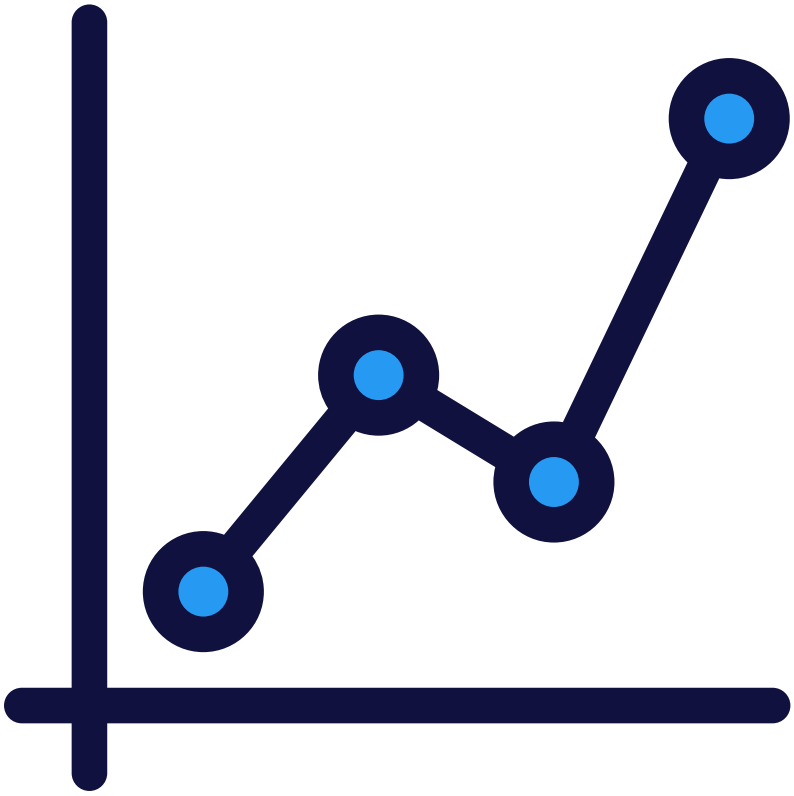
ABB ABILITY KNOWLEDGE MANAGER

LogData WebAPI

v0.62



Contents

Figures

No table of figures entries found.

Introduction

LafargeHolcim needs to export log data from KM for detailed analysis in a separate tool called SeeQ. The exported log data is not permanently stored by that tool and only kept for the duration of the analysis.

We have decided to use a web based interface that accepts requests and returns data in JSON format.

Requirements

The WebAPI must support these features

1. Get primary data (PRI, includes compressed) and also of all secondary period types (15M, 30M, HRS, SHT, DAY, WEK, MTH, YER)
2. Get data of calculated logs (transient logs that are calculated on the fly based on one or more other logs)
3. Get data of LIMS logs
4. Create tag names per SortName and location and periodtype on top of “real” logs
5. Support numeric and text values

The resulting data is a list of data points per tag. The data points are sorted by StartDate. Each data point carries this information:

- StartTime
- EndTime
- Numeric or text value (mutually exclusive)

Excluded Functionality

Only limited metadata of logs is available (folder structure, description, group number, channel number)

WebAPI Description

Base URL

All the web methods supported use the same base URL shown below. Replace localhost with the name or IP address of the KM server if called from remote.

http://localhost/km/extdataaccess/api/

Web methods

The WebAPI supports these calls

1. General calls
    a. GetAvailability()
    Check if WebAPI is available
2. Virtual tag calls
    a. GetTagList([Ptypes, NameFilter, LocationFilter, SortNameFilter, MaxRecordsToReturn])
    Returns the list of available virtual tag names.
    b. GetTagData(TagNames, StartDate, EndDate)
    Returns the data of one or more virtual tags for the requested time range
3. Stop log calls
    a. GetStopLogList([NameFilter, MaxRecordsToReturn])
    Returns the list of available stop log tag names.
    b. GetStopLogData(TagNames, StartDate, EndDate[, StopTypeFilter])
    Returns the stop logs of one or more stop log tags for the requested time range.
    Optionally filtered by the stop type property.
4. LIMS data calls (Laboratory Ixxx Management System)
    a. GetLimsDataBySampleTime(LogNames, StartDate, EndDate)
    Returns LIMS data of one or more logs for the requested time range. The time range refers to the Sample time of the data. This is to query the history of Samples.
    b. GetLimsDataByUpdateTime(LogNames, StartDate, EndDate)
    Returns LIMS data of one or more logs for the requested time range. The time range refers to the LastUpdate time of the data. This is to efficiently query new/updated data.

Authentication

The web service by default uses NTLM authentication only. Engineering can change that to allow other authentication methods if they are in line with the customers security policy.

Protocol

All the calls are to be sent as HTTP POSTs using Content-Type “application/json”. The request body is a JSON that is different for each call and described later along with the call itself.

The HTTP responses then return the resulting JSON in their body.

Date and time format

The ISO-8601 date and time format is used in all the JSON requests and responses.

General features

GetAvailability call

This call is just to verify that the WebAPI is up and running. Because it has no arguments the body can be left empty.

Arguments

None

Response

The response is a JSON that contains just the current time and looks like this:

```
{
    "TimeStamp": "2020-02-10T13:20:24.1016445+01:00",
    "PlantName": "KM",
    "Version": "v1.0.0.0"
}
```

Virtual tag support

Virtual tag id

The id of a virtual log is a string that uniquely identifies it and is immutable. It is required as an argument to identify virtual logs when requesting data.

There is a fixed a naming scheme used to create tag ids, which have a maximum length of 110 characters, but usually are much shorter.

Virtual tag names

Log names of KM logs are just strings, limited to 80 characters.

We define a naming convention to create tag names that are composed of these four components:

- KM log name (80 characters)
- Period type (3 characters)
- SortName name (20 characters)
- Location name (24 characters)

A tag name thus has basically a maximum length of 130 characters, which includes 3 separator characters for the 4 components of the name. However, since we must escape any occurrences of the separator character in the components the name can become longer in such cases.

SortName and location names are optional. If they are not defined (NULL) then they are omitted but the separator is kept ensuring name uniqueness (in case identical SortName and location names exist)

Proposed tag name construction

The separator character is configurable via a parameter and defaults to "|" (vertical bar) character.

The name construction is configurable via a parameter. The default is

`{LogName}{Separator}{SortNameName}{Separator}{LocationName}{Separator}{PeriodType}`

Example 1:

- Log name: "Kiln2_Temp_AVG"
- Period type: "PRI"
- SortName name: "CemSSO"
- Location name: "Kiln2Exit"

Will result in a tag name of "`Kiln2_Temp_AVG|CemSSO|Kiln2Exit|PRI`"

Example 2:

- Log name: "Kiln2_Temp_AVG"
- Period type: "HRS"
- SortName name: NULL
- Location name: NULL

Will result in a tag name of "`Kiln2_Temp_AVG|||HRS`"

Example 3:

- Log name: "Kiln2_Temp_AVG"
- Period type: "HRS"
- SortName name: "CemMSF"
- Location name: NULL

Will result in a tag name of "`Kiln2_Temp_AVG|CemMSF||HRS`"

Example 4:

- Log name: “Kiln3 Speed_LST”
- Period type: “DAY”
- SortName name: NULL
- Location name: “Kiln3 Input”

Will result in a tag name of “`Kiln3 Speed_LST||Kiln3 Input|DAY`”

SortName and location filtering

General behavior

If no SortName name is specified (NULL) only data that has no SortName is included. The same is true for the Location name.

Thus, to get all the data of a KM log regardless of the SortName and Location you must combine the data of multiple tags.

Primary data

For primary data that is compressed there never is SortName and Location information available. This typically applies to logs that contain non-LIMS data, so the tag names for primary non-LIMS logs are created using “*” as the SortName and the location names.

Tag name example for non-LIMS log:
PRI: “`Kiln2_Temp_AVG|*||PRI`”

For primary LIMS data (logclass “L”) the SortName information is available and the tag names do include it.
Tag name example for LIMS log:
PRI: “`CEMSSO_AVG|*|Dry|PRI`”

GetTagList call

Returns the list of tag names as described above. Because it needs no arguments the body can be left empty.

Arguments

One can pass an optional JSON in the POST body to filter the list using the fields below

- PTypes (required)
  List of KM period type strings to include (PRI, 15M, 30M, HRS, SHT, DAY, WEK, MTH, YER).
  If empty all period types are included
- Log name filter (optional)
  Case sensitive string that supports “*” and “?” wildcards.
- Location filter (optional)
  Case sensitive string that supports “*” and “?” wildcards.
  An empty string filter returns only tags that have no location information.
- SortName filter (optional)
  Case sensitive string that supports “*” and “?” wildcards.
  An empty string filter returns only tags that have no SortName information.
- Maximum number of rows to return (optional)
  Positive number of 1 or higher. Lower numbers are ignored.

Example JSON for HTTP POST:

```
{
    "Ptypes": ["PRI", "DAY" ],
    "NameFilter":"*AVG*",
    "LocationFilter":"*n*",
    "SortNameFilter":"*",
    "MaxRecordsToReturn": 1000
}
```

Example minimum JSON for HTTP POST:

```
{
    "Ptypes": []
}
```

Response

The response is a JSON that looks like the one below. If required the properties can be enabled or disabled by engineering .

- TagId
  Unique identifier of the virtual log, required (string)
- Name
  Tag name, required (string)
- KmPath
  The folder structure where the tag is located in KM (string, enabled by default).
  Configuration key `IncludeTagKmPathInResponse`
- Description
  The description of the tag given by the user (string, enabled by default, not included if empty)
  Configuration key `IncludeTagDescriptionInResponse`
- GroupNr
  The GroupNr of the tag. Not available on all tags (integer, enabled by default)
  Configuration key `IncludeTagGroupNrInResponse`
- ChannelNr
  The ChannelNr of the tag. Not available on all tags (integer, enabled by default)
  Configuration key `IncludeTagChannelNrInResponse`
- ChannelNr
  The ChannelNr of the tag. Not available on all tags (integer, enabled by default)
  Configuration key `IncludeTagChannelNrInResponse`

```
{
    "TimeStamp": "2020-04-20T09:51:07.3570398+02:00",
    "PlantName": "KM",
    "Tags": [
        {
            "Name": "F_AVG|AFSN Animal Meal|Animal meal comp M|HRS",
            "GroupNr": 0,
            "ChannelNr": 11,
            "Description": "LIMS Signal",
            "KmPath": "\\Logs\\LIMS\\Fuel"
        },
        {
            "Name": "FWH 104_AVG|Überwachung|XRD|HRS",
            "GroupNr": 0,
            "ChannelNr": 3,
            "KmPath": "\\Logs\\Monitoring"
        }
    ]
}
```

GetTagData call

This returns the actual data of one or more logs for the given time range.

Arguments

This requires a JSON in the POST body that specifies

- An array of tag ids (at least one)
- The start date of the requested time range.
- The end date of the requested time range

Example JSON for HTTP POST:

```
{
    "TagIds": [ "12345||KilnInput|DAY", "23456|CemSSO||DAY" ],
    "StartDate": "2020-02-13T17:56:50+01:00",
    "EndDate": "2020-02-13T18:59:00+01:00"
}
```

Response

The response is a JSON that looks like this:

```
[
  {
    "TagId": "12345||KilnInput|DAY",
    "TagName": "Kiln2_Temp_AVG||KilnInput|DAY",
    "DataPoints": [
      {
        "StartTime": "2019-12-04T13:56:00+01:00",
        "EndTime": "2019-12-04T13:57:00+01:00",
        "value": 80.0
      },
      {
        "StartTime": "2019-12-04T13:57:00+01:00",
        "EndTime": "2019-12-04T13:58:00+01:00",
        "value": -66.666656494140625
      },
      {
        "StartTime": "2019-12-04T13:58:00+01:00",
        "EndTime": "2019-12-04T13:59:00+01:00",
        "value": -13.333330154418945
      }
    ]
  },
  {
    "TagId": "23456|CemSSO||DAY",
    "TagName": "Silo_23_Level_LST|CemSSO||DAY",
    "DataPoints": [
      {
        "StartTime": "2019-12-04T13:56:00+01:00",
        "EndTime": "2019-12-04T13:57:00+01:00",
        "TextValue": “High”
      },
      {
        "StartTime": "2019-12-04T13:57:00+01:00",
        "EndTime": "2019-12-04T13:58:00+01:00",
        "TextValue": “High”
      },
      {
        "StartTime": "2019-12-04T13:58:00+01:00",
        "EndTime": "2019-12-04T13:59:00+01:00",
        "TextValue": “Critical”
      }
    ]
  }
]
```

Stop log support

GetStopLogList call

Returns the list of stop log tag names very similar to the GetTagList call. Because it needs no arguments the body can be left empty.

Optional arguments

One can pass an optional JSON in the POST body to filter the list using the fields below

- Log name filter
  Case sensitive string that supports “*” and “?” wildcards.
- Maximum number of rows to return
  Positive number of 1 or higher. Lower numbers are ignored.

Example optional JSON for HTTP POST:

```
{
    "NameFilter":"*Temp*",
    "MaxRecordsToReturn": 1000
}
```

Response

The response is a JSON that looks like the one below. If required the properties can be enabled or disabled by engineering .

- TagId
  Unique identifier of the virtual log, required (string)
- Name
  Stop log tag name, required (string)
- KmPath
  The folder structure where the tag is located in KM (string, enabled by default).
  Configuration key `IncludeTagKmPathInResponse`
- Description
  The description of the tag given by the user (string, enabled by default, not included if empty)
  Configuration key `IncludeTagDescriptionInResponse`
- GroupNr
  The GroupNr of the tag. Not available on all tags (integer, enabled by default)
  Configuration key `IncludeTagGroupNrInResponse`
- ChannelNr
  The ChannelNr of the tag. Not available on all tags (integer, enabled by default)
  Configuration key `IncludeTagChannelNrInResponse`

```
{
    "TimeStamp": "2020-11-25T12:15:52.9195178+01:00",
    "PlantName": "KM",
    "Tags": [
        {
            "Name": "213-002.00:RUN_RHR",
            "TagID": 878,
            "GroupNr": 6,
            "ChannelNr": 6,
            "KmPath": "\\Logs\\2.0 PIMS\\200 Raw Mat. Prep."
        },
        {
            "Name": "214-001.00:RP_RHR",
            "TagID": 40252,
            "GroupNr": 6,
            "ChannelNr": 6,
            "KmPath": "\\Logs\\2.0 PIMS\\200 Raw Mat. Prep."
```

```
        },
        {
            "Name": "331-2H1.C1:RP_RHR",
            "TagID": 40252,
            "GroupNr": 6,
            "ChannelNr": 6,
            "KmPath": "\\Logs\\2.0 PIMS\\300 Raw Mill\\331"
        }
}
```


GetStopLogData call

This returns the actual data of one or more stop logs for the given time range.

Arguments

This requires a JSON in the POST body that specifies

- An array of tag ids (at least one)
- The start date of the requested time range.
- The end date of the requested time range
- Stop type filter (optional)
  Case sensitive string that supports “*” and “?” wildcards.
- Minimum stop duration
  Minimum duration of the stop (inclusive)
- Maximum stop duration
  Maximum duration of the stop (inclusive)

Example JSON for HTTP POST:

```
{
    "TagIds": [ 878, 40252, 882 ],
    "StartDate": "2014-05-10T17:56:50+01:00",
    "EndDate": "2014-05-13T18:59:00+01:00",
    "StopTypeFilter": "*",
    "MinStopDuration": "0.01:00:00",
    "MaxStopDuration": "7.00:00:00"
}
```

Response

The response is a JSON that looks as shown below. The stops per tag are sorted by their StartTime.


```
[
    {
        "TagName": "562-002.00:RP",
        "TagId": 878,
        "Stops": [
            {
                "StartTime": "2014-05-10T13:35:00+02:00",
                "EndTime": "2014-05-11T09:17:00+02:00",
                "DurationInDays": 0.8208333333333333,
                "DurationInHours": 19.7,
                "StopType": "S",
                "MajorCause": "Electrical",
                "ReasonText": "2040 Out of operational limits",
                "Explanation": "Probleme mit 532-DB1+2",
                "UserName": "EA\\mfausch",
                "Shift": "_1",
                "AuxCause1": "YYN",
                "ErpStatus": 2,
                "D1L1Cause": "UV.532",
                "D1L2Cause": "UV.532-DB1",
                "D3L1Cause": "UV.562"
            },
            {
```

```
                "StartTime": "2014-05-12T04:00:00+02:00",
                "EndTime": "2014-05-14T19:18:00+02:00",
                "DurationInDays": 2.6374999999999997,
                "DurationInHours": 63.3,
                "StopType": "S",
                "MajorCause": "Electrical",
                "ReasonText": "2070 Planned Maintenance (2 x per month/year)",
                "Explanation": "Probleme mit 532-DB1 und DB2 --> Kontrolle durch Hasler",
                "UserName": "EA\\mfausch",
                "Shift": " _3",
                "AuxCause1": "NNN",
                "ErpStatus": 2,
                "D1L1Cause": "UV.532",
                "D1L2Cause": "UV.532-DB1",
                "D3L1Cause": "UV.562"
            }
        ]
    },
    {
        "TagName": "214-001.00:RP_RHR",
        "TagId": 40252,
        "Stops": [
            {
                "StartTime": "2013-12-13T07:51:00+01:00",
                "EndTime": "2014-10-30T14:16:00+01:00",
                "DurationInDays": 321.26736111111109,
                "DurationInHours": 7710.4166666666661,
                "StopType": "S",
                "MajorCause": "Mechanical",
                "ReasonText": "2070 Planned Maintenance (Prev/Shutdown)",
                "Explanation": "Versetzen des Brechers",
                "UserName": "EA\\mfausch",
                "Shift": " _1",
                "AuxCause1": "YYN",
                "ErpStatus": 2,
                "D1L1Cause": "UV.214",
                "D1L2Cause": "UV.214-HB1",
                "D3L1Cause": "UV.201"
            }
        ]
    },
    {
        "TagName": null,
        "TagId": 882,
        "Stops": []
    }
]
```


Stop log properties

Stop logs have many properties, most are optional and are not sent when they are empty. The table lists all of the supported properties:

Property	Data type	Description
StartTime	DateTime	Begin of stop. Required
EndTime	DateTime	End of stop, optional. if missing the stop is still ongoing
DurationInHours	number	Duration of the stop in hours. Required. If EndTime is missing, the current time is used to determine the duration.
DurationInDays	number	Duration of the stop in days. Required. If EndTime is missing, the current time is used to determine the duration.
StopType	string	Optional
MajorCause	string	Reason (Major Cause) of stop. Optional
ReasonText	string	Detail Reason (Sub-cause) of stop. Optional
ReasonText2	string	Detail Reason level 2 (Sub-sub-cause) of stop. Optional

Property	Data type	Description
Explanation	string	Explanation text for stop. Optional
UserName	string	Name of user who entered the reason information. Optional
Shift	string	Name of the Shift when the stop began. Optional
AuxCause1	string	Auxiliary cause attribute 1. Optional
AuxCause2	string	Auxiliary cause attribute 2. Optional
AuxCause3	string	Auxiliary cause attribute 3. Optional
SortName	string	SortName related to stop. Optional
ErpStatus	int	Status of Stop in external system. Optional
ExternalId	string	Id of Stop in external system. Optional
D1L1Cause	string	Dimension 1, level 1 cause. Optional
D1L2Cause	string	Dimension 1, level 2 cause. Optional
D1L3Cause	string	Dimension 1, level 3 cause. Optional
D2L1Cause	string	Dimension 2, level 1 cause. Optional
D2L2Cause	string	Dimension 2, level 2 cause. Optional
D2L3Cause	string	Dimension 2, level 3 cause. Optional
D3L1Cause	string	Dimension 3, level 1 cause. Optional
D3L2Cause	string	Dimension 3, level 2 cause. Optional
D3L3Cause	string	Dimension 3, level 3 cause. Optional
AuxText1	string	Auxiliary text attribute 1. Optional
AuxText2	string	Auxiliary text attribute 2. Optional

LIMS data support

This section describes accessing LIMS data, which includes many additional properties from the SampleList and SampleWorklist tables. The response of all these calls has the same format. Engineering can enable or disable any of the properties returned to reduce the size of the response.

Below is a list of the properties. They are not included in the JSON when they do not have a value.

Property name	Description	Included by default
SampleId	SampleId string	Yes
SampleTime	When sample was taken	Yes
MeasureTime	When sample was measured	Yes
value	Value of the measurement	Yes
SortName	Material name	Yes
SampleLoc	Location	Yes
SampleDescription	Description provided for the sample	No
SampleOperator	Operator who took the sample	No
SampleApprovedBy	Operator who approved the sample	No
SampleApprovalTime	When the sample was approved	No
SampleRegistrationTime	When the sample was registered	No
SamplePlannedCompleteTime	Planned time when the sample should be taken	No
MeasureWorkplace	Workplace where measurement was made	No
MeasureEquipment	Equipment used for the measurement	No
MeasureApprovedBy	Operator who approved the measurement	No
MeasureWorkTime	Operator who approved the measurement	No
MeasureApprovalTime	When the measurement was approved	No
MeasureOperator	Operator who did the measurement	No
MeasurePlannedTime	When the measurement was planned	No
LastUpdate	When the record was last modified	No

GetLimsDataBySampleTime call

This returns the LIMS data of one or more logs for the given SampleDate time range.

Arguments

This requires a JSON in the POST body that specifies at least

- An array of KM log names (at least one)
- The start date of the requested time range.
- The end date of the requested time range

There are optional string filters, that support the usual wildcards:

- SortNameFilter

- SampleLocFilter
- SampleDescriptionFilter
- SampleOperatorFilter
- SampleApprovedByFilter
- MeasureWorkplaceFilter
- MeasureEquipmentFilter
- MeasureOperatorFilter
- MeasureApprovedByFilter

There is an optional integer filter for the MeasureAnalysisStatus

- MeasureAnalysisStatusFilter

Example JSON for HTTP POST:

```
{
    "LogNames": [ "LR_Sandtemperatur_1", "LR_Sandtemperatur_2" ],
    "StartDate": "2012-06-27T01:56:50+01:00",
    "EndDate": "2012-07-07T18:59:00+01:00",
    "SampleLocFilter": "*O*",
    "SampleOperatorFilter": "SMITH",
    "MeasureAnalysisStatusFilter": "5"
}
```

Response

The response is a JSON that looks like this:

```
[
  {
    "LogName": "LR_Sandtemperatur_1",
    "DataPoints": [
      {
        "SampleId": "UP1211889",
        "SampleTime": "2012-06-27T05:00:00Z",
        "MeasureTime": "2012-07-09T05:48:42Z",
        "value": 20.0,
        "SortName": "MODERO 3A",
        "SampleLoc": "Sackware UV"
      },
      {
        "SampleId": "UP1211887",
        "SampleTime": "2012-06-27T05:00:00Z",
        "MeasureTime": "2012-07-09T05:47:45Z",
        "value": 20.0,
        "SortName": "GOTTARDO 310",
        "SampleLoc": "Versand Silo 591-352"
      },
      {
        "SampleId": "UP1211893",
        "SampleTime": "2012-06-27T05:00:00Z",
        "MeasureTime": "2012-07-12T13:12:47Z",
        "value": 20.0,
        "SortName": "PROVATO 3R",
        "SampleLoc": "Versand Silo 594-354"
      }
    ]
  },
  {
    "LogName": "LR_Sandtemperatur_2",
    "DataPoints": [
      {
        "SampleId": "UP1211889",
        "SampleTime": "2012-06-27T05:00:00Z",
        "MeasureTime": "2012-07-09T05:48:42Z",
        "value": 20.0,
        "SortName": "MODERO 3A",
```

```
        "SampleLoc": "Sackware UV"
      },
      {
        "SampleId": "UP1211887",
        "SampleTime": "2012-06-27T05:00:00Z",
        "MeasureTime": "2012-07-09T05:47:45Z",
        "value": 20.0,
        "SortName": "GOTTARDO 310",
        "SampleLoc": "Versand Silo 591-352"
      },
      {
        "SampleId": "UP1211893",
        "SampleTime": "2012-06-27T05:00:00Z",
        "MeasureTime": "2012-07-12T13:12:47Z",
        "value": 20.0,
        "SortName": "PROVATO 3R",
        "SampleLoc": "Versand Silo 594-354"
      }
    ]
  }
]
```


GetLimsDataByUpdateTime call

This returns the LIMS data of one or more logs for the given LastUpdate time range.

Arguments

Same as for GetLimsDataBySampleTime call.

Example JSON for HTTP POST:


```
{
    "LogNames": [ "LR_Temp_Mischende_1", "32µm" ],
    "StartDate": "2012-06-27T01:56:50+01:00",
    "EndDate": "2012-07-07T18:59:00+01:00"
}
```


Response

The response is a JSON that looks like this:


```
[
  {
    "LogName": "LR_Temp_Mischende_1",
    "DataPoints": [
      {
        "SampleId": "UP1211784",
        "SampleTime": "2012-06-26T05:00:00Z",
        "MeasureTime": "2012-07-05T11:35:13Z",
        "value": 23.3,
        "SortName": "MODERO 3B",
        "SampleLoc": "Versand Silo 594-355"
      },
      {
        "SampleId": "UP1211786",
        "SampleTime": "2012-06-26T05:00:00Z",
        "MeasureTime": "2012-07-02T06:44:51Z",
        "value": 21.9,
        "SortName": "PROVATO 4R",
        "SampleLoc": "Versand Silo 594-356"
      }
    ]
  },
  {
    "LogName": "32µm",
    "DataPoints": []
  }
]
```

Revision History

Rev.	Page (P) Chapt.(C)	Description	Date Dept. / Init.
00	all	Initial document	2020-02-03 IAPI /RoBar
01	all	Replaced term “virtual log name” with “tag name”	2020-02-07 IAPI / RoBar
02	all	Renamed calls, added optional arguments to GetTagList, added PlantName to response of GetAvailability and GetTagList, added Version to response of GetAvailability	2020-02-10 IAPI / RoBar
03		Fixed some typos and two small errors	2020-02-19 IAPI / RoBar
04	GetTagList	Modified response format to include GroupNr, ChannelNr and Description	2020-04-20 IAPI / RoBar
05	GetTagList	Added KmPath to response JSON	2020-06-29 IAPI / RoBar
0.52	GetTagList	Added bullet list for the possible tag properties	2020-11-23 IAPI / RoBar
0.53	Stop log support GetStopLogList GetStopLogData	Added these chapters as a proposal. Added chapter “Virtual tag support” to group existing functionality	2020-11-25 IAPI / RoBar
0.54	GetStopLogData	Removed property StopDuration, added properties DurationInHours and DurationInDays instead.	2020-11-25 IAPI / RoBar
0.55	GetTagList GetStopLogList	Updated default behaviour for sending Description property	2020-12-02 IAPI / RoBar
0.60	all	Added LIMS data chapter	2021-02-16 PAPI /RoBar
0.61	Introduction	Added base URL and Authentication	2021-02-25 PAPI /RoBar
0.62	GetTagList GetTagData GetStopLogList GetStopLogData	Switched to TagIds instead of TagNames to avoid issues when logs are renamed.	2021-03-15 PAPI/RoBar