TRADING
TECHNOLOGIES

USING THE X_TRADER API IN MULTI-CONNECTION CONFIGURATIONS

VERSION 7.17.X

DOCUMENT VERSION 7.17.0.DV1 3/5/14

LEGAL

Using the X_TRADER API in Multi-Connection Configurations

Overview

In the 7.17 version of X_TRADER API, support has been added for multi-connection configurations. Unlike a single connection configuration where a user is set up with a single set of credentials for each TT Gateway, a multi-connection configuration is one in which a user is set up with two or more sets of credentials for at least one TT Gateway.

When a user is set up with a multi-connection configuration, the X_TRADER API will make a separate connection to a TT Gateway for each set of credentials with which a user has been set up. Multi-connection configurations are the basis of the TT Multibroker environment, however, users can be set up in this manner in any TT environment.

Even though the X_TRADER API was enhanced to support this capability, backwards compatibility has been preserved. As such, existing X_TRADER API applications will continue to operate properly for users who are set up with a single-connection configuration. However, changes may be necessary to X_TRADER API applications if support is required for users who are set up with a multi-connection configuration.

Connection Name

In single connection configurations, a user is set up with only one set of credentials for each TT Gateway. Consequently, the TT Gateway name is used as the key to uniquely identify the connection.

In multi-connection configurations, a user is set up with more than one set of credentials for each TT Gateway. Therefore, TT Gateway name alone is not sufficient to uniquely identify a connection. Hence, a new concept called "Connection Name" has been introduced in X_TRADER API 7.17 that is used to uniquely identify a TT Gateway connection. Connection Name has the following format:

> <TT Gateway><separator><Broker Name><separator><TT Member ID><separator><TT Group ID><separator><TT Trader ID>

where separator is a middle dot "•" (Keystroke **`Alt+0183`**). An example of a Connection Name value for a user in the TT Multibroker environment might look like:

> CME-Q•JPM•TTORD•777•001

For non-multiconnection environments, the Broker Name would be **0**. For example:

> CME-Q•0•TTORD•777•001

The X_TRADER API only requires that the value for Connection Name resolves to one connection. This means that none of the parameters are required as long as the X_TRADER API has sufficient information to identify a unique connection. The following chart demonstrates how the X_TRADER API would internally resolve some example Connection Name values. Note that "Error" means that more than one result would match which means that a unique connection cannot be identified.

	Connection Name			
User Connection Names	**CME-Q**	**CME-Q•0**	**CME-Q•JPM**	**CME-Q•JPM•TTORD•777•001**
CME-Q•JPM•TTORD•777•001	CME-Q•JPM•TTORD•777•001	CME-Q•JPM•TTORD•777•001	CME-Q•JPM•TTORD•777•001	CME-Q•JPM•TTORD•777•001
CME-Q•JPM•TTORD•777•001 CME-Q•JPM•TTADM•888•001	Error	Error	Error	CME-Q•JPM•TTORD•777•001
CME-Q•JPM•TTSIM•777•001 CME-Q•JPM•TTORD•XXX•002	Error	Error	Error	CME-Q•JPM•TTORD•XXX•002
CME-Q•JPM•TTORD•777•001 CME-Q•JPM•TTORD•888•001	Error	Error	Error	CME-Q•JPM•TTORD•777•001
CME-Q•JPM•TTORD•777•001 CME-Q•GS•TTORD•888•001 CME-Q•GS•TTORD•999•002	Error	Error	CME-Q•JPM•TTORD•777•001	CME-Q•JPMTTORD•777•001
CME-Q•JPM•TTORD•777•001 Eurex•GS•TTORD•888•001 Eurex•GS•TTORD•999•002	CME-Q•JPM•TTORD•777•001	CME-Q•JPM•TTORD•777•001	CME-Q•JPM•TTORD•777•001	CME-Q•JPMTTORD•777•001
CME-Q•JPM•TTADM•777•001 CME-Q•GS•TTORD•888•001 CME-Q•GS•TTORD•999•002	Error	Error	CME-Q•JPM•TTADM•777•001	CME-Q•JPMTTADM•777•001
CME-Q•JPM•TTORDTP•XXX•001 CME-Q•GS•TTORDXY•888•001 CME-Q•GS•TTORDYZ•888•002	Error	Error	CME-Q•JPM•TTORDTP•XXX•001	Error
CME-Q•JPM•TTADM•XXX•001 CME-Q•GS•TTORD•888•001 CME-Q•GS•TTORD•999•002	Error	Error	CME-Q•JPM•TTADM•XXX•001	CME-Q•JPM•TTADM•XXX•001

When the X_TRADER API connects to a TT Gateway, it constructs a Connection Name string based on the connection details and adds it to the list exposed via the `TTGate.OrderServerConnections` property. You can use the values in this list to identify connections instead of constructing your own strings.

Data Changes

Backwards compatibility of X_TRADER API has been preserved, however, the meaning of some of the data has changed for multi-connection configurations. Specifically, the "exchange" parameter provided in the following event handlers will actually return the Connection Name instead of the TT Gateway name for multi-connection configurations.

- `TTGate::OnLoginFailed`
- `TTGate::OnExchangeStatusUpdate`
- `TTGate::OnSessionRollMessage`

In addition, the following functions have been modified to take the Connection Name as opposed to the TT Gateway name for multi-connection configurations.

- `TTGate::get_CustomerProfile`
- `TTGate::GetExchangeStatusFromKey`
- `COrderData::SetConnectionName`

Furthermore, the following attributes of the `TTGate::Get` property require the Connection Name as opposed to the TT Gateway name for multi-connection configurations.

- `Available.exchg.server`
- `ClrMember.exchg`
- `DefaultAcct.exchg`
- `Exists.exchg.server`
- `GatewayIP.exchg`
- `Group.exchg`
- `HandoffTrader.exchg`
- `HasLoginInfo.exchg`
- `Member.exchg`
- `Open.exchg`
- `ServerUp.exchg.server`
- `Trader.exchg`

Order Routing: ConnectionName Property

When a user is configured with only one set of credentials for a given TT Gateway, an order is submitted using the X_TRADER API in a manner similar to the following example:

```
int SubmitOrder(TTOrderSet orderSet, TTInstrObj pInstr, string customer,
  string buySell, int qty, string price)
{
  TTOrderProfile profile = new TTOrderProfile();

  profile.Instrument = pInstr;
  profile.Customer = customer;

  profile.Set("BuySell", buySell);
  profile.Set("Qty", qty);
  profile.Set("OrderType", "L");
  profile.Set("Limit$", price);

  return orderSet.get_SendOrder(profile);
}
```

Since an instrument is always associated with a single TT Gateway and there is only one connection for a TT Gateway in single connection configurations, the X_TRADER API derives the connection to be used from the instrument provided in the **`TTOrderProfile`** instance.

When a user is configured with more than one set of credentials for a given TT Gateway, you must explicitly set the Connection Name to be used to route the order. You do this by specifying the `ConnectionName` property of the **`TTOrderProfile`** instance as follows:

```
int SubmitOrder(TTOrderSet orderSet, TTInstrObj pInstr, string customer,
  string buySell, int qty, string price, string cn)
{
  TTOrderProfile profile = new TTOrderProfile();

  profile.Instrument = pInstr;
  profile.Customer = customer;

  profile.Set("ConnectionName", cn);
  profile.Set("BuySell", buySell);
  profile.Set("Qty", qty);
  profile.Set("OrderType", "L");
  profile.Set("Limit$", price);

  return orderSet.get_SendOrder(profile);
}
```

To reiterate, the Connection Name need only consist of enough parameters to identify a unique connection.

Order Routing: Customer Defaults

When running in a non-TT Multibroker environment (single or multi-connection configuration), only the `Customer` property needs to be specified to uniquely identify a specific Customer Defaults entry. When running in the TT Multibroker environment, you must also specify the `Broker` property to uniquely identify a specific Customer Defaults entry.

In the TT Multibroker environment, specifying only the `Customer` property is no longer enough to indicate which Customer Default entry is to be used. If an X_TRADER API application is running in the TT Multibroker environment, it must specify the `Broker` property as well as `Customer` property to uniquely identify the Customer Default to be used. For example:

```
int SubmitOrder(TTOrderSet orderSet, TTInstrObj pInstr, string customer,
  string buySell, int qty, string price, string cn, string broker)
{
  TTOrderProfile profile = new TTOrderProfile();

  profile.Instrument = pInstr;

  profile.Customer = customer;
  profile.Set("Broker", broker);

  profile.Set("ConnectionName", cn);
  profile.Set("BuySell", buySell);
  profile.Set("Qty", qty);
  profile.Set("OrderType", "L");
  profile.Set("Limit$", price);

  return orderSet.get_SendOrder(profile);
}
```

Failure to set the `Broker` property will result in the X_TRADER API picking the first Customer Default entry that matches the value provided in the `Customer` property.

Note: Due to the changes made to Customer Defaults, the X_TRADER API is now only compatible with X_TRADER 7.9.2 or higher.

New Properties in the TTFillObj and TTOrderObj Classes

The following properties have also been added the **`TTFillObj`** and **`TTOrderObj`** classes:

- `Broker` – Broker name
- `Broker#` – Broker ID
- `Company` – Company Name
- `Company#` - Company ID
- `ConnectionName` – Connection with which this order/fill is associated