

Eze EMS

Release Date: Oct 16, 2025
v2025.5.1

Eze EMS xAPI Go Sample Application Guide

This document contains information about getting started with Eze EMS xAPI using Go.



Confidentiality Notice: The information included in this document is confidential information of SS&C Eze and is intended only for SS&C Eze and its affiliates, Eze EMS clients, and their respective employees.

Table of Contents

| | |
|---------------------------------|----------|
| What's New | 3 |
| Introduction | 4 |
| Eze EMS xAPI Basics | 4 |
| Eze EMS xAPI Use Restrictions | 4 |
| Eze EMS xAPI Version | 5 |
| Download EMS xAPI | 5 |
| Developer Support | 5 |
| Getting Started Using Go | 6 |
| Prerequisites | 6 |
| Running EMS xAPI in Go | 6 |
| Using the Command Line | 6 |
| Building the Project | 8 |

Legal Information

Copyright

This document is the copyrighted work of Eze Castle Software LLC ("SS&C Eze"). SS&C Eze distributes this document pursuant to a subscription agreement containing confidentiality and license provisions and is solely for the benefit of its authorized licensees. This documentation may not be copied or transmitted, in whole or in part, in any form or by any means without the express written consent of SS&C Eze.

© 1997 to 2025 Eze Castle Software LLC. All Rights Reserved.

Content

Information in this document is subject to change without notice. In the event that you are using a version of SS&C Eze products other than the most recent version, there may be discrepancies between the content of this documentation and the operation or visual presentation of your older version of the product. SS&C Eze does not warrant that this documentation is error free.

Trademarks

SS&C Eze is a trademark of SS&C Technologies, Inc. All SS&C Eze company and product names are trademarks or registered trademarks of SS&C Technologies, Inc. or SS&C Eze.

All other company or product names mentioned herein are the trademarks or registered trademarks of their respective companies.

SS&C Eze

<http://www.ezesoft.com/>

What's New

This release includes the following updates:

| Version No. | Date | Summary of Update |
|----------------|--------------|--|
| v2025.5.1.2717 | Oct 16, 2025 | Added a note about EMS xAPI login limitations. |



Note: Refer to the [Revision History](#) section for detailed information on past release versions.

Introduction

The purpose of this document is to help clients get started with the EMS xAPI application using Go language. This document provides a step-by-step process of generating the scripts in Go language and start using the APIs.

Eze EMS xAPI is robust and easy-to-use application that allows programmers and trading businesses to complete various trading workflows, and also access key information, including:

- Automating order routing - to smart order routers, algorithms and other trading systems.
- Routing orders to multiple brokers, dark pools, ATS, and MTFs via the Eze EMS Global Routing Network - across asset classes.
- Staging or routing single or pairs orders.
- Accessing balances, positions, executions, and other order details.
- Accessing comprehensive list and basket capabilities.

Although EMS xAPI can operate with all gRPC compatible languages, only Go language references are provided in this document as an example. Refer to this [link](#) for more information on gRPC.

Eze EMS xAPI Basics

The Eze EMS xAPI operates in conjunction with your existing Eze EMS account permissioning and entitlements. **The Eze EMS xAPI is not a standalone data feed application that is provided to you independent of the Eze EMS.** Please contact Eze Client Service if you need to request or make changes to appropriate permissions for your account.

Eze EMS xAPI Use Restrictions

As an Eze EMS xAPI user, you are prohibited from retransmitting any Eze Market Data using the Eze EMS xAPI, without the express prior written consent of Eze EMS and the exchanges or other third-party data providers (referred to as **“Sources”** in your end user agreement). Any unauthorized retransmission of Eze Market Data is a breach of your end user agreement and will cause immediate termination of your use of the Eze EMS, Eze Market Data, and the Eze EMS xAPI.

Any non-display usage of Eze Market Data, such as use of real-time data in algorithmic trading or program trading, is subject to the rules, regulations, and policies of the applicable exchanges and additional exchange fees may apply. In addition, you may have a non-display usage of Eze Market Data even if a display of real-time data occurs. Please review your Eze EMS end user agreement, and the exchanges' and third-party data providers' rules, regulations, and policies that apply to your use of the Eze EMS API (which apply to Eze EMS xAPI) and/or Eze Market Data. It is the sole responsibility of the Eze EMS xAPI user and each user receiving, directly or indirectly accessing or otherwise using Eze Market Data to determine whether your receipt, access or use is reportable and/or fee liable.

Eze EMS xAPI Version

This document covers all the APIs and updates to the Eze EMS xAPI that are part of 2025.5.1.2717 release.

Download EMS xAPI

Contact your SS&C Eze client service representative for downloading Eze EMS xAPI.

Developer Support

- If you are an existing Eze EMS user, [log in](#) to access developer support documentation and sample code.
- You can [contact us](#) or [request a demo](#) if you want to explore more about EMS xAPI.
- You can send us an e-mail apisupport@ezesoft.com or call +1 312-442-8122.

Getting Started Using Go

You can now launch Eze EMS xAPI using the Go programming language. This Go-based client library is designed for interacting with APIs using gRPC and Protocol buffers.

This guide provides step-by-step instructions to set up, configure, and run the project.

Prerequisites

Ensure the following prerequisites before running EMS xAPI with Go:

- Go version 1.20 or higher
- Protobuf Compiler (protoc) version 3.20.0 or higher
- Operating System: Windows, macOS, or Linux

Running EMS xAPI in Go

You can run the EMS xAPI in Go using either of the following methods:

- [Using the command line](#)
- [Building the project](#)

Using the Command Line

You can quickly run EMS xAPI in Go by configuring the **config.ini** file and executing sample scripts directly from the command window.

To start using the EMS xAPI using Go:

1. **Install Go:** Download and install Go 1.20 or higher version from <https://go.dev/dl/>.

To verify Go installation, run the following command:

```
C:\Users\USERNAME>go version
```

2. **Download Sample Files:** You can download the latest Go sample files from the Eze EMS xAPI GitHub repository, refer [xapi_golang_sample_scripts.zip](#). Unzip the files.
3. **Configure config.ini:** Navigate to the **config** folder in the downloaded sample files folder in Step 2 and update the following details:
 - **password** — Authentication password.
 - **server** — Server address.
 - **user** — Username for authentication.

- **domain** — Authentication domain.
- **locale** — Locale setting.
- **port** — Server port number.
- **ssl** — Set to **true** or **false** (encryption recommended).
- **cert_file_path** — *(optional)* Path to certificate file.
- **keepAliveTime** — *(optional)* Keep-alive time in milliseconds.
- **keepAliveTimeout** — *(optional)* Keep-alive timeout in milliseconds.
- **maxRetryCount** — *(optional)* Maximum retry attempts.
- **retryDelayMS** — *(optional)* Delay between retries in milliseconds.
- **route** — *(optional)* Route information.
- **account** — *(optional)* Format: **BANK;BRANCH;CUSTOMER;DEPOSIT**.



Note: Ensure the **roots_qa.pem** CA certificate file is included in the **config** folder (included in the ZIP).



Note: Your account will be temporarily locked for 3 minutes if you attempt to log in three (3) times within 60 seconds, regardless of whether the attempts are successful or unsuccessful.

4. **Run the Client:** Navigate to the **xapi_golang_sample_scripts** folder and run the following command:

```
go run cmd/example-client/main.go
```

5. **Run Specific APIs:** To run a specific API (e.g., **GetTodaysActivity**), run the following command.

```
go run cmd/example-client/main.go --api=GetTodaysActivity
```

To run multiple APIs (e.g., **GetTodaysActivity** and **SubmitSingleOrder**), use commas to separate the APIs and run the following command.

```
go run cmd/example-client/main.go --api=GetTodaysActivity,SubmitSingleOrder
```

Building the Project

You can also setup the project by compiling the **.proto** files and generating Go code using protoc and gRPC plugins to build a standalone executable.

To build the EMS xAPI Go client from source:

1. **Install protoc:** Download the latest version from <https://github.com/google/protobuf/releases>.

For installation help, refer to <https://protobuf.dev/installation/>.

To verify the protoc installation, run the following command:

```
protoc --version
```

2. **Install Go Plugins for protobuf:** To install and verify the protobuf installation, run the following command:

```
go install google.golang.org/protobuf/cmd/protoc-gen-go@latest
protoc-gen-go --version
```

3. **Install Go plugins for gRPC:** To install and verify the protobuf installation, run the following command:

```
go install google.golang.org/grpc/cmd/protoc-gen-go-grpc@latest
protoc-gen-go-grpc --version
```

4. **Generate Go Code from .proto Files:**

- a. Create an *empty* folder named **generated** inside **xapi_golang_sample_scripts**. Then run the following command:

```
protoc --proto_path=proto --go_out=generated --go-grpc_out=generated proto/*.proto
```

Running the command generates the **.go** files for Market Data, Order and Utilities proto.

5. **Build the Executable:** To generate an executable file (**example-client.exe**), run the following command:

```
go build -o example-client.exe cmd/example-client/main.go
```

6. **Run Specific APIs:** To run a specific API (e.g., **GetTodaysActivity**), run the following command.

```
go run cmd/example-client/main.go --api=GetTodaysActivity
```

To run multiple APIs (e.g., **GetTodaysActivity** and **SubmitSingleOrder**), use commas to separate the APIs and run the following command.

```
go run cmd/example-client/main.go --api=GetTodaysActivity,SubmitSingleOrder
```