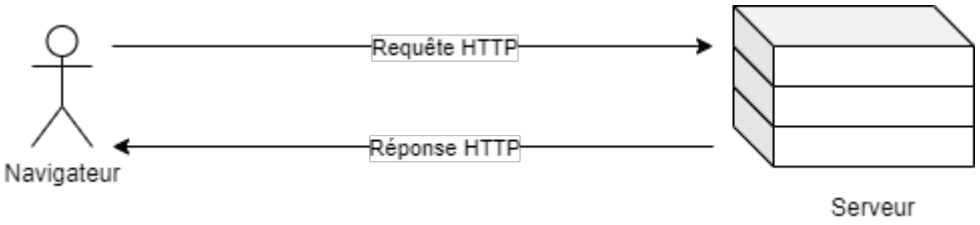
# Le Protocole HTTP

Le World Wide Web utilise principalement, le protocole HTTP (Hyper Text Transfer Protocole). Les protocole HTTP est un protocole client serveur. En effet, un client, votre navigateur, va émettre une requête à l'intention d'un serveur, qui va en retour lui fournir une réponse.

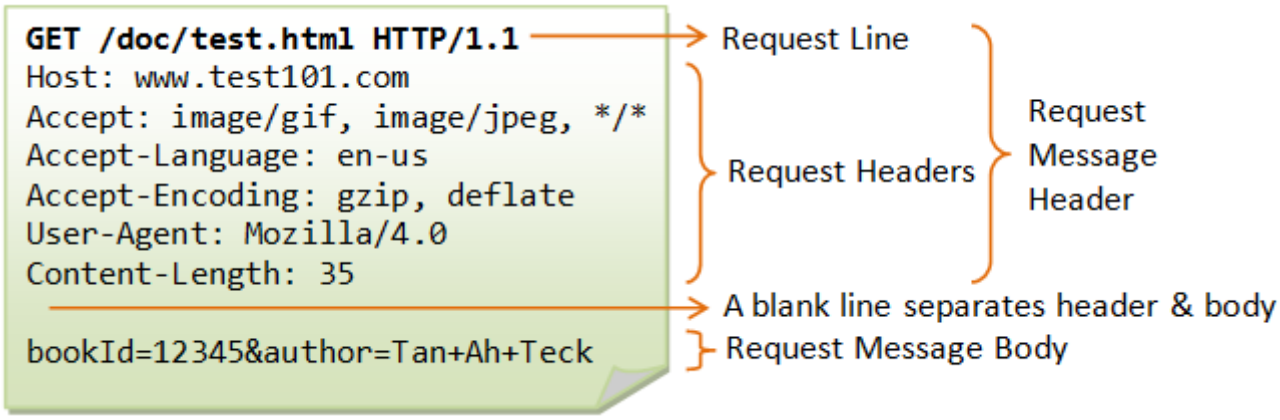



La connexion s'effectue via le protocole réseau TCP. Les échanges se font au format texte et sont très codifiés.

Etudions plus en détail la requête et la réponse HTTP.

# Requête HTTP

Une requête HTTP est constituée de plusieurs éléments




# Méthode de requête (Http method)

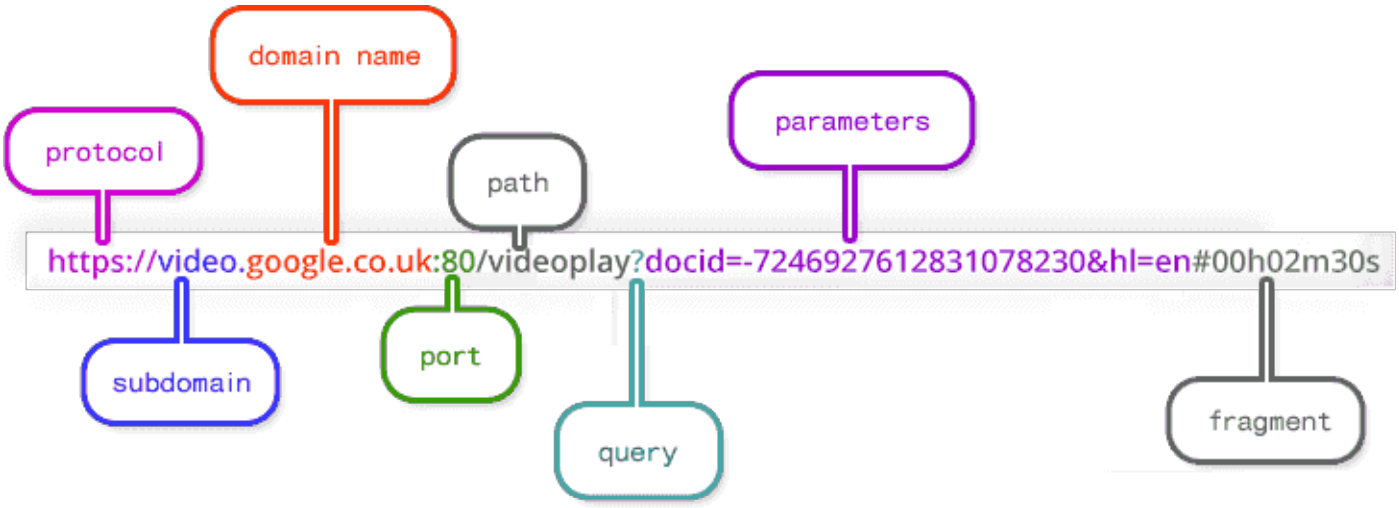
Le premier mot de la première ligne correspond à la méthode de requête (ex : GET, POST, PUT, DELETE ...), permet de décrire ce que va faire la requête, détaillé sur la page suivante.

# URL

Sur la première ligne, on retrouve également l'URL de la ressource intérogée, c'est le chemin de la ressource sur le serveur (par exemple */index.html*).

Cependant on peut ajouter des paramètres via l'URL en ajoutant un `?` ç la fin de l'URL. Les paramètre sont ensuite écrit dans l'URL à la suite sous la forme `nomParam=valeurParam` séparés par des `&`.

Exemple :
`https://example.com/posts?page=1&order=asc` → deux paramètres, `page` avec la valeur `1` et `order` avec la valeur `asc`




# Notion d'Endpoint

Un endpoint est la combinaison de la route d'une ressource et d'une méthode. On dit qu'une application possède un Endpoint pour une route et une méthode donnée. Par exemple :

GET `/api/posts` est un Endpoint

# Version HTTP

A la fin de la première ligne est affichée la version du protocole HTTP utilisée par cette requête (ex : HTTP/1, HTTP/1.1, HTTP/2).

# En tête (Headers)

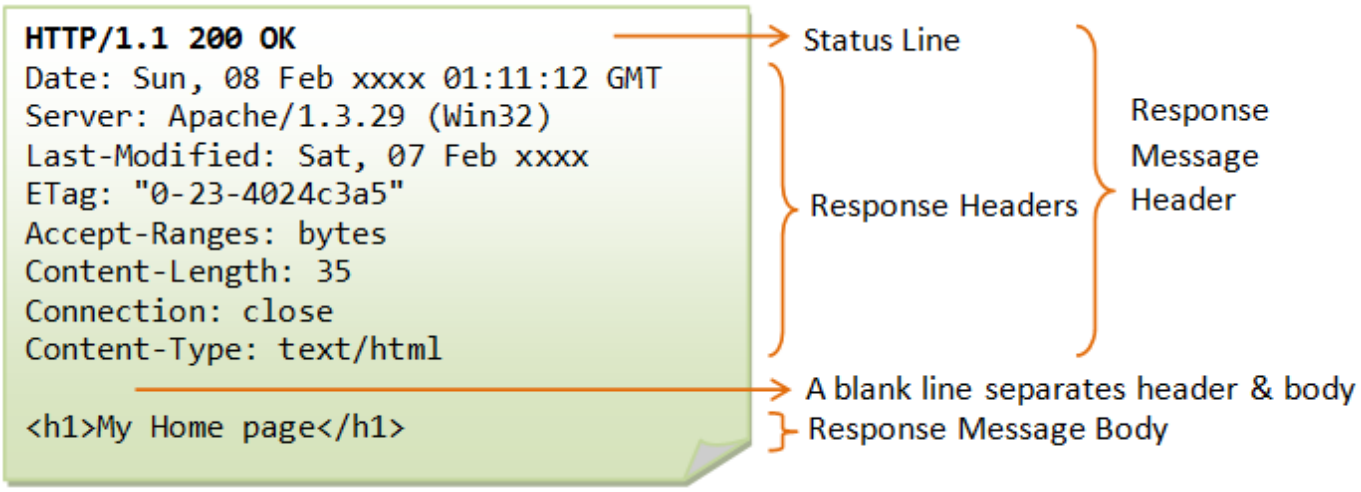
Ensuite, sur chaque ligne suivantes, on a les entêtes de la requête se présentent sous la forme d'un dictionnaire clé valeur qui renferme des méta-données sur la requête. Notamment utilisé pour l'authentification.

# Corps de la requête (Request Body)

Certaines méthodes HTTP permettent de passer des données au serveur dans la requête, ces données se retrouvent dans le Body de la requête. La longueur du body est définie dans le Header Content-Length.

# Réponse HTTP

La réponse HTTP est similaire à la requête, mais elle ne contient ni URL ni méthode, seulement des en-tête et un corps.




# Status HTTP

A la fin de la première ligne figure le status HTTP, qui informe sur le résultat de la requpête. Il est constitué d'un code (200, 404...), et d'un texte (OK, NNOT FOUND...)

# En tête (Headers)

Ensuite, sur chaque ligne suivantes, on a les entêtes de la requête se présentent sous la forme d'un dictionnaire clé valeur qui renferme des méta-données sur la réponse.

# Corps de la réponse (Response Body)

Souvent, le serveur renvoie des données en réponse, ces données se retrouvent dans le Body de la réponse. La longueur du body est définie dans le Header Content-Length.

Le corps de la réponse peut être en texte (HTML, JSON, XML, ...) ou en flux binaire.

# Principales méthodes HTTP.

## GET

Les requêtes avec la méthode GET permettent de récupérer des données du serveur, lire une ressource. Cette méthode d'autorise pas de *Request Body*. La ressource est retournée dans le body de la réponse. Les requêtes GET sont dites *sans effet de bord*, elle ne doivent rien changer à l'état du serveur.

## POST

Les requêtes avec la méthode POST permettent de créer une ressource sur le serveur. Le client envoie l'état désiré de la ressource dans le *Request Body*, l'état de la ressource après création est retournée dans le *Response Body*.

## PUT

Les requêtes avec la méthode PUT permettent de remplacer des données sur le serveur, le client envoie le nouvel état de la ressource dans le *Request Body* de la requête. L'état modifié de la ressource doit être retrourné dans le *Response Body*.

## DELETE

Les requêtes avec la méthode DELETE permettent de supprimer des données du serveur. Cette méthode n'autorise pas de *Request Body* ni de *Response Body*. Si la *Response* est en succès, cela veut dire que la ressource n'existe plus.

# Codes de réponse HTTP

Le protocole HTTP présente différents types de codes :

- 1XX : Informations
- 2XX : Succès
- 3XX : Redirection

- 4XX : Erreurs clients (requête invalide)
- 5XX : Erreurs serveurs (échec de traitement)

# Quelques exemples de succès

- 200 OK : la requête a été traitée avec succès
- 201 Created : la requête a été traitée avec succès et a créé une ressource (la réponse possède un Header "Location" qui contient l'adresse de la ressource créée
- 202 Accepted : la requête a été acceptée et son traitement est en cours
- 204 No Content : la requête a été traitée avec succès mais la réponse ne contient pas de Body.

# Quelques exemples de redirection

Les Response avec un code redirection possèdent un Header "Location" contenant le nouvel emplacement de la ressource.

- 301 Moved Permanently : la ressource a été deplacée pour de bon
- 307 Temporary Redirect : la ressource a été deplacée temporairement
- 302 Found : pointe vers une ressource à GET

# Quelques exemples d'erreur client

- 400 Bad Request : Mauvais format de requête
- 401 Unauthorized : L'authentification à échouée
- 403 Forbidden : Vous n'avez pas accès à cette ressource
- 404 Not Found : La ressource n'existe pas
- 406 Not Acceptable : Le format demandé n'est pas disponible
- 418 I'm A Teapot : Le serveur est un théière (poisson d'avril de la RFC en 1997)

# Quelques exemples d'erreurs serveur

- 500 Internal error : Erreur de traitement coté serveur
- 501 Not Implemented : Cette ressource devrait marcher mais elle n'est pas finie
- 503 Service Unavailable : Le service est en maintenance ou crashé

---

Revision #10
Created 2020-09-02 19:45:17 UTC by Arsène Lapostolet
Updated 2021-09-06 07:30:33 UTC by Arsène Lapostolet