IspRestApi API v1.1

Dział Pomocy Technicznej G Data
ul. Warcisława IV 2, 78-400 Szczecinek

tel.: +48 94 3729 650
faks: +48 94 3729 659
e-mail: pomoc@gdata.pl

Spis treści

1. Wstęp
Serwis stworzony jest w technologii REST. Parametry przekazywane muszą być w postaci JSON. Parametry zwracane również są w postaci JSON.

Preferowany scenariusz generacji kluczy:
Utworzenie jednego klienta (dla firmy) i na tego klienta generowanie wszystkich licencji i porządkowanie po swojej stronie.

2. Metoda: Login

2.1. Dostęp

https://api.gdata.pl/isp/IspRestApi

• Do weryfikacji użytkownika potrzebny jest podpis, składający się następujących danych:
• Adres IP, z którego robimy zapytanie
• Znacznik czasowy - czas w sekundach od dnia 1.01.1970 wg czasu UTC
• Nazwa api – nazwa api, do którego chcemy się zalogować
• Nazwa użytkownika
• Hasło

W celu otrzymania danych dostępu, czyli : nazwy użytkownika, hasła, kontraktu, kluczy API należy skontaktować się z działem technicznym lub BOK G DATA Software.

Podpis tworzymy następująco: składamy wyżej wymienione dane w następującej kolejności: ip + znacznik czasowy + nazwa api + nazwa użytkownika + hasło, następnie normalizujemy go do małych liter.

Następnie szyfrujemy otrzymaną wartość prywatnym kluczem API wybranym przez nas algorytmem. Dostępne obecnie algorytmy to:
- HMACSHA1
- HMACMD5
- HMACSHA256

Aby zweryfikować tożsamość i otrzymać token dostępu, należy przesłać następujące parametry do metody. Parametry muszą implementować następujący interfejs:

```
interface ICredentials {
    string UserName { get; set; }
    string Password { get; set; }
    string ApiKey { get; set; }
    string Signature { get; set; }
    string Uri { get; set; }
    string TimeStamp { get; set; }
    string ApiName { get; set; }
    int SigType { get; set; }
}
```

Pola:
• ApiKey - **klucz publiczny API**
• Signature - podpis wygenerowany według podanego powyżej algorytmu
• TiimeStamp - wartość w sekundach od 1.01.1970 do czasu obecnego, (podanego jako UTC)
• Uri - pozostawiamy puste
• ApiName - nazwa api, z którego wywołujemy metodę
• Pole SigType ma następujące wartości, odpowiadające użytemu algorytmowi:
• 0 - HMACSHA1
• 1 - HMACMD5
• 2 - HMACSHA256

2.2. Przykład wywołania API

```
class Credentials {
public $UserName; public $Password; public $ApiKey; public
$ApiName;
   public $Signature; public $TimeStamp; public $SigType; public
$Uri;
}

class Response {
public $AccessToken; public $ErrorCode; public $ErrorMessage;
}

$cred = new Credentials();
$cred->UserName = ‚użytkownik’;
$cred->Password = ‚hasło’;
$cred->ApiKey = ‚klucz publiczny api’;
$cred->SigType = 2; //typ szyfrowania
$cred->ApiName = ‚ks_plus’;
date_default_timezone_set(‚Europe/London’);
$date = new DateTime();
$cred->TimeStamp = $date->getTimestamp(); //ustawienie właściwego timestampa
$ip = ‚154.333.xxx.xxx’ //xxxxx - ip, z którego wywołujemy metodę
//konstruujemy string w celu podpisania go
$toauth = strtolower($ip.$cred->TimeStamp.’ks_plus’.$cred->UserName.$cred->Password);
//tworzymy podpis
$cred->Signature = hash_hmac(‚sha256’,utf8_encode($toauth),utf8_encode(‚klucz prywatny api’));
//kodujemy jsonem
$data_string = json_encode($cred);
//budujemy curl
$ch = curl_init(https://api.gdata.pl/IspRestApi/login);
curl_setopt($ch, CURLOPT_CUSTOMREQUEST, „POST”);
curl_setopt($ch, CURLOPT_POSTFIELDS, $data_string);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_HTTPHEADER, array(
‚Content-Type: application/json’,
‚Content-Length: ‚.strlen($data_string)));
$result = curl_exec($ch);
$dec = json_decode($result) ;
if ($dec->ErrorCode == 0) echo $dec->AccessToken;
else echo $dec->ErrorMessage;
```

3. Metoda: Logout

https://api.gdata.pl/login/{token}
Metoda HTTP: DELETE

3.1. Przykład PHP

```
$token =
  ‚E60E1D9B5B47037444E1F0573A4AADC1B7BD0C17F21A1BD6360B0CF593298533’
;
$data_string = json_encode($token);

$ch =
curl_init(‘https://api.gdata.pl/login/E60E1D9B5B47037444E1F0573A4AADC1B7BD0C17F21A1BD6360B0CF593298533’);
curl_setopt($ch, CURLOPT_CUSTOMREQUEST, „DELETE”);
```

```
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
$result = curl_exec($ch);
$dec = json_decode($result) ;
if ($dec->ErrorCode == 0) echo ‚Poprawnie';
else echo $dec->ErrorMessage;
```

4. Rejestracja Klienta

Metodę wykonujemy jednorazowo – pod zwróconym identyfikatorem będą podpięte wszystkie generowane licencje.

Url: https://api.gdata.pl/ IspRestApi/user
Metoda http: POST

gdzie metoda przesyłamy obiekt JSON:

```
{
„Contract”:”String content”,
„Token”:”String content”,
„Cust”:{
    „DataUrodzenia”:”String content”,
    „Email”:”String content”,
    „Firma”:”String content”,
    „Imie”:”String content”,
    „KodPocztowy”:”String content”,
    „KodRejestracyjny”:”String content”,
    „Miejscowosc”:”String content”,
    „NIP”:”String content”,
    „Nazwisko”:”String content”,
    „NrDomu”:”String content”,
    „NrMieszkania”:”String content”,
    „PESEL”:”String content”,
    „Ulica”:”String content”,
    „Wojewodztwo”:”String content”
  }
}
```

Minimalne wymagania – email

5. Alokacja klucza

Metoda HTTP: POST

https://api.gdata.pl/IspRestApi/key

przesyłamy obiekt

```
{
    „Contract”:”String content”,
    „Token”:”String content”,
    „CustomerId”:”String content”,
    „Item”:{
       „LicenceCount”:2147483647,
       „ProductSymbol”:”String content”
    }
}
```

Zwraca obiekt typu: AllocateKeyResponse (patrz pkt 7.)

Opis:

• **Contract** - (guid) identyfikator kontraktu (umowy) na podstawie której wykonywane będą operacje dla partnera. Identyfikator dostarczany jest przez G Data.
• **CustomerId** - (guid) identyfikator klienta zarejestrowanego w naszej bazie
• **Token** – token z autoryzacji
• **Item** – obiekt opisujący produkt do alokacji klucza, gdzie :
  o LicenceCount– ilość licencji (większa od 0)
  o ProductSymbol – symbol uzyskany z cennika G Data metodą pobierania cen

6. Zmiana ilości licencji klucza

Url: https://api.gdata.pl/IspRestApi/key

Metoda HTTP: PUT

W zapytaniu przesyłamy obiekt jak poniżej:
{
  „Contract”:”String content”, -- identyfikator kontraktu, string
  „Token”:”String content”, -- token uzyskany z autoryzacji API, string
  „Key”:”String content”, -- klucz, którego zmiana dotyczy , string
  „LicenceCnt”:2147483647 - ilość licencji, integer
}

7. Aktywacja /dezaktywacja licencji

7.1 Aktywacja:

Url: https://api.gdata.pl/IspRestApi/key

Metoda HTTP: PATCH
{
„Contract”:”String content”, -- identyfikator kontraktu, string
„Token”:”String content”, -- token uzyskany z autoryzacji API, string
„Key”:”String content”, -- klucz, którego zmiana dotyczy , string
}

7.2 Dezaktywacja

Url: https://api.gdata.pl/IspRestApi/key
Metoda HTTP: DELETE
{
„Contract”:”String content”, -- identyfikator kontraktu, string
„Token”:”String content”, -- token uzyskany z autoryzacji API, string
„Key”:”String content”, -- klucz, którego zmiana dotyczy , string
}

8. Generacja certyfikatu dla klucza

Url: https://api.gdata.pl/IspRestApi/key/{TOKEN}/{CONTRACT}/{KEY}
Metoda HTTP: GET
Zwraca obiekt typu stream (PDF)

9. Pobranie listy produktów dla kontraktu

Pobiera listę dostępnych produktów dla kontraktu
Url: http://172.16.200.8:7004/IspRestApi/product/{TOKEN}/{CONTRACT}

Metoda : **GET**

Zwraca obiekt JSON:

```
{
    „AccessToken”:”String content”,
    „ErrorCode”:0,
    „ErrorMessage”:”String content”,
    „Items”:[{
      „LicenceCount”:2147483647,
      „PriceNetto”:12678967.543233,
      „PriceNettoPerLicence”:true,
      „ProductName”:”String content”,
      „ProductSymbol”:”String content”
    }]
}
```

Items – tablica obiektów opisujących dostępny produkt

10. Zwracanie wartości

Zwracane przez metody API obiekty implementują interfejs, zawierający pola:
string AccessToken - token dostępu
ErrorCodes ErrorCode - kod błędu
string ErrorMessage – opis błędu

Gdzie ErrorCodes to typ wyliczeniowy zdefiniowany następująco:

```
    {
        OK = 0,
        SESSION_EXPIRED = 1,
        INVALID_TOKEN = 2,
        INVALID_PARAMETER = 3,
        INVALID_CREDENTIALS = 4,
        INVALID_API_CREDENTIALS = 5,
        LOGIN_EXISTS = 6,
        LOGIN_DOESNT_EXIST = 7,
        USER_DOESNT_EXIST = 8,
        WRONG_REG_NUMBER = 9,
        WRONG_SERIAL = 10,
        ALREADY_REGISTERED = 11,
        PARTNER_DOESNT_EXIST = 12,
        INVALID_DOCUMENT = 13,
        USER_DISABLED = 14,
        USER_BLOCKED = 15,
        KEY_DOESNT_EXIST = 16,
        ORDER_EXIST = 17,
        ORDER_CANCELLATION_NOT_ALLOWED = 18,
        NOT_REGISTERED = 19,
        MAC_KEY = 20,
        NETWORK_ERROR = 21,
        UANUTHORIZED = 22,
        UNKNOWN_REG_NUMBER = 23,
        OTHER = 999
}
```

10.1. Metoda Login

Zwraca obiekt

```
{
    string AccessToken
    ErrorCodes ErrorCode
    string ErrorMessage
}
```

10.2. Metoda alokacji klucza

Implementuje wspólny interfejs odpowiedzi i zawiera dodatkowe pola:
string ProgramName - zawiera nazwę programu
string Key - zawiera przydzielony klucz

10.3. Metoda pobierania listy produktów

Implementuje wspólny interfejs odpowiedzi i zawiera dodatkowe pola:
List<PriceListItem> Items – zawiera listę dostępnych produktów,

Zaś obiekt typu PriceListItem jest zdefiniowany jak poniżej:

string ProductSymbol – symbol produktu
string ProductName - nazwa produktu
decimal PriceNetto - nieużywany
bool PriceNettoPerLicence nieuzywany
int LicenceCount - ilość licencji
string ProductFriendlyName – przyjazna nazwa, zwykle opisuje grupę produktów

10.4. Metoda zwracająca listę kluczy dla kontraktu

Implementuje wspólny interfejs odpowiedzi i zawiera dodatkowe pola:
KeyItem**[] KeyItems – zawiera tablicę obiektów opisujących klucz**

Obiekt typu KeyItem jest opisany jak poniżej:

string Key - klucz
string Product - nazwa produktu
int LicNumber – ilość licencji
string AllocDate – data alokacji klucza

10.5. Metody aktywacji / dezaktywacji klucza /zmiany ilości licencji

Implementują wspólny interfejs odpowiedzi

10.6. Metoda tworzenia użytkownika

Implementuje wspólny interfejs odpowiedzi, dodatkowo zwraca pole
string CustomerUid - identyfikator nowego użytkownika – GUID jako string