SGCWEBSOCKETS · TECHNICAL DOCUMENT

OAuth2 Provider

TsgcHTTP_OAuth2_Server_Provider — pre-built OAuth 2.0 provider adapters (Google, Microsoft, GitHub, etc.) for the OAuth2 client.

eSeGeCe

WWW.ESEGECE.COM

Overview

This component allows you to integrate External OAuth2 Providers (like Azure AD, Google, Facebook.

At a glance

COMPONENT CLASS

`TsgcHTTP_OAuth2_Server_Provider`

STANDARDS / SPEC

OAuth 2.0 — RFC 6749

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	OAuth 2.0 — RFC 6749
Component class	`TsgcHTTP_OAuth2_Server_Provider` (unit `sgcHTTP_OAuth2_Server_Provider`)
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.

`OAuth2Options`	Resource-server configuration: outbound HTTP client tuning, server-side cookie store and list of private endpoints that require a valid Bearer token.
`Version`	Read-only string exposing the sgcWebSockets library version.

Main methods

The principal public methods exposed by the component.

`IsOAuth2TokenValid()`	Validates the Bearer token presented with an inbound request against the Resource Server cache, either by parsing the request headers or by taking the raw token string.
`Get()`	Sends an HTTP GET request to a remote URL through the Resource Server OAuth2 pipeline, injecting the Bearer token (and DPoP proof when enabled) that matches the caller's session.
`Post()`	Sends an HTTP POST request to a remote URL through the Resource Server OAuth2 pipeline, injecting the Bearer token (and DPoP proof when enabled) that matches the caller's session.
`Put()`	Sends an HTTP PUT request to a remote URL through the Resource Server OAuth2 pipeline, injecting the Bearer token (and DPoP proof when enabled) that matches the caller's session.

`Delete()`	Sends an HTTP DELETE request to a remote URL through the Resource Server OAuth2 pipeline, injecting the Bearer token (and DPoP proof when enabled) that matches the caller's session.
`Patch()`	Sends an HTTP PATCH request to a remote URL through the Resource Server OAuth2 pipeline, injecting the Bearer token (and DPoP proof when enabled) that matches the caller's session.
`AddToken()`	Inserts an externally-issued access/refresh token into the Resource Server token cache so subsequent Bearer-token validations succeed without a round-trip to the external identity provider.
`RemoveToken()`	Revokes a Bearer token currently held in the Resource Server cache, looking it up by the server-side session identifier.
`IsPrivateEndpoint()`	Returns whether a given URL is flagged as private and therefore requires a valid Bearer token / session cookie to be served.
`RegisterApp()`	Registers an OAuth 2.0 client application on the Resource Server and returns its generated credentials.

Public events

The component exposes the following published events; consult the online help for full event-handler signatures.

`OnOAuth2BeforeRequest`	TsgcHTTP_OAuth2_Server_Provider › Events › OnOAuth2BeforeRequest
`OnOAuth2IsPrivateEndpoint`	TsgcHTTP_OAuth2_Server_Provider › Events › OnOAuth2IsPrivateEndpoint
`OnOAuth2ProviderTokenUnknown`	TsgcHTTP_OAuth2_Server_Provider › Events › OnOAuth2ProviderTokenUnknown
`OnOAuth2ProviderTokenValid`	TsgcHTTP_OAuth2_Server_Provider › Events › OnOAuth2ProviderTokenValid

Quick Start

Drop the component on a form, configure the properties below and activate it. The snippet that follows shows the typical **OAuth2 Provider | Private Endpoints** configuration sourced from the online help.

> **About this scenario.** Every time the Server receives a HTTP Request, the event OnOAuth2IsPrivateEndpoint is called to ask if the Endpoint is private or not. By default, is not private.

Delphi (VCL / FireMonkey)

```
procedure OnOAuth2IsPrivateEndpoint(Sender: TObject; const aEndpoint: string; var IsPrivate: Boo
begin
  if aEndpoint = '/private' then
    IsPrivate := True;
end;
```

C++ Builder

```
void OnOAuth2IsPrivateEndpoint(TObject *Sender, const string aEndpoint, ref bool IsPrivate)
{
  if (aEndpoint == "/private")
  {
    IsPrivate = True;
  }
}
```

.NET (C#)

```
void OnOAuth2IsPrivateEndpoint(TObject *ender, const string aEndpoint, ref bool IsPrivate)
{
  if (aEndpoint == "/private")
  {
    IsPrivate = 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 · OAuth2 Provider | Requests

Once the Authentication has been successful, you can send requests to the OAuth2 Protected Server using the Public ID Token stored as a cookie.

Delphi (VCL / FireMonkey)

```
procedure OnCommandGet(AContext: TIdContext; ARequestInfo: TIdHTTPRequestInfo; AResponseInfo: TI
begin
  if ARequestInfo.Document = '/private' then
  begin
    // return OAuth2 profile data
    AResponseInfo.ContentText := OAuth2Provider.Get(ARequestInfo, 'https://graph.microsoft.com/v
    AResponseInfo.ContentType := 'application/json';
    AResponseInfo.ResponseNo := 200;
  end
  else
    AResponseInfo.ResponseNo := 404;
end;
```

C++ Builder

```
void OnCommandGet(TIdContext *AContext, TIdHTTPRequestInfo *ARequestInfo, TIdHTTPResponseInfo *A
{
  if (ARequestInfo->Document == "/private"
  {
    // return OAuth2 profile data
    AResponseInfo->ContentText = OAuth2Provider->Get(ARequestInfo, "https://graph.microsoft.com/
    AResponseInfo->ContentType = "application/json";
    AResponseInfo->ResponseNo = 200;
  }
  else
  {
    AResponseInfo->ResponseNo = 404;
  }
}
```

.NET (C#)

```
void OnCommandGet(TIdHTTPRequestInfo ARequestInfo, TIdHTTPResponseInfo AResponseInfo)
{
  if (ARequestInfo.Document == "/private"
  {
    // return OAuth2 profile data
    AResponseInfo.ContentText = OAuth2Provider.Get("ID Token", "https://graph.microsoft.com/v1.0
    AResponseInfo.ContentType = "application/json";
    AResponseInfo.ResponseNo = 200;
  }
  else
  {
    AResponseInfo.ResponseNo = 404;
  }
}
```

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 — OAuth 2.0 — RFC 6749	datatracker.ietf.org/doc/html/rfc6749
Online help — component page	www.esegece.com/help/sgcWebSockets/Components/HTTP/Authorization/OAuth2/Provider/TsgcHTTP_OAuth2_Server_Provider.htm
Delphi demo project (in the sgcWebSockets package)	`Demos\20.HTTP_Protocol\08.OAuth2_ServerProvider`
Component page	www.esegece.com/products/websockets/http/oauth2-provider/
Product page	www.esegece.com/products/websockets/

> **Document scope.** This document covers the publicly-documented surface of the OAuth2 Provider component shipped with sgcWebSockets. For full property, method and event reference consult the online help linked above.