4D Cookies

Par
Eric SALTZEN, Technicien support clients 4D-US
Note technique 4D-200011-30-FR
Version 1

Date 1 Novembre 2000

Résumé

Cette note technique et la base exemple vous fournissent les bases pour personnaliser la session entre vos clients et votre site WEB par l'intermédiaire des cookies.

4D Notes techniques

Introduction

Un cookie est une courte chaîne de caractères déposée dans un fichier de votre ordinateur par le navigateur, afin que le serveur puisse obtenir des renseignements sur le nombre de visites, les choix de navigation etc...

Cette note technique suit ce plan :

Protocoles

Un protocole régit le transport de l'information entre les systèmes connectés à Internet. Votre navigateur (Netscape™ ou Internet Explorer), un serveur 4D sont des exemples de systèmes.
Nous utiliserons ici le protocole http (" HyperText Transport Protocol "). Il fonctionne selon le modèle requête/réponse :

- votre navigateur demande une page HTML (requête), le serveur répond à cette demande (réponse). Ce protocole se nomme " stateless ", c'est-à-dire que chaque requête est indépendante des requêtes précédentes et futures.

Lorsqu'un serveur retourne un objet HTTP à un client, il peut aussi envoyer une paire " name=value " que le client stockera. Le contenu de cet objet est une description des URLs pour lesquelles cet état est valide, indiquant le domaine et le chemin de l'URL.

Toutes les futures requêtes HTTP faites par le client via ces URLs, utiliseront les cookies associés à ceux-ci et retourneront la valeur courante de l'objet et l'état du client à sa dernière connexion au serveur. L'objet décrivant cet état est appelé un cookie (sans raison particulière d'après Netscape™).

Le formatage d'un cookie envoyé par un navigateur WEB lors d'une requête http ressemble à ceci : Cookie: cookieName=cookieValue

Cette ligne sera incluse dans les requêtes destinées à un domaine seulement et à un chemin d'accès spécifié par le cookie original. Quand le serveur veut fixer la valeur d'un cookie, il envoie :

"Set-Cookie: cookieName=cookieValue; expires=Sat, 31-Dec-2005 08:00:00 GMT; path=/"

Il faut noter que vous ne pouvez pas voir le contenu d'un cookie en exécutant la ligne du menu " Source " à partir d'un navigateur WEB : il vous montrera la source du code HTML envoyé par le serveur WEB.

Le cookie permet de stocker une information très courte et ensuite de la récupérer à tout moment. Il est très important de comprendre que cette information est accessible uniquement par celui qui l'a créé.

C'est l'"Entête HTTP " qui, dans le processus WEB de 4D, représente le paramètre $2.

Ne confondez pas l'Entête HTTP avec le< HEAD> qui marque le début d'un document formaté en HTML. Ceci expliquant, pour la commande 4D "FIXER ENTETE HTTP", la manière d'ajouter une information à l'entête HTTP que 4D enverra comme réponse aux requêtes qu'il recevra.

Les cookies sont utilisés afin que le navigateur se souvienne d'une information spécifique que le serveur WEB pourra utiliser ultérieurement (ceci permettant entre autres la gestion de l'état de la session). Lorsqu'un utilisateur est sur le WEB, tous les cookies que les serveurs peuvent envoyer au navigateur WEB sont chargés dans la mémoire de l'ordinateur.

Quand vous quittez le " Navigateur ", tous les cookies qui n'ont pas expiré sont écrits sur le disque afin de pouvoir être rechargés lors d'une session suivante. Si vous utilisez Netscape™ Navigator sous MacOS, ce fichier est nommé "MagicCookie", sous Unix il est appelé "cookies", et sous Windows, "cookies.txt".".

Vous pouvez éditer ce fichier avec un éditeur de texte pour examiner le contenu de ces cookies, ou bien les supprimer pour vous en débarrasser.

Si vous utilisez Internet Explorer™ sous Windows ces informations sont stockées dans de multiples fichiers dans le répertoire Windows, dans un sous-répertoire nommé "cookies."
Vous avez la possibilité de les supprimer, mais non de les éditer.

Si vous ouvrez un cookie avec un éditeur de texte, vous verrez que son contenu ressemble à une suite apparemment incohérente de mots, de lettres et de chiffres.

Voici un exemple de cookie :

```
Nombre_visite
5
www.4D.fr/
0
2884424960
29366115
4268517408
29353627
*
```

Nous pouvons décrypter les deux premiers éléments :

- Name=Value (ici Nombre_visite=5) : il s'agit de la valeur définie par le serveur pour identifier le cookie. Si cet élément est assez explicite, cette valeur vous permet de comprendre l'information enregistrée. En l'occurrence, ce cookie a enregistré 5 visites sur les pages du site qui l'a envoyé.

- Nom_de_domaine et chemin d'accès (ici www.4D.fr/) : très important, ce champ détermine quel serveur a accès au cookie. Dans ce cas, il s'agit du site www.4D.fr.

Le signe " / " qui suit le nom de domaine précise que toutes les pages du serveur 4D. fr peuvent accéder au cookie.

- Parfois, vous retrouverez d'autres éléments comme la date d'expiration du cookie ou le terme "secure" qui indique que le cookie ne sera transmis que lors d'une connexion sécurisée

CookieMonster4D

" CookieMonster4D " est une base de données (MacOS et Windows) qui nous montre comment utiliser un cookie dans l'entête d'une requête HTTP envoyée par un navigateur, comment vérifier un cookie chez un internaute déjà venu, et comment écrire un cookie pour un nouveau visiteur (quelqu'un n'ayant jamais accédé au site, ou qui a effacé le cookie depuis sa dernière visite).

Si vous examinez les "Propriétés de la Base", vous verrez que CookieMonster4D démarre en mode sans contexte et n'a aucune "Racine HTML Par Défaut" ou "Page Démarrage Par Défaut". La base de données contient trois méthodes : Sur connexion WEB, ParseCookies et COMPILER_WEB.

Les pages HTML pour le site ont été créées à l'aide de Macromedia™ Dreamweaver et Flash™, avec les appels appropriés à 4DVAR et 4DCGI insérés comme balises commentaires dans XML tels que "<!--4DVAR userName-->" pour traiter les données en dynamique depuis 4D.

Détaillons la méthode "Sur connexion WEB".

Deux cas de figure :
- Le visiteur n'a pas de cookie sur sa machine, il vient donc pour la première fois sur le site
- Il a déjà un cookie, il est donc déjà venu visiter le site.

1 - Sur Connection WEB

```
  ` Méthode base : Sur connexion WEB

C_TEXTE($1;$2;$3;$4;$5;$6;userName)
C_ENTIER(numCookies)
TABLEAU  TEXTE(cookieName;0)
TABLEAU  TEXTE(cookieValue;0)

  ` vide la sélection précédente dans ce process WEB.
REDUIRE  SELECTION([WebVisitors];0)
Au  cas  ou
     ` demande l'origine d'un site ou, emploie une page 4DCGI

  : (($1="/") | ($1="/4DCGI/home.shtm") | ($1="/4DCGI/direct.htm") | ($1="/4DCGI/schedule.htm") |
($1="/4DCGI/specials.htm") | ($1="/4DCGI/shots.htm") | ($1="/4DCGI/AssignName"))
    numCookies:=ParseCookies ($2;->cookieName;->cookieValue)
```

```
    $primaryID:=Chercher dans tableau(cookieName;"primaryID")
   Si ($primaryID>0)
    CHERCHER([WebVisitors];[WebVisitors]Cookie_ID=Num(cookieValue{$primaryID}))
   Fin de si
   Si (($primaryID=-1) | (Enregistrements trouves([WebVisitors])=0))
      ` Ne trouve pas de cookies dans " navigateur response ",
      ` ou le cookie n'est pas un enregistrement courant.

    Si (Enregistrements trouves([WebVisitors])=0)
                  ` Ne trouve pas d'enregistrements courants. Il crée un ID unique en calculant
                  `le nombre de secondes écoulées depuis le 01/01/2000, qu'il convertit en
                  `Chaîne. En y ajoutant 2 chiffres qui représente les centièmes de seconde,
                  `cela nous donne un intervalle, de la valeur du cookie, qui est compris entre
                  `0 et (2^31) ?1. En passant par un numérique qui va fixer ses limites de -2^31
                  `jusqu'à zéro, qui nous permet une valeur unique de 497 jours. Nous ne pouvons
                  `rencontrer le même cookie tous les centièmes de seconde (8.6 millions par jour)
                  `Ceci garanti approximativement une ID unique pendant 1,36 années, pour tous
                  `les visiteurs de notre site.  En cas de dépassement, nous doublerons les
                  `vérifications de l'unicité de la valeur du cookie. Il en générera donc un autre
                  `s'il y a un  problème avec le cookie existant.

     Repeter
                   $tempCookie:=Num(Chaine(Date du jour-!01/01/00!*86400+Heure courante)
+Chaine(Nombre de millisecondes%100))
       CHERCHER([WebVisitors];[WebVisitors]Cookie_ID=$tempCookie)
      Jusque (Enregistrements trouves([WebVisitors])=0)
      CREER ENREGISTREMENT([WebVisitors])
       [WebVisitors]Cookie_ID:=$tempCookie
    Fin de si

     [WebVisitors]DateVisited:=Date du jour
     [WebVisitors]TimeVisited:=Heure courante
     [WebVisitors]RequestedURL:=$1
     [WebVisitors]HTTP_Header:=$2
      [WebVisitors]ClientIPAddress:=$3
    STOCKER ENREGISTREMENT([WebVisitors])
      ` nos cookies expirent le 31 décembre 2005.
     FIXER ENTETE HTTP("Set-Cookie: primaryID="+Chaine([WebVisitors]Cookie_ID)+"; expires=Sat,
31-Dec-2005 08:00:00 GMT; path=/")
     lastDate:=Chaine([WebVisitors]DateVisited)
     lastTime:=Chaine([WebVisitors]TimeVisited)
      IPAddress:=[WebVisitors]ClientIPAddress
     URLRequested:=[WebVisitors]RequestedURL
     requestHeader:=Caractere(1)+Remplacer chaine ([WebVisitors]HTTP_Header;Caractere(13)
+Caractere(10);"<br>")
       ` demande à l'utilisateur d'entrer un nom pour les enregistrements.
    ENVOYER FICHIER HTML("index.htm")
   Sinon
     [WebVisitors]DateVisited:=Date du jour
     [WebVisitors]TimeVisited:=Heure courante
     [WebVisitors]RequestedURL:=$1
     [WebVisitors]HTTP_Header:=$2
      [WebVisitors]ClientIPAddress:=$3
    Si (Longueur(userName)#0)
                  ` depuis l'index.htm formulaire, s'il est présent
       [WebVisitors]UserName:=userName
    Fin de si
```

```
    STOCKER  ENREGISTREMENT([WebVisitors])
    FIXER  ENTETE  HTTP("")
    C_TEXTE(userName;lastDate;lastTime;IPAddress;URLRequested;requestHeader)
     userName:=[WebVisitors]UserName
     lastDate:=Chaine([WebVisitors]DateVisited)
     lastTime:=Chaine([WebVisitors]TimeVisited)
      IPAddress:=[WebVisitors]ClientIPAddress
     URLRequested:=[WebVisitors]RequestedURL
                                          requestHeader:=Caractere(1)+Remplacer      chaine
([WebVisitors]HTTP_Header;Caractere(13)+Caractere(10)
;"<br>")
    ENVOYER  FICHIER  HTML("home.shtm")
  Fin  de  si

 : ($1="/4DCGI/@")
      ` s'occupe de l'alignement des images sur les pages 4DCGI
   $fileName:=Sous  chaine($1;Position("/4DCGI/";$1)+7)
  PROPRIETES  PLATE  FORME($platform)
  S i ($platform<3)
      ` fixe le chemin des fichiers pour l'utilisation sous Mac OS
    $fileName:=":"+Remplacer  chaine($fileName;"/";":")
  Fin  de  si
   $extension:=Sous  chaine($1;Longueur($1)-3)
  DOCUMENT  VERS  BLOB($fileName;myBLOB)
  ENVOYER  BLOB  HTML(myBLOB;$extension;Vrai)

Fin  de  cas
```

Premièrement, nous déclarons les variables. Ensuite nous évaluons la première expression du AU CAS OU qui vérifiera Si la requête est issue de la racine (" / ") ou Si un 4DCGI est inclu dans un lien de la HOME PAGE (Home.shtm).
Si la requête n'est pas reconnue, nous évaluerons l'expression suivante du Au cas ou qui détecte les requêtes pour les éléments de la page (filtrées par un appel à un 4DCGI) exigeant un traitement de leur URL le retour d'un document.
Si l'évaluation de la deuxième expression du Au cas ou nous renvoie Faux, les pages demandées par un /4DCGI apparaîtront sans leurs images ou tout autre fichier référencé par le document HTML de base.

Revenons à la première expression du AU CAS OU, la première chose que nous faisons est d'appeler la méthode " ParseCookies " en lui passant comme paramètres l'entête HTTP complète et deux pointeurs sur les tableaux texte contenant les paires " Name=Value " pour tous les cookies présents dans l'entête envoyée en réponse

2 - ParseCookies

```
` Methode projet : ParseCookies

` pour plus d'informations sur la façon de formater, d'utiliser et de voir les cookies
`  http://www.cis.ohio-state.edu/htbin/rfc/rfc2109.html

` $1 : en-tête HTTP à examiner (TEXT)
```

```
  ` $2 : pointeur sur tableau texte des noms
  ` $3 : pointeur sur tableau texte des valeurs

  ` $0 : retourne le nombre de cookies trouvés

$HTTPheader:=$1
$nameArrayPtr:=$2
$valueArrayPtr:=$3
$curPos:=0
$numCookies:=0
$location1:=Position("Cookie:";$HTTPheader)
Si ($location1=0)
  $0:=0
Sinon
      ` on supprime tous les caractères avant "Cookie: "
  $HTTPheader:=Sous chaine($HTTPheader;$location1+8)
      ` recherche de notre première paire
  $location1:=1
  Tant que ($location1>0)
      ` recherchons le fin du nom du premier cookie
    $location2:=Position("=";Sous chaine($HTTPheader;$location1))
    INSERER LIGNES($nameArrayPtr->;0)
      ` assignons le nom du cookie
     $nameArrayPtr->{1}:=Sous chaine($HTTPheader;$location1;$location2-1)
      ` recherche le délimiteur Retour chariot
    $returnAt:=Position(Caractere(13);$HTTPheader)
      ` recherche le délimiteur ";"
    $semicolonAt:=Position(";";$HTTPheader)
    Si (($semicolonAt#0) & ($semicolonAt<$returnAt))
      ` le point virgule arrive en premier
      $location1:=$semicolonAt
    Sinon
      ` return arrive en premier
      $location1:=$returnAt
    Fin de si
    Si ($location1=0)
      ` fin du header
      $location1:=Longueur($HTTPheader)+1
    Fin de si
    INSERER LIGNES($valueArrayPtr->;0)
      ` affectation de la valeur au cookie
     $valueArrayPtr->{1}:=Sous chaine($HTTPheader;$location2+1;$location1-$location2-1)
      ` incrementation du compteur de cookie
    $numCookies:=$numCookies+1
    Si ($location1=$returnAT)
      ` terminé, nous vidons le reste du header
      $HTTPheader:=""
    Sinon
      ` pas terminé, nous passons à la paire suivante
      $HTTPheader:=Sous chaine($HTTPheader;$location1+2)
    Fin de si
    Si (Longueur($HTTPheader)=0)
      ` on arrête la boucle d'analyse
      $location1:=0
    Sinon
      ` on continue l'analyse du cookie
      $location1:=1
    Fin de si
```

```
Fin tant que
    ` cookies trouvés en retour de fonction
$0:=$numCookies
Fin de si
```

Après l'exécution de la méthode ParseCookies, la méthode Sur connexion WEB se poursuit par un Chercher dans tableau afin de trouver un cookie dont le nom exact est "primaryID" (choisi arbitrairement pour cet exemple).

Nous présumons que pour cette première visite, le navigateur n'a pas encore de cookie. À cet instant, la sélection dans la table [WebVisitors] est vide. De ce fait la boucle Repeter s'exécutera jusqu'à ce qu'un identifiant unique, pour le cookie, soit généré.

La méthode se poursuit en créant un nouvel enregistrement dans la table [WebVisitors] pour notre internaute sans cookie, et affecte la date et l'heure courante ainsi que l'identifiant unique dans l'entête HTTP à l'aide de la commande FIXER ENTETE HTTP qui insère notre cookie dans l'entête de la réponse.
Nous envoyons alors la page d'introduction ("index.htm") en utilisant ENVOYER FICHIER HTML.

3 - index.htm

voir la source de la page

Cette page contient l'information envoyée par le navigateur dans l'entête HTTP originale. Elle contient également un champ texte permettant à l'utilisateur de saisir son nom pour une future consultation .

Une requête " 4DCGI/AssignName " est envoyée au serveur lors de la validation du formulaire, par le clic sur le bouton " Continue " (bouton Submit), celle-ci déclenchera de nouveau la méthode Sur connexion WEB pendant que le cookie envoyé avec la page sera enregistré sur le poste du navigateur.

Nous entrerons alors dans la première expression du AU CAS OU avec un appel à ParseCookies. Supposons qu'il n'y ait qu'un seul cookie créé pour ce navigateur par notre domaine, $primaryID contiendra la valeur 1, et la requête sur la table [WebVisitors] cherchera un enregistrement avec une valeur du cookie adéquate. À ce niveau, le test du Si qui suit la requête sera complètement omis et l'exécution continuera après le Sinon.

Puis nous mettons à jour les données dans la table [WebVisitors], y compris le userName s'il est disponible lors de la soumission du formulaire, alors l'appel à FIXER ENTETE HTTP() le nettoiera. La phase finale étant l'assignation des valeurs des variables qui sont destinées à paraître sur la page de bienvenue de CookieMonster4D en incluant les balises <!--4DVAR myVar >. Sans se soucier de la requête d'origine (la racine du site, un formulaire envoyé depuis la page index.htm, ou un d'autres liens /4DCGI), CookieMonster4D envoie simplement le fichier " Home.shtm "

3 - home.shtm

voir le source de la page

Le JavaScript utilisé en haut de cette page nous permet de changer l'apparence du curseur lors du " roll over " sur l'onglet de navigation situé à gauche. Les liens référencés dans cet objet font directement appel à des pages HTML statiques, qui, sélectionnées seront servies par 4D sans exécuter la méthode SUR CONNEXION WEB.et dans la hiérarchie du site, et quand ils seront sélectionnés, ils demanderont à 4D d'afficher la page statique sans repasser par la méthode SUR CONNEXION WEB (placez un point d'arrêt pour le vérifier).

Le texte des liens de navigations en bas de chaque page est mis à jour par un appel à /4DCGI qui, force 4D à exécuter la méthode base Sur connexion WEB, permettant aux variables d'être mises à jour à l'intérieur des balises 4DVAR.

Un autre moyen serait d'employer des 4DACTION dans toutes les pages requérant des données dynamiques. Par conséquent, reliant chaque page dynamique à une 4D méthode particulière se chargeant de la mise à jour de ces zones.

Exemples d'utilisations :

- Pages du site les plus visitées
- Recherches effectuées
- Visites terminées par un achat (eCommerce)
- Sur quelle page l'utilisateur quitte votre site