**秋季**

16

*JioChat HTTP API*

Vincent.Ma

HTTP Message Center API

https://www.jiochat.com/

**New Message Center HTTP API**

1. [HttpMSC] Send Message: <http://10.135.140.118:9000/v1/sendmessage>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Request Method: POST | | | | |
| Name | Parameter | Type | Description |  |
| Body | from | Text | Sender user id | Required |
|  | to | Text | Receiver user id | Required |
|  | messageId | Text | 32 character | Required |
|  | groupId | Text | Group Id | Required |
|  | tenantId | Text | Tenant Id | Required |
|  | ~~deviceId~~ | ~~Text~~ | ~~deviceToken+userId~~ | ~~Required~~ |
|  | ~~credential~~ | ~~Text~~ | ~~Sender user credential~~ | ~~Required~~ |
|  | type | Text | Reference JioChat message type. | Optional |
|  | extMap | Text | Extend property | Optional |
|  | message | Text | CinRequest(Message) | Required |
| End |  |  |  |  |

|  |  |
| --- | --- |
| Part propery description | |
| messageId | generate by client. |
| message | Use CinConvert.bytes2String(CinRequest.toBytes()) convert .  CinRequest is a message request, server will send this message request to receiver with doing noting change. |
| ~~credential~~ | ~~Reference Logon 0x03, make byte[] credential -> String credential, use CinConvert.bytes2String(byte[]).~~ |
| ~~deviceId~~ | ~~deviceId and credential they are used for check user status~~ |

Request Body With Json format:

{

"from": "100001231",

"to": "100001441",

"messageId": "837A5AC5B23286AA3D38CC9750164C1C",

"groupId": "0",

"tenantId": "tenant-jiochat",

~~"deviceId": "0FFDE494-4D03-441F-AFA2-227AF72A0C2100001231",~~

~~"credential": "",~~

"type": "1",

"extMap": "city:mumbai,age:18",

"message": ""

}

|  |  |  |  |
| --- | --- | --- | --- |
| Response |  |  |  |
| Name | Parameter | Description |  |
| Body | code | Response code | Required |
|  | msg | Response message | Required |
|  | resp | Json String for result | Optional |
| End | | | |

Response Body:

{

"code": "1",

"msg": "OK",

"resp": {

"messageIndex": "1",

"dateTime": "1530616385852"

}

}

2. [HttpMSC] Get Message: <http://10.135.140.118:9000/v1/getmessage>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| c | | | | |
| Name | Parameter | Type | Description |  |
| Body | from | Text | Sender user id | Required |
|  | tenantId | Text | Tenant Id | Required |
|  | startIndex | Text | Start index to get message | Required |
|  | count | Text | Get message count, default 20 | Optional |
| End |  |  |  |  |

Request Body:

{

"from": "100001231",

"tenantId": "tenant-jiochat",

"startIndex": "123",

"count": "20" // default20,max200.

}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Response |  |  |  | |
| Name | Parameter | Description | |  |
| Body | code | Response code | | Required |
|  | msg | Response message | | Required |
|  | resp | Json String for result | | Required |
| End | | | | |

Response Body:

{

"code": "1",

"msg": "OK",

"resp":

[

{

"session\_key": "100001441100001231",

"tenant\_id": "tenant-jiochat",

"message\_index":100,

"sender\_id":100001231,

"receiver\_id":100001441,

"group\_sender":0,

"message\_id": "837A5AC5B23286AA3D38CC9750164C1C",

“message\_content”:”CinRequest message with hexString”,

"message\_status":1,

"message\_time":1530616385852,

"delflag\_min": 100001231,

"delflag\_max":0,

“msg\_ext”:{“key1”:”value1”,”key2”:”vakue2”}

},

{},

…

]

}

3. [HttpMSC] ReadReply <http://10.10.135.140.9000/v1/readreply>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Request Method: POST | | | | |
| Name | Parameter | Type | Description |  |
| Body | Request | Text | CinRequest of ReadReplay with HexString | Required |
| End |  |  |  |  |

Request Body:

{

"request": "040104C0E5F5050204C0E5F501604C0E5F5050B0D2B39313938373635343332313000"

}

request = CinConvert.bytes2String(ReadReplyRequest.toBytes());

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Response |  |  |  | |
| Name | Parameter | Description | |  |
| Body | code | Response code | | Required |
|  | msg | Response message | | Required |
| End | | | | |

Response Body:

{

"code": "1",

"msg": "OK”

}

4. [HttpMSC] Get Delivery Message <http://10.10.135.140.9000/v1/getdeliverymessage>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Request Method: POST | | | | |
| Name | Parameter | Type | Description |  |
| Body | From | Text | From user id | Required |
|  | To | Text | To user id | Required |
| End |  |  |  |  |

Request Body:

{

"from": "10000231",

“to”:”10000232”

}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Response |  |  |  | |
| Name | Parameter | Description | |  |
| Body | code | Response code | | Required |
|  | msg | Response message | | Required |
|  | MessageIndex | Delivery message index | | Optional |
| End | | | | |

Response Body:

{

"code": "1",

"msg": "OK”,

“messageIndex”:”1,2,3,4”

}

5. [HttpMSC] Delete message by message index

<http://10.10.135.140.9000/v1/deletemessage>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Request Method: POST | | | | |
| Name | Parameter | Type | Description |  |
| Body | from | Text | From user id | Required |
|  | to | Text | To user id | Required |
|  | messageIndex | Text | Message index ids | Required |
| End |  |  |  |  |

Request Body:

{

"from": "10000231",

“to”:”10000232”,

“messageIndex”:”1,2,3”

}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Response |  |  |  | |
| Name | Parameter | Description | |  |
| Body | code | Response code | | Required |
|  | msg | Response message | | Required |
| End | | | | |

Response Body:

{

"code": "1",

"msg": "OK”

}

6. [HttpMSC] Delete chat session

<http://10.10.135.140.9000/v1/deletechatsession>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Request Method: POST | | | | |
| Name | Parameter | Type | Description |  |
| Body | from | Text | From user id | Required |
|  | to | Text | To user id | Optional |
|  | type | Text | 0: delete session with all users;  1: delete session with to user id. | Required |
| End |  |  |  |  |

Request Body:

{

"from": "10000231",

“to”:”10000232”,

“type”:”0” // clear all my chat sessions

}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Response |  |  |  | |
| Name | Parameter | Description | |  |
| Body | code | Response code | | Required |
|  | msg | Response message | | Required |
| End | | | | |

Response Body:

{

"code": "1",

"msg": "OK”

}

[Response code-message]

|  |  |
| --- | --- |
| Response Code | Response Message |
| 1 | OK |
| 2 | ERROR |
| 3 | NotExists |
| 4 | Exception |
| 5 | NotFound |