# Računarske mreže
(2OER5O03)

# CGI - Common Gateway Interface

## Auditivne vežbe

## Šta je CGI?

CGI je interfejs ka Web serveru koji omogućuje proširenje serverske funkcionalnosti.

CGI omogućuje pisanje programa koji na vrlo jednostavan način komuniciraju sa Web serverom.

Inicijalno je bio zamišljen kao veza ka bazama podataka putem Web-a, ali se ubrzo njegova primena proširila na sve vidove aplikacija. Veza sa Web serverom ostvaruje se preko standardnog ulaza/izlaza (pod određenim uslovima i preko komandne linije). Sa ostalim aplikacijama način komunikacije nije ograničen, i može se koristiti sve, od deljive memorije do soket-veze.

## Izbor jezika

- CGI nije ograničen samo na jedan jezik i može se koristiti bilo koji jezik koji:
  - štampa na standardnom izlazu
  - čita sa standardnog ulaza i
  - čita promenljive okruženja (*environment variables*)

## Protokol komunikacije

- Po pristizanju zahteva klijenta, Web server aktivira CGI program, kreirajući novi proces.
- Sve potrebne parametre Web server smešta u promenljive okruženja, a svoj ulaz povezuje na standardni izlaz CGI programa. Podatke korisnika, ukolio se radi o POST metodi, šalje na standardni ulaz CGI programa.
- CGI upisuje podatke koje treba vratiti klijentu na svoj standardni izlaz, a poruke o greškama, koje se pri tome mogu javiti, prihvata Web server i smešta u standardne log datoteke.
- Po ispunjenju zahteva, CGI program se završava, a odgovarajući proces se uništava.

## Karakteristike

- Promenljive okružena (*environment variables*) su imenovani parametri koji prenose informaciju od Web servera ka CGI programu. To ne moraju biti promenljive u okruženju operativnog sistema, ali su najčešće upravo tako implementirane.
- CGI protokol ne vodi računa o praćenju toka sesije. Ako aplikacija zahteva više transakcija da bi ostvarila željeni cilj, sam CGI program mora biti odgovoran za pamćenje stanja između dve transakcije. To se ostvaruje upisivanjem informacija u datoteke ili baze podataka, ukoliko se informacija skladišti na serverskoj strani, ili se informacija putem *cookies*-datoteka prenosi klijentu. Praćenje toka sesije klijenta korišćenjem *cookies*-datoteka je vrlo česta praksa, a informacije iz ovih datoteka se vraćaju serveru pri svakom ponovnom pristupu istoj lokaciji.

## Karakteristike

- Za pisanje CGI programa mogu se koristiti i kompajlerski i skript jezici (C/C++, Perl, TCL, bilo koji UNIX shell, Visual Basic, AppleScript, ...). Korišćenje skript jezika olakšava razvoj, ali usporava odziv. Naročito nije pogodno kod velikih i vremenski kritičnih aplikacija.
- CGI aplikacije se izvršavaju u zasebnom adresnom prostoru, tako da je gotovo nemoguće da ugroze rad Web servera u slučaju lošeg funkcionisanja. To povećava pouzdanost Web sajta. Međutim, iako ne mogu direktno da naškode Web serveru, loše CGI aplikacije utiču na performanse, pogotovu ako sistem ne uspe da očisti posledice "srušene" CGI aplikacije. Svaka instanca CGI aplikacije koja ostane u memoriji troši resurse računara na kome se izvršava server.

## Karakteristike

- Budući da je CGI zaseban proces, on se može propustiti kroz *debuger* bez potrebe za komunikacijom sa spoljašnjim aplikacijama, kao što je Web server. Svi parametri se preuzimaju iz promenljivih okruženja, ali se mogu zadati i ručno, na primer preko komandne linije. Ukoliko u sistemu postoje *Just-in-time debuger* i omogućena je izrada *debug* verzija CGI aplikacije, prilikom nastanka greške, odgovarajući modul preuzima kontrolu nad aplikacijom i dozvoljava praćenje toka izvršenja, ukazujući na liniju koda u kojoj je nastala greška.
- Najbitnija loša osobina CGI-a jesu loše performanse, budući da se novi proces kreira i uništava sa svakom konekcijom, odnosno zahtevom.

## Karakteristike

- CGI ima ograničenu funkcionalnost jer podržava jednostavnu ulogu - aplikacije koja generiše odgovor na zahtev klijenta, tako da se ne može uključiti u druge faze obrade Web zahteva, kao što su autorizacija i prijavljivanje (*log in*).
- Jedna od većih mana CGI protokola jeste i problem sigurnosti, tj. dozvola da se izvrši bilo koji program na serverskoj strani, što potencijalno predstavlja veliku opasnost. Zato se najčešće ograničava lociranje ovakvih programa samo na jedan direktorijum (*cgi-bin*). Takođe, sva obrada obavlja se na serverskoj strani, tako da postoji neželjeni saobraćaj na mreži prilikom korigovanja pogrešnih zahteva.
- Neke mane CGI protokola mogu se otkloniti korišćenjem jednog aktiviranog procesa za opsluživanje više zahteva. Takvo rešenje poznato je pod nazivom **Fast CGI**

# Prednosti

- ***jednostavnost*** - razumljivost, jednostavan protokol za preuzimanje parametara i prosleđivanje rezultata (čitanje sa standardnog ulaza i promenljivih okruženja i štampanje na standardni izlaz),
- ***nezavisnost od jezika*** - CGI aplikacije mogu biti napisane u bilo kom jeziku,
- ***izolacija procesa*** - pošto se aplikacija izvršava u posebnom procesu, rušenje aplikacije ne povlači rušenje Web servera,
- ***prenosivost*** (otvoreni standard) - neki oblik CGI-a je implementiran u svakom Web serveru,
- ***nezavisnost od arhitekture*** - CGI nije ograničen na određeni tip arhitekture servera (jednonitni, višenitni, ... ),
- ***jednostavno testiranje i otkivanje grešaka*** - može se razvijati i testirati nezavisno od Web servera,
- ***relativno stara i dovoljno usavršena tehnologija*** sa mnoštvom knjiga i gotovog koda.

## Mane

- inicijalizacija procesa je vremenski zahtevna,
- potreba za aktiviranjem i uništavanjem spoljašnjeg programa kod svakog zahteva,
- potreba za spoljašnjim mehanizmima za pamćenje stanja, jer CGI protokol nema stanja i ne može upravljati sesijama,
- nemogućnost uključivanja u druge faze obrade Web zahteva, kao što su autorizacija i prijavljivanje.

## Hello, World!

```
#include <stdio.h>
int main()
{
        printf("Content-Type: text/html;charset=us-ascii\n\n");
        printf("<html> <head>\n");
        printf("<title>Hello, World!</title>\n");
        printf("</head>\n");
        printf("<body>\n");
        printf("<h1>Hello, World!</h1>\n");
        printf("</body> </html>\n");
}
```

# MIME tipovi

| *MIME tip* | *Opis* |
|---|---|
| text/html | HyperText Markup Language (HTML) |
| text/plain | Plain text files |
| image/gif | GIF graphics files |
| image/jpeg | JPEG compressed graphics files |
| audio/basic | Sun *.au audio files |
| audio/x-wav | Windows *.wav files |

# CGI prihvatanje zahteva

```
void main(int argc, char* argv[])
{
    std::string reqmethod = getenv("REQUEST_METHOD");
    std::string q_string = getenv("QUERY_STRING");
    int br_byte = atoi(getenv("CONTENT_LENGTH"));
    std::string postInputString;

    if (reqmethod == "GET")    // GET zahtev
    {
        if (!q_string.empty())
        parse(q_string);
    }
    ***
```

# CGI prihvatanje zahteva

```
void main(int argc, char* argv[])
{
***//nastavak
      else if (reqmethod == "POST")    // POST zahtev
      {
         if (!q_string.empty())
               parse(q_string);

         if (br_byte > 0)
         {
               std::cin >> postInputString;
               parse(postInputString);
         }
       }
}
```

# Vraćanje rezultata

```
void SendFile(std::string outName)
{
        ...
        std::ifstream file(outName, std::ios::in | std::ios::binary);
        long size = file.tellg();
        std::cout // "HTTP/1.1 200
                  << "Content-Type: image/jpeg\r\n"
                  << "Content-Length: " << size << "\r\n";
                  << "\r\n";


        /* Postaviti "stdout" na binarni mod, ne koristi se za html i txt*/
        auto result = _setmode(_fileno(stdout), _O_BINARY);
        ***
```

Ako se sadržaj dinamički kreira, potreban je i "Last-Modified:"

## Vraćanje rezultata

```
void SendFile(std::string outName)
{
                ***
        if (result == -1)
                std::cout << "Ne moze se postaviti binarni mod!" << std::endl;
        else
        {
                std::vector<char> buffer;
                buffer.resize(size);
                file.seekg(0, std::ios::beg);
                file.read(buffer.data(), size);
                std::cout.write(buffer.data(), size);
                fflush(stdout);
                _setmode(_fileno(stdout), _O_TEXT);
        }
        file.close();
}
```

# Formatiranje vremena

```
auto now = std::chrono::system_clock::now();
auto in_time_t = std::chrono::system_clock::to_time_t(now);

std::cout << "Last-Modified:"
          << std::put_time(std::localtime(&in_time_t),
                           "%a, %d %b %Y %H:%M:%S GMT")
```

## Neke promenljive okruženja

| Promenljiva | Značenje |
|---|---|
| REMOTE_ADDR | IP klijentske mašine |
| REMOTE_HOST | Naziv klijenta |
| HTTP_ACCEPT | Lista MIME tipova koje klijent prepoznaje |
| HTTP_USER_AGENT | Informacije o klijentu (ime,ver.,OS.) |
| REQUEST_METHOD | GET ili POST |
| CONTENT_LENGTH | Veličina tela POST poruke |
| QUERY_STRING | Zahtev prosleđen GET metodom |
| PATH_INFO | Specificiranje putanje do podataka od strane klijenata. Uglavnom neprimenljivo! |
| PATH_TRANSLATED | |

## GET i POST metode

- GET metoda je jednostavnija i ceo zahtev se zadaje kao sastavni deo URL-a i prosleđuje CGI-u u obliku QUERY_STRING-a
- POST metod se koristi kada:
  - treba preneti veliku količinu podataka (>1024B)
  - parametre ne treba prikazati u okviru URL-a (npr. user-name i password)
- POST metoda parametre CGI-u prenosi standardnim ulazom (čita se sa konzole), a količina podataka, tj. br. bajtova očitava se iz promenljive okruženja CONTENT_LENGTH

# Kodiranje ulaza

- Polja u zahtevu se međusobno odvajaju ampersendom (&)
- Imena promenljivih se od vrednosti odvajaju znakom jednakosti (=)
- Blanko znaci zamenjuju se znakom plus (+)
- Nedozvoljeni znaci zamenjuju se znakom procenta (%) iza kog slede dve heksa-cifre, tj. ASCII kod tog znaka

Primeri:

```
name1=value1&name2=value2&name3=value3
```

```
http://server.elfak.rs/cgi1.exe?name=Petar+Peric&index=238
20
```

## Struktura HTML dokumenta

```
<html>
        <head>
                <title>A Big Bear</title>
         </head>
         <body>
                <h1>A Big Bear</h1>
                <img src="bigbear.gif">
        </body>
</html>
```

## HTML forme

- Omogućuju kreiranje vrlo složenih korisničkih interfejsa sastavljenih od: polja za unos teksta, checkbox, listbox i drugih kontrola
- Komponente interfejsa dodaju se navođenjem odgovarajućih HTML tagova
- Klikom na odgovarajuće dugme formira se upit na osnovu naziva i vrednosti unetih u odgovarajuće kontrole, koji se zatim prosleđuje zadatom CGI-u

## <FORM> tag

- Format:

  <FORM ACTION="..." METHOD=[POST | GET] [ENCTYPE="..."]> ...
  </FORM>

- Osnovni element HTML forme i navodi se u BODY delu
- Ne može se ugnježdavati
- ACTION – kaže browser-u gde da pošalje zahtev, obično se zadaje lokacija CGI programa
- METHOD – definiše kako treba poslati parametre serveru
- ENCODE – definiše kako podaci trebaju biti kodirani od strane browser-a (koristi se samo za POST metod)

# <INPUT> tag

- Format

  <INPUT TYPE="..." [NAME="..."] [VALUE="..."]
  [SIZE="..."][MAXLENGTH="..."] [ic:ccc][SRC="..."] [CHECKED]>

- Najfleksibilniji element forme
- Nema elementa za zatvaranje (ne postoji <\INPUT>)
- Od tipa (TYPE) zavisi koja kontrola se implementira
- Svi ulazni tipovi, sem SUBMIT i RESET zahtevaju ime (NAME)
- Moguće je da dve ili više kontrola imaju isto ime, ali to pretstavlja opsnost, jer ih CGI program ne može razlikovati (javljaju se dva atributa sa istim imenom i različitim vrednostima)

## type = text

- Sintaksa:

  <INPUT TYPE=TEXT NAME="..." [VALUE="..."] [MAXLENGTH="..."]
  [SIZE="..."]>

- Formira *textbox* kontrolu
- VALUE – podrazumevana vrednost koja se inicijalno prikazuje u *textbox*-u
- MAXLENGTH – maksimalna dužina teksta u karakterima
- SIZE – veličina kontrole u karakterima (ako se ne navede podrazumeva se 21)

# type = submit

- Sintaksa:

  <INPUT TYPE=SUBMIT [NAME="..."] [VALUE="..."]>

- Ovo je dugme koje inicira prosleđivanje informacije serveru
- Mora da postoji u formi osim u dva slučaja:
  - postoji samo jedan *textbox*, pritiskom na Enter automatski se poziva *submit* (mada ovo zavisi od *browser*-a)
  - postoji slika na koju se može kliknuti (type=image)
- Par NAME i VALUE koristi se samo ukoliko u formi postoji više SUBMIT dugmeta, pa ih je potrebno razlikovati

# Ostale kontrole

- <INPUT TYPE=RESET [VALUE="..."]> – briše formu i vraća je u prvobitno stanje. VALUE određuje labelu tog dugmeta
- <INPUT TYPE=PASSWORD NAME="..." [VALUE="..."] [MAXLENGTH="..."] [SIZE="..."]> – isto kao TEXT, ali se prilikom kucanja umesto znakovajavljaju zvezdice
- <INPUT TYPE=CHECKBOX NAME="..." VALUE="..." [CHECKED]> – kreira *checkbox* kontrolu. Ako je kontrola "štilirana" šalje se par NAME=VALUE, u protivnom ne šalje se ništa
- <INPUT TYPE=RADIO NAME="..." VALUE="..." [CHECKED]> – kreira *radiobutton* kontrolu. Grupa *radio* dugmića kreira se tako što se svim kontrolama u grupi dodeli isto ime (NAME). CHECKED određuje koje dugme je selektovano po *default*–u.
- <INPUT TYPE=IMAGE NAME="..." SRC="..." [ALIGN="..."]> – prikazuje sliku na koju se može kliknuti. Serveru se prosleđuju koordinate piksela na koji je kliknuto u parametrima *name.x* i *name.y*, gde je *name* ime definisano u NAME atributu. SRC i ALIGN imaju isto dejstvo kao i odgovarajući atributi IMG taga HTML–a.
- <INPUT TYPE=HIDDEN NAME="..." VALUE="..."> – nevidljivo polje. Služi za pamćenje stanja u okviru forme.

# <SELECT> tag

- Format:

```
<SELECT NAME="..." [SIZE="..."] [MULTIPLE]>
<OPTION [VALUE="..."] [SELECTED]> text1 </OPTION>
<OPTION [VALUE="..."] [SELECTED]> text2 </OPTION>
...
</SELECT>
```

- Formira *listbox* sa mogućnošću višestrukog selektovanja (ukoliko je naveden atribut MULTIPLE).
- Vrednosti se navode u zasebnim OPTION tagovima.
- Ako ima više "opcija" od veličine kontrole (broja redova) definisane atributom SIZE, automatski se dodaju *scroll bar*-ovi.
- Ako nije dozvoljena višestruka selekcija i veličina je 1 (ili nije navedena), umesto *listbox*-a dobija se *combobox*.
- Ako nije selektovana nijedna stavka, u zahtevu se neće pojaviti odgovarajući parametar (kao kod *check* ili *radio* dugmića), a ako ih je više selektovano, vrednosti se razdvajaju zarezima.

# <TEXTAREA> tag

- Format:

```
<TEXTAREA NAME="..." [ROWS="..."] [COLS="..."]>
Podrazumevani tekst ide ovde
</TEXTAREA>
```

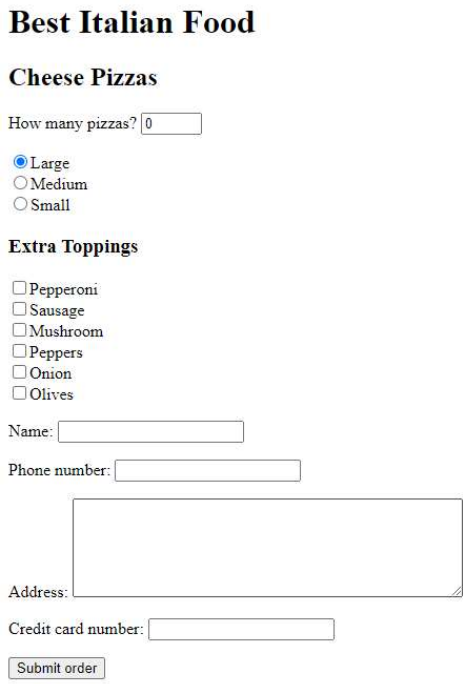
- Omogućuje unos više redova teksta
- ROWS i COLS definišu veličinu polja za unos teksta

# Primer 1

```
<html> <head>
 <title>Comments Form</title>
</head>
 <body>
 <h1>Send us your comments</h1>
 <form action="comments.cgi" method=POST>
<p>Full Name: <input name="name">
<p>Email Address: <input type=text name="email" size=50>
<p>Comments: <textarea name="comments" rows=15 cols=70> </textarea>
<input type=submit value="Submit comments">
<input type=reset value="Clear form">
</form>
</body> </html>
```

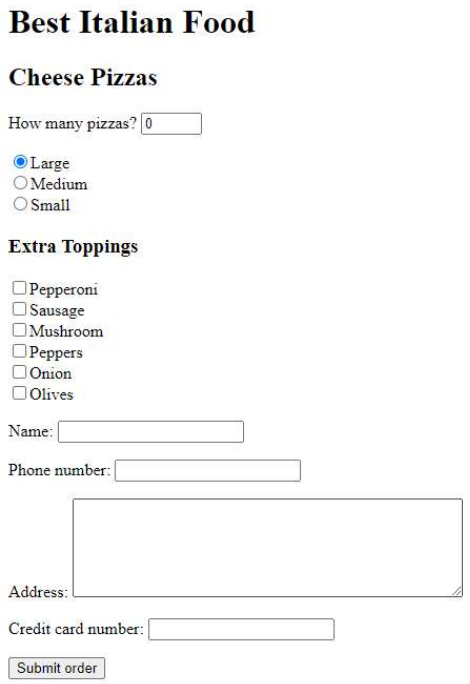
# Primer 2

```
<html> <head> <title>Best Italian Food</title> </head>
<body> <h1>Best Italian Food</h1>
<form action="order.cgi" method=POST>
<h2>Cheese Pizzas</h2>
<p>How many pizzas?

<input name="numpizzas" value="0" size=3 maxlength=3>

<p> <input type=radio name="size" value="large" checked>Large
<br> <input type=radio name="size" value="medium">Medium
<br> <input type=radio name="size" value="small">Small
```



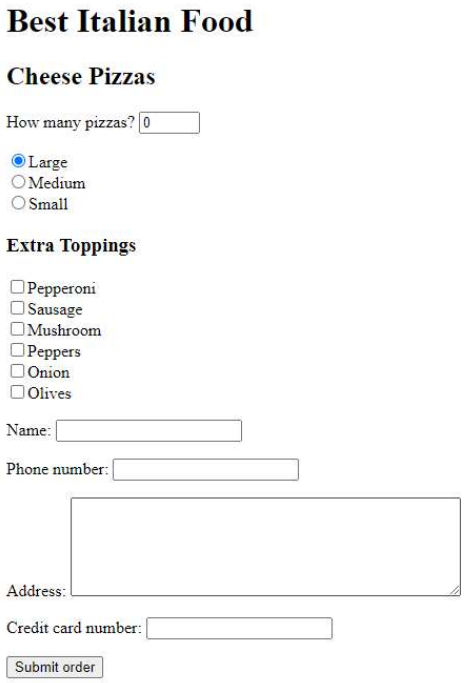
# Primer 2

```
<h3>Extra Toppings</h3>
<p> <input type=checkbox name="topping"
value="pepperoni">Pepperoni
<br> <input type=checkbox name="topping"
value="sausage">Sausage
<br> <input type=checkbox name="topping“
value="mushroom">Mushroom
<br> <input type=checkbox name="topping"
value="peppers">Peppers
<br> <input type=checkbox name="topping"
value="onion">Onion
<br> <input type=checkbox name="topping"
value="olives">Olives<br>
```



# Primer 2

```
<p>Name: <input type=text name="name">
<p>Phone number:
<input type=text name="phone"> <p>Address:
<textarea name="address" rows=6 cols=50> </textarea>
<p>Credit card number:
<input type=password name="creditcard" size=20></p>
<input type=submit value="Submit order"> </form> </body>
</html>
```



# Primer 3

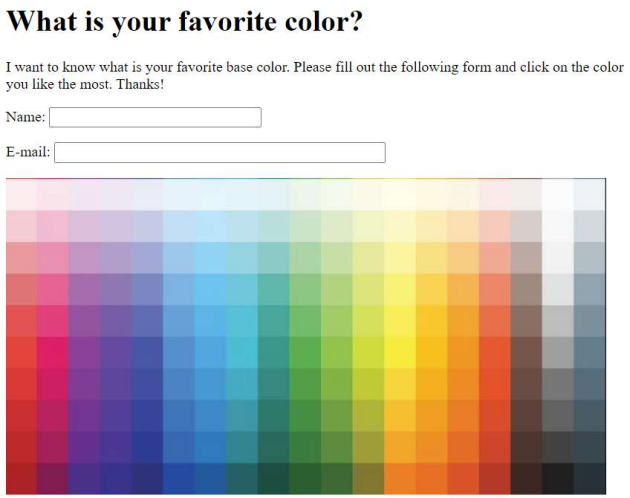

```
<html> <head>
<title>Best paper award</title>
</head>
<body>
<h1>Best paper award</h1>
<p>You can either vote for a candidate or view a candidate's paper.
<form action="vote.cgi" method=POST>
<select name="candidate">
<option>John Doe
<option>Peter Brown
<option>Chris Willings
</select>
 <p>You may <input type=submit name="action" value="vote"> for the above candidate, or you may
<input type=submit name="action" value="view"> his position paper.
</form> </body> </html>
```

candidate=John+Doe&action=vote

candidate=John+Doe&action=view

# Primer 4

```
<html> <head>
<title>Where are you from?</title>
</head>
<body>
<h1>What is your favorite color?</h1>
<p>I want to know what is your favorite base color. Please fill out
the following form and click on the color you like the most. Thanks!
<form action="colorpicker.cgi" method=POST>
<p>Name: <input name="name" size=30>
<p>E-mail: <input name="email" size=50>
<p><input type=image name="region" src="paleta.jpg">
</form>
</body> </html>
```




```
name=John+Doe&email=jschmoe@yourmachine.org&region.
x=100&region.y=180
```

# PITANJA