DEVELOPERS

WebSocket Market Data API

BTCC Socket.io API#

Socket.io API is open for public use. No authentication is required to use ticker and trade metho
authentication is rquired for order method. To use this API, you need to setup your computer w
socket.io equipped beforehand. You can find useful links here:

- http://socket.io/ (http://socket.io/)
- https://www.npmjs.org/package/socket.io (https://www.npmjs.org/package/socket.io)

Make sure you include socket.io somewhere within your code, and all sample codes in this articl
written in Javascript unless specified :

```
<script src="<YOUR PATH>/socket.io.js"></script>
```

Establish the connection with our websocket server using :

```
var socket = io ('https://websocket.btcc.com/');
```

Subscribe to the websocket server and be ready to receive the real-time data pushed by the ser

```
socket.emit('subscribe', 'marketdata_cnybtc');
socket.emit('subscribe', 'marketdata_cnyltc');
socket.emit('subscribe', 'marketdata_btcltc');
socket.emit('subscribe', 'grouporder_cnybtc');
socket.emit('subscribe', 'grouporder_cnyltc');
socket.emit('subscribe', 'grouporder_btcltc');
```

Name Type Description
market_data string The market data of 'marketdata_cnybtc', 'marketdata_cnyltc' or 'marketdata_btcltc'(one or more separated by ”,”)
grouporder string The grouporder data of 'grouporder_cnybtc', 'grouporder_cnyltc' or 'grouporder_btcltc'(one or more separated by ”,”)

Ticker#

Listen on “ticker” method to receive and process the ticker data as follows:

```
socket.on('ticker', function (data) { console.log(data); });
```

```
## Sample received data ##
{
    buy: 2940.03
    date: 1410399073
    high: 2970
    last: 2940.04
    low: 2901
    market: "btccny"
    open: 2930.15
    prev_close: 2931.03
    sell: 2940.06
    vol: 15187.3352
    vwap: 2936.02
}
```

Trade#

Listen on “trade” method to receive and process the trade data as follows:

```
socket.on('trade', function (data) { console.log(data); });
```

```
## Sample received data ##
{
    amount: 0.056
    date: 1402970632
    market: "btccny"
    price: 3735.83
    trade_id: 6069941
    type: "sell"
}
```

Grouporder#

Grouporder will give the market depth feed with 5 pairs of top open 'ask' and 'bid' orders. Lister
“grouporder” method to receive and process the grouporder data as follows:

```
socket.on('grouporder', function (data) { console.log(data); });
```

```
## Sample received data ##
{
    "quot;grouporder":
  {
    "market":"btccny",
    "ask":[
    {"price":2405.04,"type":"ask","totalamount":0.1},
    {"price":2404.24,"type":"ask","totalamount":2.9544},
    {"price":2404.21,"type":"ask","totalamount":0.011},
    {"price":2404.02,"type":"ask","totalamount":0.011},
    {"price":2403.71,"type":"ask","totalamount":0.01}
    ],
    "bid":[
    {"price":2402.74,"type":"bid","totalamount":0.099},
    {"price":2401.11,"type":"bid","totalamount":6},
    {"price":2401.1,"type":"bid","totalamount":1.0014},
    {"price":2400.66,"type":"bid","totalamount":0.1},
    {"price":2400.65,"type":"bid","totalamount":0.1066}
    ]
  }
}
```

Order#

Subscribe with 'private' method and listen on “order” method to receive and process your own c
data. Whenever the status of your own orders change, the server will push corresponding order
client side. This method requires authentication with your access key and secret key, which is us
same way as our trade API. Please refer to the following code in the Java sample code part for th
method.

```
//Use 'private' method to subscribe the order and account_info feed
    List arg = new ArrayList();
    arg.add(sm.get_payload());
    arg.add(sm.get_sign());
    socket.emit("private",arg);
```

```
## Sample received data ##
{
    "amount": 0
    "id": 3804213
    "price": 33.51
    "market": "ltccny"
    "status": "closed"
    "date":1410400468
    "type":"ask"
    "amount_original":1
}
```

Account_info Subscribe with 'private' method and listen on “account_info” method to receive an
process your balance data. Whenever the amount of your balance changes, the server will push
corresponding information to your client side. This method requires authentication with your ac
and secret key, which is used the same way as our trade API. Please refer to the following code i
Java sample code part for this method. //Use 'private' method to subscribe the order and accou
feed List arg = new ArrayList(); arg.add(sm.get_payload()); arg.add(sm.get_sign());
socket.emit("private",arg);

Sample received data#

{ "balance": { "amount":0.03490866, "symbol":"฿", "currency":"BTC" } } { "balance": { "amount":10, "symbol":"¥", "currency":"CNY" } }

Code Examples#

For more code samples of our Socket.io client, see the following links:
Python: https://github.com/BTCChina/btcchina-websocket-api-python (https://github.com/BTCChina/btcchina-websocket-api-python)
Java: https://github.com/BTCChina/btcchina-websocket-api-java (https://github.com/BTCChina/b websocket-api-java)
Ruby: https://github.com/BTCChina/btcchina-websocket-api-ruby (https://github.com/BTCChina/btcchina-websocket-api-ruby)
JavaScript: https://github.com/BTCChina/btcchina-websocket-api-js (https://github.com/BTCChina/btcchina-websocket-api-js)
C++: https://github.com/BTCChina/btcchina-api-cpp (https://github.com/BTCChina/btcchina-api-
C#: https://github.com/BTCChina/btcchina-api-csharp (https://github.com/BTCChina/btcchina-ap
Note that, in case you find any bugs from these clients, please kindly inform us and submit bugs developers of the corresponding clients in Github if possible.

JAVASCRIPT#

```
<script src="./js/socket.io.js"></script>
<script>
var socket = io('https://websocket.btcchina.com/');
socket.emit('subscribe', 'marketdata_cnybtc');
socket.emit('subscribe', 'marketdata_cnyltc');
socket.emit('subscribe', 'marketdata_btcltc');
socket.emit('subscribe', 'grouporder_cnybtc');
socket.emit('subscribe', 'grouporder_cnyltc');
socket.emit('subscribe', 'grouporder_btcltc');
socket.on('connect', function(){
socket.on('trade', function (data) {
console.log(data);
});
socket.on('ticker', function (data) {
console.log(data);
});
socket.on('grouporder', function (data) {
console.log(data);
});
});
</script>
```

JAVA#

```
  //* An example for Java Socket.IO Client
import com.github.nkzawa.emitter.Emitter;
import com.github.nkzawa.socketio.client.IO;
import com.github.nkzawa.socketio.client.Socket;
import java.net.URISyntaxException;
import java.util.ArrayList;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import javax.xml.bind.DatatypeConverter;
import org.json.JSONObject;
public class SocketMain {
private String ACCESS_KEY="YOUR_ACCESS_KEY";
private String SECRET_KEY="YOUR_SECRET_KEY";
private static String HMAC_SHA1_ALGORITHM = "HmacSHA1";
private String postdata="";
private String tonce = ""+(System.currentTimeMillis() * 1000);
public static void main(String[] args) throws Exception {
try {
IO.Options opt = new IO.Options();
opt.reconnection = true;
Logger.getLogger(SocketMain.class.getName()).setLevel(Level.FINE);
final Socket socket = IO.socket("https://websocket.btcc.com", opt);
socket.on(Socket.EVENT_CONNECT, new Emitter.Listener() {
SocketMain sm= new SocketMain();
@Override
public void call(Object... args) {
System.out.println("connected");
socket.emit("subscribe", "marketdata_cnybtc"); // subscribe
socket.emit("subscribe", "marketdata_cnyltc"); // subscribe another market
socket.emit("subscribe", "marketdata_btcltc"); // subscribe another market
socket.emit("subscribe", "grouporder_cnybtc"); // subscribe grouporder
socket.emit("subscribe", "grouporder_cnyltc"); // subscribe another market
socket.emit("subscribe", "grouporder_btcltc"); // subscribe another market
//Use 'private' method to subscribe the order and account_info feed
```

```
    try {
    List arg = new ArrayList();
    arg.add(sm.get_payload());
    arg.add(sm.get_sign());
    socket.emit("private",arg);
    } catch (Exception e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
    }
    }
    }).on("trade", new Emitter.Listener() {
    @Override
    public void call(Object... args) {
    JSONObject json = (JSONObject) args[0]; //receive the trade message
    System.out.println(json);
    }
    }).on("ticker", new Emitter.Listener() {
    @Override
    public void call(Object... args) {
    JSONObject json = (JSONObject) args[0];//receive the ticker message
    System.out.println(json);
    }
    }).on("grouporder", new Emitter.Listener() {
    @Override
    public void call(Object... args) {
    JSONObject json = (JSONObject) args[0];//receive the grouporder message
    System.out.println(json);
    }
    }).on("order", new Emitter.Listener() {
    @Override
    public void call(Object... args) {
    JSONObject json = (JSONObject) args[0];//receive your order feed
    System.out.println(json);
    }
    }).on("account_info", new Emitter.Listener() {
    @Override
    public void call(Object... args) {
    JSONObject json = (JSONObject) args[0];//receive your account_info feed
    System.out.println(json);
    }
    }).on(Socket.EVENT_DISCONNECT, new Emitter.Listener() {
    @Override
    public void call(Object... args) {
    System.out.println("disconnected");
    }
    });
    socket.connect();
    } catch (URISyntaxException ex) {
    Logger.getLogger(SocketMain.class.getName()).log(Level.SEVERE, null, ex);
    }
    }
    public String get_payload() throws Exception{
    postdata = "{\"tonce\":\""+tonce.toString()+"\",\"accesskey\":\""+ACCESS_KEY+"\",\"requestm
 \"post\",\"id\":\""+tonce.toString()+"\",\"method\": \"subscribe\", \"params\": [\"order_cnylt
count_info\"]}";//subscribe order feed for cnyltc market and balance feed
    System.out.println("postdata is: " + postdata);
    return postdata;
    }
    public String get_sign() throws Exception{
    String params = "tonce="+tonce.toString()+"&accesskey="+ACCESS_KEY+"&requestmethod=post&id="
toString()+"&method=subscribe&params=order_cnyltc,account_info"; //subscribe the order of cnylt
 and the account_info
    String hash = getSignature(params, SECRET_KEY);
    String userpass = ACCESS_KEY + ":" + hash;
    String basicAuth = DatatypeConverter.printBase64Binary(userpass.getBytes());
    return basicAuth;
    }
    public String getSignature(String data,String key) throws Exception {
    // get an hmac_sha1 key from the raw key bytes
    SecretKeySpec signingKey = new SecretKeySpec(key.getBytes(), HMAC_SHA1_ALGORITHM);
    // get an hmac_sha1 Mac instance and initialize with the signing key
    Mac mac = Mac.getInstance(HMAC_SHA1_ALGORITHM);
    mac.init(signingKey);
    // compute the hmac on input data bytes
    byte[] rawHmac = mac.doFinal(data.getBytes());
    return bytArrayToHex(rawHmac);
    }
    private String bytArrayToHex(byte[] a) {
    StringBuilder sb = new StringBuilder();
    for(byte b: a)
    sb.append(String.format("%02x", b&0xff));
    return sb.toString();
    }
    }
```

```</br>
```

```
## Ruby
```

Require installing socket.io-client-simple#

```
require 'socket.io-client-simple'
require 'base64'
require 'json'
$access_key = "<YOUR ACCESS KEY>"
$secret_key = "<YOUR SECRET KEY>"
def initial_post_data
post_data = {}
post_data['tonce'] = (Time.now.to_f * 1000000).to_i.to_s
post_data
end
def params_string(post_data)
post_data['params'] = post_data['params'].join(',')
params_parse(post_data).collect{|k, v| "#{k}=#{v}"} * '&'
end
def params_parse(post_data)
post_data['accesskey'] = $access_key #access key
post_data['requestmethod'] = 'post'
post_data['id'] = post_data['tonce'] unless post_data.keys.include?('id')
fields=['tonce','accesskey','requestmethod','id','method','params']
ordered_data = {}
fields.each do |field|
ordered_data[field] = post_data[field]
end
ordered_data
end
def sign(params_string)
signature = OpenSSL::HMAC.hexdigest(OpenSSL::Digest::Digest.new('sha1'), $secret_key, params_st
ecret key
Base64.strict_encode64($access_key+ ':' + signature)
end
socket = SocketIO::Client::Simple.connect 'https://websocket.btcc.com'
socket.on:connect do
puts "connected!"
socket.emit :subscribe, "marketdata_cnybtc"
socket.emit :subscribe, "marketdata_cnyltc"
socket.emit :subscribe, "marketdata_btcltc"
socket.emit :subscribe, "grouporder_cnybtc"
socket.emit :subscribe, "grouporder_cnyltc"
socket.emit :subscribe, "grouporder_btcltc"
post_data = initial_post_data
post_data['method'] = 'subscribe'
post_data['params'] = ["order_cnybtc", "order_cnyltc", "order_btcltc",               "account_i
payload = params_parse(post_data)
pstr = params_string(payload.clone)
signature_string = sign(pstr)
socket.emit :private, [payload.to_json, signature_string]
end
socket.on :disconnect do
puts "disconnected!"
end
socket.on :message do |data|
puts "message: "+data
end
socket.on :trade do |data|
puts 'trade:'
p data
end
socket.on :ticker do |data|
puts 'ticker:'
p data
end
socket.on :grouporder do |data|
puts 'grouporder:'
p data
end
socket.on :order do |data|
puts 'order:'
p data
end
socket.on :account_info do |data|
puts 'account_info:'
p data
end
loop do
sleep 3
end
```

```

## Socket.io API v1.2.3#
```

2014-10-27 Change title from Websocket API to Socket.io API .

Socket.io API v1.2.2#

2014-10-15 Socket.io API v1.2.2 Added 'grouporder' feed.

Socket.io API v1.2.1#

2014-09-30 Socket.io API v1.2.1 Added 'account_info', added Java sample code for 'account_info' and Ruby sample code for 'order' and 'account_info' methods.

Socket.io API v1.2#

2014-09-11 Socket.io API v1.2 Added C# and C++ sample code; added 'ticker' and 'order' methoc Java sample code for 'order' method.

Socket.io API v1.1#

2014-07-30 Socket.io API v1.1 more sample codes links to Github are added, including Java, Pyth and JavaScript.

Socket.io API v1.0#

Socket.io API v1.0 includes trade method and JavaScript sample codes.

PRODUCTS
Bitcoin Pro Exchange (https://pro.btcc.com)
Bitcoin Exchange (https://spot.btcc.com)
Bitcoin Mining Pool (https://pool.btcc.com)
Bitcoin Wallet (https://pay.btcc.com)
Blockchain Engraving (https://forever.btcc.com)
Mint Physical Bitcoins (https://mint.btcc.com)
Developer Platform

ABOUT
BTCC (https://www.btcc.com/about)
Team (https://www.btcc.com/team)
News (https://www.btcc.com/news)
Jobs (https://www.btcc.com/jobs)
Contact Us (https://www.btcc.com/contact)
Security (https://www.btcc.com/security)

RESOURCES
Account (https://www.btcc.com/account)
Support (https://www.btcc.com/contact)
Fees (https://www.btcc.com/fees)
Legal & Privacy (https://www.btcc.com/legal)

English

(https://(https