Une rapide introduction à HTML / CGI

Patrick Fuchs

Université Paris 7

Equipe de Bioinformatique Génomique et Moléculaire

PLAN

1. Généralités
2. Les Bases d'HTML
3. Outils HTML avancés
4. HTML dynamique / Interface CGI

1) Définition du HTML

- HTML = Hyper Text Mark up Language
  - langage à balises (« tags »)
  - langage qui contient des liens « hyper-text »
- Le HTML habille du texte (~ LATEX)
- Le HTML permet de partager des données situées à des endroits différents
- Ordre de consultation des données laissé à l'utilisateur
- **HTML ≠ langage de programmation**

le SGML

SGML (Standard Generalized Markup Language)

= norme pour l'échange des documents électroniques

HTML

sous-ensemble de SGML

(actuellement version 4)

XHTML = HTML "propre"
selon la norme w3c

http://www.w3c.org

XML

(Extensible Markup Language)

-> meta-langage à balises

-> version simplifiée de SGML

-> permet de définir ses propres balises

Rappel

- Serveur (mise à disposition de pages web)
- Navigateur (Netscape, Internet Explorer…)
- Accession au serveur par un navigateur via l'URL (Universal Resource Locator) ~ adresse :

`protocole://adresse-machine/rep/sous-rep/document`

  - protocole :
    - http (Hyper Text Transfert Protocol) = protocole le plus courant = protocole de transfert de page html

    e.g : http://www.python.org
    - ftp (File Transfert Protocol) = protocole de transfert de fichiers
    - etc…
  - adresse-machine :
    - numéro IP (Internet Protocol) : e.g. 131.246.50.201
    - adresse DNS (Domain Name Server) : e.g. www.ibm.com

PLAN

1. Généralités
2. Les Bases d'HTML
3. Outils HTML avancés
4. HTML dynamique / Interface CGI

Quelques sites d'intérêt

- le site de Dave Ragett @ w3c :

http://www.w3.org/MarkUp/Guide/Overview.html

- le site d'Eric Larcher :

http://larcher.com/

- le site de Philippe Moreau :

http://www.u-picardie.fr/~philippe/aideHTML/

- l'Electronic Text Center de la bibliothèque de l'Université de Virginie :

http://etext.lib.virginia.edu/helpsheets/helpsheets.html

Les Balises

- balise d'ouverture : `<balise>`
- balise de fermeture : `</balise>`
- encadre le texte :

`<balise>texte</balise>`

-> le couple de balise prend effet au niveau de : texte

- insensibles à la casse (`<BALISE>` = `<balise>`)

Rq : il existe des balises qui n'ont pas besoin d'être fermées

Structure d'une page HTML

```
<HTML>
<HEAD>
<TITLE>Titre du
  document</TITLE>
</HEAD>
<BODY>
...
...
</BODY>
</HTML>
```

- <HTML>…</HTML> : encadre le document
- <HEAD>…</HEAD> : en-tête de la page
- <TITLE>…</TITLE> : donne un titre à la page (moteurs de recherches)
- <BODY>…</BODY> : corps de la page

Rq : fichier HTML = fichier texte

Rq2 : fichier HTML = insensible aux espaces et retours-chariots sauf dans une balise <pre>

Styles d'écriture et séparateurs

souligné : <u>…</u>

gras : …

italique : …

exposant : […]

indice : _…

Niveaux de titre :

niveau 1 : <h1>…</h1>

niveau 2 : <h2>…</h2>

…

niveau6 : <h6>…</h6>

Séparateurs : nouveau paragraphe : <p>

retour à la ligne :

ligne horizontale : <hr>

Rq : pas de balise fermante pour les séparateurs

Ecriture formatée

la balise <pre>…</pre> : idéale pour les fichiers en colonne

code

```
<pre>
ATOM      5   CA   ARG      1       -0.977  -9.308  24.396
ATOM     22   CA   GLY      2       -1.731  -6.186  23.074
ATOM     27   CA   ASN      3        0.654  -5.826  20.081
ATOM     38   CA   VAL      4       -0.293  -8.716  17.970
</pre>
```

rendu

```
ATOM      5   CA   ARG      1       -0.977  -9.308  24.396
ATOM     22   CA   GLY      2       -1.731  -6.186  23.074
ATOM     27   CA   ASN      3        0.654  -5.826  20.081
ATOM     38   CA   VAL      4       -0.293  -8.716  17.970
```

Listes d'éléments

Listes numérotées :

```
<OL>
        <LI>élément 1,
        <LI>élément 2,
        <LI>élément 3.
</OL>
```

→

1. élément 1,
2. élément 2,
3. élément 3.

Listes à puces :

```
<UL>
        <LI>élément 1,
        <LI>élément 2,
        <LI>élément 3.
</UL>
```

→

- élément 1,
- élément 2,
- élément 3.

Rq : possibilité de faire des listes imbriquées

Liens hyper-textes

- Liens externes :
  - …

  e.g. : `<a href="http://www.python.org">python</a>`

- Liens internes :
  - définition d'une accroche : …
  - lien vers cette accroche : …

- URL absolues / relatives :
  - absolues : en général liens vers d'autres sites
  - relative : à utiliser au sein du même site (utile lorsqu'on déplace plusieurs pages relatives les unes aux autres)

Liens spéciaux

- Lien vers un site FTP :

  …

- Lien vers un mail :

  …

Insertion d'image

- **Balise img** : <img src="mon_image">

  (possibilité de mettre une URL pointant sur une image)

- **possibilité de mettre un lien sur une image** :

<img src="mon_image">

- **alignement de l'image** :

<img src="mon_image" align="center">

-> left / right / center

Rq : le champ align s'utilise aussi sur les balises associées à du texte (<p>, <li>, <h1>...)

Caractères spéciaux

- syntaxe générale : &code;

Æ	Æ	Á	Á	Â	Â	À	À	Å	Å	Ã	Ã	Ä	Ä	Ç	Ç
Ð	Ð	É	É	Ê	Ê	È	È	Ë	Ë	Í	Í	Î	Î	Ì	Ì
Ï	Ï	Ñ	Ñ	Ó	Ó	Ô	Ô	Ò	Ò	Ø	Ø	Õ	Õ	Ö	Ö
Þ	Þ	Ú	Ú	Û	Û	Ù	Ù	Ü	Ü	Ý	Ý	á	á	â	â
æ	æ	à	à	å	å	ã	ã	ä	ä	ç	ç	¢	¢	©	©
º	°	é	é	ê	ê	è	è	ð	ð	ë	ë	>	>	í	í
î	î	¡	¡	ì	ì	¿	¿	ï	ï	&	&	<	<	µ	µ
·	·			¬	¬	ñ	ñ	ó	ó	ô	ô	ò	ò	ø	ø
õ	õ	ö	ö	¶	¶	±	±	£	£	"	"	»	»	®	®
§	§			ß	ß	þ	þ	ú	ú	û	û	ù	ù	ü	ü
ÿ	ÿ	¥	¥												

Rq : l'utilisation directe des accents peut poser des problèmes sur les navigateurs dans les pays étrangers

Comment générer du HTML ?

- **outils WYSIWYG** (what you see is what you get):

  e.g : Netscape Composer, Front Page,…Word…

  - avantage : visualisation directe
  - inconvénients : qualité et maintenance du code
- **éditeurs de texte** (avec "syntax highlighting")

  e.g: nedit, vi, emacs
- **l'indentation** rend le code HTML plus lisible

PLAN

1. Généralités
2. Les Bases d'HTML
3. Outils HTML avancés
4. HTML dynamique / Interface CGI

Outils HTML avancés

1. **les tableaux**
2. les cadres
3. les feuilles de style en cascade
4. les formulaires

Les Tableaux

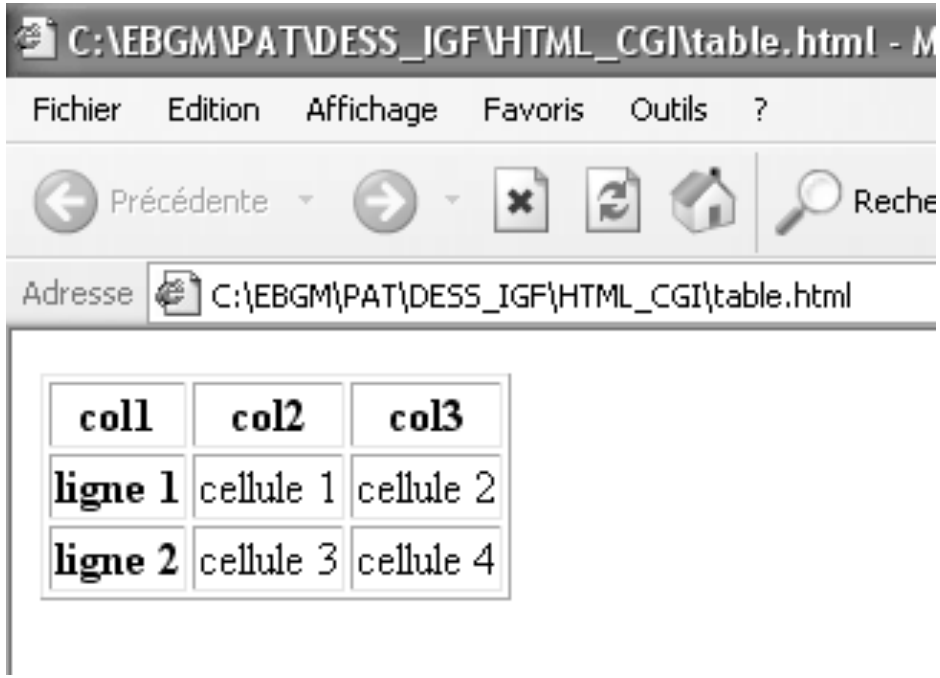
- Moyen élégant de présenter des données
- balise : <table>…</table> définit les limites du tableau
  - peut contenir les attributs :
    - border : largeur en pixel du cadre autour du tableau
    - width : largeur en pixel du tableau
    - align : pour placer le tableau (center, left, right)
    - etc…
- balise <tr> : définit une nouvelle ligne du tableau
- balise <th> : définit les en-têtes du tableau
- balise <td> : définit les données du tableau

Exemple de Tableau

code

```
<table border=1>
  <tr>
    <th>col1
    <th>col2
    <th>col3
  <tr>
    <th>ligne 1
    <td>cellule 1
    <td>cellule 2
  <tr>
    <th>ligne 2
    <td>cellule 3
    <td>cellule 4
</table>
```

rendu



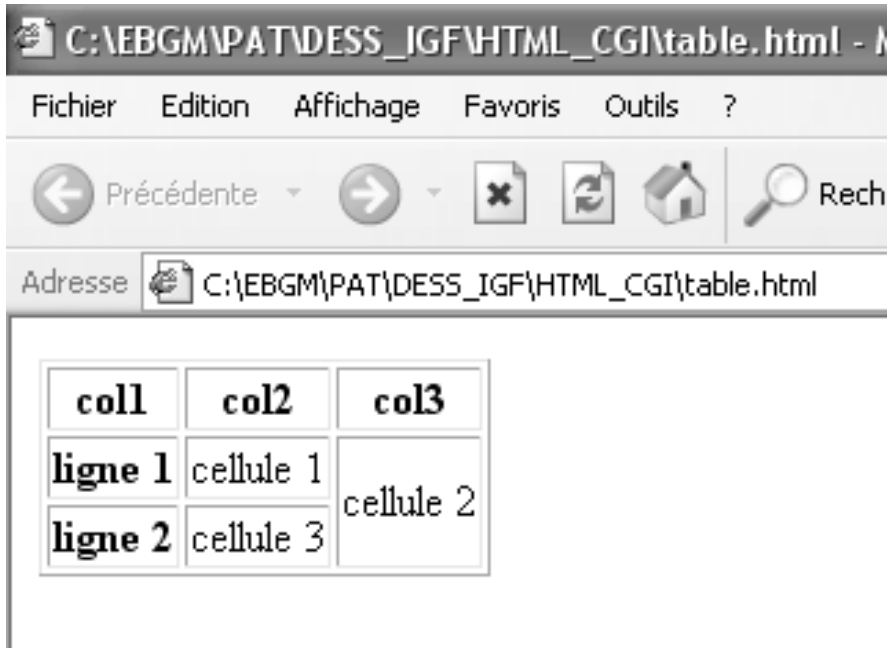
Fusion verticale de cellule (attribut rowspan)

code

```
<table border=1>
  <tr>
    <th>col1
    <th>col2
    <th>col3
  <tr>
    <th>ligne 1
    <td>cellule 1
    <td rowspan=2>cellule 2
  <tr>
    <th>ligne 2
    <td>cellule 3

</table>
```

rendu



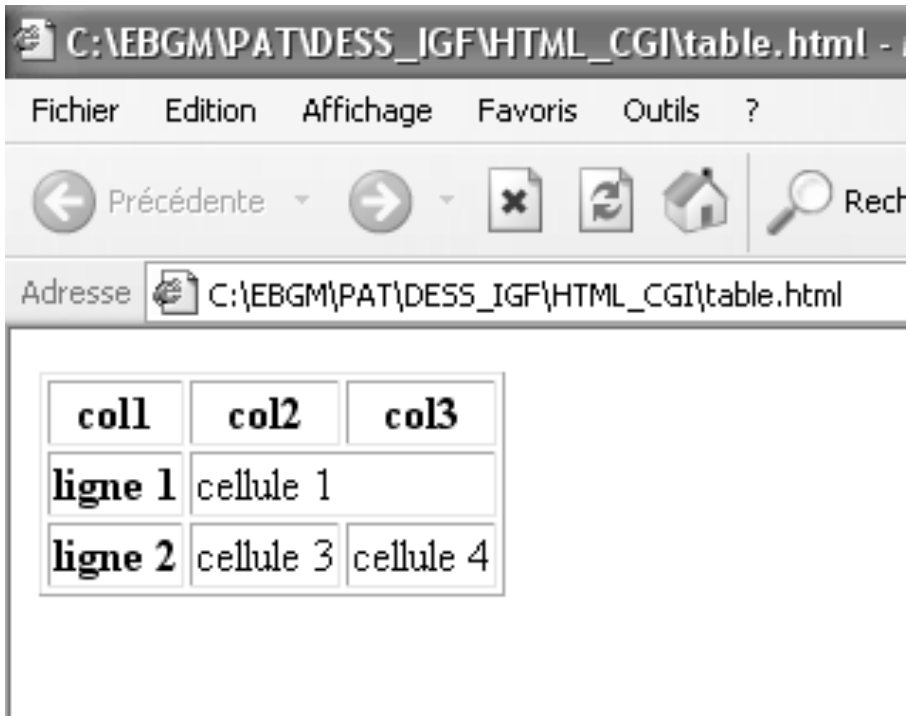
Fusion horizontale de cellule (attribut colspan)

code

```
<table border=1>
  <tr>
    <th>col1
    <th>col2
    <th>col3
  <tr>
    <th>ligne 1
    <td colspan=2>cellule 1

  <tr>
    <th>ligne 2
    <td>cellule 3
    <td>cellule 4
</table>
```

rendu



Remarques sur les tableaux

- On peut mettre autre chose que du texte dans les cellules d'un tableau :
  - image
  - liens
  - etc…
- les balises <tr>, <th> et <td> admettent l'attribut **align** (center, left, right)
- Outil sur le web pour construire des tableaux automatiquement :

http://www.bagism.coml/tablemaker

Outils HTML avancés

1. les tableaux
2. **les cadres**
3. les feuilles de style en cascade
4. les formulaires

Les Cadres

- Division de la page HTML en plusieurs zones
- la balise <frameset>…</frameset> : permet de diviser la page en différentes zones (cadres)
  - attributs de la balise <frameset> :
    - rows="20%,80%" : définit en 2 zones horizontales de 20 % et 80 % de l'espace disponible dans le navigateur
    - cols="20%,80%" : même chose mais pour 2 zones verticales
- la balise <frame> : définit vers quelle page HTML chacune des zones pointe
  - attributs de la balise <frame> :
    - name : définit le nom de la zone correspondante
    - scrolling=yes (ou no) : définit si on a des barres de défilement
    - src="fichier.html" : nom du fichier HTML qui sera intégré dans la zone

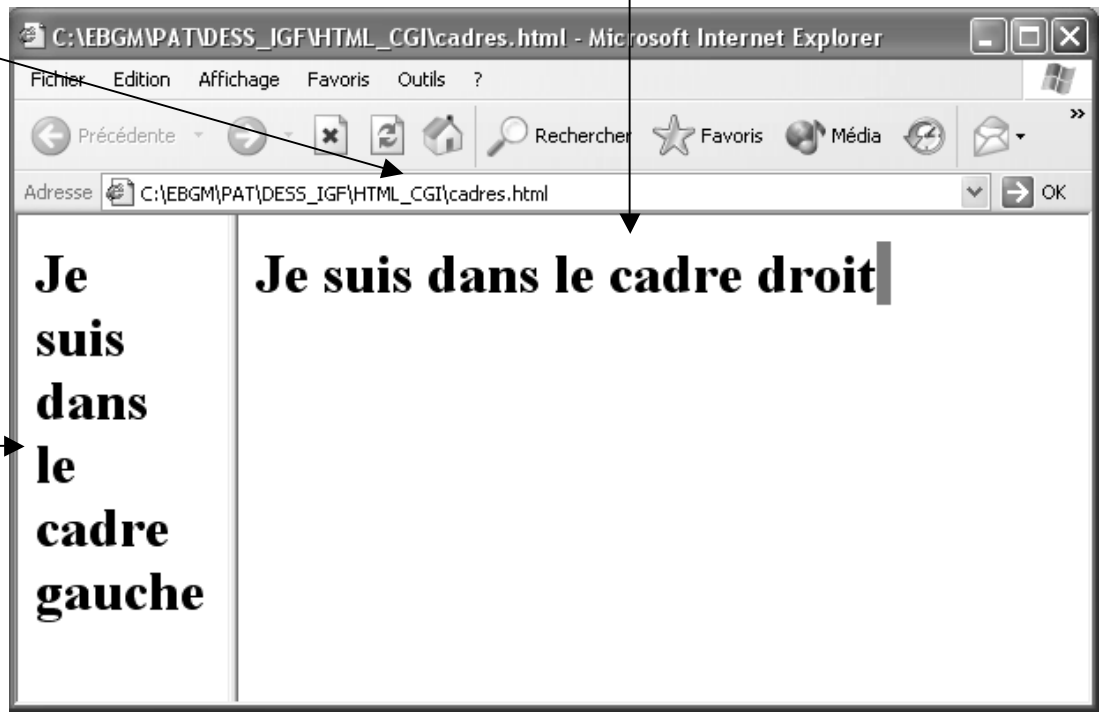
ATTENTION : Ne pas mettre de balise <body>…</body> dans une page définissant un cadre !

Exemple de cadre

```
<html>
<frameset cols="20%,80%">
<frame name="cadregauche"
src="menu.html">
<frame name="cadredroit"
src="welcome.html">
</frameset>
</html>
```

```
<html>
<body>
<h1>Je suis dans le cadre
droit</h1>
</body>
</html>
```

```
<html>
<body>
<h1>Je suis dans le cadre
gauche</h1>
</body>
</html>
```



Cadre : lien actif dans une autre zone

- utilisation de l'attribut **target** dans la balise <a>
- Exemple :

  …

  ➔le lien apparaitra dans la zone préalablement définie comme "autre_zone"

Cadre : lien actif dans une autre zone (exemple)

```
<html>
<frameset cols="20%,80%">
<frame name="cadregauche"
src="menu.html">
<frame name="cadredroit"
src="welcome.html">
</frameset>
</html>
```

```
<html>
<body>
<h1>Je suis dans le cadre
droit</h1>
</body>
</html>
```

```
<html>
<body>
<h1><a
href="http://www.python.org"
target="cadredroit">python</
a></h1>
</body>
</html>
```

Outils HTML avancés

1. les tableaux
2. les cadres
3. **les feuilles de style en cascade**
4. les formulaires

Les feuilles de style : CSS (I)

Imposer un style pour un type de balise :

```
body {
  color:#ffffff;
  font-size: small;
  background-color:#000000;
}

img {
  border:0;
}
```

Créer sa propre classe :

```
.maClasse {
  width:80%;
  color:#ffffff;
}
```

```
<html>
<head>

<title> accueil </title>

<link rel="stylesheet"
href="./css/dflt.css" type="text/css"
title="galadalfos" />

</head>

<body>

<p class="maClasse">
[…]

</body>
</html>
```

http://www.w3schools.com/css/

Les feuilles de style : CSS (II)

Définir une classe pour un type de balise :

```
img .maClassImg{
  padding:1em;
}
```

Définir une zone :

```
#albumPhoto {
  background-color:#000000;
}

#albumPhoto .maPhoto{
  border: #cccccc 1px solid;
}
```

```
<img class="maClassImg" […] >

<p id="albumPhoto">
  <img class="maPhoto">
```

```
<span class="maClass">Toto</span>
```
pas de saut de ligne

```
<div id="albumPhoto"> </div>
```
saut de ligne

Outils HTML avancés

1. les tableaux
2. les cadres
3. les feuilles de style en cascade
4. **les formulaires**

Les Formulaires

- **Définition** : feuille HTML permettant d'envoyer des informations au serveur
  -> en général ces informations sont traitées sur le serveur par un programme CGI
- **la balise <form>…</form>** :
  - encadre le formulaire
  - champ "method" : indique le moyen de passer les informations au serveur
    - GET : passage des données par l'URL
    - POST : passage des données par l'entrée standard
  - champ "action" : pointe sur le script CGI du serveur

Les Formulaires (2)

- Exemple de balise form :

```
<form method="GET" action="http://camus.ebgm.jussieu.fr/cgi-bin/monscript.cgi">
...
...
</form>
```

contenu du formulaire

- ce formulaire pointe sur le script "monscript.cgi" localisé sur la machine camus.ebgm.jussieu.fr
- utilisation de la méthode GET

Contenu d'un formulaire

- saisie de texte :

```
<input type="text" size=20 maxlength=30 name="...">
```

- cases à cocher :

```
<input type="checkbox" name="...">
```

- boutons radio :

```
<input type="radio" name="..." value="a">
<input type="radio" name="..." value="b">
```

- champ de saisie :

```
<textarea name="..." rows=4 cols=20>
</textarea>
```

➔ le champ "name" sera reconnu en tant que tel dans le script cgi

Contenu d'un formulaire (2)

- liste déroulante :

```
<select name="...">
<option value="a">option a
<option value="b">option b
<option value="c">option c
</select>
```

nom de la liste envoyé au script CGI

texte visible dans le navigateur

valeur de l'objet associée à la liste (envoyée au script CGI)

- bouton de soumission :

```
<input type="submit" name="..." value="...">
```

- bouton de reset :

```
<input type="reset" name="..." value="...">
```

texte visible dans le bouton

PLAN

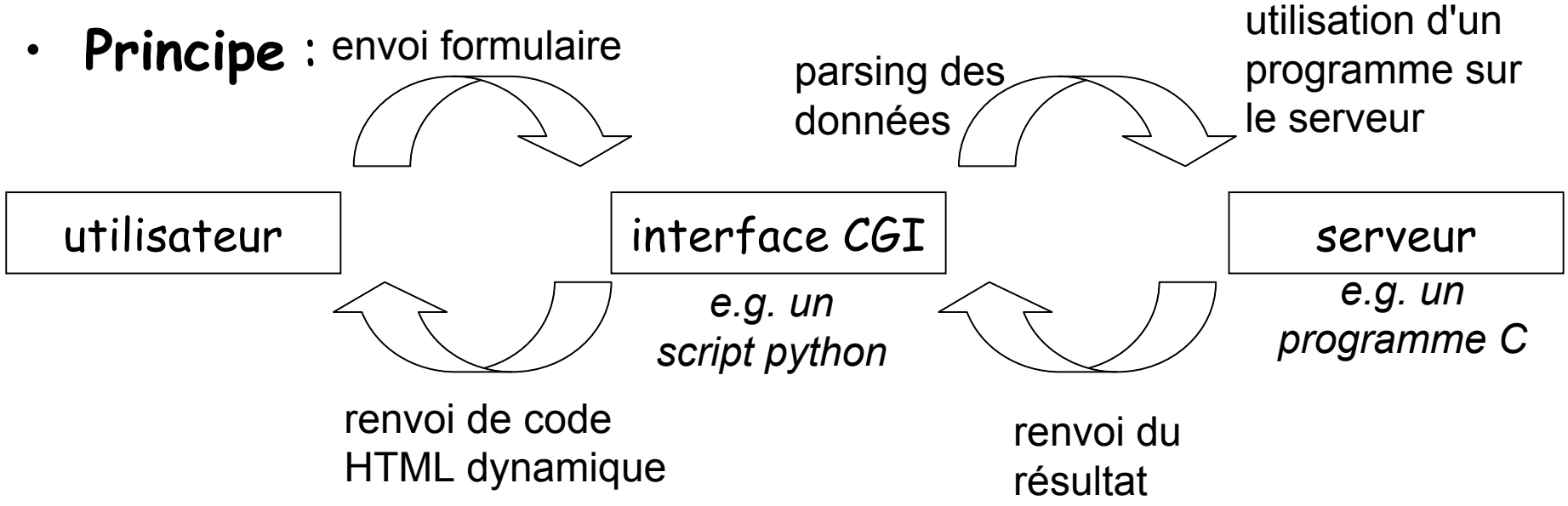
1. Généralités
2. Les Bases d'HTML
3. Outils HTML avancés
4. HTML dynamique / Interface CGI

L'interface CGI (Common Gateway Interface)

- **Définition** : programme d'interface entre l'utilisateur et le serveur

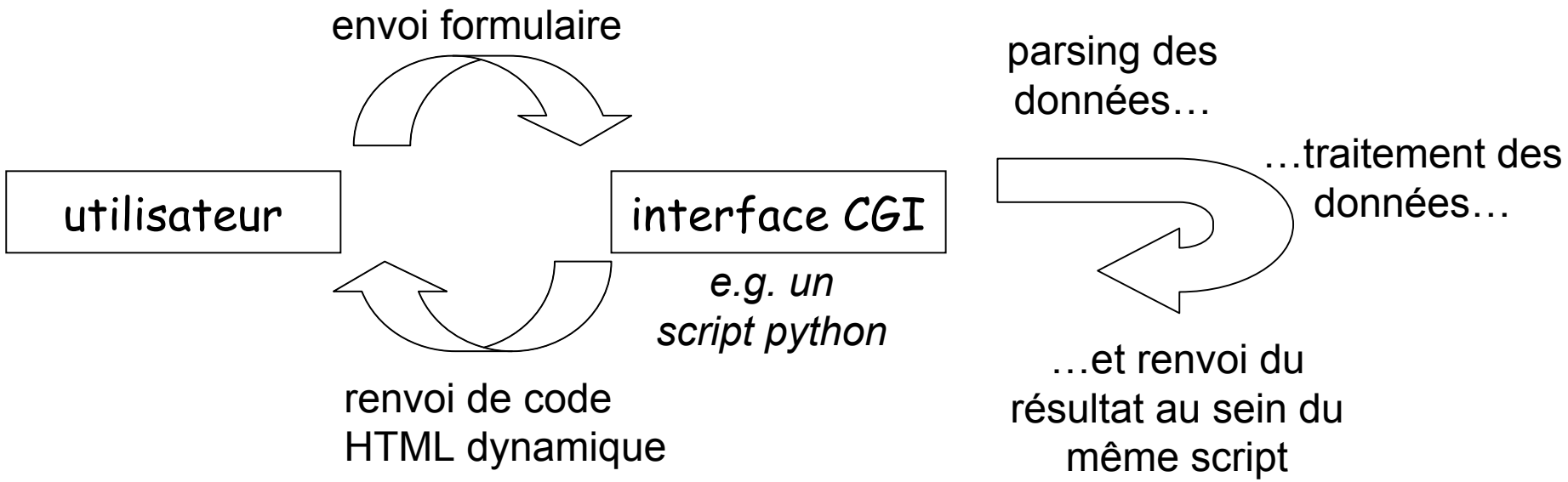
  -> permet de faire tourner un programme sur le serveur et générer une page HTML dynamiquement

- **Principe** :



L'interface CGI (2)

- **Éventuellement script autonome** :



Généralités sur les interfaces CGI

- méthodes GET / POST :
  - GET : passe par l'URL (limitée en taille, récupérable par la variable d'environnement $QUERY_STRING)
  - POST : passe par l'entrée standard (méthode plus sûre, pas de limitation de taille, permet de passer des fichiers)
- Langages des interfaces CGI :
  - shells : sh, csh, bash… (peu flexible)
  - compilés : C, C++ (rapide, lourd à mettre en œuvre)
  - interprétés : python, perl, PHP (moins rapides, flexibles, nombreuses bibliothèques)

Codage des entrées de l'utilisateur

- Exemple de flux envoyé à un script CGI :

```
sequence=bonjour+toto&format=FORMAT1&submit=print+all+the+fields+%21
```

- Codage utilisé :
  - chaque champ est séparé par le signe '&'
  - chaque nom de champ ainsi que son contenu séparés par '='
  - espaces remplacés par '+'
  - caractères spéciaux remplacés par '%' suivi d'un numéro hexadécimal
- Nécessité de décoder lorsqu'on utilise un shell
- Il existe des bibliothèques pour décoder automatiquement (e.g. cgi en python)

Sortie d'un programme CGI

Le navigateur doit recevoir 2 éléments :

- une en-tête contenant le type de donnée à afficher :

```
Content-Type : text/html¶
¶
```

-> le type de données doit être un type MIME valide (Multipurpose Internet Mail Extension)

-> le retour chariot est très important !

- l'information à afficher en HTML

Le module cgi en Python

- La fonction FieldStorage :

  -> renvoie un dictionnaire dont les clefs représentent chacun des éléments du formulaire

code Python

dictionnaire contenant les éléments du formulaire

```
form = cgi.FieldStorage()
sequence = form["zonetexte"].value
format = form["liste"].value
submit = form["submit"].value
```

code HTML

```
<textarea NAME="zonetexte" ROWS=5
COLS=50></textarea>

<select name="liste">
<option value="1">Option1
<option value="2">Option2
<option value="3">Option3
</select>

<p><input type="submit"
name="submit" value="Soumettre">
```

La fonction popen du module os

- La fonction popen permet d'effectuer un pipe sur une commande passée au shell

-> évite la création puis la destruction d'un fichier

Par exemple :

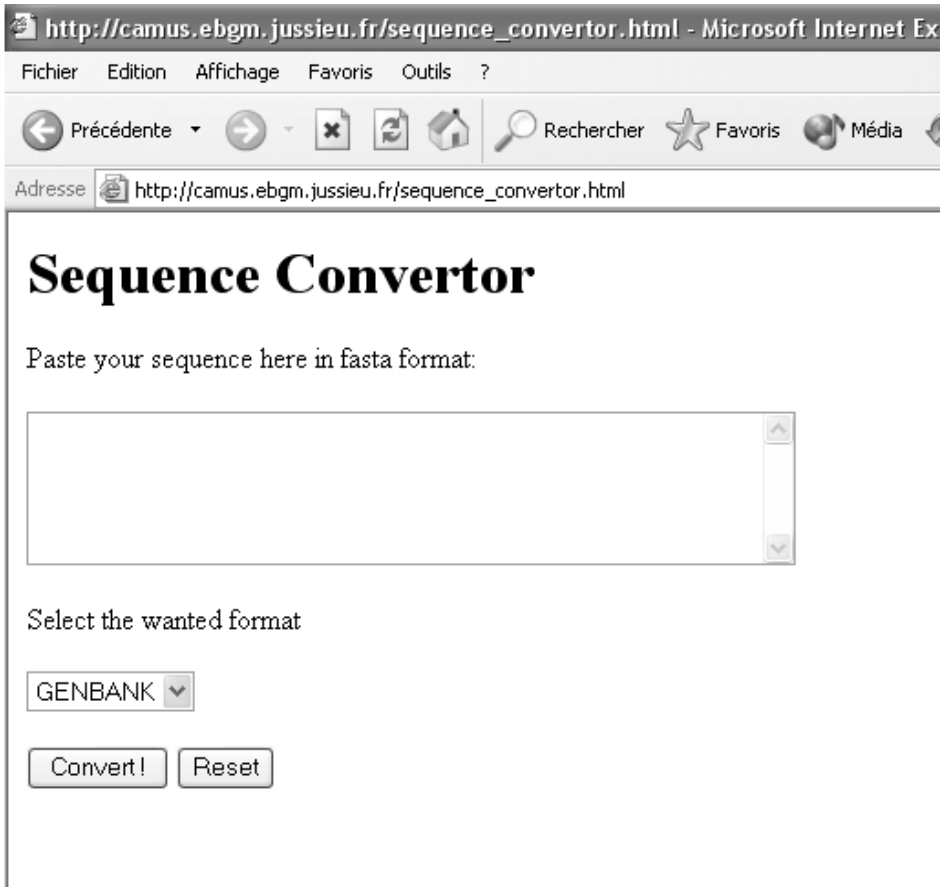
```
output = os.popen("ls -l","r").readlines()
```

au lieu de :

```
os.system("ls -l >& /tmp/tmpfile")
output = open("/tmp/tmpfile","r").readlines()
os.system("rm -f /tmp/tmpfile")
```

Exemple d'application CGI

-> convertisseur de séquence (fasta -> EMBL, GENBANK, GCG…) en utilisant le programme readseq



rq : service déjà existant à pasteur, infobiogen…

Considérations techniques

Les interfaces CGI sont gérées par le programme APACHE sur la machine 134.157.50.202

- les scripts CGI doivent être :
  - enregistrés dans le répertoire : `~etudiant/cgi-bin`
  - exécutables
- les pages webs doivent être
  - enregistrer dans le répertoire : `~etudiant/htdocs`
  - elles seront visibles à l'URL : `http://134.157.50.202`
- pour le debugging, on peut consulter le fichier log :

  `~etudiant/httpd/logs/error`

A vous !