|  |
| --- |
| 21.05.2023 |

|  |
| --- |
| HE-Arc – Ingénierie – ISC3il (2022-2023)  JEE/Spring II – cours 3293.1 |
| Saphir Lion & Topaz Lion |
| Guide d’installation (SP) |

|  |  |  |
| --- | --- | --- |
| Lieu de formation | Haute École Arc (Neuchâtel Berne Jura) – HE-Arc ingénierie  Espace de l'Europe 11, 2000 Neuchâtel | |
| Étudiant | Allemann Jonas — ISC3il-a | [jonas.allemann@he-arc.ch](mailto:jonas.allemann@he-arc.ch) GitHub : <https://github.com/Ozurah> |
| Étudiant | Chappuis Sébastien – ISC3il-a | [sebastien.chappuis1@he-arc.ch](mailto:sebastien.chappuis1@he-arc.ch)  GitHub : <https://github.com/seba-chappuis> |
| Enseignant | Chèvre Sébastien Gérard Henri | [sebastien.chevre@he-arc.ch](mailto:sebastien.chevre@he-arc.ch) |
| Source du projet #1 (Semestre d’automne) | <https://github.com/Ozurah-HES/SaphirLion> | |
| Source du projet #2 (Semestre de printemps) | <https://github.com/Ozurah-HES/TopazLion> | |



Abstract

Ce document a pour objectif de guider l’utilisateur souhaitant utiliser l’application « **Saphir Lion** » ou « **Topaz Lion** ». Allant de l’obtention du projet à son exécution sur le poste de travail. Une explication d’utilisation du site est également présente.

Pour plus d’informations sur la structure du projet et la réalisation de celui-ci, veuillez vous référer au rapport de projet. Celui-ci est disponible sur le wiki du projet, à ces 2 adresses (même rapport) :

<https://github.com/Ozurah-HES/SaphirLion/wiki>

*https://github.com/Ozurah-HES/TopazLion/wiki*

Table des matières

[1 Saphir Lion 3](#_Toc135505902)

[1.1 Installation 3](#_Toc135505903)

[1.1.1 Dépôt GIT 3](#_Toc135505904)

[1.1.2 Versions Java nécessaires 3](#_Toc135505905)

[1.1.3 Configurations possibles 3](#_Toc135505906)

[1.1.4 Démarrage du projet 5](#_Toc135505907)

[1.2 API REST 6](#_Toc135505908)

[1.2.1 Utilisation avec Insomnia 6](#_Toc135505909)

[1.3 Utilisation du site 7](#_Toc135505910)

[1.3.1 Accueil et login 7](#_Toc135505911)

[1.3.2 Ma bibliothèque 8](#_Toc135505912)

[1.3.3 Ajout, édition, suppression d’un média 9](#_Toc135505913)

[2 Topaz Lion 10](#_Toc135505914)

[2.1 Installation 10](#_Toc135505915)

[2.1.1 Dépôt GIT 10](#_Toc135505916)

[2.1.2 Démarrage du projet 10](#_Toc135505917)

[2.2 API REST 11](#_Toc135505918)

[2.2.1 Utilisation avec Insomnia 11](#_Toc135505919)

[3 Docker pour ActiveMQ 12](#_Toc135505920)

[4 Importer un fichier dans Insomnia 13](#_Toc135505921)

# Saphir Lion

## Installation

### Dépôt GIT

Le projet se trouve sur GitHub à l’adresse suivante :

* [**https://github.com/Ozurah-HES/SaphirLion**](https://github.com/Ozurah-HES/SaphirLion)

La commande Git pour cloner par HTTPS est la suivante :

git clone https://github.com/Ozurah-HES/SaphirLion.git

Les modifications apportées pour le semestre de printemps ont été réalisées sur la branche nommée « SP ». Il est donc nécessaire de changer de branche avec la commande suivante (depuis la racine du projet) :

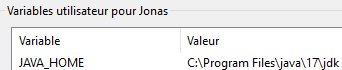
git checkout -b SP remotes/origin/SP

Vous pouvez également récupérer le projet au format zip avec cette URL (téléchargement direct) :  
<https://github.com/Ozurah-HES/SaphirLion/archive/refs/heads/SP.zip>

### Versions Java nécessaires

Le projet est configuré pour fonctionner avec Java 11 au minimum. Cependant, il n’a été testé qu’avec une machine ayant Java 17 d’installé.

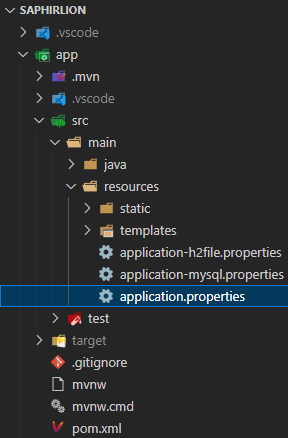
Si vous avez des problèmes pour exécuter les commandes Maven pour le démarrage du projet, assurez-vous d’avoir configuré correctement la variable d’environnement « JAVA\_HOME » pour qu’elle pointe vers le bon JDK.



Capture d'écran 1 – Variable d’environnement « JAVA\_HOME » (exemple de chemin)

### Configurations possibles

Avant de démarrer le projet, vous pouvez modifier la configuration du programme dans le fichier « application.properties ».



Arborescence 1 – Où trouver le fichier « application.properties » dans le projet

Dans ce fichier, vous pouvez changer les valeurs de :

|  |  |  |
| --- | --- | --- |
| Propriété | Valeur par défaut | Valeurs possibles, utilité, remarque |
| server.port | 8080 | Un numéro de port libre. Utilisé pour accéder au site en local.   * TopazLion est configuré pour utiliser le port **9090**. SaphirLion est configuré pour utiliser le port **8080**. Pour éviter des problèmes avec les routes définies ou devoir modifier toutes les configurations, il est conseillé de ne pas changer ces ports. |
| spring.profiles.include | mysql | Choix du type de base de données :   * « **mysql** » nécessite qu’un serveur MySql fonctionne * « **h2file** » va créer une base de données dans un fichier H2, dans le dossier « /db/ » à la racine du projet |
| spring.profiles.active | seed-prod | Choix des seeders pour la base de données :   * « **clean-db** » : vide les tables à chaque démarrage de l’application * « **seed-prod** » : ajoute les entrées de bases pour utiliser le site en production. Uniquement si la BDD est vide. * « **seed-test** » : vide les tables et ajoute des données destinées aux tests. |

* Le seeder « seed-prod » crée deux utilisateurs : « User 1 » et « User 2 » avec comme mot de passe « password ».
* Pour les seeders, il est fortement conseillé de laisser uniquement « seed-prod », de cette manière les données sont persistantes entre chaque démarrage de l’application. De plus, le site ne permet pas la création d’utilisateurs. Donc, pour utiliser pleinement le site, nous utiliserons les utilisateurs précréés par le seeder « seed-prod ».
* Il est possible d’activer le seeder « clean-db » et « seed-prod » en même temps. Pour ce faire, modifier la propriété comme suit :  
  « spring.profiles.active=seed-prod, clean-db ».
* Pour la suite du document, nous partons du principe que la configuration n’a pas été changée !

#### Configuration de la base de données

Il est peut-être nécessaire de modifier les fichiers de configuration de base de données, notamment le compte utilisé ou le mot de passe.

Pour se faire, il faut accéder au fichier « application-mysql.properties » (ou h2file, en fonction du type de BDD spécifié précédemment) et d’adapter les propriétés. Les trois propriétés principales à vérifier sont :

spring.datasource.url=jdbc:mysql://localhost:3306/SaphirLion?createDatabaseIfNotExist=true  
spring.datasource.username=root  
spring.datasource.password=

### Démarrage du projet

Pour démarrer le projet, deux manières sont expliquées dans ce document : en ligne de commande et avec VSCode.

* Si vous utilisez la base de données MySql (par défaut dans la configuration), il est nécessaire de démarrer le serveur MySql au préalable avant de démarrer le projet. Vous pouvez par exemple utiliser XAMPP. Si le serveur MySql ne fonctionne pas, les étapes qui suivent échoueront !
* Pour que Saphir Lion puisse communiquer avec Topaz Lion, il est nécessaire d’avoir le Docker qui tourne. La marche à suivre est indiquée dans le chapitre « 3  Docker pour ActiveMQ ».
* Saphir Lion peut être utilisé sans Topaz Lion (et donc Docker).

#### Avec Maven

L’exécution se fait en ligne de commande :

1. Accédez au sous-dossier « app » du projet  
   cd [chemin du projet]/app
2. Executez « maven clean » :  
   ./mvnw.cmd clean
3. Exécutez « maven install » :  
   ./mvnw.cmd install
4. Démarrer le projet :  
   ./mvnw.cmd spring-boot:run

#### Avec VSCode

* Certaines extensions VSCode sont requises pour pouvoir utiliser le projet. Nous n’allons pas les aborder dans ce document. Cependant, celles qui peuvent être utiles sont « Spring Boot Extension Pack » et « Extension Pack for Java ».

|  |  |
| --- | --- |
| 1. Ouvrez le dossier « app » du projet avec Visual Studio Code. 2. Clic droit sur « SaphirLion » dans la partie « Maven » de l’explorateur : 3. Dans le menu contextuel, exécutez dans l’ordre :    1. clean    2. install    3. Custom… 🡪 spring-boot:run | Capture d'écran 2 – Menu VSCode « Explorer » à Maven |

## API REST

Saphir Lion met à disposition une API REST afin de permettre la manipulation des données. Les différentes requêtes disponibles sont les suivantes :

* Toutes les adresses sont précédées par « localhost:8080/api/ ». Par exemple pour obtenir les utilisateurs, l’adresse complète est « localhost:8080/api/**users** ».

|  |  |  |
| --- | --- | --- |
| Suffixe de la route | Méthode | Description |
| users | GET | Liste tous les utilisateurs existants |
| medias | GET | Liste tous les médias existants |
| media | PUT | Ajoute/modifie un média |
| user/medias | GET | Liste tous les médias d’un utilisateur |
| user/media | PUT | Ajoute/modifie un média d’un utilisateur |
| user/media | DELETE | Supprime un média d’un utilisateur |

* Les données à envoyer et les réponses des routes sont indiquées dans le rapport.

### Utilisation avec Insomnia

Un fichier « Insomnia\_SaphirLion.json » est présent à la racine du projet. Celui-ci peut être ouvert avec l’application Insomnia et contient les différentes requêtes de l’API pour être testé.

* La marche à suivre pour importer le fichier dans Insomnia est expliquée dans le chapitre « 4  Importer un fichier dans Insomnia ».

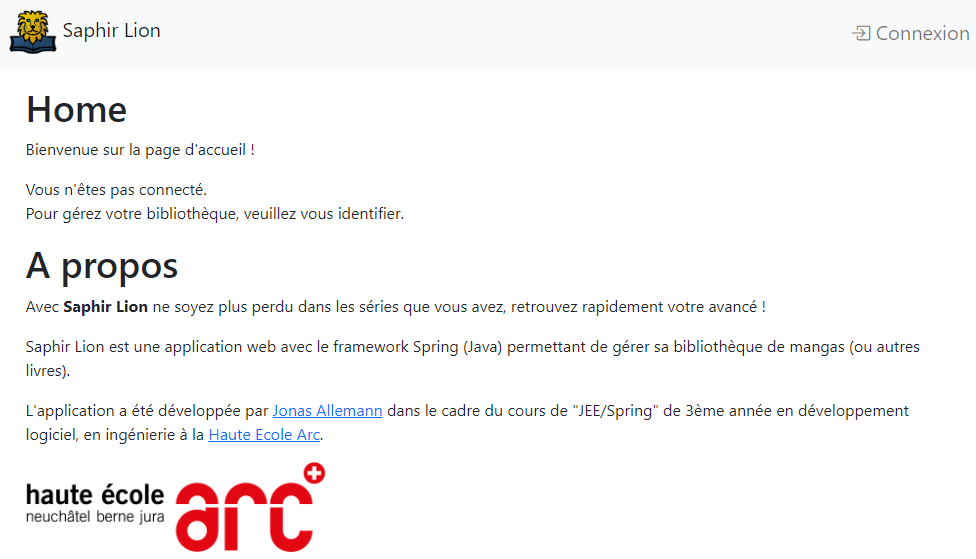
Les corps des requêtes définies dans Insomnia peuvent être utilisés sans modifications. Lors de la 1ère utilisation du projet, aucun média n’est créé. Il est donc nécessaire d’en créer un avec la route « PUT media ». Concernant les utilisateurs, 2 sont présents, avec respectivement l’ID **1** et **2**.

Certaines routes définies dans Insomnia sont destinées à échouer (pour vérifier le comportement en cas d’erreur), celles-ci ont le mot clé « **error** » dans leur nommage.

## Utilisation du site

### Accueil et login

Si votre application a correctement démarré, vous pouvez accéder à la page d’accueil du site à cette adresse : <http://localhost:8080/>.  
Vous devriez avoir cette page :



Page du site 1 – L’accueil « <http://localhost:8080/> » sans être connecté

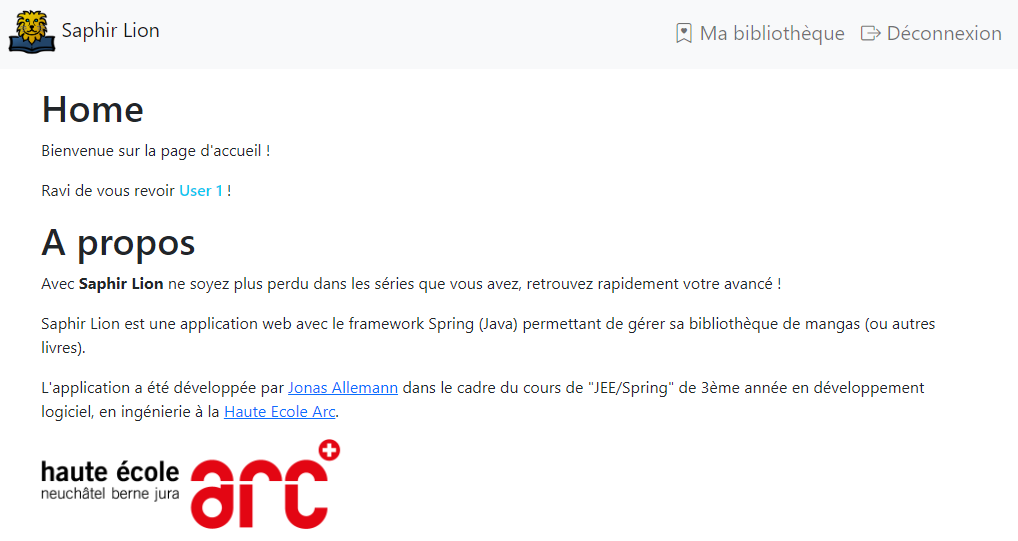
Vous pouvez ensuite vous connecter en appuyant sur « Connexion » dans le menu. Saisissez-y les identifiants disponibles (ceux générés par le seeder « seed-prod »). Pour rappel les comptes par défaut sont :

* « **User 1** » avec comme mot de passe « **password** »
* « **User 2** » avec comme mot de passe « **password** »
* Avec MySql, le nom d’utilisateur n’est pas sensible à la casse. Avec H2, il l’est.



Page du site 2 – Connexion « <http://localhost:8080/login>»

Vous serez redirigé à l’accueil. Vous pourrez y voir de légères différences :

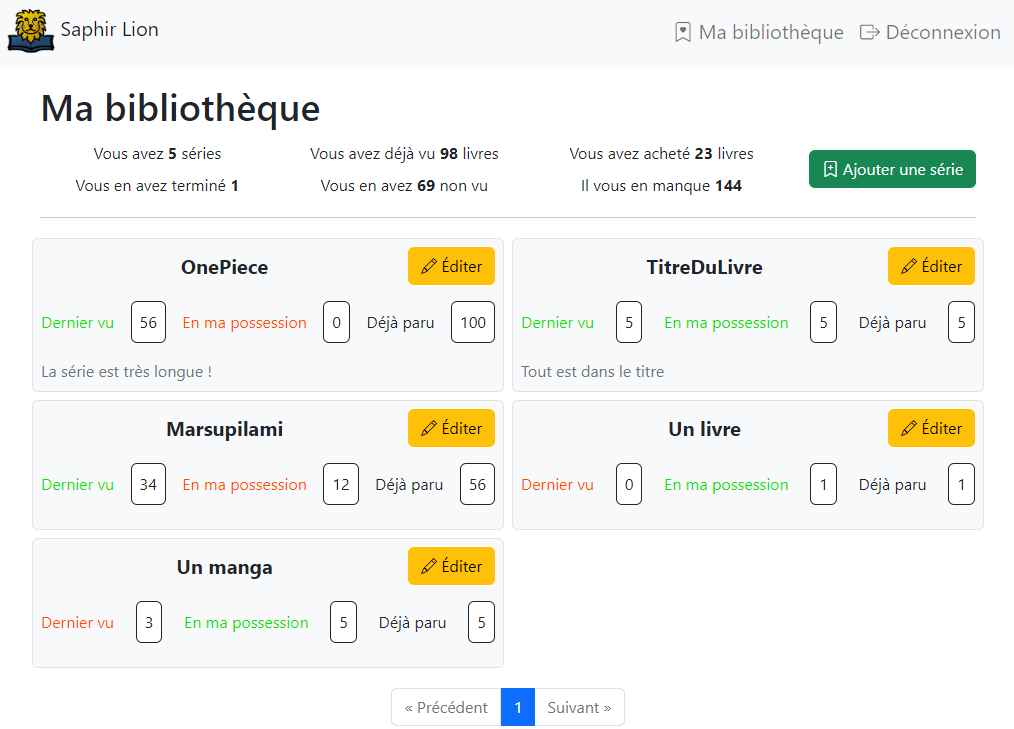


Page du site 3 – L'accueil « <http://localhost:8080/> » en étant connecté. Le menu a changé et l'on y voit le nom d’utilisateur.

Le menu vous propose désormais l’accès à votre bibliothèque et la possibilité de vous déconnecter.

### Ma bibliothèque

En accédant à « ma bibliothèque », vous devriez atteindre une page listant quelques statistiques et tous les médias enregistrés sur votre compte.



Page du site 4 – Ma bibliothèque « <http://localhost:8080/media> », avec quelques médias enregistrés

A vous d’ajouter des séries pour avoir du contenu !

* Le label « dernier vu » devient vert s’il est supérieur à « en ma possession », et orange s’il est inférieur.  
  Le label « en ma possession » est vert s’il est égal à « déjà paru », sinon orange.
* Seule une dizaine de médias sont affichés par page.

### Ajout, édition, suppression d’un média

Quand vous éditez ou ajoutez une série, le formulaire suivant s’affiche. Il s’agit du même pour les deux actions :



*Page du site 5 – Edition/ajout d'un média à sa collection*  
« <http://localhost:8080/media/add> » « <http://localhost:8080/media/edit/99> »

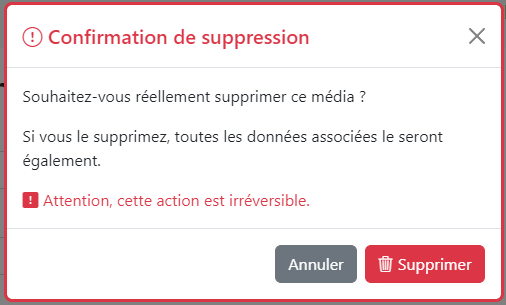
Le champ « média » permet de saisir le nom d’un média (manga, bd, etc.). Une liste des médias non présents dans sa collection est disponible. Il est également possible d’en créer un nouveau en saisissant un titre n’existant pas.

* Une fois qu’un média est enregistré, les autres utilisateurs auront la possibilité de le choisir dans la liste.
* Le nombre de parutions est lié au média, qui n’est pas modifiable par l’utilisateur. Pour modifier cette valeur, il est nécessaire d’utiliser les fonctionnalités REST proposées par l’application « Saphir Lion » ou « Topaz Lion ».

Les champs « dernier vu » et « en ma possession » doivent être inférieurs ou égaux au champ « déjà paru ». Ces 3 champs sont de type numérique et doivent être positifs.

Les champs « texte » sont soumis à une limite de caractère (relativement grande).

Si vous supprimez le média, il sera **retiré de votre collection (cela comprend également la perte des données que vous y avez saisies).** Vous devrez valider l’action :



Page du site 6 – Popup de confirmation de suppression

# Topaz Lion

## Installation

### Dépôt GIT

Le projet se trouve sur GitHub à l’adresse suivante :

* [***https://github.com/Ozurah-HES/TopazLion***](https://github.com/Ozurah-HES/TopazLion)

La commande Git pour cloner par HTTPS est la suivante :

git clone https://github.com/Ozurah-HES/TopazLion.git

Vous pouvez également récupérer le projet au format zip avec cette URL (téléchargement direct) :  
<https://github.com/Ozurah-HES/TopazLion/archive/refs/heads/main.zip>

### Démarrage du projet

Pour démarrer le projet, deux manières sont expliquées dans ce document : en ligne de commande et avec VSCode.

* **TopazLion** est un **projet d’extension de SaphirLion** (communiquant avec ce dernier). Il est donc nécessaire que SaphirLion soit démarré pour que TopazLion soit pleinement fonctionnel. Le Docker pour ActiveMQ doit également être actif, la marche à suivre est indiquée dans le chapitre « 3  Docker pour ActiveMQ ».
* TopazLion est configuré pour utiliser le port **9090**.  
  SaphirLion est configuré pour utiliser le port **8080**.  
  Pour éviter des problèmes avec les routes définies ou devoir modifier toutes les configurations, il est conseillé de ne pas changer ces ports.

#### Avec Maven

L’exécution se fait en ligne de commande :

1. Accédez au sous-dossier « app » du projet  
   cd [chemin du projet]/app
2. Exécutez « maven clean » :  
   ./mvnw.cmd clean
3. Exécutez « maven install » :  
   ./mvnw.cmd install
4. Démarrer le projet :  
   ./mvnw.cmd spring-boot:run

#### Avec VSCode

* Certaines extensions VSCode sont requises pour pouvoir utiliser le projet. Nous n’allons pas les aborder dans ce document. Cependant, celles qui peuvent être utiles sont « Spring Boot Extension Pack » et « Extension Pack for Java ».

|  |  |
| --- | --- |
| 1. Ouvrez le dossier « app » du projet avec Visual Studio Code. 2. Clic droit sur « TopazLion » dans la partie « Maven » de l’explorateur : 3. Dans le menu contextuel, exécutez dans l’ordre :    1. clean    2. install    3. Custom… 🡪 spring-boot:run | Capture d'écran 3 – Menu VSCode « Explorer » à Maven |

## API REST

Topaz Lion met à disposition une API REST afin de permettre la manipulation des données. Les différentes requêtes disponibles sont les suivantes :

* Toutes les adresses sont précédées par « localhost:9090/api/ ». Par exemple pour ajouter un média, l’adresse complète est « localhost:9090/api/**media** ».

|  |  |  |
| --- | --- | --- |
| Suffixe de la route | Méthode | Description |
| media | POST | Va envoyer un média à ajouter/éditer à **SaphirLion** au travers d’ActiveMQ |
| media/fromANN | POST | Va envoyer une requête à l’API de « [AnimeNewsNetwork](https://www.animenewsnetwork.com/encyclopedia/api.php)[[1]](#footnote-2)» pour récupérer le nom et l’image du média spécifié.  Va ensuite envoyer le média à ajouter/éditer à **SaphirLion** au travers d’ActiveMQ |
| media/last-response | GET | Permet d’obtenir la dernière réponse que **SaphirLion** a retransmis au travers d’ActiveMQ à la suite de l’ajout/édition d’un média |
| * Les routes qui suivent ont servi uniquement pour tester l’intégration de l’API REST dans le projet, et servent donc d’exemple. Elles ne sont pas détaillées dans le rapport. | | |
| test | GET | Ajoute/modifie un média |
| test/obj | GET | Liste tous les médias d’un utilisateur |
| test/param | GET | Ajoute/modifie un média d’un utilisateur |

* Les données à envoyer et les réponses des routes sont indiquées dans le rapport.

### Utilisation avec Insomnia

Un fichier « Insomnia\_TopazLion.json » est présent à la racine du projet. Celui-ci peut être ouvert avec l’application Insomnia et contient les différentes requêtes de l’API pour être testé.

* La marche à suivre pour importer le fichier dans Insomnia est expliquée dans le chapitre « 4  Importer un fichier dans Insomnia ».

Les corps des requêtes définies dans Insomnia peuvent être utilisés sans modifications.

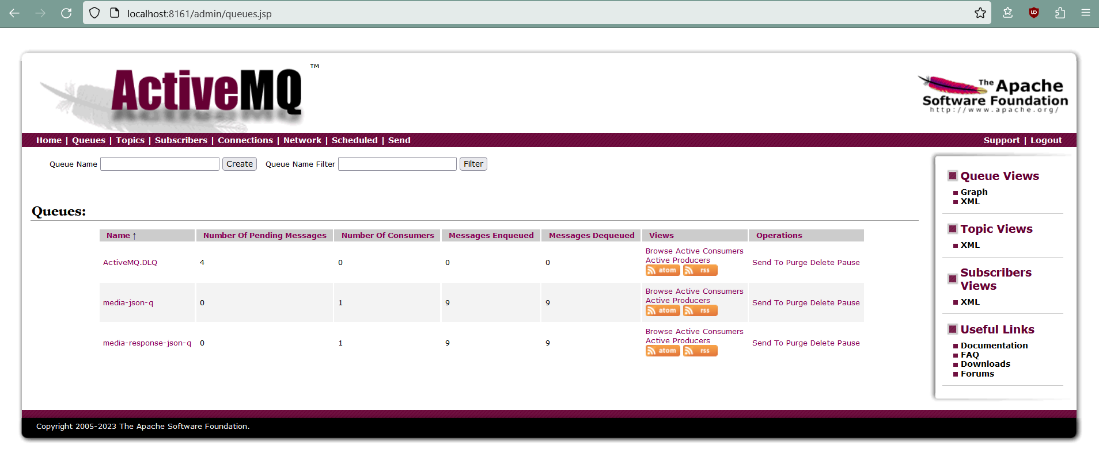
Certaines routes définies dans Insomnia sont destinées à échouer (pour vérifier le comportement en cas d’erreur), celles-ci ont le mot clé « **error** » dans leur nommage.

# Docker pour ActiveMQ

Les projets SaphirLion et TopazLion sont configurés pour communiquer entre eux grâce à **ActiveMQ**. Pour que ce dernier puisse fonctionner, il est nécessaire de démarrer un Docker dédié. La commande à exécuter est la suivante (1 seule ligne) :

docker run -it -p 61616:61616 -p 8161:8161 -p 6572:6572 -e ACTIVEMQ\_DISALLOW\_WEBCONSOLE=false -e ACTIVEMQ\_USERNAME=myactivemquser -e ACTIVEMQ\_PASSWORD=myactivemquserpass -e ACTIVEMQ\_WEBADMIN\_USERNAME=root -e ACTIVEMQ\_WEBADMIN\_PASSWORD=pass symptoma/activemq:latest

Une fois le Docker démarré, il est possible de visualiser les données qui transitent entre les 2 projets à l’adresse suivante : <http://localhost:8161>



Capture d'écran 4 – Liste des « queues » sur la page « ActiveMQ »  
URL : [*localhost:8161/admin/queues.jsp*](localhost:8161/admin/queues.jsp)

# Importer un fichier dans Insomnia

Commencer par télécharger Insomnia (<https://insomnia.rest/download>).  
Ensuite, depuis la page d’accueil de l’application Une image contenant capture d’écran, Rectangle, conception

Description générée automatiquement :

1. Appuyer sur « Create »

Une image contenant capture d’écran, Logiciel multimédia, logiciel

Description générée automatiquement

1. Puis sur « Import » :

Une image contenant texte, capture d’écran, Police, nombre

Description générée automatiquement

1. Placez-y le fichier à importer. Ici, nous avons sélectionné « Insomnia\_SaphirLion.json ». Ensuite, appuyez sur « Scan » :

Une image contenant texte, capture d’écran, Logiciel multimédia, logiciel

Description générée automatiquement

1. Nommez-le (exemple *SaphirLion*), puis appuyez sur « Import » :

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

1. Vous retrouverez vos documents sur la page d’accueil :

Une image contenant capture d’écran, texte, logiciel, Logiciel multimédia

Description générée automatiquement

1. Pour voir les routes créées, ouvrez un document, et accédez à l’onglet « debug »



1. Toutes les routes du document seront visibles dans le volet à gauche (regroupées dans des dossiers) :

Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Description générée automatiquement

1. Pour exécuter une route, ouvrez une requête (exemple « GET Medias »), puis cliquer sur « Send ». La requête est envoyée et la réponse est affichée (rectangle bleu).

Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Description générée automatiquement

1. <https://www.animenewsnetwork.com/encyclopedia/api.php> [↑](#footnote-ref-2)