Logiciel « web Display Manager »

webDM

Manuel d'utilisation de l'API Web
Fonctionnement général

Version document	Version logiciel	Date	Objet
V1.0	0.25.0	24/06/2021	Création du document

Cette notice couvre l'utilisation de l'API Web du logiciel web Display Manager Elle ne constitue pas une notice d'installation du module Amandyn 4 ni une notice de configuration et d'utilisation du logiciel embarqué

MICHAT ÉLECTRONIQUE
8 rue Anders Celsius - ZA Les Revols
26540 Mours Saint Eusèbe - FRANCE
Tel: +33 (0)4 75 71 74 23

Sommaire

1. DESCRIPTION DE L'API

Le logiciel web Display Manager fournit une API Web permettant l'accès en lecture et en écriture à l'ensemble de la configuration du système et du logiciel embarqué. Cette API est utilisée par l'interface Web pour communiquer avec le logiciel embarqué. Elle peut également être utilisée par un système tiers (comme une GTC ou une GTB) pour superviser ou piloter le logiciel web Display Manager.

L'API Web est accessible via le protocole HTTP sur le port 80. Le chemin de base des ressources mises à disposition est **/api**.

Une documentation complète de l'API Web avec toutes ses fonctionnalités est disponible au format HTML depuis l'interface Web du logiciel, onglet « Configuration système », section « Informations et documentation API ». Cette documentation est générée à partir de sources RAML (https://raml.org/) disponibles au même endroit sous forme d'une archive zip.

Un exemple d'utilisation est également disponible avec son code source en JavaScript.

Ce document apporte des précisions quant au fonctionnement général de l'API ainsi que l'ouverture d'une session utilisateur préalable à toute action et la réception des événements générés par le logiciel embarqué. D'autres documents apporteront des précisions concernant certaines fonctionnalités pouvant intéresser les éditeurs de GTC comme la gestion des fonctionnalités de comptage.

2. UTILISATEURS ET SESSIONS

Le logiciel web Display Manager est multi-utilisateur.

2.1. Caractéristiques des utilisateurs

Chaque utilisateur configuré dispose d'un nom et d'un mot de passe lui permettant de s'authentifier et d'un niveau d'accès permettant de définir les actions qu'il peut effectuer ou non.

Les niveaux d'accès sont au nombre de quatre :

- **Visualisation** : l'utilisateur peut accéder aux informations fournies par l'API en lecture uniquement. Certains paramètres critiques (comme des mots de passe permettant de s'authentifier auprès de systèmes tiers) et les statistiques de comptage ne sont pas accessibles.
- **Agent** : l'utilisateur à accès aux mêmes informations qu'au niveau « Visualisation » en lecture et peut également accéder aux statistiques de comptage. Il peut aussi réaliser certaines actions comme ajuster les compteurs locaux, modifier le mode de fonctionnement des éléments des parkings (parkings, zones et totalisateurs) et des afficheurs et piloter les sorties manuelles.
- **Chef de parc** : l'utilisateur à accès aux mêmes informations qu'au niveau « Agent » en lecture. Il peut également réaliser certaines actions supplémentaires comme la gestion des cycles horaires et des messages, la configuration et l'effacement des statistiques de comptage, la mise à jour du *firmware* et la gestion des paramètres (sauvegarde, restauration et réinitialisation).
- **Installateur** : l'utilisateur dispose d'un accès complet à l'ensemble des informations fournies par l'API en lecture et en écriture et peut réaliser n'importe quelle action.

2.2. Sessions et jetons

La grande majorité des fonctionnalités de l'API nécessite l'ouverture d'une session utilisateur qui servira à authentifier les requêtes réalisées. Toutefois quelques informations sont accessibles sans session. Il s'agit d'informations basiques pouvant être utiles avant l'ouverture d'une session concernant :

- L'application (nom, version, …)
- L'API elle-même (version et chemin d'accès à la documentation)
- Les options d'authentification (nom du module Amandyn 4, langue du système et disponibilité d'un utilisateur par défaut ne nécessitant pas de mot de passe)

Une session est ouverte à l'aide du nom et du mot de passe d'un utilisateur. Elle est associée à un identifiant permanent unique et un jeton ayant une durée de validité limitée. L'identifiant de session et le jeton sont tous deux nécessaires pour authentifier une requête.

Lors de l'ouverture d'une session un premier jeton est automatiquement généré. La durée de validité de ce jeton et de 30 minutes. Une fois ce temps passé, il ne peut plus être utilisé pour authentifier une requête et il faut en générer un nouveau à l'aide du nom et du mot de passe de l'utilisateur. Ce nouveau jeton aura, lui-aussi, une durée de validité de 30 minutes. Il est tout à fait possible de générer un nouveau jeton avant l'expiration du jeton courant. Dans ce cas, le jeton précédent perdra toute validité dès que le nouveau aura été généré.

Contrairement à la requête d'ouverture de session, la requête de renouvellement du jeton doit être authentifiée à l'aide de l'identifiant de session et du jeton courant. Si le jeton courant est expiré il peut toujours être utilisé pour authentifier cette seule requête.

La session elle-même n'a pas de limite de durée. Elle peut toutefois être fermée à la requête de l'utilisateur ou automatiquement cinq minutes après l'expiration du jeton si aucun nouveau jeton n'a été généré.

2.3. Authentification d'une requête

Une fois la session ouverte, pour authentifier une requête, il faut ajouter un en-tête « Authorization » (https://developer.mozilla.org/fr/docs/Web/HTTP/Headers/Authorization) avec comme type « SESSION-TOKEN » et comme identifiant l'identifiant de session concaténé avec le caractère ':' et le jeton.

Par exemple si l'identifiant de session est « 01234567-89ab-cdef-0123-456789abcdef » et le jeton « fedcba98-7654-3210-fedc-ba9876543210 », l'en-tête à jouter sera :

```
Authorization: SESSION-TOKEN 01234567-89ab-cdef-0123-456789abcdef:7654-3210-fedc-ba9876543210
```

Si un problème d'authentification empêche la requête d'être exécutée, la réponse contiendra le code de statut HTTP 401. Cela peut se produire dans les cas suivants :

- L'authentification est requise mais l'en-tête « Authorization » n'est pas présent dans la requête
- Le type d'authentification n'est pas « SESSION-TOKEN »
- Le format de l'identifiant n'est pas valide
- La session utilisée n'est pas valide
- Le jeton utilisé n'est pas valide pour la session ou est expiré

Si le niveau d'accès associé à la session utilisé pour authentifier une requête n'est pas suffisant pour exécuter cette requête, la réponse contiendra le code de statut HTTP 403.

2.4. Hachage du mot de passe

Pour des raisons de sécurité, le mot de passe utilisé pour ouvrir une session n'est jamais transmis en clair. Il est haché à l'aide de la formule :

```
SHA256(<nomUtilisateur>:<motDePasse>)
```

Avec :

- **nomUtilisateur** : le nom de l'utilisateur encodé en UTF-8
- **motDePasse** : le mot de passe de l'utilisateur encodé en UTF-8

Le résultat est ensuite encodé sous forme d'une chaîne de caractères en base 16 et devra donc être conforme à l'expression régulière :

```
^[0-9a-fA-F]{64}$
```

Par exemple, pour un utilisateur nommé « utilisateur » ayant pour mot de passe « 123456 », le hachage du mot de passe consistera à hacher la chaîne de caractères "utilisateur:123456" avec la fonction de hachage SHA256

(https://fr.wikipedia.org/wiki/SHA-2#SHA-256) et d'encoder ensuite le résultat sous forme d'une chaîne de caractères en base 16. Le résultat sera donc :

```
18d3cef00572c1b8855f72e00dff407f291df157aac5bf6ce5b04f83af304501
```

Cette technique ne permet pas d'empêcher d'intercepter les identifiants utilisés pour l'ouverture de la session mais permet d'éviter d'utiliser des identifiants interceptés pour accéder à d'autres services dans le cas où l'utilisateur utiliserait le même mot de passe.

Il faut noter que le mot de passe est haché une seconde fois avant d'être stocké dans le module Amandyn 4 afin d'éviter toute fuite de mot de passe en cas d'accès à la base de données.

3. ÉVENEMENTS ENVOYES PAR L'API

Les informations fournies par l'API Web du logiciel web Display Manager peuvent changer lorsqu'un utilisateur modifie la configuration ou que le statut d'un élément configuré dans l'application est mis à jour (par exemple suite au changement d'état d'un module physique associé).

Afin de permettre à tous les utilisateurs d'être informés de ces changements le plus rapidement possible et sans avoir à relire ces informations à intervalle régulier, l'API Web propose un système permettant de recevoir des événements lors de chacun de ces changements.

3.1. Génération et envoi des événements

Un événement est généré lors de chaque changement d'une donnée (paramètre ou statut) de l'application. Cet événement est ensuite transmis pour chaque session ouverte à l'exception des sessions ne disposant pas d'un niveau d'accès suffisant pour accéder à la donnée modifiée.

Pour recevoir les événements liés à sa session, le client doit lancer une requête en *long polling*. Cette requête est initiée par le client de façon normale, mais l'application ne répondra qu'au moment où un ou plusieurs évènements seront générés, gardant la requête ouverte en attendant. Il ne peut y avoir qu'une seule requête de réception d'événements active à la fois pour une même session. Le lancement d'une nouvelle requête par le client provoquera la fermeture de la précédente requête par l'application si elle n'était pas encore terminée.

Si des événements sont générés pour une session alors qu'aucune requête de lecture d'événements n'est active, ils sont mis en attente pour être envoyés lors de la prochaine requête. Si un grand nombre d'événements doivent être mis en attente, seuls les cent premiers seront effectivement envoyés. Les autres seront perdus et un événement spécifique indiquera que des événements ont été perdus.

3.2. Mode d'envoi des événements

Les événements envoyés par l'application sont toujours contenus dans un tableau JSON, permettant ainsi d'envoyer plusieurs événements pour une même requête. Chaque événement est représenté par un objet JSON contenu dans ce tableau.

Il est possible de recevoir les événements sous forme d'un flux. Dans ce cas, chaque événement est envoyé dès qu'il est généré (à condition qu'il y ait une requête de réception ouverte) sans que la requête ne soit ensuite fermée. L'application ne fermera la requête que si la session est fermée, si le jeton associé expire ou si une nouvelle requête de lecture est lancée par le client.

Le client peut également lancer une requête de lecture qui sera fermée par l'application après l'envoi des premiers événements générés. Dans ce cas, les événements sont mis en attente pendant une courte durée de l'ordre d'une demi-seconde permettant d'attendre d'éventuels événements supplémentaires qui pourront être envoyés dans la même requête. Une fois cette temporisation terminée, tous les événements disponibles sont envoyés et la requête est fermée par l'application. La génération de certains événements nécessitant un envoi urgent provoquera l'envoi immédiat des événements disponibles et la fermeture de la requête sans attendre la fin de la temporisation d'attente.

3.3. Format des événements

Chaque événement contiendra son type et la date à laquelle il a été généré sous forme d'un *timestamp* en millisecondes. Des détails complémentaires peuvent également être présents si nécessaire.

Il y a neuf types d'événements :

- **sessionTokenExpired** : indique que le jeton associé à la session utilisateur courante a expiré
- **sessionClosed** : indique que la session utilisateur courante a été fermée
- **powerOff** : indique que le module Amandyn 4 va s'arrêter
- **reboot** : indique que le module Amandyn 4 va redémarrer
- **parameter** : indique qu'un paramètre ou un statut a été modifié
- **ping** : indique que l'état d'un test de *ping* en cours pour la session utilisateur a changé
- **traceroute** : indique que l'état d'un test de *traceroute* en cours pour la session utilisateur a changé
- **firmwareUpdate** : indique que le statut du processus de mise à jour du *firmware* a changé

- **eventsLoss** : indique que des événements ont été perdus entre le dernier événement reçu et cet événement ci

A. Détails des événements de modification de paramètres

Les événements de modification de paramètres contiennent des détails complémentaires:

- le type de paramètre modifié
- l'identifiant du paramètre modifié s'il fait partie d'une collection
- l'action réalisée (ajout, suppression ou modification)
- la valeur du paramètre modifié si elle est demandée dans la requête

B. Détails des événements de *ping*, *traceroute* et mise à jour du *firmware*

Les événements de *ping*, *traceroute* et mise à jour du *firmware* contiennent, en plus du type et du *timestamp*, le statut du processus correspondant.

3.4. Filtrage des événements

Il est possible, pour chaque session, de filtrer les événements envoyés en fonction de leur sujet. La configuration de ce filtrage peut se faire en choisissant de ne recevoir que les événements se rapportant à une liste de sujets (liste blanche dans le cas où seuls un petit nombre de sujets sont pertinents) ou en choisissant d'ignorer les événements se rapportant à une liste de sujets (liste noire dans le cas où la plupart des sujets sont pertinents mais que certains ne le sont pas et polluent le flux d'événements reçus).

Le filtrage des événements est effectué lors de leur génération. De fait si des événements sont mis en attente pour une session et que les règles de filtrage sont ensuite modifiées, les événements en attente ne seront pas affectés par les nouvelles règles de filtrage.

Certains événements ne peuvent pas être exclus dans les requêtes d'événements :

- Les événements d'expiration du jeton
- Les événements de fermeture de session
- Les événements d'arrêt du module Amandyn 4
- Les événements de redémarrage du module Amandyn 4
- Les événements de perte d'événement

Les sujets disponibles sont les suivants :

- **system** : les événements se rapportant aux paramètres système
- **dateTime** : les événements se rapportant aux paramètres de date et d'heure
- **network** : les événements se rapportant aux paramètres réseau
- **serialPorts** : les événements se rapportant aux ports série
- **removableMedias** : les événements se rapportant aux supports amovibles
- **optionalFeatures** : les événements se rapportant à la gestion des fonctionnalités optionnelles
- **storage** : les événements se rapportant aux images et chaînes de caractères stockées
- **authentication** : les événements se rapportant aux paramètres d'authentification
- **users** : les événements se rapportant aux utilisateurs
- **proxy** : les événements se rapportant aux paramètres du proxy
- **gpios** : les événements se rapportant aux entrées/sorties et aux NetIO
- **counters** : les événements se rapportant aux centrales et aux compteurs
- **forcedMessages** : les événements se rapportant aux messages forcés
- **parkingElements** : les événements se rapportant aux parkings, zones et totalisateurs
- **cycles** : les événements se rapportant aux cycles horaires
- **displayGroups** : les événements se rapportant aux groupes d'afficheurs
- **luminosityCells** : les événements se rapportant aux cellules de luminosité
- **luminosityGroups** : les événements se rapportant aux groupes de luminosité
- **displays** : les événements se rapportant aux afficheurs
- **statistics** : les événements se rapportant aux statistiques de comptage

- **modbusServer** : les événements se rapportant au serveur Modbus TCP
- **exports** : les événements se rapportant à l'export des données de comptage
- **mapCounters** : les événements se rapportant aux compteurs de cartes
- **webMaps** : les événements se rapportant aux cartes Web
- **elementsOrder** : les événements se rapportant à l'ordre des éléments d'une collection
- **ping** : les événements se rapportant aux tests de *ping*
- **traceroute** : les événements se rapportant aux tests de *traceroute*
- **firmwareUpdate** : les événements se rapportant à la mise à jour du *firmware*

Il est possible de reconstituer cette liste de sujets en lisant, via l'API, la liste des sujets actuellement exclus et de ceux actuellement inclus dans les requêtes d'événements pour une session donnée.

4. STRUCTURES DE DONNEES UTILISEES

Les données contenues dans le corps des requêtes et des réponses sont structurées au format JSON.

4.1. Informations de base

A. Informations concernant l'application

Il s'agit d'un objet JSON contenant les propriétés suivantes :

Propriété	***Type***	***Description***
name	Chaîne de caractères	Le nom de l'application
version	Chaîne de caractères	La version de l'application au format x.y.z (ex. "1.0.0")
copyrightDate	Chaîne de caractères	La date de copyright de l'application (ex. "2020-2021")
organizationName	Chaîne de caractères	Le nom de l'éditeur de l'application

B. Informations concernant l'API Web

Il s'agit d'un objet JSON contenant les propriétés suivantes :

Propriété	***Type***	***Description***
version	Chaîne de caractères	La version de l'API au format x.y.z (ex. 1.0.0)
htmlDoc	Chaîne de caractères	Le chemin de la documentation HTML de l'API
ramlDescription	Chaîne de caractères	Le chemin de l'archive contenant la description au format RAML de l'API

C. Option d'authentification

Il s'agit d'un objet JSON contenant les propriétés suivantes :

Propriété	***Type***	***Description***
moduleLabel	Chaîne de caractères	Le nom du module Amandyn 4
	Valeur null	Le module Amandyn 4 n'a pas de nom
language	Chaîne de caractères	Le code de la langue du système sur deux caractères (ex. "fr")
defaultUserEnabled	booléen	Indique si l'utilisateur par défaut est activé ou non. S'il est activé, une session peut être ouverte en utilisant le nom d'utilisateur spécifié avec un mot de passe vide. Cette session aura un niveau d'accès « Visualisation » ou « Agent » en fonction de la configuration du logiciel.
defaultUsername	Chaîne de caractères	Le nom d'utilisateur à utiliser pour ouvrir une session sans mot de passe si l'utilisateur par défaut est activé

- *Le champ "defaultUsername" n'est présent que si l'utilisateur par défaut est activé.*

4.2. Gestion des sessions

A. Paramètres d'ouverture de session ou de renouvellement d'un jeton

Il s'agit d'un objet JSON contenant les propriétés suivantes :

Propriété	***Type***	***Description***
username	Chaîne de caractères	Le nom de l'utilisateur conforme à l'expression régulière *^[a-zA-Z0-9\-_]+$*
password	Chaîne de caractères	Le mot de passe de l'utilisateur haché comme indiqué dans la section 2.4 page 4

B. Informations d'une session

Il s'agit d'un objet JSON contenant les propriétés suivantes :

Propriété	***Type***	***Description***
id	Chaîne de caractères	L'identifiant unique de la session
token	Chaîne de caractères	Le dernier jeton généré pour la session
username	Chaîne de caractères	Le nom de l'utilisateur associé la session

4.3. Réception des événements

A. Détails d'un paramètre modifié

Il s'agit d'un objet JSON contenant les propriétés suivantes :

Propriété	***Type***	***Description***
type	Chaîne de caractères	Le type de paramètre modifié
id	Chaîne de caractères	L'identifiant du paramètre modifié s'il appartient à une collection
action	Chaîne de caractères	L'action réalisée sur le paramètre : "modified" si la valeur du paramètre a été modifiée "added" si un élément a été ajouté à une collection "removed" si un élément a été supprimé d'une collection
val	Non défini	La valeur du paramètre modifié si elle est demandée dans la requête

B. Événement

Il s'agit d'une des structures de données suivantes :

B.1. Événement d'expiration du jeton

Il s'agit d'un objet JSON contenant les propriétés suivantes :

Propriété	***Type***	***Description***
type	Chaîne de caractères	Le type de l'événement toujours ("sessionTokenExpired")
timestamp	Nombre entier signé	Le timestamp correspondant à la date de génération de l'événement (nombre de millisecondes écoulées depuis 1970-01-01T00:00:00.000 UTC)

B.2. Événement de fermeture de la session

Il s'agit d'un objet JSON contenant les propriétés suivantes :

Propriété	***Type***	***Description***
type	Chaîne de caractères	Le type de l'événement (toujours "sessionClosed")
timestamp	Nombre entier signé	Le timestamp correspondant à la date de génération de l'événement (nombre de millisecondes écoulées depuis 1970-01-01T00:00:00.000 UTC)

B.3. Événement d'arrêt du module Amandyn 4

Il s'agit d'un objet JSON contenant les propriétés suivantes :

Propriété	***Type***	***Description***
type	Chaîne de caractères	Le type de l'événement (toujours "powerOff")
timestamp	Nombre entier signé	Le timestamp correspondant à la date de génération de l'événement (nombre de millisecondes écoulées depuis 1970-01-01T00:00:00.000 UTC)

B.4. Événement de redémarrage du module Amandyn 4

Il s'agit d'un objet JSON contenant les propriétés suivantes :

Propriété	***Type***	***Description***
type	Chaîne de caractères	Le type de l'événement (toujours "reboot")
timestamp	Nombre entier signé	Le timestamp correspondant à la date de génération de l'événement (nombre de millisecondes écoulées depuis 1970-01-01T00:00:00.000 UTC)

B.5. Événement de modification d'un paramètre ou d'un statut

Il s'agit d'un objet JSON contenant les propriétés suivantes :

Propriété	***Type***	***Description***
type	Chaîne de caractères	Le type de l'événement (toujours "parameter")
timestamp	Nombre entier signé	Le timestamp correspondant à la date de génération de l'événement (nombre de millisecondes écoulées depuis 1970-01-01T00:00:00.000 UTC)
details	Détails du paramètre modifié	Cf. section 4.3.A page 10

B.6. Événement de *ping*

Il s'agit d'un objet JSON contenant les propriétés suivantes :

Propriété	***Type***	***Description***
type	Chaîne de caractères	Le type de l'événement (toujours "ping")
timestamp	Nombre entier signé	Le timestamp correspondant à la date de génération de l'événement (nombre de millisecondes écoulées depuis 1970-01-01T00:00:00.000 UTC)
details	État du test de *ping*	Voir la documentation complète pour plus de détails

B.7. Événement de *traceroute*

Il s'agit d'un objet JSON contenant les propriétés suivantes :

Propriété	***Type***	***Description***
type	Chaîne de caractères	Le type de l'événement (toujours "traceroute")
timestamp	Nombre entier signé	Le timestamp correspondant à la date de génération de l'événement (nombre de millisecondes écoulées depuis 1970-01-01T00:00:00.000 UTC)
details	État du test de *traceroute*	Voir la documentation complète pour plus de détails

B.8. Événement de mise à jour du *firmware*

Il s'agit d'un objet JSON contenant les propriétés suivantes :

Propriété	***Type***	***Description***
type	Chaîne de caractères	Le type de l'événement (toujours "firmwareUpdate")
timestamp	Nombre entier signé	Le timestamp correspondant à la date de génération de l'événement (nombre de millisecondes écoulées depuis 1970-01-01T00:00:00.000 UTC)
details	État du processus de mise à jour du *firmware*	Voir la documentation complète pour plus de détails

B.9. Événement de perte d'événements

Il s'agit d'un objet JSON contenant les propriétés suivantes :

Propriété	***Type***	***Description***
type	Chaîne de caractères	Le type de l'événement (toujours "eventsLoss")
timestamp	Nombre entier signé	Le timestamp correspondant à la date de génération de l'événement (nombre de millisecondes écoulées depuis 1970-01-01T00:00:00.000 UTC)

C. Règles de filtrage des événements

Il s'agit d'un objet JSON contenant les propriétés suivantes :

Propriété	***Type***	***Description***
excludedEvents	Tableau de chaînes de caractères	La liste des sujets exclus des réponses aux requêtes de lecture d'événements
includedEvents	Tableau de chaînes de caractères	La liste des sujets inclus dans les réponses aux requêtes de lecture d'événements

D. Paramètres de configuration des règles de filtrage des événements

Il s'agit d'un objet JSON contenant les propriétés suivantes :

Propriété	***Type***	***Description***
rulesType	Chaîne de caractères	Le type de règle à appliquer : "includeOnly" pour inclure seulement les sujets spécifiés "includeAllBut" pour inclure tous les sujets sauf ceux qui sont spécifiés
events	Tableau de chaînes de caractères	La liste des sujets à inclure ou exclure suivant le type de règle à appliquer

5. REQUETES UTILISEES

5.1. Lecture des informations de base

A. Lecture des informations concernant l'application

GET /applicationInformation

- Authentification requise : Non
- Niveau d'accès requis : N/A

A.1. Requête

Cette requête permet d'accéder à certaines informations concernant l'application.

A.2. Réponse HTTP 200 OK

Cette réponse indique le succès de la requête.

Le corps de la réponse contient les informations concernant l'application (cf. section 4.1.A page 9).

B. Lecture des informations concernant l'API Web

GET /apiInformation

- Authentification requise : Non
- Niveau d'accès requis : N/A

B.1. Requête

Cette requête permet d'accéder aux informations concernant l'API Web.

B.2. Réponse HTTP 200 OK

Cette réponse indique le succès de la requête.

Le corps de la réponse contient les informations concernant l'API Web (cf. section 4.1.B page 9).

C. Lecture des option d'authentification

GET /loginOptions

- Authentification requise : Non
- Niveau d'accès requis : N/A

C.1. Requête

Cette requête permet d'accéder aux options d'authentification.

C.2. Réponse HTTP 200 OK

Cette réponse indique le succès de la requête.

Le corps de la réponse contient les options d'authentification (cf. section 4.1.C page 9).

5.2. Gestion des sessions

A. Ouverture d'une session

POST /sessions

- Authentification requise : Non
- Niveau d'accès requis : N/A

A.1. Requête

Cette requête permet d'ouvrir une nouvelle session à l'aide des identifiants d'un utilisateur.

Le corps de la requête contient les paramètres d'ouverture d'une session (cf. section 4.2.A page 4.2.A).

A.2. Réponse HTTP 201 Created

Cette réponse indique le succès de l'ouverture de la session.

La réponse contient un en-tête « Location » qui indique la base du chemin de la ressource associée à cette session qui devra être utilisé pour les requêtes de gestion de la session :

```
Location /sessions/{session-id}
```

Le corps de la réponse contient les informations de la session avec le premier jeton généré (cf. section 4.2.B page 9).

A.3. Réponse HTTP 400 Bad Request

Cette réponse indique que la requête n'a pas pu être traitée car la syntaxe n'est pas valide.

A.4. Réponse HTTP 401 Unauthorized

Cette réponse indique que la session ne peut être ouverte car le nom d'utilisateur et le mot de passe indiqués dans la requête ne sont pas valides.

A.5. Réponse HTTP 415 Unsupported Media Type

Cette réponse indique que la requête n'a pas pu être traitée car son contenu n'est pas au format JSON. Cette vérification est uniquement basé sur l'en-tête « Content-Type » de la requête. Si cet en-tête est valide mais que le contenu n'est pas une valeur JSON valide, la réponse HTTP 400 sera envoyée (cf. section 5.2.A.3 page 14).

B. Renouvellement du jeton

POST /sessions/{session-id}

- Authentification requise : Oui
- Niveau d'accès requis : La requête doit être authentifiée avec la session pour laquelle on souhaite renouveler le jeton

B.1. Requête

Cette requête permet de générer un nouveau jeton pour une session à l'aide des identifiants de l'utilisateur associé.

Le chemin de la requête contient les champs variables suivants :

Champ	*Type*	*Description*
session-id	Chaîne de caractères	Identifiant unique de la session pour laquelle renouveler le jeton

Le corps de la requête contient les paramètres de renouvellement d'un jeton (cf. section 4.2.A page 4.2.A).

B.2. Réponse HTTP 200 OK

Cette réponse indique le succès du renouvellement du jeton.

Le corps de la réponse contient les informations de la session avec le nouveau jeton (cf. section 4.2.B page 9).

B.3. Réponse HTTP 400 Bad Request

Cette réponse indique que la requête n'a pas pu être traitée car la syntaxe n'est pas valide.

B.4. Réponse HTTP 401 Unauthorized

Cette réponse indique que le jeton ne peut pas être renouvelé car le nom d'utilisateur et le mot de passe indiqués dans la requête ne sont pas valides.

B.5. Réponse HTTP 403 Forbidden

Cette réponse indique que le jeton ne peut pas être renouvelé car la requête n'est pas authentifiée avec la session pour laquelle on souhaite renouveler le jeton.

B.6. Réponse HTTP 404 Not Found

Cette réponse indique que le jeton ne peut pas être renouvelé car la session n'existe pas.

B.7. Réponse HTTP 415 Unsupported Media Type

Cette réponse indique que la requête n'a pas pu être traitée car son contenu n'est pas au format JSON. Cette vérification est uniquement basé sur l'en-tête « Content-Type » de la requête. Si cet en-tête est valide mais que le contenu n'est pas une valeur JSON valide, la réponse http 400 sera envoyée (cf. section 5.2.A.3 page 14).

C. Fermeture d'une session

DELETE **/sessions/{session-id}**

- Authentification requise : Oui
- Niveau d'accès requis : La requête doit être authentifiée avec la session pour laquelle on souhaite renouveler le jeton

C.1. Requête

Cette requête permet fermer une session.

Le chemin de la requête contient les champs variables suivants :

Champ	*Type*	*Description*
session-id	Chaîne de caractères	L'Identifiant unique de la session pour laquelle renouveler le jeton

C.2. Réponse HTTP 204 No Content

Cette réponse indique le succès de la fermeture du jeton.

C.3. Réponse HTTP 401 Unauthorized

Cette réponse indique que la session n'a pas pu être fermée car la requête n'a pas pu être authentifiée (cf. section 2.3 page 4).

C.4. Réponse HTTP 403 Forbidden

Cette réponse indique que la session n'a pas pu être fermée car la requête n'est pas authentifiée avec la session que l'on souhaite fermer.

C.5. Réponse HTTP 404 Not Found

Cette réponse indique que la session n'a pas pu être fermée car elle n'existe pas.

5.3. Réception des événements

A. Lecture des événements

GET `/events`

- Authentification requise : Oui
- Niveau d'accès requis : Visualisation

A.1. Requête

Cette requête permet de lire les événements générés pour la session courante.

La requête peut contenir les paramètres suivants dans l'URL :

Paramètre	*Type*	*Description*
stream	booléen	Indique si les événements doivent envoyés sous forme d'un flux ou non (faux par défaut)
includeValues	booléen	Indique si les valeurs des paramètres et statuts modifiées doivent être incluses dans les détails des événements (faux par défaut)

A.2. Réponse HTTP 200 OK

Cette réponse indique le succès de la requête.

Le corps de la réponse contient les événements générés pour la session courante sous forme d'un tableau d'événements (cf. section 4.3.B page 10).

A.3. Réponse HTTP 401 Unauthorized

Cette réponse indique que la requête n'a pas pu être traitée car la requête n'a pas pu être authentifiée (cf. section 2.3 page 4).

A.4. Réponse HTTP 403 Forbidden

Cette réponse indique que la requête n'a pas pu être traitée car l'utilisateur associé à la session ne dispose pas du niveau d'accès requis (cf. section 2.3 page 4).

A.5. Réponse HTTP 505 HTTP Version Not Supported

Cette réponse indique que la version du protocole HTTP utilisée ne permet pas d'envoyer les événements sous forme d'un flux (requiert HTTP/1.1 au minimum).

B. Lecture des règles de filtrage des événements

GET /events/filters

- Authentification requise : Oui
- Niveau d'accès requis : Visualisation

B.1. Requête

Cette requête permet de lire les règles de filtrage des événements pour la session courante.

B.2. Réponse HTTP 200 OK

Cette réponse indique le succès de la requête.

Le corps de la réponse contient les règles de filtrage des événements pour la session courante (cf. section 4.3.C page 11).

B.3. Réponse HTTP 401 Unauthorized

Cette réponse indique que la requête n'a pas pu être traitée car la requête n'a pas pu être authentifiée (cf. section 2.3 page 4).

B.4. Réponse HTTP 403 Forbidden

Cette réponse indique que la requête n'a pas pu être traitée car l'utilisateur associé à la session ne dispose pas du niveau d'accès requis (cf. section 2.3 page 4).

C. Configuration des règles de filtrage des événements

POST /events/filters

- Authentification requise : Oui
- Niveau d'accès requis : Visualisation

C.1. Requête

Cette requête permet de configurer les règles de filtrage des événements pour la session courante.

Le corps de la requête contient les paramètres de configuration des règles de filtrage des événements (cf. section 0 page 12).

C.2. Réponse HTTP 200 OK

Cette réponse indique le succès de la configuration des règles de filtrage des événements.

Le corps de la réponse contient les nouvelles règles de filtrage des événements pour la session courante (cf. section 4.3.C page 11).

C.3. Réponse HTTP 400 Bad Request

Cette réponse indique que la requête n'a pas pu être traitée car la syntaxe n'est pas valide.

C.4. Réponse HTTP 401 Unauthorized

Cette réponse indique que la requête n'a pas pu être traitée car la requête n'a pas pu être authentifiée (cf. section 2.3 page 4).

C.5. Réponse HTTP 403 Forbidden

Cette réponse indique que la requête n'a pas pu être traitée car l'utilisateur associé à la session ne dispose pas du niveau d'accès requis (cf. section 2.3 page 4).

C.6. Réponse HTTP 415 Unsupported Media Type

Cette réponse indique que la requête n'a pas pu être traitée car son contenu n'est pas au format JSON. Cette vérification est uniquement basé sur l'en-tête « Content-Type » de la requête. Si cet en-tête est valide mais que le contenu n'est pas une valeur JSON valide, la réponse HTTP 400 sera envoyée (cf. section 5.3.C.3 page 18).