


---

---

---

---

---

---

---

---

### Common Gateway Interface

- CGI is a standard mechanism for:
  - Associating URLs with programs that can be run by a web server.
  - A *protocol* (of sorts) for how the request is passed to the external program.
  - How the external program sends the response to the client.

Netprog 2002 CGI Programming 2

---

---

---

---

---

---

---

---

### CGI URLs

- There is some mapping between URLs and CGI programs provided by a web sever. The exact mapping is not standardized (web server admin can set it up).
- Typically:
  - requests that start with `/CGI-BIN/` , `/cgi-bin/` or `/cgi/`, etc. refer to CGI programs (not to static documents).

Netprog 2002 CGI Programming 3

---

---

---

---

---

---

---

---

## Request → CGI program

- The web server sets some environment variables with information about the request.
- The web server `fork()`s and the child process `exec()`s the CGI program.
- The CGI program gets information about the request from environment variables.

Netprog 2002 CGI Programming

4

---

---

---

---

---

---

---

---

## STDIN, STDOUT

- Before calling `exec()`, the child process sets up pipes so that `stdin` comes from the web server and `stdout` goes to the web server.
- In some cases part of the request is read from `stdin`.
- Anything written to `stdout` is forwarded by the web server to the client.

Netprog 2002 CGI Programming

5

---

---

---

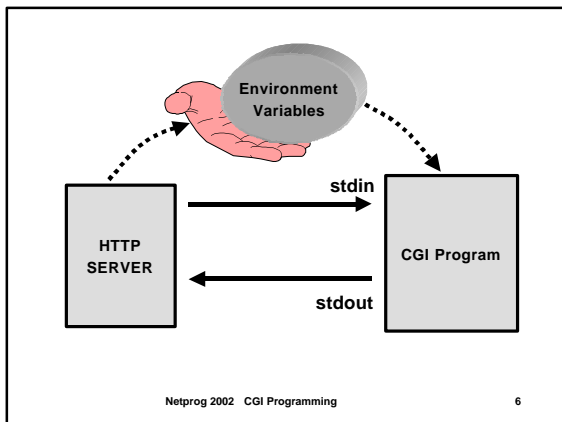
---

---

---

---

---



Netprog 2002 CGI Programming

6

---

---

---

---

---

---

---

---

## Important CGI Environment Variables

`REQUEST_METHOD`

`QUERY_STRING`

`CONTENT_LENGTH`

Netprog 2002 CGI Programming

7

---

---

---

---

---

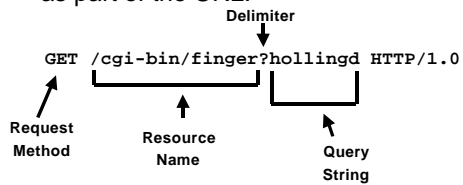
---

---

---

## Request Method: Get

- GET requests can include a *query string* as part of the URL:



Netprog 2002 CGI Programming

8

---

---

---

---

---

---

---

---

`/cgi-bin/finger?hollingd`

- The web server treats everything before the '?' delimiter as the resource name
- In this case the resource name is the name of a program.
- Everything after the '?' is a string that is passed to the CGI program.

Netprog 2002 CGI Programming

9

---

---

---

---

---

---

---

---

## Simple GET queries - ISINDEX

- You can put an `<ISINDEX>` tag inside an HTML document.
- The browser will create a text box that allows the user to enter a single string.
- If an ACTION is specified in the ISINDEX tag, when the user presses **Enter**, a request will be sent to the server specified as the ACTION.

Netprog 2002 CGI Programming

10

---

---

---

---

---

---

---

---

## ISINDEX Example

Enter a string:

```
<ISINDEX ACTION=http://foo.com/search.cgi>
```

Press Enter to submit your query.

If you enter the string "blahblah", the browser will send a request to the http server at `foo.com` that looks like this:

```
GET /search.cgi?blahblah HTTP/1.1
```

Netprog 2002 CGI Programming

11

---

---

---

---

---

---

---

---

## What the CGI sees

- The CGI Program gets `REQUEST_METHOD` using `getenv`:

```
char *method;  
method = getenv("REQUEST_METHOD");  
if (method==NULL) ... /* error! */
```

Netprog 2002 CGI Programming

12

---

---

---

---

---

---

---

---

## Getting the GET

- If the request method is GET:  

```
if (strcasecmp(method,"get")==0)
```
- The next step is to get the query string from the environment variable `QUERY_STRING`

```
char *query;  
query = getenv("QUERY_STRING");
```

Netprog 2002 CGI Programming

13

---

---

---

---

---

---

---

---

## Send back http Response and Headers:

- The CGI program can send back a http status line :

```
printf("HTTP/1.1 200 OK\r\n");
```

- and headers:

```
printf("Content-type: text/html\r\n");  
printf("\r\n");
```

Netprog 2002 CGI Programming

14

---

---

---

---

---

---

---

---

## Important!

- A CGI program doesn't have to send a status line (the http server will do this for you if you don't).
- A CGI program must **always** send back at least one header line indicating the data type of the content (usually `text/html`).
- The web server will typically throw in a few header lines of it's own (`Date`, `Server`, `Connection`).

Netprog 2002 CGI Programming

15

---

---

---

---

---

---

---

---

## Simple GET handler

```
int main() {
    char *method, *query;
    method = getenv("REQUEST_METHOD");
    if (method==NULL) ... /* error! */
    query = getenv("QUERY_STRING");
    printf("Content-type: text/html\r\n\r\n");
    printf("<H1>Your query was %s</H1>\n",
          query);
    return(0);
}
```

Netprog 2002 CGI Programming

16

---

---

---

---

---

---

---

---

## URL-encoding

- Browsers use an encoding when sending query strings that include special characters.
  - Most nonalphanumeric characters are encoded as a `'%'` followed by 2 ASCII encoded hex digits.
  - `' '` (which is hex 3D) becomes `"%3D"`
  - `'&'` becomes `"%26"`

Netprog 2002 CGI Programming

17

---

---

---

---

---

---

---

---

## More URL encoding

- The space character `' '` is replaced by `+'.'`
  - Why? (think about project 2 parsing...)
- The `+'.'` character is replaced by `"%2B"`

Example:

`"foo=6 + 7"` becomes `"foo%3D6+%2B+7"`

Netprog 2002 CGI Programming

18

---

---

---

---

---

---

---

---

## Security!!!

- It is a **very** bad idea to build a command line containing user input!
- What if the user submits: “ ; rm -r \*;”

```
grep ; rm -r *; /usr/dict/words
```

Netprog 2002 CGI Programming

19

---

---

---

---

---

---

---

---

## Beyond ISINDEX - Forms

- Many Web services require more than a simple ISINDEX.
- HTML includes support for forms:
  - lots of field types
  - user answers all kinds of annoying questions
  - entire contents of form must be stuck together and put in QUERY\_STRING by the Web server.

Netprog 2002 CGI Programming

20

---

---

---

---

---

---

---

---

## Form Fields

- Each field within a form has a name and a value.
- The browser creates a query that includes a sequence of “name=value” substrings and sticks them together separated by the ‘&’ character.

Netprog 2002 CGI Programming

21

---

---

---

---

---

---

---

---

## Form fields and encoding

- 2 fields - name and occupation.
- If user types in "Dave H." as the name and "none" for occupation, the query would look like this:

```
"name=Dave+H%2E&occupation=none"
```

Netprog 2002 CGI Programming

22

---

---

---

---

---

---

---

---

## HTML Forms

- Each form includes a METHOD that determines what http method is used to submit the request.
- Each form includes an ACTION that determines where the request is made.

Netprog 2002 CGI Programming

23

---

---

---

---

---

---

---

---

## An HTML Form

```
<FORM METHOD=GET  
  ACTION=http://foo.com/signup.cgi>  
Name:  
<INPUT TYPE=TEXT NAME=name><BR>  
Occupation:  
<INPUT TYPE=TEXT  
  NAME=occupation><BR>  
<INPUT TYPE=SUBMIT>  
</FORM>
```

Netprog 2002 CGI Programming

24

---

---

---

---

---

---

---

---

## What a CGI will get

- The query (from the environment variable `QUERY_STRING`) will be a URL-encoded string containing the name,value pairs of all form fields.
- The CGI must decode the query and separate the individual fields.

Netprog 2002 CGI Programming

25

---

---

---

---

---

---

---

---

## HTTP Method: POST

- The HTTP POST method delivers data from the browser as the content of the request.
- The GET method delivers data (query) as part of the URI.

Netprog 2002 CGI Programming

26

---

---

---

---

---

---

---

---

## GET vs. POST

- When using forms it's generally better to use POST:
  - there are limits on the maximum size of a GET query string (environment variable)
  - a post query string doesn't show up in the browser as part of the current URL.

Netprog 2002 CGI Programming

27

---

---

---

---

---

---

---

---

## HTML Form using POST

Set the form method to POST instead of GET.

```
<FORM METHOD=POST ACTION=...>
```

The browser will take care of the details...

Netprog 2002 CGI Programming

28

---

---

---

---

---

---

---

---

## CGI reading POST

- If REQUEST\_METHOD is a POST, the query is coming in STDIN.
- The environment variable CONTENT\_LENGTH tells us how much data to read.

Netprog 2002 CGI Programming

29

---

---

---

---

---

---

---

---

## Possible Problem

```
char buff[100];
char *clen =
    getenv("CONTENT_LENGTH");
if (clen==NULL)
    /* handle error */

int len = atoi(clen);

if (read(0,buff,len)<0)
    ... /* handle error */
pray_for(!hacker);
```

Netprog 2002 CGI Programming

30

---

---

---

---

---

---

---

---

## CGI Method summary

- GET:
  - `REQUEST_METHOD` is "GET"
  - `QUERY_STRING` is the query
- POST:
  - `REQUEST_METHOD` is "POST"
  - `CONTENT_LENGTH` is the size of the query (in bytes)
  - query can be read from `STDIN`

Netprog 2002 CGI Programming

31

---

---

---

---

---

---

---

---

## Form CGI Example

- Student enters first name, last name and social security number and presses a submit button.
- CGI program looks up grades for the student and returns a list of grades.

Complete example is on the course Web.

Netprog 2002 CGI Programming

32

---

---

---

---

---

---

---

---

## There's More to Come

- Keeping track of state information.
- Cookies.
- Using HTML templates
- Using JavaScript to perform form validation and other fancy stuff.
- Image Mapping
- Authentication
- Encryption

Netprog 2002 CGI Programming

33

---

---

---

---

---

---

---

---