LIV|EX

THE FINE WINE MARKET

Wine Data API v1

Document Revision 1.0
Date of Issue: 03/06/2024
Date of revision: 03/06/2024

Barnabas Mullan
Senior Business Analyst

Table of Contents

1. Purpose

To provide the API end-point information and examples of the data available from Wine Data API v1.

2. Glossary of Terms

Term	Meaning
LWIN	LWIN - the Liv-ex Wine Identification Number – serves as a universal wine identifier for the wine trade. A unique seven to eighteen-digit numerical code used to quickly and accurately identify a product. LWIN allows wine companies to keep their preferred naming system, while introducing a new universal code.
Wine	The word wine below is referring to a specific wine (the producer and brand, grape or vineyard), vintage and unit size combination.

3. Technical Standards

- Permitted users will be issued with a unique token (CLIENT_KEY) and password (CLIENT_SECRET) combination to control the access to the web service.
- The web services will consume and produce both XML and 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:
    1. POST for read operation
- Pretty printing for output readability only is supported if required
- Compression for bandwidth savings are used
- For HTTP users who can only work on GET & POST methods, we provide a Header 'X-HTTP-Method-Override' for PUT & DELETE
- Authentication mechanism will be custom based on CLIENT_KEY and CLIENT_SECRET
- For PUSH services we require a direct POST URL which should be backed by a service capable of accepting and process XML or JSON payload as POST request.
- 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. Please note that the API expects the 4 headers as listed within this documentation and submitting a request with additional headers may lead to errors and/or failed responses.

Parameter

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 or invalid content type is found in the request, then JSON format will be used by default.
CONTENT-TYPE	Y	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 or invalid content type is found in the request, then JSON format will be used by default.

Header example

```
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": "2.0",
         "timestamp": "2015-06-04T11:12:30",
         "provider": "Liv-ex"
     }
}
```

Invalid header XML response

```
<response>
      <Status>Unauthorized</Status>
      <HttpCode>401</Code>
      <Message>Request was unsuccessful</Message>
      <LivexCode>R001</LivexCode>
            <ApiInfo>
       <Version>2.0</Version>
       <Timestamp>2015-06-04T11:12:30</Timestamp>
       <Provider>Liv-ex</Provider>
     </ApiInfo>
</response>
```

5. API Listing

5.1 Wine Data (Post Method)

Description

This service will be used to retrieve Liv-ex data points for a given LWIN code or listID.

Base URI

/data/v1/wineData

URI Pagination Parameters

Name	Mandatory	Description
?limit	N Default = all results	Max value 10000 Type: integer
?offset	N Default = 1	Type: Alphanumeric

If the limit is not specified in the request then the API returns the default value on the offset 1. Example base URI including pagination: /data/v1/wineData?limit=500&offset=1

When using the listID request parameter for large stock lists, it is recommended to use a pagination limit of no more than 1000, and increment the offset value in subsequent API calls to consume all the data.

Request Parameters

Name	Mandatory	Description
lwin	N	LWIN11/16/18. Multiple values are permitted (max. 1500 per request). **Type:** string array
listID	N	The ListID (either the value "stock list" or the hexadecimal ID created from List Manager POST method). Using ListID will retrieve Liv-ex data for all every LWIN saved in the specified list. Type: alphanumeric
dataType	Y	Defines the data type to return. Currently only one value is available: **Wine Data Type** / **Attribute name** Drinking Windows / drinkingWindow Type: alphanumeric array

Operational Notes:

1. Requests may contain LWINs of varying lengths. For the purposes of retrieving data, all LWINs will be truncated to an LWIN11 length.

2. When submitting both LWINs and a ListID, API calls will prioritise LWINs and ignore the specified ListID.
3. LWIN11s are de-duplicated for the response.
4. Valid LWIN11s for which there is no matching data will be returned with NULL results. Invalid LWINs are omitted from the response.
5. If LWINs are provided in the request, the API response is sorted in order that the LWINs are submitted.
6. If ListID is provided in the request, the API response is sorted in the order that products were added to the list (from oldest to newest).

Sample JSON Request Body

```
{
    "wineData": {
        "listID": "stock list",
        "dataType": [
            "drinkingWindow"
        ]
    }
}
```

Sample XML Request Body

```
<root>
<wineData>
<lwin>10127812011</lwin>
<dataType>drinkingWindow</dataType>
</wineData>
</root>
```

Response Parameters

Parameter Name	Description
lwin	The LWIN11 code requested. Type: alphanumeric
listID	The ListID requested. Type: alphanumeric
dataType	The datatype requested Type: alphanumeric
metadata1	Information associated with the selected dataType. Not always used. For drinkingWindow: Returns the “Drink From” year. Type: alphanumeric
metadata2	Information associated with the selected dataType. Not always used. For drinkingWindow: Returns the “Drink To” year. Type: alphanumeric
metadata3	Information associated with the selected dataType. Not always used. Type: alphanumeric

metadata4	Information associated with the selected dataType. Not always used. Type: alphanumeric

Sample JSON Response

```
{
    "status": "OK",
    "httpCode": "200",
    "message": "Request completed successfully",
    "internalErrorCode": "R001",
    "apiInfo": {
        "version": "1.0",
        "timestamp": 1717407687578,
        "provider": "Liv-ex"
    },
    "pageInfo": {
        "totalResults": 1,
        "limit": 10000,
        "offset": 1
    },
    "wineData": {
        "listID": null,
        "data": [
            {
                "lwin": "10127812011",
                "dataDetail": [
                    {
                        "dataType": "drinkingWindow",
                        "metadata1": "2020",
                        "metadata2": "2038",
                        "metadata3": null,
                        "metadata4": null
                    }
                ],
                "error": null
            }
        ]
    },
    "errors": null
}
```

Sample JSON Failed Response

```
{
    "status": "Unauthorized",
    "httpCode": "401",
    "message": "Request was unsuccessful",
    "internalErrorCode": "R000",
    "apiInfo": {
        "version": "1.0",
        "timestamp": 1717408171081,
        "provider": "Liv-ex"
    }
}
```

Sample XML Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<root>
    <Status>OK</Status>
    <HttpCode>200</HttpCode>
    <Message>Request completed successfully</Message>
    <InternalErrorCode>R001</InternalErrorCode>
    <ApiInfo>
```

```
        <Version>1.0</Version>
        <Timestamp>2024-06-03T09:42:31.981Z</Timestamp>
        <Provider>Liv-ex</Provider>
    </ApiInfo>
    <pageInfo>
        <totalResults>5</totalResults>
        <limit>10000</limit>
        <offset>1</offset>
    </pageInfo>
    <wineData>
        <listID>757d15ed-56b7-4dc2-9817-9b8861cd88df</listID>
        <data>
            <lwin>10177572016</lwin>
            <dataDetail>
                <dataType>drinkingWindow</dataType>
                <metadata1>2024</metadata1>
                <metadata2>2052</metadata2>
            </dataDetail>
        </data>
    </wineData>
</root>
```

Sample XML Failed Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="https://api.liv-ex.com/v1
http://54.154.139.15:8091/schema/v1/services.xsd">
    <Status>Unauthorized</Status>
    <HttpCode>401</HttpCode>
    <Message>Unauthorized</Message>
    <InternalErrorCode/>
    <ApiInfo>
        <Version>2.0</Version>
        <Timestamp>2016-11-17T08:53:20.730Z</Timestamp>
        <Provider>Liv-ex</Provider>
    </ApiInfo>
</Response>
```

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	Request partially completed

6.1 Request validation error codes

Code	**Message**
V018	Mandatory field missing (Please provide a valid lwin or listID)

V058	Invalid / incorrect LWIN: [%s]. Please provide a valid LWIN11, LWIN16 or LWIN18 code.
V174	Invalid / incorrect listID: [%]. Please provide a valid listID value.
V189	Invalid / incorrect dataType: [%s]. Possible values are [%s]

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.