SGCWEBSOCKETS · TECHNICAL DOCUMENT

FXCM API

FXCM ForexConnect REST and streaming-quote client for Delphi — instruments, orders and account management.

eSeGeCe

WWW.ESEGECE.COM

Overview

FXCM, also known as Forex Capital Markets, is a retail broker for trading on the foreign exchange market. FXCM allows people to speculate on the foreign exchange market and provides trading in contract for difference (CFDs) on major indices and commodities such as gold and crude oil. It is based in London.

At a glance

COMPONENT CLASS `TsgcWSAPI_FXCM`	STANDARDS / SPEC **FXCM Forex Connect API — GitHub**
TRANSPORTS **TCP, TLS**	PLATFORMS **Windows, macOS, Linux, iOS, Android**
FRAMEWORKS **VCL, FireMonkey, Lazarus / FPC, .NET**	EDITION **Standard / Professional / Enterprise**

Features

- Native Delphi implementation with full ANSI/Unicode support.

Technical specification

Standards & specs	FXCM Forex Connect API — GitHub
Component class	`TsgcWSAPI_FXCM` (unit `sgcWebSocket_API_FXCM`)
Frameworks	VCL, FireMonkey, Lazarus / FPC, .NET
Platforms	Windows, macOS, Linux, iOS, Android

Main properties

The principal published / public properties used to configure and drive the component. Consult the online help for the full list.

`Client`	Published or public property used to configure or query the component.
`IO_HeartBeatTimeout`	Published or public property used to configure or query the component.
`OnConnect`	Published or public property used to configure or query the component.
`OnDisconnect`	Published or public property used to configure or query the component.
`OnMessage`	Published or public property used to configure or query the component.
`OnBinary`	Published or public property used to configure or query the component.
`OnFragmented`	Published or public property used to configure or query the component.
`OnError`	Published or public property used to configure or query the component.
`OnException`	Published or public property used to configure or query the component.
`SocketIO`	Published or public property used to configure or query the component.

Main methods

The principal public methods exposed by the component.

`SubscribeMarketData()`	Public function exposed by the component.
`UnSubscribeMarketData()`	Public function exposed by the component.

`SubscribeTradingTables()`	Public function exposed by the component.
`UnSubscribeTradingTables()`	Public function exposed by the component.
`Ping()`	Public procedure exposed by the component.
`UpdateSubscriptions()`	Public function exposed by the component.
`TradingOrder()`	Public function exposed by the component.
`SnapshotTradingTables()`	Public function exposed by the component.

Quick Start

Drop the component on a form, configure the properties below and activate it. The snippet that follows shows the typical **Binance | Connect WebSocket API** configuration sourced from the online help.

> **About this scenario.** In order to connect to Binance WebSocket API, just create a new Binance API client and attach to TsgcWebSocketClient.

Delphi (VCL / FireMonkey)

```
oClient := TsgcWebSocketClient.Create(nil);
oBinance := TsgcWSAPI_Binance.Create(nil);
oBinance.Client := oClient;
oClient.Active := True;
```

C++ Builder

```
TsgcWebSocketClient *oClient = new TsgcWebSocketClient(NULL);
TsgcWSAPI_Binance *oBinance = new TsgcWSAPI_Binance(NULL);
oBinance->Client = oClient;
oClient->Active = true;
```

.NET (C#)

```
TsgcWebSocketClient oClient = new TsgcWebSocketClient();
TsgcWSAPI_Binance oBinance = new TsgcWSAPI_Binance();
oBinance.Client = oClient;
oClient.Active = true;
```

Common scenarios

The following scenarios are lifted verbatim from the online help. Each shows the configuration and method calls needed to drive the component through a specific real-world flow.

1 · Bitmex | Connect WebSocket API

In order to connect to Bitmex WebSocket API, just create a new Bitmex API client and attach to TsgcWebSocketClient.

Delphi (VCL / FireMonkey)

```
oClient := TsgcWebSocketClient.Create(nil);
oBitmex := TsgcWSAPI_Bitmex.Create(nil);
oBitmex .Client := oClient;
oClient.Active := True;
```

C++ Builder

```
TsgcWebSocketClient oClient = new TsgcWebSocketClient();
TsgcWSAPI_Bitmex oBitmex = new TsgcWSAPI_Bitmex();
oBitmex→Client = oClient;
oClient→Active = true;
```

.NET (C#)

```
TsgcWebSocketClient oClient = new TsgcWebSocketClient();
TsgcWSAPI_Bitmex oBitmex = new TsgcWSAPI_Bitmex();
oBitmex.Client = oClient;
oClient.Active = true;
```

2 · Coinbase | Connect WebSocket API

In order to connect to Coinbase WebSocket API, just create a new Coinbase API client and attach to TsgcWebSocketClient. See below an example:

Delphi (VCL / FireMonkey)

```
oClient := TsgcWebSocketClient.Create(nil);
oCoinbase := TsgcWSAPI_Coinbase.Create(nil);
oCoinbase.Client := oClient;
oClient.Active := True;
```

C++ Builder

```
TsgcWebSocketClient oClient = new TsgcWebSocketClient();
TsgcWSAPI_Coinbase oCoinbase = new TsgcWSAPI_Coinbase();
oCoinbase->Client = oClient;
oClient->Active = true;
```

.NET (C#)

```
TsgcWebSocketClient oClient = new TsgcWebSocketClient();
TsgcWSAPI_Coinbase oCoinbase = new TsgcWSAPI_Coinbase();
oCoinbase.Client = oClient;
oClient.Active = true;
```

3 · Kucoin | Connect WebSocket API

In order to connect to Kucoin WebSocket API, just create a new Kucoin API client and attach to TsgcWebSocketClient.

Delphi (VCL / FireMonkey)

```
oClient := TsgcWebSocketClient.Create(nil);
oKucoin := TsgcWSAPI_Kucoin.Create(nil);
oKucoin.Client := oClient;
oClient.Active := True;
```

C++ Builder

```
TsgcWebSocketClient *oClient = new TsgcWebSocketClient();
TsgcWSAPI_Kucoin *oKucoin = new TsgcWSAPI_Kucoin();
oKucoin->Client = oClient;
oClient->Active = true;
```

.NET (C#)

```
TsgcWebSocketClient oClient = new TsgcWebSocketClient();
TsgcWSAPI_Kucoin oKucoin = new TsgcWSAPI_Kucoin();
oKucoin.Client = oClient;
oClient.Active = true;
```

4 · Kucoin | Futures Connect WebSocket API

In order to connect to Kucoin WebSocket API, just create a new Kucoin API client and attach to TsgcWebSocketClient.

```
Delphi (VCL / FireMonkey)
```

```
oClient := TsgcWebSocketClient.Create(nil);
oKucoin := TsgcWSAPI_Kucoin_Futures.Create(nil);
oKucoin.Client := oClient;
oClient.Active := True;
```

```
C++ Builder
```

```
TsgcWebSocketClient *oClient = new TsgcWebSocketClient();
</code><code class="delphi">TsgcWSAPI_Kucoin_Futures </code><code class="cpp">*oKucoin = new </c
oKucoin->Client = oClient;
oClient->Active = true;
```

```
.NET (C#)
```

```
TsgcWebSocketClient oClient = new TsgcWebSocketClient();
</code><code class="delphi">TsgcWSAPI_Kucoin_Futures </code><code class="csharp">oKucoin = new <
oKucoin.Client = oClient;
oClient.Active = true;
```

5 · Binance | Subscribe WebSocket Channel

Binance offers a variety of channels where you can subscribe to get real-time updates of market data, orders... Find below a sample of how to subscribe to a Ticker:

```
Delphi (VCL / FireMonkey)
```

```
oClient := TsgcWebSocketClient.Create(nil);
oBinance := TsgcWSAPI_Binance.Create(nil);
oBinance.Client := oClient;
oBinance.SubscribeTicker('bnbbtc');

procedure OnMessage(Connection: TsgcWSConnection; const aText: string);
begin
// here you will receive the ticker updates
end;
```

C++ Builder

```
TsgcWebSocketClient *oClient = new TsgcWebSocketClient(NULL);
TsgcWSAPI_Binance *oBinance = new TsgcWSAPI_Binance(NULL);
oBinance->Client = oClient;
oBinance->SubscribeTicker("bnbbtc");

void OnMessage(TsgcWSConnection *Connection, const string aText)
{
// here you will receive the ticker updates
}
```

.NET (C#)

```
TsgcWebSocketClient oClient = new TsgcWebSocketClient();
TsgcWSAPI_Binance oBinance = new TsgcWSAPI_Binance();
oBinance.Client = oClient;
oBinance.SubscribeTicker("bnbbtc");

void OnMessage(TsgcWSConnection Connection, const string aText)
{
// here you will receive the ticker updates
}
```

6 · Bitmex | Subscribe WebSocket Channel

Bitmex offers a variety of channels where you can subscribe to get real-time updates of market data, orders... Find below a sample of how subscribe to a Trade Channel:

Delphi (VCL / FireMonkey)

```
oClient := TsgcWebSocketClient.Create(nil);
oBitmex := TsgcWSAPI_Bitmex.Create(nil);
oBitmex.Client := oClient;
oBitmex.Subscribe(btmTrade, 'XBTUSD');
procedure OnBitmexMessage(Sender: TObject; const aTopic: TwsBitmexTopics; const aMessage: string
begin
// here you will receive the trade updates
end;
```

C++ Builder

```
TsgcWebSocketClient oClient = new TsgcWebSocketClient();
TsgcWSAPI_Bitmex oBitmex = new TsgcWSAPI_Bitmex();
oBitmex->Client = oClient;
oBitmex->Subscribe(btmTrade, "XBTUSD");
void OnBitmexMessage(Sender: TObject; const aTopic: TwsBitmexTopics; const aMessage: string)
{
// here you will receive the trade updates
}
```

.NET (C#)

```
TsgcWebSocketClient oClient = new TsgcWebSocketClient();
TsgcWSAPI_Bitmex oBitmex = new TsgcWSAPI_Bitmex();
oBitmex.Client = oClient;
oBitmex.Subscribe(btmTrade, "xbtusd");
void OnBitmexMessage(Sender: TObject; const aTopic: TwsBitmexTopics; const aMessage: string)
{
// here you will receive the tradeupdates
}
```

Sources used to build this document

Every external claim links back to a primary source. The online-help references decode the canonical deep-link the company maintains for this component.

Primary standard / spec — FXCM Forex Connect API — GitHub	github.com/fxcm/ForexConnectAPI
Online help — component page	www.esegece.com/help/sgcWebSockets/Components/APIs/API/API_FXCM.htm
Delphi demo project (in the sgcWebSockets package)	`Demos\05.Crypto\07.FXCM`
.NET demo project (in the sgcWebSockets package)	`.net\demos\05.Crypto\07.FXCM`
Component page	www.esegece.com/products/websockets/apis/fxcm/
Product page	www.esegece.com/products/websockets/

> **Document scope.** This document covers the publicly-documented surface of the FXCM API component shipped with sgcWebSockets. For full property, method and event reference consult the online help linked above.