

Scripturi CGI în bash

Primele scripturi CGI

– Ștefan Ciprian Tanasă și Sabin Corneliu Buraga

Înainte de a prezenta exemple de scripturi CGI relativ mai complexe, vom scrie primul nostru program care va trimite cod XHTML navigatorului. Vom denumi acest script bash `salut.cgi` (am utilizat extensia `.cgi` pentru a indica serverului Web că este vorba de un script CGI).

```
#!/bin/bash

# trimitem mai intii campul Content-type
echo "Content-type: text/html"
# obligatoriu o linie vida dupa campul HTTP
echo

# putem continua cu marcaje HTML
echo '<html><head>'
echo '<title>Salutari</title>'
echo '</head>'
echo '<body bgcolor="navy" text="white">'
echo '<h3 align="center">Salut din bash</h3>'
echo '</body></html>'
# am terminat
exit
```

Orice script va avea acces la variabilele de mediu puse la dispoziție de serverul Web. Pentru a le afișa vom folosi `set` sau `printenv`:

```
#!/bin/bash

# trimitem mai intii campul Content-type
# aici text obisnuit
echo "Content-type: text/plain"
echo

# executam 'set'
set
```

Desigur, putem să procesăm doar variabilele de mediu care ne interesează:

```
#!/bin/bash
echo "Content-type: text/html"
echo

echo "<h3>Variabile</h3>"
if [ $REMOTE_HOST ]
then
    echo "<p>Calculator client: $REMOTE_HOST</p>"
else
    echo "<p>Calculator client: <i>adresa necunoscuta</i></p>"
fi
```

```
if [ $REQUEST_METHOD ]
then
    echo "<p>Metoda utilizata: $REQUEST_METHOD</p>"
else
    echo "<p>Metoda utilizata: <i>necunoscuta</i></p>"
fi
echo "<p>Navigator folosit: $HTTP_USER_AGENT</p>"
```

Generarea de conținut dinamic

În funcție de contextul procesării, prin intermediul unui script bash putem trimite spre navigatorul clientului diverse informații, generând în mod dinamic paginile Web.

Pentru început vom scrie un program care va afișa un citat celebru extras aleatoriu dintr-un fișier text de citate. Fiecare linie a fișierului va conține un citat, convenind ca oricare citat să nu ocupe mai mult de o linie.

Scriptul CGI este următorul:

```
#!/bin/bash
# fisierul cu citate
CITATE="citate.txt"
# numarul maxim de citate
NRCITATE=100

# trimitem antetul HTTP
echo "Content-type: text/html"
echo ""
# verificam daca fisierul poate fi citit
if [ ! -r $CITATE ]
then
    echo "Citat inaccesibil."
    exit
fi
# alegem un numar aleatoriu
nrcitat=$((RANDOM%NRCITATE))
nrcitat=$((nrcitat+1))
# preluam citatul din fisier
citata=$( head -$nrcitat $CITATE | tail -1 )
# il putem afisa
echo $citata
```

Acest script va putea fi invocat prin intermediul directivei SSI `exec`:

```
...
<h5 align="right">
<!--#exec cgi="citat.cgi" -->
</h5>
...
```

Plecând de la acest model, cititorul interesat va putea ușor concepe un script care să genereze codul HTML pentru încărcarea unei imagini alese în mod aleatoriu dintr-o mulțime de fișiere grafice.

De asemenea, putem scrie un script care în funcție de sistemul de operare va redirecționa navigatorul spre un anumit document. Fragmentul de cod poate fi:

```
navigator=$HTTP_USER_AGENT
# vedem ce sistem de operare utilizeaza
if
    echo $navigator | grep "Linux" >/dev/null
then
    echo "Location: linux.html"
    echo
    exit
fi
if
    echo $navigator | grep "SunOS" >/dev/null
then
    echo "Location: sunos.html"
    echo
    exit
fi
if
    echo $navigator | grep "Mac" >/dev/null
then
    echo "Location: mac.html"
    echo
    exit
fi
if
    echo $navigator | grep "Win" >/dev/null
then
    echo "Location: windows.html"
    echo
    exit
fi
echo "Location: generic.html"
echo
```

Preluarea datelor prin metoda GET

Am dori de multe ori ca în funcție de datele introduse de utilizator (via un formular Web) să realizăm diferite acțiuni pe server, iar rezultatele să se transmită clientului.

Cea mai simplă cale este de a insera după URI-ul de invocare a scriptului CGI un șir de interogare precedat de caracterul „?” (similând astfel trimiterea datelor dintr-un formular prin metoda GET).

Pentru a afla informațiile despre un utilizator având cont pe o anumită mașină la care nu avem acces decât pe Web, putem invoca un script CGI care să execute comanda `finger` și să returneze rezultatul dorit sub formă de document HTML. Desigur, acel script va trebui instalat pe serverul respectiv.

Codul sursă al acestui program este următorul (vom afișa doar numele real al utilizatorului și data la care și-a citit ultima oară poșta electronică):

```
#!/bin/bash
echo "Content-type: text/html"
echo

if [ "$REQUEST_METHOD" = "GET" ]
then
    utilizator=$QUERY_STRING
else
```

```
    echo "<p>Metoda de invocare eronata</p>"
    exit
fi

if [ -z "$utilizator" ]
then
    echo "<p>Nu a fost specificat numele de cont</p>"
    exit
fi
nume=$( finger -pm $utilizator | grep "Login")
data=$( finger -pm $utilizator | grep "Mail last read")
if [ "$utilizator" ]
then
    echo "<p>Informatii despre <tt>$utilizator</tt></p>"
    echo "<pre>"
    echo $nume
    echo $data
    echo "</pre>"
else
    echo "<p>Nu exista contul <tt>$utilizator</tt>!</p> "
fi
```

Comanda `finger` este apelată cu parametrii `-p` pentru a nu include fișierul `.plan` și cu `-m` pentru a se afișa informații doar pentru numele de cont specificat (de exemplu dacă nu utilizăm această opțiune și cerem informații despre utilizatorul `codrin` am fi obținut datele corespunzătoare tuturor utilizatorilor cu numele *Codrin*).

Presupunând că acest script l-am denumit `finger.cgi` și l-am stocat în directorul `html` al utilizatorului `busaco` pe serverul `www.infoiasi.ro`, îl vom putea invoca prin intermediul următorului URI (în acest caz particular, vor fi afișate informații despre utilizatorul `stanasa`).

<http://www.infoiasi.ro/~busaco/finger.cgi?stanasa>

În continuare dorim să prelucrăm datele preluate dintr-un formular XHTML. Vom considera problema determinării maximului dintre două numere. Formularul Web este:

```
<form action="max.cgi"
    method="GET">
    <p>Introduceți două numere:</p>
    <input type="text" name="nr1" size="3" />
    <input type="text" name="nr2" size="3" />
    <br />
    <input type="submit" value="Afliți numărul maxim" />
</form>
```

Datele transmițându-se prin metoda GET, variabila `QUERY_STRING` va avea întotdeauna forma `nr1=valoare1&nr2=valoare2`, unde `valoare1` și `valoare2` vor fi valorile celor două numere (presupunem că utilizatorul va introduce numai valori corecte). Astfel, cu ajutorul comenzii `cut` putem divide șirul de caractere conținut de `QUERY_STRING` în perechi (*nume de câmp, valoare*). Mai întâi vom extrage fragmentul de șir care precedă „&”, din care vom prelua caracterele de după „=”. Pentru aceasta vom folosi `cut`, stabilind drept delimitatori de câmpuri caracterele „&” și „=”. Similar, vom proceda pentru al doilea număr.

Codul complet al scriptului este următorul:

```
#!/bin/bash
# Prelucrarea datelor prin metoda GET
echo "Content-type: text/html"
echo
```

```
# preluam primul numar
nr1='echo $QUERY_STRING | cut -d"&" -f1 | cut -d"=" -f2'
# preluam al doilea numar
nr2='echo $QUERY_STRING | cut -d"&" -f2 | cut -d"=" -f2'
# calculam maximul
if [ $nr1 -lt $nr2 ]
then
    max=$nr2
else
    max=$nr1
fi
# afisam maximul
echo "<p>Maximul dintre $nr1 si $nr2 este: $max</p>"
```

Cititorul interesat poate aplica acest procedeu pentru un număr variabil de câmpuri. Cu ajutorul comenzilor sed sau tr se pot decodifica valorile câmpurilor transmise către serverul Web.

Preluarea datelor prin metoda POST

În cazul metodei POST, datele vor fi disponibile de la intrarea standard, astfel încât putem folosi read pentru a stoca valoarea șirului de interogare într-o variabilă. Numărul maxim de caractere ce vor trebui citite este dat de variabila CONTENT_LENGTH. Acest lucru va fi posibil grație opțiunii -n a comenzii read care asigură citirea atâtor caractere câte trebuie.

Rescriind scriptul de mai sus, vom avea:

```
#!/bin/bash
# Prelucrarea datelor prin metoda POST
echo "Content-type: text/html"
echo

# citim de la intrare CONTENT_LENGTH caractere
read interogare -n $CONTENT_LENGTH
# preluam primul numar
nr1='echo $interogare | cut -d"&" -f1 | cut -d"=" -f2'
# preluam al doilea numar
nr2='echo $interogare | cut -d"&" -f2 | cut -d"=" -f2'
# calculam maximul
if [ $nr1 -lt $nr2 ]
then
    max=$nr2
else
    max=$nr1
fi
# afisam maximul
echo "<p>Maximul dintre $nr1 si $nr2 este: $max</p>"
```

În loc de final...

Vom vedea într-un articol viitor cum putem prelua, mai comod, informațiile pasate unui script CGI conceput în bash prin intermediul unei biblioteci flexibile – bashlib. Pana atunci, recomandam cititorului sa incerce sa prelucreze (in urma unei interogari prin intermediul unui formular Web) informatiile referitoare la utilizatorii care au conturi pe un server Linux/UNIX (se vor folosi datele stocate de fisierul sistem /etc/passwd).

Autorii sunt cadre didactice la Facultatea de Informatică, Universitatea „Al. I. Cuza” Iași și pot fi contactați la adresele stanasa@infoiasi.ro, respectiv busaco@infoiasi.ro. ■ 98

Bibliografie

- [1] Ramey C., Fox B., Bash Reference Manual, Free Software Foundation, 1998
- [2] Buraga S., Tarhon-Onu V., Tanasă Ș., Programare Web în bash și perl, Editura Polirom, Iași, 2002: <http://www.infoiasi.ro/~cgi/>
- [3] Buraga S., Tehnologii Web, Editura Matrix Rom, București, 2001: <http://www.infoiasi.ro/~busaco/books/web.html>
- [4] Asbury S. et al., CGI How-To, Macmillan Computer Publishing, 1996
- [5] ***, Rapoartele tehnice ale Consorțiului Web: <http://www.w3.org/TR/>