

# Computer Network Project 2 **Simple Webserver**

CSI4106-01

Fall, 2016

# Prelim.





- You have total 4 projects in this class.
- Each project has “**Mandatory**” assignment and “**Additional**” assignment (optional).
  - Mandatory(100 pts) + Additional( $\alpha$  pts)
- You should follow the **print format of screen** and the **file format of your deliverables**.
- We provide the example version of deliverables so we can grade your projects automatically.

# A typical HTTP Communication

- When you make a request: [www.yonsei.ac.kr](http://www.yonsei.ac.kr)
  1. The client ask IP address to DNS servers
    - (e.g.) 168.126.63.1 (kns.kornet.net)
  2. Then you get 165.132.13.38 as the IP
  - 3. The client asks a webpage to 165.132.13.38:80**
  - 4. The server returns a HTML file as response**
  - 5. The client asks additional files for rendering the HTML file.**

# How CLIENT works (example)

- Chrome → Developer Tools (F12) → Network Tab
- It shows how Chrome fetches yonsei.ac.kr webpage

Name Path	Method	Status Text	Type	Initiator	Size Content	Time Latency	Timeline – Start Time
 yonsei.ac.kr	GET	302 Found	text/html	Other	378 B 0 B	14 ms 13 ms	
 index.jsp /sc	GET	200 OK	document	<a href="http://yonsei.ac.kr/">http://yonsei.ac.kr/</a> Redirect	67.2 KB 66.9 KB	86 ms 33 ms	



















1. Request to (yonsei.ac.kr)
2. Got HTTP/1.1 302 Found (Temporarily Moved)
3. Redirect to (yonsei.ac.kr/sc/index.jsp)
4. Fetch the HTML page
5. **Render the page with requests of CSS/JS/JPG...**

```

<title>연세대학교 홈페이지에 오신것을 환영합니다.</title>
<link rel="canonical" href="http://yonsei.ac.kr/sc/index.jsp">

<link rel="stylesheet" type="text/css" href="/_res/sc/css/default.css">
<link rel="stylesheet" type="text/css" href="/_res/sc/css/board.css">
<link rel="stylesheet" type="text/css" href="/_res/sc/css/swiper.min.css">
<link rel="stylesheet" type="text/css" href="/_res/sc/css/main.css">
<link rel="stylesheet" type="text/css" href="/_res/sc/css/user.css">
<script src="/_res/sc/js/user/jquery.min.js"></script>
<script src="/_res/sc/js/user/jquery-ui.min.js"></script>

```

Name Path	Method	Status Text	Type	Initiator	Size Content	Time Latency	Timeline – Start Time	
 yonsei.ac.kr	GET	302 Found	text/html	Other	378 B 0 B	14 ms 13 ms		
 index.jsp /sc	GET	200 OK	document	http://yonsei.ac.kr/ Redirect	67.2 KB 66.9 KB	86 ms 33 ms		
 default.css /_res/sc/_css	GET	200 OK	stylesheet	index.jsp:10 Parser	32.1 KB 32.0 KB	45 ms 37 ms		
 board.css /_res/sc/_css	GET	200 OK	stylesheet	index.jsp:11 Parser	34.0 KB 33.9 KB	46 ms 44 ms		
 swiper.min.css /_res/sc/_css	GET	200 OK	stylesheet	index.jsp:12 Parser	14.2 KB 14.1 KB	25 ms 24 ms		
 main.css /_res/sc/_css	GET	200 OK	stylesheet	index.jsp:13 Parser	15.8 KB 15.6 KB	30 ms 27 ms		
 user.css /_res/sc/_css	GET	200 OK	stylesheet	index.jsp:14 Parser	2.0 KB 1.9 KB	25 ms 24 ms		
 jquery.min.js /_res/sc/_js/user	GET	200 OK	script	index.jsp:15 Parser	93.8 KB 93.7 KB	86 ms 72 ms		
 jquery-ui.min.js /_res/sc/_js/user	GET	200 OK	script	index.jsp:16 Parser	234 KB 234 KB	184 ms 133 ms		

# How WEBSERVER works

1. Socket listen to 80 port (or a specific port)
  2. Read and parse a HTTP request
    - (e.g.) `/index.html` with Mobile User-agent
  3. Write a packet of HTTP response
    - Its body has data of `/index.html`
- You may know about...
    - `socket()`, `listen()`, `connect()`, `bind()`, `accept()`...
    - TCP Socket, Stream...
    - Socket Initialize, Open and Close...

# Mandatory Assignment

- Write the code of simple http webserver **to serve webpages including html/css/js/jpg/png files**
- Implement Required Functions (1)~(3)
- You may use TCP Socket Programming
- **Follow the usage format below**

(Format) `./run.sh port rootDir`

(Example) `./run.sh 8080 /var/www`

[http://my\\_ip\\_address:port/](http://my_ip_address:port/) → <http://10.0.0.1:8080/>

**\*\* It serves /var/www/index.html as default.**

We provide sample homepage files for test.

Your webserver should work with those files.

```
sh-3.2# python project_2.py 80 /var/www
Listening on port 80 ...
```

# Required Functions

## (1) Generating Response

- When you generate a HTTP response
- For the header, you should include
  - Content-Length
  - Content-Type (follow only 5 types below)

<b>MIME-type</b>	<b>description</b>	<b>extension</b>
<code>text/html</code>	HTML file	html
<code>text/css</code>	Cascading style sheet	css
<code>text/javascript</code>	Javascript source file	js
<code>image/jpeg</code>	JPEG image file	jpg
<code>image/png</code>	png image file	png



# Required Functions

## (2) Path Translation

- <http://IP:port/> (e.g. http://165.132.0.1:8080)
- Set physical root directory: /var/www
- Set the default index page: index.html
  - GET / → GET /index.html
- Basic path processing
  - GET /css/abc.css → It actually serves /var/www/css/abc.css

# Required Functions

## (3) Error Exception

- Follow the format for 403 and 404(Mandatory)
- (e.g.) GET /nopath/nofile

```
<html><head><title>Access Forbidden</title></head><body>  
<h1>403 Forbidden</h1>  
<p>You don't have permission to access the requested URL  
/nopath/nofile. There is either no index document or the  
directory is read-protected.</p>  
</body></html>
```

```
<html><head><title>404 Not Found</title></head><body>  
<h1>404 Not Found</h1>  
<p>The requested URL /nopath/nofile was not found on this  
server.</p>  
</body></html>
```

# Implementation Scope

- **The goal is not to write a complete and perfect webserver.**
- STATIC files only (no CGI/PHP script)
- HTTP 1.1 only (not HTTPS)
- Do not care Consistent Connection
  - You just need to use Content-length only
- Do not care DNS / Domain-relevant issues
- Do not care Performance issues

# Additional Assignment (1) **+5pts**

- **Goal: Make a simple redirection**
- /go/facebook → <http://www.facebook.com/>
- /go/google → <http://www.google.com/>
- /go/%name% → http://www.%name%.com/
- Regular expression of %name%
  - ([a-zA-Z]+)
- Hint: **HTTP/1.1 302 Moved**

# Additional Assignment (2) **+15pts**

- **Goal: Request Parsing of User-agent**
- For desktop users
  - Your homepage is root folder
- For mobile users
  - Your homepage is “mobile” folder.
- **(Warning) This is not just a Redirection!!**
- When a mobile user accesses your server
  - **(X) GET / → GET /mobile/index.html**
  - **(O) GET / → GET /index.html**
  - **Fetch files from mobile folder**

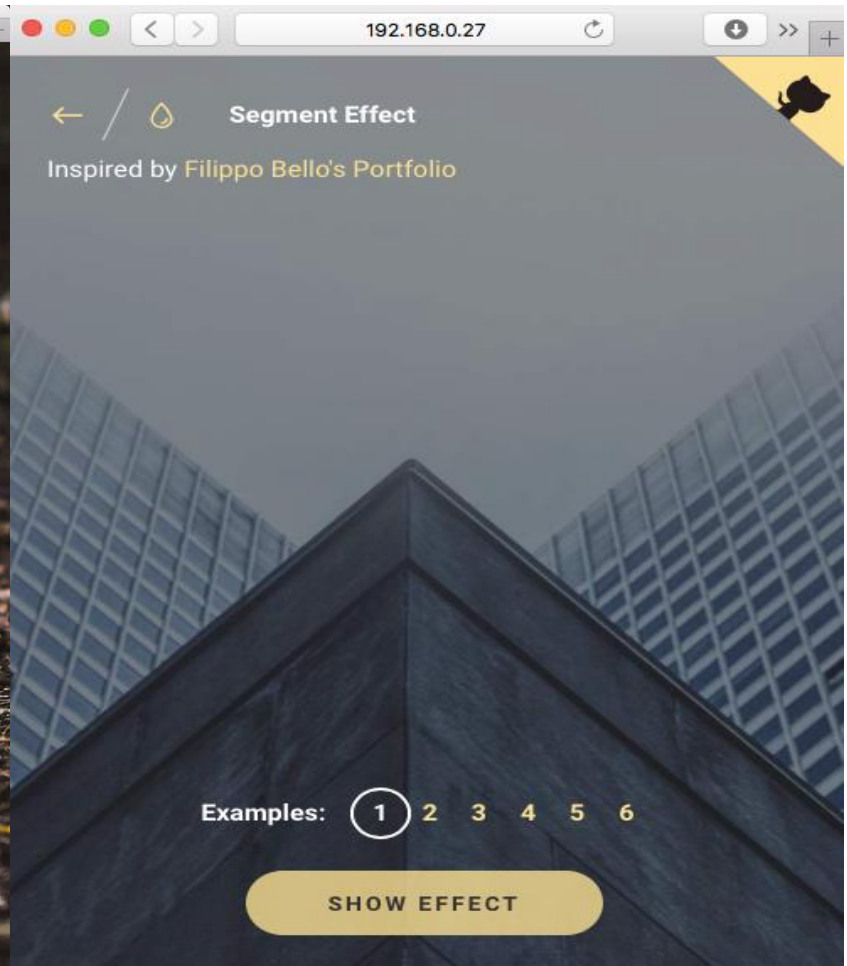
# Additional Assignment (2)

- User-agent → To decide it is mobile or not
- Desktop: Windows / Chrome
  - Mozilla/5.0 (**Windows** NT 6.1; Win64; x64)  
AppleWebKit/537.36 (KHTML, like Gecko)  
Chrome/53.0.2785.143 Safari/537.36
- Nexus 4: Android / Chrome
  - Mozilla/5.0 (Linux; **Android** 4.4.2; Nexus 4 Build/KOT49H)  
AppleWebKit/537.36 (KHTML, like Gecko)  
Chrome/34.0.1847.114 **Mobile** Safari/537.36
- iPhone 6: iOS / Safari
  - Mozilla/5.0 (**iPhone**; CPU iPhone OS 6\_0 like Mac OS X)  
AppleWebKit/536.26 (KHTML, like Gecko) Version/6.0  
**Mobile**/10A5376e Safari/8536.25

# Additional Assignment (2)



**Desktop User**



**Mobile User**

# Additional Assignment (3) **+30pts**

- **Only authenticated clients should be able to access to files in /secret**
- Without the cookie → 403 Forbidden Error Page
- With the cookie → 200 OK with **/secret/**
- Transaction: make a POST request with **?id=yonsei&pw=network**
  - Logged → 200 OK with **/secret/**
  - The server generates a custom Cookie of **Your ID**
    - Hint: **Set-Cookie**
  - The client will send a request with the cookie when it accesses to “/secret” after logged.



# Hint: Cookie for webserver

- [https://en.wikipedia.org/wiki/HTTP\\_cookie#Implementation](https://en.wikipedia.org/wiki/HTTP_cookie#Implementation)
- You can test with Postman (Chrome Extension)



# Background

- TCP (socket programming)
- HTTP Packet (request and response)
- Chrome – Developer Tools (F12 key)
- Basic idea of HTML/CSS/JS
- Regular Expression
- MIME types (Content-types)
- HTTP 1.1 (for further learning)
- User-agent switcher for chrome extension

Deliverables *(without folder)*

**[c|u]\_YourID\_2.zip**

**c=CentOS, u=Ubuntu**

*e.g. c\_2014123456\_2.zip*

- **readme.txt** *(follow the example format)*
- **project\_2.[py|c]**
  - Your code with **detail comments block by block**
- **run.sh**
  - This makes your code run
- **setup.sh**
  - This should install dependencies or compile your code
- **report.pdf** *(follow the example format)*

# Report Format (*report.pdf*)

- Introduction
- Specify what you have implemented
- Code commentary block by block
- How my code works
- Screenshots with your explanation
- What you have learned by this project
- Further research

# Directions

- **This is an individual project**
- You should follow the file/output format
- We provide you sample format and HTML files
- Language: **C or Python**
  - **C: gcc 4.8.5**
  - **Python: Python 2 ( $\geq 2.7.5$ ) or Python 3 ( $\geq 3.5.2$ )**
- OS
  - **CentOS 7 ( $\geq 7.2$ )**
  - **Ubuntu 14.04.4 LTS (only this version)**

# Directions

- You must use only *internal* libraries.
- **Any 3<sup>rd</sup> party framework: NOT ALLOWED**
- **Especially for python, below are not allowed**
  - SimpleHTTPServer
  - BaseHTTPServer
  - SocketServer

# How to use **website.zip**

- You may extract files onto server's root folder for your test
- **website.zip** includes html, css, js, image files
  - / → root folder
  - /mobile → for Additional Assignment 2
  - /secret → for Additional Assignment 3
- You can test with
  - /mobile, /secret, /index.html and so on...
  - /img/\*.jpg and /css/\*.css

# Due Date / Delay Policy

- **DUE DATE (2weeks)**

**7/Nov/2016 23:55:00 KST**

- **Delay Policy**

**-15pts per day**



Remember...

- **DO NOT COPY CODE**

- We run Code-plagiarism Program.

- The fastest way to get 0 points.

- **YOU WILL GET 0 POINTS if you CHEAT**

# Score Policy (*\*tentative*)

*Maximum Score = 100+50 pts*

We will announce the finalized version later on YSCEC

1	Not submitted or not working	0 pts
2	Overdue → Delay	-15pts/day
3	Additional Assignment is implemented	+5/15/30pts
4	Bad File/Display Formatting	-10 pts each
5	A 3 <sup>rd</sup> party framework is used	0 pts
6	Over-implementation (Suspicion of Code copy)	0 pts
7	Impolite or poor report (Lack of code commentaries)	-80 ~ -20 pts

- Please use YSCEC Q&A board to leave your question.
- Before you ask a question
  - Check project introduction PDF AGAIN.
  - Check others' Q&A.
- Duplicate questions are ignored.