Corrigé du TP1

Question 1

Connectez-vous à un serveur http à l'aide de la commande *curl* et demandez l'affichage d'un document en envoyant une requête. (Vous afficherez l'ensemble des données envoyées et reçues par curl)

curl -v http://lmbp.uca.fr/~barrel/tp1/test.html

--verbose (ou -v) permet d'utiliser curl en mode verbose et donc d'afficher les entêtes de la requête et de la réponse.

Requête générée par curl :

```
> GET http://lmbp.uca.fr/~barrel/tp1/test.html HTTP/1.1
> Proxy-Authorization: Basic YXVjb2xvbWJldDpnb2xlbWRlbGF2ZTE=
> User-Agent: curl/7.21.0 (x86_64-pc-linux-gnu) libcurl/7.21.0 OpenSSL/0.9.8o zlib/1.2.3.4 libidn/1.15 libssh2/1.2.6
> Host: lmbp.uca.fr
> Accept: */*
> Proxy-Connection: Keep-Alive
```

Note : on constate que, par défaut, curl utilise la méthode *GET* (ligne 1). L'en-tête *Host* est généré automatiquement à partir de l'URL saisie dans la ligne de commande (ligne 4).

Réponse du serveur HTTP :

```
* HTTP 1.0, assume close after body
< HTTP/1.0 200 OK
< Date: Mon, 04 Feb 2013 16:08:27 GMT
< Server: Apache/2.2.22 (FreeBSD) PHP/5.4.4 mod_ssl/2.2.22 OpenSSL/0.9.8q DAV/2
< Content-Location: test.html.txt.fr
< Vary: negotiate,accept,accept-language
< TCN: choice
< Last-Modified: Tue, 03 Feb 2015 10:07:32 GMT
< ETag: "a-50e2c3d384100;50e0fcd8c1485"
< Accept-Ranges: bytes
< Content-Length: 8
< Content-Type: text/plain
< Content-Language: fr
< Age: 2219
< X-Cache: HIT from localhost
< X-Cache-Lookup: HIT from localhost:60158
< Via: 1.0 localhost (squid/3.1.6)
* HTTP/1.0 proxy connection set to keep alive!
< Proxy-Connection: keep-alive
<
Bienvenue
```

Note : Le **code 200** indique que la page demandée a été envoyée avec succès (ligne 2). On constate que par défaut, la ressource envoyée est en français (Content-Language: fr) et au format texte (Content-Type: text/plain). Le corps de la réponse (*Bienvenue*) est contenu dans le fichier *test.html.txt.fr*.

Question 2

Quelles sont les options de communication disponibles pour la ressource?

curl -v http://lmbp.uca.fr/~barrel/tp1/test.html -X OPTIONS

--request (ou -X) permet de modifier la méthode utilisée par défaut (GET) par curl.

Requête générée par curl (seulement la 1ere ligne) :

```
> OPTIONS http://lmbp.uca.fr/~barrel/tp1/test.html HTTP/1.1
```

Note : on constate que la méthode *GET a été remplacée par la méthode OPTIONS* (ligne 1).

Réponse du serveur HTTP (seulement la ligne 6):

```
< Allow: GET,HEAD,POST,OPTIONS,TRACE
```

Note : On obtient ici la liste des méthodes autorisées par le serveur : **GET, HEAD, POST, OPTIONS et TRACE.**

Question 3

Essayez la négociation de contenu sur le type et la langue.

curl -v http://lmbp.uca.fr/~barrel/tp1/test.html -H "accept-language: en" -H "accept: text/html"

--header (ou -H) permet de modifier les en-têtes de la requête.

Requête générée par curl :

```
> GET http://lmbp.uca.fr/~barrel/TP/test.html HTTP/1.1
> Proxy-Authorization: Basic YXVjb2xvbWJldDpnb2xlbWRlbGF2ZTE=
> User-Agent: curl/7.21.0 (x86_64-pc-linux-gnu) libcurl/7.21.0 OpenSSL/0.9.8o zlib/1.2.3.4 libidn/1.15 libssh2/1.2.6
> Host: lmbp.uca.fr
> Accept: text/html
> Accept-language: en
> Proxy-Connection: Keep-Alive
```

Note : L'en-tête par défaut (*Accept: */**) est remplacée par *Accept: text/html* et l'en-tête *Accept-language: fr* est ajoutée.

Réponse du serveur HTTP (seulement le corps de la réponse):

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
        Welcome !!
</body>
</html>
```

Note : On constate que le corps de la réponse est en **anglais ET en HTML**.

Pour la sélection de la langue, on peut aussi écrire :

curl -v http://lmbp.uca.fr/~barrel/tp1/test.html -H "accept-language: fr;q=0.4, en;q=0.8"

Note : Le fait de rajouter **q=x** (q pour qualité, avec x compris entre 0 et 1) permet de hiérarchiser les langues voulues (ou le format). Dans l'exemple ci-dessus, si l'anglais n'est pas disponible (*on donne néanmoins priorité à l'anglais car q=0.8*), on demande alors le français (*dont q=0.4, inférieur à l'anglais*).

Question 4

Vous essayerez également d'utiliser les directives de gestion de caches telles que If-None-Match ou If-Modified-Since. Expliquez leur fonctionnement.

1-Utilisation de If-none-match :

curl -v http://lmbp.uca.fr/~barrel/tp1/test.html -H "If-none-match: 'a-50e2c3d384100;50e0fcd8c1485'"

L'argument passé après **If-none-match** est appelé ETag. Un ETag est un identifiant unique opaque assigné par le serveur web à chaque version d'une ressource accessible via une URL. Si l'ETag saisi dans l'en-tête **If-none-match** (qui correspond donc au fichier présent dans le cache) est le même que celui de la ressource distante, le serveur HTTP renvoie alors un code 304 signifiant que la version disponible sur le serveur est la même que la version située dans le cache de l'utilisateur. Dans le cas contraire, la nouvelle version de la ressource est envoyée (avec un code 200).

Utilisation de If-modified-since :

curl -v http://lmbp.uca.fr/~barrel/tp1/test.html -H "If-modified-since: Tue, 03 Feb 2015 10:07:32 GMT"

L'argument passé après **If-modified-since** est une date. Si la date saisie dans l'en-tête **If-modified-since** est égale (ou supérieure) à la date de la ressource distante, le serveur HTTP renvoie alors un code 304 signifiant que la version disponible sur le serveur est la même (ou est plus ancienne, ce qui semble naturellement peu probable…) que la version située dans le cache de l'utilisateur. Dans le cas contraire, la version située dans le cache est considérée comme obsolète et une nouvelle version de la ressource (plus récente) est envoyée (avec un code 200).

Question 5

Pour simuler Firefox 31.0 sous Windows :
curl -v http://lmbp.uca.fr/~barrel/tp1/page.php -H "User-agent: Mozilla/5.0 (Windows NT 5.1; rv:31.0) Gecko/20100101 Firefox/31.0"
Ou **:**
curl -v http://math.univ-bpclermont.fr/~barrel/tp1/page.php -A "Mozilla/5.0 (Windows NT 5.1; rv:31.0) Gecko/20100101 Firefox/31.0'Firefox/31.0"

Pour simuler une Nintendo Switch :
curl -v http://math.univ-bpclermont.fr/~barrel/tp1/page.php –A "Mozilla/5.0 (Nintendo Switch; WebApplet) AppleWebKit/601.6 (KHTML, like Gecko) NF/4.0.0.7.9 NintendoBrowser/5.1.0.15850

Pour simuler une Playstation 4 ver 1.52 :
curl -v http://math.univ-bpclermont.fr/~barrel/tp1/page.php -A "Mozilla/5.0 (PlayStation 4 1.52) AppleWebKit/536.26 (KHTML, like Gecko)"

Pour simuler une partie réseaux de Marvel vs Capcom 2 sur Sega Dreamcast :
curl -v http://math.univ-bpclermont.fr/~barrel/tp1/page.php -A "Mozilla/3.0 (DreamPassport/2.1; CAPCOM/MARVELVSCAPCOM2)"

Question 6

Enfin, essayez de générer les codes d'erreur 403, 404, 406 en expliquant la méthode utilisée.

Erreur 404 (Not Found)
curl -v http://lmbp.uca.fr/~barrel/tp1/toto.html
Note :La page toto.html n'existe pas.

Erreur 406 (Not Acceptable)
curl -v http://lmbp.uca.fr/~barrel/tp1/test.html -H "accept: image/png"
Note : On demande une option (format, langue...) qui n'est pas disponible pour la ressource demandée. Le serveur peut aussi être configuré pour, tout de même, envoyer une ressource avec des options par défaut.

Erreur 403 (Forbidden)
curl -v http://lmbp.uca.fr/~barrel/old
Note : L'utilisateur n'a pas le droit d'accéder au répertoire http://lmbp.uca.fr/~barrel/old

On pouvait aussi obtenir un code 403 en tapant :
curl -v http://lmbp.uca.fr/~barrel/tp1/test.html -X DELETE

Note : La méthode DELETE ne fait pas parti des commandes autorisées (voir Question 2). Dans certains cas, le serveur peut retourner un code 405 (Method not allowed)

curl -v http://lmbp.uca.fr/~barrel/tp1/test.html -X TOTO

Note : La méthode TOTO n'existe pas. Dans certains cas, le serveur peut retourner un code 501 (Method not Implemented)