4 – Protocole HTTP

I – Couche Application

La couche APPLICATION est chargée de _____ la communication entre _____ partageant un même protocole.

Les protocoles de la couche APPLICATION définissent donc comment les messages doivent être rédigés.

Exemples de protocoles :

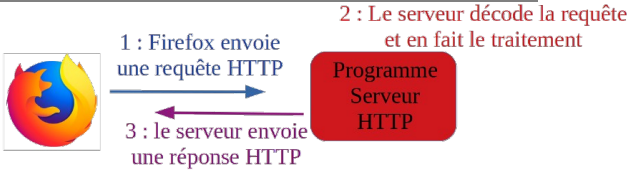
Protocole permettant la communication Client / Serveur sur le Web : _____

Protocoles permettant la communication Client / Serveur d'email : _____

Protocoles permettant le téléchargement et le téléversement : _____

II – Protocole HTTP

Le protocole HTTP veut dire _____



Il est basé sur une **communication entre deux programmes** :

Le _____ doit envoyer _____ dont la structure est

Une première ligne contenant 3 informations séparées par des espaces et finissant par un passage à la ligne (code ASCII 10).

L'**en-tête HTTP** : une suite de lignes contenant les valeurs d'une sorte de dictionnaire : **une clé** suivie de : et de la v**aleur associée**.

Une _____ signalant que l'**en-tête HTTP** est finie.

Le _____ du message : des données éventuelles dans la suite du message.

Il peut utiliser la méthode _____ : toutes les données nécessaires apparaissent dans l'URL.

Il peut utiliser la méthode _____ : les données de l'utilsateur sont transmises dans le corps du message.

Le _____ doit envoyer _____ dont la structure est

Une première ligne contenant 3 informations séparées par des espaces et finissant par un passage à la ligne (code ASCII 10).

L'**en-tête HTTP** : une suite de lignes contenant les valeurs d'une sorte de dictionnaire : **une clé** suivie de : et de la **valeur associée.**

Une _____ signalant que l'**en-tête HTTP** est finie.

Le _____ du message : code HTML, code CSS, octets d'une image...

Exemple de requête HTTP

```
.....DEBUT DE LA REQUETE HTTP

GET /act/archi/protocole-http-couche-application/ HTTP/1.1↲
Host: localhost:8000↲
Content-Length: ↲
Content-Type: text/plain↲
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; ...↲
Accept: text/html,application/xhtml+xml,application/xml;...↲
Accept-Language: fr,fr-FR;q=0.8,en-US;q=0.5,en;q=0.3↲
Accept-Encoding: gzip, deflate↲
Referer: https://www.infoforall.fr/act/archi/↲
Connection: keep-alive↲
Upgrade-Insecure-Requests: 1↲
↲
MESSAGE VIDE

.....FIN DE LA REQUETE HTTP
```

Exemple de réponse HTTP

```
.....DEBUT DE LA REPONSE HTTP

HTTP/1.1 200 OK↲
Content-Type: text/html; charset=utf-8↲
Content-Length: 36435↲
Vary: Cookie↲
Via: 1.1 alproxy↲
Date: Thu, 19 Mar 2020 12:54:47 GMT↲
↲
<!DOCTYPE html>↲
<html lang="fr">↲
<head>↲
...
: la suite, c'est donc le reste du fichier html !
...
</html lang="fr">↲

.....FIN DE LA REPONSE HTTP
```

III – Voir HTTP

Juste quelques manipulations avec Firefox.
On retiendra les codes courants :

200 → Pas de problème
30x → Voir le cache du navigateur
40x → Ressource non disponible
50x → La requête provoque une erreur

IV – Transférer des données : méthodes GET et POST

Méthode GET : les paramètres sont dans _____

Exemple : **?user=toto&password=1234**

Voici la codification du passage en GET :

- Présence de **/?** après la ressource demandée
- Présence d'un premier couple **user=toto**
- Présence optionnelle d'un **&** pour signaler qu'il y a un autre paramètre à la suite : **&password=1234**

  Tant qu'il y a des &, on continue...

Méthode POST : les paramètres sont dans _____

Exemple d'une requête HTTP POST envoyé à un serveur :

```
POST /act/tester-le-post/ HTTP/1.1↲
Host: www.infoforall.fr↲
Content-Length: 118↲
Origin: https://www.infoforall.fr↲
Referer: https://www.infoforall.fr/act/archi/protocole-http-couche-application/↲

question=sans_correction&numero=0

FIN DE LA REQUETE
```

Remarque finale : pour modifier une page, deux façons d'aborder le problème :

→ Utiliser un **script côté CLIENT** : un script _____ par exemple tourne sur l'ordinateur-client et modifie la page sans avoir besoin de communiquer avec le serveur une fois la page reçue.

→ Utiliser un **script côté SERVEUR** : un script _____ par exemple tourne sur l'ordinateur-serveur et renvoie un contenu HTML différent en fonction des paramètres reçus via la requête du client.