

## TD M2102 - Architecture des réseaux Programmation des sockets TCP

Allez dans `Bibliotheque` et copiez sur votre compte le contenu du répertoire suivant :

`M2102 - Architecture des reseaux/seanceTCP/`

*Remarque* : ce répertoire contient deux répertoires :

- `seanceTCP/C` contient les sources C de BigBen issues du support de cours, ces sources sont fournies à titre indicatif, elles ne seront pas utilisées dans cette séance.
- `seanceTCP/CSharp` contient les sources C# de BigBen et celles d'un squelette d'un client POP3 dont vous allez poursuivre le développement.

### Exercice 1 : Expérimentations avec telnet : BigBen et POP3

La commande `telnet` permet d'établir une connexion TCP avec une machine distante, sur un port donné, et peut jouer le rôle d'un "client TCP universel".

1. Les sources de BigBen vues en cours vous sont fournies au sein d'une solution Visual Studio en C# dans le répertoire `CSharp/BigBen`. Pour ouvrir la solution, cliquez sur `BigBenSocket.sln`. Consultez les sources des 4 projets : `ClientV0` et `ServeurV0`, une implémentation avec socket de base, `ClientV1` et `ServeurV1` avec des "stream".

Exécutez un serveur BigBen sur une machine, et connectez vous à ce serveur grâce :

- au client fourni (ouvrez un autre Visual Studio sur la même machine ou une autre),
- à la commande : `telnet <machine_serveur> <num_port>`.

Que remarquez-vous ? Sous quelle forme récupérez-vous les données avec `telnet` ?

Les protocoles "classiques" sont normalisés et décrits par des documents appelés RFC (Requests For Comments) qui sont accessibles très facilement sur le Web. Celui pour POP3 (rfc1939) vous est fourni dans `seanceTCP/`.

2. Vous pouvez vous connecter avec `telnet` sur divers services : mail, web, ... (sous linux, voir le fichier `/etc/services` pour les numéros de port). En particulier, essayez le serveur POP3 dont il sera question dans l'exercice suivant (compte : `mstinfo1`, mot de passe : `master`, mél : `mstinfo1@free.fr`, serveur pop3 : `pop.free.fr`), et dont voici les commandes autorisées :

Commande	Fonction
<code>USER <i>utilisateur</i></code>	Nom du compte de l'utilisateur
<code>PASS <i>mot de passe</i></code>	Mot de passe
<code>STAT</code>	Donne le nombre de messages non lus et leur taille totale
<code>RETR <i>n</i></code>	Récupère le message numéro <i>n</i>
<code>DELE <i>n</i></code>	Supprime le message numéro <i>n</i>
<code>LAST</code>	Donne le numéro du dernier message auquel on a accédé
<code>LIST [<i>n</i>]</code>	Taille de <i>n</i> -ième message ou de tous les messages
<code>RSET</code>	Annule la suppression de tous les messages.
<code>TOP <i>n k</i></code>	Affiche les entêtes et <i>k</i> lignes du messages numéro <i>n</i>
<code>NOOP</code>	Ne fait rien
<code>QUIT</code>	Temine la session POP3

## Exercice 2 : Programmation en C# d'un client pour POP3

**Travail à rendre :** cet exercice est à réaliser en binôme et à remettre à votre enseignant à la fin de la 2ème séance.

Les sources de départ se trouvent dans `seanceTCP/CSharp/ClientPOP3`. L'application fournie est une ébauche d'un client pour POP3. Elle réalise la connexion au serveur, l'envoi du nom de l'utilisateur et de son mot de passe, et récupère la réponse du serveur.

Les envois et réceptions sont réalisés à l'aide de "stream". Ceci permet en particulier une lecture ligne par ligne des réponses, et facilite énormément leur gestion.

Le but de l'exercice est d'ajouter des fonctionnalités à cette application.

1. Exécutez cette application et familiarisez vous avec les sources (le code contient des commentaires qui complètent la description sommaire précédente et donne la syntaxe d'utilisation). N'oubliez pas d'envoyer des messages dans la boîte aux lettres pour qu'il se passe quelque chose ...
2. Rajoutez un menu qui permet de réaliser les fonctionnalités suivantes.  
(**Auparavant, consultez l'annexe ci-dessous !**)
  - (a) Affichez le *nème* message.
  - (b) Affichez uniquement le nom de l'expéditeur et le sujet du *nème* message.
  - (c) Affichez le nom de l'expéditeur et le sujet de tous les messages.
3. Imaginez des extensions des fonctionnalités précédentes, et de nouvelles fonctionnalités.

## Annexe : remarques importantes

### Quelques compléments sur POP3 :

- Toute réponse d'un serveur POP3 qui contient plusieurs lignes est terminée par une ligne contenant un caractère '.' tout seul en début de ligne.
- Si, "par malchance", une ligne de données commence par un point, celui-ci est doublé. Ainsi, on ne pourra pas confondre une ligne de données qui ne contiendrait qu'un point (elle sera alors codée "..CR/LF") avec la ligne de fin (".CR/LF")
- Expérimentez avec telnet pour mieux comprendre ces remarques.
- Pour plus d'informations, voir la RFC !

Voici quelques compléments sur des manipulations de chaînes de caractères en C si vous souhaitez exploiter les sources C fournies.

### Manipulation des chaînes de caractères avec le langage C :

- La fonction `strncmp` permet de comparer le début de deux chaînes.
- La fonction `strncasecmp` fait la même chose en ignorant les différences minuscules / majuscules.
- Pour plus d'informations, voir `man` !