HTTP Parsing

pallabi ghosh (PALLABIG@ANDREW.cmu.edu)

15-441 Computer Networks

Recitation 3

Feedback for Checkpoint 1

15-441: Computer Networks

Basic code structure

```
bind(), listen()
while true select(r_fds,w_fds)
        for fd in all current fds
                if fd is ready to accpet
                        accpet() → new_fd
                if fd is ready to read
                        recv() → buffer[fd]
                if fd is ready to write && buffer[fd]
                        buffer[fd] → send()
```

Timeout error: why is it slow?

```
bind(), listen(s, 5?)
while true select(r_fds,w_fds_with_data)
    for fd in all **current** fds
        if fd is ready to accpet
            accpet() → new_fd
        if fd is ready to read
            recv() → buffer[fd]
        if fd is ready to write && buffer[fd]
            buffer[fd] → send()
```

About Makefile

- At least have make and make clean working
- make: Nothing to be done for 'all'?
- all: lisod
- lisod:

@gcc echo_server.c -o lisod -Wall -Werror

- clean: rm -f lisod
  - Added .PHONY lisod at the beginning

About code style

- Use *meaningful* comment for git commit
- And in your source code of course
- No *magic number* in your code
- Do not version-control your *.o or lisod
- Do not print debug info
  Your submission should be camera-ready

PJ 2 CP 2

- GET, POST and HEAD requests
- Read the documents first to understand the rules.
- RFC 2616
- Check the annotated RFC

HTTP request

- Request =

```
Request-Line ;
*(( general-header ;
request-header ;
entity-header ) CRLF) ;
CRLF [ message-body ] ;
```

HTTP request

- Request =

```
GET /path/file.html HTTP/1.1
Host: www.host1.com:80
User-Agent: MyBrowser/1.0
[blank line here]
```

HTTP request

- Request =

```
POST /path/script.cgi HTTP/1.1
Host: www.host1.com:80
User-Agent: MyBrowser/1.0
Content-Type: application/x-www-form-urlencoded
Content-Length: 32

name=mukerjee&age=25
```

HTTP response

- Response =

```
Status-Line ;
*(( general-header ;
 response-header ;
entity-header ) CRLF) ;
 CRLF [ message-body ] ;
```

HTTP response

- Response =

```
HTTP/1.1 200 OK
Date: Fri, 20 Sep 2013 23:59:59 GMT
Content-Type: text/html
Content-Length: 1354

<html>
...
```

Tools

•Use tools to look at these requests and see the pattern for real

–Wireshark

◦http://www.wireshark.org/

–Use the dumper code (dumper.py)

–Play with the headers

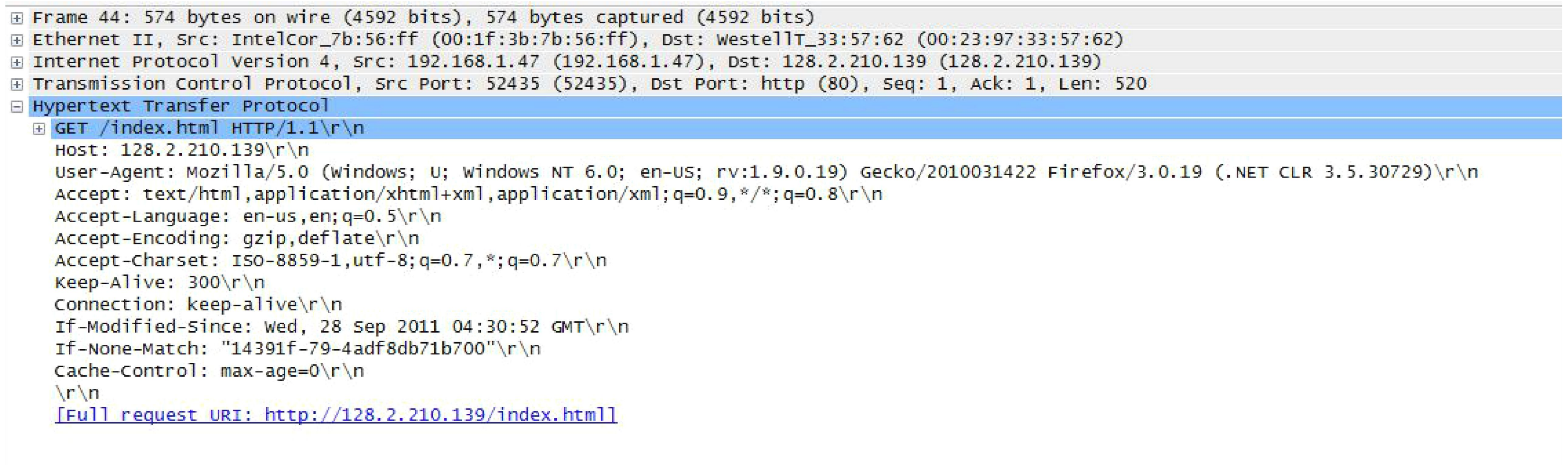
◦Tamperdata

◦Poster

–Online tools

◦http://web-sniffer.net/

Sample GET request

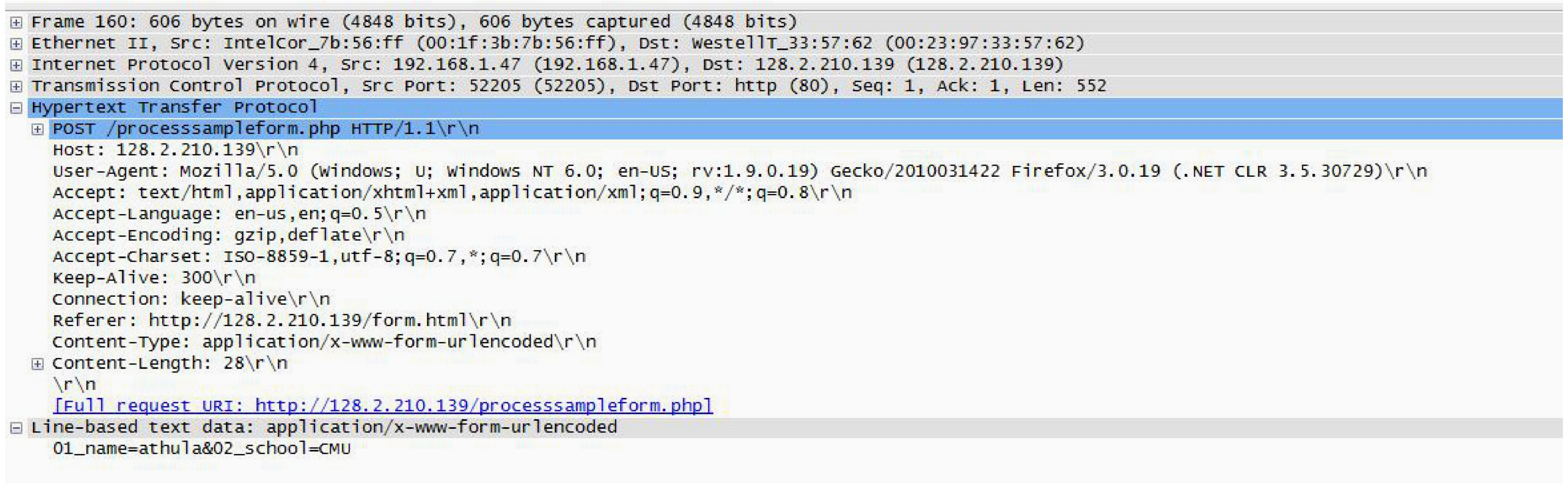


```
⊞ Frame 44: 574 bytes on wire (4592 bits), 574 bytes captured (4592 bits)
⊞ Ethernet II, Src: IntelCor_7b:56:ff (00:1f:3b:7b:56:ff), Dst: WestellT_33:57:62 (00:23:97:33:57:62)
⊞ Internet Protocol Version 4, Src: 192.168.1.47 (192.168.1.47), Dst: 128.2.210.139 (128.2.210.139)
⊞ Transmission Control Protocol, Src Port: 52435 (52435), Dst Port: http (80), Seq: 1, Ack: 1, Len: 520
⊟ Hypertext Transfer Protocol
  ⊞ GET /index.html HTTP/1.1\r\n
    Host: 128.2.210.139\r\n
    User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.0; en-US; rv:1.9.0.19) Gecko/2010031422 Firefox/3.0.19 (.NET CLR 3.5.30729)\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n
    Accept-Language: en-us,en;q=0.5\r\n
    Accept-Encoding: gzip,deflate\r\n
    Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7\r\n
    Keep-Alive: 300\r\n
    Connection: keep-alive\r\n
    If-Modified-Since: Wed, 28 Sep 2011 04:30:52 GMT\r\n
    If-None-Match: "14391f-79-4adf8db71b700"\r\n
    Cache-Control: max-age=0\r\n
    \r\n
    [Full request URI: http://128.2.210.139/index.html]
```

GET request was done on this page.

Do you see the pattern?

Sample POST request



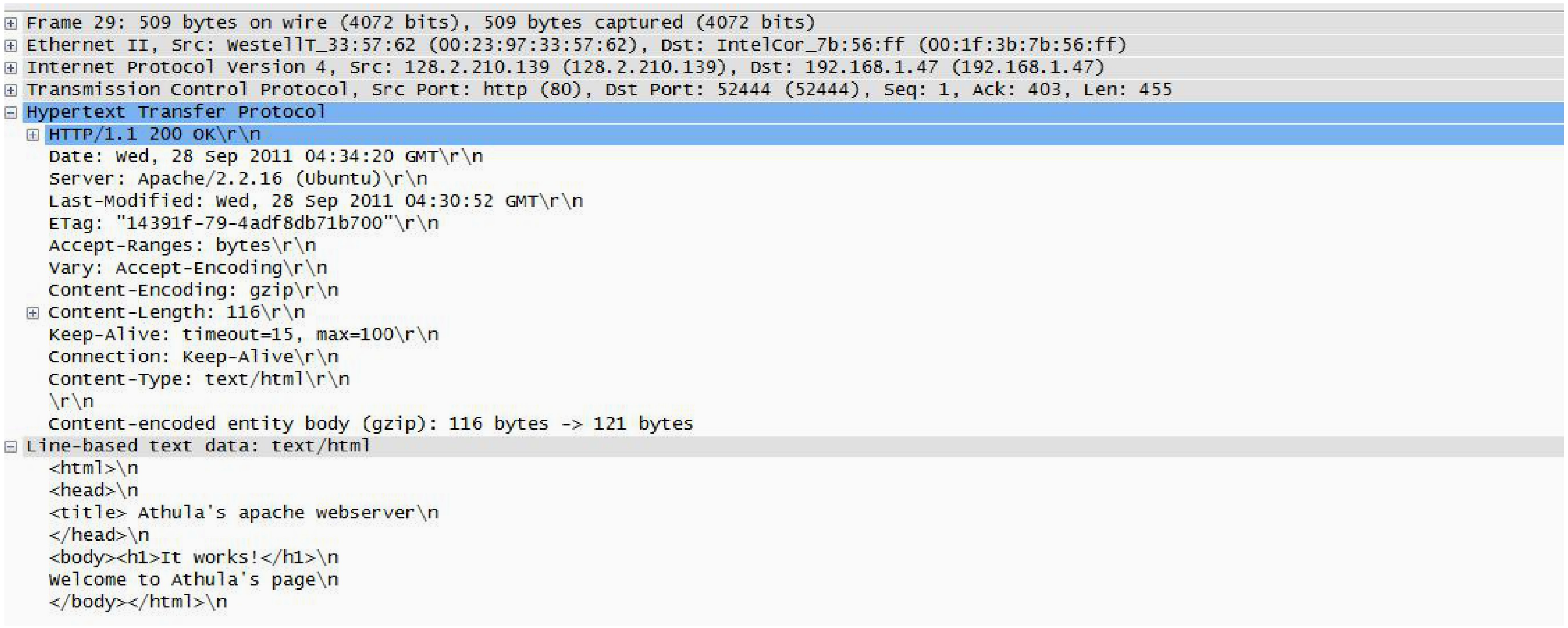
```
⊞ Frame 160: 606 bytes on wire (4848 bits), 606 bytes captured (4848 bits)
⊞ Ethernet II, Src: IntelCor_7b:56:ff (00:1f:3b:7b:56:ff), Dst: WestellT_33:57:62 (00:23:97:33:57:62)
⊞ Internet Protocol Version 4, Src: 192.168.1.47 (192.168.1.47), Dst: 128.2.210.139 (128.2.210.139)
⊞ Transmission Control Protocol, Src Port: 52205 (52205), Dst Port: http (80), Seq: 1, Ack: 1, Len: 552
⊟ Hypertext Transfer Protocol
  ⊞ POST /processsampleform.php HTTP/1.1\r\n
    Host: 128.2.210.139\r\n
    User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.0; en-US; rv:1.9.0.19) Gecko/2010031422 Firefox/3.0.19 (.NET CLR 3.5.30729)\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n
    Accept-Language: en-us,en;q=0.5\r\n
    Accept-Encoding: gzip,deflate\r\n
    Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7\r\n
    Keep-Alive: 300\r\n
    Connection: keep-alive\r\n
    Referer: http://128.2.210.139/form.html\r\n
    Content-Type: application/x-www-form-urlencoded\r\n
  ⊞ Content-Length: 28\r\n
    \r\n
    [Full request URI: http://128.2.210.139/processsampleform.php]
⊟ Line-based text data: application/x-www-form-urlencoded
    01_name=athula&02_school=CMU
```

POST method called while submitting this form.

Can you identify the values that were submitted?

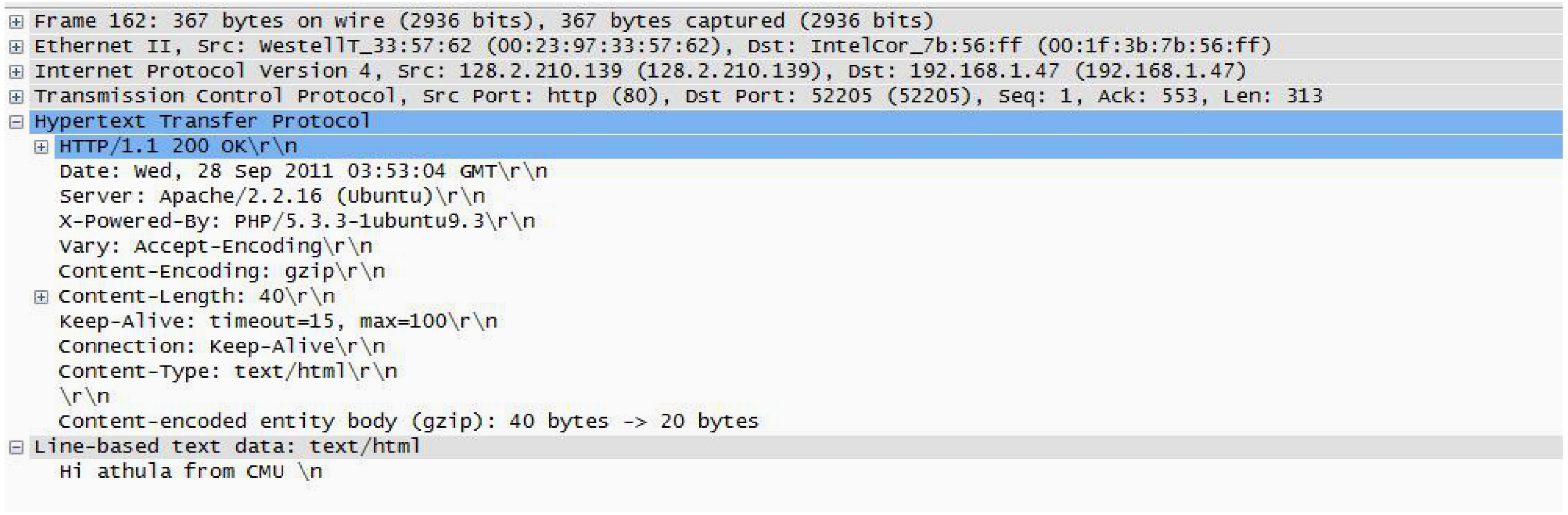
HEAD is similar to GET!
(just without the data)

Sample GET response



```
⊞ Frame 29: 509 bytes on wire (4072 bits), 509 bytes captured (4072 bits)
⊞ Ethernet II, Src: WestellT_33:57:62 (00:23:97:33:57:62), Dst: IntelCor_7b:56:ff (00:1f:3b:7b:56:ff)
⊞ Internet Protocol Version 4, Src: 128.2.210.139 (128.2.210.139), Dst: 192.168.1.47 (192.168.1.47)
⊞ Transmission Control Protocol, Src Port: http (80), Dst Port: 52444 (52444), Seq: 1, Ack: 403, Len: 455
⊟ Hypertext Transfer Protocol
  ⊞ HTTP/1.1 200 OK\r\n
    Date: Wed, 28 Sep 2011 04:34:20 GMT\r\n
    Server: Apache/2.2.16 (Ubuntu)\r\n
    Last-Modified: Wed, 28 Sep 2011 04:30:52 GMT\r\n
    ETag: "14391f-79-4adf8db71b700"\r\n
    Accept-Ranges: bytes\r\n
    Vary: Accept-Encoding\r\n
    Content-Encoding: gzip\r\n
  ⊞ Content-Length: 116\r\n
    Keep-Alive: timeout=15, max=100\r\n
    Connection: Keep-Alive\r\n
    Content-Type: text/html\r\n
    \r\n
    Content-encoded entity body (gzip): 116 bytes -> 121 bytes
⊟ Line-based text data: text/html
    <html>\n
    <head>\n
    <title> Athula's apache webserver\n
    </head>\n
    <body><h1>It works!</h1>\n
    Welcome to Athula's page\n
    </body></html>\n
```

Sample POST response



```
⊞ Frame 162: 367 bytes on wire (2936 bits), 367 bytes captured (2936 bits)
⊞ Ethernet II, Src: WestellT_33:57:62 (00:23:97:33:57:62), Dst: IntelCor_7b:56:ff (00:1f:3b:7b:56:ff)
⊞ Internet Protocol Version 4, Src: 128.2.210.139 (128.2.210.139), Dst: 192.168.1.47 (192.168.1.47)
⊞ Transmission Control Protocol, Src Port: http (80), Dst Port: 52205 (52205), Seq: 1, Ack: 553, Len: 313
⊟ Hypertext Transfer Protocol
  ⊞ HTTP/1.1 200 OK\r\n
    Date: Wed, 28 Sep 2011 03:53:04 GMT\r\n
    Server: Apache/2.2.16 (Ubuntu)\r\n
    X-Powered-By: PHP/5.3.3-1ubuntu9.3\r\n
    Vary: Accept-Encoding\r\n
    Content-Encoding: gzip\r\n
  ⊞ Content-Length: 40\r\n
    Keep-Alive: timeout=15, max=100\r\n
    Connection: Keep-Alive\r\n
    Content-Type: text/html\r\n
    \r\n
    Content-encoded entity body (gzip): 40 bytes -> 20 bytes
⊟ Line-based text data: text/html
    Hi athula from CMU \n
```

For this checkpoint you just have to send status code. More on dynamic content later!

Minimal Implementation

- Status codes
  - 200_OK
  - 404_NOT_FOUND
  - 411_LENGTH_REQUIRED
  - 500_INTERNAL_SERVER_ERROR
  - 501_NOT_IMPLEMENTED
  - 503_SERVICE_UNAVAILABLE
  - 505_HTTP_VERSION_NOT_SUPPORTED

Minimal Implementation

- General Headers
  - Connection
  - Date
- Response Headers
  - Server should always be: Liso/1.0
- Entity Headers
  - Content-Length
  - Content-Type
  - Last-Modified

Be careful with the buffers!

- Requests may straddle multiple recv calls
  - Need to maintain state information.
- If request header size > 8192 bytes
  - For now, send error message and disconnect.

Flex Tutorial

15-441: Computer Networks

What && Why

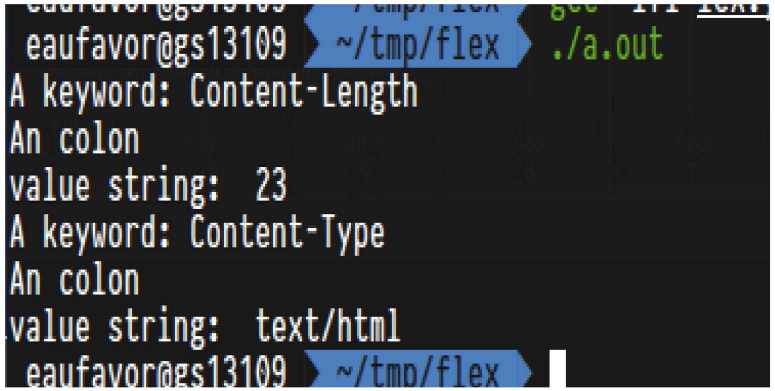
- Flex (fast lexical analyzer generator) is a lexer. it scans strings to identify keywords, numbers and tokens.
- your http parser need a lexer.
- You have two options for a lexer:
  A. hand write your lexer
  B. flex generates C code of your lexer

How

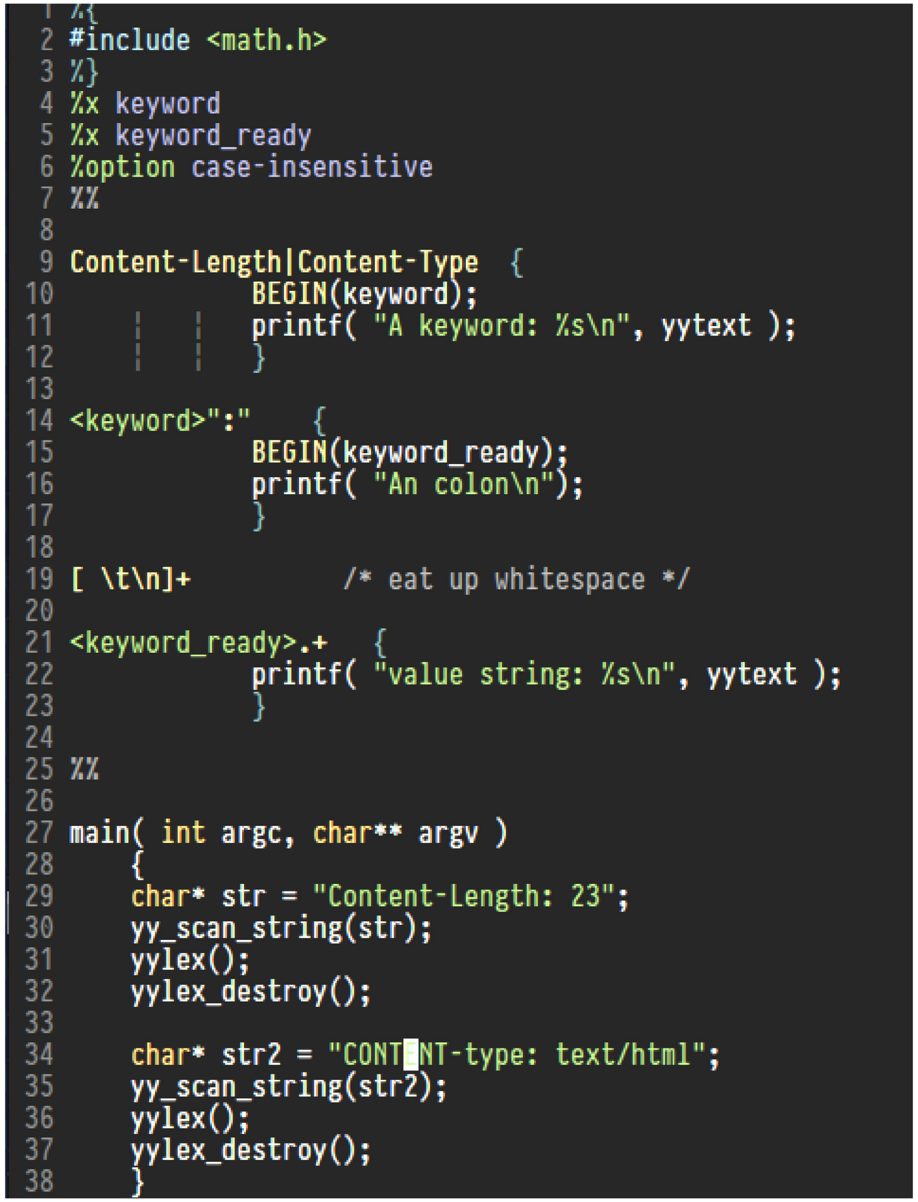
- Some knowledge of regular expression
- Write your lex file
- Compile it to a c file
- Compile your c file

```
flex -o foo.c foo.lex
gcc foo.c
```

Example



```
[illegible]
eaufavor@gs13109 ~/tmp/flex ./a.out
A keyword: Content-Length
An colon
value string:  23
A keyword: Content-Type
An colon
value string:  text/html
eaufavor@gs13109 ~/tmp/flex
```



```
%{
#include <math.h>
%}
%x keyword
%x keyword_ready
%option case-insensitive
%%

Content-Length|Content-Type  {
                BEGIN(keyword);
                printf( "A keyword: %s\n", yytext );
                }

<keyword>":"    {
                BEGIN(keyword_ready);
                printf( "An colon\n");
                }

[ \t\n]+          /* eat up whitespace */

<keyword_ready>.+   {
                printf( "value string: %s\n", yytext );
                }

%%

main( int argc, char** argv )
    {
    char* str = "Content-Length: 23";
    yy_scan_string(str);
    yylex();
    yylex_destroy();

    char* str2 = "CONTENT-type: text/html";
    yy_scan_string(str2);
    yylex();
    yylex_destroy();
    }
```

All questions?