# MANUEL D'UTILISATION

ANALYSE D'UN FICHIER LOG

05 FEVRIER 2021

JIN FRANCINE – TRAN QUANG HUY
3IF GROUPE 210

# Table des matières

Le développement de cette application a été fait sous Linux. Nous ne garantissons pas un parfait fonctionnement sous Windows.

## INTRODUCTION

Cette application permet d'analyser le contenu d'un fichier LOG, un fichier journal qui consigne toutes les actions effectuées entre un navigateur et un serveur. Chaque ligne représente une requête du navigateur et la réponse du serveur. Différentes options d'affichage sont proposées, pour cela il suffit de taper la commande correspondante dans le terminal.

## FORMAT DU FICHIER

Peu importe l'option choisi, chacune des commandes nécessite obligatoirement de passer en paramètre le nom du fichier .LOG/.TXT à analyser.

Dans le cadre du TP, les fichiers seront générés par Apache et auront donc une structure spécifique à celui-ci comme l'exemple ci-dessous :

*192.168.0.0 - - [08/Sep/2012:11:16:02 +0200] "GET /temps/4IF16.html HTTP/1.1" 200 12106 "http://intranet-if.insa-lyon.fr/temps/4IF15.html" "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:14.0) Gecko/20100101 Firefox/14.0.1"*

Chaque ligne de ce fichier devra au moins contenir les informations suivantes qui seront vérifiées par l'application :

192.168.0.0

C'est l'adresse IP du client émetteur de la requête. Chacun des quatre nombres est compris entre 0 et 255.

- -

Ces deux champs ont été anonymisés dans le cadre du TP. Le premier - correspond au nom d'utilisateur (User Logname) du visiteur tandis que le deuxième – correspond au nom d'utilisateur (Authenticated user) que l'internaute s'est attribué.

[08/Sep/2012:11:16:02 +0200]

C'est la date et l'heure de la requête. Il faut que le chiffre soit entre 1 et 31 pour le jour. Et l'heure doit être entre 00 et 23h.

"GET /temps/4IF16.html HTTP/1.1"

/temps/4IF16.html correspond à l'URL du document demandé dans la requête. Elle est relative par rapport à l'URL de base du site.

200

C'est le code que retourne le serveur, il indique l'état de la requête. Nous ne traiterons que les lignes où le code est entre 200 et 300, indiquant son bon déroulement.

# DEMARRAGE

Avant d'utiliser l'application, pensez à bien suivre les étapes suivantes pour mettre en place les éléments nécessaires à son bon fonctionnement :

1. Dézippez le dossier contenant l'application et vérifiez la présence des éléments suivants :
   - Un fichier README.md, s'il n'y est pas, ce n'est pas très grave, tout est expliqué dans ce manuel
   - Un fichier makefile qui répertorie les commandes/raccourcies utiles
   - Un dossier Tests regroupant les tests effectués mais aussi un readme contenant la description des différents éléments du dossier ainsi que leur utilité
   - Un dossier src qui contient le code source de l'application. Veuillez vérifier la présence de ApacheLogAnalyzer.cpp, Graph.cpp, Graph.h, Request.cpp et Request.h
2. Ouvrez le projet dans un environnement de développement. Nous avons utilisé Visual Studio Code. (Au pire des cas, créez un nouveau projet et copiez-collez le code de src dedans)
3. Après l'ouverture du projet, tapez dans le terminal de commandes de votre IDE :

   make

   pour vous aider dans les étapes suivantes. Cette commande va afficher l'aide. Vous pouvez aussi taper make help qui a la même fonction.

4. Nettoyez tout ce qui aurait pu être généré antérieurement en faisant la commande :

   make clean

5. Générez l'exécutable en tapant :

   make release

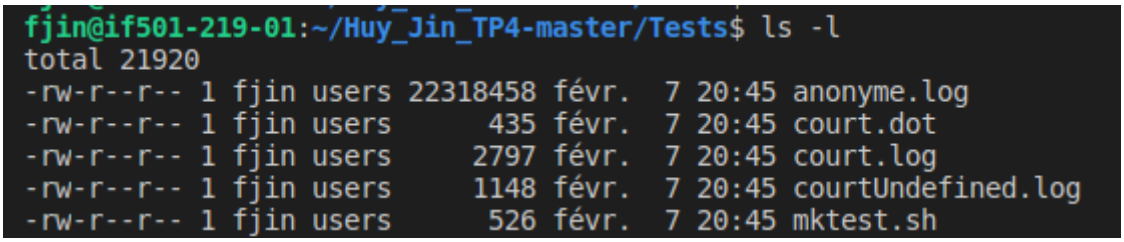
6. Pour lancer les tests, il suffit de taper la commande :

   make test

   Il se peut que les tests ne se lancent pas et des messages « Permission non accordée » apparaissent. Il suffit alors de faire

   Ls -l

   dans le dossier du fichier problématique.



Sur l'image, on peut voir qu'on n'a pas les droits en écriture et en exécution de mktest.sh. Si le terminal vous indiquait un problème sur ce fichier, il suffit alors pour vous rajouter les droits en exécution, de faire :

```
chmod u+x mktest.sh
```

Relancez les tests, en principe cela devrait marcher. Si ça ne marche pas, à la place de +x, mettez +w pour voir.

7. Vous êtes maintenant prêt à utiliser l'application !

# LES COMMANDES

Les différentes options présentées peuvent être combiné mais il faudra bien faire attention à l'ordre pour certaines qui nécessitent d'être immédiatement suivi de leur paramètre. NomFichier doit être remplacer par le nom du fichier que vous souhaitez analyser tandis que analog correspond au nom de notre programme. Aussi le fichier à analyser doit être dans le même répertoire que notre programme sinon il faudra mettre le chemin relatif qui permet d'y accéder et non plus que NomFichier !

### Sans option

```
$./analog NomFichier.log
```

C'est la commande de base pour lancer l'application. Il permet d'afficher sur la console les 10 documents les plus consultés par ordre de popularité. Quand le fichier LOG ne contient pas 10 éléments, il les affichera tous.

### -e

```
$./analog -e NomFichier.log
```

En ajoutant cette option, on exclut de l'affichage tous les fichiers de type ".css", ".png", ".bmp", ".jpg", ".gif", ".ico", ".js".

### -g

```
$./analog -g NomFichier.dot NomFichier.log
```

En ajoutant cette option, on obtient un fichier NomFichier.dot au format GraphViz qui contient les données nécessaires pour former un graphe. Pour générer le graphe sous forme d'image (ici png), il faut taper dans le terminal :

```
dot -Tpng -o NomFichier.png NomFichier.dot
```

### -t

```
$./analog -t heure NomFichier.log
```

En ajoutant cette option, le terminal va afficher toutes les lignes du fichier LOG qui sont dans le créneau [heure ; heure+1]. Heure est un nombre compris entre 00 et 23.

### -p

```
$./analog -p typeExtension NomFichier.log
```

En ajoutant cette option, on analyse que les lignes qui contiennent typeExtension dans leur URL.

### -h

```
$./analog -h
```

En ajoutant cette option, le terminal affichera le manuel d'aide de l'application MAIS elle ne peut pas être combiné avec d'autres options. Elle est prioritaire et les autres ne s'exécuteront pas.