

## 제14강

# 웹서비스통한 원격제어

Apache 웹 서버 구축 (CGI를 위한 설정)

HTML FORM 태그

CGI 프로그래밍(GET 및 POST 전달방식)

LED / Magnetic Switch 디바이스 제어(GET 및 POST 전달방식)

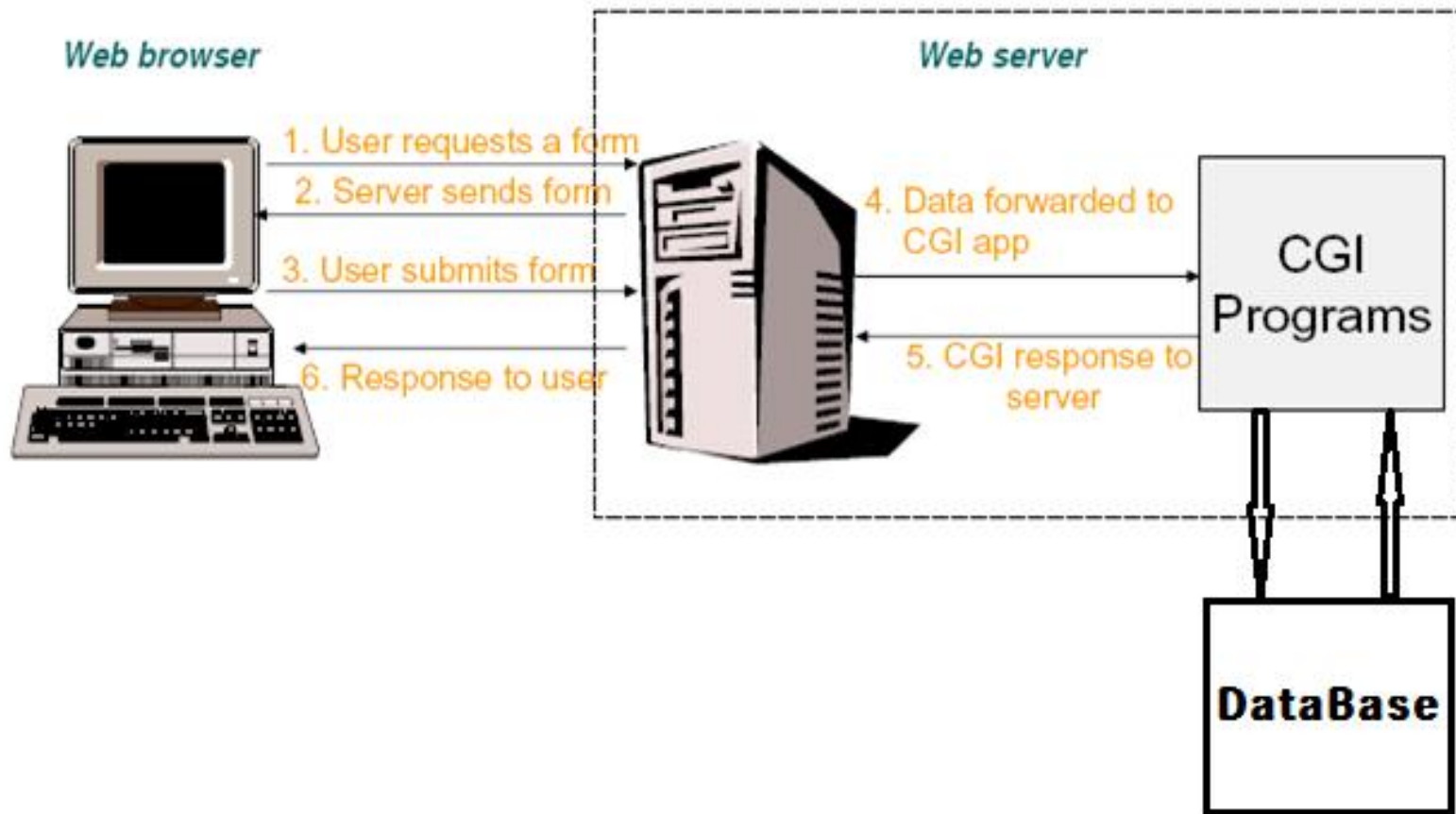
참고) 교재 제15장

## **웹 서비스**

- \* HTTP(hypertext transfer protocol) 기반 서비스**
  - : 태그기반의 HTML을 이용한 문서 등의 서비스**
  
- \* 정적 웹 서비스**
  - : 사용자는 브라우저를 통해 문서를 요청하고,**
  - 웹서버는 준비된 문서를 송출하는 단방향 정보 교환 형태**
  - : 서버측으로 정보전달 및 서버의 동적 응답을 기대할 수 없음**
  - 예) 현재 강의자료 웹서비스 형태**
  
- \* 동적 웹 서비스**
  - : 웹서버와 정보를 처리하는 프로그램 간의 연동 표준을**
  - CGI이며, 해당 프로그램을 CGI 프로그램이라 함**
  - : 정보검색, 쇼핑몰 주문, 게시판 등 DB와 연동이 일반적**

## 웹 서비스(계속)

### \* 동적 웹 서비스의 동작 절차



# Apache 웹서버

## \* Apache 웹 서버 설치

: 타깃보드에 아파치 웹서버 설치

```
$ sudo apt-get update
```

```
$ sudo apt-get install apache2
```

// /etc/apache2/에 설치 됨

```
$ sudo reboot // 재부팅
```

: 아파치 웹서버 실행여부 확인

```
$ ps -ef | grep apache
```

```
root      605      1  0 11:55 ?                00:00:00 /usr/sbin/apache2 -k start
```

.....

: 설치후 내장된 웹 서버의 홈 디렉터리는 **/var/www/html/**

```
$ ls /var/www/html
```

```
index.html ....
```

// 아파치서버의 홈 페이지 파일명

# Apache 웹서버(계속)

## \* 웹 서버 접속

: 웹 브라우저의 주소창에 타깃보드의 IP 주소를 입력

: <http://192.168.0.30>



## Apache 웹서버(계속)

### \* 참고사항

- : ./sites-enabled/000-default.conf 파일참조
- : 웹서버관련 로그정보는 /var/log/apache2/ 디렉터리에
- : 접속 로그나 오류 로그 정보 확인시 다음 파일 참고

\$ tail /var/log/apache2/access.log // 접속 로그 파일

\$ tail /var/log/apache2/error.log // 오류 로그 파일

### \* ) 장기간 구동상태에서 웹서버 동작 불능의 경우

- 로그정보로 인한 메모리 공간 부족 문제가 원인
- 로그파일을 삭제하여 공간 확보
- /var/log/apache2 디렉터리는 삭제 금함

## Apache 웹서버(계속)

### \* CGI(common gateway interface)

- : 웹서버와 외부 프로그램 간의 데이터 전달을 위한 표준방식
- : 동적 서비스 환경에서 필수
- : 시스템에서 가용한 어떠한 프로그래밍 언어로도 가능
  - 컴파일러형 : C 등
  - 인터프리터형(스크립트) : python, perl, php 등

# Apache 웹서버(계속)

## \* Apache2 디렉터리의 구조

pi@raspberrypi:/etc/apache2 \$ tree ./

```

./
├── apache2.conf                // 웹 서버의 주 환경설정 파일
├── conf-available              // 미리 준비된 환경설정 파일들
│   ├── charset.conf
│   ├── javascript-common.conf
│   ├── .....
│   └── serve-cgi-bin.conf
├── conf-enabled                // 실제 반영된 환경설정 파일들에 대한 심볼릭링크 파일들
│   ├── charset.conf -> ../conf-available/charset.conf
│   ├── .....
│   └── serve-cgi-bin.conf -> ../conf-available/serve-cgi-bin.conf
├── envvars                    // 환경변수, LANG=C 등
├── magic
├── mods-available              // 준비된 모듈들의 파일들
│   ├── access_compat.load
│   ├── .....
│   ├── cgi.load                // CGI를 위해 준비된 모듈 파일 !!!!!
│   ├── .....
│   └── xml2enc.load
├── mods-enabled                // 실제 반영된 모듈들의 심볼릭링크 파일들
│   └── access_compat.load -> ../mods-available/access_compat.load
├── ports.conf                 // 포트 설정, Listen 80
├── sites-enabled               // 실제 반영된 사이트관련 파일에 대한 심볼릭링크 파일
│   └── 000-default.conf -> ../sites-available/000-default.conf
│       // 가상호스트명 및 포트, 서비스 홈디렉터리, 로그파일 위치 지정

```



## Apache 웹서버(계속)

### \* 디렉터리 명 범례

: 앞의 디렉터리 구조에서

디렉터리명이

...-available/ : 미리 준비된 환경설정파일이나 모듈들의 위치

...-enabled/ : 서버 실행될때의 환경설정파일이나 모듈들의 링크파일 위치

## Apache 웹서버(계속)

### \* CGI 홈 디렉터리 변경

: ./conf-available/serve-cgi-bin.conf 파일 편집

: CGI 홈 디렉터를 /var/www/html/cgi-bin/로 변경

\$ cd /etc/apache2/conf-available/

\$ sudo nano serve-cgi-bin.conf

```
.....
<IfDefine ENABLE_USR_LIB_CGI_BIN>
    ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
    <Directory "/usr/lib/cgi-bin">                                // 디폴트 CGI 홈 디렉터리
        AllowOverride None
        Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
        Require all granted
.....
```

==> 밑줄 부분을 다음과 같이 CGI 프로그램을 위치할 경로로 변경

```
ScriptAlias /cgi-bin/ /var/www/html/cgi-bin/
<Directory "/var/www/html/cgi-bin">
```

: CGI 홈 디렉터리 생성

\$ sudo mkdir /var/www/html/cgi-bin/ // CGI 홈디렉터리 생성

## Apache 웹서버(계속)

### \* CGI 프로그램 실행 활성화

: cgi 관련 **cgi.load** 모듈을 활성화 ( **a2enmod** 명령 사용 )

```
$ sudo a2enmod cgi
```

```
// ./mods-available/cgi.load 모듈의 심볼릭 링크 파일을  
./mods-enabled/에 위치시키는 명령
```

### \* Apache 서버의 재시작 혹은 재부팅

```
$ sudo systemctl restart apache2
```

```
혹은, $ sudo service apache2 restart
```

```
혹은, $ sudo reboot
```

## Apache 웹서버(계속)

### [실습1] CGI 테스트 ( [./hellotest/](#) )

: C로 구현된 CGI 프로그램 실행 테스트

: Hello, .... 메시지를 출력하는 CGI 소스

```
$ nano hello.c
```

```
//=====
// hello.c
//=====
#include <stdio.h>

int main(void) {
    printf("Content-type: text/html\n\n");
    printf("<H1>Hello, World.....\n\n");
    return 0;
}
```

**\* 출력문들의 내용은 응답으로 생성할 HTML 문서 형식을 따름**

```
$ cat helloTest.html
```

```
<h1>CGI test ..... GET method ...</h1>  
<hr><p>  
<FORM method=GET action="./cgi-bin/hello.cgi">  
<INPUT type="text" name="value" maxlength="10" size="10">  
<INPUT type="submit" name="button" value="submit">  
</FORM>
```

**\* FORM 태그에서**

**method : CGI 전달 방식 설정 (GET, POST, ...)**

**action : 실행할 CGI 프로그램의 경로 및 파일명 설정**

: CGI 소스를 컴파일하여 cgi 프로그램 생성 ( **.cgi** 관례 )

```
$ gcc -o hello.cgi hello.c
```

: 관련 파일들을 서비스위해 웹서버의 홈 디렉터리 쪽에 복사

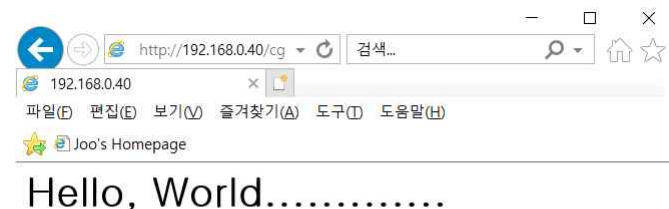
```
$ cp helloTest.html /var/www/html/
```

```
$ cp hello.cgi /var/www/html/cgi-bin/
```

: 웹브라우저 주소창에서 접속

– <http://192.168.0.30/helloTest.html>

– submit 버튼 클릭하면, CGI 실행결과 반환(우측)을 관찰 가능



## Apache 웹서버(계속)

### \* HTML FORM 태그

: HTML의 FORM 태그를 활용한 최소한의 구성

```
<FORM method=GET action="./cgi-bin/formtag.cgi">
```

.....

```
<input type=submit name=transmit value=transmit>  
</FORM>
```

- method 속성은 데이터 전송 방식을 규정 ( GET, POST 등 )
- action 속성에는 넘겨받은 데이터를 처리할 CGI 프로그램을 설정
- 최소한 하나의 input 태그를(type=submit인) 내부에 두도록 구성

## HTML FORM 태그(계속)

### \* input 태그의 type 유형

텍스트 <input type=text name=name maxlength=10>

암호 <input type=password name=pw maxlength=10>

숫자 <input type=number name=num maxlength=2>

라디오버튼 .. 동일한 name 속성을 갖는 항목들 중에서 하나의 항목만 선택

.. 기본 선택항목은 checked 속성을 사용

<input type=radio name=led value=1 checked> ON

<input type=radio name=led value=0> OFF

체크박스 .. 동일한 name 속성을 갖는 항목들 중에서 여러 항목 선택가능

.. 배열의 형태로 전달

<input type=checkbox name=ledArray value=8 checked> LED3

<input type=checkbox name=ledArray value=4> LED2

<input type=checkbox name=ledArray value=2> LED1

<input type=checkbox name=ledArray value=0 checked> LED0



## HTML FORM 태그(계속)

### \* input 태그의 type 유형(계속)

리스트 .. select 태그 내에서 option 태그로 항목을 나열  
.. 디폴트 선택항목의 표시는 selected 속성으로

```
<select name=year>  
  <option value=1> 2011 </option>  
  <option value=2> 2012 </option>  
  <option value=3 selected> 2013 </option>  
  <option value=4> 2014 </option>  
</select>
```

버튼 .. type 속성의 값이 button이면 일반 버튼을,  
.. submit이면 FORM 태그내의 내용을 전송하는 버튼,  
.. reset이면 FORM 태그 내에서 설정된 내용을 초기화

```
<input type=button name=ok value=ok>  
<input type=submit name=transmit value=transmit>  
<input type=reset name=cancel value=cancel>
```

텍스트영역 .. 여러 라인의 텍스트

```
<textarea name=memo rows=10 cols=40> memo... </textarea>
```

## HTML FORM 태그(계속)

[실습2] FORM 태그 테스트 ( [./hellotest/](#) )

: 앞에서 살펴본 input 태그의 유형들을 시험 ( CGI 없음 )

```
$ nano formTag.html
```

```
<FORM method=GET action="./cgi-bin/formtag.cgi">
```

```
<label>text</label>
```

```
<input type=text name=name maxlength=10>
```

```
<p>
```

```
<label>password</label>
```

```
<input type=password name=pw maxlength=10>
```

```
<p>
```

```
<label>number</label>
```

```
<input type=number name=num maxlength=2>
```

```
<p>
```

```
<label>radio</label>
```

```
<input type=radio name=led value=1 checked> ON
```

```
<input type=radio name=led value=0> OFF
```

<p>

<label>checkbox</label>

<input type=checkbox name=ledArray value=8 checked> LED3

<input type=checkbox name=ledArray value=4> LED2

<input type=checkbox name=ledArray value=2> LED1

<input type=checkbox name=ledArray value=0 checked> LED0

<p>

<label>list</label>

<select name=year>

<option value=1> 2011 </option>

<option value=2> 2012 </option>

<option value=3 selected> 2013 </option>

<option value=4> 2014 </option>

<option value=5> 2015 </option>

</select>

<p>

<label>textarea</label>

<textarea name=memo rows=10 cols=40> memo... </textarea>

<p>

<label>button</label>

<input type=button name=ok value=ok>

<p>

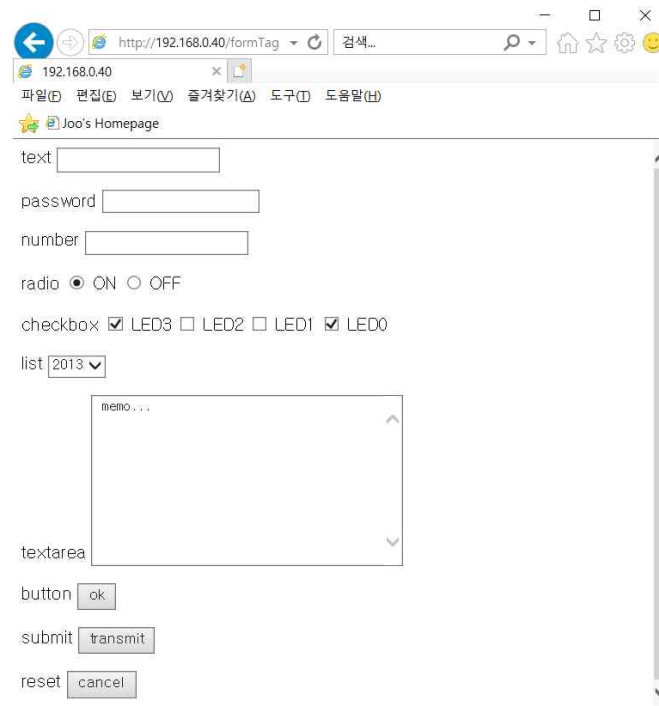
<label>submit</label>

```
<input type=submit name=transmit value=transmit>  
<p>
```

```
<label>reset</label>  
<input type=reset name=cancel value=cancel>  
<p>
```

```
</FORM>
```

: 웹 브라우저의 주소창에서 <http://192.168.0.30/formTag.html> 로 접속



The screenshot shows a web browser window with the address bar displaying <http://192.168.0.40/formTag>. The page content includes a form with the following elements:

- text:
- password:
- number:
- radio: ☒ ON ☐ OFF
- checkbox: ☒ LED3 ☐ LED2 ☐ LED1 ☒ LED0
- list:
- textarea: 

memo...
- button:
- submit:
- reset:



# CGI 프로그래밍

## \* CGI 프로그램에 데이터를 전달하는 방식

### 1) GET 전달 방식

- : 전송 가능한 정보의 길이가 한정
- : 전송 정보는 웹 브라우저의 주소창에 URL과 함께 인자형태로 전송됨  
(보안에 취약, 인자는 이름=값의 쌍으로, 각 인자간 &로 구분)

### 2) POST 전달 방식

- : 전송 가능한 정보의 길이에 제한 없음
- : 전송 정보는 웹 브라우저의 주소창에 비표시로 보안상 우위  
(전송정보는 헤더 내에 담겨 전송)

## CGI 프로그래밍(계속)

### \* CGI 관련 환경변수 및 함수

: 환경 변수

환경변수	기능	비고
REMOTE_ADDR	접속자의 IP 주소	
REQUEST_METHOD	GET, POST 등 요청 방법	
QUERY_STRING	FORM 태그 요청 문자열	
CONTENT_LENGTH	FORM 태그 요청 문자열의 길이	
HTTP_USER_AGENT	접속자의 웹브라우저 정보	
SERVER_SOFTWARE	웹서버 정보	

: 환경변수 값을 읽는 함수 ... **getenv()**

```
printf("Your IP Address : %s<br>\n", getenv("REMOTE_ADDR"));  
printf("QUERY_STRING : %s<br>\n", getenv("QUERY_STRING"));
```

참고) GET 방식에서 QUERY\_STRING 읽을 때 ... **getenv()**

참고) POST 방식에서 QUERY\_STRING 읽을 때 ... **read()**

## CGI 프로그래밍(계속)

### \* GET 전달 방식

: 주소창에 '인자명=값'으로 CGI 프로그램에 전달

CGI\_PROGRAM&para1=40H&para2=abc&....

: 전송문자 최대 256문자로 제한, 보안 취약

– html file에서

```
<FORM method=GET action="cgi-bin/cgi_get.cgi">
```

.....

```
</FORM>
```

– cgi file에서

.....

```
printf("Query string : %s<br>\n", getenv("QUERY_STRING"));
```



## CGI 프로그래밍(계속)

### \* POST 전달 방식

: 전송문자 무제한, 암호화전달, 보안에 바람직( 헤더로 )

– html file에서

```
<FORM method=POST action="cgi-bin/cgi_post.cgi">
```

.....

```
</FORM>
```

– cgi file에서

```
char buf[1024];
```

.....

```
while((n = read(0, buf, 1024)) > 0) {  
    printf("Query string : %s<br>\n", buf);
```

.....

## CGI 프로그래밍(계속)

[실습3] CGI 테스트 ( GET 방식 ) ( [./testGet 참조](#) )

: 몇몇 INPUT 태그에 대한 설정 정보 전달

```
$ nano testGet.html
```

```
<h1>CGI Test, by GET method</h1>  
<hr><p>
```

```
<FORM method=GET action="./cgi-bin/testGet.cgi">  
<label>text</label>  
<input type=text name=name maxlength=10>  
<p>
```

```
<label>number</label>  
<input type=number name=num maxlength=2>  
<p>
```

```
<label>radio</label>  
<input type=radio name=led value=1 checked> ON  
<input type=radio name=led value=0> OFF  
<p>
```

```
<label>checkbox</label>
<input type=checkbox name=ledArray value=8 checked> LED3
<input type=checkbox name=ledArray value=4> LED2
<input type=checkbox name=ledArray value=2> LED1
<input type=checkbox name=ledArray value=0 checked> LED0
<p>

<input type=submit name=transmit value=transmit>
<p>
</FORM>
```

\$ nano testGet.c

```
//=====
// testGet.c
// CGI Test, by GET method
//=====
#include <stdio.h>
#include <stdlib.h>

int main(void) {
    printf("Content-type: text/html\n\n");

    printf("<html>\n\n");

    printf("<head>\n");
```

```
printf("<title>CGI Test, by GET</title>\n");
printf("</head>\n\n");

printf("<body>\n");
printf("<h1>CGI Test, by GET</h1>\n");
printf("<hr><p>\n");

printf("Your IP Address : %s<br>\n", getenv("REMOTE_ADDR"));
printf("QUERY_STRING : %s<br>\n", getenv("QUERY_STRING"));

printf("</body>\n\n");
printf("</html>\n");

return 0;
}
```

## \* 컴파일 및 복사

: 컴파일하여 cgi 프로그램 생성

```
$ gcc -o testGet.cgi testGet.c
```

: 생성된 파일들을 서비스를 위해 웹서버의 홈 디렉터리 쪽에 복사

```
$ cp testGet.html /var/www/html/
```

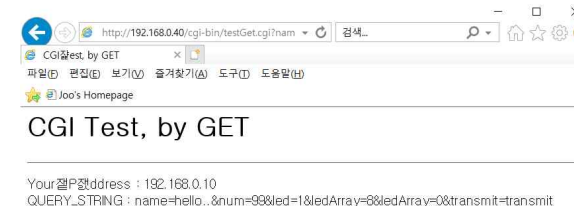
```
$ cp testGet.cgi /var/www/html/cgi-bin/
```

## \* 접속 및 결과( **깨짐 : 보기-인코딩-유니코드(UTF-8) 선택** )

: 웹브라우저의 주소창에서 <http://192.168.0.30/testGet.html> 로 접속

: 각 항목을 적절히 입력 및 선택하고 transmit 버튼을 클릭 (좌측)

: CGI 프로그램의 실행 결과 확인 (우측)



## CGI 프로그래밍(계속)

[실습4] CGI 테스트 ( POST방식 ) ( [./testPost 참조](#) )

: 몇몇 INPUT 태그에 대한 설정 정보 전달

```
$ nano testPost.html
```

```
<h1>CGI Test, by POST method</h1>  
<hr><p>
```

```
<FORM method=POST action="./cgi-bin/testPost.cgi">  
<label>text</label>  
<input type=text name=name maxlength=10>  
<p>
```

```
<label>number</label>  
<input type=number name=num maxlength=2>  
<p>
```

```
<label>radio</label>  
<input type=radio name=led value=1 checked> ON  
<input type=radio name=led value=0> OFF  
<p>
```

```
<label>checkbox</label>
<input type=checkbox name=ledArray value=8 checked> LED3
<input type=checkbox name=ledArray value=4> LED2
<input type=checkbox name=ledArray value=2> LED1
<input type=checkbox name=ledArray value=0 checked> LED0
<p>

<input type=submit name=transmit value=transmit>
<p>
</FORM>
```

```
$ nano testPost.c
```

```
//=====
// testPost.c
// CGI Test, by POST method
//=====
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main(void) {
    char buf[1024];

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

```
printf("<html>\n\n");
printf("<head>\n");
printf("<title>CGI Test, by POST</title>\n");
printf("</head>\n\n");

printf("<body>\n");
printf("<h1>CGI Test, by POST</h1>\n");
printf("<hr><p>\n");

printf("Your IP Address : %s<br>\n", getenv("REMOTE_ADDR"));

memset(buf, 0x00, 1024);
while(read(0, buf, 1024) > 0) {
    printf("POST STRING : %s<p>\n", buf);
}

printf("</body>\n\n");
printf("</html>\n");

return 0;
}
```



: 컴파일후 웹서버의 홈 디렉터리 쪽으로 복사

```
$ gcc -o testPost.cgi testPost.c
```

```
$ cp testPost.html /var/www/html/
```

```
$ cp testPost.cgi /var/www/html/cgi-bin/
```

: 접속 및 결과

: <http://192.168.0.30/testPost.html> 로 접속

: 각 항목을 적절히 입력 및 선택하고 transmit 버튼을 클릭 (좌측)

: CGI 프로그램의 실행 결과 확인 (우측)



The screenshot shows a web browser window with the address bar displaying 'http://192.168.0.40/testPost.html'. The page title is 'CGI Test, by POST method'. The form contains the following elements:

- A text input field with the value 'hello.....'.
- A number input field with the value '89'.
- A radio button group with 'ON' selected and 'OFF' unselected.
- A checkbox group with 'LED3' and 'LED0' checked, and 'LED2' and 'LED1' unchecked.
- A 'transmit' button at the bottom.



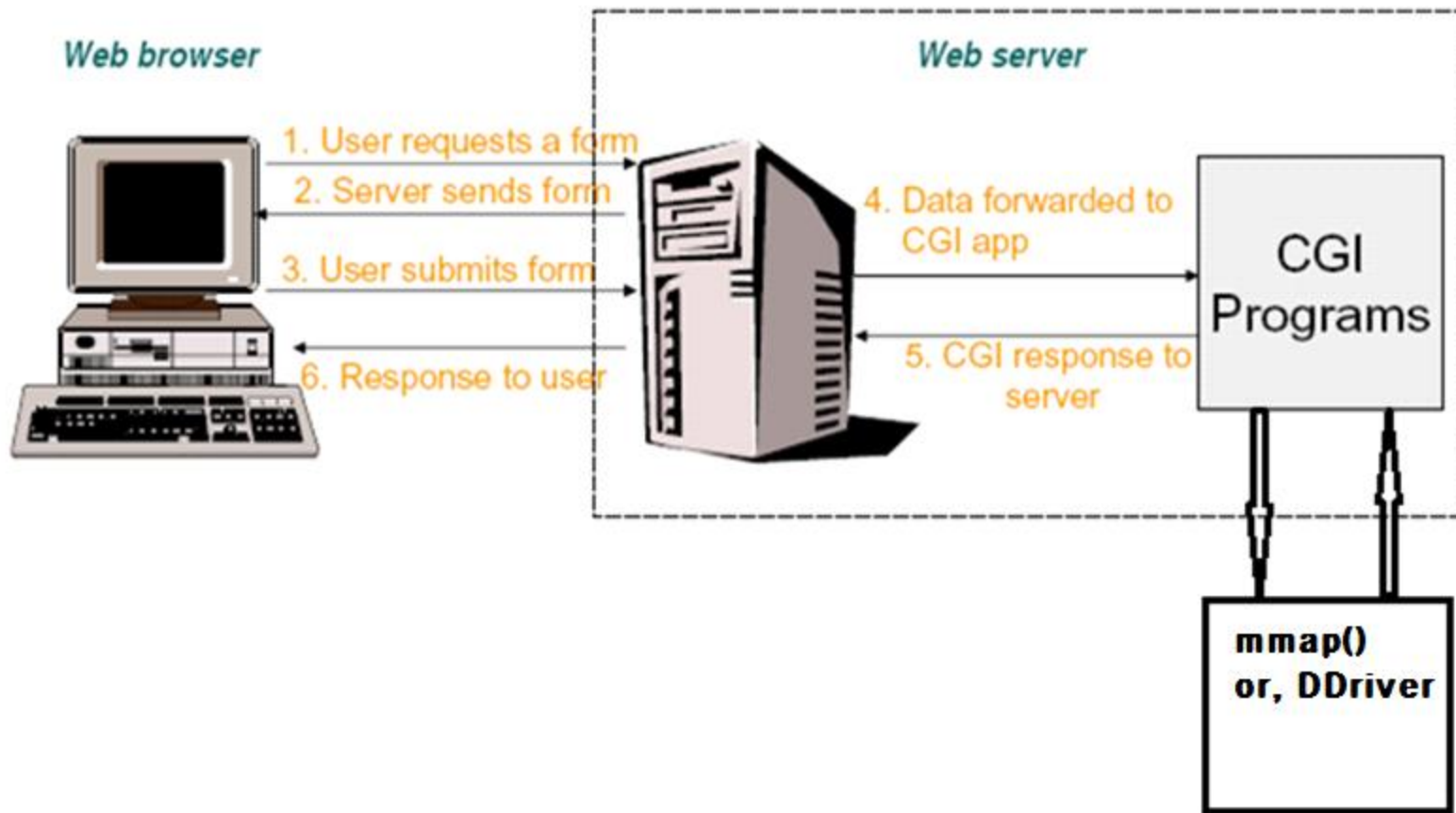
The screenshot shows a web browser window with the address bar displaying 'http://192.168.0.40/cgi-bin/testPost.cgi'. The page title is 'CGI Test, by POST'. The page content displays the following information:

- 'Your IP Address : 192.168.0.10'.
- 'POST STRING : name=hello.....&num=89&led=1&ledArray=8&ledArray=0&transmit=transmit'.



# CGI에 의한 디바이스 제어

## \* 웹기반 디바이스 원격제어



## CGI에 의한 디바이스 제어(계속)

### \* CGI에 의한 디바이스 제어 사전 조치

: wiringPi가 접근하는 디바이스 파일 **/dev/gpiomem**  
( 디바이스의 GPIO 가상주소 위해, mmap() 함수 )

: 웹 클라이언트가 접속하여 접근할 수 있도록 변경

: 현재의 접근 설정 확인

\$ **ls -l /dev/gpiomem**

crw-rw----- 1 root gpio 243, 0 Jan 15 10:44 gpiomem

: 접근 속성 변경( 웹에서 접속가능한 누구나 RWX 접근토록, **667** )

\$ **sudo chmod 667 /dev/gpiomem**

crw-rw-**rwX** 1 root gpio 243, 0 Jan 15 10:44 gpiomem

**유의) 재부팅하는 경우, 원래의 접근속성으로 되돌아감**

## CGI에 의한 디바이스 제어(계속)

참고) CGI에 의한 디바이스 제어 사전 조치

(2020.11.2)

— : raspi-config 환경설정의 Remote GPIO 항목 활성화

— : 시도해 볼 것!!!!

— \$ sudo raspi-config

— -5 Interfacing Options

— -P8 Remote GPIO

## CGI에 의한 디바이스 제어(계속)

### \* 참고) 접근속성 영구 설정

— : /dev/gpiomem 디바이스 파일에 대해 접근속성 설정

— : 다음의 파일 편집 ( </etc/udev/rules.d/> )

— : 참고사이트 ... <https://karl27.tistory.com/4>

— \$ [sudo nano /etc/udev/rules.d/99-com.rules](#)

— [SUBSYSTEM=="gpiomem", GROUP="gpio", MODE="0666"](#)

— [SUBSYSTEM=="input", GROUP="input", MODE="0660"](#)

— .....

— \$ [sudo reboot](#)

— \$ [ls -l /dev/gpiomem](#)

— [crw-rw---- 1 root gpio 246, 0 Aug 19 11:27 /dev/gpiomem](#)

— [\(안됨....., 추후 재 확인.....\)](#)

## CGI에 의한 디바이스 제어(계속)

[실습5] GET 방식의 LED 제어 ( [./led/ 참조](#) )

: 데이터를 GET 방식으로 전달

: 전달 문자열의 첫 문자(0 혹은 1)로 LED 제어

```
$ nano ledGet.html
```

```
<h1>LED control..... GET ...</h1>  
<hr><p>
```

```
<FORM method=GET action="./cgi-bin/ledGet.cgi">
```

```
<label>LED : </label>  
<input type=text name=name maxlength=10>  
<p>
```

```
<input type=submit name=transmit value=transmit>  
<p>  
</FORM>
```

- : 소스의 빨강 부분은 LED 디바이스의 제어와 관련된 부분
- : 녹색부분은 전달값의 토큰 추출과 관련된 부분
- : getToken() 함수는 문자열에서 구분자를 이용하여 토큰을 추출하는 함수

```
$ nano ledGet.c
```

```
//=====
// ledGet.c CGI
// LED controll, by GET method
//=====
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#include <wiringPi.h>      ////

#define P_LED    1        //// BCM_GPIO 18

typedef struct {
    char name[32];
    char val[32];
} entry;

void getToken(char *word, char *qStr, char deli);
```



```
int main(void) {
    char *qStr;
    entry item;

    if(wiringPiSetup() == -1)                //// wiringPi pin #
        return 1;

    pinMode(P_LED, OUTPUT);                ////

    printf("Content-type: text/html\n\n");
    printf("<html>\n\n");
    printf("<head>\n");
    printf("<title>LED CGI by GET...</title>\n");
    printf("</head>\n\n");
    printf("<body>\n");
    printf("<h1>LED CGI by GET...</h1>\n");
    printf("<hr><p>\n");

    printf("Your IP Address : %s<br>\n", getenv("REMOTE_ADDR"));

    printf("QUERY_STRING : %s<br><p>\n", getenv("QUERY_STRING"));
    qStr = (char *)getenv("QUERY_STRING");
    getToken(item.name, qStr, '=');
    getToken(item.val, qStr, '&');

    printf("Extracted token : %s = %s<br>", item.name, item.val);
```

```
//// LED control....
if(item.val[0] == '0') {
    digitalWrite(P_LED, LOW);
    printf("==> Led OFF.....<br>");
}
else if(item.val[0] == '1') {
    digitalWrite(P_LED, HIGH);
    printf("==> Led ON.....<br>");
}
else
    printf("==> Wrong data.....<br>");

printf("</body>\n\n");
printf("</html>\n");

return 0;
}

//.....
// 이하 다음 쪽에서
```

```
// get token
void getToken(char *word, char *str, char deli) {
    int i, new_i;

    // extract token
    for(i=0; ((str[i]) && (str[i] != deli)); i++)
        word[i] = str[i];

    word[i] = '\0';

    if(str[i]) // skip delimiter character
        i++;

    // make a string of remaining str
    new_i = 0;
    while(str[new_i++] = str[i++])
        ;
}
```

: 컴파일후 관련 파일들을 복사

```
$ gcc -o ledGet.cgi ledGet.c -lwiringPi
```

```
$ sudo cp ledGet.cgi /var/www/html/cgi-bin/
```

```
$ sudo cp ledGet.html /var/www/html/
```

: 실행 및 결과 관찰

- 웹 브라우저에서 <http://192.168.0.30/ledGet.html>로 접속 (좌측)
- 입력 문자열중 첫 문자는 LED 디바이스를 제어하는 문자(0 혹은 1로 시작..)
- LED를 ON 하는 경우의 실행 결과 화면( 우측 )
- 라즈베리 보드 상의 LED ON을 확인



## CGI에 의한 디바이스 제어(계속)

[실습6] POST 방식의 LED 제어 ( [./led/ 참조](#) )

: 데이터를 POST 방식으로 전달

: 전달 문자열의 첫 문자(0 혹은 1)로 LED 제어

```
$ nano ledPost.html
```

```
<h1>LED Control, by POST...</h1>  
<hr><p>
```

```
<FORM method=POST action="./cgi-bin/ledPost.cgi">
```

```
<label>LED : </label>  
<input type=text name=name maxlength=10>  
<p>
```

```
<input type=submit name=transmit value=transmit>  
<p>
```

```
</FORM>
```

```
$ nano ledPost.c
```

```
//=====
// ledPost.c CGI
// LED control, by POST method
//=====
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#include <wiringPi.h>      ////

#define P_LED  1          //// BCM_GPIO 18

typedef struct {
    char name[32];
    char val[32];
} entry;

void getToken(char *word, char *qStr, char deli);

int main(void) {
    char buf[1024];
    char *qStr;
    entry item;

    if(wiringPiSetup() == -1)        //// wiringPi pin #
```

```
        return 1;

pinMode(P_LED, OUTPUT);          ////

printf("Content-type: text/html\n\n");
printf("<html>\n\n");
printf("<head>\n");
printf("<title>LED CGI by POST...</title>\n");
printf("</head>\n\n");
printf("<body>\n");
printf("<h1>LED CGI by POST...</h1>\n");
printf("<hr><p>\n");

printf("Your IP Address : %s<br>\n", getenv("REMOTE_ADDR"));

memset(buf, 0x00, 1024);
while(read(0, buf, 1024) > 0) {
    printf("POST STRING : %s<p>\n", buf);
//    memset(buf, 0x00, 1024);
}

qStr = buf;
getToken(item.name, qStr, '=');
getToken(item.val, qStr, '&');
printf("Extracted token : %s = %s<br>", item.name, item.val);

//// LED control....
if(item.val[0] == '0') {
```

```
        digitalWrite(P_LED, LOW);
        printf("==> Led OFF.....<br>");
    }
    else if(item.val[0] == '1') {
        digitalWrite(P_LED, HIGH);
        printf("==> Led ON.....<br>");
    }
    else
        printf("==> Wrong data.....<br>");

    printf("</body>\n\n");
    printf("</html>\n");

    return 0;
}

// get token
void getToken(char *word, char *str, char deli) {
    int i, new_i;

    // extract token
    for(i=0; ((str[i]) && (str[i] != deli)); i++)
        word[i] = str[i];
    word[i] = '\0';

    if(str[i]) // skip delimiter character
        i++;
}
```



```
// make a string of remaining str
new_i = 0;
while(str[new_i++] = str[i++])
    ;
}
```

**: 컴파일후 관련 파일들을 복사**

```
$ gcc -o ledPost.cgi ledPost.c -lwiringPi
```

```
$ sudo cp ledPost.cgi /var/www/html/cgi-bin/
```

```
$ sudo cp ledPost.html /var/www/html/
```

**: 실행 및 결과 관찰**

- 웹 브라우저에서 <http://192.168.0.30/ledPost.html>로 접속 (좌측)
- 입력 문자열중 첫 문자는 LED 디바이스를 제어하는 문자(0 혹은 1로 시작..)
- LED를 OFF 하는 경우의 실행 결과 화면( 우측 )
- 라즈베리 보드 상의 LED OFF를 확인



**\* 개인 휴대폰 활용**

: 사전에 무선(WiFi)망 설정되어 있어야 함

: 웹서버로 접속하여 LED 제어 실습

: 휴대폰으로 WiFi 연결 ( 각자의 SSID 및 비밀번호 )

: <http://192.168.0.40/led/Get.html>로 접속

: 혹은, <http://192.168.0.40/ledPost.html>로 접속

## 응용 과제

[응용 1] LED 디바이스 원격제어 ( [./led/](#) )

: [실습5] 소스를 근간으로 하여.. ( GET 방식 )

: HTML 태그로 라디오 버튼을 활용하여 재구현



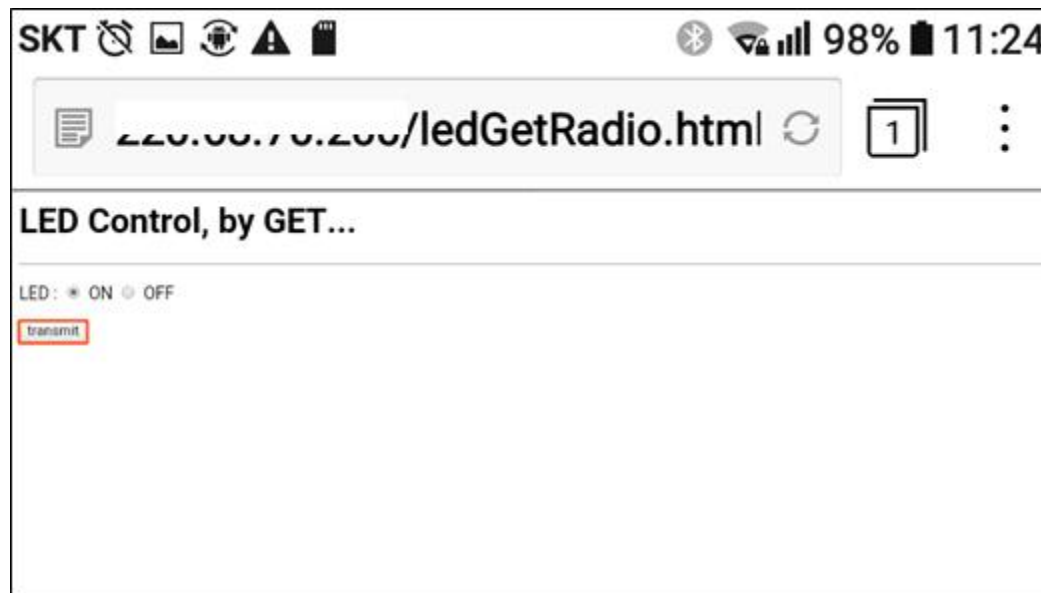
```
$ nano ledGetRadio.html
```

```
<h1>LED Control, by GET...</h1>  
<hr><p>  
<FORM method=GET action="./cgi-bin/ledGet.cgi">  
  <label>LED : </label>  
  <input type=radio name=led value=1 checked> ON  
  <input type=radio name=led value=0> OFF  
<p>  
  <input type=submit name=transmit value=transmit>  
<p>  
</FORM>
```

## 응용 과제(계속)

### [응용2] LED 디바이스 원격제어 ( [./led/](#) )

- : 집에서 개인 휴대폰으로 접속하여 테스트
- : [응용1]의 소스 참조 ( [./led/....Radio.\\*](#) )
- : 포트 포워딩 및 전산소에 포트 개방 요청 필요
- : 접속 IP 주소는 공유기의 외부 IP 주소여야 함



## 응용 과제(계속)

### [응용3] 마그네틱 스위치로 LED 제어 ( [./magled/](#) )

: 마그네틱 스위치로 LED ON/OFF 제어

: GET 방식 (아래 소스), 혹은 POST 방식 (소스 참조)

```
$ nano magledGet.html
```

```
<h1>LED Controlled by Magnetic SW (GET)</h1>
<hr><p>
<FORM method=GET action="./cgi-bin/magledGet.cgi">
<input type=submit name=transmit value=Run>
<p>
</FORM>
```

```
$ nano magledGet.c
```

```
//=====
// magledGet.c CGI
// LED controlled by Magnetic SW (GET)
//=====
#include <stdio.h>
```

```
#include <stdlib.h>
#include <string.h>

#include <wiringPi.h>          ////

#define P_LED  1          //// BCM_GPIO 18
#define P_MAG  4          //// BCM_GPIO 23

int main(void) {
    int btn;

    if(wiringPiSetup() == -1)          //// wiringPi pin #
        return 1;

    pinMode(P_LED, OUTPUT);            ////
    pinMode(P_MAG, INPUT);             ////

    printf("Content-type: text/html\n\n");
    printf("<html>\n\n");
    printf("<head>\n");
    printf("<title>LED Controlled by Magnetic (GET)</title>\n");
    printf("</head>\n\n");
    printf("<body>\n");
    printf("<h1>LED Controlled by Magnetic (GET)</h1>\n");
    printf("<hr><p>\n");

    // LED controlled by Magnetic...
    btn = digitalRead(P_MAG);
```



```
if(btn == 1) {  
    digitalWrite(P_LED, HIGH);  
    printf("Magnetic ON ==> LED ON!!!<br>");  
}  
else {  
    digitalWrite(P_LED, LOW);  
    printf("Magnetic OFF ==> LED OFF!!!<br>");  
}  
  
printf("</body>\n\n");  
printf("</html>\n");  
  
return 0;  
}
```

**\* BTN을 누른 상태에서 Run 버튼을 클릭하여 테스트**