Recherche d'Information Multimédia

Le World Wide Web : HTML et CGI

Philippe Mulhem - Georges Quénot

Plan

- **Introduction : Terminologie - Le World Wide Web,**
- **Le protocole HTTP,**
- **Les URLs,**
- **Le langage HTML.**

- **La "Common Gateway Interface" (CGI) et son emploi par HTML,**
- **La CGI coté serveur en C.**

1. Introduction

- **Terminologie :**
  - **Hypertexte : un texte que l'on peut parcourir autrement que séquentiellement par l'utilisation de liens,**
  - **Hypermédia = hypertexte + multimédia : on n'a plus uniquement du texte, mais des données images, vidéo, son…,**
  - **HTML : HyperText Markup Language (documentation : `http://www.w3.org/TR/html401/`),**
  - **HTTP : HyperText Transfert Protocol,**
  - **URL : Uniform Resource Locator,**
  - **Lien HTML : une source (visualisée par une ancre) + une cible (URL).**

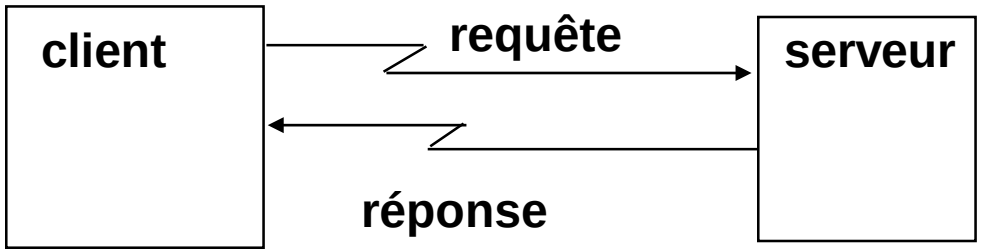
2. Le World Wide Web

- **WWW, W3, Web, etc.**
- **Système d'Information Hypermédia sur le Réseau Internet,**
- **Créé en 1989 par le CERN pour échanger des infos entre ses chercheurs,**
- **Basé sur un modèle client-serveur avec le protocole HTTP,**

- **Actuellement il est impossible de savoir le nombre de documents accessibles sur le Web (>> 10 milliards).**

2. Le World Wide Web

- **Basé sur le principe Client/Serveur**
- **le Serveur :**
  - **Répond aux requêtes du client : transfert de fichiers HTML, mais aussi exécution de programmes et retour des résultats,**
- **Le Client :**
  - **Gère l'interface avec les utilisateurs : acquisition de ce que veut l'utilisateur et restitution de ce qui est envoyé par un serveur,**
  - **Exemples : Firefox, Internet Explorer, Safari, Chrome ...**



3. Le protocole HTTP

- **Protocole de niveau application qui utilise TCP/IP,**
  - **Le client envoie des requêtes du style GET ou POST suivi d'un URL,**
  - **Le serveur renvoie au client des documents qui répondent à la norme MIME (Multipurpose Internet Mail Extensions),**
  - **Le serveur écoute sur un port TCP (80 en général, un serveur est habituellement un démon httpd),**
  - **Le client ouvre une connexion TCP sur ce même port,**
  - **Le client émet sa requête (1 ligne terminée par CRLF),**
  - **Le serveur renvoie le document demandé et coupe la connexion.**

4. Les URL

- **Permettent de décrire de manière unique des noeuds du Web :**
  méthode://machine[:port]/[répertoire/]fichier[#ancre|?listeparam]
- **Méthode : http, file, ftp, telnet, news, wais, …,**
- **Machine : un nom de machine possédant un serveur,**
- **Répertoire : peut être vide,**
- **Fichier : le nom du fichier sur le serveur,**
- **#ancre : ancre dans un fichier (c-à-d une position fixée),**
- **Listeparam : utilisation avec CGI pour des exécutables (GET).**
- **Dans le cas de pages HTML :**
  - **http://ufrima.imag.fr/PLACARD/RIM/index.html**
- **Dans le cas d'un exécutable (GET) avec les paramètres NOM et PRENOM :**
  - **http://hoff.imag.fr/RIM/post2?NOM=Dupont&PRENOM=Jean**

5. Le langage HTML
Introduction

- **Ce langage peut être utilisé pour décrire**
  - **de la documentation,**
  - **des documents structurés simples,**
  - **des documents hypertextes/hypermédia,**
- **Actuellement il est très mal utilisé, car les clients sont trop souples (plus de 90% des pages ne resptectent pas la syntaxe définie),**
- **Doc en ligne sur HTML5:**
  - **`https://www.w3.org/TR/html5/`**

5. Le langage HTML
Généralités

- Un document HTML est un texte ASCII, et peut donc être créé par un éditeur de textes (nedit, …),
- Basé sur SGML : les éléments du document sont délimités par des marqueurs qui peuvent avoir des attributes associés,
- Décrit la structure logique des documents, pas leur représentation physique à l'écran,
- Pour un marqueur appelé `X`, habituellement on utilise : `"...<X> ... </X>..."` pour délimiter la partie logique qui correspond au marqueur `X`.
- Si un marqueur `X` a un attribut `A`, on écrit : `"... <X A=...> ... </X>..."`

5. Le langage HTML
Les entités

- **Permettent d'afficher les caractères propres à une langue,**
- **L'entité "`&eacute;`" représente un "é" ,**
- **L'entité "`&egrave;`" représente un "è" ,**
- **HTML définit des entités pour les caractères ISO 8859-1,**
- **Dans un document HTML "`&eacute;t&eacute;`" correspond à l'affichage de "été",**
- **Le "&" permet à l'analyseur de savoir qu'il rencontre une entité, et le ";" final détermine la fin de l'entité.**

5. Le langage HTML
Marqueurs Généraux

- **Propriétés d'un document complet :**
  - **<HTML> ..... </HTML> : doit délimiter un document,**
  - **<HEAD> ..... </HEAD> : décrit des infos sur le document,**
  - **<TITLE> ….. </TITLE> : titre d'un document (optionnel, généralement affiché séparément du document),**
  - **<BODY> ..... </BODY> le corps du document.**

- **Un document HTML a donc la forme :**

```
<HTML>
   <HEAD>
      <TITLE> Le titre </TITLE>
   </HEAD>
   <BODY>
    ...
   </BODY>
</HTML>
```

5. Le langage HTML
Marqueur d'entête

- **BASE : indique la base à partir de laquelle des URL relatives sont utilisées dans la page HTML,**
- **HREF : attribut qui indique la base,**

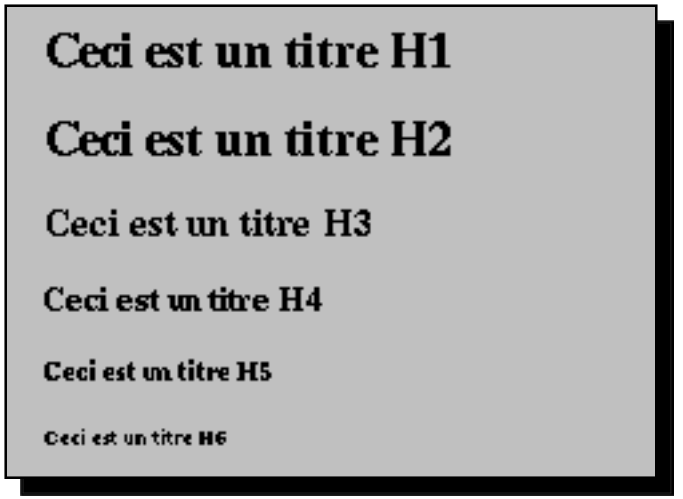
```
<HEAD>
   ...
   <BASE HREF="http://ufrima.imag.fr/PLACARD/">
</HEAD>
...
```

indique que toutes les URL relatives seront résolues en préfixant par la chaîne "http://ufrima.imag.fr/PLACARD/".

5. Le langage HTML
Marqueurs du corps

– **H*n* : Formatage de texte :**
  titre de profondeur *n*, *1 à 6*

```
<H1> Ceci est un titre H1</H1>
<H2> Ceci est un titre H2</H2>
...
```



– **P : Marque de paragraphe :**

```
<P> Ceci est un premier ... texte.</P>
<P> Ceci est un second ... lignes.</P>
```

Ceci est un premier paragraphe qui est une suite de mots qui peuvent bien sur tenir sur plusieurs lignes. Le problème est alors de fournir suffisamment de texte.

Ceci est un second paragraphe qui peut également être affiché sur plusieurs lignes.

5. Le langage HTML
Marqueurs du corps

- **Listes :**
  - **UL (liste non-numérotée) et OL (liste numérotée); les éléments de la liste sont marqués au début par LI (list item) :**

```
<UL>
   <LI> Ceci est le premier item
   <LI> Ceci est le second item
</UL>
```

On voit maintenant une liste non numérotée qui contient deux items :

- Ceci est le premier item
- Ceci est le second item

```
<OL>
   <LI> Ceci est le premier item
   <LI> ceci est le second item
</OL>
```

On trouve maintenant une liste numérotée qui contient elle aussi deux items :

1. Ceci est le premier item
2. Ceci est le second item

5. Le langage HTML
Les marqueurs du corps

- **Listes (suite)**
  - » **DL (liste de définitions)**
  - » **Un terme (DT) suivi de sa définition (DD)**

```
<DL>
  <DT> CLIPS
  <DD> Laboratoire CommunicationLangagi&eacute;re
et Interaction Personne Syst&ecirc;me
  <DT> IMAG
  <DD> Institut Informatique et Math&eacute;matiques
Appliqu&eacute;es de Grenoble.          </DL>
```

CLIPS

Laboratoire Communication Languagière et Interaction Personne-Système.

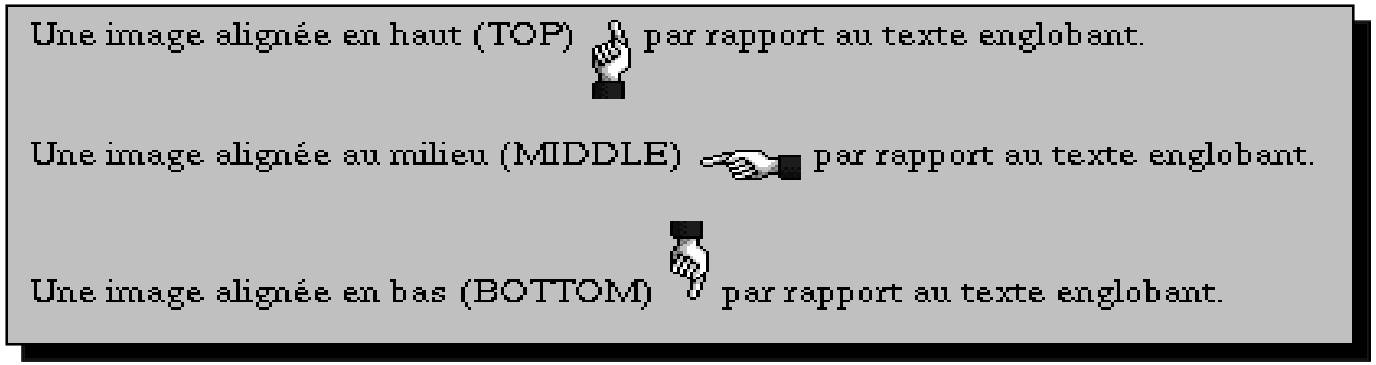
IMAG

Institut Informatique et Mathématiques Appliquées de Grenoble.

5. Le langage HTML
Les marqueurs du corps

- **Inclusion d'images**
  - » **IMG : pour des icônes ou des petites images. Attributs :**
    - **SRC : l'URL (le nom) de l'image (.gif ou .jpg) obligatoire,**
    - **ALIGN : (TOP | MIDDLE | BOTTOM) spécifie l'alignement avec le contexte de l'image,**
    - **ALT : chaîne de caractère si le programme qui lie le HTML ne peut pas afficher l'image,**
    - **WIDTH, HEIGHT : hauteur et largeur a l'affichage.**

```
Une image ... (TOP) <IMG SRC="http://hoff.imag.fr/
RIM/EXEMPLES/u_hand.gif" ALIGN=TOP> ...
```



5. Le langage HTML
Les marqueurs du corps

- **Marqueurs de présentation**
  - **Passage à la ligne :
**
  - **Ligne de séparation horizntale : <HR>**

```
Un saut <BR> de ligne.
Une ligne <HR> horizontale de s&eacute;paration.
```

5. Le langage HTML
Les marqueurs du corps

- **Marqueurs de présentation**
  - **Les tableaux : <TABLE> ... </TABLE>**
    - » **Attribut important : BORDER**
    - » **Sous-marqueurs :**
      - **Titre : <CAPTION> ... </CAPTION>**
      - **Lignes : <TR> ... </TR>**
      - **Cellules titres : <TH> ... </TH>**
      - **Cellules : <TD> ... </TD>**


```
<TABLE BORDER=1>
    <CAPTION> Table 1 </CAPTION>
    <TR>
      <TH>colonne 1</TH>
      <TH>colonne 2</TH>
    </TR>
    <TR>
      <TD>12</TD>
      <TD>14</TD>
    </TR>
</TABLE>
```

5. Le langage HTML
Les marqueurs du corps

- **Définition de liens :**

```
L'<A HREF="http://www.imag.fr/index.html">IMAG</A>
est une f&eacute;d&eacute;ration de laboratoires.
```

- **L'ancre peut être une image :**

```
L'<A HREF="http://www.imag.fr/index.html">
<IMG SRC="http://www.imag.fr/icone.gif"></A> est
une f&eacute;d&eacute;ration de laboratoires.
```

5. Le langage HTML
Les marqueurs du corps

– **Ancre interne pour des liens dans une même page :**

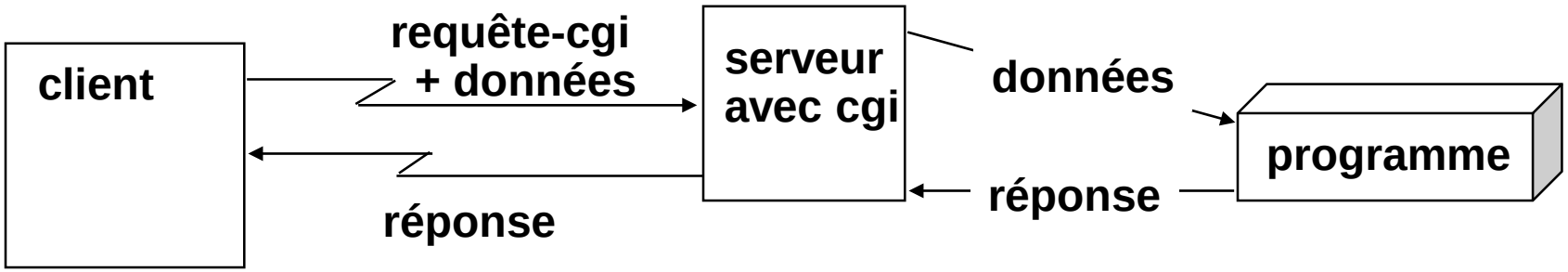
```
La <A HREF="#strato">stratosph&egrave;re</A> se
situe entre 12Km et 50Km d'altitude.

...

Le terme <A NAME="strato"> stratosph&egrave;re
vient du grec ...
```

6. La CGI
Common Gateway Interface

- **Permet d'indiquer que le serveur doit exécuter un programme (exécutable, shell script, perl) et renvoyer le résultat plutôt que de renvoyer le document cible.**



6. La CGI

- **Elle est habituellement utilisée lors de l'appel à des URL qui sont de la forme :**

  `"http://machine/cgi-bin/exécutable"`

- **On se sert habituellement de cette interface par l'utilisation de formulaires HTML avec marqueur FORM pour envoyer les données au programme,**

- **La réponse est (en général) une page HTML.**

6. La CGI
Appel dans un corps HTML

- **FORMULAIRE :**
  - **Marqueur <FORM> ... </FORM> :**
    - » **Attribut ACTION : indique quel exécutable appeler (une URL qui habituellement contient "/cgi-bin/"avec les paramètres du formulaire remplis (sous forme NOM=VALEUR&...),**
    - » **Attribut METHOD : la manière de véhiculer les paramètres vers le serveur (POST ou GET). On va dans la suite s'intéresser au POST,**
    - » **Entre <FORM> et </FORM>, description du contenu du formulaire.**

6. La CGI
Appel dans un corps HTML

- **FORMULAIRE :**
  - **Exemple :**

```
<FORM ACTION="http://hoff.imag.fr/cgi-
 bin/post2" METHOD=POST>
...
</FORM>
```

6. La CGI
Appel dans un corps HTML
Corps d'un formulaire

- **Le marqueur <INPUT> ... </INPUT> :**
  - **Attributs :**
    - **TYPE : RADIO, CHECKBOX, SUBMIT, RESET, TEXT, HIDDEN, IMAGE, …,**
    - **NAME : le nom du composant,**
    - **VALUE : valeur du champs (dépend du type),**
    - **MAXLENGTH, SIZE, CHECKED : attributs facultatifs dont l'utilisation dépend du type d'INPUT.**

6. La CGI
Appel dans un corps HTML
Corps d'un formulaire

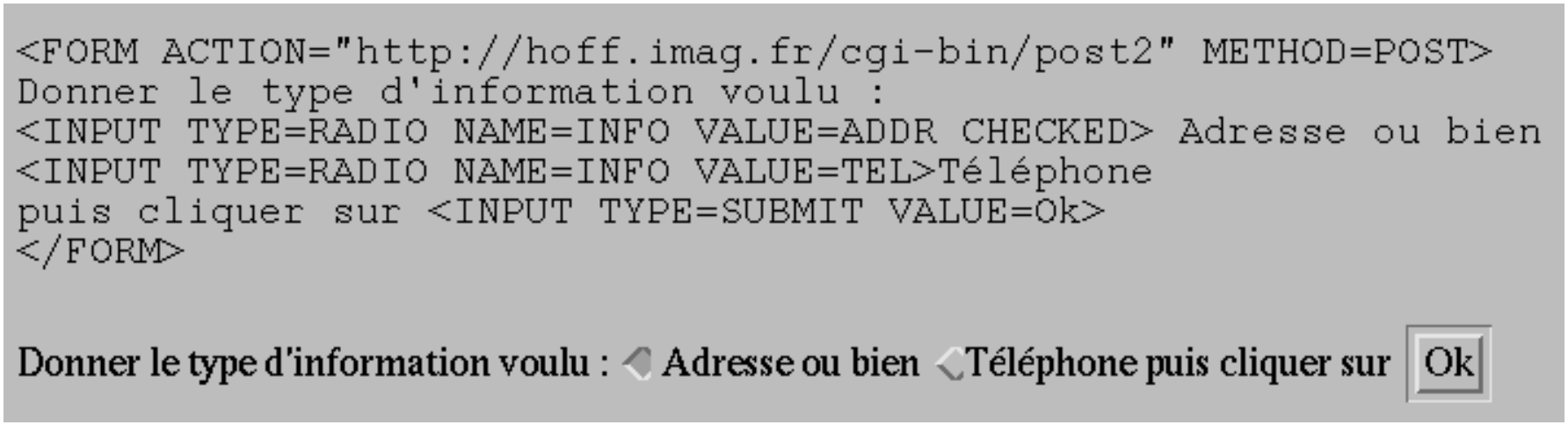
- **Un INPUT de type RADIO**
  - **Permet de définir un choix parmi plusieurs,**
  - **Utilisation conjointe obligatoire avec un input SUBMIT.**
  - **Utilisation de RESET facultative.**

```
Donner le type d'information voulu :
<INPUT TYPE=RADIO NAME=INFO VALUE=ADDR> Adresse,
ou bien <INPUT TYPE=RADIO NAME=INFO VALUE=TEL>
T&eacute;l&eacute;phone, puis cliquer sur
<INPUT TYPE=SUBMIT VALUE=Ok>
```

6. La CGI
Appel dans un corps HTML
Corps d'un formulaire

- **Un INPUT de type RADIO**
  - **Exemple:**

```
<FORM ACTION="http://hoff.imag.fr/cgi-bin/post2" METHOD=POST>
Donner le type d'information voulu :
<INPUT TYPE=RADIO NAME=INFO VALUE=ADDR CHECKED> Adresse ou bien
<INPUT TYPE=RADIO NAME=INFO VALUE=TEL>Téléphone
puis cliquer sur <INPUT TYPE=SUBMIT VALUE=Ok>
</FORM>
```



6. La CGI
Appel dans un corps HTML
Corps d'un formulaire

- **Un INPUT de type CHECKBOX :**
  - **Permet de définir plus de 1 choix parmi plusieurs,**
  - **Utilisation conjointe obligatoire avec un input SUBMIT,**
  - **RESET facultatif.**

```
Donner le type d'information voulu :
<INPUT TYPE=CHECKBOX NAME=INFO VALUE=ADDR CHECKED>
Adresse, et/ou <INPUT TYPE=CHECKBOX NAME=INFO
VALUE=TEL> T&eacute;l&eacute;phone, puis cliquuer sur
<INPUT TYPE=SUBMIT VALUE=Ok> (ou <INPUT TYPE=RESET
VALUE=Raz> pour &eacute;tat initial).
```

6. La CGI
Appel dans un corps HTML
Corps d'un formulaire

- **Un INPUT de type CHECKBOX**
  - **Exemple :**

```
<FORM ACTION="http://hoff.imag.fr/cgi-bin/post2" METHOD=POST>
Donner le type d'information voulu :
<INPUT TYPE=CHECKBOX NAME=INFO VALUE=ADDR CHECKED> Adresse ou bien
<INPUT TYPE=CHECKBOX NAME=INFO VALUE=TEL>Téléphone
puis cliquer sur <INPUT TYPE=SUBMIT VALUE=Ok> (ou bien sur
<INPUT TYPE=RESET VALUR=Raz> pour état initial)
</FORM>
```

Donner le type d'information voulu : ☐ Adresse ou bien ☐ Téléphone puis cliquer sur [Ok] (ou bien sur [Raz] pour état initial)

6. La CGI
Appel dans un corps HTML
Corps d'un formulaire

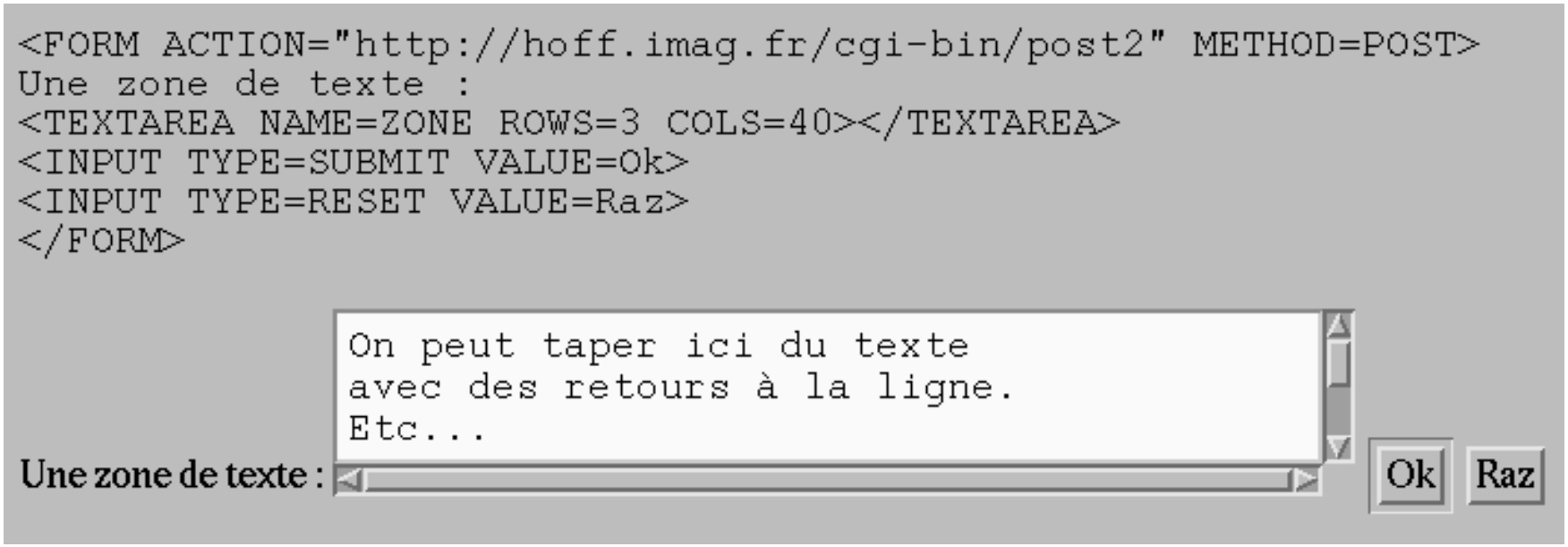
- **Un INPUT de type TEXT :**
  - **Permet de taper une chaîne de caractères.**

```
Donner le nom de la personne recherch&eacute;e :
<INPUT TYPE=TEXT NAME=NOM MAXLENGTH=30 SIZE=20>
puis cliquer sur <INPUT TYPE=SUBMIT VALUE=Ok>.
```

6. La CGI
Appel dans un corps HTML
Corps d'un formulaire

- **Un INPUT de type TEXTAREA :**
  - **Permet de taper un texte avec retours à la ligne.**

```
<FORM ACTION="http://hoff.imag.fr/cgi-bin/post2" METHOD=POST>
Une zone de texte :
<TEXTAREA NAME=ZONE ROWS=3 COLS=40></TEXTAREA>
<INPUT TYPE=SUBMIT VALUE=Ok>
<INPUT TYPE=RESET VALUE=Raz>
</FORM>
```



6. La CGI
Appel dans un corps HTML
Corps d'un formulaire

- **Le marqueur <SELECT> ... </SELECT>**
  - **Permet de choisir parmi des listes énumérées, sans redéclarer à chaque fois un marqueur INPUT,**
  - **Utiliser conjointement avec un INPUT SUBMIT,**
  - **Attributs :**
    - **NAME : le nom du sélecteur,**
    - **SIZE : le nombre d'items visibles à la fois (facultatif),**
    - **MULTIPLE : s'il est indiqué (sans valeur) on a un choix multiple possible, sinon un choix unique.**

6. La CGI
Appel dans un corps HTML
Corps d'un formulaire

- **Le sous-marqueur <OPTION> ... </OPTION> :**
  - **Délimite les choix d'un select,**
  - **Attributs :**
    - » **VALUE : la valeur retournée en cas de choix,**
    - » **SELECTED : le (ou les) choix validés initialement.**

6. La CGI
Appel dans un corps HTML
Corps d'un formulaire

- **Un SELECT unique :**

```
Faire un choix :
<SELECT NAME="NB CHOISI">
  <OPTION>Un
  <OPTION VALUE="DEUX Trois" SELECTED>Deux et trois
  <OPTION>Quatre
</SELECT>
puis cliquer sur <INPUT TYPE=SUBMIT VALUE=Ok>.
```

6. La CGI
Appel dans un corps HTML
Corps d'un formulaire

- **Un exemple de SELECT unique :**

```
<FORM ACTION="http://hoff.imag.fr/cgi-bin/post2" METHOD=POST>
Faire un choix :
<SELECT NAME="NB CHOISI" SIZE=2>
<OPTION>Un
<OPTION VALUE="DEUX Trois">Deux et Trois
<OPTION>Quatre
</SELECT>
puis cliquer sur <INPUT TYPE=SUBMIT VALUE=Ok>
</FORM>
```

6. La CGI
Appel dans un corps HTML
Corps d'un formulaire

- **Un SELECT multiple**

```
Faire un choix :
<SELECT NAME="NB CHOISI" MULTIPLE>
  <OPTION>Un
  <OPTION VALUE="DEUX Trois" SELECTED>Deux et trois
  <OPTION>Quatre
<SELECT>
puis cliquer sur <INPUT TYPE=SUBMIT VALUE=Ok>.
```

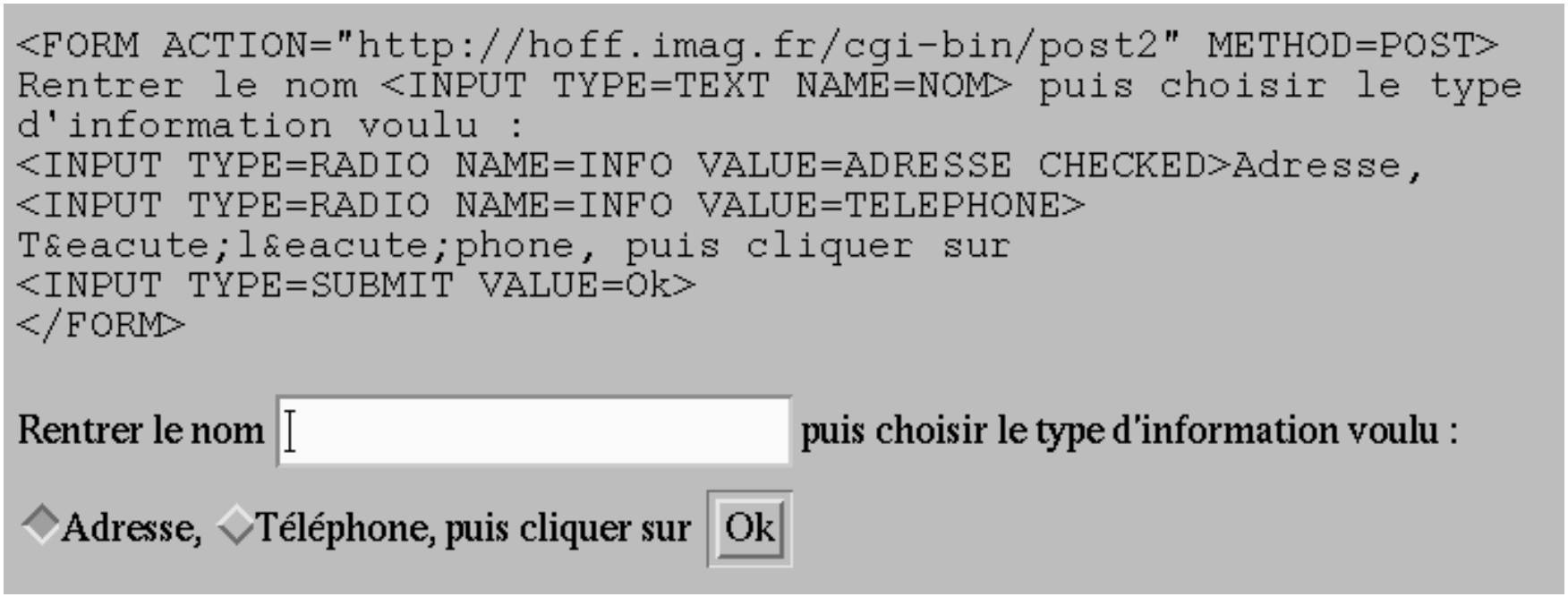
6. La CGI
Appel dans un corps HTML
Corps d'un formulaire

- **Il existe d'autres marqueurs :**
  - **Le type HIDDEN d'INPUT (en fait des input non affichés, mais passés à l'exécutable),**
  - **Le type PASSWORD d'INPUT (pour rentrer des chaînes de caractères avec des * en écho),**
  - **Le type IMAGE d'INPUT (pour pouvoir cliquer sur des images, les coordonnées x et y relatives au coin haut-gauche de l'images sont envoyées au serveur).**

6. La CGI
Appel dans un corps HTML

- **Un formulaire qui a un champ "chaîne de caractères à remplir et 2 boutons radio"** :

```
<FORM ACTION="http://hoff.imag.fr/cgi-bin/post2" METHOD=POST>
Rentrer le nom <INPUT TYPE=TEXT NAME=NOM> puis choisir le type
d'information voulu :
<INPUT TYPE=RADIO NAME=INFO VALUE=ADRESSE CHECKED>Adresse,
<INPUT TYPE=RADIO NAME=INFO VALUE=TELEPHONE>
T&eacute;l&eacute;phone, puis cliquer sur
<INPUT TYPE=SUBMIT VALUE=Ok>
</FORM>
```



6. La CGI
Coté serveur en C

- **Le programme exécuté fait 3 choses dans le cas d'un appel par un POST :**
  - **Décode les données envoyées sur son entrée standard,**
  - **Réalise ses traitements propres (par exemple calcule la valeur de correspondance entre une requête et des documents),**
  - **Génère en sortie standard un fichier HTML et des informations additionnelles.**

6. La CGI - Coté serveur en C

- **L'exécutable récupère en entrée standard (pour un POST) les valeurs des éléments du formulaire: <nom_de_champ>=<valeur>&...**
- **Exemple :**
  - **Un formulaire avec un champ INPUT de type TEXT de nom "NOM", et un champs radio de nom "INFO" et qui a pour valeur "TELEPHONE",**
  - **On tape : "{coucou -> (titi)}",**
  - **Sur l'entrée standard du programme appelé on récupère la chaîne de caractères :**

"NOM=%7Bcoucou+-%3E+%28titi%29%7D&INFO=TELEPHONE"

6. La CGI - Coté serveur en C

"NOM=%7Bcoucou+%3E+%28titi%29%7D&INFO=TELEPHONE“
({coucou -> (titi)})

- Pourquoi ? Les données doivent passer sur tous les réseaux et sur tous les systèmes d’exploitation :
  - Il y a 2 champs NOM et INFO,
  - Chaque champ a une valeur qui est après le '=',
  - Le séparateur de champ est le '&',
  - Les blancs ' ' sont transformés en '+',
  - Les caractères speciaux sont transformés en un code sur 3 octets commençant par '%' ("%7D" pour '{', car 7D est le code ASCII de '{').

6. La CGI - Coté serveur en C

- Décodage des paramètres : tableau de couples (name:char*, value:char*).
- Utilitaire C fourni : `cgiu.c` qui définit 4 fonctions de base :
  - `getword()` qui sépare des chaînes "<nom>=<valeur>"
  - `plustospace()` qui transforme les '+' en ' '
  - `unescape_url()` qui transfome les "%.." en les caractères correspondants
  - `getword()` qui sépare "<nom>" et "<valeur>"

6. La CGI - Coté serveur en C

- **Boucle de décodage (POST):**

```
#include "cgiu.h"
int main(int argc, char *argv[]) {
  entry *entries;
  int l,cl,x,m;
  char *qs;
  cl = atoi(getenv("CONTENT_LENGTH"));
  qs = (char *) malloc(sizeof(char) * (cl+1));
  for (l = 0; (l < cl) && (!feof(stdin)); l++)
    qs[l] = (char) fgetc(stdin);
  qs[l] = '\0';
  m = countword(qs,'&');
  entries = (entry *) malloc(m*sizeof(entry));
  for(x = 0; x < m; x++) {
    entries[x].val = getworda(qs,'&');
    plustospace(entries[x].val);
    unescape_url(entries[x].val);
    entries[x].name = getworda(entries[x].val,'=');
  }
  ...
```

6. La CGI - Coté serveur en C

- **Il suffit de retrouver les valeurs des noms recherchés (les noms sont dans l'ordre de leur définition en HTML).**
- **Dans notre exemple:**

  `entries[0].name` **vaut** `"NOM"`

  `entries[0].val` **vaut** `"{toto -> (titi)}"`

  `entries[1].name` **vaut** `"INFO"`

  `entries[1].val` **vaut** `"TELEPHONE"`

- **Ces données sont utilisées dans la suite du programme (`m` vaut `2`).**

6. La CGI - Coté serveur en C

- **Le résultat fournit par le programme :**
- **Le programme C fournit en sortie standard un fichier correspondant à un format MIME :**

```
Content-Type: text/html<LF><LF>
```

- **Ensuite il fournit en sortie un fichier HTML "normal".**

6. La CGI - Coté serveur en C

- **Exemple de programme qui donne la liste des couples (nom,valeur) en sortie :**

```
...
int main(int argc, char *argv[]) {
  entry *entries;
  int l,cl,x,m;

  ... /* décodage et traitements des données */
  printf("Content-type: text/html%c%c",10,10);
  printf("<HTML><HEAD><TITLE>R&eacute;ponse</TITLE></HEAD>\n");
  printf("<BODY><H1>Les donn&eacute;es re&ccedil;ues<H1><UL>\n");
  for (x =0; x <=m; x++) {
    printf("<LI><CODE>%s = %s</CODE>\n",
    entries[x].name,entries[x].val);
  }
  printf("</UL></BODY></HTML>\n");
  ...
}
```

6. La CGI - Coté serveur en C

- **Marche à suivre pour créer ces programmes :**
  - **Ecrire le programme source dans un de vos répertoires,**
  - **Inclure cgiu.h,**
  - **Le compiler (en utilisant cgiu.c),**
  - **Copier l'exécutable dans le répertoire prévu pour les exécutables cgi,**
  - **Copier éventuellement les données dans le répertoire prévu pour les données cgi.**

7. Conclusion

- **HTML sert de base à la description logique de pages sur le WEB,**
- **HTML permet d'appeler des applications que l'on programme (CGI - Common Gateway Interface) par les formulaires,**
- **Le WEB est basé sur le protocole HTTP**