SGCWEBSOCKETS · TECHNICAL DOCUMENT

Cryptohopper API

Cryptohopper trading-bot REST client — accounts, hoppers, orders and signals for Delphi.

eSeGeCe

WWW.ESEGECE.COM

Overview

Cryptohopper trading-bot REST client — accounts, hoppers, orders and signals for Delphi. The component is part of the sgcWebSockets library.

At a glance

COMPONENT CLASS
`TsgcHTTP_Cryptohopper`

STANDARDS / SPEC
Cryptohopper API documentation

TRANSPORTS
TCP, TLS

PLATFORMS
Windows, macOS, Linux, iOS, Android

FRAMEWORKS
VCL, FireMonkey, Lazarus / FPC

EDITION
Standard / Professional / Enterprise

Features

- Native Delphi implementation with full ANSI/Unicode support.

Technical specification

Standards & specs	Cryptohopper API documentation
Component class	`TsgcHTTP_Cryptohopper` (unit `sgcHTTP_API_Cryptohopper`)
Frameworks	VCL, FireMonkey, Lazarus / FPC
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.

`TLSOptions`	Published or public property used to configure or query the component.
`CryptohopperOptions`	Published or public property used to configure or query the component.
`OnWebhook`	Published or public property used to configure or query the component.
`ReadTimeout`	Published or public property used to configure or query the component.
`CircuitBreaker`	Published or public property used to configure or query the component.
`OnHTTPAPIException`	Published or public property used to configure or query the component.
`OnHTTPAPISSE`	Published or public property used to configure or query the component.
`Version`	Published or public property used to configure or query the component.

Main methods

The principal public methods exposed by the component.

`DeleteOrder()`	Public function exposed by the component.
`DeleteAllOrders()`	Public function exposed by the component.
`StartWebhook()`	Public procedure exposed by the component.
`StopWebhook()`	Public procedure exposed by the component.

`DeleteHopper()`	Public function exposed by the component.
`DeleteWebhook()`	Public function exposed by the component.
`SendSignal()`	Public function exposed by the component.
`SendTestSignal()`	Public function exposed by the component.
`Post()`	Public function exposed by the component.
`Delete()`	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 **CryptoHopper — Configuration** configuration sourced from the online help.

> **About this scenario.** Requires a Developer Account and once you have been approved you can start to create a new App. The API uses OAuth2 to authenticate, so you can retrieve the client_id and client_secret from your App.

Delphi (VCL / FireMonkey)

```
oCryptoHopper := TsgcHTTP_Cryptohopper.Create(nil);
oCryptoHopper.CryptoHopperOptions.OAuth2.ClientId := 'client_id';
oCryptoHopper.CryptoHopperOptions.OAuth2.ClientSecret := 'client_secret';
oCryptoHopper.CryptoHopperOptions.OAuth2.LocalIP := '127.0.0.1';
oCryptoHopper.CryptoHopperOptions.OAuth2.LocalPort := 8080;
oCryptoHopper.CryptoHopperOptions.OAuth2.Scope.Text := "read,notifications,manage,trade";
```

C++ Builder

```
TsgcHTTP_Cryptohopper oCryptoHopper = new TsgcHTTP_Cryptohopper();
oCryptoHopper->CryptoHopperOptions->OAuth2->ClientId = "client_id";
oCryptoHopper->CryptoHopperOptions->OAuth2->ClientSecret = "client_secret";
oCryptoHopper->CryptoHopperOptions->OAuth2->LocalIP = "127.0.0.1";
oCryptoHopper->CryptoHopperOptions->OAuth2->LocalPort = 8080;
oCryptoHopper->CryptoHopperOptions->OAuth2->Scope->Text = "read,notifications,manage,trade";
```

.NET (C#)

```
TsgcHTTP_Cryptohopper oCryptoHopper = new TsgcHTTP_Cryptohopper();
oCryptoHopper.CryptoHopperOptions.OAuth2.ClientId = "client_id";
oCryptoHopper.CryptoHopperOptions.OAuth2.ClientSecret = "client_secret";
oCryptoHopper.CryptoHopperOptions.OAuth2.LocalIP = "127.0.0.1";
oCryptoHopper.CryptoHopperOptions.OAuth2.LocalPort = 8080;
oCryptoHopper.CryptoHopperOptions.OAuth2.Scope = "read,notifications,manage,trade";
```

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 · How to Configure Webhook

Webhook allows you to receive notifications when something happens in a hopper. Webhooks require a public HTTPs Server which will listen in a URL address all messages sent by cryptohopper. The public server needs to be protected with a SSL certificate (self-signed certificates are not allowed).

Delphi (VCL / FireMonkey)

```
/* OAuth2 */
cryptohopper.CryptohopperOptions.OAuth2.ClientId = 'client_id';
cryptohopper.CryptohopperOptions.OAuth2.ClientSecret := 'client_secret';
cryptohopper.CryptohopperOptions.OAuth2.LocalIP := '127.0.0.1';
cryptohopper.CryptohopperOptions.OAuth2.LocalPort := 8080;
/* Webhook */
cryptohopper.CryptohopperOptions.Webhook.Enabled := True;
cryptohopper.CryptohopperOptions.Webhook.Host := '1.1.1.1';
cryptohopper.CryptohopperOptions.Webhook.Port := 443;
cryptohopper.CryptohopperOptions.Webhook.ValidationCode := '1234';
cryptohopper.CryptohopperOptions.Webhook.SSLOptions.CertFile := 'sgc.pem';
cryptohopper.CryptohopperOptions.Webhook.SSLOptions.KeyFile := 'sgc.pem';
cryptohopper.CryptohopperOptions.Webhook.SSLOptions.RootCertFile := 'sgc.pem';
cryptohopper.CryptohopperOptions.Webhook.SSLOptions.Password := '';
cryptohopper.StartWebhook;
```

C++ Builder

```
/* OAuth2 */
cryptohopper->CryptohopperOptions->OAuth2->ClientId = "client_id";
cryptohopper->CryptohopperOptions->OAuth2->ClientSecret = "client_secret";
cryptohopper->CryptohopperOptions->OAuth2->LocalIP = "127->0->0->1";
cryptohopper->CryptohopperOptions->OAuth2->LocalPort = 8080;
/* Webhook */
cryptohopper->CryptohopperOptions->Webhook->Enabled = True;
cryptohopper->CryptohopperOptions->Webhook->Host = "1.1.1.1";
cryptohopper->CryptohopperOptions->Webhook->Port = 443;
cryptohopper->CryptohopperOptions->Webhook->ValidationCode = "1234";
cryptohopper->CryptohopperOptions->Webhook->SSLOptions->CertFile = "sgc->pem";
cryptohopper->CryptohopperOptions->Webhook->SSLOptions->KeyFile = "sgc->pem";
cryptohopper->CryptohopperOptions->Webhook->SSLOptions->RootCertFile = "sgc->pem";
cryptohopper->CryptohopperOptions->Webhook->SSLOptions->Password = "";
cryptohopper->StartWebhook();
```

.NET (C#)

```
/* OAuth2 */
cryptohopper.CryptohopperOptions.OAuth2.ClientId = "client_id";
cryptohopper.CryptohopperOptions.OAuth2.ClientSecret = "client_secret";
cryptohopper.CryptohopperOptions.OAuth2.LocalIP = "127.0.0.1";
cryptohopper.CryptohopperOptions.OAuth2.LocalPort = 8080;
/* Webhook */
cryptohopper.CryptohopperOptions.Webhook.Enabled = True;
cryptohopper.CryptohopperOptions.Webhook.Host = "1.1.1.1";
cryptohopper.CryptohopperOptions.Webhook.Port = 443;
cryptohopper.CryptohopperOptions.Webhook.ValidationCode = "1234";
cryptohopper.CryptohopperOptions.Webhook.SSLOptions.CertFile = "sgc.pem";
cryptohopper.CryptohopperOptions.Webhook.SSLOptions.KeyFile = "sgc.pem";
cryptohopper.CryptohopperOptions.Webhook.SSLOptions.RootCertFile = "sgc.pem";
cryptohopper.CryptohopperOptions.Webhook.SSLOptions.Password = "";
cryptohopper.StartWebhook();
```

2 · How to Update Cryptohopper Config

Use the UpdateHopper method to update the Hopper Configuration. The method is overloaded so you can pass the JSON string or use the object TsgcHTTPCTHopper and use the properties to enable or disable the Hopper Properties.

Delphi (VCL / FireMonkey)

```
function EnableHopper: string;
var
  oHopper: TsgcHTTPCTHopper;
begin
  oHopper := TsgcHTTPCTHopper.Create;
  Try
    if Cryptohopper.GetHopper('1234', oHopper) then
    begin
      oHopper.Enabled := 1;
      result := Cryptohopper.UpdateHopper('1234', oHopper);
    end;
  Finally
    FreeAndNil(oHopper);
  End;
end;
```

C++ Builder

```
<code class="cpp">
public string EnableHopper()
{
  TsgcHTTPCTHopper *oHopper = new TsgcHTTPCTHopper();
  try
  {
    oHopper->Enabled = 1;
    result = Cryptohopper->UpdateHopper("1234", oHopper);
  }
  __finally
  {
    oHopper->Free();
  }
}</code>
```

.NET (C#)

```
<code class="csharp">
public string EnableHopper()
{
  TsgcHTTPCTHopper oHopper = new TsgcHTTPCTHopper();
  oHopper->Enabled = 1;
  return Cryptohopper->UpdateHopper("1234", oHopper);
}</code>
```

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 — Cryptohopper API documentation	docs.cryptohopper.com/
Delphi demo project (in the sgcWebSockets package)	`Demos\50.Other\03.Cryptohopper`
Component page	www.esegece.com/products/websockets/apis/cryptohopper/
Product page	www.esegece.com/products/websockets/

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