Webhooks

Webhooks allow you to send real-time notifications of events within tawk.to to external services.

You can configure a webhook to be sent on the following events:

- Chat starts
- Chat ends
- New chat transcripts
- New ticket is created

A webhook consists of:

- A URL you have configured, to which a webhook event will be posted

- One or more events, which will be posted to a specified URL
- A secret key, which can be used to verify a webhook payload was sent by tawk.to

When a webhook is triggered, a POST request will be made to the URL configured along with a JSON payload specific for the event type.

Reference:

Managing webhooks

Webhooks can be managed via the tawk.to dashboard in the `Admin` section.

Currently supported events

Callback functions are invoked when the page status changes. The function will receive the changed status, which will be online, away or offline. This callback is not supported in pop out chat window.

Chat:

- Start
- End
- Transcript

Ticket:

- Create

Retry policy

A webhook call will be retried for up to 12 hours if the http endpoint responds with anything but a success (2XX). If no response is received within 30 seconds, the call will be considered a failure and will also be reattempted. Each webhook event is assigned an event Id. The event Id is passed along with each request in the headers as X-Hook-Event-Id. The event Id will remain the same in case of retries.

Verifying webhook signature

Each webhook event is signed via a Hash-based Message Authentication Code (HMAC) using the webhooks secret key. The HMAC-SHA1 algorithm is used to generate the webhook payload signature. The signature is passed along with each request in the headers as 'X-Tawk-Signature.'

Examples:

Node.js

```
const WEBHOOK_SECRET = 'webhook secret key';
const crypto = require('crypto');
const bodyParser = require('body-parser');

function verifySignature (body, signature) {
    const digest = crypto
        .createHmac('sha1', WEBHOOK_SECRET)
        .update(body)
        .digest('hex');
    return signature === digest;
};
app.use(bodyParser.json({
    type: 'application/json',
    verify: function (req, res, buf) {
        req.rawBody = buf;
    }
});
app.post('/webhooks', function (req, res, next) {
    if (!verifySignature(req.rawBody, req.headers['x-tawk-
signature'])) {
        // verification failed
    }
    // verification success
});
```

PHP

```
const WEBHOOK_SECRET = 'webhook secret key';
function verifySignature ($body, $signature) {
    $digest = hash_hmac('sha1', $body, WEBHOOK_SECRET);
    return $signature === $digest ;
}
if (!verifySignature(file_get_contents('php://input'),
$_SERVER['HTTP_X_TAWK_SIGNATURE'])) {
    // verification failed
}
// verification success
```

Ruby

```
WEBHOOK_SECRET = 'webhook secret key';
post '/payload' do
    request.body.rewind
    body = request.body.read
    signature = request.env['HTTP_X_TAWK_SIGNATURE']
    unless verifySignature(body, signature))
        // verification failed
    end
    // verification success
end
def verifySignature(body, signature)
    digest = OpenSSL::HMAC.hexdigest(OpenSSL::Digest.new('sha1'),
WEBHOOK_SECRET, body)
    return digest == signature
end
```

Webhook event payload

Chat start event

Generated when the first message in a chat is sent by visitor or agent. Whispers and system notifications like triggers do not generate this event.

Chat start event payload

Property	Type	Description
event	String	Event name `chat:start`
chatId	String	Chat Id

time	String	Event generation date time in JSON format
message	Object	Message object (See below)
visitor	Object	Visitor object (see below)
property	Object	Property object (see below)

Message object

Property	Type	Description
text	String	Textual representation of message
type	String	Message type, enum value **msg, file, webrtc-call**
sender	Object	Sender info object (See below)

Sender info object

Property	Type	Description
type	String	Sender type, enum value **agent, visitor, system**

Visitor object

Property	Type	Description
name	String	Visitor name
email	String	Visitor email (Optional)
city	String	Visitor city

country	String	Visitor country in ISO 3166 format

Property object

Property	Type	Description
id	String	Property Id
name	String	Property name

Example

```
{
    event: 'chat:start',
    chatId: '70fe3290-99ad-11e9-a30a-51567162179f',
    time: '2019-06-28T14:03:04.646Z',
    message : {
        text : 'Sample message',
        type : 'msg',
        sender : {
            type : 'visitor'
        }
    },
    visitor: {
        name: 'V1561719148780935',
        email : 'hello@test.com',
        city: 'jelgava',
        country: 'LV'
    },
    property: {
        id: '58ca8453b8a7e060cd3b1ecb',
        name: 'Bobs Burgers'
    }
}
```

Chat end event

Generated when a chat has ended

Chat end event payload

Property	Type	Description
event	String	Event name `chat:end`
chatId	String	Chat Id
time	String	Event generation date time in JSON format
visitor	Object	Visitor object (see below)
property	Object	Property object (see below)

Visitor object

Property	Type	Description
name	String	Visitor name
email	String	Visitor email (Optional)
city	String	Visitor city
country	String	Visitor country in ISO 3166 format

Property object

Property	Type	Description
id	String	Property Id

name	String	Property name

Example

```
{
    event: 'chat:end',
    chatId: '70fe3290-99ad-11e9-a30a-51567162179f',
    time: '2019-06-28T14:04:08.718Z',
    visitor: {
        name: 'V1561719148780935',
        email : 'hello@test.com',
        city: 'jelgava',
        country: 'LV'
    },
    property: {
        id: '58ca8453b8a7e060cd3b1ecb',
        name: 'Bobs Burgers'
    }
}
```

Chat transcript created event

Generates full transcript of chats in a session after it has ended

Chat transcript event payload

Property	Type	Description
event	String	Event name `chat:transcript_created`
time	String	Event generation date time in JSON format
property	Object	Property object (see below)
chat	Object	Chat object (see below)

Property object

Property	Type	Description
id	String	Property Id
name	String	Property name

Chat object

Property	Type	Description
id	String	Chat Id
visitor	Object	Visitor object (see below)
messages	Object	List of messages (see below)

Visitor object

Property	Type	Description
name	String	Visitor name
email	String	Visitor email (Optional)
city	String	Visitor city
country	String	Visitor country in ISO 3166 format

Message object

Property	Type	Description

sender	Object	Sender object (see below)
type	String	Message type
msg	String	Text message
time	String	Message sent time
attchs	Object	List of attachments (see below)

Sender object

Property	Type	Description
t	String	Sender type (a - agent, v - visitor, s - system)
n	String	Sender name (sender type 'agent' or 'system')
id	String	Sender ID (sender type 'agent')

Attachment object

Property	Type	Description
type	String	Message type
content	Object	Attachment content
content.file	Object	Attachment file
content.file.url	String	File URL
content.file.name	String	File name
content.file.mimeType	String	File MIME type

content.file.size	Number	File size
content.file.extension	String	File extension

Example

```
{
    event: 'chat:transcript_created',
    time: '2024-07-03T01:02:37.780Z',
    chat: {
        id: '70fe3290-99ad-11e9-a30a-51567162179f',
        visitor: {
            name: 'V1561719148780935',
            email : 'hello@test.com',
            city: 'jelgava',
            country: 'LV'
        },
        messages: [
            {
                sender: {
                    t: 's',
                    n : 'Customer Support'
                },
                type: 'msg',
                msg : '👋 Hi! How can we help?\n[option]I have a
question\n[option]Tell me more',
                time: '2024-07-03T01:02:37.780Z'
            },
            {
                sender: {
                    t: 'v'
                },
                type: 'msg',
                msg : 'Tell me more',
                time: '2024-07-03T01:02:48.176Z'
            },
            {
                sender: {
                    t: 'a',
                    n : 'Eugene',
                    id : '666668688fe587223fdf8685'
                },
                type: 'msg',
                msg : 'msg',
                time: '2024-07-03T01:02:37.780Z',
                attchs: [
                    {
                        type: 'file',
```

```
                        content: {
                            file: {
                                url:
'https://tawkto.link/66666880e73507189f888c92/files/message/2qTwJwVVC8/taw
                                name : 'tawky_big.png',
                                mimeType: 'image/png',
                                size: 3797,
                                extension: 'png',
                            }
                        }
                    }
                ]
            }
        ]
    },
    property: {
        id: '58ca8453b8a7e060cd3b1ecb',
        name: 'Bobs Burgers'
    }
}
```

Ticket created event

Generated when a new ticket is created.

Ticket create event payload

Property	Type	Description
event	String	Event name `ticket:create`
time	String	Event generation date time in JSON format
requester	Object	Ticket requester object (see below)
property	Object	Property object (see below)
ticket	Object	Ticket object (see below)

Ticket requester object

Property	Type	Description
name	String	Visitor name
email	String	Visitor email

Property object

Property	Type	Description
id	String	Property Id
name	String	Property name

Ticket object

Property	Type	Description
id	String	Ticket Id
humanId	Number	Ticket human Id
subject	String	Ticket subject
message	String	Ticket message content

Example

```
{
    event: 'ticket:create',
    time: '2019-06-28T14:07:13.512Z',
    property: {
        id: '58ca8453b8a7e060cd3b1ecb',
        name: 'Bobs Burgers'
    },
    requester: {
        name: 'Martins',
        email: 'martins@tawk.to',
        type: 'agent'
    },
    ticket: {
        id: '02598050-99ae-11e9-8887-97564881b95b',
        humanId: 3,
        subject: 'Testing',
        message: 'Once more through the breach'
    }
}
```


Visit tawk.to (https://www.tawk.to/)