Web Services Support

Spotware Systems Ltd

Version 83, 2023-01-04

cTRADER
TRADERS FIRST™

Table Of Contents

1. Understanding the cTrader Login Domain Model

This section contains the definitions of the key entities participating in the cTrader login domain model. We highlight the functions these entities perform, how they relate to each other, and how they are represented in cTrader applications and the cTrader backend.

1.1. Understanding the cTrader Infrastructure

Spotware has always built scalable and efficient solutions designed to help brokers with avoiding disruptions to their client onboarding flows and facilitating progress along the related funnel. We strongly believe that the fewer barriers a trader has to face when attempting to login into their preferred trading application, the better.

Our authorization model also conforms to this paradigm. A brief explanation of the cTrader server infrastructure is needed before we can fully explain how our authorization model works.

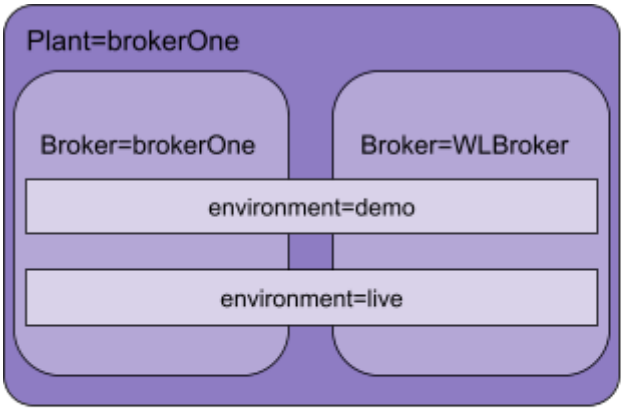
Plant. This term denotes a logical aggregation of several environments into a single 'pack'. In the vast majority of cases, a plant combines all environments managed by a single broker. This aggregation is useful to quickly and clearly identify a specific trading server (as shown below by our naming convention) and to facilitate billing.

Environment. This term defines an actual server which is itself an instance of the backend solution provided by Spotware to a broker. Our generic setup allows two environments (namely 'demo' or 'live') to exist on one plant. Brokers can request the creation of additional environments (e.g., 'live2') with any naming. A single broker may have an unlimited number of environments. However, the deployment of each new environment is billed separately and constitutes an additional charge. Please contact sales@spotware.com to get more information on adding new environments.

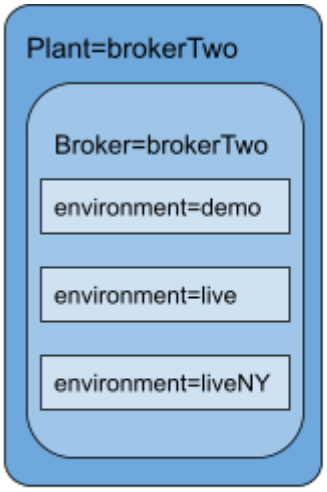
Broker (White Label). This designation applies to an entity that can be used to segregate trading accounts registered on one plant. This entity, therefore, may refer to different White Labels (WLs) and/or jurisdictions. By default, each plant contains one broker (which is denoted as 'brokerName' in this API) with the name of this broker also used as the plant name. Note that a plant may support a potentially infinite number of broker/WL entities.

Given the terms defined above, our naming convention for a trading server is <environment>.<plant>. For example, a server could be designated as **demo.superbroker**.

The relationship between plants, environments, and brokers can also be summarized graphically as follows. The picture on the left highlights a case in which a single plant is shared by two brokers with **brokerOne** being the 'default' broker. In the picture on the right, plant **brokerTwo** has several environments set up within a single broker..



Plant with two servers (environments) and two brokers



Plant with three servers (environments) and one broker

1.2. cTrader ID (cTID)

As part of our continuous improvements to the creation/authorization UX, we have introduced a solution that allows all cTrader users to have a single set of credentials for any environment in the cTrader ecosystem. This per-user set of credentials forms an entity that we have designated as cTrader ID or simply cTID.

Think of cTID as a unique identifier of a certain user-person within the cTrader ecosystem. To create a new cTID, the user must provide a valid email address.

After a user registers their email, our system finishes creating the new cTID by automatically assigning the user with a unique nickname (which is a part of the specified email). This nickname can be later changed by the user. The cTID, subsequently, allows the user to authorize in any cTrader applications (of any cTrader-affiliated broker) by entering the related set of credentials.

Note that authorization via cTID is possible either via the user's email or their custom (or automatically generated) nickname, and the correct password. To account for this, all cTrader applications denote possible logins as the user's email or the cTrader ID. The custom nickname is not mentioned as not all users may choose to set it after creating a new cTID. Nevertheless, cTID is an entity that may contain a unique email and a unique nickname, leading to the possibility of using either to login inside cTrader applications.

To reiterate, cTID allows for using a single set of credentials to log into cTrader applications and access accounts that may be registered under any number of different brokerages. For reasons of security, brokers cannot change or set cTID passwords as this would lead to brokers gaining access to sensitive data from plants that they may not own. User receives automatic email from Spotware about cTID creation and link to set password.

1.3. Account (Trader) and Linkage to cTID

As an entity, cTID is only used for user authorization. In our ecosystem, the responsibility to perform trading operations is delegated to Accounts that constitute entirely different entities compared to cTID. In this API, Accounts are referred to as Traders.

To elaborate, an Account is an entity that always belongs to a specific cTID and has the power to engage in trading activities.

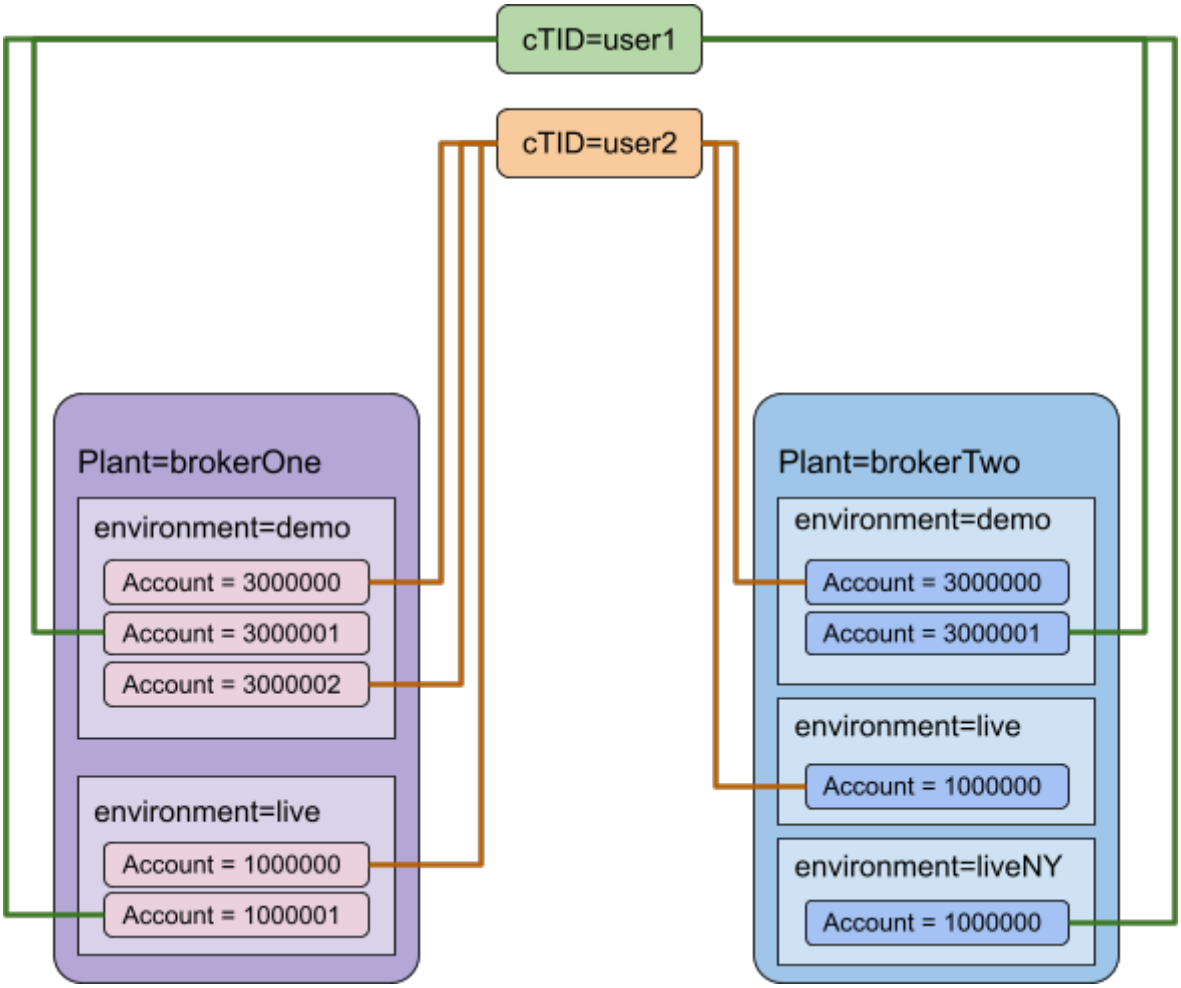
In contrast to cTIDs, Accounts constitute an entity that is always registered on a specific plant and a specific environment. As cTIDs cannot trade, they must be linked to at least one Account on any server.

Note that each cTID can have an unlimited number of linked accounts registered on different servers. Nonetheless, cTrader applications are designed so that only linked accounts from a specific broker are shown to a cTID when using a branded cTrader application. The only exception to this rule is Spotware cTrader which functions as a cross-broker application.

While an Account still has a password as an attribute, it is only used in two cases:

1. Brokers must use it to link an Account to a related cTID.
2. Traders must use when working with our FIX API solution.

The diagram below illustrates a case in which both **user1** and **user2** have registered several demo and live accounts on different plants. With the exception of the cross-broker application, **user1** will only see two accounts registered with **brokerOne** when accessing cTrader applications branded and released for **brokerOne**. In contrast, when accessing the same applications, **user2** will see three accounts registered with **brokerOne**.



1.4. Summary

As an entity, the cTID is used to authorize end users in the trading application(s) of their choice. Accounts, in turn, can perform trading operations on a per-plant per-environment basis. A cTID may have any number of linked trading accounts.

As a result, to fully register a new user, a broker must perform the following actions:

1. Create a new cTID (p. 3.1 of the current spec).
2. Create a new Account (p 2.2 of the current spec).
3. Link the new Account to the cTID (p. 3.2 of the current spec).

Note that, in contrast to other trading platforms, cTrader accounts for two different authorization flows that end users may choose to participate in.

1. Login using a cTID. The current API spec covers this model.
2. Login using credentials from a broker's client area/portal/CRM. Detailed specifications for this model can be found on our Help Center at https://help.ctrader.com/broker-oauth-inapp/introduction/.

2. Operations with accounts

We expose the following RESTful web services for operations with Accounts (see the table below).

The content type for requests and responses is text/xml or text/json.

Type definitions can be found at https://HOST:PORT/v2/webserv/schema

All URLs are relative to https://HOST:PORT/v2

The web services API is authenticated under the same Manager credentials which are used to login to the cBroker application. Usage of the API requires inclusion of an authentication token with each request by appending ?token={token} to the end of the request URL. Token doesn't have expiration time and could be generated according to section Generate Manager's token.

Manager permissions are separated by Groups as well as by brokerNames (White Labels). To have access to a specific Account or related server entities (Orders, Positions, Transactions, etc.), a Manager needs to have access to the Group to which the Account is assigned as well as the brokerName of the Account. This brokerName must not be in the disabledBrokerName array of the Manager.

Notes:

1. The Volume parameter contain the amount in 10^2 (e.g. volume=234512 is 2345.12 base asset units).
2. All fields related to money (balance, commission, etc.) contain the amount in 10^moneyDigits (e.g. balance=234512 with moneyDigits=2 is 2345.12 currency units; balance=234512 with moneyDigits=8 is 0.00234512 currency units).
3. Number of requests per hour may be limited due to abnormal and harmful usage. In case limit is activated all new PUT/POST/DELETE requests will be rejected. GET requests won't be limited.

2.1. Generate Manager’s token

HTTP Method:

POST

URL:

/webserv/managers/token

Inputs:

GetWebservTokenReqTO

- **login** is your cBroker Manager’s login
- **hashedPassword** is MD5 of your cBroker Manager’s password

Output:

GetWebservTokenResTO

Expected HTTP status code:

200

Example:

```
curl -X POST "https://HOST:PORT/v2/webserv/managers/token" -H "accept:
application/json" -H "Content-Type: application/json" -d "{ \"hashedPassword\":
\"0f94e246908667af85916300c57f74b6\", \"login\": 9017800}"
```

```
{
  "webservToken": "1dd4ef40-c5b3-61c0-0689-b1b40c97fadc"
}
```

2.2. Create Trader

HTTP Method:

POST

URL:

/webserv/traders

Inputs:

CreateTraderReqTO

Output:

TraderTO

Expected HTTP status code:

201

Example:

```
curl -X POST "https://HOST:PORT/v2/webserv/traders?token=1dd4ef40-c5b3-61c0-0689-
b1b40c97fadc" -H "accept: application/json" -H "Content-Type: application/json" -d "{
\"accessRights\": \"FULL_ACCESS\  ", \"accountType\": \"HEDGED\", \"balance\": 1000,
\"brokerName\": \"BESTBROKER\", \"contactDetails\": { \"address\": \"Moon\", \"city\":
\"Lake\", \"countryId\": 8, \"documentId\": \"0123\", \"email\":
\"president@moon.moon\", \"phone\": \"+50987654321\", \"state\": \"RE  \",
\"zipCode\": \"5500\" }, \"defaultSplitRevenue\": true, \"depositCurrency\": \"EUR\",
\"description\": \"string\", \"enabled\": true, \"frenchRisk\": true, \"groupName\":
\"Default\", \"hashedPassword\": \"d8578edf8458ce06fbc5bb76a58c5ca4\", \"lastName\":
\"President\  ", \"leverageInCents\": 10000, \"maxLeverage\": 100000, \"name\":
\"Best\", \"sendOwnStatement\": true, \"swapFree\": false,
\"totalMarginCalculationType\": \"MAX\"}"
```


```
{"login":9017800,"groupName":"Default","depositCurrency":"EUR","name":"Best","lastName
":"President","description":"string","accessRights":"FULL_ACCESS","balance":1000,"bonu
s":0,"nonWithdrawableBonus":0,"leverageInCents":10000,"contactDetails":{"email":"presi
dent@moon.moon  ","address":"Moon","city":"Lake","state":"RE","zipCode":"5500","countr
yId":8,"documentId":"0123","phone":"+50987654321"},"registrationTimestamp":15988620138
24,"lastUpdateTimestamp":1598862013825,"equity":1000,"usedMargin":0,"freeMargin":1000,
"accountType":"HEDGED","introd  ucingBroker":true,"introducingBrokerCommissionRate":0,
"pocketCommissionRate":0,"pocketMarkupRate":0,"defaultIntroducingBrokerCommissionRate"
:0,"defaultPocketCommissionRate":0,"defaultPocketMarkupRate":0,"defaultRebateRate":0,"
defaultSplitRevenue":true,"limitedRisk":false,  "sendOwnStatement":true,"swapFree":fal
se,"splitRevenue":false,"rankId":1,"ranks":{"rank":[{"id":1,"name":"Standard","volume"
:0,"parentIbPercentage":35,"brokerPercentage":30}]},"totalMarginCalculationType":"MAX"
,"maxLeverage":100000,"frenchRisk":true,"isLimitedRisk":true,"  limitedRiskMarginCalcu
lationStrategy":"ACCORDING_TO_LEVERAGE","moneyDigits":2,"sendStatementToBroker":false,
"defaultIbCommissionsType":"USD_PER_LOT","ibCommissionsType":"USD_PER_MILLION_USD"}
```

2.3. Read Trader's Details

HTTP Method:

GET

URL:

/webserv/traders/{login}

Inputs:

login

Output:

TraderTO

Expected HTTP status code:

200

Example:

```
curl -X GET "https://HOST:PORT/v2/webserv/traders/9017800?token=1dd4ef40-c5b3-61c0-
0689-b1b40c97fadc" -H "accept: application/json"
```


```
{"login":9017800,"groupName":"Default","depositCurrency":"EUR","name":"Best","lastName
":"President","description":"string","accessRights":"FULL_ACCESS","balance":1000,"bonu
s":0,"nonWithdrawableBonus":0,"leverageInCents":10000,"contactDetails":{"email":"presi
dent@moon.moon  ","address":"Moon","city":"Lake","state":"RE","zipCode":"5500","countr
yId":8,"documentId":"0123","phone":"+50987654321"},"registrationTimestamp":15988620138
24,"lastUpdateTimestamp":1598862013825,"equity":1000,"usedMargin":0,"freeMargin":1000,
"accountType":"HEDGED","introd  ucingBroker":true,"introducingBrokerCommissionRate":0,
"pocketCommissionRate":0,"pocketMarkupRate":0,"defaultIntroducingBrokerCommissionRate"
:0,"defaultPocketCommissionRate":0,"defaultPocketMarkupRate":0,"defaultRebateRate":0,"
defaultSplitRevenue":true,"limitedRisk":false,  "sendOwnStatement":true,"swapFree":fal
se,"splitRevenue":false,"rankId":1,"ranks":{"rank":[{"id":1,"name":"Standard","volume"
:0,"parentIbPercentage":35,"brokerPercentage":30}]},"totalMarginCalculationType":"MAX"
,"maxLeverage":100000,"frenchRisk":true,"isLimitedRisk":true,"  limitedRiskMarginCalcu
lationStrategy":"ACCORDING_TO_LEVERAGE","moneyDigits":2,"sendStatementToBroker":false,
"defaultIbCommissionsType":"USD_PER_LOT","ibCommissionsType":"USD_PER_MILLION_USD"}
```

2.4. Read Details of all Traders

HTTP Method:

GET

URL:

/webserv/traders/

Inputs:

- from timestamp (inclusive) formatted as `yyyy-MM-ddTHH:mm:ss.SSS` in UTC (example: 2018-01-01T12:12:12.000)
- to timestamp (exclusive) formatted as `yyyy-MM-ddTHH:mm:ss.SSS` in UTC (example: 2018-01-01T12:12:12.000)
- groupId int if specified only accounts of this group will be returned
- token Manager token

Output:

TraderListTO and field required bool hasMore

Expected HTTP status code:

200

Example:

```
curl -X GET "https://HOST:PORT/v2/webserv/traders/?from=2020-09-
03T11:55:32.773&to=2022-09-03T11:58:08.355&token=1dd4ef40-c5b3-61c0-0689-b1b40c97fadc"
-H "accept: application/json"
```


```
{"trader":[{"login":47,"groupName":"testUSD","depositCurrency":"USD","name":"brahmaput
ra","accessRights":"FULL_ACCESS","balance":10000000,"bonus":0,"nonWithdrawableBonus":0
,"leverageInCents":10000,"contactDetails":{"email":"default@ctrader.com","city":"delhi
","countryId":356},"registrationTimestamp":1632960000000,"lastUpdateTimestamp":1632960
000000,"lastConnectionTimestamp":1632960000000,"equity":10000000,"usedMargin":0,"freeM
argin":10000000,"accountType":"HEDGED","introducingBroker":false,"introducingBrokerCom
missionRate":0,"pocketCommissionRate":0,"pocketMarkupRate":0,"rebateRate":0,"defaultIn
troducingBrokerCommissionRate":0,"defaultPocketCommissionRate":0,"defaultPocketMarkupR
ate":0,"defaultRebateRate":0,"defaultSplitRevenue":false,"limitedRisk":false,"sendOwnS
tatement":false,"swapFree":false,"splitRevenue":false,"ranks":{"rank":[]},"totalMargin
CalculationType":"MAX","frenchRisk":false,"isLimitedRisk":false,"moneyDigits":2,"defau
ltIbCommissionsType":"USD_PER_MILLION_USD","ibCommissionsType":"USD_PER_MILLION_USD"},
{"login":60,...
```

2.5. Read SubAccount Details

HTTP Method:

GET

URL:

/webserv/traders/{login}/subaccounts

Inputs:

login

Output:

SubAccountsTotalTO

Expected HTTP status code:

200

Example:

```
curl -X GET "https://HOST:PORT/v2/webserv/traders/9017800/subaccounts?token=1dd4ef40-
c5b3-61c0-0689-b1b40c97fadc" -H "accept: application/json"
```


```
{"subAccountLogins":{"login":[9006731,9006762,9006765]},"totalSubAccountsBalance":4601
6,"totalSubAccountsEquity":47324}
```

2.6. Update Trader

HTTP Method:

PUT

URL:

/webserv/traders/{login}

Inputs:

UpdateTraderReqTO

Output:

none

Expected HTTP status code:

204

Important note:

On update all fields of the the trader must be sent otherwise they will be set to null or to a default value

Example:

```
curl -X PUT "https://HOST:PORT/v2/webserv/traders/9017800?token=1dd4ef40-c5b3-61c0-
0689-b1b40c97fadc" -H "accept: application/json" -H "Content-Type: application/json"
-d "{ \"accessRights\": \"FULL_ACCESS\", \"brokerName\": \"string\",
\"contactDetails\": { \"address\": \"string\", \"city\": \"string\", \"countryId\": 0,
\"documentId\": \"string\", \"email\": \"string\", \"introducingBroker1\": \"string\",
\"introducingBroker2\": \"string\", \"phone\": \"string\", \"phonePassword\":
\"string\", \"state\": \"string\", \"status\": \"string\", \"zipCode\": \"string\" },
\"defaultIntroducingBrokerCommissionRate\": 0, \"defaultPocketCommissionRate\": 0,
\"defaultPocketMarkupRate\": 0, \"defaultRebateRate\": 0, \"defaultSplitRevenue\":
true, \"description\": \"string\", \"frenchRisk\": true, \"groupName\": \"string\",
\"introducedByBroker\": 0, \"introducingBroker\": true,
\"introducingBrokerCommissionRate\": 0, \"isLimitedRisk\": true, \"lastName\":
\"string\", \"leverageInCents\": 0, \"limitedRisk\": true,
\"limitedRiskMarginCalculationStrategy\": \"ACCORDING_TO_LEVERAGE\", \"login\": 0,
\"maxLeverage\": 0, \"name\": \"string\", \"pocketCommissionRate\": 0,
\"pocketMarkupRate\": 0, \"rankId\": 0, \"ranks\": { \"rank\": [ {
\"brokerPercentage\": 0, \"id\": 0, \"name\": \"string\", \"parentIbPercentage\": 0,
\"volume\": 0 } ] }, \"sendOwnStatement\": true, \"splitRevenue\": true, \"swapFree\":
true}"
```

2.7. Delete Trader

HTTP Method:

DELETE

URL:

/webserv/traders/{login}

Inputs:

login

Output:

none

Expected HTTP status code:

204

Example:

```
curl -X DELETE "https://HOST:PORT/v2/webserv/traders/9017800?token=1dd4ef40-c5b3-61c0-
0689-b1b40c97fadc"
```

2.8. Change Trader's Balance

HTTP Method:

POST

URL:

/webserv/traders/{login}/changebalance

Inputs:

- login
- ChangeTraderBalanceReqTO

Output:

ChangeTraderBalanceResTO

Expected HTTP status code:

200

Example:

```
curl -X POST
"https://HOST:PORT/v2/webserv/traders/9017800/changebalance?token=1dd4ef40-c5b3-61c0-
0689-b1b40c97fadc" -H "accept: application/json" -H "Content-Type: application/json"
-d "{ \"amount\": 12345, \"comment\": \"string\", \"externalId\": \"string\",
\"externalNote\": \"string\", \"login\": 9017800, \"preciseAmount\": 123.45,
\"source\": \"string\", \"type\": \"DEPOSIT\"}"
```

```
{"balanceHistoryId":6340680}
```

2.9. Change Trader's Password

HTTP Method:

POST

URL:

/webserv/traders/{login}/changepassword

Inputs:

- login
- ChangeTraderPasswordReqTO

Output:

none

Expected HTTP status code:

204

Example:

```
curl -X POST
"https://HOST:PORT/v2/webserv/traders/9017800/changepassword?token=1dd4ef40-c5b3-61c0-
0689-b1b40c97fadc" -H "accept: application/json" -H "Content-Type: application/json"
-d "{ \"hashedPassword\": \"0f94e246908667af85916300c57f74b6\", \"login\": 9017800}"
```

2.10. Check Trader's Password

HTTP Method:

POST

URL:

/webserv/traders/{login}/checkpassword

Inputs:

- login
- CheckTraderPasswordReqTO

Output:

none

Expected HTTP status code:

204

Example:

```
curl -X POST
"https://HOST:PORT/v2/webserv/traders/9017800/checkpassword?token=1dd4ef40-c5b3-61c0-
0689-b1b40c97fadc" -H "accept: application/json" -H "Content-Type: application/json"
-d "{ \"hashedPassword\": \"0f94e246908667af85916300c57f74b6\", \"login\": 9017800}"
```

2.11. Change Trader's Bonus

HTTP Method:

POST

URL:

/webserv/traders/{login}/changebonus

Inputs:

- login
- ChangeTraderBonusReqTO

Output:

ChangeTraderBonusResTO

Expected HTTP status code:

200

Example:

```
curl -X POST "https://HOST:PORT/v2/webserv/traders/9017800/changebonus?token=1dd4ef40-
c5b3-61c0-0689-b1b40c97fadc" -H "accept: application/json" -H "Content-Type:
application/json" -d "{ \"amount\": 123, \"comment\": \"string\", \"externalNote\":
\"string\", \"login\": 9017800, \"preciseAmount\": 1.23, \"type\": \"DEPOSIT\"}"
```

```
{"bonusHistoryId":2825}
```

2.12. List Trader Groups

HTTP Method:

GET

URL:

/webserv/tradergroups

Inputs:

none

Output:

TraderGroupListTO

Expected HTTP status code:

200

Example:

```
curl -X GET "https://HOST:PORT/v2/webserv/tradergroups?token=1dd4ef40-c5b3-61c0-0689-
b1b40c97fadc" -H "accept: application/json"
```


```
{"traderGroup":[{"id":51350,"name":"Disabled
Group","description":""},{"id":51351,"name":"Disabled 2","description":"demo
group"},{"id":51352,"name":"EURUSD
unknown","description":""},{"id":51354,"name":"Dasha_del","description":""},{"id":5135
5,"name":"Fair Stop
Out","description":""},{"id":51100,"name":"DN717iZXZePvlaCsmsi12jZb","description":""}
,{"id":51356,"name":"30
stopout","description":""},{"id":51101,"name":"Cj0D0QgqwUpfzXqtO2IpGkS2","description"
:""},{"id":51357,"name":"234234","description":""},{"id":51102,"name":"KgWzWUiWPausD7D
ILrNxCuCP","description":""},{"id":51358,"name":"Test13","description":""},{"id":51366
,"name":"DS_stopout","description":""},{"id":51367,"name":"arta_test","description":""
},{"id":1000,"name":"Default","description":"demo
group"},{"id":51368,"name":"arta_test2","description":""},{"id":51369,"name":"OnlyAUDC
AD","description":""},{"id":51050,"name":"TEST","description":"Testing Group
Symbols"},{"id":51370,"name":"load-cat","description":""},{"id":51371,"name":"swap
test","description":""},{"id":51250,"name":"for_stopout_test","description":""},{"id":
51000,"name":"Test1","description":""},{"id":51001,"name":"Test12","description":""}]}
```

2.13. List of open positions

HTTP Method:

GET

URL:

/webserv/openPositions

Inputs:

none or login

Output:

Comma-separated values

Expected HTTP status code:

200

Example:

```
curl -X GET "https://HOST:PORT/v2/webserv/openPositions?token=1dd4ef40-c5b3-61c0-0689-
b1b40c97fadc"
```

```
login,positionId,openTimestamp,entryPrice,direction,volume,symbol,commission,swap,book
Type,stake,spreadBetting,usedMargin
1010,711,2020-09-
03T11:51:07.902,1.1,BUY,100000.00,EURUSD,0.00,0.00,BOOK_B,0.00,false,1000.00
1011,734,2020-09-
02T09:41:22.361,1.2,SELL,20000.00,EURGBP,0.00,0.00,BOOK_A,0.00,false,1000.00
```

or with filtering by the login:

```
curl -X GET "https://HOST:PORT/v2/webserv/openPositions?login=1000&token=1dd4ef40-
c5b3-61c0-0689-b1b40c97fadc"
```

```
login,positionId,openTimestamp,entryPrice,direction,volume,symbol,commission,swap,book
Type,stake,spreadBetting,usedMargin
1010,711,2020-09-
03T11:51:07.902,1.1,BUY,100000.00,EURUSD,0.00,0.00,BOOK_B,0.00,false,1000.00
```

2.14. List of closed positions

HTTP Method:

GET

URL:

/webserv/closedPositions

Inputs:

- from timestamp (inclusive) formatted as `yyyy-MM-ddTHH:mm:ss.SSS` in UTC (example: 2018-01-01T12:12:12.000)
- to timestamp (exclusive) formatted as `yyyy-MM-ddTHH:mm:ss.SSS` in UTC (example: 2018-01-01T12:12:12.000)
- login (optional)

Notes:

Period cannot be more than 2 (two) days

Output:

Comma-separated values

Expected HTTP status code:

200

Example:

```
curl -X GET "https://HOST:PORT/v2/webserv/closedPositions?from=2020-09-
03T11:55:32.773&to=2020-09-03T11:58:08.355&token=04d95575-c9af-42fb-a72e-2f0ce93f01d4"
```

```
login,positionId,openTimestamp,closeTimestamp,closePrice,direction,volume,symbol,commi
ssion,swap,pnl,depositConversionRate,usdConversionRate,bookType,stake,spreadBetting
1012,716,2020-09-03T11:58:07.580,2020-09-
03T11:58:07.626,1.20000,SELL,200000.00,EURJPY,0.00,0.00,0.00,0.8333333333333334,1.1,BO
OK_B,0.00,false
1033,1244,2020-09-03T13:08:11.607,2020-09-
03T14:22:04.889,1.25400,BUY,87000.00,EURJPY,0.00,0.00,0.00,0.82,1.2,BOOK_A,0.00,false
```

or with filtering by the login:

```
curl -X GET "https://HOST:PORT/v2/webserv/closedPositions?from=2020-09-
03T11:55:32.773&to=2020-09-03T11:58:08.355&login=2033&token=04d95575-c9af-42fb-a72e-
2f0ce93f01d4"
```

```
login,positionId,openTimestamp,closeTimestamp,closePrice,direction,volume,symbol,commi
ssion,swap,pnl,depositConversionRate,usdConversionRate,bookType,stake,spreadBetting
2033,1231,2020-09-03T11:58:07.580,2020-09-
03T11:58:07.626,1.20000,SELL,200000.00,EURJPY,0.00,0.00,0.00,0.8333333333333334,1.1,BO
OK_B,0.00,false
2033,1244,2020-09-03T13:08:11.607,2020-09-
03T14:22:04.889,1.25400,BUY,87000.00,EURJPY,0.00,0.00,0.00,0.82,1.2,BOOK_A,0.00,false
```

2.15. Create Withdrawal Request

HTTP Method:

POST

URL:

/webserv/traders/{login}/withdrawRequest

Inputs:

- login of the trader
- preciseAmount as decimal amount
- comment (optional)

Output:

none

Expected HTTP status code:

200

Example:

```
curl -X POST
"https://HOST:PORT/v2/webserv/traders/1007/withdrawRequest?token=04d95575-c9af-42fb-
a72e-2f0ce93f01d4" -H "accept: application/json" -H "Content-Type: application/json"
-d "{ \"comment\": \"test\", \"login\": 1007, \"preciseAmount\": 0.31234567}"
```

3. Operations with cTID

We expose the following RESTful web services for operations with cTID (users) (see the table below).

The content type for requests and responses is text/json.

All URLs are relative to https://HOST:PORT/cid

The web services API is authenticated under the same Manager credentials which are used to login to the cBroker application. Usage of the API requires inclusion of an authentication token with each request by appending ?token={token} to the end of the request URL. Token doesn't have expiration time and could be generated according to section Generate Manager's token.

3.1. Create cTID

HTTP Method:

POST

URL:

/cid/ctid/create

Inputs:

- "brokerName": "string"
- "email": "string"
- "preferredLanguage": "string"

Output:

- "userId": "integer"
- "nickname": "string"
- "email": "string"
- "utcCreateTimestamp": "integer"
- "status": "string"

Expected HTTP status code:

200

Example:

```
curl -X POST --header 'Content-Type: application/json' --header 'Accept:
application/json' -d '{ \
   "brokerName": "BESTBROKER", \
   "email": "president%40bestbroker.com", \
   "preferredLanguage": "EN" \
 }' 'https://HOST:PORT/cid/ctid/create?token=04d95575-c9af-42fba72e-2f0ce93f01d4'
```

```
{
  "userId": 10333,
  "nickname": "ctid10333",
  "email": "president@bestbroker.com",
  "utcCreateTimestamp": 1598938735839,
  "preferredLanguage": "en",
  "status": "CTID_NEW"
}
```

3.2. Link account to cTID

HTTP Method:

POST

URL:

/cid/ctid/link

Inputs:

- "traderLogin": "integer"
- "traderPasswordHash": "string"
- "userId": "integer"
- "brokerName": "string"
- "environmentName": "string"
- "returnAccountDetails": "boolean"

Output:

- "ctidTraderAccountId": "integer"

Expected HTTP status code:

200

Example:

```
curl -X POST --header 'Content-Type: application/json' --header 'Accept:
application/json' -d '{ \
   "environmentName": "demo", \
   "brokerName": "BESTBROKER", \
   "kycNotPassed": true, \
   "traderLogin": 9017800, \
   "traderPasswordHash": "0f94e246908667af85916300c57f74b6", \
   "userId": 10333, \
   "returnAccountDetails": true \
 }' 'https://HOST:PORT/cid/ctid/link?token=04d95575-c9af-42fba72e-2f0ce93f01d4'
```

```
{
    "ctidTraderAccountId": 9017800
}
```

3.3. Change Whitelabel of the Trading account

HTTP Method:

POST

URL:

/cid/ctid/changeCtidTraderAccount

Inputs:

- "brokerName": "string"
- "traderLogin": "integer"
- "environmentName" : "string"

Output:

none

Expected HTTP status code:

200

Example:

```
curl -X POST --header 'Content-Type: application/json' --header 'Accept:
application/json' -d '{ \
   "environmentName": "demo", \
   "brokerName": "BESTBROKER", \
   "traderLogin": 9017800, \
 }' 'https://HOST:PORT/cid/ctid/changeCtidTraderAccount?token=04d95575-c9af-42fba72e-
2f0ce93f01d4'
```

4. Error codes

Please find below the list of possible error codes you can encounter in a response body:

Error Code	Error Description
UNKNOWN_ERROR	In case of server error.
INVALID_REQUEST	In case of client invalid request (request validation failure).
NOT_ENOUGH_MONEY	In case the trader doesn't have enough money or has some open positions with unrealized profit/loss.
CHANGE_BALANCE_BAD_AMOUNT	In case the change balance amount is not valid. (e.g. amount < 0)
TRADER_NOT_FOUND	In case trader is not found.
TRADER_GROUP_NOT_FOUND	In case trader group not found on trader create/update operation.
WRONG_PASSWORD	In case trader's password is wrong on "Check Trader's Password" operation.
NOT_ENOUGH_RIGHTS	In case the client doesn't have enough rights to perform the operation.
REQUEST_FREQUENCY_EXCEEDED	In case client sends more requests per hour than allowed. GET requests are not limited.

If you have any questions please email us at support@spotware.com