

World-Wide Web et les formulaires électroniques, les images réactives

François Dagorn
francois.dagorn@univ-rennes1.fr

octobre 1995

1 Introduction

Les clients WWW sont en général utilisés pour accéder à des documents diffusés par des serveurs d'informations. En sélectionnant une ancre dans un texte, l'utilisateur déclenche le transfert de données dans le sens serveur vers client ... Ils peuvent également servir à communiquer des informations dans l'autre direction et être utilisés pour transmettre des paramètres à un serveur HTTP. L'émission de formulaires électroniques ou la présentation d'images réactives utilisent ces propriétés.

2 Les protagonistes

Depuis l'émission d'un formulaire électronique (ou d'une interaction avec une image réactive), jusqu'à l'accusé de réception du traitement adéquat par un serveur HTTP, les protagonistes sont les suivants :

- HTML qui propose les éléments `<FORM >`, `` et `<ISINDEX >` respectivement pour le traitement des formulaires, des images réactives et la communication d'un mot-clef à un moteur de recherche ;
- HTTP (et plus particulièrement la version HTTP/1.0) qui procure les méthodes `POST` et `GET` pour transmettre des paramètres à un serveur WWW ;
- les serveurs HTTP qui permettent l'appel de programmes externes pour traiter les informations transmises par les clients. Une interface standardisée appelée CGI (*Common Gateway Interface*) précise les règles d'écriture et d'exécution des procédures (les *CGI scripts*). Les principaux serveurs HTTP respectent la convention CGI ;
- des outils qui facilitent l'écriture des *CGI scripts*. Les produits *cgi-lib*¹ et *CgiParse*² sont utilisés dans cette présentation.

1. Disponible sur ftp://ftp.urec.fr/pub/reseaux/services_infos/WWW/ncsa/httpd/Unix/ncsa_httpd/cgi/.

2. Consulter <http://www.w3.org/hypertext/WWW/Daemon/User/CGI/Overview.html>.

3 Le traitement des formulaires : un exemple

L'exemple ci-dessous suppose l'utilisation d'un serveur HTTP sur une plate-forme Unix.

3.1 Le code HTML

Le premier élément conduisant au traitement d'un formulaire est la présentation HTML de ce dernier (cf. figure 1).

```
<HTML>
<HEAD>
  <TITLE>Un formulaire exemple</TITLE>
</HEAD>
<BODY>
  <H1>Un formulaire exemple</H1>
  <FORM ACTION="http://www.univ-rennes1.fr/test-cgi-form.cgi"
  METHOD="POST">
    Un champ textuel simple : <INPUT NAME="simple" SIZE=30 > <P>
    Un champ textuel sur plusieurs lignes : <P>
    <TEXTAREA NAME="lignes"
      ROWS=10 COLS=40> Ce texte est modifiable.
    </TEXTAREA>
    <P>
    <STRONG>Des boutons type radio : </STRONG> <P>
    <INPUT TYPE="radio" NAME="radio" VALUE="FM" >Modulation de fréquence <BR>
    <INPUT TYPE="radio" NAME="radio" VALUE="GO" CHECKED> Grandes Ondes <BR>
    <INPUT TYPE="radio" NAME="radio" VALUE="PO"> Petites Ondes <P>
    <STRONG>Des choix multiples : </STRONG> <P>
    <INPUT TYPE="checkbox" NAME="qcm" VALUE=1> un <BR>
    <INPUT TYPE="checkbox" NAME="qcm" VALUE=2> deux <BR>
    <INPUT TYPE="checkbox" NAME="qcm" VALUE=3> trois <P>
    Un menu déroulant :
    <SELECT NAME="menu">
      <OPTION SELECTED>50F
      <OPTION>60F
      <OPTION>70F
      <OPTION>100F
    </SELECT><P>
    Un mot de passe : <INPUT TYPE="password" NAME="passwd" SIZE="40">
    <P>
    Un champ de contexte caché : <INPUT TYPE="hidden" NAME="cache" value="secret"> <BR>
    Le bouton d'émission : <INPUT TYPE="submit" VALUE="Envoyer">
    Le bouton d'annulation : <INPUT TYPE="reset" VALUE="Annuler">
  </FORM>
</BODY>
</HTML>
```

FIG. 1 – Le code HTML d'un formulaire

La description des formulaires fait partie de HTML niveau 2, elle est bornée par `FORM` et `/FORM`. L'attribut `ACTION` donne ici l'URL de la procédure qui va traiter les champs du formulaire (une fois saisis), par défaut c'est l'URL courant (celui qui a servi pour afficher le formulaire) qui est sollicité. On utilise couramment une seule procédure pour émettre un formulaire puis réceptionner les champs saisis (cf. section 3.4).

L'attribut `METHOD` fait référence à HTTP et indique quelle est la méthode qui sera utilisée pour transmettre les champs du formulaire. La méthode `POST` est conseillée, elle transmet les champs dans le corps d'un message MIME. La méthode `GET` transmet les paramètres dans l'URL, divers problèmes liés à la taille et au nombre des champs peuvent survenir.

La description des champs du formulaire est réalisée par des balises `INPUT`, `SELECT` et `TEXTAREA` comme dans les exemples suivants :

– `<INPUT NAME="simple" SIZE=30 >`

présente un champ textuel de 30 caractères ;

– `<TEXTAREA NAME="lignes" ROWS=10 COLS=40 > </TEXTAREA >`
présente une zone de saisie de texte de 10 lignes et 40 colonnes ;

– `<INPUT TYPE="password" NAME="passwd" SIZE="40" >`
permet de masquer les caractères qui seront saisis dans un champ textuel de 40 caractères ;

– `<INPUT TYPE="hidden" NAME="cache" value="secret" >`
est un champ qui n'est pas présenté à l'utilisateur mais dont le contenu sera transmis à la procédure de réception des paramètres. Cette technique permet de tenir compte de certains éléments de contexte lors du traitement des champs du formulaire ;

– `<INPUT TYPE="radio" NAME="radio" VALUE="GO" CHECKED >GO`
`<INPUT TYPE="radio" NAME="radio" VALUE="PO" > PO`
présente deux boutons type radio, l'attribut `CHECKED` précise une valeur implicite ;

– `<INPUT TYPE="checkbox" NAME="qcm" VALUE=1 > un`
`<INPUT TYPE="checkbox" NAME="qcm" VALUE=2 > deux`
présente des choix multiples sous la forme de boutons ;

– `<SELECT NAME="menu" > <OPTION SELECTED >50F <OPTION >60F </S ELEC T>`
propose un menu déroulant dont la valeur par défaut est 50F .

L'attribut `NAME` est le nom du champ, il est transmis à la procédure de traitement du formulaire (cf. section 3.3). Le formulaire est transmis à l'URL indiqué par `ACTION` grâce à l'activation du bouton de commande de `TYPE submit`.

3.2 Le formulaire interprété par un client WWW

Un client WWW ayant reçu le code HTML de la figure 1 affiche la page suivante :

Un formulaire exemple

Un champ textuel simple :

Un champ textuel sur plusieurs lignes :

Des boutons type radio :

Modulation de fréquence

Grandes Ondes

Petites Ondes

Des choix multiples :

un

deux

trois

Un menu déroulant :

Un mot de passe :

Un champ de contexte caché :

Le bouton d'émission : Le bouton d'annulation :

3.3 La transmission par HTTP du formulaire

Lorsque l'émission du formulaire est demandée, le client effectue une requête HTTP comme dans l'exemple suivant :

```
POST /test-cgi-form.cgi HTTP/1.0
Accept: text/plain
Accept: application/x-html
Accept: application/html
Accept: application/postscript
.....
Accept: */*
User-Agent: NCSA Mosaic for the X Window System/2.4 libwww/2.12 modified
Content-type: application/x-www-form-urlencoded
Content-length: 115
simple=Bonjour+%E0+tous+lignes=Ce+texte+est+modifiable&radio=30&
qcm=1&qcm=2&menu=50 F&passwd=en+clair& cache=secret
```

Les champs sont transmis derrière l'en-tête MIME `Content-length`, ils sont séparés entre eux par le caractère `&`, les espaces sont remplacés par des `+`, les éventuels `&`, `/` ... sont encodés et remplacés par leur code ascii précédé du caractère `%` (c'est également le cas du caractère « à » qui est remplacé par `%E0`).

En réceptionnant cette requête POST le serveur HTTP ciblé passera le contrôle à la procédure citée en argument (`/test-cgi-form.cgi` ici) en lui transmettant les champs du formulaire sur son entrée standard. Ce procédé est propre à la méthode POST, dans le cas de la méthode GET les champs sont récupérés dans une variable d'environnement.

3.4 La procédure qui émet et réceptionne le formulaire

Les programmes se déroulant à l'initiative d'un serveur HTTP (les *CGI scripts*) peuvent être écrits dans n'importe quel langage produisant des fichiers exécutables. Ci-dessous deux exemples,

l'un utilisant l'interpréteur de commande sh, l'autre utilisant le langage PERL. Ils traitent les paramètres transmis par le formulaire présenté par la figure 1.

3.4.1 Une procédure CGI écrite en sh

L'exemple ci-dessous utilise CgiParse fourni avec le serveur HTTP du CERN.

```
#!/bin/sh

echo 'Content-Type: Text/html'
echo ''

case $REQUEST_METHOD in
GET)
cat <<FINHML
<HML>
<HEAD>
<TITLE>Un formulaire exemple</TITLE>
</HEAD>
<BODY>
<H1>Un formulaire exemple</H1>
<FORM ACTION="http://www.univ-rennes1.fr/test-cgi-form.cgi"
METHOD="POST">
Un champ textuel simple : <INPUT NAME="simple" SIZE=30> <P>
Un champ textuel sur plusieurs lignes : <P>
<TEXTAREA NAME="lignes"
ROWS=10 COLS=40> Ce texte est modifiable.
</TEXTAREA>
<P>
<STRONG>Des boutons type radio : </STRONG> <P>
<INPUT TYPE="radio" NAME="radio" VALUE="FM" >Modulation de fréquence <BR>
<INPUT TYPE="radio" NAME="radio" VALUE="GO" CHECKED> Grandes Ondes <BR>
<INPUT TYPE="radio" NAME="radio" VALUE="PO"> Petites Ondes <P>
<STRONG>Des choix multiples : </STRONG> <P>
<INPUT TYPE="checkbox" NAME="qcm" VALUE=1> un <BR>
<INPUT TYPE="checkbox" NAME="qcm" VALUE=2> deux <BR>
<INPUT TYPE="checkbox" NAME="qcm" VALUE=3> trois <P>
Un menu déroulant :
<SELECT NAME="menu">
<OPTION SELECTED>50F
<OPTION>60F
<OPTION>70F
<OPTION>100F
</SELECT><P>
Un mot de passe : <INPUT TYPE="password" NAME="passwd" SIZE="40">
<P>
Un champ de contexte caché : <INPUT TYPE="hidden" NAME="cache" value="secret"> <BR>
Le bouton d'émission : <INPUT TYPE="submit" VALUE="Envoyer">
Le bouton d'annulation : <INPUT TYPE="reset" VALUE="Annuler">

</FORM>
</BODY>
</HML>
FINHML
;;
POST)
echo '<HML>'
echo '<HEAD>'
echo '<TITLE>Les champs du formulaires </TITLE>'
echo '</HEAD>'
echo '<BODY>'
echo '<H1> Les champs du formulaire</H1>'
eval `/usr/local/cern-http/bin/cgiparse -form`
echo '<HR>'
echo "Le champ textuel simple : $FORM_simple '<BR>'
echo "Le champ textuel sur plusieurs lignes : '<BR>' $FORM_lignes '<BR>'
echo "Les boutons de type radio : $FORM_radio '<BR>'
echo "Les choix multiples : $FORM_qcm '<BR>'
echo "Le menu : $FORM_menu '<BR>'
echo "Le mot de passe : $FORM_passwd '<BR>'
echo "Le contexte caché : $FORM_cache '<BR>'
echo '<HR>'
echo "Mais également les entêtes HTTP : '<BR>'
echo "User-Agent : $HTTP_USER_AGENT '<P>'
echo "et aussi divers éléments de contexte : '<BR>'
echo "Le client :$REMOTE_HOST '<BR>'
echo "La taille des paramètres :$CONTENT_LENGTH
echo '<HR>'
echo '</BODY>'
echo '</HML>'

```

```
;;
*)
    echo '<HTML>'
    echo Hum ...
    echo '</HTML>'
;;
esac
```

Le contrôle est passé à la procédure de deux façons :

- à la suite d'une requête HTTP de type GET . Dans ce cas, il s'agit d'une demande d'émission du formulaire, il suffit de générer le code HTML correspondant ;
- à la suite d'une requête de type POST , l'appel à la procédure *cgiparse* avec l'argument *-form* restitue les champs du formulaire dans une variable accessible par \$FORM _ suivi du champ NAME correspondant. Ainsi \$FORM _simple contient la valeur du 1er champ du formulaire dépouillé des éventuels caractères d'échappements.

Le document retourné au client WWW peut être de n'importe quel type. On précise ici Content-Type: Text/html suivi d'une ligne vide. La présence de la ligne vide est très importante, elle stipule au client que sont omis des éléments de HTTP (Last-modified , Content-length , ...). Si cette ligne vide est absente, le client ne sait pas où débutent les spécifications HTML du document résultant.

3.4.2 Une procédure CGI écrite en PERL

```
#!/usr/local/bin/perl
#
require "cgi-lib.pl";

$method = $ENV{REQUEST_METHOD};
&print_top;

if ( $method eq "GET" ) {
    print <<"FINHTML";
    <H1>Un formulaire exemple</H1>
    <FORM ACTION="http://www.univ-rennes1.fr/test-cgi-form-perl. cgi"
    METHOD="POST">
        Un champ textuel simple : <INPUT NAME="simple" SIZE=30> <P>
        Un champ textuel sur plusieurs lignes : <P>
        <TEXTAREA NAME="lignes"
            ROWS=10 COLS=40> Ce texte est modifiable.
        </TEXTAREA>
        <P>
        <STRONG>Des boutons type radio : </STRONG> <P>
        <INPUT TYPE="radio" NAME="radio" VALUE="FM" >Modulation de fréquence <BR>
        <INPUT TYPE="radio" NAME="radio" VALUE="GO" CHECKED> Grandes Ondes <BR>
        <INPUT TYPE="radio" NAME="radio" VALUE="PO"> Petites Ondes <P>
        <STRONG> Des choix multiples : </STRONG> <P>
        <INPUT TYPE="checkbox" NAME="qcm" VALUE=1> un <BR>
        <INPUT TYPE="checkbox" NAME="qcm" VALUE=2> deux <BR>
        <INPUT TYPE="checkbox" NAME="qcm" VALUE=3> trois <P>
        Un menu déroulant :
        <SELECT NAME="menu">
            <OPTION SELECTED>50F
            <OPTION>60F
            <OPTION>70F
            <OPTION>100F
        </SELECT><P>
        Un mot de passe : <INPUT TYPE="password" NAME="passwd" SIZE="40">
        <P>
        Un champ de contexte caché : <INPUT TYPE="hidden" NAME="cache" value="secret"> <BR>
        Le bouton d'émission : <INPUT TYPE="submit" VALUE="Envoyer">
        Le bouton d'annulation : <INPUT TYPE="reset" VALUE="Annuler">
    </FORM>
    FINHTML
} else {
```

```
if ($method ne "POST") {
    &error_post;
} else {
    &readParse;
    %ar = %in;
    print <<"FINHTML";
    <HR>
    Le champ textuel simple : %ar{simple} <BR>
    Le champ textuel sur plusieurs lignes : <BR> %ar{lignes} <BR>
    Les boutons de type radio : %ar{radio} <BR>
    Les choix multiples : %ar{qm} <BR>
    Le menu : %ar{menu} <BR>
    Le mot de passe : %ar{passwd} <BR>
    Le contexte caché : %ar{cache} <BR>
    <HR>
    Mais également les en-têtes HTTP : <BR>
    User-Agent : %ENV{HTTP_USER_AGENT} <BR>
    et aussi divers éléments de contexte : <BR>
    Le client : %ENV{REMOTE_HOST} <BR>
    La taille des paramètres : %ENV{CONTENT_LENGTH}
    <HR>
FINHTML
}
}
&print_footer;

sub print_top{
print <<"EndOfTop";
Content-Type: Text/html

<HTML>
<HEAD>
<TITLE>Un formulaire exemple</TITLE>
</HEAD>
<BODY>
EndOfTop
}

sub print_footer{
print <<"EndOfFooter";
</BODY>
</HTML>
EndOfFooter
}

sub error_post{
print <<"FIN";
HUM ...
FIN
}
```

3.5 Le résultat produit

```
Les champs du formulaire

Le champ textuel simple : Bonjour à tous
Le champ textuel sur plusieurs lignes :
Ce texte est modifiable.
Les boutons de type radio : GO
Les choix multiples : 1, 2
Le menu : 50F
Le mot de passe : en clair
Le contexte caché : secret

Mais également les en-têtes HTTP :
User-Agent : NCSA Mosaic for the X Window System/2.5 libwww/2.12 modified

et aussi divers éléments de contexte :
Le client :servix.univ-rennes1.fr
La taille des parametres :129
```

Il convient de bien noter la valeur du champ correspondant au choix multiple, son interprétation reste à la charge du programmeur. L'exemple ci-dessus est le résultat de la procédure écrite en sh (utilisant *CgiParse*). La procédure écrite en PERL (utilisant *cgi-lib*) sépare les différentes réponses d'un choix multiple par le caractère nul.

Le standard *CGI* procure divers renseignements sous la forme de variable tels que :

- `REQUEST _METHOD` , la méthode utilisée pour accéder à la procédure ;
- `QUERY _STRING` , les paramètres émis par le client dans le cas d'une requête utilisant la méthode `GET` ;
- `REMOTE _HOST` , le nom de la machine cliente (si possible);
- `REMOTE _ADDR` , l'adresse IP de la machine cliente ;
- `HTTP _USER _AGENT` , l'identifiant du client WWW demandeur ;
- ...

3.6 Remarques

La mise au point des *CGI scripts* n'est pas aisée, les erreurs ne sont pas facilement décelables car ils se déroulent sous le contrôle d'un serveur HTTP qui redirige ses sorties.

4 Les images réactives

4.1 Les principes

La présentation d'images réactives (parfois dites *cliquables*) par les clients WWW facilite la réalisation d'interfaces graphiques conviviales. Les principes de fonctionnement sont les suivants :

- une image à diffuser par un serveur HTTP est découpée en régions, à chacune d'entre elles est associée une action à réaliser (sous la forme d'un URL). Pour découper une image en régions il est possible d'utiliser des outils (sur une plate-forme Unix) tels que *ImageMagick*³ ou *xv*⁴ la configuration des actions associées dépend du serveur HTTP utilisé (s'agissant du NCSA HTTPD, la lecture de *Graphical Information Map Tutorial*⁵ est recommandée) ;
- l'image est présentée aux clients WWW par l'intermédiaire d'une ancre ayant la structure suivante :

```
<A HREF="URL d'une procédure"> <IMG SRC="path de l'image" ISMAP> </A>
```

C'est l'attribut `ISMAP` qui indique au client que l'image est réactive, et qu'il convient de noter les coordonnées du pointeur de la souris sur sollicitation de l'utilisateur ;

- les coordonnées de l'endroit pointé sont transmises au serveur HTTP derrière l'URL de la procédure de traitement (en utilisant une requête HTTP de type `GET`) comme dans l'exemple suivant :

```
GET /cgi-bin/imagenap/bretagne?542,228 HTTP/1.0
Accept: text/plain
Accept: application/x-html
Accept: application/html
.....
Accept: */*
User-Agent: NCSA Mosaic for the X....
```

Dans l'exemple ci-dessus, `/bretagne` est un paramètre transmis à la procédure `/cgi-bin/imagenap`.

4.2 Remarques

Les interfaces utilisateurs utilisant des images réactives ne sont pas toujours bien comprises des usagers. En effet le client WWW n'a pas connaissance du découpage en régions, il ne peut donc pas signaler (par un effet vidéo) les zones sensibles de l'image.

3. <ftp://ftp.x.org/contrib/applications/ImageMagick>

4. <http://www.univ-rennes1.fr/pub/X11/contrib/xv-3.00.tar.Z>

5. <http://hoohoo.ncsa.uiuc.edu/agoyal/mapping.html>