웹 프로그래밍

08. 중간 요약

김승호

- 범용 스크립트 언어인 Python 과 Python으로 웹 서버를 쉽게 만들 수 있게 도와주는 패키지 조합
- (PHP처럼) 웹 서버 설정으로 특정 주소를 연동할 수도 있지만, 내장 웹서버로 바로 실행도 가능함

```python
from flask import *             # flask 패키지 불러오기
app = Flask(__name__)           # 인스턴스 생성

@app.route('/test')             # /test 주소로 접속시 실행될 함수 지정
def test():
    messages = [
        {'author': '작성자1', 'content': 'Hello world!'},
        {'author': '작성자2', 'content': ''}
    ] # Python의 list, dict, str 자료형을 사용하여 데이터 정의

    # templates/test.html 파일을 Jinja 템플릿 엔진으로 렌더링, 템플릿에서 사용할 messages변수 설정
    return render_template('test.html', messages= messages)

# 내장 웹서버 실행, 코드의 가장 마지막 줄에 있어야 함
# host='0.0.0.0': 다른 컴퓨터에서도 접속할 수 있도록 설정 / port=5000: 웹 서버를 5000포트에 실행
app.run(host='0.0.0.0', port=5000, debug=True)
```

python + Flask 의 템플릿 엔진 (Jinja)

```
messages = [
    {'author': '작성자1', 'content': 'Hello world!'},
    {'author': '작성자2', 'content': ''}
]
```

템플릿으로 데이터 주입

```
<!doctype html>
<html lang="ko">
    <head>
    <meta charset="utf-8" />
    </head>
    <body>
{% for message in messages %}
    작성자: {{ message['author'] }}<br>
    {% if message['content'] == '' %}
        입력된 내용이 없습니다.
    {% else %}
        내용: {{ message['content'] }}
    {% endif %}
    <hr>
{% endfor %}
    </body>
</html>
```

템플릿 렌더링

```
<!doctype html>
<html lang="ko">
    <head>
    <meta charset="utf-8" />
    </head>
    <body>
    작성자: 작성자1<br>
    내용: Hello world!
    <hr>
    작성자: 작성자2<br>
    입력된 내용이 없습니다.
    <hr>
    </body>
</html>
```

작성자: 작성자1
내용: Hello world!

작성자: 작성자2
입력된 내용이 없습니다.

웹브라우저 렌더링

```php
<?php
    // 한줄 주석
    echo "안녕하세요!<br>";
    $now = date("Y-M-D H:m:s");
    echo $now;
    /*
        여러줄 주석
    */
?>
```

- **<?php** 로 시작해서 **?>** 로 끝남
- 변수는 $ 기호로 시작. 예: $foo, $bar
- 변수 출력만 할 경우엔 **<?=$foo?>** 로 축약 가능

  <?php echo $foo; ?>** == **<?=$foo?>
- 웹서버 설정에 따라 작동
  - web.dgsw.kr 서버는 .php 확장자에 대해 PHP 언어를 처리하도록 설정되어 있음

```php
<!doctype html>
<html lang="ko">
    <head>
        <meta charset="utf-8" />
    </head>
    <body>
        현재시간: <?php echo date("Y-m-d H:m:s"); ?><br>
        접속한 IP: <?php echo $_SERVER["REMOTE_ADDR"]; ?><br>
    </body>
</html>
```

PHP 인터프리터가 처리

```html
<!doctype html>
<html lang="ko">
    <head>
        <meta charset="utf-8" />
    </head>
    <body>
        현재시간: 2021-03-24 09:00:00<br>
        접속한 IP: 169.254.169.254<br>
    </body>
</html>
```

웹브라우저 렌더링

현재시간: 2021-03-24 09:00:00
접속한 IP: 169.254.169.254

HTML

HyperText Markup Language

- 웹페이지에서 제목, 단락, 목록 등 본문을 위한 구조적 의미를 나타내거나,
  링크, 인용 등 구조적 문서를 만들 수 있는 방법을 제공하는 **마크업 언어**
- <tag>로 시작해서 </tag> 로 끝남
- 현재 표준이자 최신은 5번째 버전인 HTML5

```
<!doctype html>
<html>
  <head>
    <title>Hello HTML</title>
  </head>
  <body>
    <p>Hello World!</p>
  </body>
</html>
```

HTML5 버전 사용을 의미,
생략시 호환성 모드가 적용되며 최신 기능이 오작동할 수 있음

[기본 구조]
최상위로 <html> 태그가 사용
  문서의 정보를 담는 <head> 태그
  표시되는 본문 내용이 있는 <body> 태그

CSS

Cascading Style Sheets

- HTML 등 마크업 언어가 실제 표시되는 방법을 기술하는 언어
- 레이아웃, 스타일을 정의할 수 있으며 그림자, 그라데이션, 변형, 애니메이션 등도 지원함
- HTML 문서내에 <style> 태그안에 사용하거나, <link> 태그로 css 파일을 불러올 수 있음

```
p{
    font-size: 110%;
    font-family: garamond, sans-serif;
}
h2{
    color: red;
    background: white;
}
.highlight{
    color: red;
    background: yellow;
    font-weight: bold;
}

#test_id {
    color: blue;
    background: white;
}
```

[CSS 선택자 (Selector)]
스타일을 적용할 요소를 선택하는 문법
- 태그이름 { }
- #요소ID { }
- .요소class { }

Form 전송

- <form> 태그 내 <input>, <textarea> 등 태그에 입력된 데이터를 서버에 전송
- 요소의 name 속성을 key로, value를 값으로 사용해서 데이터 전송
  - ex, <input type="text" name="키" value="값">

- 태그 속성
  - method: HTTP 메소드 지정, GET 혹은 POST 만 사용가능, 생략시 GET
  - action: 폼을 전송할 URL 지정, 생략시 현재 페이지와 같은 주소
  - ex, <form method="GET" action="/login">

GET vs POST

```
GET /03/calc.php?a=1&op=add&b=2 HTTP/1.1
Host: web.dgsw.kr
User-Agent: ...
```

- URL에 파라미터로 전송
- 캐시와 방문 기록에 남음
- 길이 제한이 존재

페이지 정보, 검색어 등 단순 정보 조회용 데이터에 적합

```
POST /03/calc.php HTTP/1.1
Host: web.dgsw.kr
Content-Length: 14
Content-Type: application/x-www-form-urlencoded
User-Agent: ...

a=1&op=add&b=2
```

- HTTP Request Body 로 전송
- 캐싱되지 않으며 방문 기록에도 남지 않음
- 길이 제한 없음

글 작성, 로그인 등 길이가 길거나 캐싱되면 안되는 내용에 적합

GET, POST로 전송된 데이터 사용

Python

- 값 접근

```
request.values['키']
request.values.get('키', '기본값')
```

- 값이 존재하는지 확인

```
if '키' in request.values:
    값이 있음
else:
    값이 없음
```

PHP

- 값 접근

```
$_REQUEST['키']
```

- 값이 존재하는지 확인

```
if (isset($_REQUEST['키'])) {
    값이 있음
}else{
    값이 없음
}
```

민감 정보 소스코드에서 분리하기

Python

- 환경변수 및 설정파일로 사용
- settings.cfg

```
SECRET_KEY = '12345678'
```

- app.py

```
# 환경변수에 설정파일의 위치를 기록
import os
os.environ['APP_SETTINGS'] =
'settings.cfg'

# 환경변수에 저장된 경로의 설정파일을 불러옴
app.config.from_envvar('APP_SETTINGS')
```

PHP

- 다른 파일에 민감 정보만 모은 후 해당 파일 include

- settings.php

```
$SECRET_KEY = '12345678';
```

- index.php

```
include 'settings.php';
```

Cookie(쿠키) 와 **Session**(세션)

- HTTP Protocol은 요청/응답 후 통신을 끊으며(Connectionless, 비연결지향)
  요청을 보낼 때 이전 통신에 대한 정보를 유지하지 않음(Stateless, 상태정보 없음)

- 하지만, 로그인이나 쇼핑몰의 장바구니 등 사용자 정보를 저장하고 유지할 필요가 있음

- **Cookie**
  - 브라우저에 저장되는 최대 4KiB의 정보
  - 웹 요청을 보낼 때 함께 전송됨

- **Session**
  - 서버에 저장, 활용 (저장 방식은 구현하기 나름)

- 주로 쿠키에 세션키를 저장하고, 서버에서 해당 세션키로 사용자 정보를 불러와서 사용함

Session 사용

Python

- 세션 시작
  - (secret_key 설정)

- 값 접근
  - `session['키']`
  - `session.get('키', '기본값')`
- 값 삭제
  - `del session['키']`

- 값이 존재하는지 확인은 이전과 동일

PHP

- 세션 시작
  - `session_start();`

- 값 접근
  - `$_SESSION['키']`
- 값 삭제
  - `unset($_SESSION['키'])`

- 값이 존재하는지 확인은 이전과 동일

Database 연결

Python

- settings.cfg

```
MYSQL_DATABASE_HOST = 'web.dgsw.kr'
MYSQL_DATABASE_USER = 's2000'
MYSQL_DATABASE_PASSWORD = 'xxxx'
MYSQL_DATABASE_DB = 's2000'
```

- app.py

```
from flaskext.mysql import MySQL
from pymysql.cursors import DictCursor

mysql = MySQL(cursorclass= DictCursor)
mysql.init_app(app)
```

PHP

```
// HOST, USER, PASSWORD, DB
$mysqli = new mysqli('localhost',
's2000', 'xxxx', 's2000');
```

Database 쿼리 실행

Python

```python
conn = mysql.get_db()
cursor = conn.cursor()
cursor.execute('SELECT * FROM users;')

# 결과 1개 가져오기
row = cursor.fetchone()

# 결과 전부 가져오기
result = cursor.fetchall()

# INSERT, UPDATE, DELETE 등 수정사항 발생시
conn.commit() # 수정사항 저장
conn.rollback() # 수정사항 되돌리기
```

PHP

```php
$cursor = $mysqli->query("SELECT * FROM
users;");

// SELECT 구문일 경우 가져온 행 갯수
echo $cursor->num_rows;

// 결과 1개 가져오기
$row = $cursor->fetch_assoc();


// 결과 전부 가져오기
while ($row = $cursor->fetch_assoc()) {
    // $row['column']

}
```

XSS (Cross Site Scripting)

- 관리자가 아닌 사람이 임의의 스크립트를 삽입할 수 있는 취약점
- 웹 어플리케이션이 사용자의 입력값을 제대로 검사하지 않거나 필터링을 하지 않았을 때 발생
- 사용자의 정보(쿠키, 세션)가 탈취되거나 비정상적 기능을 수행하게 하는 등 악의적 행위 가능
- 방어법: HTML 특수문자를 HTML 엔티티로 변환
  - & → & " → " ' → ' < → < > → >

XSS 방어

Python

- Jinja 템플릿 엔진을 사용할 경우 기본적으로 방어됨

- HTML 태그를 그대로 출력하고 싶다면

```
{{ 변수명 | safe }}
```

PHP

```php
$content = "...";

echo htmlspecialchars($content);
```

- htmlspecialchars(string 변환문자열, [int 따옴표 스타일, string 인코딩, bool 중복인코드여부]);
  - 따옴표 스타일 - ENT_COMPAT: 쌍따옴표만 / ENT_QUOTES: ",' 둘다 / ENT_NOQUOTES: 따옴표는 변환 X
  - 인코딩 - 기본값은 ISO-8859-1
  - 중복인코딩여부 - 이미 HTML 엔티티로 변환된 문자를 다시 변환할지 여부. ex) < → <

XSS 방어 (2)

- HTML에서 줄바꿈을 위해선
 태그를 사용해야하지만, 앞의 방법은 모든 태그 사용을 막게 됨
- 다양한 해결법이 있지만, 예시는 escape를 해서 모든 태그를 없앤 후
태그를 만드는 방법을 사용

Python

```python
# 먼저 content변수 내용을 escape 처리
import html
content = html.escape(content)

# 개행문자를 <br> 태그로 변경
content = content.replace('\n', '<br>\n')
```

...
html 파일 에서는
`{{ content | safe }}` 로 사용

PHP

```php
// 먼저 content변수 내용을 escape 처리
$content = htmlspecialchars($content);

// 개행문자를 <br> 태그로 변경
$content = str_replace("\n", "<br>\n",
$content);

// 출력
echo $content;
```

SQL Injection

- 사용자의 입력값이 DB 쿼리문에 사용될 때, 의도하지 않은 SQL 쿼리가 실행될 수 있는 취약점

예시) `SELECT * FROM users WHERE user_id = '사용자ID' AND user_pw = '사용자PW';`

사용자ID에 "`' OR 1 = 1 --` " 같은 값이 검증 없이 입력되면, 아래와 같은 쿼리문이 됨.

```
SELECT * FROM users WHERE user_id = '' OR 1 = 1 -- ' AND user_pw = '사용자PW';
```

`--` 는 SQL에서 주석이므로 이는

```
SELECT * FROM users WHERE user_id = '' OR 1 = 1 -- ' AND user_pw = '사용자PW';
```

같이 해석되어 ID, PW 와 상관없이 사용자 데이터가 반환이 됨

- 방어법: 따옴표 등 특수문자를 escape처리 (\' 와 같이 앞에 역슬래시를 붙이기)

SQL Injection 방어

Python

```
cursor.execute('SELECT * FROM users WHERE user_id = %s AND user_pw = %s;', [id, pw])
```

PHP

```
$id = $mysqli->real_escape_string($id);
$pw = $mysqli->real_escape_string($pw);

$cursor = $mysqli->query("SELECT * FROM users WHERE user_id = '" . $id . "' AND user_pw =
'" . $pw . "';");
```

로그인, 로그아웃 만들어보기

- 제공된 HTML, CSS를 참고만 하거나 그대로 사용하여 만드셔도 됩니다.
- 언어는 Python or PHP 택1

- 로그인 (login.html 참고)
  ID, 비밀번호를 이용하여 로그인
  로그인 성공시 메인페이지로 이동하며 상단바에 이름과 로그아웃 버튼 표시 (main_logged.html 참고)
  로그인 실패시 오류 표시

- 회원가입 (join.html 참고)
  ID, 이름, 비밀번호를 입력하여 회원가입
  같은 ID로 로그인할 경우 오류 표시

- 로그아웃
  로그인된 정보 삭제