LWIN Search API v1

Document Revision 1.2
Date of Issue: 06 April 2020
Date of revision: 13 May 2020

Nick Palmer

Product Lead

Table of Contents

1. Purpose

To provide the API end point information and examples of the web services available for the LWIN Search service.

2. Glossary of Terms

Term	Meaning
LWIN	LWIN - the Liv-ex Wine Identification Number – serves as a universal wine identifier for the wine trade. LWIN is a unique seven to eighteen-digit numerical code that can be used to quickly and accurately identify a product. LWIN allows wine companies to keep their preferred naming system, while introducing a new universal code. This document refers to LWIN7 (7-digit, wine) and LWIN11 (11-digit, wine + vintage) codes only.

3. Technical Standards

- Permitted users will be issued with a unique token (CLIENT_KEY) and password (CLIENT_SECRET) combination to control the access for all the web services covered under Exchange Integration.
- The web services will consume and produce JSON. The user can provide the content type in the request header. If the user does not provide any information, then the default content type will be JSON.
- The project will support ISO 8601.
- The project will only support HTTPS protocol for client and server communications.
- The API will support the following methods:
    - POST for create operation
- Pretty printing for output readability only is supported if required
- Compression for bandwidth savings are used
- Authentication mechanism will be custom based on CLIENT_KEY and CLIENT_SECRET
- The APIs will be accessible at https://api.liv-ex.com/ followed by their specific base URIs

4. Request Header

This information will be used to authenticate valid access to the REST API. Each user will have to provide the following information in the request header.

Param

Name	Mandatory	Description
CLIENT_KEY	Y	A valid merchant GUID which will be unique for each merchant.
CLIENT_SECRET	Y	Password/Secret for the merchants CLIENT_KEY.
ACCEPT	Y	Accept header is a way for a client to specify the media type of the response content it is expecting. The values for the content type will be application/json or application/xml. If no/ invalid content type is found in the request, then JSON format will be used by default.

CONTENT-TYPE	Y for POST requests	Content-type is a way to specify the media type of request being sent from the client to the server. The values for the content type will be application/json or application/xml. If no/ invalid content type is found in the request, then JSON format will be used by default.

e.g.
CLIENT_KEY: 94B5CC70-BC3D-49C3-B636-C3C7552E543D
CLIENT_SECRET: merchantpasswd
ACCEPT: application/json
CONTENT-TYPE: application/json

Invalid header JSON response

```
{
    "status": "Unauthorized",
    "httpCode": "401",
    "message": "Request was unsuccessful",
    "livexCode": "R000"
    "apiInfo": {
        "version": "1.0",
        "timestamp": 1518524979121,
        "provider": "Liv-ex"
    }
}
```

Invalid header XML response

```
<Response>
    <Status>Unauthorized</Status>
    <HttpCode>401</Code>
    <Message>Request was unsuccessful.</Message>
    <LivexCode>R001</LivexCode>
     <ApiInfo>
        <Version>1.0</Version>
        <Timestamp>2017-11-04T11:12:30</Timestamp>
        <Provider>Liv-ex</Provider>
    </ApiInfo>
<Response>
```

5. API Listing

5.1 LWIN Search service (POST method)

Description

This service will be used to search for products and their associated metadata in the LWIN dataset. It allows the caller to search by LWIN number (both LWIN7 and LWIN11) or a text string.

Base URI

/lwin/search/v1/lwinSearch

Request Parameters

Name	Mandatory	Description
searchInput	Y	Minimum 3-character input. Case insensitive.

		Multiple terms supported **Type:** alphanumeric

Sample Request Body

JSON #1 – single string

```
{
   "searchInput": "Eagle"
}
```

JSON #2 – multiple strings

```
{
   "searchInput": "brunello di mont"
}
```

JSON #3 – LWIN7

```
{
   "searchInput": "1013296"
}
```

JSON #4 – LWIN11

```
{
   "searchInput": "10132962016"
}
```

Response

The LWIN Search service will respond with HTTP Code 200 OK in a successful response. The volume of data returned will be dependent on the searchInput input.

Response parameters

Name	Description
lwin	The LWIN7 code of the product returned by the search service **Type:** integer(7)
lwin11	The LWIN11 code related to the metadata returned. This will be the first vintage where the naming conventions shown in the metadata was first recorded **Type:** integer(11)
producerTitle	Title of producer or owner of wine **Type:** alphanumeric
producerName	Producer or owner of wine **Type:** alphanumeric
wine	Name of wine (brand and/or grape and/or technical term) **Type:** alphanumeric
country	Country of origin **Type:** alphanumeric
region	Region of origin (where applicable, specific to local laws) **Type:** alphanumeric
subRegion	Sub-region of origin (where applicable, specific to local laws) **Type:** alphanumeric
site	Site within sub-region (where applicable, specific to local laws) **Type:** alphanumeric
parcel	Parcel within site (where applicable, specific to local laws) **Type:** alphanumeric
colour	Colour of LWIN product **Type:** alphanumeric **Values:** "white", "red", "rose", null
type	LWIN beverage type **Type:** alphanumeric
subType	Subcategory of LWIN type **Type:** alphanumeric
designation	Officially assigned status (specific to local laws) **Type:** alphanumeric
classification	Officially declared quality level (where applicable, specific to local laws) **Type:** alphanumeric
vintageConfiguration	Vintage pattern of the wine (e.g. single vintage only, production ended) **Type:** alphanumeric **Values:** "sequential", "nonSequential", "singleVintageOnly"
displayName	**Type:** alphanumeric
status	**Type:** alphanumeric **Values:** "live", "deleted", "combined"

dateCreated	**Type:** datetime / epoch
lastUpdateDate	**Type:** datetime / epoch

JSON Response

The response is sent per request. A maximum of 250 results will be included. For wider queries we recommend downloading the full LWIN7 dataset (see www.li-ex.com/lwin for more information).


```
{
    "status": "OK",
    "httpCode": "200",
    "message": "Request completed successfully",
    "internalErrorCode": "R001",
    "apiInfo": {
        "version": "1.0",
        "timestamp": 1585911767765,
        "provider": "Liv-ex"
    },
    "searchResults": [
        {
            "searchResult": {
                "lwin": "2509286",
                "lwin11": "25092861000",
                "producerTitle": null,
                "producerName": "Screaming Eagle",
                "wine": "Second Flight 2006-2009 Assortment",
                "country": "United States",
                "region": "California",
                "subRegion": "Napa Valley",
                "parcel": null,
                "site": null,
                "colour": "Red",
                "type": "Wine",
                "subType": "Still",
                "designation": "AVA",
                "classification": null,
                "vintageConfiguration": "sequential",
                "displayName": "Second Flight 2006-2009 Assortment, Screaming Eagle,
Napa Valley",
                "dateCreated": "1584704476586",
                "lastUpdateDate": "1584704476586"
            }
        },
        {
            "searchResult": {
                "lwin": "2509299",
                "lwin11": "25092992015",
                "producerTitle": null,
                "producerName": "Screaming Eagle",
                "wine": "The Flight",
                "country": "United States",
                "region": "California",
                "subRegion": "Napa Valley",
                "parcel": null,
                "site": null,
                "colour": "Red",
                "type": "Wine",
                "subType": "Still",
                "designation": "AVA",
                "classification": null,
                "vintageConfiguration": "sequential",
```

```
                "displayName": "The Flight, Screaming Eagle, Napa Valley",
                "dateCreated": "1584704534000",
                "lastUpdateDate": "1584705609126"
            }
        },
        {
            "searchResult": {
                "lwin": "2509299",
                "lwin11": "25092992007",
                "producerTitle": null,
                "producerName": "Screaming Eagle",
                "wine": "Second Flight",
                "country": "United States",
                "region": "California",
                "subRegion": "Napa Valley",
                "parcel": null,
                "site": null,
                "colour": "Red",
                "type": "Wine",
                "subType": "Still",
                "designation": "AVA",
                "classification": null,
                "vintageConfiguration": "sequential",
                "displayName": "Second Flight, Screaming Eagle, Napa Valley",
                "dateCreated": "1584704534013",
                "lastUpdateDate": "1584704534013"
            }
        }
    ]
}
```


Invalid JSON response


```
{
    "status": "OK",
    "httpCode": "200",
    "message": "Request completed successfully",
    "internalErrorCode": "R001",
    "apiInfo": {
        "version": "1.0",
        "timestamp": 1586185110983,
        "provider": "Liv-ex"
    },
    "searchInput": "mo",
    "errors": {
        "error": [
            {
                "code": "L047",
                "message": "Please enter a minimum of 3 characters."
            }
        ]
    }
}
```


6. Response Codes

This section describes the response codes that will be returned by the Exchange Integration services.

Code	Message
R000	Request was unsuccessful
R001	Request completed successfully

R002	Please enter a minimum of 3 characters.

6.1 Request validation error codes

Code	Message
L002	Incorrect LWIN: [%]
L007	Invalid LWIN7 [%] and vintage combination.
L047	Please enter a minimum of 3 characters.

6.2 HTTP Status codes

HTTP defines a bunch of meaningful status codes that can be returned from our API. These can be leveraged to help our API Merchants/consumers route their responses accordingly:

Code	Message
200 OK	Response to a successful GET, POST, PUT, DELETE. Can also be used for a POST that doesn't result in a creation.
201 Created	Response to a POST that results in a creation.
202 Accepted	The request has been accepted and will be processed later. It is a classic answer to asynchronous calls (for better UX or performances).
204 No Content	Response to a successful request that won't be returning a body (like a DELETE request)
400 Bad Request	The request is malformed, such as if the body does not parse
401 Unauthorized	When no and/or invalid authentication details are provided. Can also be used to trigger an auth popup if API is used from a browser
403 Forbidden	When authentication succeeded but authenticated user doesn't have access to the resource
404 Not Found	When a non-existent resource is requested
405 Method Not Allowed	When an HTTP method is being requested that isn't allowed for the authenticated user
406 Not Acceptable	Nothing matches the Accept-* Header of the request. As an example, you ask for an XML formatted resource, but it is only available as JSON.
409 Conflict	Indicates one or more supplied parameters are triggering a validation error. A relevant TR code should be returned in the response.

410 Gone	Indicates that the resource at this end point is no longer available. Useful as a blanket response for old API versions
415 Unsupported Media Type	If incorrect content type was provided as part of the request
422 Unprocessable Entity	Used for validation errors. Should be used if the server cannot process the entity, e.g. if an image cannot be formatted or mandatory fields are missing in the payload.
429 Too Many Requests	When a request is rejected due to rate limiting
500 Internal Server Error	The general catch-all error when the server-side throws an exception. The request may be correct, but an execution problem has been encountered at our end.

7. LWIN search SDK

The lwinSearch API is also available as a JavaScript SDK for easy implementation into applications and websites.

The .js library file can downloaded here: search_lib.js

The component can be called using the following parameters:

```
<script type="text/javascript">
    this search = new SearchLib({
        autoSuggestionDiv: "lwinAutoSuggestions",
        displayInSearch: "displayname",
        listSize: 5,
        apiUrl: "https://api.liv-ex.com/lwin/search/v1/lwinSearch",
        CLIENT_KEY: "<liv-ex_supplied_key>",
        CLIENT_SECRET: "<liv-ex_supplied_secret>","
    })
        function onSearch(e) {
        search.searchApi(e);
    }
</script>
```

Adjustable SDK Parameters

Name	Description
listSize	The number of autocomplete suggestions to show beneath the search box **Type:** integer
apiUri	The address of the service
CLIENT_KEY	A valid merchant GUID which will be unique for each merchant.
CLIENT_SECRET	Password/Secret for the merchants CLIENT_KEY.

END