ELO Sync

Konfiguration

Inhaltsverzeichnis

Konfiguration des Dienstes 3
Templates für Logging Konfigurationen 18

Konfiguration des Dienstes

Dieser Abschnitt beschreibt die Konfiguration des ELO Sync Dienstes entweder mithilfe der `appsettings.json` und `appsettings.Production.json` in der lokalen Installation oder durch Änderung der im ELO Repository gespeicherten Anwendungseinstellungen.

Es wird empfohlen, während der Installation die mitgelieferte Datei `appsettings.json` in eine neue Datei `appsettings.Production.json` zu kopieren und die Einstellungen entsprechend anzupassen.

Da die Datei `appsettings.Production.json` nie in das verteilte Paket aufgenommen wird, wird sie bei Aktualisierungen nie überschrieben.

Aus technischen Gründen müssen einige Einstellungen in der Datei `appsettings.json` konfiguriert werden und können nicht über die Weboberfläche oder die REST API geändert werden.

Jeder Abschnitt enthält immer ein Beispiel, das den einzelnen Abschnitt innerhalb der vollständigen Datei beschreibt. Wenn Sie mehrere Abschnitte kombinieren möchten, müssen Sie die äußeren Klammern weglassen.

Beispiel:

```
{
  "Key": {            // erste zu kopierende Zeile
    "MySetting": 42
  }                   // letzte zu kopierende Zeile
}
```

Die Abschnitte sind in alphabetischer Reihenfolge aufgeführt, um die Navigation zu erleichtern.

AllowedHosts

Diese Konfigurationseinstellung definiert die Host-Filterung, die vom eingebauten Kestrel-Server verwendet wird.

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

Api

Dieser Abschnitt enthält die Einstellungen für die Verwendung der REST API.

```
{
  "Api": {
    // Gibt an, ob der SingleTenant-Login verwendet werden soll.
    // Wenn für ELO Sync der AppType 'SingleTenant' ausgewählt ist,
```

```
    // dann muss diese Einstellung auf true gesetzt werden. Dadurch
    // findet die Anmeldung dann direkt auf diesem Tenant statt.
    "ForceSingleTenantLogin": false
  }
}
```

AzureAd

Dieser Abschnitt enthält die Einstellungen für die (statische) App-Registrierung in Microsoft Azure. Die Azure-Einstellungen können auch im ELO Repository definiert werden, siehe Konfiguration im ELO Repository gespeichert.

```
{
  "AzureAd": {
    // Der für die Authentifizierung verwendete Microsoft Entra ID-Endpunkt muss nur für
    "Instance": "https://login.microsoftonline.com/",

    // Der Domänenname, unter dem diese Anwendung registriert ist
    "Domain": "mydomain.onmicrosoft.com",

    // Die Anwendungs-(Client-)ID der registrierten Anwendung
    "ClientId": "00000000-0123-0123-0123-c0ffeec0ffee",

    // Die Tenant (Mandanten) ID der registrierten Anwendung
    "TenantId": "c0ffee00-0123-4567-4567-000000000000",

    // Für die Authentifizierung verwendetes Client-Secret
    "ClientSecret": "SecretGeneratedByAzure",

    // Der Callback-Pfad, der für OAuth-Redirect-Ergebnisse verwendet wird
    "CallbackPath": "/signin-oidc-custom",

    // Für die Authentifizierung verwendete(s) Client-Zertifikat(e).
    "ClientCertificates": [
      {
        // Die Eigenschaften, die zum Lesen des Zertifikats verwendet werden, müssen ein
        // "Path" - Der private Schlüssel des Zertifikats wird in einer lokalen Datei ge
        //          Die Eigenschaften „CertificateDiskPath“ und „CertificatePassword“ so
        // "StoreWithThumbprint" - Der private Schlüssel wird in einem Windows-Zertifika
        //          Die Eigenschaften "CertificateStorePath" und "CertificateThumbprint"
        // "StoreWithDistinguishedName" - Der private Schlüssel wird in einem Windows-Ze
        //          Die Eigenschaften "CertificateStorePath" und "CertificateDistinguish
        "CertificateSource": "Path",

        // Pfad zum Zertifikatspeicher
        "CertificateStorePath": "CurrentUser/My",
        // Thumbprint des Zertifikats
        "CertificateThumbprint": "0123456789abcdef",
        // Unterscheidbarer Name des Zertifikates
```

```
        "CertificateDistinguishedName": "MyCertificateName",

        // Passwort zum Entschlüsseln des Zertifikats
        "CertificatePassword": "MySecretPassword",

        // Pfad zur Zertifikatsdatei
        "CertificateDiskPath": "C:\\path\\to\\certificate.pfx"
      }
    ]
  }
}
```

ConnectionStrings

Dieser Abschnitt enthält die Connection Strings, welche von verschiedenen Datenbankanbietern verwendet werden.

Nur der Connection String des konfigurierten Datenbankproviders muss gesetzt werden.

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

Cookies

Dieser Abschnitt enthält die Einstellungen für die von ELO Sync verwendeten Cookies.

```
{
  "Cookies": {
    // Die Lebensdauer des Authentifizierungs-Cookies
    "ExpireTimeSpan": "00:30:00"
  }
}
```

> **Information**
>
> 1. Eine Änderung dieser Einstellung wird erst nach Neustart des Dienstes wirksam.
> 2. Existierende Cookies behalten ihre ursprüngliche Lebensdauer, bis sie ablaufen. Wenn ein altes Cookie eine Lebensdauer von einer Stunde hat und diese Einstellung auf 30 Minuten geändert wird, dann kann das alte Cookie auch nach 45 Minuten noch verwendet werden; danach wir ein 30-minütiges Cookie ersetzt.

CORS

In diesem Abschnitt können die Einstellungen für die CORS-Filterung (Cross-Origin-Resource-Sharing) konfiguriert werden.

```
{
  "Cors": {
    // Die Liste der erlaubten Ursprünge, die auf ELO Sync zugreifen dürfen.
    // Wenn Sie alle Ursprünge zulassen möchten, sollte die Einstellung auf null oder []
    "AllowedOrigins": [
      "https://my-elo-server:9093",
      "https://my-other-origin.com"
    ]
  }
}
```

Database

Dieser Konfigurationsschlüssel legt die Datenbank fest, die von ELO Sync verwendet werden soll.

Nachdem Sie diesen Schlüssel gesetzt haben, müssen Sie auch den entsprechenden ConnectionString setzen.

```
{
  // Der Typ des Datenbankanbieters, der verwendet werden soll.
  // Muss einer der folgenden Werte sein:
  // "Postgres" - PostgreSQL Datenbank (https://www.postgresql.org/)
  //              Version: 14.x oder neuer, empfohlen 16.x
  // "MsSql"    - Microsoft SQL Server (https://www.microsoft.com/en-us/sql-server/sql-s
  //              Version: 2012 oder neuer, 2019 empfohlen
  "Database": "Postgres"
}
```

DownstreamApis

Dieser Schlüssel enthält die Konfigurationseinstellungen für alle Drittanbieter-APIs, die von ELO Sync verwendet werden.

Jede API muss eine Eigenschaft `Api` haben, die zur Identifizierung der API verwendet wird, aber alle zusätzlichen Eigenschaften sind spezifisch für die jeweilige konfigurierte API.

DownstreamApis:Microsoft Graph

Dieser Abschnitt enthält die spezifischen Einstellungen für den Zugriff auf die Microsoft Graph API.

```
{
  "DownstreamApis": [
```

```
    {
      // Fester Bezeichner für die Microsoft Graph API
      "Api": "Microsoft Graph",
      // Basis-URL für den Zugriff auf Microsoft Graph, muss nur für nationale Cloud-Ber
      // HINWEIS: Derzeit wird nur der stabile Endpunkt /v1.0 unterstützt, der Endpunkt
      "BaseUrl": "https://graph.microsoft.com/v1.0",
      // Die Berechtigungen, die bei der Verwendung dieser API angefordert werden sollte
      "Scopes": "user.read Sites.Read.All Sites.ReadWrite.All",
      // Der bei der Authentifizierung mit dieser API verwendete Scope
      "DefaultGraphScope": "https://graph.microsoft.com/.default"
    }
  ]
}
```

HttpClient

Dieser Abschnitt enthält die allgemeinen Einstellungen für HTTP(S)-Verbindungen, die von ELO Sync aufgebaut werden.

Diese Einstellungen gelten nur, sofern es keine spezifischeren Einstellungen an anderer Stelle gibt. In diesem Fall haben die spezifischeren Einstellungen Vorrang und können nicht über diese allgemeine Einstellung geändert werden. Dies gilt auch, wenn die spezifischere Einstellung nicht in der Datei vorhanden ist, da dann der Standardwert für die spezifischere Einstellung verwendet wird.

```
{
  "HttpClient": {
    // Der Timeout in Sekunden für alle HTTP-Anfragen
    "Timeout": 100,
  }
}
```

InstanceName

Der optionale Instance Name kann für die horizontale Skalierung des ELO Sync-Dienstes verwendet werden. Wenn Sie diesen Namen festlegen, können zusätzliche Konfigurationsdateien aus dem ELO Repository geladen werden. Siehe Konfiguration im ELO Repository gespeichert für weitere Informationen.

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

> **Achtung**
>
> Es gibt keine Unterstützung für mehrere Instanzen von ELO Sync, die dieselbe Datenbank verwenden.

Der Instanzname beeinflusst nur die Anwendungskonfiguration, sodass für jede Instanz eine eigene Datenbank verwendet werden kann.

Kestrel

Dieser Konfigurationsabschnitt enthält die Einstellungen für den eingebauten Kestrel-Webserver.

Information

Damit ELO Sync auf Microsoft 365 zugreifen kann, ist es erforderlich, dass die Benutzerendpunkte das HTTPS-Protokoll verwenden.

Ein HTTP-Endpunkt kann nur verwendet werden, wenn dem ELO Sync-Server ein Reverse-Proxy vorgeschaltet ist und der Proxy für die Verwendung von HTTPS konfiguriert ist.

Siehe auch:

- 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

In diesem Abschnitt wird die von ELO Sync verwendete Komponente *Microsoft.Extension.Logging* konfiguriert.

Normalerweise leitet diese Komponente das gesamte Logging zu Serilog um, wird aber auch verwendet, um das Logging-Level einzelner Komponenten zu ändern.

Siehe Configure logging without code für weitere Informationen.

```
{
  "Logging": {
    "LogLevel": {
      // Information und darüber für alle unkonfigurierten Komponenten verwenden
      "Default": "Information",
      // Error und darüber für alle Microsoft Komponenten verwenden
```

```
      "Microsoft": "Error",
      // Debug und darüber für alle Elo Komponenten verwenden
      "Elo": "Debug"
    }
  }
}
```

OAuth

Dieser Abschnitt enthält die Einstellungen für die Verwendung von OAuth mit ELO Sync.

```
{
  "OAuth": {
    // Die ID der OAuth-Konfiguration, die mit dem ELOauth-Plugin verwendet werden soll.
    "ConfigId": "elo_sync_oauth"
  }
}
```

Der gewählte Name für das OAuth-Konfigurationsprofil muss auch in der Konfiguration des ELOauth-Plug-ins für den Indexserver vorhanden sein.

Die OAuth ConfigId kann auch im ELO Repository definiert werden, siehe Konfiguration im ELO Repository gespeichert.

Beachten Sie die Dokumentation für das ELOauth-Plug-in, um ein Authentifizierungsprofil zu konfigurieren.

OpenTelemetry

ELO Sync verwendet die OpenTelemetry Infrastruktur für Tracing und Überwachung des Zustands des Dienstes.

Jeder Bereich kann separat konfiguriert werden und ist im default deaktiviert.

OpenTelemetry:Logging

In diesem Abschnitt wird der Logging-Exporter für OpenTelemetry unter Verwendung des OTLP-Protokolls konfiguriert.

```
{
  "OpenTelemetry": {
    "Logging": {
      // Ob der OTLP logging exporter aktiviert werden soll
      "Enabled": true,

      // Der Endpunkt an den die Logs gesendet werden sollen
      "Endpoint": "http://localhost:4317/v1/logs",

      // Das Protokoll, das zum Senden der Logs an den Endpunkt verwendet wird, muss ein
```

```
      // "HttpProtobuf" - HTTP Protokoll welches das Protobuf Format verwendet
      // "Grpc" - gRPC Protocol (Default)
      "Protocol": "HttpProtobuf",

      // Weitere Header, die mit jeder Anfrage versendet werden sollten, sie werden typi
      "Headers": "Authorization: Bearer MySecret",

      // Die Exporter Optionen
      "BatchExportProcessorOptions": {
        // Siehe unten für weitere Details
      }
    }
  }
}
```

OpenTelemetry:Tracing

In diesem Abschnitt wird der Tracing-Exporter für OpenTelemetry unter Verwendung des OTLP-Protokolls konfiguriert.

```
{
  "OpenTelemetry": {
    "Tracing": {
      // Ob der OTLP Tracing exporter aktiviert werden soll
      "Enabled": true,

      // Der Endpunkt an den die Traces gesendet werden sollen
      "Endpoint": "http://localhost:4317/v1/traces",

      // Das Protokoll, das zum Senden der Traces an den Endpunkt verwendet wird, muss e
      // "HttpProtobuf" - HTTP Protokoll welches das Protobuf Format verwendet
      // "Grpc" - gRPC Protocol (Default)
      "Protocol": "HttpProtobuf",

      // Weitere Header, die mit jeder Anfrage versendet werden sollten, sie werden typi
      "Headers": "Authorization: Bearer MySecret",

      // Die Exporter Optionen
      "BatchExportProcessorOptions": {
        // Siehe unten für weitere Details
      }
    }
  }
}
```

OpenTelemetry:Metrics

In diesem Abschnitt wird der Metrik-Exporter für OpenTelemetry unter Verwendung des OTLP-Protokolls konfiguriert.

```
{
  "OpenTelemetry": {
    "Metrics": {
      // Ob der OTLP Metriken exporter verwendet werden soll
      "Enabled": true,

      // Der Endpunkt an den die Metriken gesendet werden sollen
      "Endpoint": "http://localhost:4317/v1/metrics",

      // Das Protokoll, das zum Senden der Metriken an den Endpunkt verwendet wird, muss
      // "HttpProtobuf" - HTTP Protokoll welches das Protobuf Format verwendet
      // "Grpc" - gRPC Protocol (Default)
      "Protocol": "HttpProtobuf",
      // Weitere Header, die mit jeder Anfrage versendet werden sollten, sie werden typi
      "Headers": "Authorization: Bearer MySecret",

      // Die Exporter Optionen
      "BatchExportProcessorOptions": {
        // Siehe unten für weitere Details
      }
    }
  }
}
```

OpenTelemetry:*:BatchExportProcessorOptions

Dieser Abschnitt wird von allen OpenTelemetry-Exportern gemeinsam genutzt und konfiguriert den Batch-Export-Processor.

```
{
  "OpenTelemetry": {
    "*": {
      "BatchExportProcessorOptions": {
        // Verzögerung zwischen zwei geplanten Exporten in Millisekunden
        "ScheduledDelayMilliseconds": 5000,

        // Die Anzahl der Items pro Export Batch
        "MaxExportBatchSize": 512,

        // Die maximale ANzahl der Items in der Warteschlange bevor weitere Items verwor
        "MaxQueueSize": 2048,

        // Der Timeout in Millisekunden, nach dem ein Export abgebrochen wird
```

```
        "ExporterTimeoutMilliseconds": 30000
      }
    }
  }
}
```

OpenTelemetry:Zipkin

In diesem Abschnitt wird der Zipkin-Exporter konfiguriert.

```
{
  "OpenTelemetry": {
    "Zipkin": {
      // Wenn der Zipkin-Exporter verwendet werden soll
      "Enabled": true,

      // Der Endpunkt an den die Traces gesendet werden sollen
      "Endpoint": "http://localhost:9411/api/v2/spans"
    }
  }
}
```

PrometheusServer

Dieser Abschnitt konfiguriert den Server für den Prometheus-Metrik-Anbieter.

```
{
  "PrometheusServer": {
    // Die Endpunkt-URL, unter der die Metriken veröffentlicht werden sollen
    // https wird aktuell nicht unterstützt
    "Endpoint": "http://localhost:9090/metrics",

    // Legt fest, ob der Zugriff über einen beliebigen Hostnamen erlaubt ist
    "AllowAllHosts": true
  }
}
```

PublicUrl

Dieser Eintrag enthält die öffentliche URL, die für den Zugriff auf ELO Sync verwendet werden kann, wenn die Zugriffs-URL von der in der Kestrel-Sektion eingestellten internen URL abweicht, z.B. durch Verwendung eines Proxys.

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

Repositories

In diesem Konfigurationsabschnitt werden alle ELO Repositorys aufgeführt, auf die bei Verwendung von ELO Sync zugegriffen werden kann.

Ein Benutzer kann eines dieser Repositorys auswählen, wenn er sich über die Weboberfläche anmeldet.

```
{
  "Repositories": [
    {
      // Erforderlich; der Anzeigename für den Server in der Benutzeroberfläche
      "name": "My ELO Server",

      // Erforderlich; Ein eindeutiger Schlüssel für diesen Server. Dieser Schlüssel
      // wird auch beim Aufruf der REST-API zur Identifizierung des Repositorys verwende
      "key": "MyELOServer",

      // Erforderlich; der ELOix-Endpunkt, der für den Zugriff auf das Repository verwen
      "url": "https://my-elo-server:9093/ix-Repository1/ix",

      // Optional; die ELO-Web-Client-URL, die für die Vorschau oder die Verknüpfung mit
      // Wenn nicht festgelegt, wird die Web-Client-URL aus der Eigenschaft "url" gebild
      "webclienturl": "https://my-elo-server:9093/ix-Repository1/plugin/de.elo.ix.plugin

      // Optional; die von diesem Repository verwendete OAuth-Callback-URL
      "oauthcallbackurl": "https://my-elo-server:9093/ix-Repository1",

      // der von diesem Repository verwendete Anmeldemodus; muss einer der folgenden Wer
      // "ELOauth" - Für die Authentifizierung wird das ELOauth-Plugin verwendet, das au
      // "auth2"   - Das Auth2-Plugin wird für die Authentifizierung verwendet
      "loginmode": "auth2",

      // Cookie-Token-Exchange-URL, die zur Authentifizierung einer ELO-Session unter Ve
      // Diese Eigenschaft wird ignoriert, wenn „loginmode“ gesetzt ist
      "cookietokenexchangeurl": "https://my-elo-server:9093/ix-Repository1/plugin/de.elo

      // Die Anmelde-URL, die verwendet wird, wenn ein Benutzer auf eine ELO-Web-Benutze
      // ¶ClientUrl¶ wird durch die Ziel-URL ersetzt
      // Diese Eigenschaft wird ignoriert, wenn „loginmode“ gesetzt ist
      "loginurl": "https://my-elo-server:9093/ix-Repository1/plugin/de.elo.ix.plugin.aut

      // Optional; Der Timeout in Sekunden für den Zugriff auf das Repository
      // Der Standardwert sind 10 Sekunden, da ELO Sync oft auf demselben Gerät ausgeführ
      "timeout": 10
    }
  ]
}
```

Der Einfachheit halber sind im Folgenden einige Vorlagenkonfigurationen für Repositorys aufgeführt.

```
{
  "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; mit öffentlichem Zugang zum Web Client
    {
      "name": "My Repository",
      "key": "MyRepository",
      "url": "https://my-elo-server:9093/ix-Repository1/ix",
      "loginmode": "auth2",
      // Public URLs für Web & Login; selbe Domain muss verwendet werden
      "webclienturl": "https://elo.company.com/web/",
      "loginurl": "https://elo.company.com/authorize?redirect_uri=¶ClientUrl¶&response_t
    },
    // < ELO 23.4; mit öffentlichem Zugang zum Web Client
    {
      "name": "My Repository",
      "key": "MyRepository",
      "url": "https://my-elo-server:9093/ix-Repository1/ix",
      "loginmode": "ELOauth",
      // Public URLs für Web & Login; selbe Domain muss verwendet werden
      "webclienturl": "https://elo.company.com/web/",
      "loginurl": "https://elo.company.com/login?clientUrl=¶ClientUrl¶&configId=¶ConfigI
    }
  ]
}
```

Login mode

Die Eigenschaft `loginmode` steuert die zusätzlichen Eigenschaften, die für die Authentifizierung mit einem ELO Repository erforderlich sind.

Sie kann je nach ELO Indexserverversion und -Konfiguration entweder auf `ELOauth` oder `auth2` gesetzt werden.

`ELOauth` kennzeichnet die Verwendung des ELOauth-Plug-ins zur Authentifizierung, das von ELO Indexserverversion vor 23.4 unterstützt wird.

`auth2` kennzeichnet die Verwendung des Auth2-Plug-ins für die Authentifizierung, das von ELO Indexserverversion 23.4 und später unterstützt wird.

Cookie Token exchange URL

Diese URL wird zur Authentifizierung einer ELO Sitzung mithilfe eines Zugriffstokens von Azure/ Entra ID verwendet.

Die angegebene URL kann einen Platzhalter `¶ConfigId¶` enthalten, der durch die Einstellung `OAuth:ConfigId` ersetzt wird.

Alternativ dazu muss der Parameter `configId` manuell in der URL gesetzt werden. ELO Sync prüft nicht, ob die URL korrekt ist.

Wenn die Eigenschaft manuell angegeben wird, wird die Einstellung `OAuth:ConfigId` ignoriert und der Parameter `configId` muss mit der erforderlichen Konfiguration für ELO Sync übereinstimmen.

Die angegebene URL muss mit dem Abfrageparameter `jwt_token` enden, da das Zugriffstoken einfach an das Ende angehängt wird.

> **Information**
>
> Dies könnte sich in einer zukünftigen Version ändern, je nach anderen Anforderungen.

Wird die Eigenschaft nicht angegeben, wird eine Standard-URL unter Verwendung der Basis-URL des ursprünglichen Authentifizierungs-Plug-ins (`de.elo.ix.plugin.auth`) erstellt.

Wenn das Auth2-Plug-in für die Authentifizierung verwendet wird (standardmäßig aktiviert auf IX 23.4+), muss diese Eigenschaft explizit angegeben werden.

Beispiel:

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

Der `configId`-Parameter muss mit der Konfiguration übereinstimmen, die für die Login-Konfiguration im Plug-in verwendet wurde.

Serilog

Die aktuell verwendete Logging-Lösung von ELO Sync ist Serilog.

In diesem Abschnitt werden die Senken und Filter für das Logging konfiguriert. Für weitere Details siehe Serilog.Settings.Configuration

Im Folgenden finden Sie ein Beispiel, das ein grundlegendes Login eine fortlaufende Logdatei schreibt.

```
{
  "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"
    }
  }
}
```

Alternative Beispiele für die Konfiguration des Loggings finden Sie hier.

ServiceUser

Die Einstellungen für den ELO Service User, der von ELO Sync verwendet werden soll.

Dieser Benutzer muss bereits in der ELOam Instanz existieren und über administrative Rechte verfügen.

```
{
  "ServiceUser": {
    // Benutzer in der ELOam-Instanz, der über administrative Rechte verfügt
    "UserName": "ELO Service",
    // Passwort für den Benutzer
    "Password": "MySecretPassword"
  }
}
```

Regelmäßige Abfrage der Konfiguration aus dem Repository

```
{
  "EnableRepositoryConfigPolling":
  //
  // Prüft in regelmäßigen Zeitabständen, ob sich die Konfiguration für ELO Sync in Repo
  // Hat sich die Konfiguration geändert, dann:
  //   - die Konfiguration wird aus dem Repository geladen
  //   - /api/status gibt HTTP-Statuscode 500 mit dem HTTP-Header X-RestartNeccessary
  //     zurück. In diesem Fall muss der ELO Synd Dienst neu gestartet werden
  //
  "true"
  //
  // oder
  //
  // Lädt einmalig die Konfiguration für ELO Sync aus dem Repository beim Start von ELO
  // Spätere Änderungen können nur übernommen werden, wenn ELO Sync neu gestartet wird
  //
  "false"
}
```

Speichern der Konfiguration im ELO Repository

Die Anwendungskonfiguration kann teilweise in einem ELO Repository gespeichert werden.

Aus diesem Grund muss die lokale `appsettings.json` (oder `appsettings.Production.json`) ein Repository und die Einstellung Service User besitzen.

ELO Sync verwendet dann diese lokale Konfiguration und verbindet sich mit dem Repository und fügt die Arbeitsversion des folgenden json-Dokuments als Konfigurationsdatei ein:

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

  Wenn dieses Dokument fehlt oder keine gültige Konfiguration hat, wird sein Inhalt ignoriert und eine Warnung protokolliert.

  Wenn mehrere Dokumente mit diesem Namen existieren, wird nur das erste (von ELOix ermittelte) geladen.

  Wenn der `InstanceName` in der lokalen Konfiguration festgelegt ist, wird das folgende Konfigurationsdokument zusätzlich zu den vorherigen Konfigurationselementen geladen.

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

  Der obige Platzhalter `<InstanceName>` wird durch den Wert der Einstellung `InstanceName` ersetzt.

Templates für Logging Konfigurationen

Auf dieser Seite sind verschiedene Vorlagen für die Logging-Konfiguration aufgeführt. Viele dieser Vorlagen können einfach kopiert werden, um eine bestehende Logging-Konfiguration zu ersetzen.

Überlaufende Logdatei

Das Protokoll wird in eine Datei geschrieben, die für jeden Tag oder nach 75 Megabyte überläuft.

```
{
  "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"
    }
  }
}
```

Einfache Konsolenausgabe

Log wird nur in die Konsolenausgabe (stdout) des ELO Sync-Prozesses geschrieben.

```
{
  "Serilog": {
    "MinimumLevel": "Debug",
    "WriteTo": [
```

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

Asynchrone Konsolenausgabe

Log wird nur auf die Konsolenausgabe (stdout) geschrieben, aber asynchron, sodass die Protokollierung selbst den Prozess nicht wesentlich verlangsamt. Der größte Nachteil ist, dass Log-Meldungen in der falschen Reihenfolge geschrieben werden können.

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

Konsolenausgabe im JSON-Format

Log wird nur auf die Konsolenausgabe (stdout) in einem JSON-Format geschrieben.

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

Konsolenausgabe im kompakten JSON-Format (CLEF)

Log wird nur in die Konsolenausgabe (stdout) im JSON-basierten CLEF-Format geschrieben.

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

Konsolenausgabe im CLEF-Format und Überlaufende Logdatei

Ein Beispiel für die Kombination mehrerer Log-Ausgaben.

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