FXCM Global Services, LLC

FXCM ForexConnect Lite 3rd Party OAUTH

User Guide

**Date: 11/29/2022**

**Author: Matthew Schwartz**

**Revised by:**

**Revision: 1**

Contents

[1 Introduction 2](#_Toc39165189)

[2 Grant Type: auth\_code 3](#_Toc39165190)

[3 Grant Type: implicit 4](#_Toc39165203)

[4 FXCM Sample Application 5](#_Toc39165211)

[5 Revision History 6](#_Toc39165222)

# Introduction

The purpose of this document is to describe the functionality and workflow of FXCM’s 3rd Party OAUTH authentication.

FXCM’s Forex Connect Lite API requires a JWT to be passed in order to authenticate an end user and establish a trading session. The 3rd party OAUTH authentication model allows a 3rd party application to direct the end user to FXCM’s authentication page. Upon successful authentication, FXCM will redirect back to the 3rd party application and provide the token. This model follows standard OAUTH2 protocol and requires each 3rd party to register their callback URL with FXCM and get issued a client\_id and client\_secret.

This document outlines the steps a 3rd party should take in order to integrate with the OAUTH authentication model, as well as provide instructions to access and run sample a sample front end application.

# Grant Type: authorization\_code

The *authorization\_code* grant type allows for the token exchange to occur in the back channel and is completely hidden to the end user.

**OAUTH 2.1 RFC:** <https://datatracker.ietf.org/doc/html/draft-ietf-oauth-v2-1-01>

## 3rd Party application must launch FXCM’s login page

### https://{fxcm\_domain}/oauth/authorize?client\_id={client\_id}&response\_type=code&redirect\_uri={redirect\_uri}&scope=openid%20trading

### Required parameters:

#### fxcm\_domain is environment specific

##### Production Real: tradingstation.fxcm.com

##### Production Demo: tradingstation.fxcm.com

##### UAT: titanx.fxcorporate.com

##### QA: titanx.fxcorporate.com

#### client\_id: string value provided by FXCM and must match exactly

#### response\_type: always ‘code’

#### redirect\_uri: must be agreed between FXCM and 3rd party and must match exactly

##### Multiple redirect\_uri are available upon request

#### scope: always ‘openid trading’

### Optional parameters for PKCE support:

#### code\_challenge\_method= Set to S256 to indicate that SHA-256 hashing is used to transform the code verifier.

#### code\_challenge= The BASE64URL-encoded SHA-256 hash of a random 32 bytes called code verifier which the client must generate and store internally and which is intended to prevent code injection and CSRF attacks. Originally specified in the PKCE extension (RFC 7336) to OAuth 2.0.

#### state= Optional opaque value set by the client which the authorisation server will echo verbatim in the authorisation response. Enables the client to encode application state information to appear at the redirect\_uri.

#### nonce= String value used to associate a Client session with an ID Token, and to mitigate replay attacks. The value is passed through unmodified from the Authentication Request to the ID Token. If present in the ID Token, Clients MUST verify that the nonce Claim Value is equal to the value of the nonce parameter sent in the Authentication Request. If present in the Authentication Request, Authorization Servers MUST include a nonce Claim in the ID Token with the Claim Value being the nonce value sent in the Authentication Request. Authorization Servers SHOULD perform no other processing on nonce values used. The nonce value is a case sensitive string.

## The end user must enter their FXCM login credentials

## The user will be prompted to and must subsequently approve the request

## FXCM Auth Server will redirect back to {redirect\_uri} with the query parameter: code

## 3rd party application will receive response with header.location: {redirect\_uri}?code={code}

## 3rd party application must submit a POST as follows:

### <https://{fxcm_domain}/oauth2/token>

#### fxcm\_domain must match from step 2.1

### POST BODY must contain:

#### code: value from ‘code’ query parameter in step 2.5

#### grant\_type: ‘authorization\_code’

#### client\_id: string value provided by FXCM

#### client\_secret: string value provided by FXCM

#### redirect\_uri: exact redirect\_uri from step 2.1

## Client receives JSON response: {access\_token: *access\_token,* refresh\_token: *refresh\_token*, token\_type: ‘Bearer’}

### access\_token: the API access token as a JWT to create a session with ForexConnect Lite

### refresh\_token: a refresh token which can be used to get a new access token

## 3rd party application may submit post to get new access token through the refresh mechanism

### https://{fxcm\_domain}/oauth2/token

#### POST BODY must contain:

##### grant\_type: ‘refresh\_token’

##### redirect\_uri: must be agreed between FXCM and 3rd party and must match exactly

##### client\_id: string value provided by FXCM and must match exactly

##### client\_secret: must be agreed between FXCM and 3rd party and must match exactly

##### refresh\_token: value received from json response in step 2.7

# FXCM Sample Application

## Visit FXCM’s Github page

### <https://github.com/FXCM/3rd-party-oauth>

## Clone the master branch of the project:

### Note the client\_id, client\_secret, and redirect\_urls in the sample are not configured in the test environment. Please reach out to FXCM customer support to request Demo and Real environment access.

## Navigate to the cloned repository and run the command: ‘npm install’

## Run the command: ‘node server’

### Note that node version should be at least v10.16.0

## Visit localhost:[port] on a browser

### Port is set in config.json

## The browser page will display options to verify OAUTH workflow

### Run the browser network tab to view inbound and outbound network requests.

### Log in to FXCM using credentials:

#### Connection: REAL

#### Username: oauth\_user

#### Password: fxcm1234

# Revision History

| Version | Date Revised | Revised By | Description |
| --- | --- | --- | --- |
| 1.0 | 11/29/2022 | Matthew Schwartz | Document Creation, First Draft |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |