Auction Events API v1

Document Revision 1.0
Date of Issue: 03/10/2023
Date of revision: 03/10/2023

Fred Haselton
Business Analyst

Table of Contents

1. Purpose

To provide the API end point information and examples of the Auction Events API v1. The web service handles the POST method to facilitate the recalling of Auction data from a specified wine.

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.

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 both XML and JSON. The user can provide the contents 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.
- For HTTP users who can only work on GET & POST methods, we provide a Header 'X-HTTP-Method-Override' for DELETE
- Authentication mechanism will be custom based on CLIENT_KEY and CLIENT_SECRET.
- For any PUSH services we require a direct POST URL which should be backed by as service capable of accepting and process XML 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.

Name	Mandatory	Description
CLIENT_KEY	Y	A valid user GUID which will be unique for each merchant
CLIENT_SECRET	Y	Password/Secret for the user CLIENT_KEY
ACCEPT	Y	Accept header is a way for the user 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.

Example Header (JSON)

```
CLIENT_KEY: 12A34BC56-DE7F-89G0-H1J2345K678L
CLIENT_SECRET: dummy_password
ACCEPT: application/json
CONTENT-TYPE: application/json
```

Invalid header (JSON response)

```
{
    "status": "Unauthorized",
    "httpCode": "401",
    "message": "Unauthorized",
    "internalErrorCode": null,
    "apiInfo": {
        "version": "1.0",
        "timestamp": 1550676412005,
        "provider": "Liv-ex"
    }
}
```

Invalid header (XML response)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="https://aby-uat-api.liv-ex.com/v1 https://aby-uat-api.liv-
ex.com/schema/v1/services.xsd">
    <Status>Unauthorized</Status>
    <HttpCode>401</HttpCode>
    <Message>Unauthorized</Message>
    <InternalErrorCode xsi:nil="true"/>
    <ApiInfo>
        <Version>1.0</Version>
        <Timestamp>2024-02-20T15:28:48.623Z</Timestamp>
        <Provider>Liv-ex</Provider>
    </ApiInfo>

</Response>
```

5. API Listing

5.1 View the line data of a list – POST method

Description

This service will be used to recall auction information for a specified wine, based on the specified LWIN11. A successful POST request will be responded with the auction event data corresponding with this LWIN11.

Base URI

data/v1/auctionEvents

Request parameters

Name	Mandatory	Description
lwin	Y	LWIN11. Only one LWIN permitted per request. **Type:** alphanumeric
format	N Default = all	Multiple values permitted per request. **Type:** alphanumeric array **Values:** 'all', 'cases', 'bottles', 'halves', 'mags', 'large'
location	N Default = all	The global region the auction event(s) took place in. **Type:** alphanumeric array **Values:** 'all', 'asia', 'europe', 'online', 'usa', 'uk'

recordCount	N Default = 5	The maximum number of auction event records that should be returned. **Type:** integer **Values:** 5, 10, 20
currency	Y	Only one value permitted per request **Type:** alphanumeric **Values:** 'gbp', 'eur', 'chf', 'usd', 'hkd', 'jpy', 'sgd', 'gbp/btt', 'eur/btt', 'chf/btt'

Additional information – format definitions

Format	Pack Size	Bottle size, ml
'All'	Any	Any
'Cases'	> 01	00750
'Bottles'	01	00750
'Halves'	Any	00375
'Mags'	Any	01500
'Large'	Any	> 01500

Response parameters

Name	Description
lwin	LWIN11 code requested. **Type:** alphanumeric
auctionDate	Recorded day the auction event took place. **Type:** datetime (epoch if JSON)
auctionHouse	Name of the auction house where the event sale took place. **Type:** alphanumeric
location	Geographic location of auction event. May be a town/city or online depending on nature of auction event and the data provided to Liv-ex. **Type:** alphanumeric
lotNumber	The lot number associated with the specific auction event. **Type:** alphanumeric
packSize	The LWIN pack size description e.g. '06'. **Type:** alphanumeric

bottleSize	The LWIN bottle size description e.g. '00750'. **Type**: alphanumeric
quantity	The unit quantity value associated with the specific auction event. **Type:** integer
price	The unit price value associated with the specific auction event. **Type:** double
currency	The currency type the price value has been calculated as. **Type:** alphanumeric

Sample Request Body

JSON Request

```
{
   "auctionEvents": {
      "lwin": "10287041998",
      "format": ["cases","bottles"],
      "location": ["asia","europe"],
      "recordCount": 5,
      "currency": "gbp"
   }
}
```

XML Request

```
<root>
<auctionEvents>
 <lwin>10287041998</lwin>
 <format>cases</format>
 <format>bottles</format>
 <location>asia</location>
 <location>europe</location>
 <recordCount>5</recordCount>
 <currency>gbp</currency>
</auctionEvents>
</root>
```

Sample Response Body

JSON Response

```
{
   "status": "OK",
   "httpCode": "200",
   "message": "Request completed successfully",
```

```
  "internalErrorCode": "R001",
  "apiInfo": {
    "version": "1.0",
    "timestamp": 1585911767765,
    "provider": "Liv-ex"
  },
  "auctionEventsResponse": {
    "lwin": "10059921999",
    "currency": "GBP",
    "auctionEvents": [
      {
        "date": 1593444778000,
        "auctionHouse": "Christies",
        "location": "Oxford",
        "lotNumber": "321",
        "packSize": "01",
        "bottleSize": "00750",
        "quantity": 2,
        "price": 356.90
      },
      {
        "date": 1593444778000,
        "auctionHouse": "Zachys",
        "location": "Online",
        "lotNumber": "44",
        "packSize": "01",
        "bottleSize": "00750",
        "quantity": 1,
        "price": 377.23
      }
    ],
    "errors": null
  }
}
```


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>2022-07-11T15:37:41.402Z</timestamp>
  <provider>Liv-ex</provider>
 </apiInfo>
 <auctionEventsResponse>
```

```
    <lwin>10059921999</lwin>
    <currency>gbp</currency>
    <auctionEvents>
      <date>2022-07-11T15:37:41.402Z</date>
      <auctionHouse>Christies</auctionHouse>
      <location>Oxford</location>
      <lotNumber>321</lotNumber>
      <packSize>1</packSize>
      <bottleSize>488</bottleSize>
      <quantity>2</quantity>
      <price>356.9</price>
    </auctionEvents>
    <auctionEvents>
      <date>2022-07-11T15:37:41.402Z</date>
      <auctionHouse>Zachys</auctionHouse>
      <location>Online</location>
      <lotNumber>44</lotNumber>
      <packSize>1</packSize>
      <bottleSize>488</bottleSize>
      <quantity>1</quantity>
      <price>377.23</price>
    </auctionEvents>
    <errors/>
  </auctionEventsResponse>
</root>
```

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 [%s].
V061	Invalid / incorrect currency: [%s]. Possible values are 'gbp', 'eur', 'chf', 'usd', 'hkd', 'jpy', 'sgd', 'gbpbtt', 'eurbtt', 'chfbtt', 'gbp/btt', 'eur/btt', 'chf/btt'.

V064	Invalid / incorrect LWIN and vintage: [LWIN] combination.
V114	Invalid / incorrect LWIN: [${v}]. Please provide a valid LWIN11 code.

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.