BBM Chat API Documentation

The BBM Chat API is a RESTful HTTP interface that enables third-party services to receive and respond to chat conversations with BBM end users.

The initial audience for this API is designed for Bots and can only respond to end-user initiated conversations in the context of a BBM Channel 1:1 chat. Future editions of this API may expand its capabilities for a non-BOT audience.

Getting Started

In order to use the Chat API a service or bot must first be provisioned with BBM.
To provisioning your service requires that you:

1. Have a BBM Channel with which your service or bot will be connected.
2. Provide a HTTPS callback URL that BBM will use to POST chat messages to your service.

Your callback URL must be https and must end in "/v1/chats" e.g.:

```
https://www.example.com/something/v1/chats
```

We will only POST to your callback on port 443.

Once provisioned, your service will be given an Client Sig (HMAC) key and set of client_credentials.

The Client Sig key e.g.:

```
9c1d207d26f7c498b4c239bbf4e5282b9ef4fe1c18075e2e79d4db24e215904b
```

is used to generate a signature on each POST to your callback. Your service must verify this signature using the Client Sig key to ensure that the request comes from the BBM Partner Framework.

The client_credentials e.g.:

```
ZWQ3MmE5YTlmODQ1YTQ5NzpDUlBwWmVwd3RueUkxbGplNkRHeDM0U25nT3VrOEdiL0hSNmZyQ0YxTjk0PQ==
```

is a set of client credentials and is used to retrieve a BBM access token from our Token Service API. A BBM access token required to call BBM Partner Framework APIs. BBM access tokens expire at preset intervals. Your service must call the Token Service API to retrieve a new access token when the current access token expires. You should call the Token Service API for a new token before the existing one has expired, but may call after expiry if preferred

Notifications

Delivery notifications

BBM uses delivery status notifications to indicate whether a message has been delivered to, or ready by the recipient. There are several delivery states, each with a corresponding symbol that shows up next to the message on the user's client:

Symbol	Name	Description
✓	Checked (CHK)	Message has been accepted by the BBM infrastructure and is scheduled for delivery.
D	Delivered (DLV)	Message has been delivered to the end user.
R	Read	Message has been read by the end user.
🕒	Waiting	Message is queued for sending to server.
☒	Error	Message could not be delivered.

These notifications are important to how users perceive your service's behavior when reading and responding to messages.
Starting from version 1.x, you can choose whether to turn delivery notifications On or Off when provisioning your service. Prior to version 1.x delivery notifications are turned Off. Turning off delivery notifications means the messages will be marked as read immediately after successfully sending the message to your service via the HTTP callback URI.

The following table illustrates the behavior for turning delivery notifications on or off:

Condition	Status Code	On	Off	Since

Message sent to service via HTTP callback	200 OK	Delivered	Read	1.x
	4xx-5xx*	Error	Error	1.x
Message sent to user via Chat API	200 OK	Delivered	--	1.x
	202 Accepted	Checked	--	1.x
	4xx-5xx*	Error	Error	1.x
*The Chat API will attempt to deliver the message up to three times if possible.				

When delivery notifications are turned on, the Chat API will send delivery and read actions to the HTTP callback URI. When turned on, the service MUST send read actions to the client when the message is "read" by the service or operator.

Typing Notifications

BBM can send typing notifications to your service via the HTTP callback URI when the user starts typing in a chat. BBM typing notifications set the user's "is typing..." state and do not clear it.
Instead, your service should automatically clear the user's typing state when:

1. You receive a message from the user in the chat, or
2. After 30 seconds if you do not receive another typing action.

Your service can also send typing notifications to the end user. This is useful if you have an operator, or if your service are performing an operation that will take some time.

BBM recommends and follows these rules for sending typing notifications.

Outgoing:

1. When a user starts typing, set a timer to send a typing action to the recipient in 5 seconds if and only if you have not sent a typing action in the last 30 seconds.

2. If the user sends the message before the 5 seconds have elapsed, cancel the timer and do not send the typing action.
3. If the 5 seconds has elapsed, send the typing action and remember the time the typing action is sent for determining whether there was a typing action in the last 30 seconds.
4. After 30 seconds, only schedule a typing action if the user is currently typing, not if they have been idle.
5. When a user sends a message, clear the time of the last typing action to reset the 30 second condition. I.e.: Treat the first typing action after sending a message as having not sent a message in the last 30 seconds.

Incoming:

1. When you receive a typing action set the user's state to "is typing a message..." or "is typing..." or similar.
2. When you receive a message clear the user's typing state.
3. If 30 seconds elapses without receiving a typing action, clear the user's typing state.

You can choose whether to receive typing notifications when you provision your service. If you turn typing notifications off you can still send them to the user.

Leave notifications

This action is used to inform the service when the user has left or ended the chat. Once you receive this notification your service MUST stop sending messages to this user using this chat id and token. The end user's client will reject further messages for the given chat id once they have left the chat.

If your service has an operator, your service MUST indicate that the user has left the chat, and SHOULD disable the chat UI for writing and sending messages on the operator's console for this chat.

HTTP callback URI

The BBM Chat API will send messages to your service by HTTPS POST to your HTTP callback URI.

```
POST <registered callback URL>/v1/chats
```

Request Headers

Header Name	Type	Required	Description	Since
BBM-Sig	String	Yes	Signature which is computed by the BBM platform.	1.0
Content-Type	String	Yes	Example: application/json; charset=utf-8	1.0
Content-Length	Integer	Yes	The length of the request body.	1.0

Signature validation

BBM-Sig signatures are computed by:

1. Calculating a digest value of the request body with HMAC-SHA256 algorithm using the Client Sig key value as the secret key.
2. Encoding the digest value using BASE-64.

Sample:

```
String clientSigKey =
"9c1d207d26f7c498b4c239bbf4e5282b9ef4fe1c18075e2e79d4db24e215904b";
String httpRequestBody = "...";
Mac mac = Mac.getInstance("HmacSHA256");
SecretKeySpec secretKey = new SecretKeySpec(clientSigKey.getBytes(), "HmacSHA256");
mac.init(secretKey);

String signature =
Base64.encodeBase64String(mac.doFinal(httpRequestBody.getBytes("UTF-8")));
```

Request Body

The required values for sending text to the BOT API server are as follows.

Name	Type	Required	Description	Since
mType	String	Yes	The notfication type. Supported values: message | bot Possible values: private | retract | timed	1.0
chId	String	No	Channel Id for the associated BBM channel.	1.0
chatId	String	Yes	A chat Id identifying a conversation with the user.	1.0
from	String	Yes	A BBM Id identifying the user that the messages are from.	1.0
to	String	Yes	A BBM Id identifying the user that the message is to.	1.0
mTok	String	No	A token required for sending a message to this chat id.	1.0
actions	Array	No	An array of action objects containing one or more actions or status notifications.	1.0
messages	Array	No	An array of message objects containing one or more message payloads.	1.0
userInfos	Map	Yes	A map of BBM Id to userinfo objects.	1.0
Additional properties must be supported by your implementation.				

Sample:

```
POST <registered callback URL>/v1/chats

BBM-Sig: Rg/9uO51033OmfPOmnhpzkIzphnBy6y0XIOnSoQ/rec=
Content-Type: application/json; charset=UTF-8
Content-Length: 418

{
  "mType": "message",
  "chId": "C00012B1E",
  "chatId": "837680",
  "from": "100006006205",
  "to": "31045128202",
  "mTok": "MS4xNDc2ODM4MDk3LjdkZDk4MjkxM2ZjNDBhMDc5YzI0ODJmYTQ5OTQ0OTg2",
  "actions": [ <one or more action objects> ],
  "messages":[ <one or more message objects> ],
  "userInfos":{ <bbmid>:<userInfo object>, ... },
  "additional properties": "must be supported and ignored",
  ...
}
```

Action object

Action objects indicate the user or service has performed an action. Actions include read and delivery notifications of messages, typing notifications, and when the user leave's the chat.

You can choose whether to turn on delivery notifications:

- If you turn off delivery notification, messages sent to your service will automatically be marked as read in the user's chat when the message is sent to your service.
- If you turn on delivery notifications, your service MUST send read notifications in response to the user's messages when your service reads them.

Name	Type	Required	Description	Since
type	String	Yes	The action type. Supported values: leave | typing	1.0
ts	Integer	Yes	The timestamp of the message in milliseconds since Jan 1, 1970.	1.0
Plus additional properties from one of the below action types.				

Sample:

```
{
  "type": <one of the action types>,
  "ts":1332394961610,
  "additional properties": "from one of the below message types",
  ...
}
```

Leave action type

This action is used to inform the service that the user has left the chat. The service MUST stop sending messages to this user using this chat id and token. The service SHOULD disable the chat UI for writing and sending messages on the operator's console for this chat, and SHOULD indicate that the user has left the chat. Sending further message to the user with the given chat id will fail.

Name	Type	Required	Description	Since
--none--	--	--	--	1.0
Additional properties must be supported by your implementation.				

Sample:

```
{
  "type": "leave",
  "ts":1332394961610,
   ...
}
```

Typing action type

This action is used to indicate when the user has started typing in the chat. This is used to set the " < user > is typing..." state.
No action is sent to clear the typing state. Your service MUST clear this status automatically when you receive a message from the user in the same chat or after 30 seconds without receiving a typing action for that chat. See the typing notification section for further details.

Name	Type	Required	Description	Since
--none--	--	--	--	1.0

Additional properties must be supported by your implementation.

Sample:

```
{
  "type": "typing",
  "ts":1332394961610,
   ...
}
```

Delivery (DLV) action type

BBM can notify your service when a message as been delivered to the end user's client (Delivery Receipt Notification.) This action indicates one or more messages have been delivered to the end user and should have their delivery status changed to "D" or delivered.

Name	Type	Required	Description	Since
mids	Array	Yes	An array of one or more message ids indicating the messages that have changed their delivery status to delivered.	1.x
Additional properties must be supported by your implementation.				

Sample:

```
{
  "type": "dlv",
  "ts":1332394961610,
  "mids": ["7443411170bf9696", ...]
  ...
}
```

Read action type

BBM can notify your service when a message as been read by the end user (Read Receipt Notification.) This action indicates one or more messages have been read by the end user and should have their delivery status changed to "R" or read.

Name	Type	Required	Description	Since

mids	Array	Yes	An array of one or more message ids indicating the messages that have changed their delivery status to delivered.	1.x
Additional properties must be supported by your implementation.				

Sample:

```
{
  "type": "read",
  "ts":1332394961610,
  "mids": ["7443411170bf9696", ...]
  ...
}
```

Message object

Name	Type	Required	Description	Since
index	Integer	No	A message index indicating order. Starts at 1. Not required when only one message is sent.	1.0
type	String	Yes	The message type. Supported values: image | text Possible values: audio | music | richmedia | video etc	1.0
mid	String	Yes	A unique message Id.	1.0
ts	Integer	Yes	The timestamp of the message in milliseconds since Jan 1, 1970.	1.0
Plus additional properties from one of the below message types. Unrecognized properties should be ignored.				

Sample:

```
{
  "index": 1,
  "type": <one of the message types>,
  "mid": "2524be69e40ec933",
  "ts":1332394961610,
  "additional properties": "from one of the below message types",
  ...
}
```

Text type

Name	Type	Required	Description	Since
text	String	Yes	For text messages, the actual UTF-8 text of the message. Limit: 2000 characters.	1.0

Sample:

```
{
  "index": 1,
  "type": "text",
  "mid": "2524be69e40ec933",
  "ts":1332394961610,
  "text": "This is my text message.",
  ...
}
```

Image type

Name	Type	Required	Description	Since
image	Image	Yes	An image object defining the image to display.	1.0

Image object

Name	Type	Required	Description	Since
preview	String	No	URL for an optional medium preview image to display. Recommended sizes: 256x144, 300x250, 300x300, 384x216 px Supported formats: Png, Jpg, Gif, Animated Gif. Limits: 4096 character limit for URL. 150 KB image or smaller	1.0

url	String	Yes	URL to the image to display. Supported formats: Png, Jpeg, Gif, Animated Gif Limits: 4096 character limit for URL. 10 MB image or smaller	1.0
Additional properties must be supported by your implementation.				

Sample:

```
{
  "index": 1,

  "type": "image",

  "mid": "2524be69e40ec933",
  "ts":1332394961610,

  "image": {
    "preview":
"https://placeholdit.imgix.net/~text?txtsize=33&txt=256%C3%97144&w=256&h=144",
    "url":
"https://placeholdit.imgix.net/~text?txtsize=33&txt=1080×566&w=1080&h=566",
    ...
  }

}
```

UserInfo object

Name	Type	Required	Description	Since
name	String	Yes	User's display name.	1.0
Additional properties must be supported by your implementation.				

Avatar object

Field Name	Type	Required	Description	Since
imageSize	Integer	Yes	Avatar's image size in bytes	1.2

Field Name	Type	Required	Description	Since
resolution	String	Yes	Avatar's image resolution	1.2
imageType	String	Yes	Avatar's image mime type	1.2
url	String	Yes	Avatar's URL	1.2

Sample:

```
POST <registered callback URL>/v1/chats

BBM-Sig: UhmliDu5Bz/YI1T5G5bQ5DodDWi8YmmDLsP2cT5sm9w=
Content-Type: application/json; charset=UTF-8
Content-Length: 380

{
  "mType": "message",
  "chId": "C00012B1E",
  "chatId": "837680",
  "from": "100006006205",
  "to": "31045128202",
  "mTok": "MS4xNDc2ODM4MDk3LjdkZDk4MjkxM2ZjNDBhMDc5YzI0ODJmYTQ5OTQ0OTg2",
  "actions:" [ < one or more action objects > ],
  "messages":[ < one or more message objects > ],

  "userInfos":{
    "100006006205": {
       "name": "Alice",
       "avatar": [
           {
              "imageSize": 1166,
              "imageType": "image/jpeg",
              "resolution": "60x60",
              "url": "http://bislab-rim.akamaized.net/cdn/vg002/kronos-
olystr.olympia.str.blackberry.com/bbmavatar-13/..."
           },
           {
              "imageSize": 1757,
              ...
           },
           ...
        ],
        ...
     }
  }

  ...

}
```

HTTP Response

Your HTTP Callback URI should return standard HTTP responses. The following status codes have specific meaning to the BBM Chat API and must supported by your service

Status Code	Name	Description	Since
200	OK	Return to indicate that the message(s) was delivered. If you have turned on delivery notifications, BBM will mark the messages as "D" or delivered; otherwise the messages will be marked as "R" or read.	1.0
207	Multi-Status	Return to indicate that at least one message has failed to deliver.	1.0
400	Bad Request	Return if the request contains invalid parameters or is otherwise badly formed.	1.0
401	Unauthorized	Return if the BBM-Sig does not validate.	1.0
500	Internal Server Error	Return if your service encounters an unexpected error. BBM will retry at a later time.	1.0
503	Service Unavailable	Return if your server temporarily unavailable. BBM will retry at a later time.	1.0

The HTTP response codes above indicate the result for the whole request. If the HTTP response code returned is an error this means that the entire HTTP callback request has failed.
If the HTTP callback call was successful return a 200 OK, or if partially successful, return a 207 Multi-Status response code. In both of these cases you must return a response body with the result status for each message sent to your callback URI.

Response headers

Name	Type	Required	Description	Since
Content-Type	String	Yes	The content-type and encoding for the request body. It must be: application/json; charset=utf-8	1.0
Content-Length	Integer	Yes	The length of the request body.	1.0

Response body

The message objects in the result body are a simplified form of the message object. Include one message object in the results for each message sent to your URI. Actions are considered lower priority notifications and do not require a result object.

Name	Type	Required	Description	Since
messages	Array	No	An array of zero or more messages objects, one per message sent. If no messages were sent (only actions) this array is not required.1.0	1.0

Message object

Name	Type	Required	Description	Since
mid	String	No	A unique message Id. Not required when the code indicates an error.	1.0
result	Result	Yes	A result object containing on of the result codes below.	1.0

Result object

Name	Tyoe	Required	Description	Since
code	String	Yes	One of the result codes from the table below.	1.0
error	String	No	A short error description if desired.	1.0

Codes

The result object codes are similar to the HTTP status codes above. In this case, each code reflects the status of an individual message.

Code	Code Name	Description	Since
200	OK	This message has been delivered to the end user. If you have turned on delivery notifications, BBM will mark the message as "D" or delivered; otherwise the message will be marked as "R" or read.	1.0
400	Bad Request	This message contains invalid parameters or is otherwise badly formed.	1.0
409	Conflict	The message was not set because a preceding message or action in this request failed.	1.0
500	Internal Server Error	The server encountered unexpected software error. Retry again later, it may or may not succeed.	1.0
503	Service Unavailable	Server temporarily unavailable to handle the request. Retry again later, it will not succeed until the server is available again.	1.0

Sample:

```
HTTP 1/1 200 OK
```

```
{
  "messages": [
    {
      "mid": "2524be69e40ec933",
      "result": {
        "code": 200
      }
    }
  ]
}
```

Sending a Message

This API allows a service to send messages and actions in response to a user. The conversation must be user initiated. The message bodies are very similar to those for the HTTP Callback URI for receiving messages. One major difference however is that this API can only be used to send messages to a single user at a time.

```
POST https://chat-beta.bbm.blackberry.com/v1/chats/<chatId>?mTok=<token>
```

Request Headers

Name	Tyoe	Description	Since
Authorization	String	This API requires a BBM access token. Example: Authorization: Bearer < BBM access token >	1.0
Content-Type	String	The content-type and encoding for the request body. It must be: application/json; charset=utf-8	1.0
Content-Length	Integer	The length of the request body.	1.0

Request Parameters

Name	Type	Required	Description	Since
chatId	String	Yes	Chat ID	1.0
mTok	String	Yes	A message token required for sending a message to this chat id.	1.0

Request Body

Name	Type	Required	Description	Since
mType	String	Yes	The notfication type. Supported values: message | bot Possible values: private | retract | timed	1.0
chId	String	No	Channel Id for the associated BBM channel.	1.0
chatId	String	Yes	A chat Id identifying a conversation with the user.	1.0
from	String	Yes	A bbmid identifying the user that the messages are from.	1.0
to	String	Yes	The BBM Id of the user the message(s) is to.	1.0
sessionId	String	No	An unique chat session identification. This identification can be used to identify the start and end of a chat session.	1.2
actions	Array	No	An array of action objects containing one or more actions or status notifications.	1.0
messages	Array	No	An array of message objects containing one or more message payloads.	1.0
userInfos	Map	Yes	A map of BBM Id to userinfo objects.	1.0

Sample:

```
POST https://chat-
beta.bbm.blackberry.com/v1/chats/837680?mTok=MS4xNDc2ODM4MDk3LjdkZDk4MjkxM2ZjNDBhMDc5
YzI0ODJmYTQ5OTQ0OTg2

Authorization: Bearer < BBM uses JWE access tokens obtained from the BBM Token
Service >
Content-Type: application/json; charset=UTF-8
Content-Length: 268
</pre>
<pre>
{
  "mType": "bot",
  "chId": "C00012B1E",
  "chatId": "837680",
  "from": "31045128202",
  "to": "100006006205",
  "actions": [ < one or more action objects > ],
  "messages":[ < one or more message objects > ],
  "userInfos":{ < bbmid >:< userInfo object >, ... }
}
```

Action object

Action objects indicate the user or service has performed an action. Actions include read receipts of messages and typing status.

You can choose whether to use delivery notifications with your service:

- If you turn off delivery notifications messages sent to your service will automatically be marked as read in the user's chat when the message is sent to your service. In this case, you should not send read receipt actions to the user.
- If you turn on delivery notifications, your service MUST send read receipt notifications in response to the user's messages when your service reads them.

You can send typing actions to the end user. This is useful if you have operators typing the response, or if your service will take a long time to prepare the message.

Name	Type	Required	Description	Since
type	String	Yes	The action type. Supported values: read | typing	1.0
			text | link | postback (exclusively used in buttons message)	1.1
Plus additional properties from one of the below message types.				

Sample:

```
{
  "type": <one of the action types>,
  "additional properties": "from one of the below message types"
}
```

Typing action type

This action is used to set the "< user > is typing..." state. This is typically used if you have an operator typing a message but can also be used if your service will take a long time to prepare a message. No action is sent to clear the typing state. BBM clears this status automatically when the user receives a message in the same chat or after 30 seconds without receiving a typing action for that chat.

See the typing notification section for further details.

Name	Type	Required	Description	Since
-- none --	--	--	--	1.0

Sample:

```
{

  "type": "typing"

}
```

Read action type

Use this action to notify the end user when when a message as been read by your service or operator (Read notification.) This action indicates one or more messages have been read and should have their delivery status changed to "R" or read.

Name	Type	Required	Description	Since
mids	Array	Yes	An array of one or more message ids indicating the messages that have changed their delivery status to read.	1.x

Sample:

```
{

  "type": "read",

  "mids": ["2524be69e40ec933", ...]

}
```

Postback action type

BBM can notify your service when this action is tapped. A postback action is returned via webhook with the specified string in the data field.

Name	Type	Required	Description	Since
label	String	Yes	Label for the action Limits: 20 characters	1.1
data	String	Yes	String returned via webhook in the postback.data property of the postback action Limits: 300 characters	1.1
text	String	No	Text sent when the action is performed. Limits: 300 characters	1.1

Sample:

```
{
    "type": "postback"
    "postback" : {
       "label": "Book this offer",
       "data": "action=book&location=rome&offer=123"
    }
}
```

Text action type

BBM can notify your service when this action is tapped, the string in the text field is sent as a message from the user.

Name	Type	Required	Description	Since
label	String	Yes	Label for the action Limits: 20 characters	1.1
text	String	Yes	Text sent when the action is performed Limits: 300 characters	1.1

Sample:

```
{
    "type": "text",
    "text": {
        "label": "Discover latest offers",
        "text": "Discover latest offers"
   }
}
```

Link action type

When this action is tapped, the url specified in the url field is opened.
If you have included the text field, the string in the text field is sent as a message from the user. Otherwise, the label field is sent as a message from the user.

Name	Type	Required	Description	Since
label	String	Yes	Label for the action Limits: 20 characters	1.1
url	String	Yes	URL to display to the user. Limits: 4096 characters	1.1
text	String	No	Text sent when the action is performed Limits: 300 characters	1.1

Sample:

```
{
   "type": "link",
   "link": {
      "label": "Go to our website",
      "url": "http://example.com/page/123"
   }
}
```

Message object

Name	Type	Required	Description	Since

index	Integer	No	A message index indicating order. Starts at 1. Not required when only one message is sent.	1.0
type	String	Yes	The message type. Supported values: image | link | text |	1.0
			buttons	1.1
Plus additional properties from one of the below message types.				

Sample:

```
{
  "index": 1,
  "type": <one of the message types>,
  "additional properties": "from one of the below message types"
}
```

Text type

Name	Type	Required	Description	Since
text	String	Yes	For text messages, the actual UTF-8 text of the message. Limit: 2000 characters.	1.0

Sample:

```
{
  "index": 1,

  "type": "text",
  "text": "This is my text message."

}
```

Image type

Name	Type	Required	Description	Since

image	Image	Yes	An image object defining the image to display.	1.0

Image object

Name	Type	Required	Description	Since
preview	String	No	URL for an optional medium preview image to display. Recommended sizes: 256x144, 300x250, 300x300, 384x216 px Supported formats: Png, Jpg, Gif, Animated Gif. Limits: 4096 character limit for URL. 150 KB image or smaller	1.0
url	String	Yes	URL to the image to display. Supported formats: Png, Jpeg, Gif, Animated Gif Limits: 4096 character limit for URL. 10 MB image or smaller	1.0

Sample:

```
{
  "index": 1,

  "type": "image",
  "image": {
    "preview":
"https://placeholdit.imgix.net/~text?txtsize=33&txt=256%C3%97144&w=256&h=144",
    "url": "https://placeholdit.imgix.net/~text?txtsize=33&txt=1080×566&w=1080&h=566"
  }

}
```

Link type

Name	Type	Required	Description	Since
link	Link	Yes	A link object defining the link to display.	1.0

Link object

Name	Type	Required	Description	Since
url	String	Yes	URL to display to the user. Limit: 4096 characters.	1.0
target	String	No	How to open the link. Values: def | extdef: launches the link in a new webview. ext: launches the link in an external browser.	1.0

Sample:

```
{
  "index": 1,

  "type": "link",
  "link": {
    "url":
"https://placeholdit.imgix.net/~text?txtsize=33&txt=256%C3%97144&w=256&h=144",
    "target": "def"
  }

}
```

Buttons type

Name	Type	Required	Description	Since
buttons	Buttons	Yes	A buttons message defining an image, title, description and multiple actions.	1.1

Buttons object

Name	Type	Required	Description	Since
imageUrl	String	No	URL of the image to display. Supported formats: Png, Jpeg, Gif, Animated Gif	1.1

			Limits: 1000 character limit for URL. 10 MB image or smaller	
title	String	No	Title string that shows up on top of the message. Limits: 40 characters	1.1
desc	String	Yes	Descriptive message. Limits: 160 characters (no image or title) Limits: 60 characters (message with an image or title)	1.1
actions	Array of actions	Yes	A list of actions. Limits: 4	1.1

Sample:

```
{
  "type": "buttons",
  "buttons": {
    "imageUrl": "https://example.com/bot/images/image.jpg",
    "title": "Rome Dream Discount",
    "desc": "Exclusive for BBM users. Special discount package for families.",
    "actions": [
      {
        "type": "text",
        "text": {
          "label": "Discover latest offers",
          "text": "Discover latest offers"
        }
      },
      {
        "type": "postback",
        "postback": {
          "label": "Book this offer",
          "data": "action=book&location=rome&offer=123"
        }
      },
      {
        "type": "postback",
        "postback": {
          "label": "Summer catalogue",
          "data": "season=summer&location=rome"
        }
      },
      {
        "type": "link",
        "link": {
          "label": "Go to our website",
          "url": "http://example.com/page/123",
          "text": "Go to our website"
        }
      }
    ]
  }
}
```

UserInfo object

When using the Send a Message functionality of this API, the userInfos must be populated with one entry for each BBM Id that corresponding message or action is from.

Name	Type	Required	Description	Since
name	String	Yes	User's display name	1.0

Sample:

```
POST https://chat-beta.bbm.blackberry.com/v1/chats/837680?
mTok=MS4xNDc2ODM4MDk3LjdkZDk4MjkxM2ZjNDBhMDc5YzI0ODJmYTQ5OTQ0OTg2

Authorization: Bearer < BBM uses JWE access tokens obtained from the BBM Token
Service >
Content-Type: application/json; charset=UTF-8
Content-Length: 294

{
  "mType": "message",
  "chId": "C00012B1E",
  "chatId": "837680",
  "from": "31045128202",
  "to": "100006006205",
  "actions:" [ < one or more action objects > ],
  "messages":[ < one or more message objects > ],

  "userInfos":{
    "31045128202": { "name": "BobBot", ... }
  }

}
```

HTTP Response

Your HTTP Callback URI should return standard HTTP responses. The following status codes have specific meaning to the BBM Chat API and must supported by your service.

Response Codes

Status Code	Code Name	Description	Since
200	OK	The request has been successfully submitted to the server. See the response body for the result objects for each individual message sent.	1.0
207	Multi-Status	The request has been submitted to the server, but at least one message has failed with an error. See the response body for the result objects for each individual message sent.	1.0
400	Bad Request	One or more of your parameters is invalid or the request is otherwise badly formed and cannot be processed.	1.0
401	Unauthorized	The authentication token is missing, expired, or is otherwise invalid.	1.0
403	Forbidden	The service lacks sufficient permissions to perform the operation. Example: Sending a message to a user who has not initiated the given chat id, or without a valid message token.	1.0
404	Not Found	The message cannot be sent because the conversation (chat id) or user (bbmid) does not exist or no longer exists	1.0

429	Too Many Requests	The service has exceeded the maximum number of requests allowed in a given period.	1.0
500	Internal Server Error	The server encountered unexpected software error. Retry again later, it may or may not succeed.	1.0
503	Service Unavailable	Server temporarily unavailable to handle the request. Retry again later, it will not succeed until the server is available again.	1.0

Response headers

Name	Type	Required	Description	Since
Content-Type	String	Yes	The content-type and encoding for the request body. It must be: application/json; charset=utf-8	1.0
Content-Length	Integer	Yes	The length of the request body.	1.0

Response body

When you send a message using the Chat API, the API returns a message object for each message sent containing a result code for that individual message. If the message was sent successfully, this object will contain the message's mid and time stamp. Your service can use mid for handling delivery and read receipt notifications from the client. If the message encountered an error the message result will contain the error code. Your service can use the index, chatId and to and from fields to match the message result to the message sent.

When sending more than one message they are processed in index order. The response will contain one message result object for each message in the request. If an error is encountered while processing a message, the server will stop the operation and any subsequent messages in index order will not be sent for the request. The success or failure state of each message is indicated in the result list. Failed messages may be retried by your service once the issue has been corrected.

Action objects are processed first, before message objects. If an error is encountered while processing an action, the server may continue the operation depending on whether it is able to recover. Whether successful or otherwise, no result are added for action objects. If the server elects to stop the operation due to an error with an action none of the messages will be sent.

Name	Type	Required	Description	Since
messages	Array	No	An array of zero or more message objects, one per message sent. If no messages were sent (only actions) this array is not required.	1.0
Additional properties must be supported by your implementation.				

Message result object

Name	Type	Required	Description	Since
index	Integer	No	A message index indicating order. Starts at 1.Not required when only one message is sent.	1.0
type	String	Yes	The message type. Supported values: result	1.0
mid	String	No	A unique message Id. Not required when the code indicates an error.	1.0
ts	Integer	No	The timestamp of the message in milliseconds since Jan 1, 1970. Not required when the code indicates an error.	1.0

result	Result	Yes	A result object containing on of the result codes below.	1.0
Additional properties must be supported by your implementation.				

Result object

Name	Type	Required	Description	Since
code	String	Yes	One of the result codes from the table below.	1.0
error	String	No	A short error description. Not required when the code indicates a success.	1.0

Codes

The result object codes are similar to the HTTP status codes above. In this case, each code reflects the status of an individual message.

Code	Code Name	Description	Since
200	OK	The message(s) have been delivered to the end user. If you don't display a message delivery status you can turn off delivery notifications. If you turn on delivery notifications, the message's status is "D" or delivered (DLV).	1.0
202	Accepted	If you have turned off delivery notifications you will not receive this response. If you have delivery notifications turned on this response means the server has accepted the message for delivery but it has not yet been received. This corresponds to a checkmark (CHK) in BBM.	1.x

		You will be sent a delivery action to notify you when the message is delivered (DLV).	
403	Forbidden	The service lacks sufficient permissions to perform the operation. Example: Sending a message to a user who has not initiated the given chat id.	1.0
409	Conflict	The message was not set because a preceding message or action in this request failed.	1.0
429	Too Many Requests	The service has exceeded the maximum number of messages allowed in a given period or request.	1.0
500	Internal Server Error	The server encountered unexpected software error. Retry again later, it may or may not succeed.	1.0
503	Service Unavailable	Server temporarily unavailable to handle the request. Retry again later, it will not succeed until the server is available again.	1.0

Sample:

```
HTTP 1/1 200 OK

{
  "messages":[
    {
      "index": 1,
      "type": "result",
      "mid": "344eab12c9923a80",
      "ts": "1332394962100",
      "result": {
          "code": 200,
          ...
        }
      ...
    } ],
  ...
}
```

Putting it together

What follows is a sample use case of a user, Alice, initiating a conversation with a channel chat bot, BobBot. BobBot replies (see sending messages below) with a read receipt and a response to the initial message. Alice sends a typing notification. Then two messages are sent to BobBot.

Single text message

The following example illustrates a single user sending a text message "Hello World!". For a new chat conversation, a typing notification is not sent prior to sending the first message.

```
POST <registered callback URL>/v1/chats

BBM-Sig: 3pKSFUlRUEAv339oBTouy/XII7FM1yqMXqAYeo0JT8o=
Content-Type: application/json; charset=UTF-8
Content-Length: 452

{
  "mType": "message",
  "chId": "C00012B1E",
  "chatId": "837680",
  "from": "100006006205",
  "to": "31045128202",
  "mTok": "MS4xNDc2ODM4MDk3LjdkZDk4MjkxM2ZjNDBhMDc5YzI0ODJmYTQ5OTQ0OTg2",
  "messages": [
    {
      "index": 1,
      "type": "text",
      "mid": "2524be69e40ec933",
      "ts": 1332394961610,
      "text": "Hello World!"
    }
  ],
  "userInfos": {
    "100006006205": {
      "name": "Alice",
      "avatar": [
        {
          "imageSize": 1166,
          "imageType": "image/jpeg",
          "resolution": "60x60",
          "url": "http://bislab-rim.akamaized.net/cdn/vg002/kronos-
olystr.olympia.str.blackberry.com/bbmavatar-13/zipper-
00001/1493212126832/1493218504060/4b52b96f38337e31b93a771771a55e9308b391567e0976fdb0f
4845d08f37e41_60x60.jpg?dwnAuth=1814730075_a63a57955d24639c6afc74b8e2503230"
        },
        {
          "imageSize": 1757,
          "imageType": "image/jpeg",
          "resolution": "100x100",
          "url": "http://bislab-rim.akamaized.net/cdn/vg002/kronos-
olystr.olympia.str.blackberry.com/bbmavatar-13/zipper-
00001/1493212126832/1493218504060/4b52b96f38337e31b93a771771a55e9308b391567e0976fdb0f
4845d08f37e41_100x100.jpg?dwnAuth=1814730075_9dfded8e029736f631f7958e175a6dcb"
        },
        {
          "imageSize": 19807,
          "imageType": "image/JPEG",
          "resolution": "320x320",
          "url": "http://bislab-rim.akamaized.net/cdn/vg002/kronos-
olystr.olympia.str.blackberry.com/bbmavatar-13/zipper-
00001/1493212126832/1493218504060/4b52b96f38337e31b93a771771a55e9308b391567e0976fdb0f
4845d08f37e41.jpg?dwnAuth=1814730075_85f2a3b47f06a722ff274dce9162d341"
        }
      ]
    }
  }
}

HTTP 1/1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 128

{
  "messages":[
    {
      "mid": "344eab12c9923a80",
      "result": {
        "code": 200
        }
    } ]
}
```

Replying with a read receipt and text message

The following example illustrates the service replying with a read receipt notification and a message back to the user.

```
POST https://chat-beta.bbm.blackberry.com/v1/chats/837680?
mTok=MS4xNDc2ODM4MDk3LjdkZDk4MjkxM2ZjNDBhMDc5YzI0ODJmYTQ5OTQ0OTg2

Authorization: Bearer < BBM uses JWE access tokens that are approximately 620 bytes
long >
Content-Type: application/json; charset=UTF-8
Content-Length: 383

{
  "mType": "bot",
  "chId": "C00012B1E",
  "chatId": "837680",
  "from": "31045128202",
  "to": "100006006205",
  "actions": [
    {
      "type": "read",
      "mids": ["2524be69e40ec933"]
    } ],
 "messages":[
    {
      "index": "1",
      "type": "text",
      "text": "Hello Alice!"
    } ],
  "userInfos":{
    "31045128202": { "name": "BobBot" }
  }
}

HTTP 1/1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 208

{
  "messages":[
    {
      "index": 1,
      "type": "result",
      "mid": "344eab12c9923a80",
      "ts": "1332394962100",
      "result": {
        "code": 200
        }
    } ]
}
```

Read receipt and typing notification

The following example illustrates the user sending a typing action back to the service.

```
POST < registered callback URL >/v1/chats

BBM-Sig: 8tglM7XsTYBIwFgmjO0xMUM63j23hY3raGOCUsQxW4c=
Content-Type: application/json; charset=UTF-8
Content-Length: 298

{
  "mType": "message",
  "chId": "C00012B1E",
  "chatId": "837680",
  "from": "100006006205",
  "to": "31045128202",
  "actions":[
    {
      "type": "typing",
      "ts": 1332394963010
    } ],
  "userInfos":{
    "100006006205": {
      "name": "Alice"
    }
  }
}

HTTP 1/1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 4

{
}
```

Multiple message from a single user

The following example illustrates a single user sending a text message followed by an image message in the same chat conversation. Note that the service must clear the typing state for the user because it received a message in the same chat conversation.

```
POST < registered callback URL >/v1/chats

BBM-Sig: VVxiuGY0aITEqYxdZHmQIOg0W6AFllOOE+f27rk9Jzg=
Content-Type: application/json; charset=UTF-8
Content-Length: 810

{
  "mType": "message",
  "chId": "C00012B1E",
  "chatId": "837680",
  "from": "100006006205",
  "to": "31045128202",
  "mTok": "MS4xNDc2ODM4MDk3LjdkZDk4MjkxM2ZjNDBhMDc5YzI0ODJmYTQ5OTQ0OTg2",
  "messages": [
    {
      "index": "1",
      "type": "text",
      "mid": "3f4e094e9785fc70",
      "ts": 1332394966880,
      "text": "Can you identify this item?"
    },
    {
      "index": "2",
      "type": "image",
      "mid": "07dcb3e02ae3d19c",
      "ts": 1332394968200,
      "image": {
        "preview":
"https://placeholdit.imgix.net/~text?txtsize=33&txt=256%C3%97144&w=256&h=144",
        "url":
"https://placeholdit.imgix.net/~text?txtsize=33&txt=1080×566&w=1080&h=566"
      }
    } ],
  "userInfos": {
    "100006006205": {
      "name": "Alice"
    }
  }
}

HTTP 1/1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 233

{
  "messages":[
    {
      "mid": "3f4e094e9785fc70",
      "result": {
        "code": 200
        }
    },
    {
      "mid": "07dcb3e02ae3d19c",
      "result": {
        "code": 200
        }
    } ]
}
```

Buttons message

The following example illustrates a buttons message sent by a Chat BOT and followed by a postback action sent to the Chat BOT by the user in the same chat conversation.

```
POST https://chat-beta.bbm.blackberry.com/v1/chats/837680?
mTok=MS4xNDc2ODM4MDk3LjdkZDk4MjkxM2ZjNDBhMDc5YzI0ODJmYTQ5OTQ0OTg2

Authorization: Bearer < BBM uses JWE access tokens that are approximately 620 bytes
long >
Content-Type: application/json; charset=UTF-8
Content-Length: 1298


{
  "mType": "bot",
  "chId": "C00012B1E",
  "chatId": "837680",
  "from": "31045128202",
  "to": "100006006205",
  "messages": [
    {
      "type": "buttons",
      "buttons": {
        "imageUrl": "https://example.com/bot/images/image.jpg",
        "title": "Rome Dream Discount",
        "desc": "Exclusive for BBM users. Special discount package for families.",
        "actions": [
          {
            "type": "text",
            "text": {
              "label": "Discover latest offers",
              "text": "Discover latest offers"
            }
          },
          {
            "type": "postback",
            "postback": {
              "label": "Book this offer",
              "data": "action=book&location=rome&offer=123"
            }
          },
          {
            "type": "postback",
            "postback": {
              "label": "Summer catalogue",
              "data": "season=summer&location=rome"
            }
          },
          {
            "type": "link",
            "link": {
              "label": "Go to our website",
              "url": "http://example.com/page/123",
              "text": "Go to our website"
            }
          }
        ]
      }
    }
  ],
  "userInfos": {
    "31045128202": {
      "name": "BobBot"
    }
  }
}

HTTP 1/1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 186

{
  "messages": [
    {
      "index": 1,
      "type": "result",
      "mid": "344eab12c9923a88",
      "ts": "1332394962100",
      "result": {
        "code": 200
      }
    }
  ]
}
```

The postback action sent by the user:

```
{
  "mType": "message",
  "chId": "C00012B1E",
  "chatId": "837680",
  "from": "100006006205",
  "to": "31045128202",
  "actions": [
    {
       "type": "postback",
       "postback": {
          "label": "Book this offer",
          "data": "action=book&location=rome&offer=123"
       }
     }
  ],
  "userInfos": {
    "100006006205": {
      "name": "User 1"
    }
  }
}
```