YPCLOUD

MoteChat

API of MoteChat

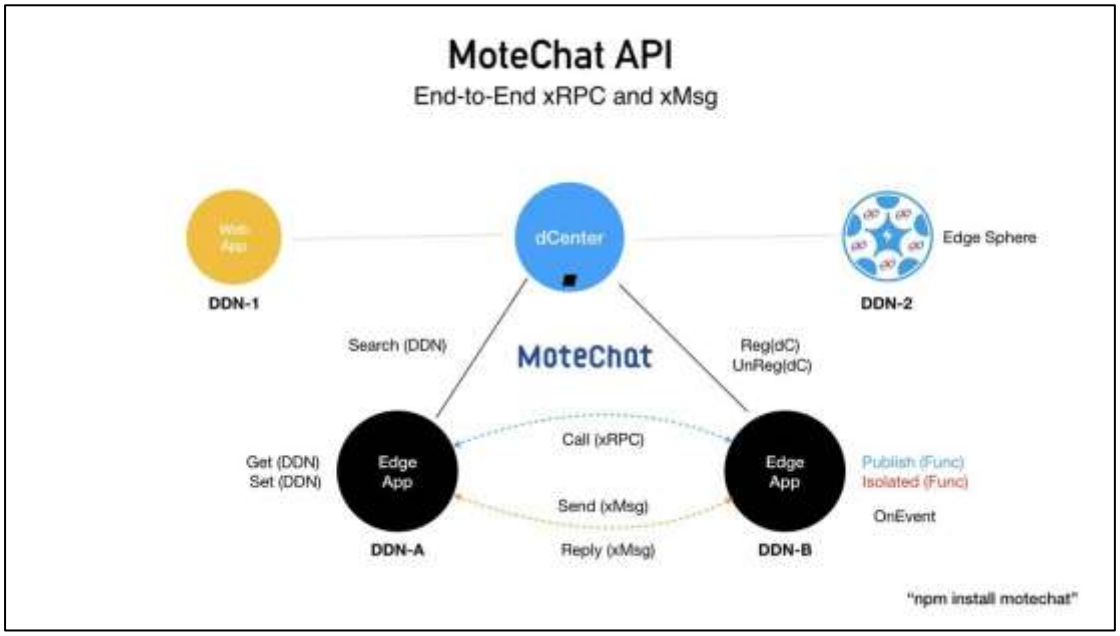
YPCloud Inc.

Last Updated : 2019/12/10

MoteMsg Cloud

Summary

System Diagram



List of API

Command	Description
Open	Open motechat
Close	Close motechat
Publish	Publish function
Isolated	Publish isolated function
Reg	Register to device center
UnReg	Un-register from device center
Call	Call function of another device
Send	Send message to another device
Get	Get the information of my device
Set	Set the device information of my device
Search	Search device by key
Nearby	Search nearby device
mbCall	Call function of another device by motebus
mbSend	Send message to another device by motebus
OnEvent	Set event handler

Function List

Command	Function	Status
Open	mChat.Open()	Updated
Close	mChat.Close()	
Publish	mChat.Publish()	
Isolated	mChat.Isolated()	
Reg	mChat.Reg()	
UnReg	mChat.Unreg()	
Call	mChat.Call()	
Send	mChat.Send()	
Get	mChat.Get()	
Set	mChat.Set()	
Search	mChat.Search()	
Nearby	mChat.Nearby()	
mbCall	mChat.mbCall()	Add
mbSend	mChat.mbSend()	Add
OnEvent	mChat.OnEvent()	Updated

Command

Open

Input:

conf: the configuration object for init:

{ "AppName":"", "AppKey":"","DCenter":"", "IOC":"","MotebusGW":"" }

AppName: the name of motebus MMA

AppKey: the key string of app

DCenter: the MMA of device center

IOC: the MMA of IOC

MotebusGW: the ip and port of motebus gateway, default is 127.0.0.1:6161

reg: the information of register for auto reg (option), the info of reg to DC

EiToken: device token

SToken: app token

WIP: WAN IP

LIP: LAN IP

EdgeInfo: (option), the information object of edge device

EiName: device name

EiType: device type

EiTag: device tag

EiLoc: device location

cb: callback ({ErrCode, ErrMsg} or {ErrCode, ErrMsg, result})

Example 1:

```
var conf = { "AppName":"", "AppKey":"","DCenter":"","IOC":"",
"MotebusGW":"127.0.0.1:6161" }
conf.AppName = 'myfunc';
conf.DCenter = 'dc@dc.ypcloud.com:6789';
conf.AppKey = 'YfgEeop5';
var mChat = require('motechat');
mChat.Open(conf, function(result){
    console.log('init result=%s', JSON.stringify(result));
}
```

Example 2: reg to DC directly

```
var conf = { "AppName":"", "AppKey":"","DCenter":"","IOC":"",
```

```
"MotebusGW":"127.0.0.1:6161", "UseWeb":"" }
conf.AppName = 'myfunc';
conf.DCenter = 'dc@dc.ypcloud.com:6789';
conf.AppKey = 'YfgEeop5';
var reginfo = {"EiToken":"8dilCCKj","SToken":"baTi52uE","WIP":"","LIP":""};
var mChat = require('motechat');
mChat.Open(conf, reginfo, function(result){
    console.log('init result=%s', JSON.stringify(result));
}
```

Example 3: reg to DC and set EI info directly

```
var conf = { "AppName":"", "AppKey":"","DCenter":"","IOC":"",
"MotebusGW":"127.0.0.1:6161", "UseWeb":"" }
conf.AppName = 'myfunc';
conf.DCenter = 'dc@dc.ypcloud.com:6789';
conf.AppKey = 'YfgEeop5';
var ei = {"EiName":"aifunc","EiType":".func","EiTag":"#ai","EiLoc":""}
var reginfo = {"EiToken":"8dilCCKj","SToken":"baTi52uE",
"WIP":"","LIP":"","EdgeInfo":ei};
var mChat = require('motechat');
mChat.Open(conf, reginfo, function(result){
    console.log('init result=%s', JSON.stringify(result));
}
```

Close

Input:

cb: callback({ErrCode, ErrMsg })

Publish

Input:

app: the name of name

func: the user function entry which is published

cb: callback({ErrCode, ErrMsg})

Example:

```
var app = 'motechat';
var XrpcMcService = {
    "echo": function(head, body){
        console.log("xrpc echo: head=%s", JSON.stringify(head));
        if ( typeof body == 'object')
            sbody = JSON.stringify(body);
        else
            sbody = body;
        console.log("xrpc echo: body=%s", sbody);
        return {"echo":body};
    }
}
mChat.Publish( app, XrpcMcService, function(result){
  console.log('motechat publish: result=%s', JSON.stringify(result));
});
```

Isolated

Input:

func: the user function entry which is published

cb: callback({ErrCode, ErrMsg})

Example:

```
var XrpcMcSecService = {
    "echo": function(head, body){
        console.log("xrpc echo: head=%s", JSON.stringify(head));
        if ( typeof body == 'object')
            sbody = JSON.stringify(body);
        else
            sbody = body;
        console.log('xrpc echo: body=%s', sbody);
        return {"echo":body};
    }
}
mChat.Isolated( XrpcMcSecService, function(result){
  console.log('motechat isolated: result=%s', JSON.stringify(result));
});
```

Reg

Input:

data: the information for registration, { "EiToken":"","SToken":"","WIP":""}

EiToken: device token

SToken: app token

WIP: WAN ip (empty means the same as dc)

cb: callback({ErrCode, ErrMsg, result} or {ErrCode, ErrMsg})

Example:

```
var mydev = {"EiToken":"8dilCCKj","SToken":"baTi52uE","WIP":""};
mChat.Reg(mydev, function(result){
    console.log('StartSession result=%s', JSON.stringify(result));
});
```

Note: At first time of the device, EiToken and SToken is empty.

UnReg

Input:

data: the information for registration, { "SToken":"" }

SToken: app token

cb: callback({ErrCode, ErrMsg})

Example:

```
var mydev = {"SToken":"baTi52uE"};
mChat.UnReg(mydev, function(result){
    console.log('EndSession result=%s', JSON.stringify(result));
});
```

Call

Input:

xrpc: xrpc control object, { "SToken":"", "DDN":"", "Topic":"", "Func":"", "Data":{},"SendTimeout": 6,"WaitReply": 12 }

SToken: app token

DDN: DDN of device

Topic: topic of app

Func: the function name

Data: the data object for function

SendTimeout: Integer, Timeout of send message, by sec.

WaitReply: Integer, The wait time of reply, by sec.

cb: callback({ErrCode, ErrMsg}) or callback(reply)

reply:

{"IN":{"From":{"DDN":"","Name":"","Type":"","Uid":"","Topic":""},"To":{"DDN":"","Name":"","Type":"","Topic":""},"State":{"ErrCode":0,"ErrMsg":"OK","By":""}},"Reply":{"ErrCode":0,"ErrMsg":"OK"}}

Example 1:

```
var ddn = 'kvGuHVUy';
var topic = 'flow/echo';
var func = 'echo';
var data = {"time":"2018/4/24 10:12:08"};
var t1 = 6;
var t2 = 12;
var xrpc = {"SToken":mydev.SToken, "DDN":ddn, "Topic":topic,
"Func":func,"Data":data, "SendTimeout":t1, "WaitReply":t2};
mChat.Call( xrpc, function(reply){
    console.log('CallSession reply=%s', JSON.stringify(reply));
});
```

Send

Input:

xmsg: xmsg control object, { "SToken":"", "DDN":"", "Topic":"", "Data":{},
"SendTimeout": 6,"WaitReply": 12 }

SToken: app token

DDN: DDN of device

Topic: the app topic

Data: the data which want to be sent

SendTimeout: Integer, Timeout of send message, by sec.

WaitReply: Integer, The wait time of reply, by sec.

cb: callback({ErrCode,ErrMsg}) or callback(reply)

reply:

{"IN":{"From":{"DDN":"","Name":"","Type":"","Uid":"","Topic":""},"To":{"DDN":"","Name":"","Type":"","Topic":""},"State":{"ErrCode":0,"ErrMsg":"OK","By"

:""}},"Reply":{"ErrCode":0,"ErrMsg":"OK"}}

Example 1:

```
var stoken = mydev.SToken;
var ddn = 'kvGuHVUy';
var topic = 'flow/msg';
var data = {"message":"Hello World"};
var t1 = 6;
var t2 = 12;
var xmsgctl = {"SToken":stoken, "DDN":ddn, "Topic":topic,"Data":data,
"SendTimeout":t1,"WaitReply":t2};
mChat.Send(xmsgctl, function(reply){
    console.log('sendxmsg reply=%s', JSON.stringify(reply));
});
```

Get

Input:

data: the input data object, { "SToken":"" }

SToken: app token

cb: callback({ErrCode, ErrMsg}) or callback(reply)

Example:

```
var data = {"SToken":mydev.SToken};
mChat.Get(data, function(result){
    console.log('GetDeviceInfo result=%s', result);
});
```

Set

Input:

data: input data object, { "SToken":"", "EdgeInfo":{} }

SToken: app token

EdgeInfo: {"EiName":"","EiType":"","EiTag":"","EiLoc":""}

cb: callback({ErrCode, ErrMsg}) or callback(reply)

Example:

```
var info = {"EiName":"myEi","EiType":".ei","EiTag":"#my","EiLoc":""};
var data = {"SToken":mydev.SToken,"EdgeInfo":info};
mChat.Set(data, function(result){
    console.log('SetDeviceInfo result=%s', result);
});
```

Search

Input:

data: input data object, { "SToken":"","Keyword":""}

SToken: app token

Keyword: keyword for search

Local: search local [true | false], default is false

cb: callback({ErrCode, ErrMsg}) or callback(reply)

Example:

```
var data = {"SToken":mydev.SToken, "Keyword":"#test"};
mChat.Search(data, function(result){
    console.log('Search result=%s', result);
});
```

Nearby

Input:

data: input data object, { "SToken":""}

SToken: app token

Local: search local [true | false], default is false

cb: callback({ErrCode, ErrMsg}) or callback(reply)

Example:

```
var data = {"SToken":mydev.SToken};
mChat.Nearby(data, function(result){
    console.log('Search result=%s', result);
});
```

mbCall

Input:

xrpc: xrpc control object, { "MMA":"", "Func":"", "Data":{},"SendTimeout":6,"WaitReply": 12 }

MMA: mma of target device

Func: the function name

Data: the data object for function

SendTimeout: Integer, Timeout of send message, by sec.

WaitReply: Integer, The wait time of reply, by sec.

cb: callback({ErrCode, ErrMsg}) or callback(reply)

Example 1:

```
var mma = 'hello@192.168.1.10';
var func = 'echo';
var data = {"time":"2018/4/24 10:12:08"};
var t1 = 6;
var t2 = 12;
var xrpc = {"MMA":mma, "Func":func,"Data":data, "SendTimeout":t1, "WaitReply":t2};
mChat.mbCall( xrpc, function(reply){
    console.log('mbCall reply=%s', JSON.stringify(reply));
});
```

mbSend

Input:

xmsg: xmsg control object, { "MMA":"", "Data":{},"SendTimeout":6,"WaitReply": 12 }

MMA: mma of target device

Data: the data object for function

SendTimeout: Integer, Timeout of send message, by sec.

WaitReply: Integer, The wait time of reply, by sec.

cb: callback({ErrCode, ErrMsg}) or callback(result)

Example 1:

```
var mma = 'hello@192.168.1.10';
var data = {"degree":30};
var t1 = 6;
var t2 = 12;
var xmsg = {"MMA":mma, "Data":data, "SendTimeout":t1, "WaitReply":t2};
mChat.mbSend( xmsg, function(result){
    console.log('mbSend reply=%s', JSON.stringify(result));
});
```

OnEvent

Input:

stype: string

"message" is for get msg of motechat

"state" is for state changed

"mbus" is for get msg of motebus

cb: callback function

"message": function(channel, in, data, cb)

"state": function(state, ddn)

"mbus": function(from, data, cb)

webtype: string

default: ""

web: "wsocket"

Output:

return is boolean (true or false)

Example:

```
var InmsgRcve = function(ch, inctl, data, retcb){
    console.log('InmsgRcve: channel=%s, from=%s, to=%s, data=%s', ch,
    JSON.stringify(inctl.From), JSON.stringify(inctl.To), JSON.stringify(data));
    if ( typeof retcb == 'function') retcb({"ErrCode":0, "ErrMsg":"OK"})
}
var InState = function(state, ddn){
    if ( ddn ) console.log('InState=%s, ddn=%s', state, ddn);
    else console.log('InState=%s', state);
}
```

```
var mbusRcve = function(from, data, retcb){
    console.log('mbusRcve: from=%s, data=%s', from, JSON.stringify(data));
    if ( typeof retcb == 'function') retcb({"ErrCode":0, "ErrMsg":"OK"})
}
mChat.OnEvent('message',InmsgRcve);
mChat.OnEvent('state', InState);
mChat.OnEvent('mbus', mbusRcve)
```

System Topic

Use UC function by mms call

Input:

xrpc: xrpc control object, { "SToken":"", "DDN":"", "Topic":"", "Func":"", "Data":{},"SendTimeout": 6,"WaitReply": 12 }

- SToken: app token
- DDN: ">>uc"
- Topic: topic of UC (note1)
- Func: function name of UC (note1)
- Data: the data object for function (note1)
- SendTimeout: Integer, Timeout of send message, by sec.
- WaitReply: Integer, The wait time of reply, by sec.

cb: callback({ErrCode, ErrMsg})

Example 1:

```
let ddn = '>>uc';
let topic = 'uc://setuserinfo';
let func = 'cmd;
let data = {"UserInfo":{"MobileNo":"0936123123","NickName":"judy","Sex":0}};
let t1 = 6;
let t2 = 12;

let xrpc = {"SToken":mydev.SToken, "DDN":ddn, "Topic":topic,
"Func":func,"Data":data, "SendTimeout":t1, "WaitReply":t2};
mChat.Call( xrpc, function(reply){
    console.log('UCFunc set info reply=%s', JSON.stringify(reply));
});
```

Note 1:

No	Topic	Func	Data
1	uc://login	cmd	{UserName, Password, KeepLogin}
2	uc://logout	cmd	{}
3	uc://signup	cmd	{UserName, Password, UserInfo} (note 2)

4	uc://checkuser	cmd	{UserName}
5	uc://getuserinfo	cmd	{}
6	uc://setuserinfo	cmd	{UserInfo} (note 2)
7	uc://getusersetting	cmd	{KeyName}
8	uc://setusersetting	cmd	{KeyName, Setting} (note 3)

Note 2: UserInfo: {MobileNo, NickName, FirstName, LastName, Sex}

Note 3: Setting: Application defined

Note 4: UserName: E-mail address of user

Note 5: KeepLogin: true or false

Use IN function by mms send

Input:

xmsg: xmsg control object, { "SToken":"", "DDN":"", "Topic":"", "Data":{},"SendTimeout": 6,"WaitReply": 12 }

SToken: app token

DDN: ">>sys"

Topic: topic of IN (note1)

Data: the data object for function (note1)

SendTimeout: Integer, Timeout of send message, by sec.

WaitReply: Integer, The wait time of reply, by sec.

cb: callback({ErrCode, ErrMsg})

Example 1:

```
let ddn = '>>sys';
let topic = 'in://mnL6QHsd ';
let data = 'ping';
let t1 = 6;
let t2 = 12;

let xmsg= {"SToken":mydev.SToken, "DDN":ddn, "Topic":topic, "Data":data,
"SendTimeout":t1, "WaitReply":t2};
mChat.Send( xmsg, function(reply){
    console.log('IN ping reply=%s', JSON.stringify(reply));
});
```

Note 1:

No	Topic	Data	Description
1	in://local	ping	Ping to local motechat
2	in://dc	ping	Ping to DC
3	in://mnL6QHsd	ping	Ping to device which DDN is mnL6QHsd
4	in://local	trace	Trace to local motechat
5	in://dc	trace	Trace to DC
6	in://mnL6QHsd	trace	Trace to device which DDN is mnL6QHsd

Use XShare function by mms call

Input:

xrpc: xrpc control object, { "SToken":"", "DDN":"", "Topic":"", "Func":"", "Data":{},"SendTimeout": 6,"WaitReply": 12 }

SToken: app token

DDN: ">>sys"

Topic: topic of XShare (note1)

Func: function name of XShare (note1)

Data: the data object for function (note1)

SendTimeout: Integer, Timeout of send message, by sec.

WaitReply: Integer, The wait time of reply, by sec.

cb: callback({ErrCode, ErrMsg})

Example 1:

```
let ddn = '>>sys';
let topic = 'xs://config';
let func = 'get';
let data = {"catalog":"myapp","idname":"userinfo"};
let t1 = 6;
let t2 = 12;

let xrpc = {"SToken":mydev.SToken, "DDN":ddn, "Topic":topic,
"Func":func,"Data":data, "SendTimeout":t1, "WaitReply":t2};
mChat.Call( xrpc, function(reply){
    console.log('XShare get config reply=%s', JSON.stringify(reply));
});
```

Note 1:

Topic, Function and Arguments

No	Topic	Func	Data
1	xs://config	get	{catalog, idname}
2	xs://config	set	{catalog,idname, data}
3	xs://cached	get	{catalog, idname}
4	xs://cached	set	{catalog, idname, data}
5	xs://cached	remove	{catalog, idname}
6	xs://cached	clear	{catalog}
7	xs://bucket	get	{catalog, idname, datatype} *note2
8	xs://bucket	set	{catalog, idname, data}
9	xs://bucket	list	{catalog}
10	xs://bucket	remove	{catalog, idname}
11	xs://secret	get	{catalog, idname, password}
12	xs://secret	set	{catalog, idname, data, password}

Note2:

datatype: 'hex', 'ascii', 'utf8', 'base64', or 'binary'

Error Code

No	Code	Message
1	0	OK
2	-10199	in error
3	-10101	in: open XRPC error
4	-10102	in: XRPC not open
5	-10103	in: motebus not open
6	-10104	in: send error
7	-10105	in: open XMsg error
8	-10106	in: invalid data
9	-10299	motechat error
10	-10201	motechat: no DC setting
11	-10202	motechat not open
12	-10203	motechat: invalid data
13	-10204	motechat: invalid stoken
14	-10205	motechat: no rcve function
15	-10306	motechat: no matched DDN
16	-10207	motechat: no DDN
17	-10399	wsocket error
18	-10301	wsocket: invalid data
19	-10302	wsocket: no socket id