Université Paris Nanterre
Université Sorbonne nouvelle
Institut national des langues et civilisations orientales

pluriTAL

Master Traitement Automatique des Langues

http://plurital.org

Programmation et projet encadré - L7TI005

Web: HTML, HTTP, récupérer des pages

Yoann Dupont yoann.dupont@sorbonne-nouvelle.fr
Pierre Magistry pierre.magistry@inalco.fr

2024-2025

Université Sorbonne-Nouvelle
INALCO
Université Paris-Nanterre

Les enjeux de cette portion de cours :

- comprendre ce qu'est une page web
- les outils pour la récupérer sur sa machine
- débuter l'écriture de la chaîne de traitement
  - étape 1 : faire des aspirations de pages web à partir d'une liste d'URL et en extraire le contenu textuel
  - pourra/devra être fait sur des URL "bidons"

HTML

HTML, c'est quoi

HTML (**H**yper**T**ext **M**arkup **L**anguage) est un langage de balisage pour représenter des pages web. Format reconnu par tous les navigateurs.

HTML, c'est quoi

HTML (**H**yper**T**ext **M**arkup **L**anguage) est un langage de balisage pour représenter des pages web. Format reconnu par tous les navigateurs.

Permet de :

- structurer l'information
- enrichir le texte
- donner des indications pour interagir avec l'utilisateur

HTML, c'est quoi

HTML (**H**yper**T**ext **M**arkup **L**anguage) est un langage de balisage pour représenter des pages web. Format reconnu par tous les navigateurs.

Permet de :

- structurer l'information
- enrichir le texte
- donner des indications pour interagir avec l'utilisateur

Aujourd'hui, on survolera HTML, plus de détail à venir.

Exemple de balisage : www.perdu.com

C'est une page simple, utile pour un exemple.

Exemple de balisage : www.perdu.com

C'est une page simple, utile pour un exemple.

```
<html>
  <head>
    <title>Vous Etes Perdu ?</title>
  </head>
  <body>
    <h1>Perdu sur l'Internet ?</h1>
    <h2>Pas de panique, on va vous aider</h2>
    <strong><pre> * <—— vous &ecirc;tes ici</pre></strong>
  </body>
</html>
```

Exemple de balisage : www.perdu.com

C'est une page simple, utile pour un exemple.

```
<html>
  <head>
    <title>Vous Etes Perdu ?</title>
  </head>
  <body>
    <h1>Perdu sur l'Internet ?</h1>
    <h2>Pas de panique, on va vous aider</h2>
    <strong><pre> * <—— vous &ecirc;tes ici</pre></strong>
  </body>
</html>
```

Où :

- head : l'entête du fichier (avec les métadonnées)
- body : le corps du fichier (avec le contenu textuel et la structure)

Dans le cadre de PPE

Dans le cadre de PPE, on va :

- retirer tout le métatexte de pages web pour en extraire le contenu textuel
- analyser ce contenu textuel pour en déduire un usage (linguistique de corpus)
- créer (plus tard) nos propres pages web pour présenter les procédés et résultats

Comment on dialogue avec internet ?

HTTP (HyperText Transfer Protocol) est un protocole créé pour communiquer sur le WWW. Il a été créé par Tim Berners-Lee, l'un des créateurs du World Wide Web. Il a inventé notamment HTML, HTTP et la notion d'URL.

Comment on dialogue avec internet ?

HTTP (HyperText Transfer Protocol) est un protocole créé pour communiquer sur le WWW. Il a été créé par Tim Berners-Lee, l'un des créateurs du World Wide Web. Il a inventé notamment HTML, HTTP et la notion d'URL.

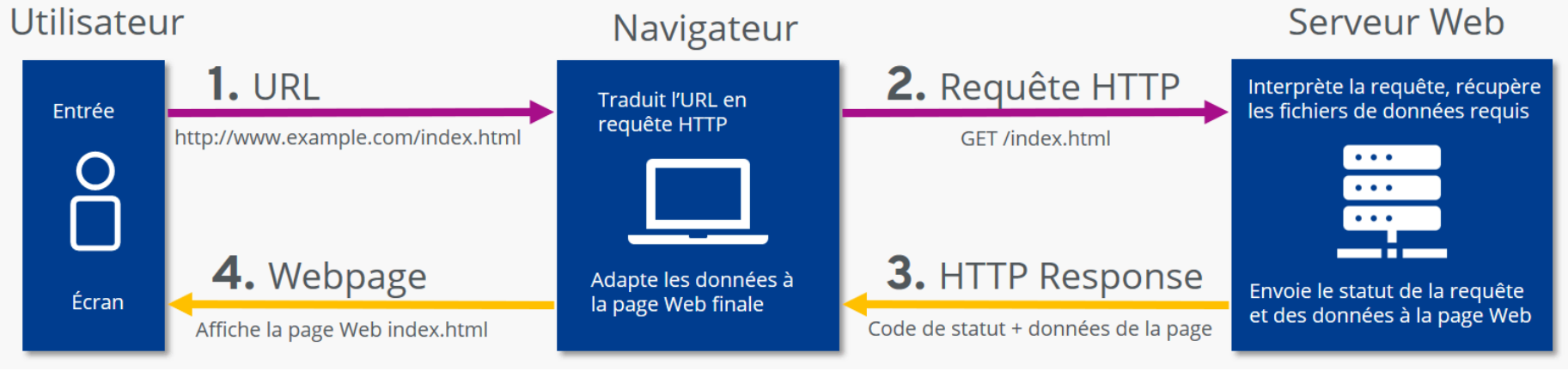

Figure 1: Schématisation de HTTP (source : www.ionos.fr)

Comment on dialogue avec internet ?

HTTP (HyperText Transfer Protocol) est un protocole créé pour communiquer sur le WWW. Il a été créé par Tim Berners-Lee, l'un des créateurs du World Wide Web. Il a inventé notamment HTML, HTTP et la notion d'URL.

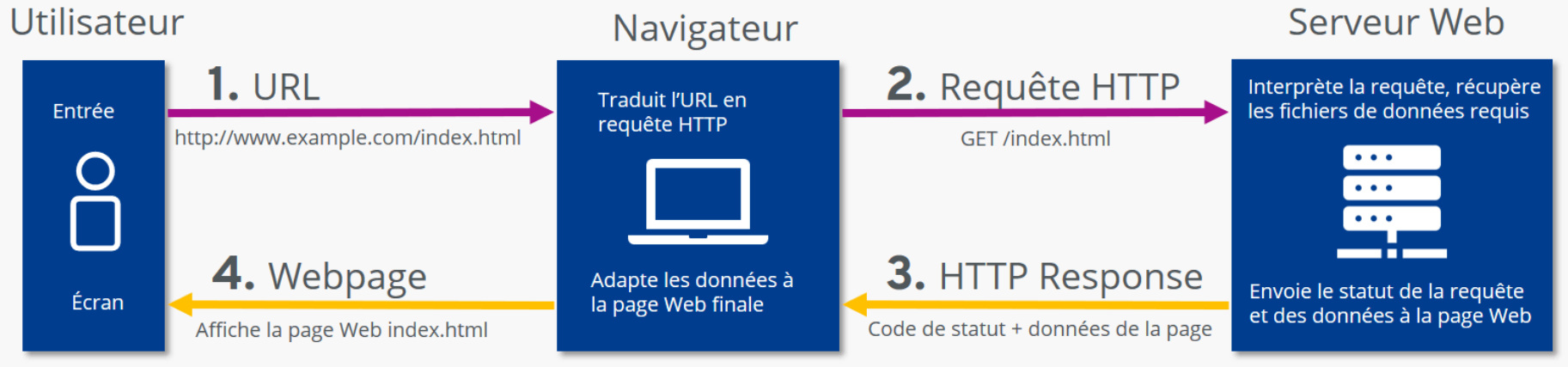

Figure 1: Schématisation de HTTP (source : `www.ionos.fr`)

Ce qui nous intéresse tout particulièrement :

- l'étape 3 est la réponse à votre requête (réussite/échec, etc.).
- l'étape 4 est l'interprétation de la page par votre navigateur.

Nomenclature des codes HTTP

Lorsqu'on utilise des outils qui requêtent des pages web (ex: navigateur), le code de statut (3. *HTTP Response*) permet d'avoir une idée du résultat d'une requête.

Source : https://fr.wikipedia.org/wiki/Liste_des_codes_HTTP

Nomenclature des codes HTTP

Lorsqu'on utilise des outils qui requêtent des pages web (ex: navigateur), le code de statut (3. *HTTP Response*) permet d'avoir une idée du résultat d'une requête.

Permettent d'avoir le statut de la réponse :

- 1xx : information
- 200 : réussite
- 3xx : redirections
- 4xx : erreurs du client
- 5xx : erreurs du serveur

Ces codes sont utilisés par les navigateurs pour savoir quoi faire.
On utilisera ces codes pour (in)valider les requêtes dans les fichiers d'URL.

Source : https://fr.wikipedia.org/wiki/Liste_des_codes_HTTP

Lynx

Présentation

Lynx[1] est un navigateur web en terminal. Cela lui donne des propriétés particulièrement intéressantes (affichage dépouillé).

[1] Site web : https://lynx.browser.org
[2] Installer homebrew : https://osxdaily.com/2018/03/07/how-install-homebrew-mac-os/
[3] Installer Lynx : https://osxdaily.com/2011/07/26/get-lynx-for-mac-os-x-10-7-lion/

Présentation

Lynx[1] est un navigateur web en terminal. Cela lui donne des propriétés particulièrement intéressantes (affichage dépouillé).

Installer sur Linux (terminal) :

```
sudo apt install lynx
```

Installer sur Mac OS X avec `homebrew`[2,3] (terminal) :

```
brew install lynx
```

[1] Site web : https://lynx.browser.org
[2] Installer homebrew : https://osxdaily.com/2018/03/07/how-install-homebrew-mac-os/
[3] Installer Lynx : https://osxdaily.com/2011/07/26/get-lynx-for-mac-os-x-10-7-lion/

Démonstration

Récupérer le contenu d'une page avec Lynx

Lynx permet de récupérer et afficher dans le terminal (sans navigation) une page web avec uniquement du texte et des liens.

Deux options permettent de :

1. récupérer le contenu textuel d'une page pour l'afficher (sans navigation)
2. retirer la liste des liens d'une page à l'affichage

Comment les chercher et quelles sont-elles ?

Récupérer le contenu d'une page avec Lynx

Lynx permet de récupérer et afficher dans le terminal (sans navigation) une page web avec uniquement du texte et des liens.

Deux options permettent de :

1. récupérer le contenu textuel d'une page pour l'afficher (sans navigation)
2. retirer la liste des liens d'une page à l'affichage

Comment les chercher et quelles sont-elles ?

5 minutes de recherche puis on prend les réponses !

Réponses

Deux options permettent de :

1. récupérer le contenu textuel d'une page pour l'afficher (sans navigation)

2. retirer la liste des liens d'une page à l'affichage

Réponses

Deux options permettent de :

1. récupérer le contenu textuel d'une page pour l'afficher (sans navigation)
   → option **-dump**
2. retirer la liste des liens d'une page à l'affichage

Réponses

Deux options permettent de :

1. récupérer le contenu textuel d'une page pour l'afficher (sans navigation)
   → option **-dump**
2. retirer la liste des liens d'une page à l'affichage
   → option **-nolist**

wget, cURL

Pourquoi

`wget` et `cURL` (*client URL Request Library*) sont deux commandes qui vont pouvoir nous permettre de récupérer des pages web sans passer par un navigateur. Les deux commandes ont des différences qui les rendent intéressantes, même si on privilégiera `cURL`.

Pourquoi

`wget` et `cURL` (*client URL Request Library*) sont deux commandes qui vont pouvoir nous permettre de récupérer des pages web sans passer par un navigateur. Les deux commandes ont des différences qui les rendent intéressantes, même si on privilégiera `cURL`.

Dans notre cas, la différence principale entre les deux commandes et que `wget` écrit dans un fichier et `cURL` écrit dans le terminal.

Installer cURL et wget

Sous Linux via le terminal (wget normalement déjà installé) :

```
sudo apt install curl
```

Sous Mac OS X avec homebrew :

```
brew install wget
brew install curl
```

La commande cURL

Lancer la commande cURL :

```
curl <URL>
```

Où <URL> est une URL vers une page sur le web.

[4] "tiret i majuscule", équivalent à l'option "head"

La commande cURL

Lancer la commande cURL :

```
curl <URL>
```

Où <URL> est une URL vers une page sur le web.

Quelques options utiles :

- -i : va donner des informations sur l'interaction avec le serveur
- -L : suit les redirections
- -o <fichier> : indique un <fichier> de sortie
- d'autres options à voir par vous-même : -I[4], -w, -s

[4] "tiret i majuscule", équivalent à l'option "head"

Quelques tests avec cURL I

```
curl https://www.google.com
curl https://fr.wikipedia.org/wiki/Chat
curl https://www.github.com
curl http://www.plurital.org
```

Quelques tests avec cURL II

Regardons déjà les cas où ça va (ou ça fait semblant d'aller ?) :

```
curl -i https://www.perdu.com
curl -i https://www.google.com
```

Quelques tests avec cURL II

Regardons déjà les cas où ça va (ou ça fait semblant d'aller ?) :

```
curl -i https://www.perdu.com
curl -i https://www.google.com
```

Et les cas où ça ne va pas :

```
curl -i http://www.github.com
curl -i http://www.plurital.org
```

Quelques tests avec cURL III

Ce qu'on voit avec ces commandes, ce sont des informations sur la réponse du serveur. Deux lignes nous intéressent tout particulièrement :

- La première de chaque bloc d'entête : `HTTP/1.1 XZY <message>`
- Celle souvent juste après : `content-type: <informations>`

Quelques tests avec cURL III

Ce qu'on voit avec ces commandes, ce sont des informations sur la réponse du serveur. Deux lignes nous intéressent tout particulièrement :

- La première de chaque bloc d'entête : `HTTP/1.1 XZY <message>`
- Celle souvent juste après : `content-type: <informations>`

Il s'agit d'un résumé de la communication entre le client (nous) et le serveur (le site hébergé quelque part sur le net). Ces informations sont issues du protocole HTTP.

Pour la prochaine fois.

Consignes

- Si vous êtes en retard sur les semaines précédentes : faites les exercices manquants !
- Il faudra créer un nouveau dépôt GitHub pour le groupe avant.
  - Ajouter les membres du projet en collaborateur(ice)s.
  - Chaque contributeur(ice) modifiera le README.md *du groupe* en ajoutant son nom et le lien vers son GitHub personnel.
  - Indiquer dans le README.md *personnel* le lien vers le GitHub du groupe[5].
- Tout déposer jusqu'au mercredi 30/10 à 23h59 via icampus / mail.
  - Créer un tag `gitbash-fin`

[5] Le dépôt Github du groupe doit être public.