]*, it will send a random sticker from sticker pack with name *[stickerPackName]*.

Example

```
/sticker Cuppy
```

Set bot status

When a contact sends */status [newStatus]*, it will change its own status to *[newStatus]*.

Example

```
/status I'm a bot
```

Set bot pushname

When a contact sends */pushname [name]*, it will change its own pushname to *[name]*.

Example

```
/pushname I'm a bot
```

3.3.2.6 Get stickers

It fetch all sticker packs installed. It fetch all sticker for each sticker pack. And, finally, it downloads all sticker images.

```
$ PYTHONPATH=.:$PYTHONPATH python3 examples/getstickers.py
```

3.3.2.7 Monitor presences

It monitors user presences. It prints any change on them.

Firefox backend

```
$ PYTHONPATH=.:$PYTHONPATH python3 examples/presencemonitor.py
```

Chromium backend

```
$ PYTHONPATH=.:$PYTHONPATH python3 examples/presencemonitor-chromium.py
```

3.3.2.8 Get statuses

It will get all unread statuses, download the media content to the *output* folder and send a read command.

Firefox backend

```
$ PYTHONPATH=.:$PYTHONPATH python3 examples/getstatusv3.py
```

Chromium backend

```
$ PYTHONPATH=.:$PYTHONPATH python3 examples/getstatusv3-chromium.py
```

3.3.2.9 Get live locations

It will get all active live locations and will monitor them.

Firefox backend

```
$ PYTHONPATH=.:$PYTHONPATH python3 examples/getlivelocations.py
```

Chromium backend

```
$ PYTHONPATH=.:$PYTHONPATH python3 examples/getlivelocations-chromium.py
```

3.4 API reference

3.4.1 Whalesong Service

class whalesong.**Whalesong**(*profile=None*, ***, *autostart=True*, *headless=False*, *extra_options=None*, *driver=None*, *loop=None*, ***kwargs*)

Bases: *whalesong.managers.BaseManager*

Whalesong service.

The main Whalesong manager.

Parameters

- **profile** (Optional[str]) – Path to firefox profile.
- **autostart** (bool) – Whether driver must start immediately.
- **headless** (bool) – Whether browser must be started with headless flag. In production environments it should be set to True.
- **extra_options** (Optional[dict]) – Extra parametres for browser commandline.
- **loop** (Optional[AbstractEventLoop]) – Event loop.
- **loadstyles** (*bool*) – Whether CSS styles must be loaded. It is need in order to get QR image. (Only for Firefox)
- **interval** (*float*) – Polling responses interval in seconds. Default 0.5 seconds. (Only for Firefox)

storage
StorageManager

Manager for local storage.

stream
StreamManager

Manager for stream object.

conn
ConnManager

Manager for connection object

contacts
ContactCollectionManager

Manager for contact collection.

chats
ChatCollectionManager

Manager for chat collection.

messages
MessageCollectionManager

Manager for messages collection.

wap
WapManager

Manager for wap object.

sticker_packs
StickerPackCollectionManager

Manager for sticker pack collection.

status
StatusCollectionManager

Manager for status collection.

display_info
DisplayInfoManager

Manager for display information.

live_locations
LiveLocationCollectionManager

Manager for live locations collection.

mutes
MuteCollectionManager

Manager for mutes collection.

status_v3
StatusV3CollectionManager

Manager for statuses version 3 (Stories) collection.

loop
Event loop.

Returns Event loop.

await start()
Start Whalesong service.

await stop()
Stop Whalesong service.

await wait_until_stop()
Wait until Whalesong service is stopped.

await screenshot()
Take a screenshot of whole page.

Return type BytesIO

Returns It returns a stream of a PNG image.

await qr()
Take a screenshot of QR.

Return type BytesIO

Returns It returns a stream of a PNG image.

stop_monitor(*monitor*)
Stop a given monitor.

Parameters **monitor** (*MonitorResult*[~T]) – Monitor object to stop.

Return type *Result*[None]

await cancel_iterators()
Cancel all iterators.

await download_file(*url*)
Download a file by URL

Parameters **url** (str) – URL to the file

Return type BytesIO

Returns It returns a stream.

3.4.2 Whalesong Driver

3.4.2.1 Base driver

class whalesong.driver.**BaseWhalesongDriver**(*, *autostart=True*, *headless=False*, *extra_options=None*, *logger=None*, *loop=None*)

Bases: abc.ABC

await start_driver()

abstractmethod await connect()

abstractmethod await refresh()

await run_scriptlet()

await screenshot()

Return type BytesIO

await screenshot_element(*css_selector*)

Return type BytesIO

execute_command(*command*, *params=None*, *, *result_class=None*)

process_result_sync(*result*)

await process_result(*result*)

await close()

await cancel_iterators()

await cancel_monitors()

await download_file(*url*)

Return type BytesIO

await wait_until_stop()

3.4.2.2 Firefox driver

class whalesong.driver_firefox.**WhalesongDriver**(*profile=None*, *, *autostart=True*, *headless=False*, *interval=0.5*, *loadstyles=False*, *extra_options=None*, *logger=None*, *loop=None*)

Bases: *whalesong.driver.BaseWhalesongDriver*

free_port()
Determines a free port using sockets.

await connect()

await refresh()

await poll()

3.4.2.3 Chromium driver

class whalesong.driver_chromium.**WhalesongDriver**(*profile=None*, ***, *autostart=True*, *headless=False*, *extra_options=None*, *logger=None*, *loop=None*)

Bases: *whalesong.driver.BaseWhalesongDriver*

await connect()

await refresh()

3.4.3 Result types

class whalesong.results.**BaseResultMixin**(*result_id*, ***, *fn_map=None*)

Bases: abc.ABC, typing.Awaitable

Base result mixin.

result_id: str
Result unique identifier.

fn_map: Callable[[dict], Union[BaseModel, dict]]
Mapping function used to map result.

Parameters

- **result_id** (str) – Result unique identifier.
- **fn_map** (Optional[Callable[[dict], ~T]]) – Mapping function used to map result.

map(*data*)
Maps data from browser to an object if *fn_map* function is defined.

Parameters **data** (dict) – Data from browser.

Return type ~T

Returns Mapped object.

await set_final_result(*data*)

await set_error_result(*data*)

class whalesong.results.**Result**(*result_id*, ***, *fn_map=None*)

Bases: *whalesong.results.BaseResultMixin*, _asyncio.Future

Result of command. It is a subtype of asyncio.Future, so in order to get result value you must *await* it.

```
value = await result
```

Parameters

- **result_id** (str) – Result unique identifier.
- **fn_map** (Optional[Callable[[dict], ~T]]) – Mapping function used to map result.

class whalesong.results.**BasePartialResult**(*result_id*, *, *fn_map=None*)
Bases: *whalesong.results.BaseResultMixin*, typing.AsyncIterable

await set_partial_result(*data*)

cancel()

class whalesong.results.**BaseIteratorResult**(*result_id*, *, *fn_map=None*)
Bases: *whalesong.results.BasePartialResult*

Base iterable result.

class whalesong.results.**IteratorResult**(*result_id*, *, *fn_map=None*)
Bases: *whalesong.results.BaseIteratorResult*

Iterator result. It is used as result of command which returns a list of object.

> **Warning:** It is an async iterator.

How to use

```
async for item in result_iterator:
    print(item)
```

map(*data*)
Maps data from browser to an object if *fn_map* function is defined.

Parameters **data** – Data from browser.

Return type ~T

Returns Mapped object.

class whalesong.results.**MonitorResult**(*result_id*, *, *fn_map=None*)
Bases: *whalesong.results.BaseIteratorResult*

Monitor result. It is used as result of monitor command. It is a infinite iterator. Each change on object or field it is monitoring will be a new item on iterator.

> **Warning:** It is an async iterator.

How to use

```
async for item_changed in monitor:
    print(item_changed)
```

add_callback(*fn*)
Add a callback to be called each time object or field change.

Parameters **fn** (Callable[[~T], Awaitable[Any]]) – Callback function

start_monitor()
Starts automatic monitor iteration. Useful when callback functions are defined.

class whalesong.results.**ResultManager**
Bases: object

get_next_id()

Return type str

remove_result(*result_id*)

Return type Union[*Result*[~T], *IteratorResult*[~T], *MonitorResult*[~T], None]

request_result(*result_class*)

Return type ~TypeResult

request_final_result()

Return type *Result*[~T]

request_iterator_result()

Return type *IteratorResult*[~T]

request_monitor_result()

Return type *MonitorResult*[~T]

cancel_result(*result_id*)

cancel_all()

await set_final_result(*result_id*, *data*)

await set_error_result(*result_id*, *data*)

await set_partial_result(*result_id*, *data*)

get_iterators()

Return type List[*IteratorResult*[~T]]

get_monitors()

Return type List[*MonitorResult*[~T]]

3.4.4 Base Models

class whalesong.models.**Base64Field**(*name=None*, *alias=None*, *getter=None*, *setter=None*, *read_only=False*, *default=None*, *doc=None*)

Bases: dirty_models.fields.BytesField

Byte field which allows to set base64 string data.

convert_value(*value*)

Converts value to field type

class whalesong.models.**DateTimeField**(*parse_format=None*, *default_timezone=None*, *force_timezone=False*, ***kwargs*)

Bases: dirty_models.fields.DateTimeField

Date time field that allow timestamps in microseconds.

Parameters

- **parse_format** (*str or dict*) – String format to cast string to datetime. It could be an string format or a dict with two keys:
  - parser key to set how string must be parsed. It could be a callable.
  - formatter key to set how datetime must be formatted. It could be a callable.
- **default_timezone** (*datetime.tzinfo*) – Default timezone to use when value does not have one.

- **force_timezone** – If it is True value will be converted to timezone defined on `default_timezone` parameter. It `default_timezone` is not defined it is ignored.

Type bool

convert_value(*value*)
Converts value to field type

Model whalesong.models.**BaseModel**(*data=None*, *flat=False*, **args*, ***kwargs*)
Bases: dirty_models.models.BaseModel

Base model which convert field name from underscore-style to camelCase-style automatically.

property id : str *[READ ONLY]*
Unique identifier.

class whalesong.models.**ModelFormatterIter**(*model*)
Bases: dirty_models.utils.ModelFormatterIter

format_field(*field*, *value*)

class whalesong.models.**JSONEncoder**(***, *skipkeys=False*, *ensure_ascii=True*, *check_circular=True*, *allow_nan=True*, *sort_keys=False*, *indent=None*, *separators=None*, *default=None*)
Bases: dirty_models.utils.JSONEncoder

Constructor for JSONEncoder, with sensible defaults.

If skipkeys is false, then it is a TypeError to attempt encoding of keys that are not str, int, float or None. If skipkeys is True, such items are simply skipped.

If ensure_ascii is true, the output is guaranteed to be str objects with all incoming non-ASCII characters escaped. If ensure_ascii is false, the output can contain non-ASCII characters.

If check_circular is true, then lists, dicts, and custom encoded objects will be checked for circular references during encoding to prevent an infinite recursion (which would cause an OverflowError). Otherwise, no such check takes place.

If allow_nan is true, then NaN, Infinity, and -Infinity will be encoded as such. This behavior is not JSON specification compliant, but is consistent with most JavaScript based encoders and decoders. Otherwise, it will be a ValueError to encode such floats.

If sort_keys is true, then the output of dictionaries will be sorted by key; this is useful for regression tests to ensure that JSON serializations can be compared on a day-to-day basis.

If indent is a non-negative integer, then JSON array elements and object members will be pretty-printed with that indent level. An indent level of 0 will only insert newlines. None is the most compact representation.

If specified, separators should be an (item_separator, key_separator) tuple. The default is (', ', ': ') if *indent* is `None` and (',', ': ') otherwise. To get the most compact JSON representation, you should specify (',', ':') to eliminate whitespace.

If specified, default is a function that gets called for objects that can't otherwise be serialized. It should return a JSON encodable version of the object or raise a `TypeError`.

default_model_iter
alias of *ModelFormatterIter*

3.4.5 Base Managers

class whalesong.managers.**BaseManager**(*driver*, *manager_path=''*)
Bases: object

Base manager.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

class whalesong.managers.**BaseModelManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseManager*, typing.Generic

Base model manager.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

classmethod map_model(*data*)

Return type +MODEL_TYPE

classmethod get_model_result_class()

Return type *Result*[+MODEL_TYPE]

classmethod get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

classmethod get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

class whalesong.managers.**BaseCollectionManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseManager*, typing.Generic

Base collection manager.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

classmethod get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

classmethod get_iterator_result_class()

Return type *IteratorResult*[+MODEL_TYPE]

classmethod get_item_result_class()

Return type *Result*[+MODEL_TYPE]

get_items()
Get all items on collection.

Return type *IteratorResult*[+MODEL_TYPE]

Returns Async iterator

get_length()
Get collection items count.

Return type *Result*[int]

Returns Items count

get_item_by_id(*item_id*)
Get model by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

find_item_by_id(*item_id*)
Find model by identifier. If item is not in collection it will try to load it.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

remove_item_by_id(*item_id*)
Remove item by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[None]

get_first()
Get first item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_last()
Get last item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

monitor_add()
Monitor add item collection. Iterate each time a item is added to collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_remove()
Monitor remove item collection. Iterate each time a item is removed from collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_change()
Monitor change item collection. Iterate each time a item change in collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_field(*field*)
Monitor item's field change. Iterate each time a field changed in any item of collection.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model object iterator

get_submanager(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

3.4.6 Contact references

3.4.6.1 Managers

class whalesong.managers.contact.**ContactCollectionManager**(*driver*, *manager_path=''*)

Bases: *whalesong.managers.BaseCollectionManager*

Contact collection manager. It allows manage contact collection.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)

Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

__getattr__(*name*)

Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

add_submanager(*name*, *submanager*)

Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

find_item_by_id(*item_id*)

Find model by identifier. If item is not in collection it will try to load it.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_commands()

Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_first()

Get first item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_by_id(*item_id*)

Get model by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_result_class()

Return type *Result*[+MODEL_TYPE]

get_items()
Get all items on collection.

Return type *IteratorResult*[+MODEL_TYPE]

Returns Async iterator

get_iterator_result_class()

Return type *IteratorResult*[+MODEL_TYPE]

get_last()
Get last item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_length()
Get collection items count.

Return type *Result*[int]

Returns Items count

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

monitor_add()
Monitor add item collection. Iterate each time a item is added to collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_change()
Monitor change item collection. Iterate each time a item change in collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_field(*field*)
Monitor item's field change. Iterate each time a field changed in any item of collection.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model object iterator

monitor_remove()
Monitor remove item collection. Iterate each time a item is removed from collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

remove_item_by_id(*item_id*)
Remove item by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[None]

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

MODEL_MANAGER_CLASS
alias of *ContactManager*

resync_contacts()

Return type *Result*[None]

get_me()

Return type *Result*[*Contact*]

class whalesong.managers.contact.**ContactManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseModelManager*

Contact manager. It allows manage a contact.

profile_pic_thumb
ProfilePictureManager

Contact's picture thumb manager.

__getitem__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

MODEL_CLASS
alias of *Contact*

block()
Block contact.

Return type *Result*[None]

unblock()
Unblock contact.

Return type *Result*[None]

3.4.6.2 Models

Model whalesong.managers.contact.**Contact**(*data=None*, *flat=False*, **args*, ***kwargs*)

Bases: *whalesong.models.BaseModel*

property name : str
Contact name. It will be name on phone contact list.

property pushname : str
Contact defined name. It is set by contact on its whatsapp application.

property type : str
¿?

property userhash : str
¿?

property userid : str
User identifier. It used to be the phone number.

property status : *Status*
Contact status.

property id : str *[READ ONLY]*
StringIdField field [READ ONLY]

property formattedName : str
StringIdField field

Aliases

- **formatted_name**

property shortName : str
StringIdField field

Aliases

- **short_name**

property verifiedLevel : str
StringIdField field

Aliases

- **verified_level**

property verifiedName : str
StringIdField field

Aliases

- **verified_name**

property profilePicThumbObj : *ProfilePicture*
ModelField field (*whalesong.managers.profile_pic_thumb.ProfilePicture*)

Aliases

- **profile_pic_thumb_obj**

property isUser : bool

Default value True

BooleanField field

Aliases

- **is_user**

property isBusiness : bool

Default value False

BooleanField field

Aliases

- **is_business**

property isContactBlocked : bool

Default value False

BooleanField field

Aliases

- **is_contact_blocked**

property isEnterprise : bool

Default value False

BooleanField field

Aliases

- **is_enterprise**

property isHighLevelVerified : bool

Default value False

BooleanField field

Aliases

- **is_high_level_verified**

property isMe : bool

Default value False

BooleanField field

Aliases

- **is_me**

property isMyContact : bool

Default value True

BooleanField field

Aliases

- **is_my_contact**

property isPSA : bool

Default value False

BooleanField field

Aliases

- **is_psa**

property isVerified : bool

Default value False

BooleanField field

Aliases

- **is_verified**

property isWAContact : bool

Default value True

BooleanField field

Aliases

- **is_wa_contact**

property plaintextDisabled : bool

Default value False

BooleanField field

Aliases

- **plaintext_disabled**

property statusMute : bool

Default value False

BooleanField field

Aliases

- **status_mute**

property sectionHeader : str

StringIdField field

Aliases

- **section_header**

property labels : List of str

Default value []

List of contact labels ¿?

to_vcard()

Build vCard from contact.

Return type <function vCard at 0x7f6946120d90>

Returns vCard object of contact

3.4.7 Chat references

3.4.7.1 Managers

class whalesong.managers.chat.**ChatCollectionManager**(*driver*, *manager_path=''*)

Bases: *whalesong.managers.BaseCollectionManager*

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)

Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

__getattr__(*name*)

Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

add_submanager(*name*, *submanager*)

Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

find_item_by_id(*item_id*)

Find model by identifier. If item is not in collection it will try to load it.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_commands()

Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_first()

Get first item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_by_id(*item_id*)

Get model by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_result_class()

Return type *Result*[+MODEL_TYPE]

get_items()
Get all items on collection.

Return type *IteratorResult*[+MODEL_TYPE]

Returns Async iterator

get_iterator_result_class()

Return type *IteratorResult*[+MODEL_TYPE]

get_last()
Get last item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_length()
Get collection items count.

Return type *Result*[int]

Returns Items count

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

monitor_add()
Monitor add item collection. Iterate each time a item is added to collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_change()
Monitor change item collection. Iterate each time a item change in collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_field(*field*)
Monitor item's field change. Iterate each time a field changed in any item of collection.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model object iterator

monitor_remove()
Monitor remove item collection. Iterate each time a item is removed from collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

remove_item_by_id(*item_id*)
Remove item by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[None]

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

MODEL_MANAGER_CLASS
alias of *ChatManager*

get_active()
Returns chat selected.

Return type *Result*[*Chat*]

Returns Chat object

resync_messages()
Resynchronize messages.

Return type *Result*[None]

ensure_chat_with_contact(*contact_id*)
Ensure there is a chat with a given contact. If it does not exist it will be created.

Parameters **contact_id** (str) – Contact's identifier.

Return type *Result*[*Chat*]

Returns Chat object

create_group(*name*, *contact_ids*, *picture=None*, *picture_preview=None*)
Create a new chat group.

Parameters

- **name** (str) – Group's name
- **contact_ids** (List[str]) – List of contact identifier to invite.
- **picture** (Optional[BytesIO]) – JPG image.

Return type *Result*[*Chat*]

Returns Chat object.

forward_messages_to_chats(*message_ids*, *chat_ids*)
Forward messages.

Return type *Result*[dict]

Returns Operation result.

class whalesong.managers.chat.**ChatManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseModelManager*

Chat manager. It allows manage a chat.

msgs
MessageCollectionManager

Chat's message collection manager.

msg_load_state
MsgLoadStateManager

Chat's message load state manager.

metadata:
GroupMetadataManager

Chat's group metadata manager.

contact
ContactManager

Chat's contact manager.

live_location
LiveLocationManager

Live location manager. You should call to *find_live_location()* before use it.

__getitem__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

MODEL_CLASS
alias of *Chat*

send_text(*text*, *quoted_msg_id=None*, *mentions=None*, *link_desc=None*)
Send text message to current chat.

Parameters

- **text** (str) – Message to send.
- **quoted_msg_id** (Optional[str]) – Quoted message's identifier.
- **mentions** (Optional[List[str]]) – List of user ids mentioned.
- **link_desc** – Link description.

Return type *Result*[str]

Returns New message's identifier

send_contact(*contact_id*, *quoted_msg_id=None*)
Send contact to current chat.

Parameters

- **contact_id** (str) – Contact identifier to send.
- **quoted_msg_id** (Optional[str]) – Quoted message's identifier.

Return type *Result*[str]

Returns New message's identifier

send_contact_phone(*contact_name*, *phone_number*, *quoted_msg_id=None*)
Send contact to current chat using contact name and phone number.

Parameters

- **contact_name** (str) – Contact's name.
- **phone_number** (str) – Contact's phone number.
- **quoted_msg_id** (Optional[str]) – Quoted message's identifier.

Return type *Result*[str]

Returns New message's identifier

send_media(*media_data*, *content_type=None*, *filename=None*, *caption=None*, *quoted_msg_id=None*, *mentions=None*)
Send media file to current chat.

Parameters

- **media_data** (BytesIO) – io.ByteIO
- **content_type** (Optional[str]) – File content type. It is used by Whatsapp to known how to render it.
- **filename** (Optional[str]) – File name.
- **caption** (Optional[str]) – Media caption.
- **quoted_msg_id** (Optional[str]) – Quoted message's identifier.
- **mentions** (Optional[List[str]]) – List of user ids mentioned.

Return type *Result*[str]

Returns New message's identifier

leave_group()
Leave current chat group.

> **Warning:** Only available on group chats.

Return type *Result*[None]

delete_chat()
Delete chat.

Return type *Result*[None]

send_seen()
Mark chat as seen.

Return type *Result*[None]

load_earlier_messages()
Load earlier messages.

Tip: You should monitor chat messages in order to get new messages loaded.

```
async for msg in driver.chat[chat_id].msgs.monitor_add():
    print(msg)
```

Return type *Result*[None]

load_all_earlier_messages()
Load ALL earlier messages.

Tip: You should monitor chat messages in order to get new messages loaded. And perhaps you should remove them after process in order to save memory.

```
async for msg in driver.chat[chat_id].msgs.monitor_add():
    print(msg)
    await driver.chat[chat_id].msgs.remove_item_by_id(item.id)
```

Return type *Result*[None]

set_subject(*subject*)
Set group subject/title.

Parameters **subject** (*str*) – New group subject/title string

Return type *Result*[None]

Returns None

Model LiveLocation(*data=None*, *flat=False*, **args*, ***kwargs*)
Bases: *whalesong.models.BaseModel*

Live location model.

clear()
Clears all the data in the object, keeping original data

clear_all()
Clears all the data in the object

clear_modified_data()
Clears only the modified data

copy()
Creates a copy of model

delete_attr_by_path(*field_path*)
It deletes fields looked up by field path. Field path is dot-formatted string path: `parent_field.child_field`.

Parameters **field_path** (*str*) – field path. It allows * as wildcard.

delete_field_value(*name*)
Mark this field to be deleted

export_data()
Get the results with the modified_data

export_deleted_fields()
Resturns a list with any deleted fields form original data. In tree models, deleted fields on children will be appended.

export_modifications()
Returns model modifications.

export_modified_data()
Get the modified data

export_original_data()
Get the original data

flat_data()
Pass all the data from modified_data to original_data

get_1st_attr_by_path(*field_path*, ***kwargs*)
It returns first value looked up by field path. Field path is dot-formatted string path: `parent_field.child_field`.
Parameters
- **field_path** (*str*) – field path. It allows * as wildcard.
- **default** – Default value if field does not exist. If it is not defined `AttributeError` exception will be raised.

Returns value

get_attrs_by_path(*field_path*, *stop_first=False*)
It returns list of values looked up by field path. Field path is dot-formatted string path: `parent_field.child_field`.
Parameters
- **field_path** (*list or None.*) – field path. It allows * as wildcard.
- **stop_first** (*bool*) – Stop iteration on first value looked up. Default: False.

Returns A list of values or None it was a invalid path.
Return type `list` or `None`

get_default_data()
Returns a dictionary with default data.
Returns dict

get_field_obj(*name*)

get_field_value(*name*)
Get the field value from the modified data or the original one

get_fields()
Returns used fields of model

get_original_field_value(*name*)
Returns original field value or None

get_parent()
Returns parent model

get_read_only()
Returns whether model could be modified or not

get_real_name(*name*)

get_structure()
Returns a dictionary with model field objects.
Returns dict

import_data(*data*)
Set the fields established in data to the instance

import_deleted_fields(*data*)
Set data fields to deleted

is_locked()
Returns whether model is locked

is_modified()
Returns whether model is modified or not

is_modified_field(*name*)
Returns whether a field is modified or not

lock()
Lock model to avoid modification on read only fields

reset_attr_by_path(*field_path*)
It restores original values for fields looked up by field path. Field path is dot-formatted string path: `parent_field.child_field`.

Parameters **field_path** (*str*) – field path. It allows * as wildcard.

reset_field_value(*name*)
Resets value of a field

set_field_value(*name*, *value*)
Set the value to the field modified_data

set_parent(*value*)
Sets parent model

set_read_only(*value*)
Sets whether model could be modified or not

unlock()
Unlock model to be able to write even it's read only

property id : str *[READ ONLY]*
StringIdField field [READ ONLY]

property duration : timedelta
TimedeltaField field

property participants : List of *Participant*
Array of ModelField field (*whalesong.managers.live_location.Participant*)

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (*str*) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

mark_composing()
Set "typing…" message for 2.5 seconds.

Return type *Result*[None]

Returns None

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

mark_recording()
Set "recording audio…" message.

Return type *Result*[None]

Returns None

mark_paused()
Unset "typing…" or "recording audio…" message.

Return type *Result*[None]

Returns None

can_archive()
Check whether chat could be archived.

Return type *Result*[bool]

Returns Whether chat could be archived.

can_send()
Check whether current user could send message in chat.

Return type *Result*[bool]

Returns Whether current user could send message in chat.

can_pin()
Check whether chat could be pinned.

Return type *Result*[bool]

Returns Whether chat could be pinned.

archive()
Archive chat.

Return type *Result*[bool]

Returns Operation result.

unarchive()
Unarchive chat.

Return type *Result*[bool]

Returns Operation result.

pin()
Pin chat.

Return type *Result*[bool]

Returns Operation result.

unpin()
Unpin chat.

Return type *Result*[bool]

Returns Operation result.

set_group_description(*description*)
Set group description.

Return type *Result*[bool]

Returns Operation result.

star_messages(*message_ids*)
Star messages.

Return type *Result*[bool]

Returns Operation result.

unstar_messages(*message_ids*)
Unstar messages.

Return type *Result*[bool]

Returns Operation result.

send_not_spam()
Send not spam report.

Return type *Result*[bool]

Returns Operation result.

send_spam_report()
Send spam report.

Return type *Result*[bool]

Returns Operation result.

find_live_location()
It find chat's live location. If it does not exist it will be created.

Return type *Result*[*LiveLocation*]

Returns Live location.

class whalesong.managers.chat.**MsgLoadStateManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseModelManager*

Message load state manager

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

MODEL_CLASS
alias of *MsgLoadState*

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

3.4.7.2 Models

Model whalesong.managers.chat.**Chat**(*data=None*, *flat=False*, **args*, ***kwargs*)
Bases: *whalesong.models.BaseModel*

Chat model.

property name : str
Chat title.

property pin : int
Pin type (¿?)

property archive : bool

Default value False

Whether chat is archived or not.

property contact : *Contact*
Contact object.

property kind : str
¿?

property mute : *Mute*
Mute information.

property id : str *[READ ONLY]*
StringIdField field [READ ONLY]

property lastMessageTs : datetime
DateTimeField field

Aliases

- **last_message_ts**
- **t**

property unreadCount : int
IntegerField field

Aliases

- **unread_count**

property changeNumberNewJid : str
StringIdField field

Aliases

- **change_number_new_jid**

property changeNumberOldJid : str
StringIdField field

Aliases

- **change_number_old_jid**

property groupMetadata : *GroupMetadata*
ModelField field (*whalesong.managers.group_metadata.GroupMetadata*)

Aliases

- **group_metadata**

property isAnnounceGrpRestrict : bool

Default value False

BooleanField field

Aliases

- **is_announce_grp_restrict**

property isGroup : bool

Default value False

BooleanField field

Aliases

- **is_group**

property isReadOnly : bool

Default value False

BooleanField field

Aliases

- **is_read_only**

property lastReceivedKey : **str**
StringIdField field

Aliases

- **last_received_key**

property modifyTag : **str**
StringIdField field

Aliases

- **modify_tag**

property muteExpiration : **int**
IntegerField field

Aliases

- **mute_expiration**

property notSpam : **bool**

Default value False

BooleanField field

Aliases

- **not_spam**

Model whalesong.managers.chat.**MsgLoadState**(*data=None*, *flat=False*, **args*, ***kwargs*)
Bases: *whalesong.models.BaseModel*

property id : **str** *[READ ONLY]*
StringIdField field [READ ONLY]

property contextLoaded : **bool**

Default value False

BooleanField field

Aliases

- **context_loaded**

property isLoadingAroundMsgs : **bool**

Default value False

BooleanField field

Aliases

- **is_loading_around_msgs**

property isLoadingEarlierMsgs : **bool**

Default value False

BooleanField field

Aliases

- **is_loading_earlier_msgs**

property isLoadingRecentMsgs : bool

Default value False

BooleanField field

Aliases

- **is_loading_recent_msgs**

property noEarlierMsgs : bool

Default value False

BooleanField field

Aliases

- **no_earlier_msgs**

3.4.8 Message references

3.4.8.1 Managers

class whalesong.managers.message.**MessageCollectionManager**(*driver*, *manager_path=''*)

Bases: *whalesong.managers.BaseCollectionManager*

Message collection manager.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)

Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

__getattr__(*name*)

Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

add_submanager(*name*, *submanager*)

Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

find_item_by_id(*item_id*)

Find model by identifier. If item is not in collection it will try to load it.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_first()
Get first item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_by_id(*item_id*)
Get model by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_result_class()

Return type *Result*[+MODEL_TYPE]

get_items()
Get all items on collection.

Return type *IteratorResult*[+MODEL_TYPE]

Returns Async iterator

get_iterator_result_class()

Return type *IteratorResult*[+MODEL_TYPE]

get_last()
Get last item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_length()
Get collection items count.

Return type *Result*[int]

Returns Items count

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

monitor_add()
Monitor add item collection. Iterate each time a item is added to collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_change()
Monitor change item collection. Iterate each time a item change in collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_field(*field*)
Monitor item's field change. Iterate each time a field changed in any item of collection.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model object iterator

monitor_remove()
Monitor remove item collection. Iterate each time a item is removed from collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

remove_item_by_id(*item_id*)
Remove item by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[None]

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

MODEL_MANAGER_CLASS
alias of *MessageManager*

monitor_new()
Monitor new messages.

Return type *MonitorResult*[*BaseMessage*]

Returns New message monitor.

await download_media(*model*)
Download message's attached media file. It will decrypt media file using key on message object.

Parameters **model** (*MediaMixin*) – MediaMixin

Return type BytesIO

Returns Media stream.

class whalesong.managers.message.**MessageManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseModelManager*

Message object manager.

info
MessageInfoManager

Message information manager.

__getitem__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

MODEL_CLASS
alias of *BaseMessage*

await download_media()
Download message's attached media file. It will decrypt media file using key on message object.

Return type BytesIO

Returns Media stream.

fetch_info()
Fetch message information. It must fetch before try to use message information manager.

Return type *Result*[*MessageInfo*]

Returns Message information (*MessageInfo*)

can_star()
Check whether message could be starred.

Return type *Result*[bool]

Returns Whether message could be starred.

star()
Star message.

Return type *Result*[bool]

unstar()
Unstar message.

Return type *Result*[bool]

can_revoke()
Check whether message could be revoked (deleted for other).

Return type *Result*[bool]

Returns Whether message could be revoked.

revoke(*clear_media=True*)
Revoke message.

Return type *Result*[str]

class whalesong.managers.message.**MessageInfoManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseModelManager*

Message information object manager.

delivery
MessageAckCollectionManager

Message delivery acknowledgement collection manager.

read
MessageAckCollectionManager

Message read acknowledgement collection manager.

played
MessageAckCollectionManager

Message played acknowledgement collection manager.

__getitem__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

MODEL_CLASS
alias of *MessageInfo*

class whalesong.managers.message.**MessageAckCollectionManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseCollectionManager*

Message acknowledgement collection manager.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

__getattr__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

find_item_by_id(*item_id*)
Find model by identifier. If item is not in collection it will try to load it.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_first()
Get first item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_by_id(*item_id*)
Get model by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_result_class()

Return type *Result*[+MODEL_TYPE]

get_items()
Get all items on collection.

Return type *IteratorResult*[+MODEL_TYPE]

Returns Async iterator

get_iterator_result_class()

Return type *IteratorResult*[+MODEL_TYPE]

get_last()
Get last item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_length()
Get collection items count.

Return type *Result*[int]

Returns Items count

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

monitor_add()
Monitor add item collection. Iterate each time a item is added to collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_change()
Monitor change item collection. Iterate each time a item change in collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_field(*field*)
Monitor item's field change. Iterate each time a field changed in any item of collection.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model object iterator

monitor_remove()
Monitor remove item collection. Iterate each time a item is removed from collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

remove_item_by_id(*item_id*)
Remove item by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[None]

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

MODEL_MANAGER_CLASS
alias of *MessageAckManager*

class whalesong.managers.message.**MessageAckManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseModelManager*

Message acknowledgement object manager.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

MODEL_CLASS
alias of *MessageAck*

3.4.8.2 Models

class whalesong.managers.message.**MessageTypes**
Bases: enum.Enum

An enumeration.

NOTIFICATION_TEMPLATE = 'notification_template'

GROUP_NOTIFICATION = 'group_notification'

GP2 = 'gp2'

BROADCAST_NOTIFICATION = 'broadcast_notification'

E2E_NOTIFICATION = 'e2e_notification'

CALL_LOG = 'call_log'

PROTOCOL = 'protocol'

CIPHERTEXT = 'ciphertext'

REVOKED = 'revoked'

UNKNOWN = 'unknown'

CHAT = 'chat'

IMAGE = 'image'

VCARD = 'vcard'

MULTI_VCARD = 'multi_vcard'

LOCATION = 'location'

PAYMENT = 'payment'

DOCUMENT = 'document'

AUDIO = 'audio'

PTT = 'ptt'

VIDEO = 'video'

STICKER = 'sticker'

class whalesong.managers.message.**Ack**
Bases: enum.Enum

An enumeration.

ERROR = -1

PENDING = 0

SERVER = 1

DEVICE = 2

READ = 3

PLAYED = 4

Model whalesong.managers.message.**VCardItem**(*data=None*, *flat=False*, **args*, ***kwargs*)
Bases: *whalesong.models.BaseModel*

vCard item.

property vcard : str
Serialized vCard object.

property id : str *[READ ONLY]*
StringIdField field [READ ONLY]

property displayName : str
StringIdField field

Aliases

- **display_name**

class whalesong.managers.message.**MessageMetaclass**(*name*, *bases*, *classdict*)
Bases: dirty_models.models.CamelCaseMeta

Message metaclass. It will build message model according to type.

Model whalesong.managers.message.**BaseMessage**(*data=None*, **args*, ***kwargs*)
Bases: *whalesong.models.BaseModel*

Base message model.

property type : *MessageTypes* *[READ ONLY]*
Message type.

property subtype : str
Message subtype.

property body : str
Message content.

property timestamp : datetime
Message timestamp.

Aliases

- **t**

property to : str
¿?

property author : str
¿?

property sender : str
Sender's contact identifier.

property self : str

Default value in

¿?

property ack : *Ack*
Acknowledge state.

property invis : bool

Default value False

¿?

property star : bool

Default value False

Whether it is starred or not.

property links : List of str
List of links of message.

property chat : *Chat*
Chat object where message was sent.

property dir : str
¿?

property rtl : bool

Default value False

Whether message's content is a right to left text.

property id : str *[READ ONLY]*
StringIdField field [READ ONLY]

property notifyName : str
StringIdField field

Aliases

- **notify_name**

property from : str
StringIdField field

Aliases

- **from**

property senderObj : *Contact*
ModelField field (*whalesong.managers.contact.Contact*)

Aliases

- **sender_obj**

property isNewMsg : bool

Default value False

BooleanField field

Aliases

- **is_new_msg**

property isForwarded : bool

Default value False

BooleanField field

Aliases

- **is_forwarded**

property isGroupMsg : bool

Default value False

BooleanField field

Aliases

- **is_group_msg**

property isStatusV3 : bool

Default value False

BooleanField field

Aliases

- **is_status_v3**

property isPSA : bool

Default value False

BooleanField field

Aliases

- **is_psa**

property statusV3TextBg : str

StringIdField field

Aliases

- **status_v3_text_bg**

property isSentByMe : bool

Default value False

BooleanField field

Aliases

- **is_sent_by_me**

property isNotification : bool

Default value False

BooleanField field

Aliases

- **is_notification**

property isGroupNotification : bool

Default value False

BooleanField field

Aliases

- **is_group_notification**

property isBizNotification : bool

Default value False

BooleanField field

Aliases

- **is_biz_notification**

property isMedia : bool

Default value False

BooleanField field

Aliases

- **is_media**

property isLink : bool

Default value False

BooleanField field

Aliases

- **is_link**

property hasLink : bool

Default value False

BooleanField field

Aliases

- **has_link**

property isDoc : bool

Default value False

BooleanField field

Aliases

- **is_doc**

property isMms : bool

Default value False

BooleanField field

Aliases

- **is_mms**

property isRevoked : bool

Default value False

BooleanField field

Aliases

- **is_revoked**

property showForwarded : bool

Default value False

BooleanField field

Aliases

- **show_forwarded**

property containsEmoji : bool

Default value False

BooleanField field

Aliases

- **contains_emoji**

property isFailed : bool

Default value False

BooleanField field

Aliases

- **is_failed**

property isPersistent : bool

Default value False

BooleanField field

Aliases

- **is_persistent**

property isUserCreatedType : bool

Default value False

BooleanField field

Aliases

- **is_user_created_type**

property hasPromises : bool

Default value False

BooleanField field

Aliases

- **has_promises**

Model whalesong.managers.message.**QuotedMessageMixin**(*data=None*, *flat=False*, **args*, ***kwargs*)

Bases: *whalesong.models.BaseModel*

property id : str *[READ ONLY]*
StringIdField field [READ ONLY]

property quotedMsgObj : *BaseMessage*
ModelField field (*whalesong.managers.message.BaseMessage*)

Aliases

- **quoted_msg_obj**

property quotedStanzaID : str
StringIdField field

Aliases

- **quoted_stanza_id**

property quotedParticipant : str
StringIdField field

Aliases

- **quoted_participant**

property quotedRemoteJid : str
StringIdField field

Aliases

- **quoted_remote_jid**

Model whalesong.managers.message.**MentionsMixin**(*data=None*, *flat=False*, **args*, ***kwargs*)
Bases: *whalesong.models.BaseModel*

property id : str *[READ ONLY]*
StringIdField field [READ ONLY]

property mentionedJidList : List of str
Array of StringIdField field

Aliases

- **mentioned_jid_list**

Model whalesong.managers.message.**LinkContentMixin**(*data=None*, *flat=False*, **args*, ***kwargs*)
Bases: *whalesong.models.BaseModel*

property description : str
Page description.

property title : str
Page title.

property thumbnail : None
Page thumbnail.

property links : List of str
List of links. ¿?

property id : str *[READ ONLY]*
StringIdField field [READ ONLY]

property matchedText : str
StringIdField field

Aliases

- **matched_text**

property canonicalUrl : str
StringIdField field

Aliases

- **canonical_url**

property linkPreview : bool

Default value False

BooleanField field

Aliases

- **link_preview**

Model whalesong.managers.message.**MediaMixin**(*data=None*, *flat=False*, **args*, ***kwargs*)

Bases: *whalesong.models.BaseModel*

property type : ***MessageTypes*** *[READ ONLY]*
EnumField field [READ ONLY] (*whalesong.managers.message.MessageTypes*)

property mimetype : **str**
StringIdField field

property caption : **str**
StringIdField field

property filehash : **str**
StringIdField field

property size : **int**
IntegerField field

property id : **str** *[READ ONLY]*
StringIdField field [READ ONLY]

property clientUrl : **str**
StringIdField field

Aliases

- **client_url**

property directPath : **str**
StringIdField field

Aliases

- **direct_path**

property mediaKey : **str**
StringIdField field

Aliases

- **media_key**

property isUnsentMedia : **bool**

Default value False

BooleanField field

Aliases

- **is_unsent_media**

Model whalesong.managers.message.**MediaFrameMixin**(*data=None*, *flat=False*, **args*, ***kwargs*)

Bases: *whalesong.models.BaseModel*

property body : **None**
Base64Field field

property height : **int**
IntegerField field

property width : **int**
IntegerField field

property id : **str** *[READ ONLY]*
StringIdField field [READ ONLY]

Model whalesong.managers.message.**AuthorMixin**(*data=None*, *flat=False*, **args*, ***kwargs*)

Bases: *whalesong.models.BaseModel*

property author : **str**
StringIdField field

property id : **str** *[READ ONLY]*
StringIdField field [READ ONLY]

Model whalesong.managers.message.**MediaStreamMixin**(*data=None*, *flat=False*, **args*, ***kwargs*)

Bases: *whalesong.models.BaseModel*

property streamable : **bool**

Default value False

BooleanField field

property durantion : **int**
IntegerField field

property id : **str** *[READ ONLY]*
StringIdField field [READ ONLY]

property isGif : **bool**

Default value False

BooleanField field

Aliases

- **is_gif**

property gifAttribution : **bool**

Default value False

BooleanField field

Aliases

- **gif_attribution**

Model whalesong.managers.message.**TextMessage**(*data=None*, **args*, ***kwargs*)

Bases: *whalesong.managers.message.QuotedMessageMixin*, *whalesong.managers.message.LinkContentMixin*, *whalesong.managers.message.MentionsMixin*, *whalesong.managers.message.BaseMessage*

Text message.

property font : **str**
StringIdField field

property id : **str** *[READ ONLY]*
StringIdField field [READ ONLY]

property quotedMsgObj : ***BaseMessage***
ModelField field (*whalesong.managers.message.BaseMessage*)

Aliases

- **quoted_msg_obj**

property quotedStanzaID : **str**
StringIdField field

Aliases

- **quoted_stanza_id**

property quotedParticipant : str
StringIdField field

Aliases

- **quoted_participant**

property quotedRemoteJid : str
StringIdField field

Aliases

- **quoted_remote_jid**

property matchedText : str
StringIdField field

Aliases

- **matched_text**

property canonicalUrl : str
StringIdField field

Aliases

- **canonical_url**

property description : str
StringIdField field

property title : str
StringIdField field

property thumbnail : None
Base64Field field

property linkPreview : bool

Default value False

BooleanField field

Aliases

- **link_preview**

property links : List of str
Array of StringIdField field

property mentionedJidList : List of str
Array of StringIdField field

Aliases

- **mentioned_jid_list**

property type : *MessageTypes* *[READ ONLY]*
EnumField field [READ ONLY] (*whalesong.managers.message.MessageTypes*)

property subtype : str
StringIdField field

property body : str
StringIdField field

property timestamp : datetime
DateTimeField field

Aliases

- **t**

property notifyName : str
StringIdField field

Aliases

- **notify_name**

property from : str
StringIdField field

Aliases

- **from**

property to : str
StringIdField field

property author : str
StringIdField field

property sender : str
StringIdField field

property senderObj : *Contact*
ModelField field (*whalesong.managers.contact.Contact*)

Aliases

- **sender_obj**

property self : str

Default value in

StringIdField field

property ack : *Ack*
EnumField field (*whalesong.managers.message.Ack*)

property invis : bool

Default value False

BooleanField field

property isNewMsg : bool

Default value False

BooleanField field

Aliases

- **is_new_msg**

property star : bool

Default value False

BooleanField field

property isForwarded : bool

Default value False

BooleanField field

Aliases

- **is_forwarded**

property chat : *Chat*
ModelField field (*whalesong.managers.chat.Chat*)

property isGroupMsg : bool

Default value False

BooleanField field

Aliases

- **is_group_msg**

property isStatusV3 : bool

Default value False

BooleanField field

Aliases

- **is_status_v3**

property isPSA : bool

Default value False

BooleanField field

Aliases

- **is_psa**

property statusV3TextBg : str
StringIdField field

Aliases

- **status_v3_text_bg**

property isSentByMe : bool

Default value False

BooleanField field

Aliases

- **is_sent_by_me**

property isNotification : bool

Default value False

BooleanField field

Aliases

- **is_notification**

property isGroupNotification : bool

Default value False

BooleanField field

Aliases

- **is_group_notification**

property isBizNotification : bool

Default value False

BooleanField field

Aliases

- **is_biz_notification**

property isMedia : bool

Default value False

BooleanField field

Aliases

- **is_media**

property isLink : bool

Default value False

BooleanField field

Aliases

- **is_link**

property hasLink : bool

Default value False

BooleanField field

Aliases

- **has_link**

property isDoc : bool

Default value False

BooleanField field

Aliases

- **is_doc**

property isMms : bool

Default value False

BooleanField field

Aliases

- **is_mms**

property isRevoked : bool

Default value False

BooleanField field

Aliases

- **is_revoked**

property showForwarded : bool

Default value False

BooleanField field

Aliases

- **show_forwarded**

property containsEmoji : bool

Default value False

BooleanField field

Aliases

- **contains_emoji**

property isFailed : bool

Default value False

BooleanField field

Aliases

- **is_failed**

property dir : str

StringIdField field

property rtl : bool

Default value False

BooleanField field

property isPersistent : bool

Default value False

BooleanField field

Aliases

- **is_persistent**

property isUserCreatedType : bool

Default value False

BooleanField field

Aliases

- **is_user_created_type**

property hasPromises : bool

Default value False

BooleanField field

Aliases

- **has_promises**

property textColor : str
StringIdField field

Aliases

- **text_color**

property backgroundColor : str
StringIdField field

Aliases

- **background_color**

Model whalesong.managers.message.**ImageMessage**(*data=None*, **args*, ***kwargs*)
Bases: *whalesong.managers.message.QuotedMessageMixin*, *whalesong.managers.message.MentionsMixin*, *whalesong.managers.message.MediaMixin*, *whalesong.managers.message.MediaFrameMixin*, *whalesong.managers.message.BaseMessage*

Image message.

property id : str *[READ ONLY]*
StringIdField field [READ ONLY]

property quotedMsgObj : *BaseMessage*
ModelField field (*whalesong.managers.message.BaseMessage*)

Aliases

- **quoted_msg_obj**

property quotedStanzaID : str
StringIdField field

Aliases

- **quoted_stanza_id**

property quotedParticipant : str
StringIdField field

Aliases

- **quoted_participant**

property quotedRemoteJid : str
StringIdField field

Aliases

- **quoted_remote_jid**

property mentionedJidList : List of str
Array of StringIdField field

Aliases

- **mentioned_jid_list**

property type : *MessageTypes* *[READ ONLY]*
EnumField field [READ ONLY] (*whalesong.managers.message.MessageTypes*)

property clientUrl : str
StringIdField field

Aliases

- **client_url**

property directPath : str
StringIdField field

Aliases

- **direct_path**

property mimetype : str
StringIdField field

property caption : str
StringIdField field

property filehash : str
StringIdField field

property size : int
IntegerField field

property mediaKey : str
StringIdField field

Aliases

- **media_key**

property isUnsentMedia : bool

Default value False

BooleanField field

Aliases

- **is_unsent_media**

property body : None
Base64Field field

property height : int
IntegerField field

property width : int
IntegerField field

property subtype : str
StringIdField field

property timestamp : datetime
DateTimeField field

Aliases

- **t**

property notifyName : str
StringIdField field

Aliases

- **notify_name**

property from : *str*
StringIdField field

Aliases

- **from**

property to : *str*
StringIdField field

property author : *str*
StringIdField field

property sender : *str*
StringIdField field

property senderObj : *Contact*
ModelField field (*whalesong.managers.contact.Contact*)

Aliases

- **sender_obj**

property self : *str*

Default value in

StringIdField field

property ack : *Ack*
EnumField field (*whalesong.managers.message.Ack*)

property invis : *bool*

Default value False

BooleanField field

property isNewMsg : *bool*

Default value False

BooleanField field

Aliases

- **is_new_msg**

property star : *bool*

Default value False

BooleanField field

property isForwarded : *bool*

Default value False

BooleanField field

Aliases

- **is_forwarded**

property links : *List of str*
Array of StringIdField field

property chat : *Chat*
ModelField field (*whalesong.managers.chat.Chat*)

property isGroupMsg : bool

Default value False

BooleanField field

Aliases

- **is_group_msg**

property isStatusV3 : bool

Default value False

BooleanField field

Aliases

- **is_status_v3**

property isPSA : bool

Default value False

BooleanField field

Aliases

- **is_psa**

property statusV3TextBg : str
StringIdField field

Aliases

- **status_v3_text_bg**

property isSentByMe : bool

Default value False

BooleanField field

Aliases

- **is_sent_by_me**

property isNotification : bool

Default value False

BooleanField field

Aliases

- **is_notification**

property isGroupNotification : bool

Default value False

BooleanField field

Aliases

- **is_group_notification**

property isBizNotification : bool

Default value False

BooleanField field

Aliases

- **is_biz_notification**

property isMedia : bool

Default value False

BooleanField field

Aliases

- **is_media**

property isLink : bool

Default value False

BooleanField field

Aliases

- **is_link**

property hasLink : bool

Default value False

BooleanField field

Aliases

- **has_link**

property isDoc : bool

Default value False

BooleanField field

Aliases

- **is_doc**

property isMms : bool

Default value False

BooleanField field

Aliases

- **is_mms**

property isRevoked : bool

Default value False

BooleanField field

Aliases

- **is_revoked**

property showForwarded : bool

Default value False

BooleanField field

Aliases

- **show_forwarded**

property containsEmoji : **bool**

Default value False

BooleanField field

Aliases

- **contains_emoji**

property isFailed : **bool**

Default value False

BooleanField field

Aliases

- **is_failed**

property dir : **str**

StringIdField field

property rtl : **bool**

Default value False

BooleanField field

property isPersistent : **bool**

Default value False

BooleanField field

Aliases

- **is_persistent**

property isUserCreatedType : **bool**

Default value False

BooleanField field

Aliases

- **is_user_created_type**

property hasPromises : **bool**

Default value False

BooleanField field

Aliases

- **has_promises**

Model whalesong.managers.message.**VideoMessage**(*data=None*, **args*, ***kwargs*)

Bases: *whalesong.managers.message.QuotedMessageMixin*, *whalesong.managers.message.MentionsMixin*, *whalesong.managers.message.MediaMixin*, *whalesong.managers.message.MediaFrameMixin*, *whalesong.managers.message.MediaStreamMixin*, *whalesong.managers.message.BaseMessage*

Video message.

property id : str *[READ ONLY]*
StringIdField field [READ ONLY]

property quotedMsgObj : BaseMessage
ModelField field (*whalesong.managers.message.BaseMessage*)

Aliases

- **quoted_msg_obj**

property quotedStanzaID : str
StringIdField field

Aliases

- **quoted_stanza_id**

property quotedParticipant : str
StringIdField field

Aliases

- **quoted_participant**

property quotedRemoteJid : str
StringIdField field

Aliases

- **quoted_remote_jid**

property mentionedJidList : List of str
Array of StringIdField field

Aliases

- **mentioned_jid_list**

property type : MessageTypes *[READ ONLY]*
EnumField field [READ ONLY] (*whalesong.managers.message.MessageTypes*)

property clientUrl : str
StringIdField field

Aliases

- **client_url**

property directPath : str
StringIdField field

Aliases

- **direct_path**

property mimetype : str
StringIdField field

property caption : str
StringIdField field

property filehash : str
StringIdField field

property size : int
IntegerField field

property mediaKey : str
StringIdField field

Aliases

- **media_key**

property isUnsentMedia : bool

Default value False

BooleanField field

Aliases

- **is_unsent_media**

property body : None
Base64Field field

property height : int
IntegerField field

property width : int
IntegerField field

property streamable : bool

Default value False

BooleanField field

property durantion : int
IntegerField field

property isGif : bool

Default value False

BooleanField field

Aliases

- **is_gif**

property gifAttribution : bool

Default value False

BooleanField field

Aliases

- **gif_attribution**

property subtype : str
StringIdField field

property timestamp : datetime
DateTimeField field

Aliases

- **t**

property notifyName : str
StringIdField field

Aliases

- **notify_name**

property from : str
StringIdField field

Aliases

- **from**

property to : str
StringIdField field

property author : str
StringIdField field

property sender : str
StringIdField field

property senderObj : *Contact*
ModelField field (*whalesong.managers.contact.Contact*)

Aliases

- **sender_obj**

property self : str

Default value in

StringIdField field

property ack : *Ack*
EnumField field (*whalesong.managers.message.Ack*)

property invis : bool

Default value False

BooleanField field

property isNewMsg : bool

Default value False

BooleanField field

Aliases

- **is_new_msg**

property star : bool

Default value False

BooleanField field

property isForwarded : bool

Default value False

BooleanField field

Aliases

- **is_forwarded**

property links : List of str
Array of StringIdField field

property chat : *Chat*
ModelField field (*whalesong.managers.chat.Chat*)

property isGroupMsg : bool

Default value False

BooleanField field

Aliases

- **is_group_msg**

property isStatusV3 : bool

Default value False

BooleanField field

Aliases

- **is_status_v3**

property isPSA : bool

Default value False

BooleanField field

Aliases

- **is_psa**

property statusV3TextBg : str
StringIdField field

Aliases

- **status_v3_text_bg**

property isSentByMe : bool

Default value False

BooleanField field

Aliases

- **is_sent_by_me**

property isNotification : bool

Default value False

BooleanField field

Aliases

- **is_notification**

property isGroupNotification : bool

Default value False

BooleanField field

Aliases

- **is_group_notification**

property isBizNotification : bool

Default value False

BooleanField field

Aliases

- **is_biz_notification**

property isMedia : bool

Default value False

BooleanField field

Aliases

- **is_media**

property isLink : bool

Default value False

BooleanField field

Aliases

- **is_link**

property hasLink : bool

Default value False

BooleanField field

Aliases

- **has_link**

property isDoc : bool

Default value False

BooleanField field

Aliases

- **is_doc**

property isMms : bool

Default value False

BooleanField field

Aliases

- **is_mms**

property isRevoked : bool

Default value False

BooleanField field

Aliases

- **is_revoked**

property showForwarded : bool

Default value False

BooleanField field

Aliases

- **show_forwarded**

property containsEmoji : bool

Default value False

BooleanField field

Aliases

- **contains_emoji**

property isFailed : bool

Default value False

BooleanField field

Aliases

- **is_failed**

property dir : str

StringIdField field

property rtl : bool

Default value False

BooleanField field

property isPersistent : bool

Default value False

BooleanField field

Aliases

- **is_persistent**

property isUserCreatedType : bool

Default value False

BooleanField field

Aliases

- **is_user_created_type**

property hasPromises : bool

Default value False

BooleanField field

Aliases

- **has_promises**

Model whalesong.managers.message.**AudioMessage**(*data=None*, **args*, ***kwargs*)

Bases: *whalesong.managers.message.QuotedMessageMixin*, *whalesong.managers.message.MentionsMixin*, *whalesong.managers.message.MediaMixin*, *whalesong.managers.message.MediaStreamMixin*, *whalesong.managers.message.BaseMessage*

Audio message.

property id : **str** *[READ ONLY]*
StringIdField field [READ ONLY]

property quotedMsgObj : ***BaseMessage***
ModelField field (*whalesong.managers.message.BaseMessage*)

Aliases

- **quoted_msg_obj**

property quotedStanzaID : **str**
StringIdField field

Aliases

- **quoted_stanza_id**

property quotedParticipant : **str**
StringIdField field

Aliases

- **quoted_participant**

property quotedRemoteJid : **str**
StringIdField field

Aliases

- **quoted_remote_jid**

property mentionedJidList : **List of str**
Array of StringIdField field

Aliases

- **mentioned_jid_list**

property type : ***MessageTypes*** *[READ ONLY]*
EnumField field [READ ONLY] (*whalesong.managers.message.MessageTypes*)

property clientUrl : **str**
StringIdField field

Aliases

- **client_url**

property directPath : **str**
StringIdField field

Aliases

- **direct_path**

property mimetype : **str**
StringIdField field

property caption : **str**
StringIdField field

property filehash : str
StringIdField field

property size : int
IntegerField field

property mediaKey : str
StringIdField field

Aliases

- **media_key**

property isUnsentMedia : bool

Default value False

BooleanField field

Aliases

- **is_unsent_media**

property streamable : bool

Default value False

BooleanField field

property durantion : int
IntegerField field

property isGif : bool

Default value False

BooleanField field

Aliases

- **is_gif**

property gifAttribution : bool

Default value False

BooleanField field

Aliases

- **gif_attribution**

property subtype : str
StringIdField field

property body : str
StringIdField field

property timestamp : datetime
DateTimeField field

Aliases

- **t**

property notifyName : str
StringIdField field

Aliases

- **notify_name**

property from : *str*
StringIdField field

Aliases

- **from**

property to : *str*
StringIdField field

property author : *str*
StringIdField field

property sender : *str*
StringIdField field

property senderObj : *Contact*
ModelField field (*whalesong.managers.contact.Contact*)

Aliases

- **sender_obj**

property self : *str*

Default value in

StringIdField field

property ack : *Ack*
EnumField field (*whalesong.managers.message.Ack*)

property invis : *bool*

Default value False

BooleanField field

property isNewMsg : *bool*

Default value False

BooleanField field

Aliases

- **is_new_msg**

property star : *bool*

Default value False

BooleanField field

property isForwarded : *bool*

Default value False

BooleanField field

Aliases

- **is_forwarded**

property links : *List of str*
Array of StringIdField field

property chat : *Chat*
ModelField field (*whalesong.managers.chat.Chat*)

property isGroupMsg : bool

Default value False

BooleanField field

Aliases

- **is_group_msg**

property isStatusV3 : bool

Default value False

BooleanField field

Aliases

- **is_status_v3**

property isPSA : bool

Default value False

BooleanField field

Aliases

- **is_psa**

property statusV3TextBg : str
StringIdField field

Aliases

- **status_v3_text_bg**

property isSentByMe : bool

Default value False

BooleanField field

Aliases

- **is_sent_by_me**

property isNotification : bool

Default value False

BooleanField field

Aliases

- **is_notification**

property isGroupNotification : bool

Default value False

BooleanField field

Aliases

- **is_group_notification**

property isBizNotification : bool

Default value False

BooleanField field

Aliases

- **is_biz_notification**

property isMedia : bool

Default value False

BooleanField field

Aliases

- **is_media**

property isLink : bool

Default value False

BooleanField field

Aliases

- **is_link**

property hasLink : bool

Default value False

BooleanField field

Aliases

- **has_link**

property isDoc : bool

Default value False

BooleanField field

Aliases

- **is_doc**

property isMms : bool

Default value False

BooleanField field

Aliases

- **is_mms**

property isRevoked : bool

Default value False

BooleanField field

Aliases

- **is_revoked**

property showForwarded : bool

Default value False

BooleanField field

Aliases

- **show_forwarded**

property containsEmoji : bool

Default value False

BooleanField field

Aliases

- **contains_emoji**

property isFailed : bool

Default value False

BooleanField field

Aliases

- **is_failed**

property dir : str

StringIdField field

property rtl : bool

Default value False

BooleanField field

property isPersistent : bool

Default value False

BooleanField field

Aliases

- **is_persistent**

property isUserCreatedType : bool

Default value False

BooleanField field

Aliases

- **is_user_created_type**

property hasPromises : bool

Default value False

BooleanField field

Aliases

- **has_promises**

Model whalesong.managers.message.**PTTMessage**(*data=None*, **args*, ***kwargs*)

Bases: *whalesong.managers.message.AudioMessage*

Push to talk message.

property id : **str** *[READ ONLY]*
StringIdField field [READ ONLY]

property quotedMsgObj : ***BaseMessage***
ModelField field (*whalesong.managers.message.BaseMessage*)

Aliases

- **quoted_msg_obj**

property quotedStanzaID : **str**
StringIdField field

Aliases

- **quoted_stanza_id**

property quotedParticipant : **str**
StringIdField field

Aliases

- **quoted_participant**

property quotedRemoteJid : **str**
StringIdField field

Aliases

- **quoted_remote_jid**

property mentionedJidList : **List of str**
Array of StringIdField field

Aliases

- **mentioned_jid_list**

property type : ***MessageTypes*** *[READ ONLY]*
EnumField field [READ ONLY] (*whalesong.managers.message.MessageTypes*)

property clientUrl : **str**
StringIdField field

Aliases

- **client_url**

property directPath : **str**
StringIdField field

Aliases

- **direct_path**

property mimetype : **str**
StringIdField field

property caption : **str**
StringIdField field

property filehash : **str**
StringIdField field

property size : **int**
IntegerField field

property mediaKey : str
StringIdField field

Aliases

- **media_key**

property isUnsentMedia : bool

Default value False

BooleanField field

Aliases

- **is_unsent_media**

property streamable : bool

Default value False

BooleanField field

property durantion : int
IntegerField field

property isGif : bool

Default value False

BooleanField field

Aliases

- **is_gif**

property gifAttribution : bool

Default value False

BooleanField field

Aliases

- **gif_attribution**

property subtype : str
StringIdField field

property body : str
StringIdField field

property timestamp : datetime
DateTimeField field

Aliases

- **t**

property notifyName : str
StringIdField field

Aliases

- **notify_name**

property from : str
StringIdField field

Aliases

- **from**

property to : str
StringIdField field

property author : str
StringIdField field

property sender : str
StringIdField field

property senderObj : *Contact*
ModelField field (*whalesong.managers.contact.Contact*)

Aliases

- **sender_obj**

property self : str

Default value in

StringIdField field

property ack : *Ack*
EnumField field (*whalesong.managers.message.Ack*)

property invis : bool

Default value False

BooleanField field

property isNewMsg : bool

Default value False

BooleanField field

Aliases

- **is_new_msg**

property star : bool

Default value False

BooleanField field

property isForwarded : bool

Default value False

BooleanField field

Aliases

- **is_forwarded**

property links : List of str
Array of StringIdField field

property chat : *Chat*
ModelField field (*whalesong.managers.chat.Chat*)

property isGroupMsg : bool

Default value False

BooleanField field

Aliases

- **is_group_msg**

property isStatusV3 : bool

Default value False

BooleanField field

Aliases

- **is_status_v3**

property isPSA : bool

Default value False

BooleanField field

Aliases

- **is_psa**

property statusV3TextBg : str

StringIdField field

Aliases

- **status_v3_text_bg**

property isSentByMe : bool

Default value False

BooleanField field

Aliases

- **is_sent_by_me**

property isNotification : bool

Default value False

BooleanField field

Aliases

- **is_notification**

property isGroupNotification : bool

Default value False

BooleanField field

Aliases

- **is_group_notification**

property isBizNotification : bool

Default value False

BooleanField field

Aliases

- **is_biz_notification**

property isMedia : bool

Default value False

BooleanField field

Aliases

- **is_media**

property isLink : bool

Default value False

BooleanField field

Aliases

- **is_link**

property hasLink : bool

Default value False

BooleanField field

Aliases

- **has_link**

property isDoc : bool

Default value False

BooleanField field

Aliases

- **is_doc**

property isMms : bool

Default value False

BooleanField field

Aliases

- **is_mms**

property isRevoked : bool

Default value False

BooleanField field

Aliases

- **is_revoked**

property showForwarded : bool

Default value False

BooleanField field

Aliases

- **show_forwarded**

property containsEmoji : bool

Default value False

BooleanField field

Aliases

- **contains_emoji**

property isFailed : bool

Default value False

BooleanField field

Aliases

- **is_failed**

property dir : str

StringIdField field

property rtl : bool

Default value False

BooleanField field

property isPersistent : bool

Default value False

BooleanField field

Aliases

- **is_persistent**

property isUserCreatedType : bool

Default value False

BooleanField field

Aliases

- **is_user_created_type**

property hasPromises : bool

Default value False

BooleanField field

Aliases

- **has_promises**

Model whalesong.managers.message.**DocumentMessage**(*data=None*, **args*, ***kwargs*)

Bases: *whalesong.managers.message.QuotedMessageMixin*, *whalesong.managers.message.MentionsMixin*, *whalesong.managers.message.MediaMixin*, *whalesong.managers.message.BaseMessage*

Document message.

property body : None

Base64Field field

property id : **str** *[READ ONLY]*
StringIdField field [READ ONLY]

property quotedMsgObj : ***BaseMessage***
ModelField field (*whalesong.managers.message.BaseMessage*)

Aliases

- **quoted_msg_obj**

property quotedStanzaID : **str**
StringIdField field

Aliases

- **quoted_stanza_id**

property quotedParticipant : **str**
StringIdField field

Aliases

- **quoted_participant**

property quotedRemoteJid : **str**
StringIdField field

Aliases

- **quoted_remote_jid**

property mentionedJidList : **List of str**
Array of StringIdField field

Aliases

- **mentioned_jid_list**

property type : ***MessageTypes*** *[READ ONLY]*
EnumField field [READ ONLY] (*whalesong.managers.message.MessageTypes*)

property clientUrl : **str**
StringIdField field

Aliases

- **client_url**

property directPath : **str**
StringIdField field

Aliases

- **direct_path**

property mimetype : **str**
StringIdField field

property caption : **str**
StringIdField field

property filehash : **str**
StringIdField field

property size : **int**
IntegerField field

property mediaKey : str
StringIdField field

Aliases

- **media_key**

property isUnsentMedia : bool

Default value False

BooleanField field

Aliases

- **is_unsent_media**

property subtype : str
StringIdField field

property timestamp : datetime
DateTimeField field

Aliases

- **t**

property notifyName : str
StringIdField field

Aliases

- **notify_name**

property from : str
StringIdField field

Aliases

- **from**

property to : str
StringIdField field

property author : str
StringIdField field

property sender : str
StringIdField field

property senderObj : *Contact*
ModelField field (*whalesong.managers.contact.Contact*)

Aliases

- **sender_obj**

property self : str

Default value in

StringIdField field

property ack : *Ack*
EnumField field (*whalesong.managers.message.Ack*)

property invis : bool

Default value False

BooleanField field

property isNewMsg : bool

Default value False

BooleanField field

Aliases

- **is_new_msg**

property star : bool

Default value False

BooleanField field

property isForwarded : bool

Default value False

BooleanField field

Aliases

- **is_forwarded**

property links : List of str

Array of StringIdField field

property chat : *Chat*

ModelField field (*whalesong.managers.chat.Chat*)

property isGroupMsg : bool

Default value False

BooleanField field

Aliases

- **is_group_msg**

property isStatusV3 : bool

Default value False

BooleanField field

Aliases

- **is_status_v3**

property isPSA : bool

Default value False

BooleanField field

Aliases

- **is_psa**

property statusV3TextBg : str

StringIdField field

Aliases

- **status_v3_text_bg**

property isSentByMe : bool

Default value False

BooleanField field

Aliases

- **is_sent_by_me**

property isNotification : bool

Default value False

BooleanField field

Aliases

- **is_notification**

property isGroupNotification : bool

Default value False

BooleanField field

Aliases

- **is_group_notification**

property isBizNotification : bool

Default value False

BooleanField field

Aliases

- **is_biz_notification**

property isMedia : bool

Default value False

BooleanField field

Aliases

- **is_media**

property isLink : bool

Default value False

BooleanField field

Aliases

- **is_link**

property hasLink : bool

Default value False

BooleanField field

Aliases

- **has_link**

property isDoc : bool

Default value False

BooleanField field

Aliases

- **is_doc**

property isMms : bool

Default value False

BooleanField field

Aliases

- **is_mms**

property isRevoked : bool

Default value False

BooleanField field

Aliases

- **is_revoked**

property showForwarded : bool

Default value False

BooleanField field

Aliases

- **show_forwarded**

property containsEmoji : bool

Default value False

BooleanField field

Aliases

- **contains_emoji**

property isFailed : bool

Default value False

BooleanField field

Aliases

- **is_failed**

property dir : str

StringIdField field

property rtl : bool

Default value False

BooleanField field

property isPersistent : bool

Default value False

BooleanField field

Aliases

- **is_persistent**

property isUserCreatedType : bool

Default value False

BooleanField field

Aliases

- **is_user_created_type**

property hasPromises : bool

Default value False

BooleanField field

Aliases

- **has_promises**

property pageCount : int

IntegerField field

Aliases

- **page_count**

Model whalesong.managers.message.**VCardMessage**(*data=None*, **args*, ***kwargs*)

Bases: *whalesong.managers.message.QuotedMessageMixin*, *whalesong.managers.message.MentionsMixin*, *whalesong.managers.message.BaseMessage*

vCard message.

property id : str *[READ ONLY]*

StringIdField field [READ ONLY]

property quotedMsgObj : *BaseMessage*

ModelField field (*whalesong.managers.message.BaseMessage*)

Aliases

- **quoted_msg_obj**

property quotedStanzaID : str

StringIdField field

Aliases

- **quoted_stanza_id**

property quotedParticipant : str

StringIdField field

Aliases

- **quoted_participant**

property quotedRemoteJid : str

StringIdField field

Aliases

- **quoted_remote_jid**

property mentionedJidList : List of str
Array of StringIdField field

Aliases

- **mentioned_jid_list**

property type : *MessageTypes* [READ ONLY]
EnumField field [READ ONLY] (*whalesong.managers.message.MessageTypes*)

property subtype : str
StringIdField field

property body : str
StringIdField field

property timestamp : datetime
DateTimeField field

Aliases

- **t**

property notifyName : str
StringIdField field

Aliases

- **notify_name**

property from : str
StringIdField field

Aliases

- **from**

property to : str
StringIdField field

property author : str
StringIdField field

property sender : str
StringIdField field

property senderObj : *Contact*
ModelField field (*whalesong.managers.contact.Contact*)

Aliases

- **sender_obj**

property self : str

Default value in

StringIdField field

property ack : *Ack*
EnumField field (*whalesong.managers.message.Ack*)

property invis : bool

Default value False

BooleanField field

property isNewMsg : bool

Default value False

BooleanField field

Aliases

- **is_new_msg**

property star : bool

Default value False

BooleanField field

property isForwarded : bool

Default value False

BooleanField field

Aliases

- **is_forwarded**

property links : List of str

Array of StringIdField field

property chat : *Chat*

ModelField field (*whalesong.managers.chat.Chat*)

property isGroupMsg : bool

Default value False

BooleanField field

Aliases

- **is_group_msg**

property isStatusV3 : bool

Default value False

BooleanField field

Aliases

- **is_status_v3**

property isPSA : bool

Default value False

BooleanField field

Aliases

- **is_psa**

property statusV3TextBg : str

StringIdField field

Aliases

- **status_v3_text_bg**

property isSentByMe : bool

Default value False

BooleanField field

Aliases

- **is_sent_by_me**

property isNotification : bool

Default value False

BooleanField field

Aliases

- **is_notification**

property isGroupNotification : bool

Default value False

BooleanField field

Aliases

- **is_group_notification**

property isBizNotification : bool

Default value False

BooleanField field

Aliases

- **is_biz_notification**

property isMedia : bool

Default value False

BooleanField field

Aliases

- **is_media**

property isLink : bool

Default value False

BooleanField field

Aliases

- **is_link**

property hasLink : bool

Default value False

BooleanField field

Aliases

- **has_link**

property isDoc : bool

Default value False

BooleanField field

Aliases

- **is_doc**

property isMms : bool

Default value False

BooleanField field

Aliases

- **is_mms**

property isRevoked : bool

Default value False

BooleanField field

Aliases

- **is_revoked**

property showForwarded : bool

Default value False

BooleanField field

Aliases

- **show_forwarded**

property containsEmoji : bool

Default value False

BooleanField field

Aliases

- **contains_emoji**

property isFailed : bool

Default value False

BooleanField field

Aliases

- **is_failed**

property dir : str

StringIdField field

property rtl : bool

Default value False

BooleanField field

property isPersistent : bool

Default value False

BooleanField field

Aliases

- **is_persistent**

property isUserCreatedType : bool

Default value False

BooleanField field

Aliases

- **is_user_created_type**

property hasPromises : bool

Default value False

BooleanField field

Aliases

- **has_promises**

Model whalesong.managers.message.**MultiVCardMessage**(*data=None*, **args*, ***kwargs*)

Bases: *whalesong.managers.message.QuotedMessageMixin*, *whalesong.managers.message.MentionsMixin*, *whalesong.managers.message.BaseMessage*

Multi vCard message.

property id : str *[READ ONLY]*

StringIdField field [READ ONLY]

property quotedMsgObj : *BaseMessage*

ModelField field (*whalesong.managers.message.BaseMessage*)

Aliases

- **quoted_msg_obj**

property quotedStanzaID : str

StringIdField field

Aliases

- **quoted_stanza_id**

property quotedParticipant : str

StringIdField field

Aliases

- **quoted_participant**

property quotedRemoteJid : str

StringIdField field

Aliases

- **quoted_remote_jid**

property mentionedJidList : List of str

Array of StringIdField field

Aliases

- **mentioned_jid_list**

property type : *MessageTypes* *[READ ONLY]*
EnumField field [READ ONLY] (*whalesong.managers.message.MessageTypes*)

property subtype : str
StringIdField field

property body : str
StringIdField field

property timestamp : datetime
DateTimeField field

Aliases

- **t**

property notifyName : str
StringIdField field

Aliases

- **notify_name**

property from : str
StringIdField field

Aliases

- **from**

property to : str
StringIdField field

property author : str
StringIdField field

property sender : str
StringIdField field

property senderObj : *Contact*
ModelField field (*whalesong.managers.contact.Contact*)

Aliases

- **sender_obj**

property self : str

Default value in

StringIdField field

property ack : *Ack*
EnumField field (*whalesong.managers.message.Ack*)

property invis : bool

Default value False

BooleanField field

property isNewMsg : bool

Default value False

BooleanField field

Aliases

- **is_new_msg**

property star : bool

Default value False

BooleanField field

property isForwarded : bool

Default value False

BooleanField field

Aliases

- **is_forwarded**

property links : List of str

Array of StringIdField field

property chat : *Chat*

ModelField field (*whalesong.managers.chat.Chat*)

property isGroupMsg : bool

Default value False

BooleanField field

Aliases

- **is_group_msg**

property isStatusV3 : bool

Default value False

BooleanField field

Aliases

- **is_status_v3**

property isPSA : bool

Default value False

BooleanField field

Aliases

- **is_psa**

property statusV3TextBg : str

StringIdField field

Aliases

- **status_v3_text_bg**

property isSentByMe : bool

Default value False

BooleanField field

Aliases

- **is_sent_by_me**

property isNotification : bool

Default value False

BooleanField field

Aliases

- **is_notification**

property isGroupNotification : bool

Default value False

BooleanField field

Aliases

- **is_group_notification**

property isBizNotification : bool

Default value False

BooleanField field

Aliases

- **is_biz_notification**

property isMedia : bool

Default value False

BooleanField field

Aliases

- **is_media**

property isLink : bool

Default value False

BooleanField field

Aliases

- **is_link**

property hasLink : bool

Default value False

BooleanField field

Aliases

- **has_link**

property isDoc : bool

Default value False

BooleanField field

Aliases

- **is_doc**

property isMms : bool

Default value False

BooleanField field

Aliases

- **is_mms**

property isRevoked : bool

Default value False

BooleanField field

Aliases

- **is_revoked**

property showForwarded : bool

Default value False

BooleanField field

Aliases

- **show_forwarded**

property containsEmoji : bool

Default value False

BooleanField field

Aliases

- **contains_emoji**

property isFailed : bool

Default value False

BooleanField field

Aliases

- **is_failed**

property dir : str

StringIdField field

property rtl : bool

Default value False

BooleanField field

property isPersistent : bool

Default value False

BooleanField field

Aliases

- **is_persistent**

property isUserCreatedType : bool

Default value False

BooleanField field

Aliases

- **is_user_created_type**

property hasPromises : `bool`

Default value False

BooleanField field

Aliases

- **has_promises**

property vcardList : `List of` *VCardItem*

Array of ModelField field (*whalesong.managers.message.VCardItem*)

Aliases

- **vcard_list**

Model whalesong.managers.message.**LocationMessage**(*data=None*, **args*, ***kwargs*)

Bases: *whalesong.managers.message.QuotedMessageMixin*, *whalesong.managers.message.MentionsMixin*, *whalesong.managers.message.BaseMessage*

Location message.

property body : `None`

Base64Field field

property lat : `float`

FloatField field

property lng : `float`

FloatField field

property loc : `str`

StringIdField field

property accuracy : `int`

IntegerField field

property speed : `int`

IntegerField field

property degrees : `float`

FloatField field

property comment : `str`

StringIdField field

property sequence : `int`

IntegerField field

property id : `str` *[READ ONLY]*

StringIdField field [READ ONLY]

property quotedMsgObj : *BaseMessage*

ModelField field (*whalesong.managers.message.BaseMessage*)

Aliases

- **quoted_msg_obj**

property quotedStanzaID : str
StringIdField field

Aliases

- **quoted_stanza_id**

property quotedParticipant : str
StringIdField field

Aliases

- **quoted_participant**

property quotedRemoteJid : str
StringIdField field

Aliases

- **quoted_remote_jid**

property mentionedJidList : List of str
Array of StringIdField field

Aliases

- **mentioned_jid_list**

property type : *MessageTypes* *[READ ONLY]*
EnumField field [READ ONLY] (*whalesong.managers.message.MessageTypes*)

property subtype : str
StringIdField field

property timestamp : datetime
DateTimeField field

Aliases

- **t**

property notifyName : str
StringIdField field

Aliases

- **notify_name**

property from : str
StringIdField field

Aliases

- **from**

property to : str
StringIdField field

property author : str
StringIdField field

property sender : str
StringIdField field

property senderObj : *Contact*
ModelField field (*whalesong.managers.contact.Contact*)

Aliases

- **sender_obj**

property self : str

Default value in

StringIdField field

property ack : *Ack*
EnumField field (*whalesong.managers.message.Ack*)

property invis : bool

Default value False

BooleanField field

property isNewMsg : bool

Default value False

BooleanField field

Aliases

- **is_new_msg**

property star : bool

Default value False

BooleanField field

property isForwarded : bool

Default value False

BooleanField field

Aliases

- **is_forwarded**

property links : List of str
Array of StringIdField field

property chat : *Chat*
ModelField field (*whalesong.managers.chat.Chat*)

property isGroupMsg : bool

Default value False

BooleanField field

Aliases

- **is_group_msg**

property isStatusV3 : bool

Default value False

BooleanField field

Aliases

- **is_status_v3**

property isPSA : bool

Default value False

BooleanField field

Aliases

- **is_psa**

property statusV3TextBg : str

StringIdField field

Aliases

- **status_v3_text_bg**

property isSentByMe : bool

Default value False

BooleanField field

Aliases

- **is_sent_by_me**

property isNotification : bool

Default value False

BooleanField field

Aliases

- **is_notification**

property isGroupNotification : bool

Default value False

BooleanField field

Aliases

- **is_group_notification**

property isBizNotification : bool

Default value False

BooleanField field

Aliases

- **is_biz_notification**

property isMedia : bool

Default value False

BooleanField field

Aliases

- **is_media**

property isLink : bool

Default value False

BooleanField field

Aliases

- **is_link**

property hasLink : bool

Default value False

BooleanField field

Aliases

- **has_link**

property isDoc : bool

Default value False

BooleanField field

Aliases

- **is_doc**

property isMms : bool

Default value False

BooleanField field

Aliases

- **is_mms**

property isRevoked : bool

Default value False

BooleanField field

Aliases

- **is_revoked**

property showForwarded : bool

Default value False

BooleanField field

Aliases

- **show_forwarded**

property containsEmoji : bool

Default value False

BooleanField field

Aliases

- **contains_emoji**

property isFailed : bool

Default value False

BooleanField field

Aliases

- **is_failed**

property dir : str
StringIdField field

property rtl : bool

Default value False

BooleanField field

property isPersistent : bool

Default value False

BooleanField field

Aliases

- **is_persistent**

property isUserCreatedType : bool

Default value False

BooleanField field

Aliases

- **is_user_created_type**

property hasPromises : bool

Default value False

BooleanField field

Aliases

- **has_promises**

property isLive : bool

Default value False

BooleanField field

Aliases

- **is_live**

property shareDuration : timedelta
TimedeltaField field

Aliases

- **share_duration**

property finalThumbnail : None
Base64Field field

Aliases

- **final_thumbnail**

property finalLat : float
FloatField field

Aliases

- **final_lat**

property finalLng : *float*
FloatField field

Aliases

- **final_lng**

property finalAccuracy : *int*
IntegerField field

Aliases

- **final_accuracy**

property finalSpeed : *int*
IntegerField field

Aliases

- **final_speed**

property finalDegrees : *float*
FloatField field

Aliases

- **final_degrees**

property finalTimeOffset : *timedelta*
TimedeltaField field

Aliases

- **final_time_offset**

property duration : *timedelta*
TimedeltaField field

property text : *str*
StringIdField field

Model whalesong.managers.message.**PaymentMessage**(*data=None*, **args*, ***kwargs*)
Bases: *whalesong.managers.message.BaseMessage*

Payment message.

property id : *str* *[READ ONLY]*
StringIdField field [READ ONLY]

property type : *MessageTypes* *[READ ONLY]*
EnumField field [READ ONLY] (*whalesong.managers.message.MessageTypes*)

property subtype : *str*
StringIdField field

property body : *str*
StringIdField field

property timestamp : *datetime*
DateTimeField field

Aliases

- **t**

property notifyName : str
StringIdField field

Aliases

- **notify_name**

property from : str
StringIdField field

Aliases

- **from**

property to : str
StringIdField field

property author : str
StringIdField field

property sender : str
StringIdField field

property senderObj : *Contact*
ModelField field (*whalesong.managers.contact.Contact*)

Aliases

- **sender_obj**

property self : str

Default value in

StringIdField field

property ack : *Ack*
EnumField field (*whalesong.managers.message.Ack*)

property invis : bool

Default value False

BooleanField field

property isNewMsg : bool

Default value False

BooleanField field

Aliases

- **is_new_msg**

property star : bool

Default value False

BooleanField field

property isForwarded : bool

Default value False

BooleanField field

Aliases

- **is_forwarded**

property links : List of str
Array of StringIdField field

property chat : *Chat*
ModelField field (*whalesong.managers.chat.Chat*)

property isGroupMsg : bool

Default value False

BooleanField field

Aliases

- **is_group_msg**

property isStatusV3 : bool

Default value False

BooleanField field

Aliases

- **is_status_v3**

property isPSA : bool

Default value False

BooleanField field

Aliases

- **is_psa**

property statusV3TextBg : str
StringIdField field

Aliases

- **status_v3_text_bg**

property isSentByMe : bool

Default value False

BooleanField field

Aliases

- **is_sent_by_me**

property isNotification : bool

Default value False

BooleanField field

Aliases

- **is_notification**

property isGroupNotification : bool

Default value False

BooleanField field

Aliases

- **is_group_notification**

property isBizNotification : bool

Default value False

BooleanField field

Aliases

- **is_biz_notification**

property isMedia : bool

Default value False

BooleanField field

Aliases

- **is_media**

property isLink : bool

Default value False

BooleanField field

Aliases

- **is_link**

property hasLink : bool

Default value False

BooleanField field

Aliases

- **has_link**

property isDoc : bool

Default value False

BooleanField field

Aliases

- **is_doc**

property isMms : bool

Default value False

BooleanField field

Aliases

- **is_mms**

property isRevoked : bool

Default value False

BooleanField field

Aliases

- **is_revoked**

property showForwarded : bool

Default value False

BooleanField field

Aliases

- **show_forwarded**

property containsEmoji : bool

Default value False

BooleanField field

Aliases

- **contains_emoji**

property isFailed : bool

Default value False

BooleanField field

Aliases

- **is_failed**

property dir : str

StringIdField field

property rtl : bool

Default value False

BooleanField field

property isPersistent : bool

Default value False

BooleanField field

Aliases

- **is_persistent**

property isUserCreatedType : bool

Default value False

BooleanField field

Aliases

- **is_user_created_type**

property hasPromises : bool

Default value False

BooleanField field

Aliases

- **has_promises**

Model whalesong.managers.message.**GroupNotificationMessage**(*data=None*, **args*, ***kwargs*)

Bases: *whalesong.managers.message.BaseMessage*

Notification message.

property id : **str** *[READ ONLY]*
StringIdField field [READ ONLY]

property type : ***MessageTypes*** *[READ ONLY]*
EnumField field [READ ONLY] (*whalesong.managers.message.MessageTypes*)

property subtype : **str**
StringIdField field

property body : **str**
StringIdField field

property timestamp : **datetime**
DateTimeField field

Aliases

- **t**

property notifyName : **str**
StringIdField field

Aliases

- **notify_name**

property from : **str**
StringIdField field

Aliases

- **from**

property to : **str**
StringIdField field

property author : **str**
StringIdField field

property sender : **str**
StringIdField field

property senderObj : ***Contact***
ModelField field (*whalesong.managers.contact.Contact*)

Aliases

- **sender_obj**

property self : **str**

Default value in

StringIdField field

property ack : ***Ack***
EnumField field (*whalesong.managers.message.Ack*)

property invis : **bool**

Default value False

BooleanField field

property isNewMsg : bool

Default value False

BooleanField field

Aliases

- **is_new_msg**

property star : bool

Default value False

BooleanField field

property isForwarded : bool

Default value False

BooleanField field

Aliases

- **is_forwarded**

property links : List of str

Array of StringIdField field

property chat : *Chat*

ModelField field (*whalesong.managers.chat.Chat*)

property isGroupMsg : bool

Default value False

BooleanField field

Aliases

- **is_group_msg**

property isStatusV3 : bool

Default value False

BooleanField field

Aliases

- **is_status_v3**

property isPSA : bool

Default value False

BooleanField field

Aliases

- **is_psa**

property statusV3TextBg : str

StringIdField field

Aliases

- **status_v3_text_bg**

property isSentByMe : bool

Default value False

BooleanField field

Aliases

- **is_sent_by_me**

property isNotification : bool

Default value False

BooleanField field

Aliases

- **is_notification**

property isGroupNotification : bool

Default value False

BooleanField field

Aliases

- **is_group_notification**

property isBizNotification : bool

Default value False

BooleanField field

Aliases

- **is_biz_notification**

property isMedia : bool

Default value False

BooleanField field

Aliases

- **is_media**

property isLink : bool

Default value False

BooleanField field

Aliases

- **is_link**

property hasLink : bool

Default value False

BooleanField field

Aliases

- **has_link**

property isDoc : bool

Default value False

BooleanField field

Aliases

- **is_doc**

property isMms : bool

Default value False

BooleanField field

Aliases

- **is_mms**

property isRevoked : bool

Default value False

BooleanField field

Aliases

- **is_revoked**

property showForwarded : bool

Default value False

BooleanField field

Aliases

- **show_forwarded**

property containsEmoji : bool

Default value False

BooleanField field

Aliases

- **contains_emoji**

property isFailed : bool

Default value False

BooleanField field

Aliases

- **is_failed**

property dir : str

StringIdField field

property rtl : bool

Default value False

BooleanField field

property isPersistent : bool

Default value False

BooleanField field

Aliases

- **is_persistent**

property isUserCreatedType : bool

Default value False

BooleanField field

Aliases

- **is_user_created_type**

property hasPromises : bool

Default value False

BooleanField field

Aliases

- **has_promises**

property urlText : str

StringIdField field

Aliases

- **url_text**

property urlNumber : int

IntegerField field

Aliases

- **url_number**

property recipients : List of str

Array of StringIdField field

property broadcast : bool

Default value False

BooleanField field

property multicast : bool

Default value False

BooleanField field

Model whalesong.managers.message.**StickerMessage**(*data=None*, **args*, ***kwargs*)

Bases: *whalesong.managers.message.ImageMessage*

Sticker message.

property id : str *[READ ONLY]*

StringIdField field [READ ONLY]

property quotedMsgObj : *BaseMessage*

ModelField field (*whalesong.managers.message.BaseMessage*)

Aliases

- **quoted_msg_obj**

property quotedStanzaID : str
StringIdField field

Aliases

- **quoted_stanza_id**

property quotedParticipant : str
StringIdField field

Aliases

- **quoted_participant**

property quotedRemoteJid : str
StringIdField field

Aliases

- **quoted_remote_jid**

property mentionedJidList : List of str
Array of StringIdField field

Aliases

- **mentioned_jid_list**

property type : *MessageTypes* *[READ ONLY]*
EnumField field [READ ONLY] (*whalesong.managers.message.MessageTypes*)

property clientUrl : str
StringIdField field

Aliases

- **client_url**

property directPath : str
StringIdField field

Aliases

- **direct_path**

property mimetype : str
StringIdField field

property caption : str
StringIdField field

property filehash : str
StringIdField field

property size : int
IntegerField field

property mediaKey : str
StringIdField field

Aliases

- **media_key**

property isUnsentMedia : bool

Default value False

BooleanField field

Aliases

- **is_unsent_media**

property body : None
Base64Field field

property height : int
IntegerField field

property width : int
IntegerField field

property subtype : str
StringIdField field

property timestamp : datetime
DateTimeField field

Aliases

- **t**

property notifyName : str
StringIdField field

Aliases

- **notify_name**

property from : str
StringIdField field

Aliases

- **from**

property to : str
StringIdField field

property author : str
StringIdField field

property sender : str
StringIdField field

property senderObj : *Contact*
ModelField field (*whalesong.managers.contact.Contact*)

Aliases

- **sender_obj**

property self : str

Default value in

StringIdField field

property ack : *Ack*
EnumField field (*whalesong.managers.message.Ack*)

property invis : bool

Default value False

BooleanField field

property isNewMsg : bool

Default value False

BooleanField field

Aliases

- **is_new_msg**

property star : bool

Default value False

BooleanField field

property isForwarded : bool

Default value False

BooleanField field

Aliases

- **is_forwarded**

property links : List of str

Array of StringIdField field

property chat : *Chat*

ModelField field (*whalesong.managers.chat.Chat*)

property isGroupMsg : bool

Default value False

BooleanField field

Aliases

- **is_group_msg**

property isStatusV3 : bool

Default value False

BooleanField field

Aliases

- **is_status_v3**

property isPSA : bool

Default value False

BooleanField field

Aliases

- **is_psa**

property statusV3TextBg : str

StringIdField field

Aliases

- **status_v3_text_bg**

property isSentByMe : bool

Default value False

BooleanField field

Aliases

- **is_sent_by_me**

property isNotification : bool

Default value False

BooleanField field

Aliases

- **is_notification**

property isGroupNotification : bool

Default value False

BooleanField field

Aliases

- **is_group_notification**

property isBizNotification : bool

Default value False

BooleanField field

Aliases

- **is_biz_notification**

property isMedia : bool

Default value False

BooleanField field

Aliases

- **is_media**

property isLink : bool

Default value False

BooleanField field

Aliases

- **is_link**

property hasLink : bool

Default value False

BooleanField field

Aliases

- **has_link**

property isDoc : bool

Default value False

BooleanField field

Aliases

- **is_doc**

property isMms : bool

Default value False

BooleanField field

Aliases

- **is_mms**

property isRevoked : bool

Default value False

BooleanField field

Aliases

- **is_revoked**

property showForwarded : bool

Default value False

BooleanField field

Aliases

- **show_forwarded**

property containsEmoji : bool

Default value False

BooleanField field

Aliases

- **contains_emoji**

property isFailed : bool

Default value False

BooleanField field

Aliases

- **is_failed**

property dir : str

StringIdField field

property rtl : bool

Default value False

BooleanField field

property isPersistent : bool

Default value False

BooleanField field

Aliases

- **is_persistent**

property isUserCreatedType : bool

Default value False

BooleanField field

Aliases

- **is_user_created_type**

property hasPromises : bool

Default value False

BooleanField field

Aliases

- **has_promises**

Model whalesong.managers.message.**MessageAck**(*data=None*, *flat=False*, **args*, ***kwargs*)

Bases: *whalesong.models.BaseModel*

Message acknowledgement.

property id : str *[READ ONLY]*

StringIdField field [READ ONLY]

property timestamp : datetime

Ack timestamp.

Aliases

- **t**

Model whalesong.managers.message.**MessageInfo**(*data=None*, *flat=False*, **args*, ***kwargs*)

Bases: *whalesong.models.BaseModel*

Message information.

property id : str *[READ ONLY]*

StringIdField field [READ ONLY]

property deliveryRemaining : int

Default value 0

IntegerField field

Aliases

- **delivery_remaining**

property isPtt : bool

Default value False

BooleanField field

Aliases

- **is_ptt**

property playedRemaining : int

Default value 0

IntegerField field

Aliases

- **played_remaining**

property readRemaining : int

Default value 0

IntegerField field

Aliases

- **read_remaining**

property delivery : List of *MessageAck*

Delivery message acknowledgement list.

property played : List of *MessageAck*

Played message acknowledgement list.

property read : List of *MessageAck*

Read message acknowledgement list.

await whalesong.managers.message.**download_media**(*driver*, *model*)

Download message's attached media file. It will decrypt media file using key on message object.

Parameters

- **driver** (*BaseWhalesongDriver*) –
- **model** (*MediaMixin*) – MediaMixin

Return type BytesIO

Returns Media stream.

3.4.9 Group metadata references

3.4.9.1 Managers

class whalesong.managers.group_metadata.**GroupMetadataCollectionManager**(*driver*, *manager_path=''*)

Bases: *whalesong.managers.BaseCollectionManager*

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)

Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

__getattr__(*name*)

Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

find_item_by_id(*item_id*)
Find model by identifier. If item is not in collection it will try to load it.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_first()
Get first item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_by_id(*item_id*)
Get model by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_result_class()

Return type *Result*[+MODEL_TYPE]

get_items()
Get all items on collection.

Return type *IteratorResult*[+MODEL_TYPE]

Returns Async iterator

get_iterator_result_class()

Return type *IteratorResult*[+MODEL_TYPE]

get_last()
Get last item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_length()
Get collection items count.

Return type *Result*[int]

Returns Items count

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

monitor_add()
Monitor add item collection. Iterate each time a item is added to collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_change()
Monitor change item collection. Iterate each time a item change in collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_field(*field*)
Monitor item's field change. Iterate each time a field changed in any item of collection.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model object iterator

monitor_remove()
Monitor remove item collection. Iterate each time a item is removed from collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

remove_item_by_id(*item_id*)
Remove item by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[None]

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

MODEL_MANAGER_CLASS
alias of *GroupMetadataManager*

class whalesong.managers.group_metadata.**GroupMetadataManager**(*driver*, *manager_path=''*)

Bases: *whalesong.managers.BaseModelManager*

Group metadata manager. It allows manage groups, further than a chat.

participants
ParticipantCollectionManager

Group's participants collection manager.

__getitem__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

MODEL_CLASS
alias of *GroupMetadata*

group_invite_code()

Return type *Result*[None]

revoke_group_invite()

Return type *Result*[None]

class whalesong.managers.group_metadata.**ParticipantCollectionManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseCollectionManager*

Participant collection manager. It allows manage group participants.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

__getattr__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

MODEL_MANAGER_CLASS
alias of *ParticipantManager*

add_participants(*contact_ids*)

Return type *Result*[None]

can_add(*contact_id*)

Return type *Result*[bool]

remove_participants(*contact_ids*)

Return type *Result*[None]

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

can_remove(*contact_id*)

Return type *Result*[bool]

find_item_by_id(*item_id*)
Find model by identifier. If item is not in collection it will try to load it.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_first()
Get first item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_by_id(*item_id*)
Get model by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_result_class()

Return type *Result*[+MODEL_TYPE]

get_items()
Get all items on collection.

Return type *IteratorResult*[+MODEL_TYPE]

Returns Async iterator

get_iterator_result_class()

Return type *IteratorResult*[+MODEL_TYPE]

get_last()
Get last item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_length()
Get collection items count.

Return type *Result*[int]

Returns Items count

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

monitor_add()
Monitor add item collection. Iterate each time a item is added to collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_change()
Monitor change item collection. Iterate each time a item change in collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_field(*field*)
Monitor item's field change. Iterate each time a field changed in any item of collection.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model object iterator

monitor_remove()
Monitor remove item collection. Iterate each time a item is removed from collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

remove_item_by_id(*item_id*)
Remove item by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[None]

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

promote_participants(*contact_ids*)

Return type *Result*[None]

can_promote(*contact_id*)

Return type *Result*[bool]

demote_participants(*contact_ids*)

Return type *Result*[None]

can_demote(*contact_id*)

Return type *Result*[bool]

class whalesong.managers.group_metadata.**ParticipantManager**(*driver*, *manager_path=''*)

Bases: *whalesong.managers.BaseModelManager*

Participant manager.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)

Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)

Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

MODEL_CLASS

alias of *Participant*

add_submanager(*name*, *submanager*)

Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()

Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()

Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

3.4.9.2 Models

Model whalesong.managers.group_metadata.**Participant**(*data=None*, *flat=False*, **args*, ***kwargs*)

Bases: *whalesong.models.BaseModel*

property id : str *[READ ONLY]*
StringIdField field [READ ONLY]

property isAdmin : bool

Default value False

BooleanField field

Aliases

- **is_admin**

property isSuperAdmin : bool
BooleanField field

Aliases

- **is_super_admin**

Model whalesong.managers.group_metadata.**GroupMetadata**(*data=None*, *flat=False*, **args*, ***kwargs*)

Bases: *whalesong.models.BaseModel*

Group metadata model.

property announce : str
¿?

property creation : datetime
Group creation timestamp.

property desc : str
Group description.

property owner : str
Who made group.

property participants : List of *Participant*
List of participants.

property restrict : str
¿?

property id : str *[READ ONLY]*
StringIdField field [READ ONLY]

property descOwner : str
StringIdField field

Aliases

- **desc_owner**

property descTime : datetime
DateTimeField field

Aliases

- **desc_time**

property groupInviteLink : str
StringIdField field

Aliases

- **group_invite_link**

property inviteCode : str
StringIdField field

Aliases

- **invite_code**

3.4.10 Wap references

3.4.10.1 Managers

class whalesong.managers.wap.**WapManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseModelManager*

Entry point to request data to phone.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

MODEL_CLASS
alias of *Wap*

query_exist(*contact_id*)
Check whether a contact identifier exists on Whatsapp or not.

Parameters **contact_id** (str) – Contact identifier

Return type *Result*[bool]

Returns Bool

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

3.4.10.2 Models

Model whalesong.managers.wap.**Wap**(**args*, ***kwargs*)
Bases: dirty_models.models.FastDynamicModel

3.4.11 Sticker pack references

3.4.11.1 Managers

class whalesong.managers.sticker_pack.**StickerPackCollectionManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseCollectionManager*

Sticker pack collection manager. It allows manage sticker pack collection.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

__getattr__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

MODEL_MANAGER_CLASS
alias of *StickerPackManager*

fetch_page(*page*)
Fetch a sticker pack page.

Return type *Result*[~T]

fetch_all_pages()
Fetch all sticker pack pages.

Return type *Result*[bool]

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

find_item_by_id(*item_id*)
Find model by identifier. If item is not in collection it will try to load it.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_first()
Get first item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_by_id(*item_id*)
Get model by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_result_class()

Return type *Result*[+MODEL_TYPE]

get_items()
Get all items on collection.

Return type *IteratorResult*[+MODEL_TYPE]

Returns Async iterator

get_iterator_result_class()

Return type *IteratorResult*[+MODEL_TYPE]

get_last()
Get last item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_length()
Get collection items count.

Return type *Result*[int]

Returns Items count

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

monitor_add()
Monitor add item collection. Iterate each time a item is added to collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_change()
Monitor change item collection. Iterate each time a item change in collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_field(*field*)
Monitor item's field change. Iterate each time a field changed in any item of collection.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model object iterator

monitor_remove()
Monitor remove item collection. Iterate each time a item is removed from collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

remove_item_by_id(*item_id*)
Remove item by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[None]

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

reset()
Reset sticker pack collection.

Return type *Result*[~T]

get_item_by_name(*name*)
Get sticker pack by name.

Parameters **name** (str) – Sticker pack name.

Return type *Result*[*StickerPack*]

Returns Sticker pack object.

class whalesong.managers.sticker_pack.**StickerPackManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseModelManager*

Sticker pack manager. It allows manage a sticker pack.

stickers
StickerCollectionManager

Sticker collection manager.

__getitem__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

MODEL_CLASS
alias of *StickerPack*

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

class whalesong.managers.sticker_pack.**StickerCollectionManager**(*driver*, *manager_path=''*)

Bases: *whalesong.managers.BaseCollectionManager*

Sticker collection manager. It allows manage sticker collection.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

__getattr__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

MODEL_MANAGER_CLASS
alias of *StickerManager*

fetch()
Fetch all stickers. You must fetch stickers before try to us them.

Return type *Result*[None]

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

find_item_by_id(*item_id*)
Find model by identifier. If item is not in collection it will try to load it.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_first()
Get first item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_by_id(*item_id*)
Get model by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_result_class()

Return type *Result*[+MODEL_TYPE]

get_items()
Get all items on collection.

Return type *IteratorResult*[+MODEL_TYPE]

Returns Async iterator

get_iterator_result_class()

Return type *IteratorResult*[+MODEL_TYPE]

get_last()
Get last item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_length()
Get collection items count.

Return type *Result*[int]

Returns Items count

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

monitor_add()
Monitor add item collection. Iterate each time a item is added to collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_change()
Monitor change item collection. Iterate each time a item change in collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_field(*field*)
Monitor item's field change. Iterate each time a field changed in any item of collection.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model object iterator

monitor_remove()
Monitor remove item collection. Iterate each time a item is removed from collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

remove_item_by_id(*item_id*)
Remove item by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[None]

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

class whalesong.managers.sticker_pack.**StickerManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseModelManager*

Sticker manager. It allows manage a sticker.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

MODEL_CLASS
alias of *Sticker*

send_to_chat(*chat_id*, *quoted_msg_id=None*)
Send this sticker to a chat.

Parameters

- **chat_id** (str) – Chat indentifier where to send sticker.
- **quoted_msg_id** (Optional[str]) – Quoted message identifier.

Return type *Result*[str]

Returns Message identifier.

await download_image()
Download sticker's image file. It will decrypt image file using key on sticker object.

Return type BytesIO

Returns Image stream

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

3.4.11.2 Models

Model whalesong.managers.sticker_pack.**Sticker**(*data=None*, *flat=False*, **args*, ***kwargs*)
Bases: *whalesong.managers.message.MediaMixin*, *whalesong.models.BaseModel*

property type : ***MessageTypes*** *[READ ONLY]*

Default value MessageTypes.STICKER

EnumField field [READ ONLY] (*whalesong.managers.message.MessageTypes*)

property id : **str** *[READ ONLY]*
StringIdField field [READ ONLY]

property clientUrl : **str**
StringIdField field

Aliases

- **client_url**

property directPath : **str**
StringIdField field

Aliases

- **direct_path**

property mimetype : **str**
StringIdField field

property caption : **str**
StringIdField field

property filehash : str
StringIdField field

property size : int
IntegerField field

property mediaKey : str
StringIdField field

Aliases

- **media_key**

property isUnsentMedia : bool

Default value False

BooleanField field

Aliases

- **is_unsent_media**

Model whalesong.managers.sticker_pack.**StickerPack**(*data=None*, *flat=False*, **args*, ***kwargs*)
Bases: *whalesong.models.BaseModel*

property name : str
Sticker pack name.

property url : str
Sticker pack url image.

property id : str *[READ ONLY]*
StringIdField field [READ ONLY]

3.4.12 Connection references

3.4.12.1 Managers

class whalesong.managers.conn.**ConnManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseModelManager*

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

MODEL_CLASS
alias of *Conn*

update_pushname(*name*)
Update user's push name.

Parameters **name** (str) – New push name

Return type *Result*[None]

can_update_pushname()
Whether it is possible to update push name or not.

Return type *Result*[bool]

Returns Whether it is possible to update push name or not.

3.4.12.2 Models

Model whalesong.managers.conn.**PhoneDescription**(*data=None*, *flat=False*, **args*, ***kwargs*)
Bases: *whalesong.models.BaseModel*

property mcc : str
¿?

property mnc : str
¿?

property id : str *[READ ONLY]*
StringIdField field [READ ONLY]

property whatsappVersion : str
StringIdField field

Aliases

- **whatsapp_version**
- **wa_version**

property osVersion : str
StringIdField field

Aliases

- **os_version**

property deviceManufacturer : str
StringIdField field

Aliases

- **device_manufacturer**

property deviceModel : str
StringIdField field

Aliases

- **device_model**

property osBuildNumber : str
StringIdField field

Aliases

- **os_build_number**

Model whalesong.managers.conn.**Conn**(*data=None*, *flat=False*, **args*, ***kwargs*)
Bases: *whalesong.models.BaseModel*

property ref : **str**
Client token reference. Used on QR.

property refTTL : **int**
Ref time to live

property connected : **bool**
Whether is connected or not.

property me : **str**
Whatsapp user identifier.

property id : **str** *[READ ONLY]*
StringIdField field [READ ONLY]

property whatsappId : **str**
StringIdField field

Aliases

- **whatsapp_id**
- **wid**

property protoVersion : **List of int**
Array of IntegerField field

Aliases

- **proto_version**

property clientToken : **str**
StringIdField field

Aliases

- **client_token**

property serverToken : **str**
StringIdField field

Aliases

- **server_token**

property isResponse : **bool**
BooleanField field

Aliases

- **is_response**

property smbTos : **int**
IntegerField field

Aliases

- **smb_tos**

property battery : **int**
Phone battery level, in percentage.

property plugged : **bool**
Whether phone is plugged to charger or not.

property locale : *str*
Phone locale.

Aliases

- **lc**

property language : *str*
Phone language.

Aliases

- **lg**

property locales : *str*
Phone locale-language.

property is_24h : *bool*
Whether time must be in 24h format.

Aliases

- **is24h**

property platform : *str*
Platform (android, iphone, wp7, etc...)

property phone : *PhoneDescription*
Phone description.

property tos : *int*
¿?

property pushname : *str*
Current user's push name.

3.4.13 Stream references

3.4.13.1 Managers

class whalesong.managers.stream.**StreamManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseModelManager*

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (*str*) – Manager prefix path.

__getitem__(*name*)
Get a submanager.

Parameters **name** (*str*) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (*str*) – Field where submanager was stored.

Return type *BaseManager*

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

MODEL_CLASS
alias of *Stream*

poke()
Refresh ref field. It is used to refresh QR image when it expires.

Return type *Result*[None]

takeover()
Refresh login. It is used to take session again when other browser has been started session.

Return type *Result*[None]

logout()
Logs out of currently logged in session

Return type *Result*[None]

3.4.13.2 Models

Model whalesong.managers.stream.**Stream**(*data=None*, *flat=False*, **args*, ***kwargs*)
Bases: *whalesong.models.BaseModel*

Connection stream model.

class State
Bases: enum.Enum

Connection states.

OPENING = 'OPENING'
Opening stream.

PAIRING = 'PAIRING'
Pairing WhatsappWeb with a phone.

UNPAIRED = 'UNPAIRED'
Unpaired WhatsappWeb with a phone. QR is available.

UNPAIRED_IDLE = 'UNPAIRED_IDLE'
Unpaired WhatsappWeb with a phone. QR is not available.

CONNECTED = 'CONNECTED'
WhatsappWeb is connected to a phone.

TIMEOUT = 'TIMEOUT'
WhatsappWeb connection to a phone is timeout.

CONFLICT = 'CONFLICT'
Other browser has initiated WhatsappWeb with same phone.

UNLAUNCHED = 'UNLAUNCHED'
WhatsappWeb application has not been launched.

PROXYBLOCK = 'PROXYBLOCK'
Proxy is blocking connection.

TOS_BLOCK = 'TOS_BLOCK'
¿?

SMB_TOS_BLOCK = 'SMB_TOS_BLOCK'
¿?

class Stream
Bases: enum.Enum

An enumeration.

DISCONNECTED = 'DISCONNECTED'
Stream disconnected.

SYNCING = 'SYNCING'
Synchronizing data with phone.

RESUMING = 'RESUMING'
Resuming connection with phone.

CONNECTED = 'CONNECTED'
Connected to phone.

property id : str *[READ ONLY]*
StringIdField field [READ ONLY]

property backoffGeneration : int
IntegerField field

Aliases

- **backoff_generation**

property canSend : bool
BooleanField field

Aliases

- **can_send**

property hasSynced : bool
BooleanField field

Aliases

- **has_synced**

property isIncognito : bool
BooleanField field

Aliases

- **is_incognito**

property retryTimestamp : datetime
DateTimeField field

Aliases

- **retry_timestamp**

property syncTag : str
StringIdField field

Aliases

- **sync_tag**

launch_generation = None
¿?

property launched : bool
Whether it has been launched.

property state : *State*
Current stream connection state.

property stream : *Stream*
Current stream state

3.4.14 Presence references

3.4.14.1 Managers

class whalesong.managers.presence.**PresenceCollectionManager**(*driver*, *manager_path=''*)

Bases: *whalesong.managers.BaseCollectionManager*

Presence collection manager.

Note: Be aware Whatsapp has some limitations about presence announcement.

- You must be available in order to receive presence announcements. Look at *display information*.
- After a while, other peer must see Whalesong user available in order to send presence announcement. But if it is not available Whalesong user is not going to announce its presence. So, presence announcements are made after first message.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

__getattr__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

MODEL_MANAGER_CLASS
alias of *PresenceManager*

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

find_item_by_id(*item_id*)
Find model by identifier. If item is not in collection it will try to load it.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_first()
Get first item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_by_id(*item_id*)
Get model by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_result_class()

Return type *Result*[+MODEL_TYPE]

get_items()
Get all items on collection.

Return type *IteratorResult*[+MODEL_TYPE]

Returns Async iterator

get_iterator_result_class()

Return type *IteratorResult*[+MODEL_TYPE]

get_last()
Get last item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_length()
Get collection items count.

Return type *Result*[int]

Returns Items count

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

monitor_add()
Monitor add item collection. Iterate each time a item is added to collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_change()
Monitor change item collection. Iterate each time a item change in collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_field(*field*)
Monitor item's field change. Iterate each time a field changed in any item of collection.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model object iterator

monitor_remove()
Monitor remove item collection. Iterate each time a item is removed from collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

remove_item_by_id(*item_id*)
Remove item by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[None]

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

class whalesong.managers.presence.**PresenceManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseModelManager*

Presence manager.

__getitem__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

MODEL_CLASS
alias of *Presence*

subscribe()

Return type *Result*[*Presence*]

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (*str*) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (*str*) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (*str*) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (*str*) – Field where submanager was stored.

Return type *Result*[~T]

class whalesong.managers.presence.**ChatStateCollectionManager**(*driver*, *manager_path=''*)

Bases: *whalesong.managers.BaseCollectionManager*

Chat state collection manager.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)

Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

__getattr__(*name*)

Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

MODEL_MANAGER_CLASS

alias of *ChatStateManager*

add_submanager(*name*, *submanager*)

Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

find_item_by_id(*item_id*)

Find model by identifier. If item is not in collection it will try to load it.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_commands()

Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_first()

Get first item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_by_id(*item_id*)

Get model by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_result_class()

Return type *Result*[+MODEL_TYPE]

get_items()
Get all items on collection.

Return type *IteratorResult*[+MODEL_TYPE]

Returns Async iterator

get_iterator_result_class()

Return type *IteratorResult*[+MODEL_TYPE]

get_last()
Get last item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_length()
Get collection items count.

Return type *Result*[int]

Returns Items count

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

monitor_add()
Monitor add item collection. Iterate each time a item is added to collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_change()
Monitor change item collection. Iterate each time a item change in collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_field(*field*)
Monitor item's field change. Iterate each time a field changed in any item of collection.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model object iterator

monitor_remove()
Monitor remove item collection. Iterate each time a item is removed from collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

remove_item_by_id(*item_id*)
Remove item by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[None]

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

class whalesong.managers.presence.**ChatStateManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseModelManager*

Chat state manager.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

MODEL_CLASS
alias of *ChatState*

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

3.4.14.2 Models

Model whalesong.managers.presence.**ChatState**(*data=None*, *flat=False*, **args*, ***kwargs*)
Bases: *whalesong.models.BaseModel*

class Type
Bases: enum.Enum

An enumeration.

AVAILABLE = 'available'

UNAVAILABLE = 'unavailable'

COMPOSING = 'composing'

RECORDING = 'recording'

property deny : bool
BooleanField field

property timestamp : datetime
DateTimeField field

Aliases

- **t**

property type : *Type*
EnumField field (whalesong.managers.presence.Type)

property id : **str** *[READ ONLY]*
StringIdField field [READ ONLY]

property isState : **bool**
BooleanField field

Aliases

- **is_state**

property updateTime : **datetime**
DateTimeField field

Aliases

- **update_time**

Model whalesong.managers.presence.**Presence**(*data=None*, *flat=False*, **args*, ***kwargs*)
Bases: *whalesong.models.BaseModel*

Presence model.

property id : **str** *[READ ONLY]*
StringIdField field [READ ONLY]

property chatActive : **bool**
BooleanField field

Aliases

- **chat_active**

property chatState : *ChatState*
ModelField field (*whalesong.managers.presence.ChatState*)

Aliases

- **chat_state**

property chatStates : **List of** *ChatState*
Array of ModelField field (*whalesong.managers.presence.ChatState*)

Aliases

- **chat_states**
- **chatstates**

property hasData : **bool**
BooleanField field

Aliases

- **has_data**

property isGroup : **bool**
BooleanField field

Aliases

- **is_group**

property isUser : **bool**
BooleanField field

Aliases

- **is_user**

property isOnline : **bool**
BooleanField field

Aliases

- **is_online**

property isSubscribed : **bool**
BooleanField field

Aliases

- **is_subscribed**

3.4.15 Profile pictures references

3.4.15.1 Managers

class whalesong.managers.profile_pic_thumb.**ProfilePictureCollectionManager**(*driver*, *manager_path=''*)

Bases: *whalesong.managers.BaseCollectionManager*

Profile picture collection manager.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

__getattr__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

find_item_by_id(*item_id*)
Find model by identifier. If item is not in collection it will try to load it.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_first()
Get first item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_by_id(*item_id*)
Get model by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_result_class()

Return type *Result*[+MODEL_TYPE]

get_items()
Get all items on collection.

Return type *IteratorResult*[+MODEL_TYPE]

Returns Async iterator

get_iterator_result_class()

Return type *IteratorResult*[+MODEL_TYPE]

get_last()
Get last item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_length()
Get collection items count.

Return type *Result*[int]

Returns Items count

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

monitor_add()
Monitor add item collection. Iterate each time a item is added to collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_change()
Monitor change item collection. Iterate each time a item change in collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_field(*field*)
Monitor item's field change. Iterate each time a field changed in any item of collection.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model object iterator

monitor_remove()
Monitor remove item collection. Iterate each time a item is removed from collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

remove_item_by_id(*item_id*)
Remove item by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[None]

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

MODEL_MANAGER_CLASS
alias of *ProfilePictureManager*

class whalesong.managers.profile_pic_thumb.**ProfilePictureManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseModelManager*

Profile picture manager. It allows manage a contact picture.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

MODEL_CLASS
alias of *ProfilePicture*

can_set()
Whether can set a new picture or not.

Return type *Result*[bool]

set_picture(*picture*, *picture_preview=None*)
Set a new picture.

Return type *Result*[bool]

can_delete()
Whether can delete picture or not.

Return type *Result*[bool]

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

delete_picture()
Delete current picture.

Return type *Result*[bool]

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

3.4.15.2 Models

Model whalesong.managers.profile_pic_thumb.**ProfilePicture**(*data=None*, *flat=False*, **args*, ***kwargs*)
Bases: *whalesong.models.BaseModel*

property eurl : str
Url to contact picture.

property raw : str
¿?

property tag : str
¿?

Note: I guess it is used to know when contact picture changed.

property id : str *[READ ONLY]*
StringIdField field [READ ONLY]

3.4.16 Status references

3.4.16.1 Managers

class whalesong.managers.status.**StatusCollectionManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseCollectionManager*

Status collection manager.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

__getattr__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

MODEL_MANAGER_CLASS
alias of *StatusManager*

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

find_item_by_id(*item_id*)
Find model by identifier. If item is not in collection it will try to load it.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_first()
Get first item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_by_id(*item_id*)
Get model by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_result_class()

Return type *Result*[+MODEL_TYPE]

get_items()
Get all items on collection.

Return type *IteratorResult*[+MODEL_TYPE]

Returns Async iterator

get_iterator_result_class()

Return type *IteratorResult*[+MODEL_TYPE]

get_last()
Get last item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_length()
Get collection items count.

Return type *Result*[int]

Returns Items count

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

monitor_add()
Monitor add item collection. Iterate each time a item is added to collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_change()
Monitor change item collection. Iterate each time a item change in collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_field(*field*)
Monitor item's field change. Iterate each time a field changed in any item of collection.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model object iterator

monitor_remove()
Monitor remove item collection. Iterate each time a item is removed from collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

remove_item_by_id(*item_id*)
Remove item by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[None]

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

set_my_status(*new_status=None*)
Set current user status.

Return type *Result*[bool]

Returns Operation result.

class whalesong.managers.status.**StatusManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseModelManager*

Status manager.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

MODEL_CLASS
alias of *Status*

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

3.4.16.2 Models

Model whalesong.managers.status.**Status**(*data=None*, *flat=False*, **args*, ***kwargs*)
Bases: *whalesong.models.BaseModel*

property status : str
Contact status.

property id : str *[READ ONLY]*
StringIdField field [READ ONLY]

3.4.17 Live location references

3.4.17.1 Managers

class whalesong.managers.live_location.**LiveLocationCollectionManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseCollectionManager*

Live locations collection manager.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

__getattr__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

find_item_by_id(*item_id*)
Find model by identifier. If item is not in collection it will try to load it.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_first()
Get first item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_by_id(*item_id*)
Get model by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_result_class()

Return type *Result*[+MODEL_TYPE]

get_items()
Get all items on collection.

Return type *IteratorResult*[+MODEL_TYPE]

Returns Async iterator

get_iterator_result_class()

Return type *IteratorResult*[+MODEL_TYPE]

get_last()
Get last item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_length()
Get collection items count.

Return type *Result*[int]

Returns Items count

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

monitor_add()
Monitor add item collection. Iterate each time a item is added to collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_change()
Monitor change item collection. Iterate each time a item change in collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_field(*field*)
Monitor item's field change. Iterate each time a field changed in any item of collection.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model object iterator

monitor_remove()
Monitor remove item collection. Iterate each time a item is removed from collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

remove_item_by_id(*item_id*)
Remove item by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[None]

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

MODEL_MANAGER_CLASS
alias of *LiveLocationManager*

class whalesong.managers.live_location.**LiveLocationManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseModelManager*

Live location manager.

participants
ParticipantCollectionManager

Live location's participants collection manager.

__getitem__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

MODEL_CLASS
alias of *LiveLocation*

subscribe()
Subscribe to a live location. It is needed in order to receive updates.

Return type *Result*[None]

unsubscribe()
Unsubscribe to a live location.

Return type *Result*[None]

stop_my_live_location()
Stop to share current user live location in this chat.

Return type *Result*[None]

class whalesong.managers.live_location.**ParticipantCollectionManager**(*driver*, *manager_path=''*)

Bases: *whalesong.managers.BaseCollectionManager*

Participant collection manager. It allows manage live location participants.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

__getattr__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

find_item_by_id(*item_id*)
Find model by identifier. If item is not in collection it will try to load it.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_first()
Get first item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_by_id(*item_id*)
Get model by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_result_class()

Return type *Result*[+MODEL_TYPE]

get_items()
Get all items on collection.

Return type *IteratorResult*[+MODEL_TYPE]

Returns Async iterator

get_iterator_result_class()

Return type *IteratorResult*[+MODEL_TYPE]

get_last()
Get last item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_length()
Get collection items count.

Return type *Result*[int]

Returns Items count

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

monitor_add()
Monitor add item collection. Iterate each time a item is added to collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_change()
Monitor change item collection. Iterate each time a item change in collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_field(*field*)
Monitor item's field change. Iterate each time a field changed in any item of collection.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model object iterator

monitor_remove()
Monitor remove item collection. Iterate each time a item is removed from collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

remove_item_by_id(*item_id*)
Remove item by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[None]

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

MODEL_MANAGER_CLASS
alias of *ParticipantManager*

class whalesong.managers.live_location.**ParticipantManager**(*driver*, *manager_path=''*)

Bases: *whalesong.managers.BaseModelManager*

Participant manager.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver

- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

MODEL_CLASS
alias of *Participant*

3.4.17.2 Models

Model whalesong.managers.live_location.**Participant**(*data=None*, *flat=False*, **args*, ***kwargs*)
Bases: *whalesong.models.BaseModel*

Live location participant model.

property accuracy : **int**
Location accuracy in meters.

property comment : **str**
Comment sent with live location message.

property degrees : **float**
User direction in degrees.

property expiration : **datetime**
When live location will expire.

property lat : **float**
Latitude.

property lng : **float**
Longitude.

property msg : ***BaseMessage***
Message used to start live location.

property contact : ***Contact***
Live location owner.

property sequence : **int**
Sequence number.

property speed : **int**
User speed.

property id : **str** *[READ ONLY]*
StringIdField field [READ ONLY]

property lastUpdate : **datetime**
DateTimeField field

Aliases

- **last_update**

Model whalesong.managers.live_location.**LiveLocation**(*data=None*, *flat=False*, **args*, ***kwargs*)

Bases: *whalesong.models.BaseModel*

Live location model.

property duration : timedelta
Live location duration.

property id : str *[READ ONLY]*
StringIdField field [READ ONLY]

property participants : List of *Participant*
List of participants.

3.4.18 Mute references

3.4.18.1 Managers

class whalesong.managers.mute.**MuteCollectionManager**(*driver*, *manager_path=''*)

Bases: *whalesong.managers.BaseCollectionManager*

Mutes collection manager. It allows manage global mute as well.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

__getattr__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

MODEL_MANAGER_CLASS
alias of *MuteManager*

get_global_notifications()

Return type *Result*[bool]

set_global_notifications(*state*)

Return type *Result*[None]

get_global_sounds()

Return type *Result*[bool]

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (*str*) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

find_item_by_id(*item_id*)
Find model by identifier. If item is not in collection it will try to load it.

Parameters **item_id** (*str*) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_commands()
Get manager available static commands.

Return type *Result*[*List*[*str*]]

Returns Manager static commands.

get_first()
Get first item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_by_id(*item_id*)
Get model by identifier.

Parameters **item_id** (*str*) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_result_class()

Return type *Result*[+MODEL_TYPE]

get_items()
Get all items on collection.

Return type *IteratorResult*[+MODEL_TYPE]

Returns Async iterator

get_iterator_result_class()

Return type *IteratorResult*[+MODEL_TYPE]

get_last()
Get last item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_length()
Get collection items count.

Return type *Result*[*int*]

Returns Items count

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

monitor_add()
Monitor add item collection. Iterate each time a item is added to collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_change()
Monitor change item collection. Iterate each time a item change in collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_field(*field*)
Monitor item's field change. Iterate each time a field changed in any item of collection.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model object iterator

monitor_remove()
Monitor remove item collection. Iterate each time a item is removed from collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

remove_item_by_id(*item_id*)
Remove item by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[None]

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

set_global_sounds(*state*)

Return type *Result*[None]

get_global_previews()

Return type *Result*[bool]

set_global_previews(*state*)

Return type *Result*[None]

class whalesong.managers.mute.**MuteManager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseModelManager*

Mute manager. It allows manage chat mute.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver

- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

MODEL_CLASS
alias of *Mute*

mute(*expiration*)

Return type *Result*[None]

can_mute()

Return type *Result*[bool]

unmute()

Return type *Result*[None]

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

3.4.18.2 Models

Model whalesong.managers.mute.**Mute**(*data=None*, *flat=False*, **args*, ***kwargs*)
Bases: *whalesong.models.BaseModel*

Mute model.

property expiration : datetime
Expiration date time.

property id : str *[READ ONLY]*
StringIdField field [READ ONLY]

property isMuted : bool
BooleanField field

Aliases

- **is_muted**

property isState : bool
BooleanField field

Aliases

- **is_state**

3.4.19 Display information references

3.4.19.1 Managers

class whalesong.managers.display_info.**DisplayInfoManager**(*driver*, *manager_path=''*)

Bases: *whalesong.managers.BaseModelManager*

Manage display information.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

MODEL_CLASS
alias of *DisplayInfo*

mark_available()
Mark current user as available. It is need to get presence from other users.

Return type *Result*[None]

mark_unavailable()
Mark current user as unavailable.

Return type *Result*[None]

unobscure()
Unobscure display.

Return type *Result*[None]

set_available_permanent()
Set user available permanently. It starts a loop in order to set availability each 30 seconds.

unset_available_permanent()
Unset user available permanently. It stops permanent availability loop.

is_available_permanent()
Checks whether permanent availability loop is running.

Returns Permanent availability loop state.

3.4.19.2 Models

Model whalesong.managers.display_info.**DisplayInfo**(*data=None*, *flat=False*, **args*, ***kwargs*)

Bases: *whalesong.models.BaseModel*

Connection stream model.

class StreamInfo

Bases: enum.Enum

Stream information.

OFFLINE = 'OFFLINE'
Offline.

OPENING = 'OPENING'
Opening connection.

PAIRING = 'PAIRING'
Pairing phone.

SYNCING = 'SYNCING'
Synchronizing data.

RESUMING = 'RESUMING'
Resuming connection.

CONNECTING = 'CONNECTING'
Connecting.

NORMAL = 'NORMAL'
Normal.

TIMEOUT = 'TIMEOUT'
Connection timeout.

class StreamMode

Bases: enum.Enum

Stream mode.

QR = 'QR'
Wait for QR scan.

MAIN = 'MAIN'
Main.

SYNCING = 'SYNCING'
Synchronizing data.

OFFLINE = 'OFFLINE'
Not connected.

CONFLICT = 'CONFLICT'
Other browser has opened session.

PROXYBLOCK = 'PROXYBLOCK'
Proxy blocks connection.

TOS_BLOCK = 'TOS_BLOCK'
¿?

SMB_TOS_BLOCK = 'SMB_TOS_BLOCK'
¿?

DEPRECATED_VERSION = 'DEPRECATED_VERSION'
Using a deprecated version.

class DisplayState
Bases: enum.Enum

Display state.

SHOW = 'SHOW'
Display showing.

OBSCURE = 'OBSCURE'
Display obscured.

HIDE = 'HIDE'
Display hidden.

property available : bool

Default value False

Whether current user is available.

property id : str *[READ ONLY]*
StringIdField field [READ ONLY]

property clientExpired : bool
BooleanField field

Aliases

- **client_expired**

property couldForce : bool
BooleanField field

Aliases

- **could_force**

property displayInfo : *StreamInfo*
EnumField field (whalesong.managers.display_info.StreamInfo)

Aliases

- **display_info**

property isState : bool
BooleanField field

Aliases

- **is_state**

property phoneAuthed : bool
BooleanField field

Aliases

- **phone_authed**

property uiActive : bool
BooleanField field

Aliases

- **ui_active**

property resumeCount : int
IntegerField field

Aliases

- **resume_count**

hard_expired = None
¿?

property info : *StreamInfo*
Same than display info?

property obscurity : *DisplayState*
Current display state.

property mode : *StreamMode*
Stream mode.

3.4.20 Status V3 references

3.4.20.1 Managers

class whalesong.managers.status_v3.**StatusV3CollectionManager**(*driver*, *manager_path=''*)

Bases: *whalesong.managers.BaseCollectionManager*

Manage a collection of StatusV3.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

__getattr__(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

MODEL_MANAGER_CLASS
alias of *StatusV3Manager*

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

find_item_by_id(*item_id*)
Find model by identifier. If item is not in collection it will try to load it.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_first()
Get first item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_by_id(*item_id*)
Get model by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_item_result_class()

Return type *Result*[+MODEL_TYPE]

get_items()
Get all items on collection.

Return type *IteratorResult*[+MODEL_TYPE]

Returns Async iterator

get_iterator_result_class()

Return type *IteratorResult*[+MODEL_TYPE]

get_last()
Get last item in collection.

Return type *Result*[+MODEL_TYPE]

Returns Model object.

get_length()
Get collection items count.

Return type *Result*[int]

Returns Items count

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager. It could be a explicit submanager or contained model manager.

Parameters **name** (str) – Field where submanager was stored.

Return type Union[*BaseManager*, ~MODEL_MANAGER_TYPE]

monitor_add()
Monitor add item collection. Iterate each time a item is added to collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_change()
Monitor change item collection. Iterate each time a item change in collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

monitor_field(*field*)
Monitor item's field change. Iterate each time a field changed in any item of collection.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model object iterator

monitor_remove()
Monitor remove item collection. Iterate each time a item is removed from collection.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model object iterator

remove_item_by_id(*item_id*)
Remove item by identifier.

Parameters **item_id** (str) – Model identifier.

Return type *Result*[None]

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

get_unexpired(*unread=True*)
Get the read or unread StatusV3 collection

Parameters **unread** (bool) – List read or unread statuses

Return type *IteratorResult*[*StatusV3*]

Returns List of StatusV3

sync()
Sync Statuses

Return type *Result*[None]

Returns None

get_my_status()
Get the own user StatusV3

Return type *Result*[*StatusV3*]

Returns StatusV3 object

class whalesong.managers.status_v3.**StatusV3Manager**(*driver*, *manager_path=''*)
Bases: *whalesong.managers.BaseModelManager*

StatusV3 manager. Allow to manage a WhatsApp status.

__getitem__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

MODEL_CLASS
alias of *StatusV3*

send_read_status(*message_id*)
Mark a statusV3 as read.

Parameters **message_id** (str) – Message serialized ID to be marked

Return type *Result*[bool]

add_submanager(*name*, *submanager*)
Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()
Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_field_monitor_result_class(*field*)

Return type *MonitorResult*[Dict[str, Any]]

get_model()
Get model object

Return type *Result*[+MODEL_TYPE]

Returns Model object

get_model_result_class()

Return type *Result*[+MODEL_TYPE]

get_monitor_result_class()

Return type *MonitorResult*[+MODEL_TYPE]

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

map_model(*data*)

Return type +MODEL_TYPE

monitor_field(*field*)
Monitor any change on a model's field.

Parameters **field** (str) – Field to monitor.

Return type *MonitorResult*[Dict[str, Any]]

Returns Model monitor

monitor_model()
Monitor any change on model.

Return type *MonitorResult*[+MODEL_TYPE]

Returns Model monitor

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

3.4.20.2 Models

Model whalesong.managers.status_v3.**StatusV3**(*data=None*, *flat=False*, **args*, ***kwargs*)
Bases: *whalesong.models.BaseModel*

StatusV3 model

property contact : ***Contact***
Contact object

property id : **str** *[READ ONLY]*
StringIdField field [READ ONLY]

property unreadCount : **int**
IntegerField field

Aliases

- **unread_count**

property expireTs : **datetime**
DateTimeField field

Aliases

- **expire_ts**

property lastReceivedKey : **str**
StringIdField field

Aliases

- **last_received_key**

property readKeys : **HashMapModel hash map which values are str**
HashMapField field (dirty_models.models.HashMapModel)

Aliases

- **read_keys**

3.4.21 Storage references

3.4.21.1 Managers

class whalesong.managers.storage.**StorageManager**(*driver*, *manager_path=''*)

Bases: *whalesong.managers.BaseManager*

Local storage manager. It allows manage browser local storage.

Parameters

- **driver** (*BaseWhalesongDriver*) – Whalesong driver
- **manager_path** (str) – Manager prefix path.

__getitem__(*name*)

Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

__getattr__(*name*)

Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

get_storage()

Return type *Result*[Dict[str, Any]]

get_item(*key*)

Return type *Result*[Any]

set_item(*key*, *value*)

Return type *Result*[None]

set_storage(*data*)

Return type *Result*[None]

monitor_storage()

Return type *MonitorResult*[Dict[str, Any]]

monitor_item_storage()

Return type *MonitorResult*[Any]

add_submanager(*name*, *submanager*)

Add a submanager.

Parameters

- **name** (str) – Field where manager will be stored.
- **submanager** (*BaseManager*) – Submanager

get_commands()

Get manager available static commands.

Return type *Result*[List[str]]

Returns Manager static commands.

get_submanager(*name*)
Get a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *BaseManager*

remove_submanager(*name*)
Remove a submanager.

Parameters **name** (str) – Field where submanager was stored.

Return type *Result*[~T]

3.4.22 Whalesong Errors

exception whalesong.errors.**WhalesongException**
Bases: Exception

exception whalesong.errors.**ManagerNotFound**
Bases: *whalesong.errors.WhalesongException*

exception whalesong.errors.**UnknownError**
Bases: *whalesong.errors.WhalesongException*

exception whalesong.errors.**ChatNotFoundError**
Bases: *whalesong.errors.WhalesongException*

exception whalesong.errors.**ContactNotFoundError**
Bases: *whalesong.errors.WhalesongException*

exception whalesong.errors.**StopMonitor**
Bases: StopAsyncIteration, *whalesong.errors.WhalesongException*

exception whalesong.errors.**StopIterator**
Bases: StopAsyncIteration, *whalesong.errors.WhalesongException*

exception whalesong.errors.**RequiredExecutionId**
Bases: *whalesong.errors.WhalesongException*

exception whalesong.errors.**RequiredCommandName**
Bases: *whalesong.errors.WhalesongException*

exception whalesong.errors.**ModelNotFound**
Bases: *whalesong.errors.WhalesongException*

3.4.23 Firefox Profile

class whalesong.firefox_profile.**FirefoxProfile**(*profile_directory=None*)
Bases: selenium.webdriver.firefox.firefox_profile.FirefoxProfile

3.5 How to develop

3.5.1 Development requirements

- node (only for development)
- npm (only for development)

- make (only for development)

3.5.2 Install library requirements

```
$ make requirements
```

3.5.3 Build Javascript scriplet

You have to rebuild scriptlet after any change if you want to use in Python code.

```
$ make build-js
```

3.5.4 Beautify code

You must to beautify code before make a pull request. Ugly code will not be accepted.

```
$ make beautify
```

3.6 Changelog

3.6.1 Version 0.9.1 (Work in progress)

- Fix Issue #106

3.6.2 Version 0.9.0

- Added Whatsapp Stories management (StatusV3). Thx to @jabolina.
- Added logout feature. Thx to @parthibd.
- Fixed stop/start driver (see Issue #82). Thx to @parthibd.
- Fixed Firefox driver initialization in order to allow more than one process. Thx to @parthibd.
- Fixed *wait_until_stop* driver method.
- Examples now stop gracefully.

3.6.3 Version 0.8.4

- Fixed requirements installation out of virtual enviorment. Thx to @Theblood.
- Enabled media codecs on Firefox profile template.
- Fixed problems with live location. Issue #73
- Fixed problem monitoring models.
- Added new examples: *getlivelocations-chromium.py* and *getlivelocations.py*

3.6.4 Version 0.8.2

- Fixed QR screenshot on Firefox.

3.6.5 Version 0.8.1

- Fixed Wap object discovery. Thx to @jabolina.
- Fixed screenshot method.
- Added Chromium version of some examples.

3.6.6 Version 0.8.0

- Live locations management.
- Mutes management.
- Revoke messages. Delete messages for others.
- Minimized scriptlet.

3.6.7 Version 0.7.2

- Capability to manage display information. It allows to mark current user as available in other to get presences from other users.

  There are some limitations. Look at `documentation`.
- Minor fixes.
- Remove some debug messages.
- Fix issues with presence.
- Added manual ping-pong for Chromium backend.
- Modified *presencemonitor-chromium.py* and *presencemonitor.py* examples in order to get presences permanently.

3.6.8 Version 0.7.1

- Fixed Chromium driver.
- Forced Websockets 6.0
- Added new example *presencemonitor-chromium.py*. It is same than *presencemonitor.py* but using Chromium.

3.6.9 Version 0.7.0

- Added support for Chromium.
- Added support for backends that are able to push results.
- Defined Connection model.
- Added new connection's method *updatePushname*.

- Defined Stream model and enumerations.
- Added Presence manager.
- Added new example: *presencemonitor.py*. It monitor user presences.
- Simplified some code.
- Fields' value are mapped on monitors.
- Profile and group's picture management.
- New chat methods: *pin*, *unpin*, *archive*, *unarchive*, *set_group_description*, *star_messages*, *unstar_messages*, *send_not_spam*, *send_spam_report*, *can_archive*, *can_send* and *can_pin*.
- New chat collection method: *forward_messages_to_chats*.
- New message methods: *can_star*, *star* and *unstar*.
- Status (old one) management.
- New commands in *minibot.py* example: */status* and */pushname*.

3.6.10 Version 0.6.0

- Ability to get message information, it includes message acks (with timestamps). In addition, it is possible to monitor ack changes.

  You must fetch message info before be able to monitor it.

```
msg_info: MessageInfo = await driver.messages[message_id].fetch_info()
```

  And in order to monitor acks (message information changes):

```
async for event in driver.messages[message_id].info.monitor_model():
    print(event)
```

- Added Sticker message.
- Load and send stickers.
- Better type hinting.
- Better documentation.
- Added new command to *minibot.py* example: */sticker*.
- Added new example: *getstickers.py*. It downloads all stickers registered.

3.6.11 Version 0.5.3

- Add *query_exist* method to wap manager in order to get whether a contact indentifier exists or not.
- When send a text message with an url it will try to get link preview and attach to message. It's not compatible with quoted messages.
- Added two new commands to *minibot.py* example: */link* and */exist*.

3.6.12 Version 0.5.2

- Fixed bug when sending docs. Thx to @jabolina.
- Added *set_subject* method to chat manager in order to be able to change group title.
- Added *mark_composing* method to chat manager in order to show "typing..." message.
- Added *mark_recording* method to chat manager in order to show "recording audio..." message.
- Added *mark_paused* method to chat manager in order to remove "typing..." or "recording audio..." message.

3.6.13 Version 0.5.1

- Fixed bug with user chats.
- Added new command */send* to minibot example.

3.6.14 Version 0.5.0

- Added *ensure_chat_with_contact* to chat collection manager. Ensure chat with a whatsapp user, if it does not exist it will be created. (Be careful with SPAM)
- Added *create_group* to chat collection manager.
- Added *block* and *unblock* methods to contact manager.
- Added group participants management: add, remove, promote, demote.
- Added group link management.

3.6.15 Version 0.4.4

- Allow extra options for Firefox driver.
- Added *leave_group*. Thx to @jabolina.
- Added *delete_chat*. Thx to @jabolina.

3.6.16 Version 0.4.0

- Removed *send_vcard* on chats. It is not possible now because WhatsappWeb changes.
- Added *send_contact* and *send_contact_phone* in order to send contacts using contact id or contact name and phone.
- Small changes and refactors.

3.6.17 Version 0.3.0

- Reduce Firefox footprint.
- Message classes.
- Improved getMessages example. Now, it downloads media files.
- Package published at Pypi.

3.6.18 Version 0.2.0

Warning: Command separator changed from . to |.

- Simplified code to manage models.
- Added *remove_item_by_id*, *get_length*, *get_first* and *get_last* methods to collection managers.
- Added *load_earlier_messages* and *load_all_earlier_messages* methods to chat manager.

CHAPTER 4

Related projects

Port of Whalesong to Node.js.

Author: @jabolina

Repository: https://github.com/jabolina/whalesong-js

CHAPTER 5

Indices and tables

- genindex
- modindex
- search

CHAPTER 6

Legal

This code is in no way affiliated with, authorized, maintained, sponsored or endorsed by WhatsApp or any of its affiliates or subsidiaries. This is an independent and unofficial software. Use at your own risk.

Python Module Index

w

Index

Symbols

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W