

# MEXC API



MEXC WebSocket and REST client — spot and futures market data plus authenticated trading.

## Overview

---

MEXC is a global cryptocurrency exchange that exposes streaming market and account data over secure WebSocket connections.

## At a glance

---

### COMPONENT CLASS

TsgcWSAPI\_MEXC

### STANDARDS / SPEC

MEXC Spot V3 API

### 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	<a href="#">MEXC Spot V3 API</a> · <a href="#">MEXC Contract V1 API</a>
Component class	<code>TsgcWSAPI_MEXC</code> (unit <code>sgcWebSocket_API_MEXC</code> )
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.

<code>Client</code>	Published or public property used to configure or query the component.
<code>MEXCUserDataStreams</code>	Published or public property used to configure or query the component.
<code>OnConnect</code>	Published or public property used to configure or query the component.
<code>OnMEXCResponse</code>	Published or public property used to configure or query the component.
<code>OnMEXCMarketStream</code>	Published or public property used to configure or query the component.
<code>OnMEXCUserDataStream</code>	Published or public property used to configure or query the component.
<code>OnMEXCException</code>	Published or public property used to configure or query the component.
<code>OnDisconnect</code>	Published or public property used to configure or query the component.
<code>MEXCAPI</code>	Published or public property used to configure or query the component.
<code>RawMessages</code>	Published or public property used to configure or query the component.

---

## Main methods

The principal public methods exposed by the component.

<code>SubscribeTrade()</code>	Public procedure exposed by the component.
<code>UnSubscribeTrade()</code>	Public procedure exposed by the component.

---

---

<code>SubscribeBookTickerBatch()</code>	Public procedure exposed by the component.
<code>UnSubscribeBookTickerBatch()</code>	Public procedure exposed by the component.
<code>SubscribeAccountUpdate()</code>	Public procedure exposed by the component.
<code>UnSubscribeAccountUpdate()</code>	Public procedure exposed by the component.
<code>SubscribeAccountOrders()</code>	Public procedure exposed by the component.
<code>UnSubscribeAccountOrders()</code>	Public procedure exposed by the component.
<code>SubscribeKline()</code>	Public procedure exposed by the component.
<code>UnSubscribeKline()</code>	Public procedure 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 — MEXC Spot V3 API [www.mexc.com/api-docs/spot-v3/introduction](http://www.mexc.com/api-docs/spot-v3/introduction)

---

Primary standard / spec — MEXC Contract V1 API [www.mexc.com/api-docs/contract-v1/introduction](http://www.mexc.com/api-docs/contract-v1/introduction)

---

Online help — component page [www.egegece.com/help/sgcWebSockets/Components/APIs/API/API\\_MEXC.htm](http://www.egegece.com/help/sgcWebSockets/Components/APIs/API/API_MEXC.htm)

---

Delphi demo project (in the sgcWebSockets package) `Demos\05.Crypto\15.MEXC`

---

.NET demo project (in the sgcWebSockets package) `.net\demos\05.Crypto\15.MEXC`

---

Component page [www.egegece.com/products/websockets/apis/mexc/](http://www.egegece.com/products/websockets/apis/mexc/)

---

Product page [www.egegece.com/products/websockets/](http://www.egegece.com/products/websockets/)

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