RP3 Woensdag 9 dec 2020

**Pushover Message API**

Pushover uses a simple, versioned [REST](https://en.wikipedia.org/wiki/REST) API to receive messages and broadcast them to devices running our [device clients](https://pushover.net/clients).

[**https://pushover.net/clients**](https://pushover.net/clients)

1. Android
2. IOS
3. Desktop
4. 7-day trial
5. Teams

## Pushover-Powered Applications & Plugins

## Email Gateway

## IFTTTEbay

## Price watcher

## Dopmoticz

## Wordpress

**Register your application Separate Licence for Android – IOS – PC ( 5 USD elk, eenmalig )**

<https://pushover.net/signup>

### **Your User Key umpvs3iq2dvci87nkn7axg1u**

To receive notifications from a Pushover-powered [application](https://pushover.net/apps), service, or website, just supply your user key:

  
To receive Pushover notifications from e-mails, send to:



**Help https://support.pushover.net/**

**Critical alerts!!!!!!!!!!!!!!!**

Hi, you can force a sound if the device has enabled the "Play high priority through alarm channel" setting on Android or "Critical alerts for high priority" on iOS, and your message is being sent with a priority of 1 or 2.

Messages sent with a priority of 2 repeat for the interval and duration that you specify.

<https://pushover.net/api#priority>

<https://blog.pushover.net/posts/2020/2/ios-critical-alerts>

#### **Emergency Priority (2)**

Emergency-priority notifications are similar to high-priority notifications, but they are repeated until the notification is acknowledged by the user. These are designed for dispatching and on-call situations where it is critical that a notification be repeatedly shown to the user (or all users of the group that the message was sent to) until it is acknowledged. The first user in a group to acknowledge a message will cancel retries for all other users in the group.

Applications sending emergency notifications are issued a [receipt](https://pushover.net/api#receipt) that can be used to get the status of a notification and find out whether it was acknowledged, or automatically receive a callback when the user has acknowledged the notification.

To send an emergency-priority notification, the priority parameter must be set to 2 and the retry and expire parameters must be supplied.

The retry parameter specifies how often (in seconds) the Pushover servers will send the same notification to the user. In a situation where your user might be in a noisy environment or sleeping, retrying the notification (with sound and vibration) will help get his or her attention. This parameter must have a value of at least 30 seconds between retries.

The expire parameter specifies how many seconds your notification will continue to be retried for (every retry seconds). If the notification has not been acknowledged in expire seconds, it will be marked as expired and will stop being sent to the user. Note that the notification is still shown to the user after it is expired, but it will not prompt the user for acknowledgement. This parameter must have a maximum value of at most 10800 seconds (3 hours).

For example, sending a retry parameter of 60 and an expire parameter of 3600 will cause your notification to be retried every 60 seconds for 1 hour.

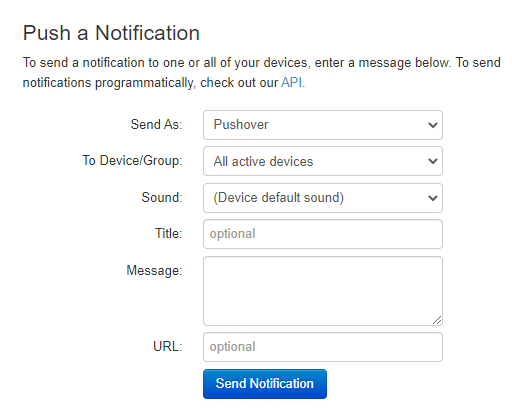
The optional callback parameter may be supplied with a publicly-accessible URL that our servers will send a request to when the user has acknowledged your notification.

When your application sends an emergency-priority notification, our API will respond with a receipt value that can be used to get information about whether the notification has been acknowledged. See our [receipts and callbacks](https://pushover.net/api/receipts) section for more information.

If your application is not capable of storing receipt identifiers, you can send a tags parameter of comma-separated, arbitrary tags which will be stored with the receipt on our servers. You can then use the [cancel\_by\_tag](https://pushover.net/api/receipts) API call to cancel all receipts with a specific tag.

**Eerste test via Pushover platform**

<https://pushover.net/>



**Tweede test , zend een email**

cacqk6kswzj@pomail.net

token=azGDORePK8gMaC0QYAMyEEuzJnyUi&user=uQiRzpo4DXghDmr9Qzzfu27cmVRsG&device=droid4&title=Backup+finished+-+SQL1&message=Backup+of+database+%22example%22+finished+in+16+minutes.

That message would appear like this in the Pushover client on an Android device:

**token=ahbf2c758jxe2qgxhrs7z8aj6ip6r&user=umpvs3iq2dvci87nk7axg1u3ubtgk&title=living&message=alarm&priority=2&retry=30&expire=120&sound=persistent**

Content-Type : application/x-www-form-urlencoded

**Example code and Pushover libraries**

<https://support.pushover.net/i44-example-code-and-pushover-libraries>

Pushover's API is designed to be easy to use without the overhead of complicated authentication mechanisms like OAuth.  For full details on our API endpoints, see our [API Documentation](https://pushover.net/api).

To quickly get started sending Pushover messages with a wide variety of programming languages, we have some sample code as well as a number of 3rd party libraries written and maintained by Pushover users.

In all of these examples, replace ***USER\_KEY*** with your Pushover User Key (which can be found on [your dashboard](https://pushover.net/dashboard)) or that of the user/group to which you are sending, and ***API\_TOKEN*** with your application's API Token.  If you don't already have an application API Token, you can [create one for free](https://pushover.net/api#registration).

**Note**: if your language of choice is not listed here, try [browsing all GitHub libraries tagged "pushover"](https://github.com/topics/pushover).

* [C](https://support.pushover.net/i44-example-code-and-pushover-libraries#c)
* [C#/.NET](https://support.pushover.net/i44-example-code-and-pushover-libraries#csharp)
* [Go](https://support.pushover.net/i44-example-code-and-pushover-libraries#go)
* [Haskell](https://support.pushover.net/i44-example-code-and-pushover-libraries#haskell)
* [Java](https://support.pushover.net/i44-example-code-and-pushover-libraries#java)
* [Lua](https://support.pushover.net/i44-example-code-and-pushover-libraries#lua)
* [Node](https://support.pushover.net/i44-example-code-and-pushover-libraries#node)
* [Perl](https://support.pushover.net/i44-example-code-and-pushover-libraries#perl)
* [PHP](https://support.pushover.net/i44-example-code-and-pushover-libraries#php)
* [PHP with Image Attachment](https://support.pushover.net/i44-example-code-and-pushover-libraries#php-image)
* [Python](https://support.pushover.net/i44-example-code-and-pushover-libraries#python)
* [Python with Image Attachment](https://support.pushover.net/i44-example-code-and-pushover-libraries#python-image)
* [R](https://support.pushover.net/i44-example-code-and-pushover-libraries#r)
* [Ruby](https://support.pushover.net/i44-example-code-and-pushover-libraries#ruby)
* [Rust](https://support.pushover.net/i44-example-code-and-pushover-libraries#rust)
* [Visual Basic for Applications (VBA)](https://support.pushover.net/i44-example-code-and-pushover-libraries#vba)
* [Unix command line](https://support.pushover.net/i44-example-code-and-pushover-libraries#unix)
* [Unix command line with Image Attachment](https://support.pushover.net/i44-example-code-and-pushover-libraries#unix-image)
* [Windows command line](https://support.pushover.net/i44-example-code-and-pushover-libraries#windows)

## Pushover Message API

Pushover uses a simple, versioned [REST](https://en.wikipedia.org/wiki/REST) API to receive messages and broadcast them to devices running our [device clients](https://pushover.net/clients). To simplify the user registration process and usage of our API, there are no complicated out-of-band authentication mechanisms or per-call signing libraries required, such as OAuth. Standard HTTP libraries available in just about every language, or even from the command line, can be used without any custom modules or extra dependencies needed. See our [Knowledge Base](https://support.pushover.net/i44) for examples in different programming languages.

To simplify the process of collecting user keys when sending to multiple users, we have a new [Subscription API](https://pushover.net/api/subscriptions) available.

1. [Register your application](https://pushover.net/apps/build), set its name and upload an icon, and get an API token in return (often referred to as APP\_TOKEN in our documentation and [code examples](https://support.pushover.net/i44)).
2. POST an HTTPS request to https://api.pushover.net/1/messages.json with the following parameters:
   * token (required) - your application's API token
   * user (required) - the user/group key (not e-mail address) of your user (or you), viewable when logged into [our dashboard](https://pushover.net/dashboard) (often referred to as USER\_KEY in our documentation and [code examples](https://support.pushover.net/i44))
   * message (required) - your message

Some optional parameters may be included:

* + attachment - an image attachment to send with the message; see [attachments](https://pushover.net/api#attachments) for more information on how to upload files
  + device - your user's device name to send the message directly to that device, rather than all of the user's devices (multiple devices may be separated by a comma)
  + title - your message's title, otherwise your app's name is used
  + url - a [supplementary URL](https://pushover.net/api#urls) to show with your message
  + url\_title - a title for your supplementary URL, otherwise just the URL is shown
  + priority - send as -2 to generate no notification/alert, -1 to always send as a quiet notification, 1 to display as [high-priority](https://pushover.net/api#priority) and bypass the user's quiet hours, or 2 to also require confirmation from the user
  + sound - the name of one of the [sounds](https://pushover.net/api#sounds) supported by device clients to override the user's default sound choice
  + timestamp - a Unix timestamp of your message's date and time to display to the user, rather than the time your message is received by our API

That's it. Make sure your application is [friendly to our API servers](https://pushover.net/api#friendly) and you're all set. For more information on each parameter, keep reading or jump to a section at the left.

Python 3

import requests

url = 'https://api.pushover.net/1/messages.json'

data\_to\_send = "token=ahbf2c758j9xe2qgxh7z8aj6ip6r&user=umpvs3iq2dvc7nkn7axg1u3ubtgk&title=kot&message=alarm&priority=2&retry=30&expire=60&sound=persistent"

x = requests.post(url, data = data\_to\_send, headers = {"Content-Type": "application/x-www-form-urlencoded"})

print(x.text)

of

import http.client, urllib

conn = http.client.HTTPSConnection("api.pushover.net:443")

conn.request("POST", "/1/messages.json",

urllib.parse.urlencode({

"token": "APP\_TOKEN",

"user": "USER\_KEY",

"message": "hello world",

}), { "Content-type": "application/x-www-form-urlencoded" })

conn.getresponse()

 Alternatively, a number of 3rd party packages are available:  [pushnotify](http://pypi.python.org/pypi/pushnotify) package written by Jeffrey Goettsch, [pushover](https://github.com/Wyattjoh/pushover) Python 3 package written by Wyatt Johnson, [Chump](http://chump.readthedocs.org/en/latest/) Python package written by Karan Lyons, and [python-pushover](https://github.com/Thibauth/python-pushover) Python package written by Thibaut Horel.

### **Python with Image Attachment**

To send an [image attachment](https://pushover.net/api#attachments), the 3rd party [Requests](http://docs.python-requests.org/en/master/) library can be used. For example, to send the file "your\_image.jpg":

import requests

r = requests.post("https://api.pushover.net/1/messages.json", data = {

"token": "APP\_TOKEN",

"user": "USER\_KEY",

"message": "hello world"

},

files = {

"attachment": ("image.jpg", open("your\_image.jpg", "rb"), "image/jpeg")

})

print(r.text)

## Receipts and Callbacks API

Applications sending [emergency-priority](https://pushover.net/api#priority) notifications will receive a receipt parameter from our API when a notification has been queued. This parameter is a 30 character string containing the character set [A-Za-z0-9]. This receipt can be used to periodically poll our receipts API to get the status of your notification, up to 1 week after your notification has been received.

Submit a GET request (no faster than once every 5 seconds) to:

https://api.pushover.net/1/receipts/(your receipt).json?token=(your app token)

Include the receipt in the URL and your application's token as the token parameter. If the receipt is valid, our API will respond with some integer-valued parameters about your notification:

status = 1 (indicating your receipt request was successful)

acknowledged = 1 (1 or 0 whether the user has acknowledged the notification)

acknowledged\_at = 1360019238 (a Unix timestamp of when the user acknowledged, or 0)

acknowledged\_by = (user key) (the user key of the user that first acknowledged the notification)

acknowledged\_by\_device = (device name) (the device name of the user that first acknowledged the notification)

last\_delivered\_at = 1360001238 (a Unix timestamp of when the notification was last retried, or 0)

expired = 1 (1 or 0 whether the expiration date has passed)

expires\_at = 1360019290 (a Unix timestamp of when the notification will stop being retried)

called\_back = 1 (1 or 0 whether our server has called back to your callback URL if any)

called\_back\_at = 1360019239 (a Unix timestamp of when our server called back, or 0)

### **Callback URLs**

Rather than periodically polling our receipts API, you may also include a callback parameter when submitting your emergency notification. This must be a URL (HTTP or HTTPS) that is reachable from the Internet that our servers will call out to as soon as the notification has been acknowledged. If a callback URL exists, we will submit a POST request to the URL with the following parameters:

receipt = (your receipt ID)

acknowledged = 1

acknowledged\_at = 1360019238 (a Unix timestamp of when the notification was acknowledged)

acknowledged\_by = (user key) (the user key of the user that first acknowledged the notification)

acknowledged\_by\_device = (device name) (the device name of the user that first acknowledged the notification)

If our API servers do not receive a successful (2xx) HTTP response from your callback URL, we will retry again in one minute.

### **Canceling Emergency-Priority Retries**

An [emergency-priority](https://pushover.net/api#priority2) notification will continue to be sent to devices until it reaches its original expire value. To cancel an emergency-priority notification early, you can send a POST request to our API:

https://api.pushover.net/1/receipts/(your receipt)/cancel.json

Along with your POST request, include your application's API token as the token parameter.

### **Canceling Emergency-Priority Retries By Tag**

[Canceling](https://pushover.net/api/receipts#cancel) retries normally requires a receipt ID which is returned by our API when the message is [created](https://pushover.net/api) and must be stored by your application. If your application is not capable of storing receipt IDs or you want to cancel a handful of similar messages (possibly sent to different recipients) all at once, you can use tags.

When [creating](https://pushover.net/api#priority2) an emergency-priority message, a tags parameter may be supplied which stores a comma-separated list of arbitrary tags with the message. Any of these tags may be used to cancel all emergency-priority messages with that tag that were sent by your application.

For example, a network monitor application sending a "down" event might include tags of the affected server, rack, and location.

message = Server mail01 failed to respond to 3 pings

tags = s=mail01,r=23,l=chicago

If many messages for down servers were sent but later correlated to a single datacenter outage, the network monitor may generate an "up" event that cancels all retries for messages with the l=chicago tag.

To cancel retries for all messages with a given tag, submit a POST request to:

https://api.pushover.net/1/receipts/cancel\_by\_tag/(your tag).json

Along with your POST request, include your application's API token as the token parameter. All active emergency-priority messages with the given tag will be canceled.

## Pushover Open Client API

Pushover now features a public client API for advanced users to write their own native clients to receive push notifications like our official [mobile and desktop clients](https://pushover.net/clients). While this API is in beta, there may be small changes to server responses or recommended behavior, but there will most likely not be any large backwards-incompatible changes.

Licensing for Open Clients is done as a [desktop device](https://pushover.net/clients/desktop), so a user must have or purchase a Pushover for Desktop license within 5 days of activation of an Open Client device to continue using it.

Before creating (**and especially before distributing**) your own client, please see our [distribution guidelines](https://pushover.net/api/client#distribution).

### **TL;DR**

1. [Login](https://pushover.net/api/client#login) with a Pushover account e-mail address and password, then securely store the session secret.
2. [Register your device](https://pushover.net/api/client#register) and store its uuid.
3. [Download all outstanding messages](https://pushover.net/api/client#download) and then [delete them](https://pushover.net/api/client#delete) from our servers. In most cases you will not want to generate notifications for these old messages.
4. Establish a [websocket](https://pushover.net/api/client#websocket) connection to our servers and listen for notification of a new message.
5. When new messages are pending, [download them](https://pushover.net/api/client#download), do some action with them such as generate a local notification, and then [delete them](https://pushover.net/api/client#delete) from our servers.

### **User Login**

All API actions that deal with device clients require a user session secret, which is a randomly generated token assigned at login time and is unique to each login. A session secret should not be re-used between devices.

If the user does not yet have a Pushover account, direct them to <https://pushover.net/signup>. Once the user has account, you can retrieve the user's Pushover key and a secret by submitting a POST request to our users/login endpoint:

https://api.pushover.net/1/users/login.json

Include the email and password parameters for the user you are logging in as. Both values are case-sensitive. If the login was successful, you will receive a JSON object with status set to 1 (as with all of our API functions), the user's Pushover user key (id), and the user's session secret. Store these values in a secure manner and discard the password.

$ curl \

--form-string "email=user@example.com" \

--form-string "password=hunter2" \

https://api.pushover.net/1/users/login.json

{"status":1,

"request":"7df577c3-da18-4fb3-898b-c1ab4985633b",

"id":"uQiRzpo4DXghDmr9QzzfQu27cmVRsG",

"secret":"SGx2Su5onMcXU2EVozWG41Fws42bHo0aOrmA3tQ3jjRMSu1HwMEmOWNWPD7J"}

If the response has an HTTP status 412 (the status in the JSON response will be 0), the user's account requires two-factor authentication to login. Prompt the user for their current two-factor authentication code and submit the login request again with the previous email and password parameters, in addition to the user's two-factor code as the twofa parameter.

### **Device Registration**

Once you have the user's session secret, you can register a new Desktop device by submitting a POST request to our devices endpoint:

https://api.pushover.net/1/devices.json

Include the user's secret, the short name of the device ([A-Za-z0-9\_-], up to 25 characters long), and an os value of O for Open Client. If the device creation was successful, you will receive a JSON object with the device's unique id. Store this value in a secure location.

$ curl \

--form-string "secret=SGx2Su5onMcXU2EVozWG41Fws42bHo0aOrmA3tQ3jjRMSu1HwMEmOWNWPD7J" \

--form-string "name=my\_device" \

--form-string "os=O" \

https://api.pushover.net/1/devices.json

{"status":1,

"request":"5e400a78-127b-4078-85e7-8f63cc78c9b2",

"id":"zQie8WjzFTWkMz5CcGrUNK2t5rR9zGTsfYQ7HHGs"}

If registration of the device failed for any reason (such as the device name being invalid or in use already), you will get a status of 0 and an errors object detailing each problem.

{"status":0,

"request":"ffb7a544-befe-4f46-b901-e15c4e399e5f",

"errors":{"name":["has already been taken"]}}

### **Message Downloading**

When your client starts up, it should download all existing messages waiting for the device. If your client has not been run in a while, these messages are probably old so you will most likely want to discard them rather than spamming the user with a bunch of notifications at startup.

To download all pending messages for a device, submit a GET request to our messages endpoint:

https://api.pushover.net/1/messages.json

Include the user secret and device\_id parameters in the GET request, and a JSON object will be returned with all of the device's messages.

$ curl "https://api.pushover.net/1/messages.json?secret=SGx2Su5onMcXU2EVozWG41Fws42bHo0aOrmA3tQ3jjRMSu1HwMEmOWNWPD7J&device\_id=zQie8WjzFTWkMz5CcGrUNK2t5rR9zGTsfYQ7HHGs"

{"status":1,

"request":"36fddffb-9f62-444b-bd17-5e7c7febb258",

"messages":[

{"id":380698801670733826,"id\_str":"380698801670733826","message":"This message confirms that you are now able to receive messages on this device (my\_device).\n\nVisit https://pushover.net/apps to view apps, plugins, and services to use with Pushover.","app":"Pushover","aid":380698715771387905,"aid\_str":"380698715771387905","icon":"HopmnR5uQ4cmXen","date":1409605784,"priority":0,"acked":0,"umid":380698801670733826,"umid\_str":"380698801670733826","title":"Welcome to Pushover!"},

{"id":380698969174458372,"id\_str":"380698969174458372","message":"test","app":"Pushover","aid":380698715771387905,"aid\_str":"380698715771387905","icon":"default","date":1409605795,"priority":0,"acked":0,"umid":380699115203346437,"umid\_str":"380699115203346437","title":""}

]}

A successful message download will have a status code of 1 and a messages array. Each object in the messages array will include at least the following fields, though others may be present which can be ignored:

id - The unique id of the message, relative to this device. A string value of this 64-bit integer is also available as id\_str.

umid - The unique id of the message relative to all devices on the same user's account. When a message is received by our API and sent to all devices on a user's account, each message is given the same umid value. This is primarily used for cross-device notification dismissal sync. A string value of this 64-bit integer is also available as umid\_str.

title - The title of the message, if present. If not present, the name of the application (app) should be displayed.

message - The text of the message.

app - The name of the [application](https://pushover.net/apps) that sent the message. This may not be unique.

aid - The unique id of the [application](https://pushover.net/apps) that sent the message. A string value of this 64-bit integer is also available as aid\_str.

icon - The icon name of the [application](https://pushover.net/apps) that sent the message. The image data can be fetched at https://api.pushover.net/icons/<icon name>.png. When an application changes its icon, this value will change.

date - A Unix timestamp of the time the message was received by our API, unless the sender [overrode](https://pushover.net/api#timestamp) the timestamp when sending it to us.

priority - The [priority](https://pushover.net/api#priority) of the message.

sound - If a message specified a particular sound, this two-character value will point to the file located at https://api.pushover.net/sounds/<sound file>.mp3 (each sound is also available as a .wav). Your client should fetch this resource, play it, and cache the file.

url - If a message specified a [supplementary URL](https://pushover.net/api#urls), it will be included here.

url\_title - If a message specified a [supplementary URL title](https://pushover.net/api#urls), it will be included here.

acked - Whether the message was acknowledged. Used for [Emergency-Priority messages](https://pushover.net/api/client#p2).

receipt - For [Emergency-Priority messages](https://pushover.net/api/client#p2), this will be the message's receipt code.

html - Whether message contains HTML (1) or plain text.

### **Message Deleting**

After you have successfully downloaded all outstanding messages from our server, stored them, processed them, and done whatever action you need to with them (such as generating a local notification), you should delete those messages from our server. This is done by submitting a POST request to update\_highest\_message with the device id in the URL:

https://api.pushover.net/1/devices/<device id>/update\_highest\_message.json

In addition to the device id in the URL, the POST parameters must include the user's secret and the message POST parameter set to the highest message id value that you saw in the previous message download. Once this responds with a status of 1, all messages for that device up to and including the supplied message id will have been permanently deleted.

$ curl \

--form-string "secret=SGx2Su5onMcXU2EVozWG41Fws42bHo0aOrmA3tQ3jjRMSu1HwMEmOWNWPD7J" \

--form-string "message=2" \

https://api.pushover.net/1/devices/zQie8WjzFTWkMz5CcGrUNK2t5rR9zGTsfYQ7HHGs/update\_highest\_message.json

{"status":1,

"request":"040c63bc-e4b2-4496-b767-db37be2edb7b"}

### **Real-Time Message Notification**

Now that your client has downloaded and deleted all outstanding messages, you can now connect to our [WebSocket](https://en.wikipedia.org/wiki/WebSocket) server and listen for real-time notification of new messages. WebSocket is a relatively new and complex protocol that "upgrades" an existing HTTPS connection to our web servers, enabling a full-duplex frame-based communication mode.

Please note that our web servers' primary compatibility will be for the major web browsers Firefox, Chrome, and Safari, to work with our own [Desktop client](https://pushover.net/clients/desktop). If your client is unable to establish or maintain a WebSocket connection to our web servers due to incompatibilities with your WebSocket implementation, we will most likely not be able to make any accommodations on our web servers.

To start listening for new messages, establish a secure WebSocket connection over HTTPS to our web server at:

wss://client.pushover.net/push

Once your WebSocket connection has been established, you must login with the user's secret and the Open Client device's id. This is done by sending "login:", followed by the device's id, followed by a colon (:), followed by the user's secret, and then terminated by a newline character (ASCII 10).

login:zQie8WjzFTWkMz5CcGrUNK2t5rR9zGTsfYQ7HHGs:SGx2Su5onMcXU2EVozWG41Fws42bHo0aOrmA3tQ3jjRMSu1HwMEmOWNWPD7J\n

If your login was successful, your connection will remain open and you will begin receiving data. If your login failed, you will receive an error frame (detailed below) and your connection will be closed.

Our WebSocket data protocol is very compact, intended to save bandwidth while preventing overly aggressive firewalls from closing seemingly-idle connections. While you are connected to our WebSocket server, you will receive frames with one of the following single-byte characters:

* # - Keep-alive packet, no response needed.
* ! - A new message has arrived; you should perform a [sync](https://pushover.net/api/client#download).
* R - Reload request; you should drop your connection and re-connect.
* E - Error; a permanent problem occured and you should not automatically re-connect. Prompt the user to login again or re-enable the device. A - Error; the device logged in from another session and this session is being closed. Do not automatically re-connect.

When your WebSocket connection receives a ! frame indicating that new messages are available, your client should open a new, secondary HTTPS connection to our API to do a [download](https://pushover.net/api/client#download) and [delete](https://pushover.net/api/client#delete), as well as any other API calls such as fetching application icons, message sounds, etc.

### **Emergency-Priority Messages**

As with our native device clients, we will send repeated notifications to Open Client connections for [emergency-priority](https://pushover.net/api#priority2) messages according to the message receipt's retry parameter until its expire parameter has been reached.

When your client [downloads](https://pushover.net/api/client#download) a message with a priority of 2 (or higher, for future compatibility) that does not have an acked value of 1, it should present the notification to the user in such a way that gets the user's attention, such as a modal dialog, until it is manually dismissed by the user. When the user acknowledges the notification by performing some action (such as clicking a button, not just when the notification times out), you must submit an API call to acknowledge the receipt and stop repeated notifications.

To acknowledge an emergency-priority message, submit a POST request to the following URL, including the message's receipt parameter in the URL:

https://api.pushover.net/1/receipts/(receipt id)/acknowledge.json

Along with your POST request, include your user's secret parameter.

### **Being Friendly to our API**

As with our [Message API guidelines](https://pushover.net/api#friendly), the way our servers respond is critical in determining how your client should react. If our servers respond with a 200 HTTP code and a status of 1, you've done well. Any other response, such as a 4xx or 5xx HTTP code, or a status of 0 means something went wrong. When something goes wrong, don't retry the same query over and over without any delay.

If our WebSocket server drops your connection after a successful login, wait some time before reconnecting.

In all HTTP calls to our API, your application must send a User-Agent header that clearly shows the name (and other useful information such the version or OS) of your application. This will help us identify any potential problems with users of your application.

### **Distribution Guidelines**

Pushover's Open Client API is intended for advanced users to write their own native clients to receive push notifications on platforms we do not support. Applications using the Pushover Open Client API must follow our [Logos and Usage](https://support.pushover.net/i63-pushover-logos-and-usage) guidelines. In particular, your client may not use the Pushover logo as its main application icon, it may not use Pushover in its name (e.g., "Pushover for x"), it must clearly state that it is an unofficial Pushover Open Client, and that it is not released or supported by Pushover.

* You may not store Pushover account or device information in any location that the Pushover user does not have sole control over. You may not make a server-based client that users login to and then act on their behalf. All Open Client functionality must be done on the user's device/computer and be controlled by the user. This is to limit exposure of a user's e-mail address, password, and/or account secret.
* You may release your Open Client application in binary or source code form and you may charge money for it, but you cannot intervene in licensing the end-user's account (as a Pushover for Desktop license is required to use any Open Client applications).
* You may not do bulk purchases of Pushover for Desktop licenses and then resell them to your users.
* You must direct users to our website at <https://pushover.net/> for account creation and licensing.
* Your client may not be used for delivering notifications to multiple users of a single Pushover account.

In general, don't be a jerk. Don't abuse our Open Client API to get around our licensing, don't try to hide that your client uses Pushover for its message delivery, and don't abuse our API and servers. We rely on users paying for licenses to be able to continue developing and supporting Pushover.

If you have any questions about whether your Open Client is complying with our guidelines, please [contact us](https://pushover.net/support) before developing or releasing it. We'll be happy to answer any questions.