Finalto Client and Login Administration Web Service

Contents

Finalto Financial Services Limited
11th Floor · Broadgate Tower · 20 Primrose Street
EC3A 2EW · London · United Kingdom

Phone: +44 (0) 203 455 8751
Email: info@finalto.com
Web: www.finalto.com

Finalto Client and login administration web service v1.2

General information

The purpose of this web service is to allow a tight integration with IT partner systems, and allow them to implement their own creation of trading system clients and do maintenance.

Security is always important for public available services like this, so the service uses HTTPS basic authentication (Username/Password and authorization), based on the Backoffice system security model. This controls access to the functionality by the roles defined for the various counterpart levels the login has been defined for.

Notice that some of the functions are based on the Client Template implementation, and in general, clients can only be created through a predefined setup of the various parameters. If needed, multiple templates can be defined to support different setups. It is also possible to modify some parameters for existing clients, like the login and password, and on account and client level.

Client import of interfaces and WSDL

MS Visual Studio can out of the box implement a proxy class to access the web service, but since the web service also check security on this level, a valid BackOffice login of custom type and password must be created in Backoffice and can be used during the WSDL retrieval, but is not needed. Notice that retrieving a valid WSDL does not automatic mean that the BackOffice user has the correct roles defined to call the functions.

In general, we have implemented throwing meaningful exceptions on all levels to allow troubleshooting the calling applications, so a proper catch must be implemented in the calling program.

The demo WSDL can be obtained on: https://apiwsdemo.finalto.com:8098/DemoClientAdmin

The live URL will be given out to clients, once the development has finished and has been tested on the demo system.

Finalto Financial Services Limited
11th Floor · Broadgate Tower · 20 Primrose Street
EC3A 2EW · London · United Kingdom

Phone: +44 (0) 203 455 8751
Email: info@finalto.com
Web: www.finalto.com

Connecting to the service

Your first goal should be to successfully call the ***ValidateService*** method, and get a true bool back. This method is purely for test (or watchdog requests) and it will verify that the BackOffice login you are using has the needed role to execute the client administration functions.

The external service address for demo is: https://apiwsdemo.finalto.com:8098/DemoClientAdmin

And you need the correct setting for running HTTPS and basic authentication scheme. In a Microsoft WCF world, this is defined as:

```
<security mode="Transport">
            <transport clientCredentialType="Basic" proxyCredentialType="None" realm="" />
             <message clientCredentialType="UserName" algorithmSuite="Default" />
</security>
```


Finalto Financial Services Limited
11th Floor · Broadgate Tower · 20 Primrose Street
EC3A 2EW · London · United Kingdom

Phone: +44 (0) 203 455 8751
Email: info@finalto.com
Web: www.finalto.com

The methods

All access is through the interface `ITSClientAdmin,` implementing the following functions:

```
bool ValidateService(out string Message);
```
This is simply a test method to validate transport layer security, returns the string “ok” if ok.

```
TPClientParams CreateOneClickUser(int TemplateID, int BrokerID);

TPClientParams CreateUserByName(int TemplateID, int BrokerID, string ClientName, string
ClientMail);

TPClientParams CreateClientFromTemplate(int TemplateID, int BrokerID, string
ClientDisplayName, string LoginName, string LoginPassword, string ClientMail);

TPClientParams CreateClientByDisplayNamePhoneAndMail(int templateId, int brokerId, string
clientName, string clientMail, string clientPhone);

TPClientParams CreateClientByLoginPhoneAndMail(int TemplateID, int BrokerID, string
ClientDisplayName, string LoginName, string LoginPassword, string ClientMail, string
ClientPhone);

TPClientParams CreateClientFromTemplate2(   int TemplateID, int BrokerID, string
LoginDisplayName, string LoginName, string LoginPassword, string AccountName, string
ClientName, string ClientMail);

string FormatParams(TPClientParams CliPars);
```

These are all variations of the functions to create a new login with counterpart and account, based on an existing template. Different parameters can be supported to overwrite the template generated suggestion, but also leaving a parameter empty will force the API do use a template replacement, except for the mail parameter where the template cannot really do anything. The BrokerID parameter is not really used as such, but will ensure that you actually have the rights and template access needed to do this function.

Each function returns a full parameter set for the user being created (or an exception if not), and these parameters can be saved locally in the partners environment. The support function, FormatParams, take out the main parameters and format it into a single string with line breaks, useful if you have made a GUI application, or want to save the reply in a text file.

Finalto Financial Services Limited
11th Floor · Broadgate Tower · 20 Primrose Street
EC3A 2EW · London · United Kingdom

Phone: +44 (0) 203 455 8751
Email: info@finalto.com
Web: www.finalto.com

```
bool IsLoginValidAndUnUsed(string LoginName, out string Message);
```

This function checks a username to be unused and legal to use. Primary targeted for websites where a user can select he's own login name.

Below functions, allow change of username or password. You will always need to first lookup the ID of the user, then call the Change functions. Returns true for done or exceptions.

```
int GetLoginIDFromName( string LoginName );
bool ChangeLoginName( int LoginID, string NewLoginName );
bool ChangeLoginPassword(int LoginID, string NewLoginPassword);
int GetClientIDFromLoginName(string LoginName);
```

The following accounting functions allow cash move between accounts controlled by the broker, and works the same way as if the request was initiated from the backoffice system, with audit trail etc.

```
string DepositFundsTransaction(int AccountID,
                                    int ContraAccountID,
                                    decimal Amount,
                                    string Currency,
                                    string Comment);

string WithdrawFundsTransaction(int AccountID,
                                    int ContraAccountID,
                                    decimal Amount,
                                    string Currency,
                                    string Comment);

string WithdrawAllFundsTransaction(int AccountID,
                                    int ContraAccountID,
                                    string Comment);
```


Finalto Financial Services Limited
11th Floor · Broadgate Tower · 20 Primrose Street
EC3A 2EW · London · United Kingdom

Phone: +44 (0) 203 455 8751
Email: info@finalto.com
Web: www.finalto.com

This function can be used to get the official middle rate used in our backoffice for various conversions.

```
decimal GetEODMidRate(DateTime TradeDate, string InstrumentSymbol);
```

```
bool ChangeClientProperty(int ClientID, ClientProperty field, string NewValue);

bool ChangeLoginProperty(int LoginID, LoginProperty field, string NewValue);

bool ChangeAccountProperty(int AccountID, AccountProperty field, string NewValue);
```

These update functions can alter selected parameters on client, account and login level. The field list is extensible, so more fields can be added upon request. The definitions are currently:

```
    public enum ClientProperty
    {
        ClientName = 1
    }
    public enum LoginProperty
    {
        LoginName = 1,
        LoginPassword = 2,
        DisplayName = 3
    }
    public enum AccountProperty
    {
        AccountName = 1
    }
```

```
decimal GetEODMidRate(DateTime TradeDate, string InstrumentSymbol);
```

Calling this method will return the EoD (End Of Day) middle price for the instrument and can be used to calculate closed P/L when the End of Day process executed. A new value will be added each tradeday shortly after 17 pm NY Time. Requesting EOD rate for none trade dates or today, will fail.
This function can be used to get the official middle rate used in our backoffice for various conversions.

```
decimal GetLastMidRate(string InstrumentSymbol);
```

Calling this method, returns the latest updated market value from a mix of all liquidity providers. This method should not be used for real-time calculation, since it will only be updated every 10-20 seconds and not with each new tick.

For both the GetEODMidRate and for the GetLastMidRate methods, a currency cross synthesis engine was added, so it is possible to get rates for almost any cross combination like. For example, ZARDKK or even the reverse DKKZAR if this is required. It can be useful if client needs to convert P/L to base currency or exposure etc. Both functions are case insensitive and support XXX/YYY syntax, i.e. EURUSD and EUR/USD.

Note: To use the below two methods, '*AddAdditionalAccountByTpParams*' and '*CreateAccount*' requires refresh of the currently used WSDL.

Finalto Financial Services Limited
11th Floor · Broadgate Tower · 20 Primrose Street
EC3A 2EW · London · United Kingdom

Phone: +44 (0) 203 455 8751
Email: info@finalto.com
Web: www.finalto.com

```
[OperationContract]
int AddAdditionalAccountByTpParams(TPClientParams CliPars, string
AccountNameOrMask, int PriceGroupID);
```

Description

This purpose of this method is to add more accounts to a new counterpart after the counterpart has been created using the '`CreateClientFromTemplate`' call.

Arguments

CliPars - the returned parameter block from '`CreateClientFromTemplate`', and the account name and the pricegroup must be specified different from the values set in the original template.

AccountNameOrMask - parameter supports the inline template tokens as {ID} and {DN} to automate also the name generation of additional accounts.

Even not recommended, it is also possible to alter any parameter in the CliPars block and this way create accounts with any wanted setup.

**In demo mode, this new account will be initial funded as specified in the template.

Returned value

The ID of the new account created.

Finalto Financial Services Limited
11th Floor · Broadgate Tower · 20 Primrose Street
EC3A 2EW · London · United Kingdom

Phone: +44 (0) 203 455 8751
Email: info@finalto.com
Web: www.finalto.com

```
[OperationContract]

int CreateAccount(int BrokerID,
            AccountType AccountTypeID,
            string AccountName,
            string ExternalAccountID,
            bool IsActive,
            bool EnabledForTrading,
            string AccountCurrency,
            int PriceGroupID,
            int FXrollProfileID,
            int CFDrollProfileID,
            int CommissionGroupID,
            int ConversionProfileID,
            int MarginRequirementProfileID,
            int MarginActionProfileID,
            string CrmReference);
```

Description

This method is the basic account creation function, whereby the call has to specify each parameter. It is flexible, however it requires good knowledge of the valid values for each parameter.

Returned value

ID of the new account created.

```
[OperationContract]
bool ResetLogin2FAOTP(int LoginID)
```

Description

This methods resets the 2-Factor-Authentication One-Time-Password for the login id.

Returned value

Boolean indicating if method call was successful.

Finalto Financial Services Limited
11th Floor · Broadgate Tower · 20 Primrose Street
EC3A 2EW · London · United Kingdom

Phone: +44 (0) 203 455 8751
Email: info@finalto.com
Web: www.finalto.com