ELO Sync

Configuration

Table of contents

Service configuration 3
Templates for logging configurations 19

Service configuration

This section describes how to configure the ELO Sync service either using the `appsettings.json` and `appsettings.Production.json` files in the local installation or by changing the user settings stored in the ELO repository.

We recommend copying the supplied `appsettings.json` file into a new file `appsettings.Production.json` during installation and updating the settings accordingly.

As the `appsettings.Production.json` file is never added to the distributed package, it is never overwritten during updates.

For technical reasons, some settings must be configured in the `appsettings.json` file and cannot be changed in the web interface or the REST API.

Each section always contains an example that describes the individual section within the complete file. If you want to combine several sections, you must omit the outer brackets.

Example:

```
{
  "Key": {            // First line to be copied
    "MySetting": 42
  }                   // Last line to be copied
}
```

The sections are listed in alphabetical order for easier navigation.

AllowedHosts

This configuration setting defines the host filtering that is used by the built-in Kestrel server.

```
{
  "AllowedHosts": "*"
}
```

Api

This section contains the settings for using the REST API.

```
{
  "Api": {
    // Indicates whether to use single-tenant login.
    // If the app type 'SingleTenant' is selected for ELO Sync,
    // this setting has to be set to true. As a result,
```

```
    // login is performed directly on this tenant.
    "ForceSingleTenantLogin": false
  }
}
```

AzureAd

This section contains the settings for the (static) app registration in Microsoft Azure. The Azure settings can also be defined in the ELO repository, see Configuration stored in the ELO repository.

```
{
  "AzureAd": {
    // The Entra ID endpoint used for authentication must be changed for national cloud dep
    "Instance": "https://login.microsoftonline.com/",

    // The domain name under which this application is registered
    "Domain": "mydomain.onmicrosoft.com",

    // The application (client) ID of the registered application
    "ClientId": "00000000-0123-0123-0123-c0ffeec0ffee",

    // The tenant ID of the registered application
    "TenantId": "c0ffee00-0123-4567-4567-000000000000",

    // The client secret used for authentication
    "ClientSecret": "SecretGeneratedByAzure",

    // The callback path used for the OAuth Redirect results
    "CallbackPath": "/signin-oidc-custom",

    // Client certificate(s) used for authentication.
    "ClientCertificates": [
      {
        // The properties used to read the certificate must have one of the following value
        // "Path" - The private key of the certificate is stored in a local file, optionall
        //          The properties "CertificateDiskPath" and "CertificatePassword" should b
        // "StoreWithThumbprint" - The private key is stored in a Windows certificate store
        //          The properties "CertificateStorePath" and "CertificateThumbprint" shoul
        // "StoreWithDistinguishedName" - The private key is stored in a Windows certificat
        //          The properties "CertificateStorePath" and "CertificateDistinguishedName
        "CertificateSource": "Path",

        // Path to the certificate store
        "CertificateStorePath": "CurrentUser/My",
        // Certificate thumbprint
```

```
        "CertificateThumbprint": "0123456789abcdef",
        // Certificate distinguished name
        "CertificateDistinguishedName": "MyCertificateName",

        // Password for decrypting the certificate
        "CertificatePassword": "MySecretPassword",

        // Path to the certificate file
        "CertificateDiskPath": "C:\\path\\to\\certificate.pfx"
      }
    ]
  }
}
```

ConnectionStrings

This section contains the connection strings used by the various database providers.

Only the connection string of the configured database provider must be set.

```
{
  "ConnectionStrings": {
    "Postgres": "User ID=myUsername;Password=myPassword;Server=mydbserver;Port=5432;Databas
    "MsSql": "Server=mydbserver,1433;Database=MySyncDb;User Id=myUsername;Password=myPasswo
  }
}
```

Cookies

This section contains the settings for the cookies used by ELO Sync.

```
{
  "Cookies": {
    // The lifetime of the authentication cookie
    "ExpireTimeSpan": "00:30:00"
  }
}
```

> **Information**
>
> 1. Changes to these settings only take effect after the service has been restarted.
> 2. Existing cookies keep their original lifetime until they expire. If an old cookie has a lifetime of one hour and this setting is changed to 30 minutes, the old cookie can still be used 45 minutes later; after this, it is replaced with a 30-minute cookie.

Database

This configuration key defines the database that ELO Sync will use.

After you have set this key, you also have to set the corresponding ConnectionString.

```
{
  // The type of the database provider to be used.
  // Must be one of the following values:
  // "Postgres" - PostgreSQL database (https://www.postgresql.org/)
  //              Version: 14.x or higher, 16.x recommended
  // "Sqlite"   - Sqlite database (https://sqlite.org/);
  //              NOTE: Should only be used for test purposes, as it is up to 10x slower th
  // "MsSql"    - Microsoft SQL Server (https://www.microsoft.com/en-us/sql-server/sql-serv
  //              Version: 2012 or higher, 2019 recommended
  // "Oracle"   - Oracle database (https://www.oracle.com/database/)
  //              Version: 11g release 2 or higher, 19c recommended
  "Database": "Postgres"
}
```

DownstreamApis

This key contains the configuration settings for all third-party APIs that are used by ELO Sync.

Each API must have the property `Api`, which is used to identify the API. All other properties are specifically for the configured API.

DownstreamApis:Microsoft Graph

This section contains the specific settings for access to the Microsoft Graph API.

```
{
  "DownstreamApis": [
    {
      // Fixed name for the Microsoft Graph API
      "Api": "Microsoft Graph",
      // Base URL for access to Microsoft Graph, only has to be changed for national cloud
      // NOTE: Currently, only the stable endpoint /v1.0 is supported. The endpoint /beta i
      "BaseUrl": "https://graph.microsoft.com/v1.0",
      // The permissions to be requested when using this API
      "Scopes": "user.read Sites.Read.All Sites.ReadWrite.All",
      // The scope used when authenticating with this API
      "DefaultGraphScope": "https://graph.microsoft.com/.default"
    }
  ]
}
```

HttpClient

This section contains the general settings for HTTP(S) connections established by ELO Sync.

These settings apply only if there are no more specific settings elsewhere. In this case, the more specific settings take precedence and cannot be overridden by this general setting. This also applies if the more specific setting is not present in the file, in which case the default value for the more specific setting is used.

```
{
  "HttpClient": {
    // The timeout in seconds for all HTTP requests
    "Timeout": 100,
  }
}
```

InstanceName

The optional instance name can be used for horizontal scaling of the ELO Sync service. When you set this name, additional configuration files from the ELO repository can be loaded. See Configuration stored in the ELO repository for more information.

```
{
  "InstanceName": "EloSync1"
}
```

Important

There is no support for multiple instances of ELO Sync using the same database.

The instance name only affects the application configuration so that a separate database can be used for each instance.

Kestrel

This configuration section contains the settings for the built-in Kestrel web server.

Information

The user endpoints must use the HTTPS protocol for ELO Sync to access Microsoft 365.

An HTTP endpoint can only be used if a reverse proxy is upstream of the ELO Sync server and the proxy is configured to use HTTPS.

See also:

- Kestrel Endpoints: Configure HTTPS in appsettings.json

```
{
  "Kestrel": {
    "Endpoints": {
      "HttpsInlineCertFile": {
        "Url": "https://my-elo-sync-server:7224",
        "Certificate": {
          "Path": "C:\\path\\to\\certificate.pfx",
          "Password": "MySecretPassword"
        }
      }
    }
  }
}
```

Logging

The *Microsoft.Extension.Logging* component used by ELO Sync is configured in this section.

Normally, this component redirects all logging to Serilog, but is also used to change the logging level of individual components.

See Configure logging without code for more information.

```
{
  "Logging": {
    "LogLevel": {
```

```
      // Use Information and higher for all unconfigured components
      "Default": "Information",
      // Use Error and higher for all Microsoft components
      "Microsoft": "Error",
      // Use Debug and higher for all ELO components
      "Elo": "Debug"
    }
  }
}
```

OAuth

This section contains the settings for using OAuth with ELO Sync.

```
{
  "OAuth": {
    // The ID of the OAuth configuration to be used with the ELOauth plug-in.
    "ConfigId": "elo_sync_oauth"
  }
}
```

The selected name for the OAuth configuration profile must also be entered in the configuration of the ELOauth plug-in for the Indexserver.

The OAuth ConfigId can also be defined in the ELO repository, see Configuration stored in the ELO repository.

Note the documentation for the ELOauth plug-in to configure an authentication profile.

OpenTelemetry

ELO Sync uses the OpenTelemetry infrastructure for tracing and monitoring the status of the service.

Each area can be configured separately and is disabled by default.

OpenTelemetry:Logging

The Log Exporter for OpenTelemetry is configured in this section using the OTLP protocol.

```
{
  "OpenTelemetry": {
    "Logging": {
      // Whether the OTLP Log Exporter should be enabled
      "Enabled": true,
```

```
      // The endpoint to be sent the logs
      "Endpoint": "http://localhost:4317/v1/logs",

      // The protocol used to send the logs to the endpoint, must be one of the following v
      // "HttpProtobuf" - HTTP protocol that uses the Protobuf format
      // "Grpc" - gRPC protocol (default)
      "Protocol": "HttpProtobuf",

      // Additional headers to be sent with each request, typically used for authentication
      "Headers": "Authorization: Bearer MySecret",

      // Exporter options
      "BatchExportProcessorOptions": {
        // See below for more details
      }
    }
  }
}
```

OpenTelemetry:Tracing

The Trace Exporter for OpenTelemetry is configured in this section using the OTLP protocol.

```
{
  "OpenTelemetry": {
    "Tracing": {
      // Whether the OTLP Trace Exporter should be enabled
      "Enabled": true,

      // The endpoint to be sent the traces
      "Endpoint": "http://localhost:4317/v1/traces",

      // The protocol used to send the traces to the endpoint, must be one of the following
      // "HttpProtobuf" - HTTP protocol that uses the Protobuf format
      // "Grpc" - gRPC protocol (default)
      "Protocol": "HttpProtobuf",

      // Additional headers to be sent with each request, typically used for authentication
      "Headers": "Authorization: Bearer MySecret",

      // Exporter options
      "BatchExportProcessorOptions": {
        // See below for more details
      }
    }
```

```
  }
}
```

OpenTelemetry:Metrics

The Metrics Exporter for OpenTelemetry is configured in this section using the OTLP protocol.

```
{
  "OpenTelemetry": {
    "Metrics": {
      // Whether the OTLP Metrics Exporter should be enabled
      "Enabled": true,

      // The endpoint to be sent the metrics
      "Endpoint": "http://localhost:4317/v1/metrics",

      // The protocol used to send the metrics to the endpoint, must be one of the followin
      // "HttpProtobuf" - HTTP protocol that uses the Protobuf format
      // "Grpc" - gRPC protocol (default)
      "Protocol": "HttpProtobuf",
      // Additional headers to be sent with each request, typically used for authentication
      "Headers": "Authorization: Bearer MySecret",

      // Exporter options
      "BatchExportProcessorOptions": {
        // See below for more details
      }
    }
  }
}
```

OpenTelemetry:*:BatchExportProcessorOptions

This section is used by all OpenTelemetry exporters and configures the Batch Export Processor.

```
{
  "OpenTelemetry": {
    "*": {
      "BatchExportProcessorOptions": {
        // Delay between two planned exports in milliseconds
        "ScheduledDelayMilliseconds": 5000,

        // The number of items per export batch
        "MaxExportBatchSize": 512,
```

```
        // The maximum number of items in the queue before further items are discarded
        "MaxQueueSize": 2048,

        // The timeout in milliseconds after which an export is canceled
        "ExporterTimeoutMilliseconds": 30000
      }
    }
  }
}
```

OpenTelemetry:Zipkin

The Zipkin Exporter is configured in this section.

```
{
  "OpenTelemetry": {
    "Zipkin": {
      // To use the Zipkin Exporter
      "Enabled": true,

      // The endpoint the traces are to be sent to
      "Endpoint": "http://localhost:9411/api/v2/spans"
    }
  }
}
```

PrometheusServer

The server for the Prometheus metric provider is configured in this section.

```
{
  "PrometheusServer": {
    // The endpoint URL where the metrics will be published
    // https is not currently supported
    "Endpoint": "http://localhost:9090/metrics",

    // Defines whether access is allowed via any host name
    "AllowAllHosts": true
  }
}
```

PublicUrl

This entry contains the public URL that can be used to access ELO Sync if the access URL is different from the internal URL set in the Kestrel section, e.g. due to the use of a proxy.

```
{
  "PublicUrl": "https://my-elo-server:9093/ix-Repository1/plugin/de.elo.ix.plugin.proxy/syn
}
```

Repositories

All ELO repositories that can be accessed when using ELO Sync are listed in this configuration section.

A user can select one of these repositories when signing into the web interface.

```
{
  "Repositories": [
    {
      // Required; the display name for the server in the user interface
      "name": "My ELO Server",

      // Required; a unique key for this server. This key
      // is also used when calling the REST API to identify the repository.
      "key": "MyELOServer",

      // Required; the ELOix endpoint to be used to access the repository
      "url": "https://my-elo-server:9093/ix-Repository1/ix",

      // Optional; the ELO Web Client URL used for the preview or linking with existing doc
      // If not defined, the ELO Web Client URL from the property "url" is used
      "webclienturl": "https://my-elo-server:9093/ix-Repository1/plugin/de.elo.ix.plugin.pr

      // Optional; the OAuth callback URL used by this repository
      "oauthcallbackurl": "https://my-elo-server:9093/ix-Repository1",

      // The logon mode used by this repository; must be one of the following values:
      // "ELOauth" - The ELOauth plug-in is used for authentication, which is provided for
      // "auth2"   - The Auth2 plug-in is used for authentication
      "loginmode": "auth2",

      // Cookie token exchange URL used to authenticate an ELO session with an access token
      // This property is ignored if "loginmode" is set
      "cookietokenexchangeurl": "https://my-elo-server:9093/ix-Repository1/plugin/de.elo.ix
```

```
      // The login URL used if a user is redirected to an ELO web user interface
      // ¶ClientUrl¶ is replaced with the target URL
      // This property is ignored if "loginmode" is set
      "loginurl": "https://my-elo-server:9093/ix-Repository1/plugin/de.elo.ix.plugin.auth/l

      // Optional; the timeout in seconds for access to the repository
      // The default value is 10 seconds, as ELO Sync often runs on the same devices as the
      "timeout": 10
    }
  ]
}
```

For simplicity, several template configurations for repositories are listed in the following.

```
{
  "Repositories": [
    // >= ELO 23.4
    {
      "name": "My Repository",
      "key": "MyRepository",
      "url": "https://my-elo-server:9093/ix-Repository1/ix",
      "loginmode": "auth2"
    },
    // < ELO 23.4
    {
      "name": "My Repository",
      "key": "MyRepository",
      "url": "https://my-elo-server:9093/ix-Repository1/ix",
      "loginmode": "ELOauth"
    },
    // >= ELO 23.4; with public access to the ELO Web Client
    {
      "name": "My Repository",
      "key": "MyRepository",
      "url": "https://my-elo-server:9093/ix-Repository1/ix",
      "loginmode": "auth2",
      // Public URLs for web &amp;amp; login; same domain must be used
      "webclienturl": "https://elo.company.com/web/",
      "loginurl": "https://elo.company.com/authorize?redirect_uri=¶ClientUrl¶&response_type
    },
    // < ELO 23.4; with public access to the ELO Web Client
    {
      "name": "My Repository",
      "key": "MyRepository",
      "url": "https://my-elo-server:9093/ix-Repository1/ix",
```

```
      "loginmode": "ELOauth",
      // Public URLs for web & login; same domain must be used
      "webclienturl": "https://elo.company.com/web/",
      "loginurl": "https://elo.company.com/login?clientUrl=¶ClientUrl¶&configId=¶ConfigId¶"
    }
  ]
}
```

Login mode

The property `loginmode` controls the additional properties required for authentication with an ELO repository.

Depending on the ELO Indexserver version and configuration, it can be set to either `ELOauth` or `auth2`.

`ELOauth` designates the use of the ELOauth plug-in for authentication, which is supported by ELO Indexserver versions before 23.4.

`auth2` designates the use of the Auth2 plug-in for authentication, which is supported by ELO Indexserver version 23.4 and higher.

Cookie token exchange URL

This URL is used to authenticate an ELO session with the help of an access token from Azure/Entra ID.

The entered URL can contain a placeholder `¶ConfigId¶`, which is replaced by the setting `OAuth:ConfigId`.

Alternatively, the parameter `configId` has to be set in the URL manually. ELO Sync does not check whether the URL is correct.

If the property is entered manually, the setting `OAuth:ConfigId` is ignored and the parameter `configId` must correspond to the configuration required for ELO Sync.

The URL entered must end with the `jwt_token` query parameter, as the access token is simply added at the end.

> **Information**
>
> This may change in a future version, depending on other requirements.

If the property is not entered, a default URL is created using the basic URL of the original authentication plug-in (`de.elo.ix.plugin.auth`).

If the Auth2 plug-in is used for authentication (enabled for ELOix 23.4+ by default), this property must be entered explicitly.

Example:

```
{
  "cookietokenexchangeurl": "https://my-elo-server:9093/ix-Repository1/plugin/de.elo.ix.plu
}
```

The `configId` parameter must match the configuration that was used for the login configuration in the plug-in.

Serilog

The logging solution currently used by ELO Sync is Serilog.

The sinks and filters for logging are configured in this section. For more details, see Serilog.Settings.Configuration

Below you will find an example that writes a basic log in a rolling log file.

```
{
  "Serilog": {
    "Using": [
      "Serilog.Sinks.File"
    ],
    "MinimumLevel": "Debug",
    "WriteTo": [
      {
        "Name": "File",
        "Args": {
          "fileSizeLimitBytes": 79857600,
          "retainedFileCountLimit": 20,
          "path": "Logs/ELOSync.log",
          "rollingInterval": "Day"
        }
      }
    ],
    "Enrich": [
      "FromLogContext",
      "WithMachineName",
      "WithThreadId"
    ],
    "Properties": {
      "Application": "ELO Sync"
    }
  }
}
```

You will find alternative examples for configuring logging here.

ServiceUser

The settings for the ELO Service user to be used by ELO Sync.

This user must already exist in the ELOam instance and have administrator rights.

```
{
  "ServiceUser": {
    // User in the ELOam instance with administrator rights
    "UserName": "ELO Service",
    // Password for the user
    "Password": "MySecretPassword"
  }
}
```

Regular polling of the configuration from the repository

```
{
  "EnableRepositoryConfigPolling":
  //
  // Checks at regular intervals whether the configuration for ELO Sync in the repository h
  // If the configuration has changed:
  //   - The configuration is loaded from the repository
  //   - /api/status returns HTTP status code 500 with the HTTP header X-RestartNeccessary
  //     . In this case, the ELO Sync service has to be restarted
  //
  "true"
  //
  // or
  //
  // Loads the configuration for ELO Sync from the repository once when starting ELO Sync.
  // Subsequent changes can only be applied by restarting ELO Sync
  //
  "false"
}
```

Save the configuration in the ELO repository

Parts of the application configuration can be stored in an ELO repository.

For this reason, the local `appsettings.json` (or `appsettings.Production.json`) must have a repository and the setting Service User.

ELO Sync then uses this local configuration, connects to the repository, and inserts the working version of the following json document as a configuration file:

- `Administration/ELO Sync/config/appsettings.public`

  If this document is missing or does not contain a valid configuration, its content is ignored and a warning is logged.

  If multiple documents with this name exist, only the first one (identified by ELOix) is loaded.

  If the `InstanceName` is defined in the local configuration, the following configuration document is loaded in addition to the previous configuration elements.

- `Administration/ELO Sync/config/<InstanceName>/appsettings.public`

  The placeholder above `<InstanceName>` is replaced with the value from the `InstanceName` setting.

Templates for logging configurations

Different templates for the logging configuration are listed on this page. Many of these templates can simply be copied to replace an existing logging configuration.

Log overflow

The log is written to a file that overflows each day or after 75 megabytes.

```
{
  "Serilog": {
    "Using": [
      "Serilog.Sinks.File"
    ],
    "MinimumLevel": "Debug",
    "WriteTo": [
      {
        "Name": "File",
        "Args": {
          // 75 MiB file size limit
          "fileSizeLimitBytes": 78643200,
          "retainedFileCountLimit": 20,
          "path": "Logs/ELOSync.log",
          "rollingInterval": "Day"
        }
      }
    ],
    "Enrich": [
      "FromLogContext",
      "WithMachineName",
      "WithThreadId"
    ],
    "Properties": {
      "Application": "ELO Sync"
    }
  }
}
```

Plain console output

The log is only written to the console output (stdout) of the ELO Sync process.

```
{
  "Serilog": {
```

```
    "MinimumLevel": "Debug",
    "WriteTo": [
      {
        "Name": "Console"
      }
    ],
    "Enrich": [
      "FromLogContext",
      "WithMachineName",
      "WithThreadId"
    ],
    "Properties": {
      "Application": "ELO Sync"
    }
  }
}
```

Asynchronous console output

The log is only written on the console output (stdout), but asynchronously, preventing logging itself from slowing down the process significantly. The greatest disadvantage is that log messages can be written in the wrong order.

```
{
  "Serilog": {
    "MinimumLevel": "Debug",
    "WriteTo": [
      {
        "Name": "Async",
        "Args": {
          "configure": {
            "Name": "Console"
          }
        }
      }
    ],
    "Enrich": [
      "FromLogContext",
      "WithMachineName",
      "WithThreadId"
    ],
    "Properties": {
      "Application": "ELO Sync"
    }
```

```
  }
}
```

Console output in JSON format

The log is only written to the console output (stdout) in JSON format.

```
{
  "Serilog": {
    "MinimumLevel": "Debug",
    "WriteTo": [
      {
        "Name": "Console",
        "Args": {
          "formatter": "Serilog.Formatting.Json.JsonFormatter"
        }
      }
    ],
    "Enrich": [
      "FromLogContext",
      "WithMachineName",
      "WithThreadId"
    ],
    "Properties": {
      "Application": "ELO Sync"
    }
  }
}
```

Console output in compact JSON format (CLEF)

The log is only written to the console output (stdout) in the JSON-based CLEF format.

```
{
  "Serilog": {
    "MinimumLevel": "Debug",
    "WriteTo": [
      {
        "Name": "Console",
        "Args": {
          "formatter": "Serilog.Formatting.Compact.CompactJsonFormatter, Serilog.Formatting
        }
      }
    ],
    "Enrich": [
```

```
      "FromLogContext",
      "WithMachineName",
      "WithThreadId"
    ],
    "Properties": {
      "Application": "ELO Sync"
    }
  }
}
```

Console output in the CLEF format and log overflow

An example for combining multiple log outputs.

```
{
  "Serilog": {
    "MinimumLevel": "Debug",
    "WriteTo": [
      {
        "Name": "Console",
        "Args": {
          "formatter": "Serilog.Formatting.Compact.CompactJsonFormatter, Serilog.Formatting
        }
      },
      {
        "Name": "File",
        "Args": {
          // 75 MiB file size limit
          "fileSizeLimitBytes": 78643200,
          "retainedFileCountLimit": 20,
          "path": "Logs/ELOSync.log",
          "rollingInterval": "Day"
        }
      }
    ],
    "Enrich": [
      "FromLogContext",
      "WithMachineName",
      "WithThreadId"
    ],
    "Properties": {
      "Application": "ELO Sync"
    }
  }
}
```