INTERACTIVE BROKERS

WT Web API for iBrokers

User Documentation

Version 2.0 - September 18, 2015

Contents

WT Web API

1. Introduction

The purpose is to provide third parties with a web-based interface that supports:

- User authentication.
- Market data subscriptions.
- Chart subscription.

2. Implementation

This API uses window.postMessage that allows third parties to get data from the IB Server.

- The third party website loads a page from the IB site inside an iframe.
- The API implements a bi-directional postMessage cross-domain communication.
- The postMessage message source must be validated on both sides.

Requirements:

- Browser must support window.postMessage (IE9+, FF31+, Chrome31+, Safari7+).
- IBroker application must be implemented as a single page application.
- Websocket support (IE10+, FF31+, Chrome31+, Safari7+).

3. Supported Calls and their Responses

The communication between the iframe and the parent is made using PostMessage messages. Both incoming and outgoing messages are JSON objects.The data JSON Object will contain an ACTION property that identifies which package should handle the message. We use the following ACTION Values:

- MD and CONTRACT for market data.
- CHART for chart data.

The output package could also contain error codes in an ERRORS property (array type).

Note: In case of IE9, you have to use JSON.parse() and JSON.stringify().

Examples provided with some of the calls in later sections.

4. Third Party Calls to IB

Prerequisites:

Each IBroker needs to:

- Have a white branded ID (string) that is validated and known to IB.
- Know how long the server should keep the session (minutes, integer between 1 and 30).
- Have an iframe in the DOM (to display the login form).
- Select a language for the login form (available: en, cn, de, es, fr, it, jp, nl, ru, tw).

Requests

Third party makes a call to IB API, and that API invokes call to the backend that generates a response that is sent back to the caller.

The sample implementation of the communication of a third party with the IB server is by using the module wtapi.js (Appendix A). This module wtapi.js can be accessed with the methods of window.WTAPIobject. This module is standalone, does not require any additional library to work.

The communication is asynchronous and implemented with PostMessage messages.

4.1 Initialization

1. Load the IB page with GET parameters from the IB server to the iframe. The parameters are the following:

- WB_ID <string> : white branded ID
- lang<string> : (optional) display language of the login form

Ex: https://zh.wgw.interactivebrokers.com/wtapibeta/?WB_ID=Asterix&lang=en

Once the iframe loads, it sends a message to parent:

{ method: 'loaded'}

If IB server is down or being restarted, no response is sent back to parent.

2. Next an INIT call is made to iframe:

{ method: 'init', wb_id: <string>, max_timeout: <number> }

Response:

```
{ method: 'initialized'}
```

If the INIT call is made successfully, the frame URL and white-branded ID are validated and the login page within the iframe is shown.

If INIT is not successful, one of the following messages is sent back to parent:

```
{ method: 'message', message: 'Initialization failed', ERRORS: <Array> }
```

Potential Error Codes
WTE099
WTE018
WTE019
WTE023

4.2 Authentication

Authentication is done within the iframe. If login succeeds, the following messages are sent back to the parent:

```
{ method: 'message', message: 'Logged in' }

{ method: 'setPing', timeout: 60000 }

{ method: 'data', message:
{"ACCOUNTS":"DU91474","RESULT":true,"MESSAGE_TYPE":"FORWARD","ID":" d53d96e0-0095-4ef0-bc33-2b7117d6ca03","ERRORS":[],"USER":"qapap908"}}
```

If user authentication fails, user cannot proceed beyond the login page.

4.2 Ping

A ping message needs to be sent to the iframe every 1 minute (defined as timeout below). The ping should start after a successful login and stops when thelogout() method is called.

If ping stops and the max_timeout defined in Section 4.1 is reached, the server will clean out the session and no market data or chart data will be sent from the server.

After successful login, the iframe sends a ping message as below:
```
{ method: 'setPing', timeout: 60000 }
```

After that the parent needs to send a ping message to iframe, at the timeout interval defined above.

The parent frame has to send the following message periodically:
```
{ method: 'ping' }
```

4.3 Market Data Subscribe

To subscribe to market data the application should call the 'mdSubscribe' method with a conid-exchange-pairs parameter. This parameter can contain one or more conid-exchange pairs separated by a semicolon character. Ex: 8314:SMART;43645865:SMART

Input below:
```
{ method: 'mdSubscribe', md: <string> }
```

After a successful subscribe, the data callback function will be called. For market data, two types of ACTIONS are relevant:

- CONTRACT with static data describing the contract properties and
- MD with prices and other market data attributes.

The data callback function will call the correct package based on the ACTION property in the message. See parse MarketDataPackage() function in client.js for detailed info.

```
{ method: 'data',
message :{"RESULT":true,"MESSAGE_TYPE":"FORWARD","symbol":"IBM","sec_type":"STK",
"ACTION":"CONTRACT","exchange":"SMART","currency":"USD","conid":8314}}

{ method: 'data', message:
```

{"MESSAGE_TYPE":"FORWARD","ACTION":"MD","MD":[{"change_price":"-1.72","bid_size":"2","last_price":"145.65","is_delayed":false,"open_price":"147.37","change_percent":"-1.17%","conid":8314,"bid_price":"145.43","ask_price":"146.47","last_trading_day":"20150911","volume":"2.91M","close_price":"147.37","dividend_yield":"3.6%","high":"147.37","low":"145.41","exchange":"SMART","ask_size":"2"}]}}

If you subscribe to incorrect conid-exchange pair, you get an error message:
{method: "message", message: "Market Data subscription failed", ERRORS:["WTE020"]}

Potential Error Codes
WTE008
WTE020

Data fields returned after subscription in MD Action is:

MD Object key value pair of:

- con_id
- exchange
- change_percent
- change_price
- bid_size
- bid_price
- ask_size
- ask_price
- volume (in K, M, B)
- high
- low
- market_value
- average_price
- symbol
- company_name
- contract_description_1: contains the symbol in most cases
- last_price
- yield_last
- mark_price
- is_delayed(true/false): If user doesn’t have market data permission for this conid, exchange pair; but we have delayed market data available then this flag is true.
- open
- close
- open_price
- close_price
- dividend_yield

4.4 Market Data Unsubscribe

To unsubscribe market data, the app should call 'mdUnsubscribe' method with a conid-exchange-pairs parameter. This parameter can contain one or more conid-exchange pairs separated by a semicolon character. Ex: 8314:SMART;43645865:SMART

Input below:
```
{ method: 'mdSubscribe', md: <string> }
```

After unsubscribe, market data stops ticking for the unsubscribed conid-exchange pair to which you subscribed earlier.

If you unsubscribe to an incorrect conid-exchange pair, you get an error like below:
```
{method: "message", message: "Cancelling  Market Data subscription failed",
ERRORS:["WTE020"]}
```

Potential Error Codes
WTE008
WTE020

4.5 Chart Data Subscribe

To subscribe to chart data, the application should call the 'chartSubscribe' method with three parameters:

- The first parameter contains one (and only one) conid-exchange-pair.
- The second parameter is the length of the chart. This value can be selected from a predefined list (1d, 1w, 2w, 1m, 6m, 1y, 5y).
- The third is a boolean for “Outside Regular Trading Hours”.

Input below:
```
{ method: 'chartSubscribe', md: <string> , time: <string> , orth: <boolean> }
```

After a successful Chart subscribe, the data callback function called the chart packages is invoked. The CHART type packages contain the data in following format:

The chart data is sent to parent frame every 1 minute.

{ method: 'data', message: {"RESULT":true,"MESSAGE_TYPE":"BOTH","SESSION":"d53d96e0-0095-4ef0-bc33-2b7117d6ca03","data_points":[{"open":147.37,"time":1442237430000,"volume":607,"high":14

7.37,"low":147,"close":147.2},{"open":147.21,"time":1442237490000,"volume":19,"high":147.31,"low":147.18,"close":147.31},{"open":147.14,"time":1442237550000,"volume":81,"high":147.31,"low":147.06,"close":147.14},{"open":146.35,"time":1442238150000,"volume":53,"high":146.38,"low":146.24,"close":146.38},{"open":146.29,"time":1442238210000,"volume":23,"high":146.29,"low":146.14,"close":146.14},{"open":145.46,"time":1442257830000,"volume":18,"high":145.48,"low":145.44,"close":145.46},{"open":145.47,"time":1442257890000,"volume":71,"high":145.54,"low":145.47,"close":145.53}],"start_time":"2015091413:30:30","ACTION":"CHART"}}

If your chart subscribe fails, you get an error like below:
{ method: 'data', message: {"RESULT":false,"MESSAGE_TYPE":"BOTH","SESSION":"6008e42c-6e22-4c08-97a9-a092c6201517","ERRORS":["WTE021"],"ACTION":"CHART_UNSUBSCRIBE"}}

Potential Error Codes
WTE008
WTE021
WTE022

4.6 Chart Data Unsubscribe

To unsubscribe chart data the app should call the 'chartUnsubscribe' method with one (and only one) conid-exchange-pair. Once unsubscribe is called, the chart details stop being sent to parent every 1 minute.

Input:
{ method: 'chartUnsubscribe', md: <string> }

If chart unsubscribe fails, you will get message like below:
{ method: 'data', message: {"RESULT":false,"MESSAGE_TYPE":"BOTH","SESSION":"6008e42c-6e22-4c08-97a9-a092c6201517","ERRORS":["WTE021"],"ACTION":"CHART_UNSUBSCRIBE"}}

Potential Error Codes
WTE008
WTE021

4.7 Logout

To end interaction with the iframe and to clean up resources on the server, call the'logout' method.
Input below:
{ method: 'logout' }

On successful logout, the response is:
```
{ method: 'message', message: 'Logged out' }
```

After a successful logout the parent should unload the iframe (null the iframe source).

4.7 Competition

If the user is logged into any other IB trading application like Webtrader, TWS or another WTWebApi version with the same username, and tries to login into WTWebApi, there will be a competition message. This competition callback function is called with a message 'competition'. The callback function should display a popup message (with the native confirm or a custom one like jQuery Dialog) asking the user to either continue the session (and stop the competing session) or stop the current session login.

Competition message:
```
{ method: 'competition', message: <string> }
```

If the user wants to continue the current session, the app should call:
```
{ method: 'keepSession' }
```
To kill the current session, call the method:
```
{ method: 'logout' }
```

5. Unsolicited Messages from IB

All market data messages are sent to the parent once they arrive. On subscription of market data message, the server pushes every tick to the client, and the parent does not need to poll for it. If websocket is enabled for the browser and it is within regular trading hours, market data ticks every 250 ms. If the browser does not support websocket or if websocket is disconnected for any reason, market data is sent to the client every 3seconds.

```
{ method: 'data',
message:{"MESSAGE_TYPE":"FORWARD","ACTION":"MD","MD":[{"symbol":"IBKR","conidex":"4
3645865","company_name":"INTERACTIVE BROKERS GRO-CL
A","expiry_type":"0","sec_type":"STK","listing_exchange":"NASDAQ.NMS","contract_descriptio
n_1":"IBKR","exchange":"SMART","conid":43645865}]}}
{ method: 'data',
message:{"MESSAGE_TYPE":"FORWARD","ACTION":"MD","MD":[{"change_price":"-
1.90","bid_size":"3","last_price":"145.47","is_delayed":false,"open_price":"147.37","change_pe
rcent":"-
1.29%","market_value":"1,018","conid":8314,"mark_price":"145.47","bid_price":"145.47","ask_
price":"145.48","last_trading_day":"20150911","volume":"1.61M","close_price":"147.37","divid
```

end_yield":"3.6%","high":"147.37","low":"145.41","exchange":"SMART","ask_size":"8"}]}}

Additionally, the Chart message is sent to the parent frame every 1 minute by IBServer.

The following messages are sent from the server if there is any application problem. The client should either try to reconnect or show login window again.

WT Error	What to do
WTE002	Show the login window
WTE003	Show the login window
WTE006	Let user decide to end current session or disconnect other session; call force initialization accordingly
WTE008	Show the login window

6. Error Codes

All errors codes with descriptions. The last column indicates if the message can be seen in the local login page, is sent to parent, or both.

WT Error	Description	Where
WTE001	Not Supported Message	Both
WTE002	Generic Error	Both
WTE003	Disconnected	Both
WTE004	Unable to Connect	Both
WTE005	Invalid username or password	Local
WTE006	Competition: user connected on another machine	Both
WTE007	Unable to connect to api	Both
WTE008	Not connected to server	Both
WTE009	Restricted IP	Local
WTE010	User not specified	Local
WTE011	User locked out	Local
WTE012	Password expired	Local
WTE013	Session Token Authentication Failed	Local
WTE014	Session Token Authentication Passed	Local
WTE015	AUTHENTICATED	Local
WTE016	NEED_AUTHENTICATION	Local
WTE017	Guaranteed Dollar user not supported	Local
WTE018	Invalid Whitebranded user	Parent
WTE019	Invalid Parent domain	Parent

WTE020	Invalid subscription	Parent
WTE021	Invalid chart request	Parent
WTE022	Chart Result failed	Parent
WTE023	Invalid Timeout	Parent
WTE024	Invalid contract	Parent
WTE099	Min Version	Parent

7. Security Checks

Only valid white branded users and valid domains are allowed to use this application. Applications will need to register their test and prod domains with IB (this will be discussed during setup).

Also check with IB for the correct iframe domain to be used for PostMessage validations on the parent side. This could be different for test and production environments.

8. Stylesheet for each IBroker

Custom CSS style sheets to alter the look of the login page can be used for each IBroker. These styles sheets will be hosted by IB, separately from the application. WTWebApi will contain default configuration but will allow a different style sheet per white branding ID to be loaded after the default one. This stylesheet for the white branded user will be served from a different location so that stylesheet changes will not require a release.

The custom style sheet can overwrite the colors, background colors, font family, font size, etc. Recommended selectors to use are: body, input. text, button.submit, h1, title. IBrokers will have to test their modified style sheet on every major supported browser before sending to IB. IB will not be held responsible for a broken layout of the login page after incorporating the custom style sheet.

Style sheets to be incorporated for each white-branded user (if any) will be confirmed during setup.

9. Translation

IBrokers can be provided with the Error codes for translation (as needed). The process followed would be the same as for other IB applications and would be through CLAMS. Users can choose from a list of languages that we offer for translation and translated text, and error codes will be shown on the login page.

Translation needs for each white-branded user (if any) will be confirmed during setup.

Appendix A

The demo url is:

https://zh.wgw.clientam.com/wtapibeta/frame/index.jsp

The sample wtapi.js can be found at:

https://zh.wgw.clientam.com/wtapibeta/frame/wtapi.js

The sample client.js can be found at:

https://zh.wgw.clientam.com/wtapibeta/frame/client.js

Please copy the two sample files (wtapi.js and client.js) and customize as needed in your application.