Programmazione server-side: applicazioni CGI

Corso di ***Applicazioni Telematiche***

A.A. 2005-06 – Lezione n.7

Prof. Roberto Canonico

Università degli Studi di Napoli Federico II
Facoltà di Ingegneria

Argomenti

- Programmazione server-side
- Interfaccia CGI
- Uso dei form HTML per l'input di utente
- Configurazione di Apache per l'uso di CGI
- Esempi di programmazione server-side CGI
  - Esempio in C
  - Esempio in Tcl

Programmazione server-side

- Consente di generare “dinamicamente” tutto o parte del documento HTML richiesto da un browser, in seguito all’esecuzione di un programma sulla macchina su cui è in esecuzione il processo “web server”
- Serve a costruire pagine web il cui contenuto è determinato da informazioni fornite dall’utente e/o da dati reperiti da sorgenti esterne
  - Previsioni meteo, quotazioni di borsa, news, siti di e-commerce, sistemi di e-learning, ecc. …
- Il web server deve essere opportunamente istruito e configurato per mandare in esecuzione un programma in modo da generare il contenuto “dinamico”

Programmazione server-side (2)

- Sono attualmente disponibili diverse tecniche di programmazione server side
- Si differenziano per:
  - i linguaggi di programmazione supportati
  - i web server supportati
  - i meccanismi di “aggancio” al web server ed il loro impatto sulle prestazioni
  - il particolare ambito applicativo per il quale sono concepite

Programmazione server-side (3)

- Codice *embedded* in HTML
  - Server Side Includes (SSI) – Apache
  - ASP – Microsoft
  - PHP
  - JSP
- Codice separato, associato ad una URL
  - CGI
  - Java servlet
  - NSAPI o server API

Common Gateway Interface (CGI)

- Primo tentativo di “agganciare” codice esterno ad un processo web server
- Indipendente dal linguaggio
  - un programma CGI può essere un eseguibile generato da un compilatore a partire da un sorgente in un linguaggio di alto livello (es. C) oppure uno script testuale (in un linguaggio tipo Tcl, Perl, Python, …) interpretato a tempo di esecuzione

Common Gateway Interface (CGI) - 2

- Il programma è eseguito al di fuori del processo del server
- Si attiva un nuovo processo per ogni richiesta
- Il codice HTML della pagina richiesta deve essere composto *interamente* da un unico programma CGI
- Il programma produce il documento HTML sullo standard output, anticipato dalla linea:
  - `Content: text/html\n\n`

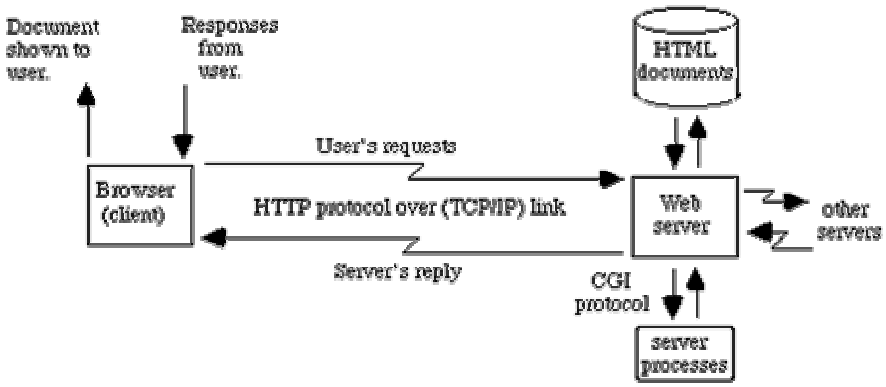
Common Gateway Interface (CGI) - 3

- Il programma CGI produce un output verso il client
- Alcune degli header HTTP sono generati dal proramma CGI
  - L'header "Content-Type" è obbligatorio
  - Altri header che può generare il programma CGI sono Date, Server, Content-Length
- Altri header HTTP sono generati dal web server

Common Gateway Interface (CGI) - 4

- Un programma CGI di solito è attivato da una pagina HTML contenente una FORM, mediante la quale l'utente fornisce un input
- Il programma può ricevere l'input da:
  - stdin se il client usa il metodo POST di HTTP
  - variabili d'ambiente se il client usa GET con parametri su query string
  - command line con ISINDEX (più raro)
- La scelta del meccanismo di input è collegata al metodo di input definito nella FORM

Common Gateway Interface (CGI) - 5



Variabili d'ambiente CGI

Name	Purpose	Example
REQUEST_METHOD	What kind of HTTP request is being handled	GET or POST
SCRIPT_NAME	The path to the script that's executing	/cgi-bin/post_photo.tcl
QUERY_STRING	The query parameters following "?" in the URL	name=mydog.jpg& expires=never
CONTENT_TYPE	The type of any extra data being sent with the request	img/jpeg
CONTENT_LENGTH	How much extra data is being sent with the request (in bytes)	17290

Configurazione di Apache per CGI

```
# … File http.conf
#
# ScriptAlias: This controls which directories contain server scripts.
# ScriptAliases are essentially the same as Aliases, except that
# documents in the realname directory are treated as applications and run
# by the server when requested rather than as documents sent to the client.
# The same rules about trailing "/" apply to ScriptAlias directives as to
# Alias.
#
ScriptAlias /cgi-bin/ "C:/www/cgi-bin/"

<Directory "C:/wwww/cgi-bin/">
    AllowOverride None
    Options ExecCGI
    AddHandler cgi-script .cgi .pl .tcl
    Order allow,deny
    Allow from all
</Directory>

# …
```

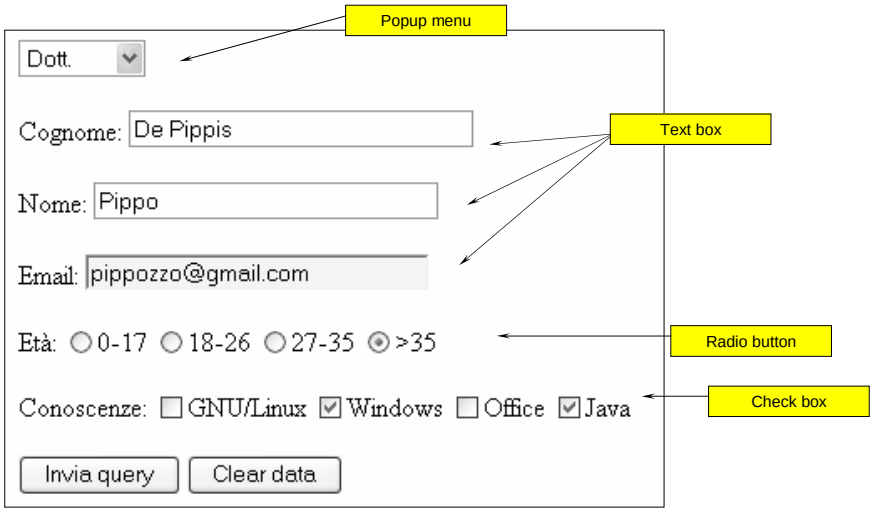
Codifica dei parametri in CGI

- application/x-www-form-urlencoded format
  - space characters → "+"
  - escape (%xx) reserved characters
  - name=value pairs separated by &
- GET:
  `foo.cgi?name=Roberto+Canonico&city=Napoli&nickname=RC`
- POST: include in body of message

Form HTML

- Il meccanismo dei FORM consente di inviare informazioni da un web browser ad un programma in esecuzione lato server
- Per quanto rudimentale, è lo strumento essenziale per trasformare un browser web in un'interfaccia grafica (GUI) universale
- HTML definisce degli elementi interattivi e gli eventi che determinano l'invio delle informazioni

Elementi di una form HTML



Form HTML: esempio

```
<FORM action="/cgi-bin/ex1.exe" method="GET">
<SELECT NAME="titolo" SIZE=1>
    <OPTION SELECTED> Sig.
    <OPTION> Sig.ra
    <OPTION> Dott.
    <OPTION> Dott.ssa
    <OPTION> Prof.
    <OPTION> Prof.ssa
</SELECT><p>
Cognome: <INPUT TYPE="text" NAME="cognome" SIZE=30><p>
Nome: <INPUT TYPE="text" NAME="nome" SIZE=30><p>
Email: <INPUT TYPE="text" NAME="email" SIZE=30><p>
Età:
<INPUT TYPE="radio" NAME="eta" VALUE="a">0-17
<INPUT TYPE="radio" NAME="eta" VALUE="b">18-26
<INPUT TYPE="radio" NAME="eta" VALUE="c">27-35
<INPUT TYPE="radio" NAME="eta" VALUE="d">&gt;35
<p>Conoscenze:
<INPUT TYPE="checkbox" NAME="skill" VALUE="Linux">GNU/Linux
<INPUT TYPE="checkbox" NAME="skill" VALUE="Windows">Windows
<INPUT TYPE="checkbox" NAME="skill" VALUE="Office">Office
<INPUT TYPE="checkbox" NAME="skill" VALUE="Java">Java
<p><INPUT TYPE="submit">
<INPUT TYPE="reset" NAME="resetbutton" VALUE="Clear data">
</FORM>
```

Form con invio dati tramite GET

- I dati vengono inviati attraverso la URL del metodo GET mediante una query string
- Formato application/x-www-form-urlencoded:
  - space characters → "+"
  - escape (%xx) reserved characters
  - name=value pairs separated by &
- Esempio: `http://localhost/cgi-bin/ex1.exe?titolo=Dott.&cognome=De+Pippis&nome=Pippo&email=pippozzo@gmail.com&eta=d&skill=Linux&skill=Windows&skill=Java`
- E' possibile inviare solo una piccola quantità di dati
- I dati restano visibili nel file di log del server!
- Il server inserisce i dati in una *variabile di ambiente* a disposizione del programma CGI: $ENV{'QUERY-STRING'}

Esempio di form con uso di GET

- ```
  <form action="env.cgi" method="GET">
  Enter some text here:
  <input type="text" name="sample_text" size=30>
  <input type="text" name="other_text" size=30>
  <input type="submit"><p></form>
  ```

- **`test.cgi?sample_text=paperino&other_text=qui+quo+qua`**


  ```

Form con invio dati tramite POST

- I dati vengono inviati dal client al server web nel body del metodo POST
- I dati non sono visibili nel file di log del server
- Il programma CGI riceve l'input dal web server tramite stdin

Esempio di form con uso di POST

```
<form action="env.cgi" method="POST">
Enter some text here:
<input type="text" name="sample_text" size=30>
<input type="text" name="other_text" size=30>
<input type="submit"><p></form>
```

- Nessuna apparente differenza per l'utente
- La differenza è nei messaggi HTTP che viaggiano sulla rete e nella struttura del programma CGI che riceve l'input da stdin

Programma CGI "somma due interi" in C

```
#include <stdio.h>
#include <stdlib.h>
unsigned char  *getval(unsigned char *);
int main()
{  int  x,y;
   char *str1,*str2;
   printf("Content-type: text/html\n\n");
   printf("<html><head><title>CGI C Example #2</title></head>\n");
   printf("<body><h1>CGI C Example #2</h1>\n");
   str1 = getval("number1");
   str2 = getval("number2");
   if(str1 == NULL || str2 == NULL)
   {    printf("<p>Input data error\n");   }
   else
   {    x = atoi(str1);
        y = atoi(str2);
        printf("<p>The sum of %d and %d is %d\n",x,y,x+y);
   }
   printf("</body></html>\n");
}
```

getval è definita in un file a parte (cgis.c)

L'eseguibile si chiama cgisum.exe e va copiato nella directory cgi-bin del server

Form HTML associata a cgisum.exe

```
<html>
<head>
<title>CGI Example 2</title>
</head>
<body>
<form action="/cgi-bin/cgisum.exe" method="get">
Enter first number <input type=text name=number1><br>
Enter second number <input type=text name=number2><br>
<input type=submit value="Calculate sum">
</form>
</body>
</html>
```

Programma CGI "somma due interi" in Tcl

```
#!C:/Tcl/bin/tclsh.exe
source GetPostedData.tcl
# Output the appropriate MIME header ...
puts "Content-type: text/html"
puts ""
# Output a complete HTML header ....
puts "<HTML><HEAD>"
puts "<TITLE>Add Up Script</TITLE>"
puts "</HEAD><BODY>"
puts "<H1>This document was generated<BR>"
puts "by a Tcl script."
puts "</H1><P>"
# Obtain the data from the form
GetPostedData
# Calculate the answer
set theAnswer [ expr $FormData(number1) + $FormData(number2) ]
# Send the information back to the browser
puts "<H3>$FormData(number1) plus $FormData(number2) "
puts "is $theAnswer. </H3>"
puts "<P><HR><H3>"
puts "Use the back button to return to "
puts "Tcl scripting for HTML FORMS."
puts "</H3><HR>"
puts "</BODY></HTML>"
```

Interpreta il file come script Tcl

La procedura GetPostedData è definita in uno script esterno

Carica i dati forniti su stdin dalla form HTML nell'array FormData

Form HTML associata a cgisum.tcl

```
<html>
<head>
<title>CGI Example</title>
</head>
<body>
<form action="/cgi-bin/cgisum.tcl" method="post">
Enter first number <input type=text name=number1><br>
Enter second number <input type=text name=number2><br>
<input type=submit value="Calculate sum">
</form>
</body>
</html>
```

Domande?