Architecture et langages Web
(NFA040)

Implantation du protocole HTTP

Ce TP porte sur l'implantation d'un protocole textuel (client et serveur), et du protocole HTTP en particulier, en Java (des sources partiels sont fournis).

Protocole textuel

Exercice 1 : Connexion TCP

1. On souhaite implanter une application client-serveur simple Java, où le serveur crée une socket d'écoute sur le port 55000 et attend qu'un client se connecte. On affichera un message à chacune des étapes, côté client et côté serveur.

2. Compléter le code précédent pour que le client envoie un nombre au serveur qui doit alors l'afficher, et renvoyer le double de ce nombre au client, qui affiche alors le résultat.

Exercice 2 : Application client-serveur

On souhaite implanter une application client-serveur avec le protocole textuel simple suivant :

- le client envoie "hello"
- le serveur répond "hello"
- le client envoie "name" suivi de son nom
- le serveur répond alors "bye" suivi du nom du client
- le client répond enfin "bye"

Questions

1. Implanter le protocole ci-dessus en Java (client et serveur). On vérifiera que le protocole est respecté à la lettre coté client et côté serveur, sinon l'exception IOException sera levée (avec un message explicite).

2. Modifier le protocole de manière à transmettre le nom suivi du prénom ("firstname") au serveur.

Exercice 3 : netcat

`netcat` est un utilitaire utilisé initialement pour se connecter via TCP sur une machine distance. L'utilitaire `netcat` peut en particulier être utilisé pour simuler un protocole textuel. La commande, qui s'appelle `nc`, prend en paramètre le nom du serveur et le port sur lequel la connexion doit être établie.

Questions

1. Tester votre serveur de l'exercice précédent en utilisant la commande `nc` pour simuler le client.

Protocole HTTP

Le protocole HTTP est documenté ici : https://developer.mozilla.org/en-US/docs/Web/HTTP

Exercice 4 : Serveur textuel « factice »

Questions

1. Implanter en Java un serveur textuel « factice » qui se contente d'afficher sur sa sortie standard les requêtes reçues sans les traiter.

2. Utiliser votre serveur factice pour observer les réquêtes de différents clients (firefox, wget, curl ...)

Exercice 5 : Format d'une requête HTTP

Questions

1. Quels sont les différents éléments d'un URL? Le « `www` » est-il indispensable pour un serveur web ? Faire le test avec « `www.education.gouv.fr` » en utilisant la commande `nslookup` ou la commande `dig` (installer le package `dnsutils` si nécéssaire).

2. Quelle est la syntaxe des deux premières lignes d'une requête HTTP pour demander le contenu de la page d'accueil du site www.cnam.fr ?

Exercice 6 : netcat

Le but de cet exercice est de simuler certaines requêtes du protocole `HTTP` avec l'utilitaire `netcat`. Cela peut s'avérer très utile pour tester et diagnostiquer des problèmes sur un serveur. Sous unix, l'option `-c` (ou `-C`) peut être nécéssaire pour forcer l'encodage de saut de ligne par CRLF.

Questions

1. Avec l'utilitaire `netcat`, récupérez la page `html` qui se trouve à l'adresse suivante :

   http://cedric.cnam.fr/sys/crolard/index.html.

   Pour cela, utilisez `nc` pour vous connecter à l'hôte `cedric.cnam.fr` sur son port 80 et écrivez une requête GET permettant de demander la page /sys/crolard/index.html.

2. Décrivez la réponse obtenue.

3. Écrivez la requête `HTTP` qui permet de ne récupérer que l'entête `HTTP`.

Exercice 7 : Client HTTP simple

Questions

1. Implanter en Java un client HTTP simple qui demande une page à un serveur http et l'affiche sur la sortie standard. Testez votre client.

2. (*optionnel*). Modifier votre client pour qu'il sauvegarde le fichier si c'est un fichier texte ou HTML. Tester par exemple avec l'URL : `http://cedric.cnam.fr/sys/crolard/index.html`

3. (*optionnel*). Modifier votre client pour qu'il sauvegarde le fichier si c'est un fichier binaire (pdf par exemple). Il faudra pour cela récupérer la taille du fichier (champ « `Content-Length:` » de la réponse. Tester avec l'URL : `http://cedric.cnam.fr/sys/crolard/TP1.pdf`

Exercice 8 : Serveur HTTP simple

Questions

1. Implanter en Java un serveur HTTP simple qui sait répondre uniquement à une requête `get` de la norme HTTP 1.1. On se limitera au cas où le fichier est un fichier texte.

2. (*optionnel*). Modifier votre client pour qu'il réponde aussi si le fichier demandé n'existe pas.

3. (*optionnel*). Modifier votre client pour qu'il réponde aussi si le fichier demandé n'est pas textuel.