ANNÉE PASSERELLE EN INGÉNIERIE

Chapitre 13 — HTTP

Introduction à la Téléinformatique

Vincent Audergon · HEIA-FR

Année académique 2026-2027

Plan du chapitre

01 Le Web et HTTP
02 URI et URL
03 Requêtes et réponses HTTP
04 HTTP/1.1 vs HTTP/2

Objectifs du chapitre

HEIA-FR
HTA-FR

À l'issue de ce chapitre, vous saurez :

- **décrire** l'utilité du protocole HTTP dans la communication client-serveur ;
- **distinguer** URI et URL et connaître la structure d'une URL ;
- **lister** les principales méthodes HTTP et leur rôle ;
- **interpréter** une requête et une réponse HTTP, y compris les codes de statut ;
- **décrire** les différences entre HTTP/1.1, HTTP/2 et HTTP/3.

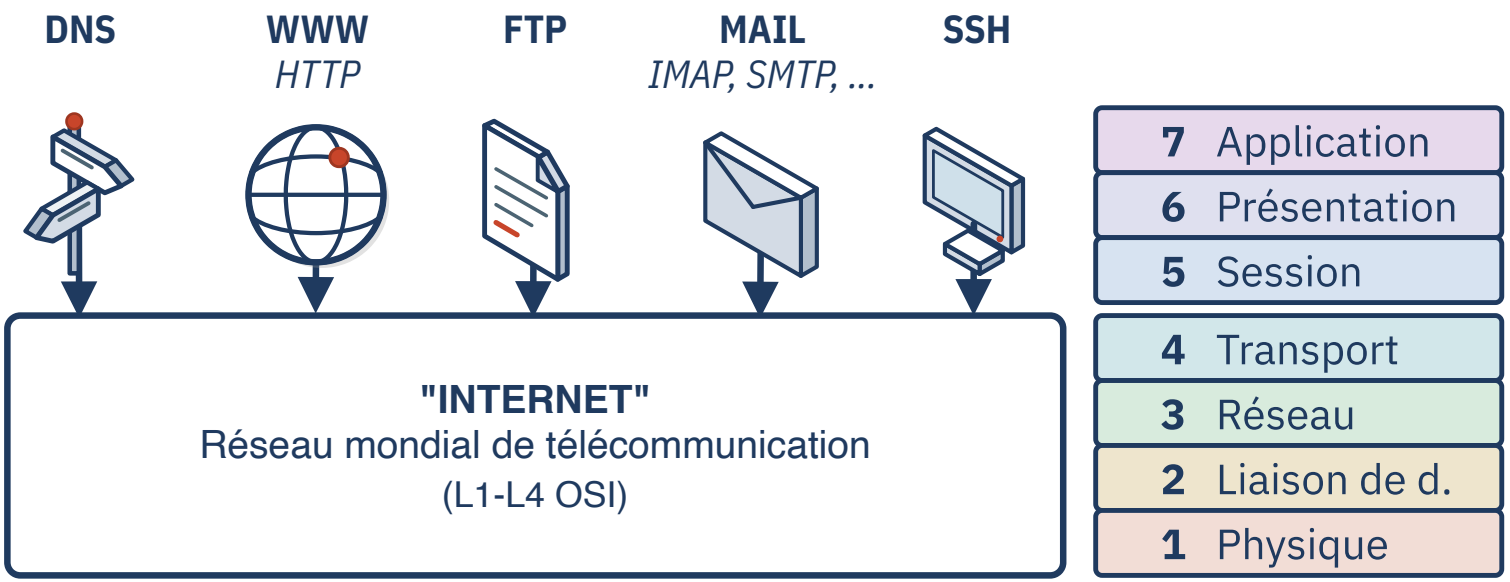
SECTION

01

Le Web et HTTP

01 **Le Web et HTTP**
02 URI et URL
03 Requêtes et réponses HTTP
04 HTTP/1.1 vs HTTP/2

Internet, ce n’est pas que le Web



Le **World Wide Web** (WWW) est un service parmi d’autres qui est servi par **Internet** : email (SMTP/IMAP), transfert de fichiers (FTP), DNS, SSH…

HyperText et HTTP

DÉFINITION · HTTP

HyperText Transfer Protocol : protocole applicatif client-serveur qui permet de transférer des ressources (pages, images…) sur le Web.

EXEMPLE · HYPERTEXT

Du grec *hyper*, « au-delà ». L'idée : des documents reliés entre eux par des **hyperliens**, cliquables pour naviguer d'une ressource à l'autre. Popularisé par le HTML (HyperText Markup Language).

Le client (navigateur) envoie des **requêtes** ; le serveur répond avec les **ressources** demandées.

? QUESTION 1

Le DNS, le FTP et SSH tournent-ils « sur » le Web, ou « sur » Internet ?

RÉPONSE
Sur **Internet**. Le Web (HTTP) n'est qu'un service parmi d'autres qui utilisent l'infrastructure d'Internet, au même titre que DNS, FTP ou SSH.

Internet $\neq$ **Web**, mais **Web** $\subset$ **Internet**.

SECTION

02 —

URI et URL

01 Le Web et HTTP
02 **URI et URL**
03 Requêtes et réponses HTTP
04 HTTP/1.1 vs HTTP/2

Identifier une ressource

HEIA-FR
HTA-FR

DÉFINITION · URI

Uniform Resource Identifier : chaîne de caractères qui identifie de manière unique une ressource.

DÉFINITION · URL

Uniform Resource Locator : une URI qui précise en plus la **localisation** de la ressource et le **protocole** pour y accéder.

```
https://www.example.com:443/chemin/vers/ressource?query=1
```

Anatomie d'une URL


```
<schéma>://<hôte>[:<port>]/<chemin>[?<paramètres>][#<fragment>]
```

- **Schéma** : protocole à utiliser (`http`, `https`, `ftp`…)
- **Hôte** : nom de domaine ou IP (IPv6 entre crochets : `[2001:db8::1]`)
- **Port** (optionnel) : port well-known par défaut (80 / 443 pour HTTP/HTTPS)
- **Chemin** : accès à la ressource sur le serveur, suivi éventuellement de paramètres (`?`), puis d'un fragment (`#`) pour un ancrage dans la page.

> **ATTENTION**
> Espaces et caractères spéciaux (`#`, `?`…) doivent être **encodés** : un espace devient `%20` (un `%` suivi de deux chiffres hexadécimaux).

? QUESTION 2

`http://www.heia-fr.ch/travaux pratiques/` — cette URL est-elle valide ?

> **RÉPONSE**
> **Non** : le chemin contient une espace non encodée. Il faudrait écrire `travaux%20pratiques`.

SECTION

03 —

Requêtes et réponses HTTP

01 Le Web et HTTP
02 URI et URL
03 **Requêtes et réponses HTTP**
04 HTTP/1.1 vs HTTP/2

Ports et versions

HTTP fonctionne en **texte brut**, sur le modèle client-serveur.

- Well-known ports : **80** (HTTP), **443** (HTTPS).
- **HTTP/1.1** (1999, RFC 9112), encore répandu.
- **HTTP/2** (2015, RFC 9113), multiplexage, format binaire.
- **HTTP/3** (2022, RFC 9114), support de **QUIC** (au-dessus d'UDP).

> **REMARQUE**
> Depuis 2022, la sémantique commune (méthodes, statuts, en-têtes) est définie à part, dans le RFC 9110.

Structure d'une requête / réponse

HEIA-FR
HTA-FR

Requête

- **Méthode** (GET, POST…)
- **URL** de la ressource
- **En-têtes**
- **Corps** (optionnel)

Réponse

- **Code de statut**
- **En-têtes**
- **Corps** (optionnel)

REMARQUE
En-têtes et corps sont séparés par une **ligne vide** ; chaque en-tête occupe une ligne.

Un échange HTTP en clair


```
GET /index.html HTTP/1.1
Host: www.example.com
Accept: text/html
```

```
HTTP/1.1 200 OK
Content-Type: text/html

<!DOCTYPE html>
<html>
<head><title>Example</title></head>
<body><h1>Hello, world!</h1></body>
</html>
```

Un échange HTTP en clair

> **ATTENTION**
> Ne pas confondre **HTTP** (le protocole de transport) et **HTML** (le format du contenu transporté) : deux concepts complémentaires mais distincts.

Le HTML est un langage de marquage, qui, affiché en texte brut, ressemble à ceci :

```
<h1>Hello, world!</h1>
```

Mais dans un navigateur, il est interprété et rendu graphiquement.

Hello, world!

Les méthodes principales HTTP

Méthode	Rôle
`GET`	Demander une ressource. Paramètres dans l'URL (`?...`).
`POST`	Envoyer des données pour **créer** une ressource (corps de requête).
`HEAD`	Comme `GET`, mais sans le corps — juste les en-têtes.
`PUT`	Créer ou **remplacer** une ressource à l'URL donnée.
`DELETE`	Supprimer la ressource à l'URL donnée.

ASTUCE
`GET` = paramètres visibles dans l'URL. `POST` = données dans le corps, invisibles dans l'URL, utile pour les formulaires et les données sensibles.

Les codes de statut

Regroupés en cinq classes, selon le premier chiffre :

Classe	Signification
1xx	Information, ex. `100 Continue`
2xx	Succès, ex. `200 OK`, `201 Created`
3xx	Redirection, ex. `301 Moved Permanently`, `302 Found`
4xx	Erreur **client**, ex. `400 Bad Request`, `404 Not Found`
5xx	Erreur **serveur**, ex. `500 Internal Server Error`

Observer une requête réelle



REMARQUE

Les outils de développement du navigateur (F12, onglet *Réseau*) permettent d'inspecter chaque requête : méthode, statut, version HTTP, en-têtes…

? QUESTION 3

Vous demandez la page web suivante, avec un chemin inventé : `https://www.heia-fr.ch/j_ai_invente_ce_chemin.html`. Quel code de statut vous attendez-vous à recevoir ?

> **RÉPONSE**
> Classe **4xx** (erreur côté client), précisément `404 Not Found`. Il n'y a très probablement pas de ressource à ce chemin sur le serveur.

SECTION
04

HTTP/1.1 vs HTTP/2

01 Le Web et HTTP
02 URI et URL
03 Requêtes et réponses HTTP
04 **HTTP/1.1 vs HTTP/2**

Ce que HTTP/2 améliore

HTTP/1.1

- Une requête à la fois par connexion
- Format **textuel**
- Simple à lire « à l'œil nu »

HTTP/2

- **Multiplexage** : plusieurs requêtes/réponses en parallèle sur une connexion
- Format **binaire** : compression, efficacité
- En-têtes compressés, priorisation des flux

REMARQUE

Server Push (envoi proactif de ressources par le serveur) faisait partie de HTTP/2, mais a été désactivé dans la plupart des navigateurs modernes (2022-2024), jugé peu efficace en pratique.

? QUESTION 4

Pourquoi le multiplexage de HTTP/2 réduit-il la latence perçue lors du chargement d'une page web contenant de nombreuses ressources (images, scripts, styles) ?

RÉPONSE

En HTTP/1.1, une seule requête à la fois par connexion : les ressources sont récupérées en série (ou via plusieurs connexions parallèles, coûteuses). En HTTP/2, toutes les requêtes partagent une connexion et sont traitées en parallèle : plus d'attente en file, chargement plus rapide.

Démo HTTP/1.1 vs HTTP/2

- https://www.tunetheweb.com/performance-test-360/

Synthèse

CE QU'IL FAUT RETENIR

- **HTTP** est le protocole client-serveur du Web. URL = localisation d'une ressource.
- Une requête = **méthode** + URL + en-têtes (+ corps) ; une réponse = **code de statut** + en-têtes (+ corps).
- Méthodes clés : **GET**, **POST**, **PUT**, **DELETE**, **HEAD**.
- Codes de statut par classe : **2xx** succès, **3xx** redirection, **4xx** erreur client, **5xx** erreur serveur.
- **HTTP/2** apporte multiplexage et format binaire par rapport à HTTP/1.1. **HTTP/3** tourne sur QUIC (UDP).

Des questions ?

Sources

Les sources utilisées pour ce cours (normes, documentation officielle, articles, ouvrages de référence) sont citées et listées intégralement dans la **bibliographie du support de cours écrit**, disponible sur le site du cours :

https://api-teleinfo.isc.heia-fr.ch

