Highlight API V2.4

July 2018

Contents

1. Introduction 2
2. Objective of the document 2
3. Requirements 3
    - Watch Management (Live Bearer/Device) 3
    - Authentication/Authorization/Audit (AAA) 3
4. Access and Security 3
    - Access to your Highlight Service 3
    - Authorization Header 4
    - Obtaining an Authorization Header 4
5. Authorization API 4
6. Watch API 5
    - Find a watch 5
    - Query watch details 6
    - Update Watch details 6
    - Create a watch 6
7. Data Transfer Objects 7
    - Watch Record DTO 7
    - Update Watch DTO 7
    - Create Watch DTO 8
    - Credentials DTO 8
    - Access Token DTO 8
    - Error DTO 8

1. Introduction

The importance of easing the day to day operational tasks that involve working with Highlight is key to the continued success of the service.

Automation via an API will allow partners to integrate Highlight into their own service layer, providing the ability to add new watches in support of the customers' circuit and device onboarding processes.

In the domain of Highlight, a device is typically physical hardware, and a watch is a node in the Highlight tree representing data for that device. Folders and locations make up the structure of the Highlight tree. This API is primarily focused upon the management of watches, however folders and locations may be created during the provision of a new watch.

2. Objective of the document

To provide our partners with an overview of Highlight's Watch Creation and Management API.

3. Requirements

To support a range of partners, and their common requirement for automated provisioning, the following initial requirements have been established:

Watch Management (Live Bearer/Device)

The scope of the API covers

- Creation of new watches
- The ability to query the state of a watch
- The ability to change the device IP address
- The ability to change device Interface
- The ability to enable or disable alerting on a watch
- The ability to enable or disable a watch for billing management
- The ability to query or find a watch based on a description filter
- The ability to enable or disable the discovery of VLANs

Authentication/Authorization/Audit (AAA)

Authentication is via an Access Token, where the token is provided via a mechanism leveraging Highlight as an identity provider.

Authorization requires that (a) use of the API is enabled for the user’s folder and (b) the user also has the required permissions to allow the API functionality.

Audit includes logging the actions performed on behalf of the user account through use of the API.

4. Access and Security

Access to your Highlight Service

Access to your Highlight service via the API is granted by use of an Access Token. Every Token is unique and is associated with a user account in Highlight.

Authorization Header

In addition to any other request headers, when making a request to the Watch API service endpoints you will need to include a valid Access Token.

This is achieved by use of the Authorization header, and the HLA authorization scheme.

REQUEST HEADER: Authorization: HLA <**Access Token**>

Obtaining an Authorization Header

The Authorization API provides a mechanism to acquire Access Tokens. The endpoint returns an Access Token when provided with

a) The username of a Highlight user with the correct permissions
b) The Password for this Highlight user
c) The URL of the Highlight Service where this user would login

5. Authorization API

Summary: Return an Access Token for the provided user credentials. The API must have been enabled at the user folder level in Highlight. The user must have the required permissions: **View pending list, Edit watches**; **Approve new watches**; **Activate new watches; Manage folders/locations**

Method	URI
POST	https://api.highlighter.net/auth/v1/account/authorize

URI PARAMETERS: None

REQUEST HEADERS: **content-type: application/json**

REQUEST BODY: see Credentials DTO

STATUS CODES:
200 (OK)
400 (Bad Request) Invalid user or highlight details
403 (Forbidden) User is not allowed to use the API

RESPONSE BODY: see Access Token DTO

6. Watch API

Find a watch

Summary: Return the first watch resource, that is either a partial or exact match for the search term.

Method	URI
`GET`	`https://api.highlighter.net/api/v1/watches/?search=<search string>&exactMatch=<true|false>`

URI PARAMETERS:

search find a (first matching) watch by matching the given search string against the description or name field

exactMatch if this is set to true then only exact matches for the search term will be returned.

REQUEST HEADERS: **content-type: application/json** also, see Authorization Header

REQUEST BODY: None

STATUS CODES: 200 (OK)
400 (Bad Request)
401 (Unauthorized)

RESPONSE BODY:

Code	Body	Notes:
200	Watch Record DTO	With successful search
200	"no results found"	No match
200	"multiple results found"	More than one match and exact=true
400	ErrorDTO	Validation Issue
401	ErrorDTO	Invalid Token

Query watch details

Summary: Return a watch resource.

Method	URI
`GET`	`https://api.highlighter.net/api/v1/watches/<watchId>`

URI PARAMETERS: ***watchId*** the id of the watch record to be returned

REQUEST HEADERS: **content-type: application/json** also, see Authorization Header

REQUEST BODY: None

STATUS CODES: A successful operation returns status code 200 (OK).

RESPONSE BODY: see Watch Record DTO

Update Watch details

Summary: Update the properties of an existing watch

Method	URI
`PATCH`	`https://api.highlighter.net/api/v1/watches/<watchId>`

URI PARAMETERS: ***watchId*** the id of the watch record to be returned

REQUEST HEADERS: **content-type: application/json** also, see Authorization Header

REQUEST BODY: see Update Watch DTO

STATUS CODES: A successful operation returns status code 200 (OK).

RESPONSE BODY: see Watch Record DTO

Create a watch

Summary: Creates a watch with the given properties, currently this only supports SNMPv2

Method	URI
POST	https://api.highlighter.net/api/v1/watches/

URI Parameters: None

Request Headers: **content-type: application/json** also, see Authorization Header

Request Body: see Create Watch DTO

Status Codes: A successful operation returns status code 200 (OK).

Response Body: see Watch Record DTO

7. Data Transfer Objects

Watch Record DTO

Summary: Return details of a given watch.

```
{
        "watchId": 12345,
        "enabled": "true",
        "deviceAddress": "100.10.1.2",
        "alerting": "false",
        "selectedInterface": "GigabitEthernet0/1",
        "discoverVlans": "false",
        "description": "deviceId:12345",
        "name": "watchName"
}
```

Update Watch DTO

Summary: Update the details of a given watch

```
{
        "enabled": "true", (true|false)
        "deviceAddress": "100.10.1.4",
        "alerting": " true ",
        "selectedInterface": “GigabitEthernet0/1”,
        "discoverVlans": "false"
}
```

Create Watch DTO

Summary: Create a new device watch

```
{
    "enabled": "true", (true|false)
    "deviceAddress": "100.10.1.4",
    "alerting": "false",
    "selectedInterface": "GigabitEthernet0/1",
    "discoverVlans": "false",
    "description": "my watch description",
    "name": "watch name",
    "poRef": "My Ref",
    "productType": "my product type description",
    "type": "Adsl", (Adsl|DedicatedAccess|Dmax|MplsClass|Sdsl)
    "bandwidth1": 2000000,
    "bandwidth2": 40000000,
    "snmpReadCommunityString": "highlightREAD",
    "locationPath": "customerXYZ/Europe/UK/London"
}
```

Credentials DTO

Summary: Authentication details use to obtain an access token

```
{
       "url": "https://xxx.highlighter.net",
       "username": "user@domain.com",
       "password": "userpassword"
}
```

Access Token DTO

Summary: Details of an Access Token

```
{
       "AccessToken": <ACCESS TOKEN>,
       "TokenType": "Bearer",
       "ExpiresIn": <time in ms>
}
```

Error DTO

Summary: Details of an Access Token

```
{
       "Message": <type of issue>,
       "Error": <detail>
}
```