

# Teaching-HEIGVD-RES-2018-Labo-SMTP

## Contexte

---

MailRobot java application a été développée pour le cours RES, @HEIG-VD.

## Description

---

Dans ce laboratoire, nous avons développé une application client avec le protocole qui utilise un API socket pour communiquer avec le serveur SMTP. Pour le réaliser nous sommes basés sur les webcasts proposés par le professeur et avons suivi les instructions. Le but du labo était de générer des plaisanteries. de réaliser une application permettant d'envoyer automatiquement les emails spam. Ces mails sont construits à partir d'un groupe de victimes stockés dans un fichier texte contenant des adresses mails de ces personnes. Un groupe consiste en un émetteur et un set de récepteurs pris au hasard.

L'application va alors créer un mail forgé dont le corps du message est lu dans un fichier texte contenant une liste de messages. Un message est pris de manière aléatoire.

## Utilisation

---

L'application communique avec un serveur SMTP local à travers le port 25. On peut modifier les propriétés du serveur à partir du fichier de configuration(config.properties) situé à la racine du projet. On peut également changer l'ensemble des adresses emails(vitims.utf8) des victimes ou les messages de plaisanterie(messages.utf8).

Pour pouvoir faire tourner notre application nous avons utilisé Docker. le Docker file qui vous a été fourni dans le dossier docker est le suivant:

```
FROM java:openjdk-8-jre-alpine

ADD src /opt/src/

WORKDIR /opt/src/

ENTRYPOINT [ "java", "-jar", "MockMock.jar", "-p", "2525", "-h", "8080" ]
```

Commande pour la construction de notre image `docker build -t mockmock .`

commande pour créer notre conteneur pour faire tourner notre serveur MockMock

`docker run -p 2525:2525 -p 8080:8080 mockmock` le premier numéro de port à savoir `2525`


est le numero de port que le client utilisera pour se connecter au serveur afin d'envoyer les messages. ce numero est modifiable dans le fichier `config.properties`.

le second `8080` est le numero de port de l'interface web de mockmock

# Implementation

---

## Diagramme de classe

ci dessus le diagramme de classe de notre laboratoire.  Désolé pour la qualité de l'image. Vous pouvez retrouver le diagramme de classe réalisé avec starUml dans le dossier README.

## Description des differentes classes les plus importantes

### ConfigurationManager

Cette classe nous permet de recuperer toutes les entrees à savoir les adresses, les numeros de port, les groupes etc... Bref tout ce dont a besoin l'application concernant le serveur.

### SmtClient

Cette classe nous permet d'envoyer les messages grace au protocole SMTP vu en cours de RES

### PranGenerator

Cette classe nous permet de generer des plaisanteries, les groupes a qui ont envoie et fait l'envoi via SmtClient.