COMARCH e-COMMERCE

Comarch e-Sklep Sync

wersja 2019.3

Data produkcji wersji: **29 maja 2019**

Comarch e-Sklep

Spis treści

Comarch e-Sklep Sync – wstęp

Comarch e-Sklep Sync 2019.3 to narzędzie umożliwiające pobieranie danych na żądanie z baz systemów Comarch ERP Optima, Comarch ERP XL, Comarch ERP Altum. Aplikacja działa w postaci usługi, która potrafi komunikować się z Comarch e-Sklep od wersji 2019.3.

Obecnie usługa umożliwia pobranie danych na temat: faktur, płatności czy zamówień z systemów Comarch ERP. Usługa posiada otwarty interfejs umożliwiający pobieranie dowolnych danych z użyciem procedur SQL a także rozszerzeń aplikacji (pliki dll).

1 Wymagania instalacji

Comarche-SklepSync2019	
Wymaganie	**Opis**
Komputer z systemem Windows	Zalecane: Windows Server 2012 i wyższy Możliwe do uruchomienia: Windows 7, Windows 10 Zalecamy instalację na systemach serwerowych Wymagany .NET Framework 4.0
Dostęp do internetu	Nieograniczony i stabilny dostęp do / z internetu po porcie 443. Aplikacja wymaga zainstalowanego certyfikatu SSL w sklepie internetowym
Comarch e-Sklep	Wymagana wersja Comarch e-Sklep B2B 2019.3
Klucz wirtualny	Comarch e-Sklep Sync wymaga licencji w kluczu wirtualnym. Licencję można uzyskać w dziale logistyki Comarch ERP pisząc na adres logistyka.erp@comarch.pl lub u swojego Partnera Biznesowego

2 Proces instalacji krok po kroku

1. Pobrać aplikację z Indywidualnych Stron Klienta lub Partnera
   Strony \ e-commerce \ Comarch e-Sklep \ Dokumentacja
2. W razie potrzeby odblokować plik ZIP (prawy klawiszy myszy \ właściwości pliku \ odblokuj) oraz zdjąć atrybut „tylko do odczytu”
3. Rozpakować w dowolnym miejscu, zgodnym z założeniami administratora serwera.
4. Wykonać konfigurację przez plik **configWorkers.json**, podając odpowiednie parametry.

Parametry podstawowe pliku configWorkers.json

PlikconfigWorkers.json	
Parametr	**Opis**
Id	Identyfikator, dowolny ciąg znaków i liczb, w przykładzie poniżej jest ESHOP_1, zalecamy zmienić na swój indywidualny ciąg znaków
Url	Adres sklepu internetowego, z którym Comarch e-Sklep Sync będzie współpracował. Uwaga, adres musi kończyć się ***/sync***
AppId	Dowolny ciąg, max. 64 znaków, sekretny identyfikator aplikacji, zalecamy losowy ciąg znaków, co najmniej 16.
AppKey	Dowolny ciąg, max. 64 znaków, sekretny klucz aplikacji, zalecamy losowy ciąg znaków, co najmniej 16.
Server	Serwer klucza wirtualnego, jeśli jest instancja należy wpisać ****
Number	Numer klucza wirtualnego

Przykładowy plik:

```
{
   "Id": "ESHOP_1",
   "Url": "http://adres-sklep.pl/sync",
   "AppId": "mkdsfs84hfswkldl90245235",
   "AppKey": "djfsdkj9902034jsxzsvf234567856rtDD##@",
   "LogLevel": "Info",
   "License": {
      "Server": "luigi-vm",
      "Number": "5000000046"
   },
```

Etap 2 – parametry połączenia do bazy danych

*Data Source=Serwer SQL; - **w przypadku instancji SQL zapis powinien zawierać \\ zamiast ***
Initial Catalog=Baza Systemu ERP;
user id=UzytkownikSQL;
password=HasloUzytkownikaSQL;
Asynchronous Processing=true; Application Name=Comarch_eSklep_Sync"

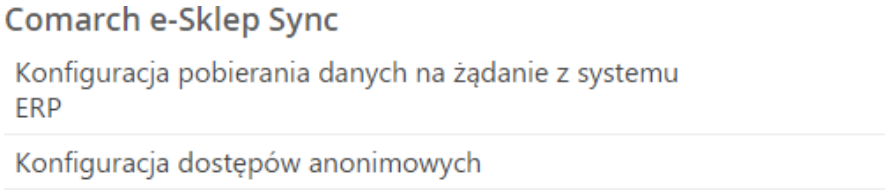
Uwaga! W przykładach korzystamy z użytkownika SQL o nazwie: erpDataUser oraz loginu: erpDataLogin.
Zmiany można dokonać w pliku SQL o nazwie SQLCommon.sql

Przykładowy zapis:

```
"Config": {
  "CS": "Data Source=server;
                    Initial Catalog=CDN_BazaDemo;
                    user id=test1;password=test1;
                    Asynchronous Processing=true;
                    Application Name=Comarch_eSklep_Sync"
}
```

Zalecamy przygotować nowego użytkownika z ograniczonymi prawami do bazy danych tak jak to jest w przykładzie poniżej.

5. Ustawić takie same wartości **AppId, AppKey** w nowym panelu administracyjnym w sekcji: **Ustawienia \ Ogólne \ Comarch e-Sklep Sync \ Konfiguracja pobierania danych na żądanie z systemu ERP**



6. Nadać prawa zapisu na katalog instalacyjny dla użytkownika USŁUGA SIECIOWA (NETWORK SERVICE). Ten użytkownik jest domyślnym użytkownikiem na którego usługa się instaluje.
7. Zmodyfikować odpowiedni plik instalacyjny w zależności od posiadanego systemu Comarch ERP: **installOptima.bat / installXL.bat, installAltum.bat** wpisując dane dotyczące serwera SQL i nazwę bazy danych systemu Comarch ERP
   Aplikacja instaluje się w domyślnym katalogu: C:\Program Files (x86)\Comarch e-Sklep Sync, jest możliwość jego zmiany przez modyfikację pliku jak wyżej
   Uwaga! Przed uruchomieniem pliku w punkcie 8, należy zainstalować narzędzie SQLCMD
   https://docs.microsoft.com/en-us/sql/tools/sqlcmd-utility?view=sql-server-2017
8. Zainstalować usługę z użyciem odpowiedniego pliku z punktu 6, usługa zainstaluje się pod nazwą Comarch e-Sklep Sync
   Uwaga! Plik należy uruchomić z prawami administratora
9. Uruchomić usługę w systemie Windows
   (Panel sterowania\Wszystkie elementy Panelu sterowania\Narzędzia administracyjne\Usługi)
10. W razie niepowodzenia instalacji lub uruchomienia zweryfikować logi:
    - dostępne w katalogu instalacyjnym usługi w pod katalogu **Logs**
    - w systemie Windows (Narzędzia Administracyjne \ Dziennik zdarzeń)

Do najbardziej popularnych komunikatów należą:

- 403 Forbidden (Disabled) – usługa jest wyłączona.
- 403 Forbidden (Authorization) - w większości przypadków spowodowane jest niepoprawnym Appid albo Appkey. Może tez wystąpić w przypadku starszej wersji Comarch e-Sklep Sync i nowego Comarch e-Sklep.
- 403 Forbidden (TS) - niezgodny timestamp. W środowisku instalacyjnym został ustawiony niepoprawny czas.
- 403 Forbidden (Args) - błędny request. Żądanie nie było wysłane przez Comarch e-Sklep Sync albo starszej wersji oprogramowania.
- 403 Forbidden (DEMO) – Comarch e-Sklep jest w wersji Demo albo sklep nie ma licencji.

Komunikaty związane z licencjami:

- Get License: True/False – nie została pobrana licencja.
- Renew License: True/False – nie udało się odnowić licencji, która wcześniej została pobrana (możliwe, że program pobrał licencję, albo menedżer licencji został zresetowany).
- Release License – w przypadku zatrzymania Comarch e-Sklep Sync, zostaje oddana licencja do menedżera licencji
- e-Shop not synchronized – nie została przeprowadzona synchronizacja do Comarch e-Sklep
- No e-Shop Sync license – brak licencji w menedżerze licencji dla Comarch e-Sklep Sync

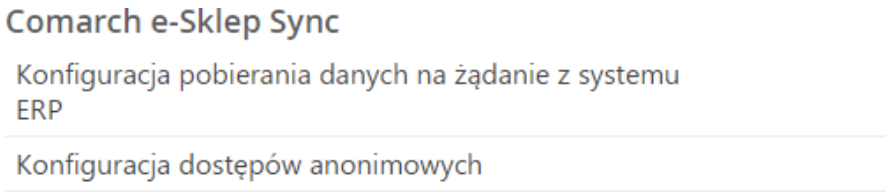
Inne komunikaty:

- Invalid Url - w configWorkers.json został podany niepoprawny URL Comarch e-Sklep.
- Awaiting message – Comarch e-Sklep Sync czeka na żądanie z Comarch e-Sklep.
- Nothing to do - worker nie otrzymał żądania z Comarch e-Sklep i zaczyna czekać od nowa (jeżeli brak tego komunikaty należy sprawdzić proxy oraz dostęp do sieci internetowej.

3 Dostęp anonimowy

Od wersji Comarch e-Sklep Sync 2017.6 aplikacja umożliwia realizację dostępu do danych dla użytkownika niezalogowanego. Inaczej mówiąc możemy pobrać dane z systemów Comarch ERP dla użytkowników, którzy anonimowo przeglądają naszą stronę internetową. W celu konfiguracji usługi do dostępu anonimowego należy na poziomie nowego panelu administracyjnego jawnie wskazać, który z workerów może być wykonywany bez potrzeby autoryzacji użytkownika.

Nazwę workera podajemy w sekcji: **Ustawienia \ Ogólne \ Comarch e-Sklep Sync \ Konfiguracja dostępów anonimowych**



Przykład:
Jeśli worker w pliku **configworkers.json** nazywa się erpData należy jawnie podać jego nazwę.
Takich workerów można dodać dowolną ilość.

Uwaga!
Przy wołaniu metody w silniku Liquid sync/exec, należy jawnie podać nazwę workera
Ważne są duże i małe litery w nazwie workera

4 Podstawowy przykład użycia

1. Podstawowy przykład użycia jest gotowy w szablonie SZAFIR, wystarczy zainstalować ten szablon
2. W Profilu Klienta są dostępne Zamówienia Klienta oraz Faktury Klienta pobierane z użyciem Comarch e-Sklep Sync

5 Zaawansowane możliwości

Silnik graficzny Liquid ma akcje sync/exec . Ta akcja odpowiada za komunikację interfejsu z Comarch e-Sklep Sync. Akcja przyjmuje parametry:

Parametr	Opis
worker	Nazwa zadania, które ma uruchomić Comarch e-Sklep Sync Domyślnie ‘erpData’ – wywołuje zapytania na SQL, ‘erpDataOrder’ – odpowiada za obsługę stanów magazynowych. Jeśli jest używany własny worker trzeba tutaj podać jego nazwę. Dodatkowe workery konfiguruje się w pliku **configWorkers.json**

message	Treść polecenia do wykonana przez worker.

Po wykonaniu akcji zwracana jest odpowiedź. Odpowiedź zawiera obiekt lub kolekcję obiektów, które należy wyświetlić na interfejsie. Kolekcja obiektów zwraca jest przykładowo, gdy procedura SQL zwraca kilka recordsetów.

Przykład wywołania:

```
function getData() {
    $.post(null, { __action: 'sync/exec', __CSRF: __CSRF,
            worker: 'erpData',
            message: JSON.stringify(
                {
                    command: 'CDN.eShop_GetOrders',
                    parameters: { DateFrom: '2017-01-01', DateTo: '2017-03-31', LanguageId: __lngId, PageNo: '1'}
                }) }, function (d) {
        if (!d.action.Result) {
        /* wystąpił błąd */
            console.log(JSON.stringify(d))
            return;
        }

        var r = d.action.Object[0];
        /* wynik */
        console.log(JSON.stringify(d))

        /* wynik kolekcja */
        var r = d.action.Object[0];
        console.log(JSON.stringify(r))

        var r2 = d.action.Object[1];
        console.log(JSON.stringify(r2))

        var r3 = d.action.Object[2];
        console.log(JSON.stringify(r3))
    });
}
```

Kod wywołania:

```
function getData() {
            $.post(null, {_action: 'sync/exec',_CSRF:_CSRF,
                        worker: 'erpData',
                        message: JSON.stringify(
                                {
                                     command: 'CDN.eShop_GetOrders',
                                      parameters: { DateFrom: '2017-01-01', DateTo:
        '2017-03-31', LanguageId:_lngId, PageNo: '1'}
                                }) }, function (d) {
                if (!d.action.Result) {
                /* wystąpił błąd */
                        console.log(JSON.stringify(d))
                        return;
                }
                var r = d.action.Object[0];
                /* wynik */
```

```
                console.log(JSON.stringify(d))

                /* wynik kolekcja */
                var r = d.action.Object[0];
                console.log(JSON.stringify®)

                var r2 = d.action.Object[1];
                console.log(JSON.stringify(r2))

                var r3 = d.action.Object[2];
                console.log(JSON.stringify(r3))
            });
}
```

6 Dla programistów

6.1 Worker

Obsługa requestów, które aplikacja pobiera z Comarch e-Sklep odbywa się przy pomocy tzw. Workerów. Są to klasy dziedziczące po klasie **Comarch.eShop.Isync.Worker**

Klasa:

```
public abstract class Worker

   {
        protected Dictionary<string, string> _config;

        public virtual void Initialize(Dictionary<string, string> d)
        {
            _config = d;
        }

        public abstract Task<Msg> Execute(IPool pool, Msg msgOuter);
   }
```

Przykładowy kod:

```
        public abstract class Worker
          {
            protected Dictionary<string, string> _config;

            public virtual void Initialize(Dictionary<string, string> d)
            {
              _config = d;
            }

            public abstract Task<Msg> Execute(Ipool pool, Msg msgOuter);
          }
```

oraz implementujące metodę **Execute**

```
namespace WorkerTime
{
    using System;
    using System.Threading.Tasks;
    using Comarch.eShop.ISync;

    public class Time : Worker
    {
        private bool _utc;

        public override void Initialize(System.Collections.Generic.Dictionary<string, string> d)
        {
            _utc = d["utc"] == "1";
        }

        public override Task<Msg> Execute(/*IPool pool,*/ Msg msgOuter)
        {
            return Task.Factory.StartNew((o) =>
            {
                var m = o as Msg;
                m.Response = string.Concat("\"", (_utc ? DateTime.UtcNow : DateTime.Now).ToString("o"), "\"");
                return m;
            }, msgOuter);
        }
    }
}
```

Przykładowy kod:

```
namespace WorkerTime
{
   using System;
   using System.Threading.Tasks;
   using Comarch.eShop.Isync;

   public class Time : Worker
   {
      private bool _utc;

      public override void Initialize(System.Collections.Generic.Dictionary<string, string> d)
      {
         _utc = d["utc"] == "1";
      }

      public override Task<Msg> Execute(/*Ipool pool,*/ Msg msgOuter)
      {
         return Task.Factory.StartNew((o) =>
         {
            var m = o as Msg;
            m.Response = string.Concat("\"", (_utc ? DateTime.UtcNow : DateTime.Now).ToString("o"), "\"");
            return m;
         }, msgOuter);
      }
   }
         }
```

Metoda **Execute** jako drugi parametr przyjmuje obiekt klasy Msg, który posiada property Message oraz Response. W Message jest json, który zawiera dane przesłane requestem z Comarch e-Sklep, których po deserializacji można użyć jako parametrów dla pisanego workera.
Dane wynikowe, które chcemy zwrócić do Comarch e-Sklep należy również serializować do json'a oraz przypisać je do property Response klasy Msg.

6.2 Konfiguracja

Tak napisaną klasę należy następnie dodać do pliku **configWorkers.json**, który zawiera wszystkie Workery zarejestrowane w aplikacji.
Wpis dodający workera wygląda następująco:

```
{
    "Key": "timeLocal",
    "Assembly": "WorkerTime.dll",
    "Type": "WorkerTime.Time",
    "Default": false,
    "Config": {
        "utc": "0"
    }
}
```

```
{
  "Key": "timeLocal",
  "Assembly": "WorkerTime.dll",
  "Type": "WorkerTime.Time",
  "Default": false,
  „Config": {
    „utc": „0"
  }
}
```

gdzie,

PlikconfigWorkers.json	
Parametr	**Opis**
Key	To unikatowa nazwa workera przesyłana z requestem dzięki, której aplikacja wie któryz zarejestrowanych workerów powinien obsłużyć request,
Assembly	Nazwa assembly, gdzie znajduje się worker, którego chcemy dodać
Type	Nazwa klasy będącej workerem
Default	Określa czy jest to domyślny worker. Domyślny worker obsługuje requesty, które nie zawierały nazwy workera

W sekcji **config** należy dodać listę parametrów wymaganych do inicjalizacji workera.

```
public override void Initialize(System.Collections.Generic.Dictionary<string, string> d)
        {
            _utc = d["utc"] == "1";
        }
```

public override void Initialize(System.Collections.Generic.Dictionary<string, string> d)
{
_utc = d[„utc”] == „1”;
}

7 Nowość

W 2019.3 aktualizację Comarch e-Sklep Sync będzie można przeprowadzić w terminie dogodnym dla użytkownika (nie będzie wymagana natychmiastowa aktualizacja).