Laboratoire SMTP

Auteurs

Maxime Chantemargue & Charles Matrand

Description du projet

Ce projet rentre dans le cadre d'un laboratoire du cours DAI de l'HEIG-VD. C'est une application client qui utilise le Socket API pour communiquer avec un serveur SMTP mais cela sans utiliser de librairie qui s'occuperait des détails du protocole.

Qu'est-ce que MockMock

MockMock est un serveur SMTP de test qui permet de tester des envois/réceptions de mails. Il permet d'éviter d'utiliser un vrai SMTP avec toutes les différentes procédures de sécurité qui nous bloquerait instantanément. Nous pouvons voir MockMock comme un environnement de test unitaire pour SMTP.

Mise en place de MockMock

Dans le cas où vous souhaitez tester l'envoi de mail, vous pouvez utiliser MockMock dans un Docker. L'avantage de Docker est que l'installation de MockMock dans celui-ci ne modifiera pas l'intégrité de votre machine. C'est une machine virtuelle isolée à votre système qui tournera en arrière-plan avec laquelle vous pourrez interagir par le biais de ligne de commande.

Installer Docker

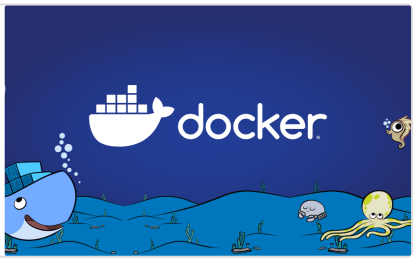
Si votre machine tourne sur un processeur ARM, vous ne pourrez pas effectuer les étapes suivantes à compter de maintenant.

Rendez-vous sur le site de Docker et installez-le sur votre machine.

Docker: Accelerated, Containerized Application Development

Docker is a platform designed to help developers build, share, and run modern applications. We handle the tedious setup, so you can focus on the code.

https://www.docker.com/



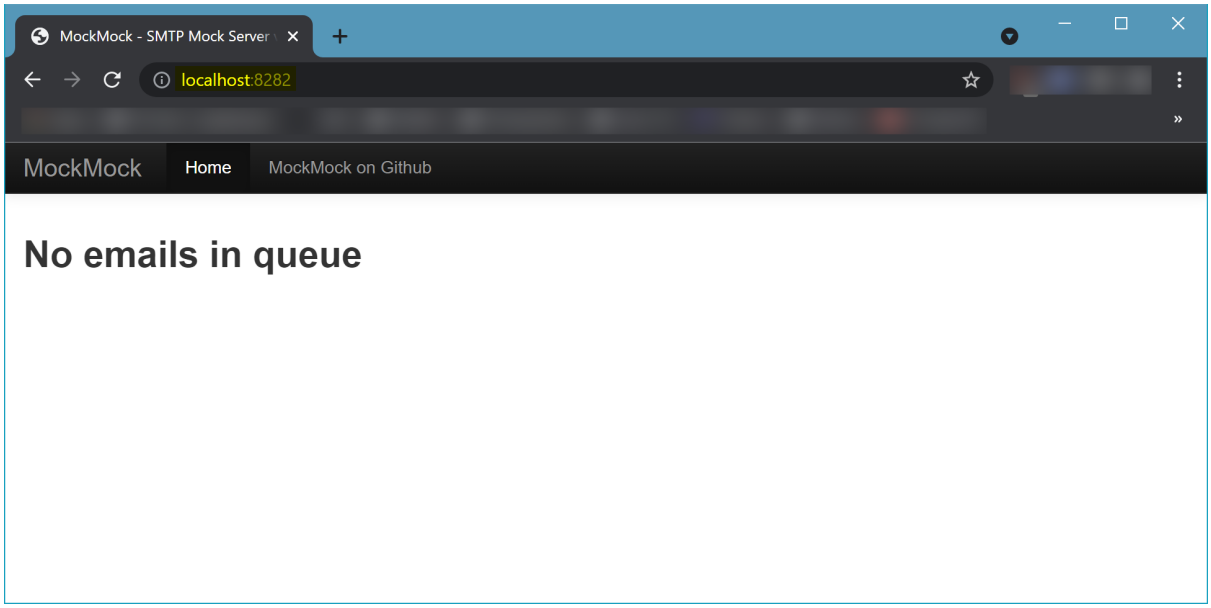
Installer MockMock dans Docker

Ouvrez un terminal sur votre machine et exécuter ensuite la commande suivante :

1. `docker pull peccu/mockmock`
2. `docker run -it -p 8282:8282 -p 25:25 peccu/mockmock`

Après exécution de ces deux commandes MockMock sera installé et actif sur votre machine et vous pourrez commencer à interagir avec par le biais de votre choix.

Vous devriez être en mesure d'accéder à l'UI de MockMock sur l'adresse suivante : **http://localhost:8282/** avec l'interface ci-dessous →



Configuration

Pour faire fonctionner la démo il est **impératif** de passer au programme 3 fichiers de configuration au format suivant :

- `config.properties` :

```
numberOfGroups=6
smtpServerAddress=localhost
smtpServerPort=25
```

- `messages.json` :

```
[
  {
    "subject": "This is a spam message",
    "body": "I trolled you"
  },
```

```
  {
    "subject": "Chocolate offer",
    "body": "I will pay you 1000$ for your chocolate"
  },
]
```

- `victims.json` :

```
[
  {
    "email": "john.doe@example.com"
  },
  {
    "email": "jane.doe@example.com"
  },
]
```

Il est **OBLIGATOIRE** que ces fichiers listés ci-contre soient formatés de la sorte(le contenu de ces fichiers est fictif et donc utilisé à titre d'exemple, vous pouvez créer vous même ces fichiers dans la mesure où le formatage est respecté).

Après avoir créés ces fichiers vous pouvez ensuite lancer le programme avec la commande suivante : `MailRobot config.properties victims.json messages.json`

Implementation

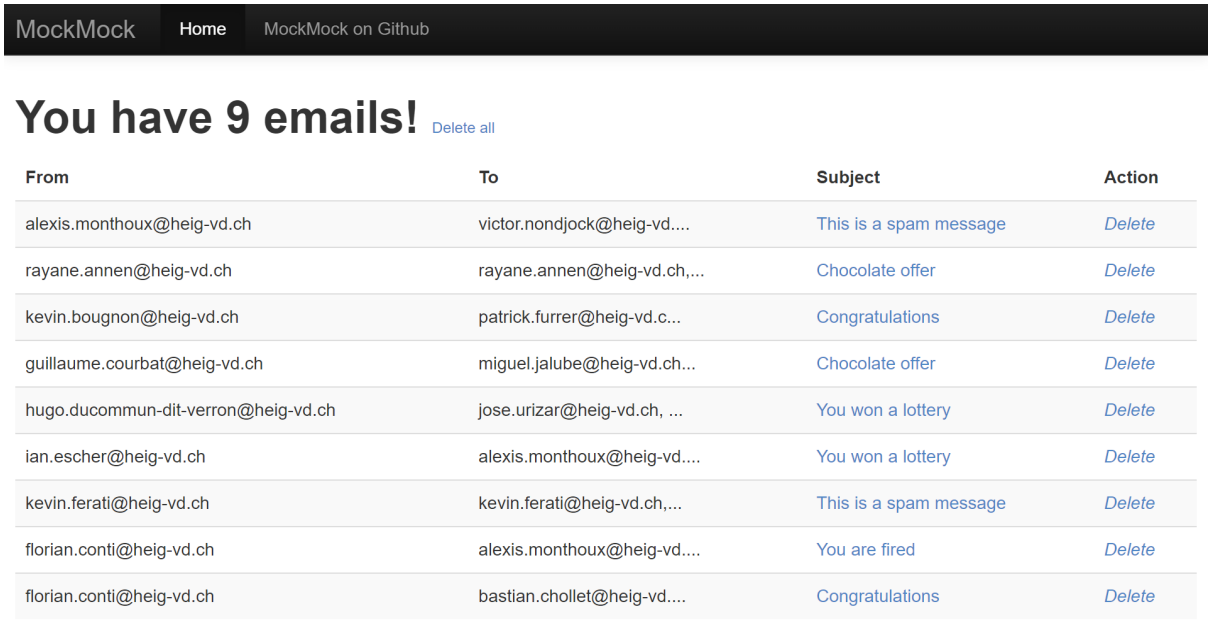
Voici ci-dessous un diagramme global de l'implémentation du programme :

- `FileParser`: permet de traiter le contenu des fichiers de config pour ensuite utiliser celui-ci dans les autres classes.
- `SmtpClient`: représente le client Smtp qui va interroger un serveur smtp et effectuer toutes les commandes relatives au protocle SMTP.
- `SmtpResponse`: représente les messages de réponse du serveur SMTP avec lequel le client communique.
- `Email` : représente le traitement d'une adresse email. Cette classe a été créée dans le but de garantir le respect de l'orienté objet. Elle vérifie qu'une adresse email est valide, ainsi lorsqu'un objet `Person`est créé, il est garanti que celui-ci ait une adresse email valide.
- `Person`: représente les envoyeurs, receveurs de mail.
- `Message`: représente le contenu d'un email soit : un sujet ainsi qu'un corps.

- Group : représente un groupe d'objet `Person` soit : un envoyeur ainsi qu'une liste de receveurs.
- `MailRobot` : programme principal testant notre implémentation

Résultats

Voici ci-dessous le résultat obtenu lors de l'exécution de notre programme :


MockMock
Home
MockMock on Github
You have 9 emails!
Delete all
From
To
Subject
Action
alexis.monthoux@heig-vd.ch
victor.nondjock@heig-vd....
This is a spam message
Delete
rayane.annen@heig-vd.ch
rayane.annen@heig-vd.ch,...
Chocolate offer
Delete
kevin.bougnon@heig-vd.ch
patrick.furrer@heig-vd.c...
Congratulations
Delete
guillaume.courbat@heig-vd.ch
miguel.jalube@heig-vd.ch...
Chocolate offer
Delete
hugo.ducommun-dit-verron@heig-vd.ch
jose.urizar@heig-vd.ch, ...
You won a lottery
Delete
ian.escher@heig-vd.ch
alexis.monthoux@heig-vd....
You won a lottery
Delete
kevin.ferati@heig-vd.ch
kevin.ferati@heig-vd.ch,...
This is a spam message
Delete
florian.conti@heig-vd.ch
alexis.monthoux@heig-vd....
You are fired
Delete
florian.conti@heig-vd.ch
bastian.chollet@heig-vd....
Congratulations
Delete