Common methods

Contents

class **pywa.types.base_update.BaseUpdate**

Base class for all update types.

id: *str*

The ID for the message that was received by the business.

timestamp: *datetime.datetime*

Timestamp indicating when the WhatsApp server received the message from the customer (in UTC).

raw: *dict*

The raw update dict from WhatsApp.

stop_handling() → None

Call this method to break out of the handler loop. other handlers will not be called.

- Use `.continue_handling()` to continue to the next handler in the handlers loop.

This method just raises `StopHandling` which is caught by the handler loop and breaks out of it.

Example

```
>>> from pywa import WhatsApp
>>> from pywa.types import Message
>>> wa = WhatsApp(...)
```

latest

```
>>> @wa.on_message()
... def callback(_: WhatsApp, msg: Message):
...     msg.reply_text("Hello from PyWa!")
...     msg.stop_handling()
```

```
>>> @wa.on_message()
... def callback_not_called(_: WhatsApp, msg: Message):
...     msg.reply_text("This message will not be sent")
```

`continue_handling()` → None

Call this method to continue to the next handler in the handlers loop.

- Use `.stop_handling()` to break out of the handler loop.

This method just raises `ContinueHandling` which is caught by the handler loop and continues the loop.

Example

```
>>> from pywa import WhatsApp
>>> from pywa.types import Message
>>> wa = WhatsApp(...)
```

```
>>> @wa.on_message()
... def callback(_: WhatsApp, msg: Message):
...     msg.reply_text("Hello from PyWa!")
...     msg.continue_handling()
```

```
>>> @wa.on_message()
... def callback_not_called(_: WhatsApp, msg: Message):
...     msg.reply_text("This message will be sent")
```

class `pywa.types.base_update.BaseUserUpdate`

Base class for all user-related update types (message, callback, etc.).

property `sender`*:* *str*

The WhatsApp ID of the sender who sent the message. - Shortcut for `.from_user.wa_id`.

latest

reply_text(*text: str, header: str | None = None, footer: str | None = None, buttons: Iterable[Button] | ButtonUrl | FlowButton | SectionList | None = None, quote: bool = False, preview_url: bool = False, tracker: str | CallbackData | None = None*) → SentMessage

Reply to the message with text.

- Shortcut for `send_message()` with `to` and `reply_to_message_id`.

Example

```
>>> msg.reply_text(
...     text="Hello from PyWa! (https://github.com/david-lev/pywa)",
...     quote=True,
... )
```

Parameters:

- **text** – The text to reply with (markdown allowed, max 4096 characters).
- **header** – The header of the reply (if buttons are provided, optional, up to 60 characters, no markdown allowed).
- **footer** – The footer of the reply (if buttons are provided, optional, up to 60 characters, markdown has no effect).
- **buttons** – The buttons to send with the message (optional).
- **quote** – Whether to quote the replied message (default: False).
- **preview_url** – Whether to show a preview of the URL in the message (if any).
- **tracker** – The data to track the message with (optional, up to 512 characters, for complex data You can use `CallbackData`).

Returns:

The ID of the sent reply.

reply_image(*image: str | pathlib.Path | bytes | BinaryIO, caption: str | None = None, footer: str | None = None, buttons: Iterable[Button] | ButtonUrl | FlowButton | None = None, quote: bool = False, mime_type: str | None = None, tracker: str | CallbackData | None = None*) → SentMessage

Reply to the message with an image.

- Shortcut for `send_image()` with `to` and `reply_to_message_id`.
- Images must be 8-bit, RGB or RGBA.

latest

Example

```
>>> msg.reply_image(
...     image="https://example.com/image.png",
...     caption="This is an image!",
...     quote=True,
... )
```

Parameters:

- **image** – The image to reply (either a media ID, URL, file path, bytes, or an open file object. When buttons are provided, only URL is supported).
- **caption** – The caption of the image (required when buttons are provided, markdown allowed).
- **footer** –

  The footer of the message (if buttons are provided, optional, markdown has no effect).
- **buttons** – The buttons to send with the image (optional).
- **mime_type** – The mime type of the image (optional, required when sending an image as bytes or a file object, or file path that does not have an extension).
- **quote** – Whether to quote the replied message (default: False).
- **tracker** – The data to track the message with (optional, up to 512 characters, for complex data You can use `CallbackData`).

Returns:

The ID of the sent reply.

reply_video(*video: str | pathlib.Path | bytes | BinaryIO, caption: str | None = None, footer: str | None = None, buttons: Iterable[Button] | ButtonUrl | FlowButton | None = None, quote: bool = False, mime_type: str | None = None, tracker: str | CallbackData | None = None*) → SentMessage

Reply to the message with a video.

- Shortcut for `send_video()` with `to` and `reply_to_message_id`.
- Only H.264 video codec and AAC audio codec is supported.
- Videos with a single audio stream or no audio stream are supported.

Example

```
>>> msg.reply_video(
...     video="https://example.com/video.mp4",
...     caption="This is a video",
...     quote=True,
... )
```

latest

Parameters:

- **video** – The video to reply (either a media ID, URL, file path, bytes, or an open file object. When buttons are provided, only URL is supported).
- **caption** –

  The caption of the video (required when sending a video with buttons, markdown allowed).
- **footer** –

  The footer of the message (if `buttons` are provided, optional, markdown has no effect).
- **buttons** – The buttons to send with the video (optional).
- **mime_type** – The mime type of the video (optional, required when sending a video as bytes or a file object, or file path that does not have an extension).
- **quote** – Whether to quote the replied message (default: False).
- **tracker** – The data to track the message with (optional, up to 512 characters, for complex data You can use `CallbackData`).

Returns:

The ID of the sent reply.

reply_audio(*audio: str | pathlib.Path | bytes | BinaryIO, quote: bool = False, mime_type: str | None = None, tracker: str | CallbackData | None = None*) → SentMessage

Reply to the message with an audio.

- Shortcut for `send_audio()` with `to` and `reply_to_message_id`.

Example

```
>>> msg.reply_audio(
...     audio='https://example.com/audio.mp3',
... )
```

Parameters:

- **audio** – The audio file to reply with (either a media ID, URL, file path, bytes, or an open file object).
- **quote** – Whether to quote the replied message (default: False).
- **mime_type** – The mime type of the audio (optional, required when sending a audio as bytes or a file object, or file path that does not have an extension).
- **tracker** – The data to track the message with (optional, up t latest
  for complex data You can use `CallbackData`).

Returns:

The ID of the sent message.

reply_document(*document: str | pathlib.Path | bytes | BinaryIO, filename: str | None = None, caption: str | None = None, footer: str | None = None, buttons: Iterable[Button] | ButtonUrl | FlowButton | None = None, quote: bool = False, mime_type: str | None = None, tracker: str | CallbackData | None = None*) → SentMessage

Reply to the message with a document.

- Shortcut for `send_document()` with `to` and `reply_to_message_id`.

Example

```
>>> msg.reply_document(
...     document="https://example.com/example_123.pdf",
...     filename="example.pdf",
...     caption="Example PDF",
...     quote=True,
... )
```

Parameters:

- **document** – The document to reply (either a media ID, URL, file path, bytes, or an open file object. When buttons are provided, only URL is supported).
- **filename** – The filename of the document (optional, The extension of the filename will specify what format the document is displayed as in WhatsApp).
- **caption** –

  The caption of the document (required when sending a document with buttons, markdown allowed).
- **footer** –

  The footer of the message (if buttons are provided, optional, markdown has no effect).
- **buttons** – The buttons to send with the document (optional).
- **mime_type** – The mime type of the document (optional, required when sending a document as bytes or a file object, or file path that does not have an extension).
- **quote** – Whether to quote the replied message (default: False).
- **tracker** – The data to track the message with (optional, up to 512 characters, for complex data You can use `CallbackData`).

Returns:

The ID of the sent reply.

reply_location(*latitude: float, longitude: float, name: str | None = None, address: str | None = None, quote: bool = False, tracker: str | CallbackData | None = None*) → SentMessage

Reply to the message with a location.

- Shortcut for `send_location()` with `to` and `reply_to_message_id`.

Example

```
>>> msg.reply_location(
...     latitude=37.4847483695049,
...     longitude=−122.1473373086664,
...     name='WhatsApp HQ',
...     address='Menlo Park, 1601 Willow Rd, United States',
... )
```

Parameters:

- **latitude** – The latitude of the location.
- **longitude** – The longitude of the location.
- **name** – The name of the location (optional).
- **address** – The address of the location (optional).
- **quote** – Whether to quote the replied message (default: False).
- **tracker** – The data to track the message with (optional, up to 512 characters, for complex data You can use `CallbackData`).

Returns:

The ID of the sent reply.

reply_location_request(*text: str, quote: bool = False, tracker: str | CallbackData | None = None*) → SentMessage

Reply to the message with a request for the user's location.

- Shortcut for `request_location()` with `to` and `reply_to_message_id`.

Example

```
>>> msg.reply_location_request(
...     text='Please share your location',
... )
```

Parameters:

- **text** – The text to send with the request.
- **quote** – Whether to quote the replied message (default: False).

latest

- **tracker** – The data to track the message with (optional, up to 512 characters, for complex data You can use `CallbackData`).

Returns:

The ID of the sent reply.

reply_contact(*contact: Contact | Iterable[Contact], quote: bool = False, tracker: str | CallbackData | None = None*) → SentMessage

Reply to the message with a contact/s.

- Shortcut for `send_contact()` with `to` and `reply_to_message_id`.

Example

```
>>> from pywa.types import Contact
>>> msg.reply_contact(
...     contact=Contact(
...         name=Contact.Name(formatted_name='David Lev', first_name='Dav
...         phones=[Contact.Phone(phone='1234567890', wa_id='1234567890',
...         emails=[Contact.Email(email='test@test.com', type='WORK')],
...         urls=[Contact.Url(url='https://exmaple.com', type='HOME')],
...     ),
...     quote=True,
... )
```

Parameters:

- **contact** – The contact/s to send.
- **quote** – Whether to quote the replied message (default: False).
- **tracker** – The data to track the message with (optional, up to 512 characters, for complex data You can use `CallbackData`).

Returns:

The ID of the sent reply.

reply_sticker(*sticker: str | pathlib.Path | bytes | BinaryIO, quote: bool = False, mime_type: str | None = None, tracker: str | CallbackData | None = None*) → SentMessage

Reply to the message with a sticker.

- Shortcut for `send_sticker()` with `to` and `reply_to_message_id`.
- A static sticker needs to be 512x512 pixels and cannot exceed 100 KB.
- An animated sticker must be 512x512 pixels and cannot exceed 500 KB

latest

Example

```
>>> msg.reply_sticker(
...     sticker='https://example.com/sticker.webp',
... )
```

Parameters:

- **sticker** – The sticker to reply with (either a media ID, URL, file path, bytes, or an open file object).
- **quote** – Whether to quote the replied message (default: False).
- **mime_type** – The mime type of the sticker (optional, required when sending a sticker as bytes or a file object, or file path that does not have an extension).
- **tracker** – The data to track the message with (optional, up to 512 characters, for complex data You can use `CallbackData`).

Returns:

The ID of the sent reply.

reply_template(*template: Template, quote: bool = False, tracker: str | CallbackData | None = None*) → SentTemplate

Reply to the message with a template.

– Shortcut for `send_template()` with `to` and `reply_to_message_id`.

Example

```
>>> from pywa.types import Template as Temp
>>> wa = WhatsApp(...)
>>> wa.send_template(
...     to='1234567890',
...     template=Temp(
...         name='buy_new_iphone_x',
...         language=Temp.Language.ENGLISH_US,
...         header=Temp.TextValue(value='15'),
...         body=[
...             Temp.TextValue(value='John Doe'),
...             Temp.TextValue(value='WA_IPHONE_15'),
...             Temp.TextValue(value='15%'),
...         ],
...         buttons=[
...             Temp.UrlButtonValue(value='iphone15'),
...             Temp.QuickReplyButtonData(data='unsubscribe_from_marketin
...             Temp.QuickReplyButtonData(data='unsubscribe_from_all_mess
...         ],
...     ),
... )
```

Example for Authentication Template:

```
>>> from pywa.types import Template as Temp
>>> msg.reply_template(
...     template=Temp(
...         name='auth_with_otp',
...         language=Temp.Language.ENGLISH_US,
...         buttons=Temp.OTPButtonCode(code='123456'),
...     ),
...     quote=True
... )
```

Parameters:

- **template** – The template to send.
- **quote** – Whether to quote the replied message (default: False).
- **tracker** – The data to track the message with (optional, up to 512 characters, for complex data You can use `CallbackData`).

Returns:

The ID of the sent reply.

Raises:

`reply_catalog`(*body: str, footer: str | None = None, thumbnail_product_sku: str | None = None, quote: bool = False, tracker: str | CallbackData | None = None*) → SentMessage

Reply to the message with a catalog.

- Shortcut for `send_catalog()` with `to` and `reply_to_message_id`.

Example

```
>>> msg.reply_catalog(
...     body='This is a catalog',
...     footer='Powered by PyWa',
...     thumbnail_product_sku='SKU123',
... )
```

Parameters:

- **body** – Text to appear in the message body (up to 1024 characters).
- **footer** – Text to appear in the footer of the message (optional, up to 60 characters).
- **thumbnail_product_sku** – The thumbnail of this item will be used as the message's header image (optional, if not provided, the first item in the catalog will be used).
- **quote** – Whether to quote the replied message (default: False).

- **tracker** – The data to track the message with (optional, up to 512 characters, for complex data You can use `CallbackData`).

Returns:

The ID of the sent reply.

`reply_product(catalog_id: str, sku: str, body: str | None = None, footer: str | None = None, quote: bool = False, tracker: str | CallbackData | None = None) → SentMessage`

Reply to the message with a product.

- Shortcut for `send_product()` with `to` and `reply_to_message_id`.
- To reply with multiple products, use `reply_products()`.

Parameters:

- **catalog_id** – The ID of the catalog to send the product from. (To get the catalog ID use `get_commerce_settings()` or in the Commerce Manager).
- **sku** – The product SKU to send.
- **body** – Text to appear in the message body (up to 1024 characters).
- **footer** – Text to appear in the footer of the message (optional, up to 60 characters).
- **quote** – Whether to quote the replied message (default: False).
- **tracker** – The data to track the message with (optional, up to 512 characters, for complex data You can use `CallbackData`).

Returns:

The ID of the sent reply.

`reply_products(catalog_id: str, product_sections: Iterable[ProductsSection], title: str, body: str, footer: str | None = None, quote: bool = False, tracker: str | CallbackData | None = None) → SentMessage`

Reply to the message with a product.

- Shortcut for `send_products()` with `to` and `reply_to_message_id`.
- To reply with multiple products, use `reply_products()`.

Example

latest

```
>>> from pywa.types import ProductsSection
>>> msg.reply_products(
...     catalog_id='1234567890',
...     title='Tech Products',
...     body='Check out our products!',
...     product_sections=[
...         ProductsSection(
...             title='Smartphones',
...             skus=['IPHONE12', 'GALAXYS21'],
...         ),
...         ProductsSection(
...             title='Laptops',
...             skus=['MACBOOKPRO', 'SURFACEPRO'],
...         ),
...     ],
...     footer='Powered by PyWa',
...     quote=True,
... )
```

Parameters:

- **catalog_id** –

  The ID of the catalog to send the product from (To get the catalog ID use `get_commerce_settings()` or in the Commerce Manager).
- **product_sections** – The product sections to send (up to 30 products across all sections).
- **title** – The title of the product list (up to 60 characters).
- **body** – Text to appear in the message body (up to 1024 characters).
- **footer** – Text to appear in the footer of the message (optional, up to 60 characters).
- **quote** – Whether to quote the replied message (default: False).
- **tracker** – The data to track the message with (optional, up to 512 characters, for complex data You can use `CallbackData`).

Returns:

The ID of the sent reply.

react(*emoji: str, tracker: str | CallbackData | None = None*) → SentMessage

React to the message with an emoji.

- Shortcut for `send_reaction()` with `to` and `message_id`.

Example

```
>>> msg.react('👍')
```

latest

Parameters:

- **emoji** – The emoji to react with.
- **tracker** – The data to track the message with (optional, up to 512 characters, for complex data You can use `CallbackData`).

Returns:

The ID of the sent reaction.

`unreact(tracker: str | CallbackData | None = None) → SentMessage`

Remove the reaction from the message.

- Shortcut for `remove_reaction()` with `to` and `message_id`.

Example

```
>>> msg.unreact()
```

Parameters:

tracker – The data to track the message with (optional, up to 512 characters, for complex data You can use `CallbackData`).

Returns:

The ID of the sent unreaction.

`mark_as_read() → bool`

Mark the message as read.

- Shortcut for `mark_message_as_read()` with `message_id`.

Returns:

Whether it was successful.

`indicate_typing() → bool`

Mark the message as read and display a typing indicator so the WhatsApp user knows you are preparing a response. This is good practice if it will take you a few seconds to respond.

The typing indicator will be dismissed once you respond, or after 25 seconds, whichever comes first. To prevent a poor user experience, only display a typing indicator if you are going to respond.

latest

- Shortcut for `indicate_typing()` with `message_id`.

Returns:

Whether it was successful.

property **recipient**: *str*

The WhatsApp ID which the message was sent to. - Shortcut for `.metadata.phone_number_id`.

block_sender() → bool

Block the sender of the message.

- Shortcut for `block_users()` with `sender`.

unblock_sender() → bool

Unblock the sender of the message.

- Shortcut for `unblock_users()` with `sender`.

property **message_id_to_reply**: *str*

The ID of the message to reply to.

If you want to `wa.send_x` with `reply_to_message_id` in order to reply to a message, use this property instead of `id` to prevent errors.