

Les formulaires



L'importance des formulaires en ASP n'est plus à démontrer : ils permettent de mémoriser en vue d'un traitement ultérieur les informations laissées par les internautes. Mais quand formulaire rime avec ASP, il est nécessaire de prendre en compte plusieurs paramètres, que ce soit pour en améliorer l'interface... ou éviter les erreurs!

Ici, nous parlerons [sécurité et différence](#) entre les deux méthodes d'envoi de formulaires. Côté interface nous verrons comment [personnaliser les menus déroulants](#), et [rediriger un visiteur sur un site en fonction de son choix](#). Enfin, plus technique, il faudra [corriger le bug de l'apostrophe](#) bien connu des développeurs, et [récupérer les cases à cocher](#) sélectionnées...

1. Méthode GET ou méthode POST?

La différence principale entre les deux méthodes d'envoi du formulaire est simple : POST cache les informations pendant la transmission, tandis que GET assimile le formulaire à un lien de type "page.asp?variable=valeur".

Pour être simple, la méthode GET ne crypte ni ne sécurise la transmission des informations : elle peut être interceptée et les données récupérées... De même, on peut forcer l'accès à vos pages de traitement des formulaires... sans passer par les formulaires (en utilisant page.asp?nom=un_nom)! Risqué si ils permettent l'accès à un espace sécurisé...

2. Création d'une liste déroulante à partir du contenu d'un champ

Créer une liste déroulante (ou des cases à cocher, des boutons radios, etc...) à partir d'un champ d'une table peut s'avérer très utile : en effet, si vous en changez le contenu, la liste se met à jour automatiquement, sans que vous n'ayez à modifier le code de votre page HTML ou ASP.

Voici une liste déroulante en HTML :

... qui correspond au code :

```
<select name="liste" size="1">  
<option value="un">un</option>  
<option value="deux">deux</option>  
</select>
```

et dont vous récupérez la valeur avec un :

```
<%valeur=Request.Form("liste")%>  
(ou request.querystring...)
```

Comment réaliser cette liste à partir d'un champ de formulaire? Il faut ouvrir la base, et sélectionner le champ de la table qui contient les informations ; à chaque enregistrement trouvé, on crée une nouvelle occurrence pour le champ de formulaire :

```

<%'-----ouverture de la base, de la connection, création du recordset
DSN_BASE = "DBQ=" & Server.MapPath("ma_base.mdb") & ";Driver=
{Microsoft Access Driver (*.mdb)};DriverId=25"
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open DSN_BASE
Set rs = Server.CreateObject("ADODB.Recordset")
rs.Open "Select DISCTINCT mon_champ from ma_table" , Conn
'-----note : le DISTINCT permet d'éviter d'afficher deux fois la même valeur%>

<%'-----création du formulaire-----%>
<FORM Method="POST" Action="page.asp">
<select name="liste" size="1">
<%'---boucle ASP pour récupérer toutes les valeurs du champ
rs.MoveFirst
do while not rs.eof%>
<option value="<%=rs("mon_champ")%>"><%=rs("mon_champ")%></option>
<%rs.MoveNext
loop%>
</select>
</FORM>

<%'-----fermetures-----
rs.Close
Set rs=Nothing
Conn.Close
Set Conn=Nothing%>

```

Exemple d'utilisation :

Un service de livraisons sur Paris :

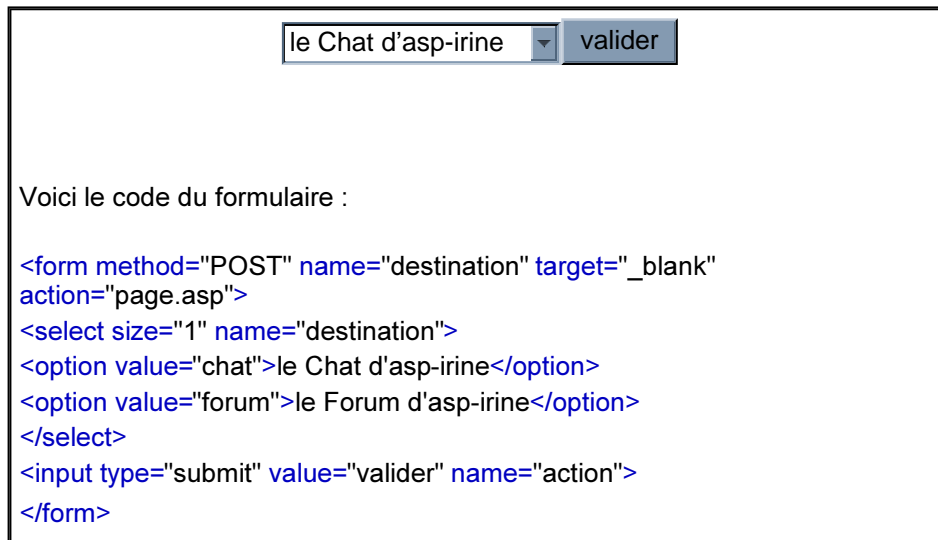
- une table avec un champ "arrondissement" --> un menu déroulant avec tous les arrondissements desservis
- si la zone de livraison s'élargit, il n'y a qu'à rajouter les informations nécessaires dans la table : rien à modifier sur la page du formulaire!

3. Rediriger un internaute par un élément de formulaire

Pour permettre à vos visiteurs une navigation plus souple, vous pouvez leur proposer de se déplacer dans votre site (ou chez vos partenaires) par l'intermédiaire d'éléments de formulaires : menus déroulants, boutons radios...

Après avoir effectué son choix, l'internaute valide le formulaire : en fonction de sa réponse, vous utilisez un `Response.Redirect` que vous faites pointer vers la destination en question!

Voici un exemple de code pour un menu déroulant :



The screenshot shows a web form within a rectangular border. At the top, there is a dropdown menu with the text 'le Chat d'asp-irine' and a small downward arrow. To the right of the dropdown is a button labeled 'valider'. Below the form elements, the text 'Voici le code du formulaire :' is followed by the HTML code for the form.

```
<form method="POST" name="destination" target="_blank"
action="page.asp">
<select size="1" name="destination">
<option value="chat">le Chat d'asp-irine</option>
<option value="forum">le Forum d'asp-irine</option>
</select>
<input type="submit" value="valider" name="action">
</form>
```

Puis, les toutes premières lignes de "page.asp" doivent traiter les informations :

```
<%'-----pour permettre la redirection :----->
Response.Buffer=true
'-----récupération du choix de l'internaute :
lieu=Request.Form("destination")
'-----définition des localisations :
if lieu="chat" then URL="http://www.asp-irine.com/newsfoot/on_line/chat.asp" end if
if lieu="forum" then URL="http://www.asp-irine.com/rubriques/forum/forum.asp" end if
'-----redirection :
Response.Redirect URL%>
```

Exemples d'utilisations :

- un plan du site
- la liste de vos partenaires
- ...

4. Le bug de l'apostrophe... et autres caractères spéciaux

On le sait, c'est arrivé à tout le monde, on fait remplir un champ de formulaire à un internaute, on cherche à insérer les valeurs dans une table, et là...

Erreurs! Erreurs d'exécution ASP!

En fait le problème est lié à certains caractères spéciaux, tels les apostrophes ou les "<" : ce dernier tronque le code HTML, l'autre signifie la fermeture d'une requête SQL, bref la page n'est plus interprétée comme vous le vouliez...

Pour y remédier, il faut remplacer tous ces caractères par les "Entities" (par exemple, "`" à la place de "è"), à l'aide de la commande **Replace(chaine_de_caractères,"a_remplacer","remplacement")** :

```

<%'---r cup ration du contenu du champ de formulaire "contenu"
message=Request.querystring("contenu")
'---remplacement de tous les caract res qui peuvent poser probl me
message=Replace(message,"'", "")
message=Replace(message,"<","&lt;")
message=Replace(message,">","&gt;")
message=Replace(message,vbCrLf,"<br>")

```

Vous pouvez ensuite ins rer sans probl me "message" dans votre base!

5. R cup rer les cases coch es par l'internaute

La diff rence entre cases   cocher et boutons radios, c'est que dans le premier cas, il est possible de s lectionner plusieurs choix.

Le probl me se pose alors pour r cup rer toutes les informations coch es, ni plus, ni moins...

Vous aimez...

☐ le foot
☐ le tennis
☐ le curling
☐ le badmington

Voici le code HTML de ce formulaire : (sans la mise en forme...)

```

<form method="POST" action="page.asp">
<input type="checkbox" name="sport" value="foot">le foot
<input type="checkbox" name="sport" value="tennis">le tennis
<input type="checkbox" name="sport" value="curling">le curling
<input type="checkbox" name="sport" value="badmington">le
badmington
<input type="submit" value="envoyer" name="action">
</form>

```

Sur page.asp, il vous faut alors r cup rer tout ce qui a  t  coch  :

Ben... vous n'aimez pas le sport ou quoi?

Voici le code, d taill  par des commentaires :

```

<%'---z=nombre de cases coch es :
z=request.form("check").Count
---si z n'est pas nul (donc au moins une case a  t  coch e
if request.form("check").Count then%>
Vous avez coch  <%=z%> sport(s)!<br>Les voici :<br>
<%'---du premier au dernier
for u=1 to z

```

```
'---on récupère la valeur cochée dans la variable "val", et on la traite  
val=request.form("check")(u)%>  
le <%=val%>  
<%if val="foot" then%> (allez les Girondins!!!) <%end if%><br>  
<%next  
end if%>  
<%'---si aucune case n'a été cochée  
else%>  
Ben... vous n'aimez pas le sport ou quoi?<br>  
<%end if%>
```

Exemples d'utilisation (au niveau du traitement des valeurs cochées) :

- insertion/suppression de données cochées dans une table ;
- paramétrage de l'affichage ;
- statistiques ;
- ...



www.asp-irine.com