TaaS API SPECIFICATION

Contents

Introduction

The TaaS API is accessible using Representational State Transfer (REST) and provides methods for accessing TaaS public and private data and terminology extraction methods.

Authentication and Authorization

Accessing API

The TaaS API is accessible using Representational State Transfer (REST) and basic authentication over HTTPS protocol. This means that every request must contain credentials for accessing API methods – username and password encoded with BASE64 encoding schema and include in the header of request.

Authentication constructs as following:

1. User credentials are concatenated “username:password”
2. Concatenated string is then encoded using Base64 encoding schema
3. Add value for Authorization header with method “Basic”, the space and encoded string.

Example of Authentication header of request:

```
Authorization: Basic xxxxxxxxxxxxxxxxxx
```

Impersonating

In the TaaS System there are different user types:

- TaaS Standard User – can be authorized into the TaaS Web portal and consume the TaaS API
- TaaS Machine User – can consume the TaaS API and can impersonate at TaaS API as a Standard TaaS User if the user has approved that the TaaS Machine User has the right impersonating as the user.

To support other services (Machine Users) that act in behaviour of their user access to the TaaS platform, the API consumers can add the user key generated by the TaaS system in the request’s header. This way Machine Users will be able to authorise with their credentials (as TaaS Machine Users) and impersonate as a Standard User by providing the User Key. The TaaS Machine User can also perform lookups and execute text extraction methods on public data without impersonating as a Standard User, but in that case there is no access to any private data.

To perform impersonation, together with basic authorization of the TaaS Machine User (account for external system) an additional custom header is added in request’s header. The header name is “TaaS-User-Key” and the value is the User Key of the end user:

```
Authorization: Basic xxxxxxxxxxxxxxxxxx
TaaS-User-Key: yyyyyyyyyyyyyyyyyyy
```

User Key Retrieval

A User Key can be retrieved from the TaaS system by sending the user to the TaaS portal where:

- the user can log in or register to the system and then log in
- accept User Key retrieval for specific API machine client
- The TaaS system generates a new User Key and displays it to the user
- The user can copy the User Key and paste it into the external system

External systems have to send the users to the following web address: https://[taas-webportal-url]/account/keys/create/?system=SystemUserName

The parameter "system" includes information on which external system is requesting the User Key. The value of the parameter is the username of the TaaS Machine User, and the User Key will be valid only for impersonating with that TaaS Machine User account.

Methods for Terminology Lookup[1]

Term lookup (translation)

This method looks up translations for a given term and optionally performs look-ups in the specified target language and domain. Results can be filtered out by collection types and/or by specifying collection IDs.

If the authorized user is a Machine User account without impersonation, the lookup is performed only in public data.

Request:

Synopsis: GET {WS_URL}/lookup/[sourceLang]/[term]

where

sourceLang – source language code (ISO 639-1: two-letter codes)

term – term txt

Request Headers: Accept (application/xml, application/json), Authorization, TaaS-User-Key

Request Parameters:

targetLang – target language code (can be empty) (ISO 639-1: two-letter codes)

domain – TaaS domain code (can be empty) (must start with "TaaS-")

collection – collection type or collection id (can be empty, can be multiple) (see below for possible values).

Request Message Body: empty

[1] These and further methods are available to users with appropriate access rights.

Response:

Response Headers: Content-Length, Content-Type.

Response Message Body: list of results, as json or xml (depends on the value of the *Accept* header)

Each result contains:

term – translation equivalent of source term

entryID- entry ID of translation term

collectionID – ID of the collection

collectionName – name of the collection in plain text

collectionType – type of the collection:

1- EuroTermBank Collection
2- Public TaaS Collection
3- Private TaaS Collection

domainID – domain ID of the found entry (null if no domain is specified)

domainName – domain name

language - target language code of translation (ISO 639-1: two-letter code)

If the target language is missing, then only distinct list of collections is returned, where such term is present.

Response Status: 200

Example Request:

Look-up for English translations of the Latvian term ‘dators’ in Electroniks and electrical Engineering, in EuroTermBank collections (collection=1) and also only in following collection with ids – 263 and 301.

```
GET lookup/lv/dators?targetLang=en&domain=TaaS-1500&collection=1&coll
ection=263&collection=301 HTTP/1.1
 Authorization: Basic xxxxxxxxxxxxxxxxxx
 TaaS-User-Key: yyyyyyyyyyyyyyyyyyyy
 Content-Type: application/json; charset=utf-8
```

Example Response:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Content-Length: 146
 [
   {
        "term": "computer",
        "entryID": 464626,
        "collectionID": 263,
        "collectionName":     "Latviešu-angļu     enerģētikas     un
elektrotehnikas vārdnīca",
        "collectionType": 1,
        "domainID": "TaaS-1505",
        "domainName": "Electronics and electrical engineering",
```

```
        "language": "en"
    },
    {
        "term": "computer",
        "entryID": 616107,
        "collectionID": 301,
        "collectionName": "Europos žodynas: Eurovoc, 4.2 versija",
        "collectionType": 1,
        "domainID": "TaaS-2000",
        "domainName": "Social sciences",
        "language": "en"
    }
]
```


Term Extraction

This method tags term candidates in text: method 1 marks term candidates only and method 2 includes also references to terminology entries, i.e., marks term candidates and also searches in TaaS external resources (e.g., Eurotermbank.com) for translations.

If the authorized user is a Machine User account without impersonation, lookup is performed only in public data.

Request:

Synopsis: POST {WS_URL}/extraction

Request Headers: Authorization, TaaS-User-Key, Content-Type

Request Parameters:

sourceLang – source language code (for the source text's language) (ISO 639-1: two-letter code)

targetLang – target language code (for the language in which translation candidates should be looked up) (can be empty) (ISO 639-1: two-letter code)

method – 1-statistical terminology annotation, 2- statistical terminology annotation with references to terminology entries, 4- Terminology DB based terminology annotation (fast)

domain – TaaS domain code (can be empty)

collection – collection ID (method 4 only, can be specified multiple times)

external – true/false (method 4 only, default: false) – whether to include terms from external sources

outformat – rdf (can be omitted, method 4 only) – returns the term collection in RDF format instead of TBX

Request Message Body: plain text

Response:

Response Headers: Content-Length, Content-Type.

Response Message Body: xml

Response Status: 200

Example Request Method 1:

Tags terms in plain text. Returns the result in an XML document

```
 POST  /extraction/?sourceLang=lv&method=1&targetLang=en&domain=TaaS-
1500 HTTP/1.1
 Authorization: Basic xxxxxxxxxxxxxxxxxx
 TaaS-User-Key: yyyyyyyyyyyyyyyyyyyyyyyyyyyy
 Content-Length: 19
 Content-Type: text/plain

 Personālais dators
```

Example Response:

```
HTTP/1.1 200 OK
Cache-Control: no-cache
Pragma: no-cache
Content-Type: application/xml; charset=utf-8
Expires: -1
Server: Microsoft-IIS/7.5
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Date: Thu, 17 Jan 2013 07:50:05 GMT
Content-Length: 203
<extractionResult>
    <text>
        <![CDATA[<TENAME  SCORE="1.0"  MSD="A-msnp-y-----------------
f- N-msn---------n-----------l-" LEMMA="personāls dators">Personālais
dators</TENAME>]]>
    </text>
</extractionResult>
```

Example Request Method 2

Tags terms in plain text and adds references to term entries that are found in the TaaS platform. Term entries are added inline. The call returns the result in an XML document

```
 POST  /extraction/?sourceLang=lv&method=2&targetLang=en&domain=TaaS-
1500 HTTP/1.1
 Authorization: Basic xxxxxxxxxxxxxxxxxx
 TaaS-User-Key: yyyyyyyyyyyyyyyyyyyyyyyy
 Content-Type: text/plain

 Personāls dators
```

Example Response:

```
HTTP/1.1 200 OK
Cache-Control: no-cache
Pragma: no-cache
Content-Type: application/xml; charset=utf-8
Expires: -1
Server: Microsoft-IIS/7.5
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Date: Thu, 17 Jan 2013 07:50:05 GMT
Content-Length: 146

<extractionResult>
<text>
<![CDATA[<TENAME termID="etb-1" SCORE="1.0" MSD="A-msnp-n------------
------f-     N-msn---------n-----------l-"     LEMMA="person&#x101;ls
dators">Personāls dators</TENAME>]]></text>
```

```
<terms>
<![CDATA[<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE martif SYSTEM
'http://www.ttt.org/oscarstandards/tbx/TBXcoreStructV02.dtd'><martif
type="TBX" xml:lang="lv"><martifHeader><fileDesc><sourceDesc><p>From
the TaaS Terminology Annotation
Service</p></sourceDesc></fileDesc><encodingDesc><p
type="XCSURI">http://www.ttt.org/oscarstandards/tbx/TBXXCS.xcs</p></e
ncodingDesc></martifHeader><text><body><termEntry id="etb-1"><admin
type="sourceLanguage">lv</admin><descrip
type="subjectField"/><langSet
xml:lang="lv"><ntig><termGrp><term>Personāls
dators</term><termCompList
type="lemma"><termCompGrp><termComp>personāls</termComp><termNote
type="partOfSpeech">A</termNote><termNote
type="grammaticalGender">masculine</termNote><termNote
type="grammaticalNumber">singular</termNote><termNote
type="transferComment">A-msnp-n-----------------f-
</termNote></termCompGrp><termCompGrp><termComp>dators</termComp><ter
mNote type="partOfSpeech">N</termNote><termNote
type="grammaticalGender">masculine</termNote><termNote
type="grammaticalNumber">singular</termNote><termNote
type="transferComment">N-msn---------n----------l-
</termNote></termCompGrp></termCompList></termGrp><descrip
type="reliabilityCode">1</descrip><admin
type="score">1.0</admin><admin target="ba33ef7c-e7e7-4f3a-aed6-
390d1e255cc4.output.step1.completed" type="sourceIdentifier"/><xref
target="http://www.eurotermbank.com/Collection.aspx?collectionid=149"
type="xSource">ISO IT (ISO 2382-1)</xref><xref
target="http://www.eurotermbank.com/Collection.aspx?collectionid=297"
type="xSource">Angļu-latviešu muitas terminu vārdnīca</xref><xref
target="http://www.eurotermbank.com/Collection.aspx?collectionid=315"
type="xSource">ISO/IEC 2382-1:1993 Informācijas tehnoloģijas -
Terminu vārdnīca - 1. daļa: Pamattermini</xref><xref
target="http://www.eurotermbank.com/Collection.aspx?collectionid=396"
type="xSource">Angļu-latviešu-krievu informātikas termini</xref><xref
target="http://www.eurotermbank.com/Collection.aspx?collectionid=420"
type="xSource">Microsoft Public Terminology
Collection</xref></ntig></langSet></termEntry></body></text></martif>
]]></terms>
</extractionResult>
```

Example Request Method 4

Tags terms with terms from terminology collections in plain text and adds references to term entries that are found in the TaaS platform. Term entries are added inline. The call returns the result in an XML document

```
POST /extraction?sourceLang=en&method=4&targetLang=hr&domain=TaaS-
1500 HTTP/1.1
Authorization: Basic xxxxxxxxxxxxxxxxxx
TaaS-User-Key: yyyyyyyyyyyyyyyyyyyyyy
Content-Type: text/plain

I have personal computer
```

Example Response:

```
HTTP/1.1 200 OK
Cache-Control: no-cache
Pragma: no-cache
Content-Type: application/xml; charset=utf-8
Expires: -1
Server: Microsoft-IIS/7.5
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Date: Wed, 12 Mar 2014 08:05:03 GMT
Content-Length: 9409

<extractionResult>
    <text>
        <![CDATA[
I have <TENAME ID="1227550,1227399">personal computer</TENAME>
        ]]>
    </text>
    <terms>
        <![CDATA[
<martif type="TBX"><martifHeader><encodingDesc><p
type="XCSURI">http://www.ttt.org/oscarstandards/tbx/TBXXCS.xcs</p></encodingDesc></
martifHeader><text><body>
<termEntry id="1227550"><admin type="sourceLanguage">en</admin><admin
type="subsetOwner">Microsoft Corp.</admin><admin
type="securitySubset">public</admin><transacGrp><transac
type="transactionType">creation</transac><transacNote
type="responsibility">Imported automatically by
Tilde</transacNote><date>13:58:53, Mon Jan 11,
2010</date></transacGrp><descrip
type="subjectField">3236</descrip><note>information technology and
data processing</note><langSet lang="en"><transacGrp><transac
type="transactionType">creation</transac><transacNote
type="responsibility">Imported automatically by
Tilde</transacNote><date>13:58:53, Mon Jan 11,
2010</date></transacGrp><ntig><transacGrp><transac
type="transactionType">creation</transac><transacNote
type="responsibility">Imported automatically by
Tilde</transacNote><date>13:58:53, Mon Jan 11,
2010</date></transacGrp><termGrp><term>personal
computer</term></termGrp></ntig><descripGrp><descrip
type="definition">A microcomputer designed for use by one person at a
time. Personal computers do not need to share the processing, disk,
and printer resources of another
computer.</descrip></descripGrp></langSet><langSet
lang="hr"><transacGrp><transac
type="transactionType">creation</transac><transacNote
type="responsibility">Imported automatically by
Tilde</transacNote><date>13:58:53, Mon Jan 11,
2010</date></transacGrp><ntig><transacGrp><transac
type="transactionType">creation</transac><transacNote
type="responsibility">Imported automatically by
Tilde</transacNote><date>13:58:53, Mon Jan 11,
2010</date></transacGrp><termGrp><term>osobno
računalo</term></termGrp></ntig></langSet></termEntry>
<termEntry id="1227399"><admin type="sourceLanguage">en</admin><admin
type="subsetOwner">Microsoft Corp.</admin><admin
type="securitySubset">public</admin><transacGrp><transac
type="transactionType">creation</transac><transacNote
type="responsibility">Imported automatically by
Tilde</transacNote><date>13:58:53, Mon Jan 11,
2010</date></transacGrp><descrip
```

```
type="subjectField">3236</descrip><note>information technology and
data processing</note><langSet
lang="en"><transacGrp><ntig<termGrp><term>personal
computer</term></termGrp></ntig></langSet><langSet
lang="hr"><ntig><termGrp><term>osobno
računalo</term></termGrp></ntig></langSet></termEntry>
</body></text></martif>]]>
    </terms>
</extractionResult>
```

Methods for Terminology Management

Retrieve Entry

This method retrieves an entry by its entryID. If the collection is private and not shared, only the authorized user of the collection owner's project can access the entry.

If the authorized user is a Machine User account without impersonation, only a public entry can be retrieved.

Request:

Synopsis: Get {WS_URL}/terms/[entryID]

The [entryID] is the ID of the term entry to be retrieved.

Request Headers: Authorization, TaaS-User-Key

Request Parameters: -

Request Message Body: -

Response:

Response Headers: Content-Length, Content-Type

Response Message Body: terminology entry in TBX format

Response Status: 200

Example Request:

Retrieves the entry with the id 1281789.

```
GET /terms/638733 HTTP/1.1
Authorization: Basic xxxxxxxxxxxxxxxxxx
TaaS-User-Key: MjIyMy1yRUVGUi01NTY3LTIyMzAtNDRGRkY=
```

Example Response:

```
HTTP/1.1 200 OK
Content-Type: application/xml; charset=utf-8
Content-Length: 125

<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE martif  SYSTEM 'http://www.ttt.org/oscarstandards/tbx/TBXcoreStruc
tV02.dtd'>
<martif type="TBX" xml:lang="lv">
   <martifHeader>
      <fileDesc>
         <sourceDesc>
            <p>TaaS - Terminology As a Service</p>
```

```
        </sourceDesc>
      </fileDesc>
      <encodingDesc>
       <p type="XCSURI">http://www.ttt.org/oscarstandards/tbx/TBXXCS.xcs</p>
      </encodingDesc>
  </martifHeader>
  <text>
    <body>
      <termEntry id="1281935">
         <langSet xml:lang="lv">
           <ntig>
             <termGrp>
                  <term>sample text</term>
            </termGrp>
           </ntig>
         </langSet>
      </termEntry>
    </body>
  </text>
</martif>
```

Insert a New Entry or Update an Existing Entry

This method updates an existing entry or inserts a new entry. If an entry contains attribute “id” at entry level in TBX document, the collection ID is retrieved from DB. If the entry id is not present, the API checks the received collection id. If the user has appropriate rights (editor or administrator of this collection), the entry is inserted as a new entry to this collection or if the entry is a new version of an existing entry, the existing entry is replaced with the updated version. After insertion, all search tables are immediately updated.

If the authorized user is a Machine User account without impersonation, no action is performed.

Request:

Synopsis: POST {WS_URL}/terms?collectionID=[collectionID]

Request Headers: Authorization, TaaS-User-Key, Content-Length,

Request Parameters: -

Request Message Body: an entry in the format of the TBX element

Response:

Response Headers: Content-Length, Content-Type.

Response Message Body: ID of insert or updated entry.

Response Status: 200, 500 (in case of error)

Example Request:

Inserts a new entry into the collection with the ID 3464

```
POST /collections/3464 HTTP/1.1
Authorization: Basic xxxxxxxxxxxxxxxxxx
TaaS-User-Key: yyyyyyyyyyyyyyyyyy
Content-Length: 202

<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE martif  SYSTEM 'http://www.ttt.org/oscarstandards/tbx/TBXcoreStruc
tV02.dtd'>
<martif type="TBX" xml:lang="lv">
   <martifHeader>
      <fileDesc>
```

```
          <sourceDesc>
             <p>TaaS - Terminology As a Service</p>
          </sourceDesc>
       </fileDesc>
       <encodingDesc>
        <p type="XCSURI">http://www.ttt.org/oscarstandards/tbx/TBXXCS.xcs</p>
       </encodingDesc>
  </martifHeader>
  <text>
    <body>
      <termEntry id="1281935">
         <langSet xml:lang="lv">
           <ntig>
             <termGrp>
                  <term>sample text</term>
            </termGrp>
           </ntig>
         </langSet>
      </termEntry>
    </body>
  </text>
</martif>
```

Example Response:

```
HTTP/1.1 200 OK
Content-Type: text/plain
Content-Length: 4

3335
```

Methods for Collection Retrieval

Get a List of Collections

This method retrieves a list of available collections. If a filter is passed along the request, the collections are filtered using the filter parameters. Available filters are: a list of languages, and a list of domains.

If the authorized user is a Machine User account without impersonation, only public collections are returned.

Request:

Synopsis: GET {WS_URL}/collections

Request Headers: Accept (application/xml, application/json), Authorization, TaaS-User-Key

Request Parameters:

lang - collection languages (can be multiple),

domain – collection domain (can be multiple)

Request Message Body:

Response:

Response Headers: Content-Length, Content-Type.

Response Message Body: a list of collections

Response Status: 200

Example Request:

```
GET /collections?lang=en&lang=lv$domain=Taas-1500 HTTP/1.1
Authorization: Basic xxxxxxxxxxxxxxxxxx
TaaS-User-Key: MjIyMy1yRUVGUi01NTY3LTIyMzAtNDRGRkY=
Accept: application/json
```

Example Response:

```
HTTP/1.1 200 OK
Content-Length: 4566
Content-Type: application/json; charset=utf-8

[
 {
   "id": 5249
  ,"name":"ISO IT (ISO 2382-1)"
  ,"description":"International Standard ISO/IEC 2382-1"
  ,"languages":
       [
        {
          "id":"en"
          ,"count":"15"
        }
        ,{
          "id":"lv"
          ,"count":"25"
        }
       ]
 }
,{
    "id": 5246
   ,"name":"Mežtehnikas, mežsaimniecības terminu vārdnīca"
   ,"description":"Dictionary of forestry"
   ,"languages":
       [
        {
          "id":"fr"
          ,"count":"17"
        }
        ,{
          "id":"lv"
          ,"count":"34"
        }
       ]
 }
,{
   "id": 5243
   ,"name":"SYNABA – Słownik sł&#243;w kluczowych"
   ,"description":"SYNABA Dictionary of key words"
   ,"languages":
       [
        {
          "id":"hu"
          ,"count":"65"
        }
        ,{
          "id":"lv"
          ,"count":"85"
        }
       ,{
```

```
            "id":"de"
            ,"count":"105"
          }
        ]
  }
 ]
```

Get a Collection

This method retrieves all entries of a selected collection.

If the authorized user is a Machine User account without impersonation, only entries from public collection are returned.

Request:

Synopsis: GET {WS_URL}/collections/[collectionID]

Request Headers: Authorization, TaaS-User-Key

Request Parameters:

format – export format (optional).

Possible values:

tbx – export data in TBX format (default, used if parameter not specified)

tsv – export data as Tab Separated Values

csv – export data as Comma Separated Values

moses – export data in Moses format

Request Message Body: -

Response:

Response Headers: Content-Length, Content-Type

Response Message Body: term entries in the TBX format

Response Status: 200

Example Request:

```
GET /collections/3345?format=tbx HTTP/1.1
Authorization: Basic xxxxxxxxxxxxxxxxxx
TaaS-User-Key: MjIyMy1yRUVGUi01NTY3LTIyMzAtNDRGRkY=
```

Example Response:

```
HTTP/1.1 200 OK
Content-Length: 4566
Content-Type: application/xml; charset=utf-8

<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE martif  SYSTEM 'http://www.ttt.org/oscarstandards/tbx/TBXcoreStruc
tV02.dtd'>
<martif type="TBX" xml:lang="lv">
   <martifHeader>
      <fileDesc>
         <sourceDesc>
            <p>TaaS - Terminology As a Service</p>
        </sourceDesc>
      </fileDesc>
      <encodingDesc>
       <p type="XCSURI">http://www.ttt.org/oscarstandards/tbx/TBXXCS.xcs</p>
      </encodingDesc>
  </martifHeader>
```

```
  <text>
    <body>
      <termEntry id="1281935">
         <langSet xml:lang="lv">
           <ntig>
             <termGrp>
                  <term>sample text</term>
              </termGrp>
           </ntig>
         </langSet>
      </termEntry>
      <termEntry id="1281936">
         <langSet xml:lang="lv">
           <ntig>
             <termGrp>
                  <term>sample text Nr 2</term>
              </termGrp>
           </ntig>
         </langSet>
      </termEntry>
    </body>
  </text>
</martif>
```

Methods for Classifiers

Get a List of Domains

This method retrieves a list of available domains from the TaaS domain classification.

Request:

Synopsis: GET {WS_URL}/domains

Request Headers: Accept (application/xml, application/json), Authorization

Request Parameters: -

Request Message Body: -

Response:

Response Headers: Content-Length, Content-Type.

Response Message Body: a list of domains

Response Status: 200

Example Request:

```
GET /domains HTTP/1.1
Authorization: Basic xxxxxxxxxxxxxxxxxx
Accept: application/json
```

Example Response:

```
HTTP/1.1 200 OK
Content-Length: 4566
Content-Type: application/json; charset=utf-8

[
  {
    "id": "TaaS-2100",
    "parent": null,
```

```
    "name": "Natural sciences"
}
,{
    "id": "TaaS-2101",
    "parent": "TaaS-2100",
    "name": "Astronomy"
  }
]
```

Get a List of Languages

This method retrieves a list of languages supported by the TaaS platform.

Request:

Synopsis: GET {WS_URL}/languages

Request Headers: Accept (application/xml, application/json), Authorization

Request Parameters: -

Request Message Body: -

Response:

Response Headers: Content-Length, Content-Type.

Response Message Body: a list of languages

Response Status: 200

Example Request:

```
GET /languages HTTP/1.1
Authorization: Basic xxxxxxxxxxxxxxxxxx
Accept: application/json
```

Example Response:

```
HTTP/1.1 200 OK
Content-Length: 4566
Content-Type: application/json; charset=utf-8

[
  {
    "id":"lv"
   ,"name":"Latvian"

  }
,{
    "id":"en"
   ,"name":"English"

  }
]
```

Consumption Example of the API from .NET Code

Get entry by id, with basic authentication and also impersonation

```
public async Task<String> GetEntry(string entryID, string userKey)
  {
```

```
        //create authorization token
        string authorization = Convert.ToBase64String(
                System.Text.Encoding.ASCII.GetBytes(
                  String.Format(
                        "{0}:{1}",
                        ConfigurationManager.AppSettings["API_UserName"],
                        ConfigurationManager.AppSettings["API_Password"])
                   )
                );

        // create request object
        var httpClient = new HttpClient();
        httpClient.BaseAddress = new Uri(
          ConfigurationManager.AppSettings["API_URL"]
          );

        //add headers to the request
        httpClient.DefaultRequestHeaders.Authorization
          = new AuthenticationHeaderValue("basic", basic);
        httpClient.DefaultRequestHeaders.Add("TaaS-User-Key", useKey);
        httpClient.DefaultRequestHeaders.Add("Accept", "application/xml");

        //perform request and wait results
        var response = await httpClient.GetAsync("/terms/" + entryID);
        return await response.Content.ReadAsStringAsync();
    }
```