



# Flask Start

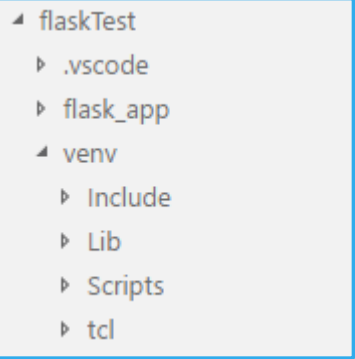
2018년 7월

# Contents

- 플라스크 웹 개발 작업 환경 설정
- Flask Start
- Route
- 동적 파라미터 전달하기

# 플라스크 웹 개발 작업 환경 설정

- 파이썬 3.5.1
- Visual Studio Code (VSCODE)
- 가상환경 설치
  - `pip install virtualenv`
- 가상환경 활성화
  - 작업 폴더로 이동 : `cd 작업폴더명`
  - `virtualenv venv`



```
flaskTest
├── .vscode
├── flask_app
└── venv
    ├── Include
    ├── Lib
    ├── Scripts
    └── tcl
```

작업 폴더 아래에  
venv 폴더 생성


# 플라스크 웹 개발 작업 환경 설정

- FLASK 0.10.1

- 작업폴더로 이동 후 venv/Scripts 폴더로 이동
- pip install flask

```
C:\Users\queen>cd C:\_StudyPython\workspace\flaskTest  
C:\_StudyPython\workspace\flaskTest>cd venv/Scripts  
C:\_StudyPython\workspace\flaskTest\venv\Scripts>pip install flask
```

- VSCODE에서 PyDev 설치



## Python (PyDev)

fabioz.vscode-pydev

Fabio Zadrozny | 17,184 | ★★★★★ | 리포지토리 | 라이선스

Python with the PyDev Language Server (Linting, Intellisense, Code Formatting, Quick Fixes and more).

[사용 안 함 ▼](#) [제거](#)

# FLASK Start

## Step1. flask\_stat.py

```
# 플라스트 모듈 импорт
from flask import Flask

# Flask 객체를 app에 할당
app = Flask(__name__)

# app 객체를 이용하여 라우팅 경로 설정
@app.route("/")

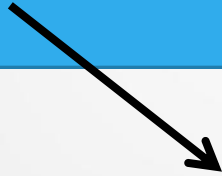
# 해당 라우팅 경로로 요청이 올 경우 실행할 함수 작성
def helloworld():
    return "Hello World"

# 메인 모듈로 실행될 때 플라스크 서버 구동
if __name__ == "__main__":
    app.run()
```

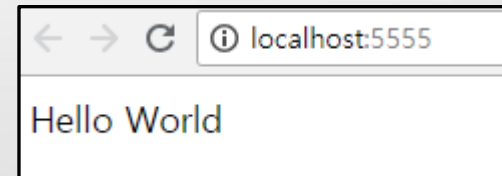
# FLASK Start

Step2 - flask\_stat.py 파일 실행

Running on <http://localhost:5555/> (Press CTRL+C to quit)



Vs code 프로그램의 경우  
Ctrl 클릭해서 웹브라우저에서 확인



# 라우트

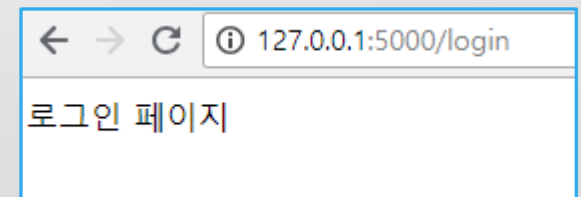
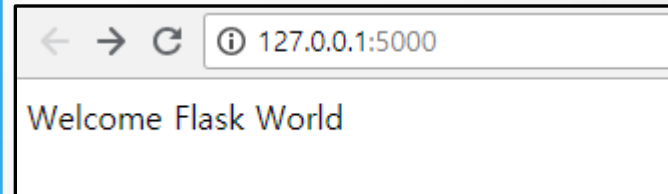
- 라우트:
  - 클라이언트의 요청의 URL를 보고 어떤 함수가 처리할것인지(응답) 연결해주는 기능(라우팅)

```
from flask import Flask  
app = Flask(__name__)
```

```
@app.route('/')  
def home():  
    return 'Welcome Flask World'
```

```
@app.route('/login')  
def login():  
    return '로그인 페이지'
```

```
if __name__ == '__main__':  
    app.run(debug=True)
```



# 동적 파라미터로 데이터 전달하기

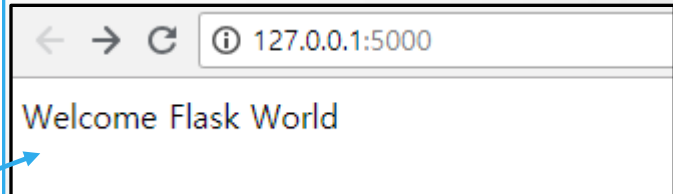
- 동적 파라미터:
  - URL 경로에 데이터를 전달한다.
  - 형식 : ~/<데이터1>/<데이터2>
  - 함수의 인자를 통해서 전달받아서 return 값으로 수정 가능

```
from flask import Flask
app = Flask(__name__)

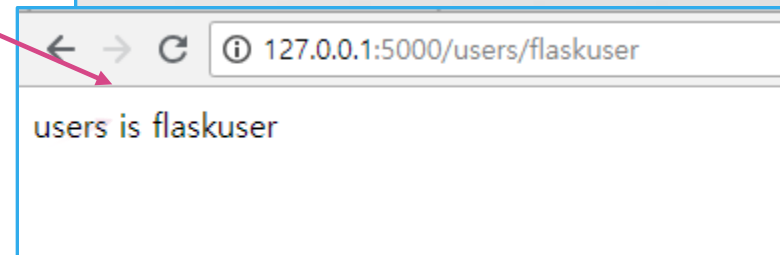
@app.route('/')
def home():
    return 'Welcome Flask World'

@app.route('/users/<userId>')
def users(userId):
    return 'users is %s' % userId

if __name__ == '__main__':
    app.run(debug=True)
```



~/users/전달값





# 동적 파라미터로 데이터 전달하기

- 동적 파라미터:
  - 여러 개의 파라미터 전달시 ~ /<데이터1>/<데이터2>

```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def home():
    return 'Welcome Flask World'

@app.route('/users2/<userId>/<userName>')
def users2(userId,userName):
    return '아이디 %s, 이름 %s' % (userId,userName)

if __name__ == '__main__':
    app.run(debug=True)
```

~/users/전달값1/전달값2

← → ↻ ⓘ 127.0.0.1:5000/users2/flaskUser/마동탁

아이디 flaskUser, 이름 마동탁

# 동적 파라미터의 데이터 타입

- 동적 파라미터의 타입 제한
  - 초기 데이터 타입값은 String
  - ~/라우터명/<데이터타입:데이터>
  - 예) <int/number>

~/users/<데이터타입:데이터>

```
from flask import Flask
app = Flask(__name__)
```

```
@app.route('/')
def home():
    return 'Welcome Flask World'
```

```
@app.route('/news/<int:newsid>/<int:start>')
def news(newsid, start):
    return "뉴스 %s %s" % (newsid, start)
```

```
if __name__ == '__main__':
    app.run(debug=True)
```

← → ↻ ⓘ 127.0.0.1:5000/news/2000/456

뉴스 2000 456

- **int**: 정수(integer)로 사용합니다.
- **float**: 부동소수 값(float)
- **path**: 디렉토리에 사용된 '/'를 독립된 character로 사용

# 동적 파라미터의 데이터 타입

- 동적 파라미터의 타입 제한
  - 초기 데이터 타입값은 String
  - ~/라우터명/<데이터타입:데이터>
  - 예) <path/number>

```
# path로 전달
@app.route('/p/<path:id>')
def p(id):
    # print(type(id))
    data = id.split('/')
    for d in data:
        print(d)
    return "path 방식 : %s" % id
```

~/users/<데이터타입:데이터>

← → ↻ ⓘ 127.0.0.1:5000/p/홍길동

path 방식 : 홍길동

홍길동

127.0.0.1 - - [11/Aug/2018 10:02:37] "GET /p/%ED%99%8D%EA%B8%B8%EB%8F%99 HTTP/1.1" 200 -

# redirect

- 리다이렉트
- url\_for() 함수 이용하기
- Test\_request\_context() 함수 이용하기

```
from flask import Flask, url_for
app = Flask(__name__)
@app.route('/')
def home():return 'home'
@app.route('/users/login')
def users():return 'login'
@app.route('/users/<userID>')
def users2(userID):return userID
```

*# url 요청 테스트*

```
with app.test_request_context():
    print( url_for('home') )
    print( url_for('users') )
    print( url_for('users2', userID='ry327282') )
```

```
if __name__ == '__main__':
    app.run(debug=True)
```

```
/
/users/login
/users/ry327282
```

# Restful

- Request 모듈 불러오기
- Get/post 방식
- ~/login: 로그인 폼 : GET / ~/login: 로그인 처리 : POST

```
from flask import Flask, request
app = Flask(__name__)
@app.route('/')
def home():
    return 'Welcome Flask World'
```

```
# get방식, post 방식 모두 지원
@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'GET':
        return 'login GET'
    else:
        return 'login POST'
```

```
if __name__ == '__main__':
    app.run(debug=True)
```



# Static

- 정적 파일 위치 : /static
- html, css, js, images, 업로드파일 용도로 이용

```
from flask import Flask, url_for
```

```
app = Flask(__name__)
```

```
@app.route('/')
```

```
def home():
```

```
    return 'Welcome Flask World'
```

```
with app.test_request_context():
```

```
    print( url_for('static', filename='images/a.jpg') )
```

```
if __name__ == '__main__':
```

```
    app.run(debug=True)
```

```
else:
```

```
    print('본 서버는 메인으로 구동시에만 작동된다.')
```

확인하기 :

<http://127.0.0.1:5000/static/images/a.png>

# Jinja Template

- Jinja2 엔진 사용 : render\_template 모듈 이용
- render\_template('html 파일경로')
- Html 렌더링 처리
- html 파일 위치 : /template

```
from flask import Flask, render_template
```

```
app = Flask(__name__)
```

```
@app.route('/')  
def home():
```

```
    return render_template('index.html')
```

```
if __name__ == '__main__':  
    app.run(debug=True)
```

확인하기 :

<http://127.0.0.1:5000> -> index.html

# Jinja Template

- html 파일 위치 : /template

```
<html>
<head>
<meta charset='utf-8'>
<title>플라스크월드</title>
  <style type="text/css">
    #wrap {
      text-align:center;
      background:red;
      color:aliceblue;
      width:400px;
      margin:50px auto;
      padding:20px
    }
  </style>
</head>
```

```
<body>
  <div id="wrap">
    <h2>Welcome Flask World</h2>
  </div>
</body>
</html>
```



# Jinja Template - 값전달

- 데이터값 전달하기

```
# html 렌더링 처리  
# Jinja2 엔진 사용  
from flask import Flask, render_template
```

```
app = Flask(__name__)
```

```
@app.route('/')  
@app.route('/<uid>')  
def home(uid=None):  
    return render_template('index_user.html', user=uid)
```

```
if __name__ == '__main__':  
    app.run(debug=True)
```

확인하기 :

<http://127.0.0.1:5000> -> index\_user.html

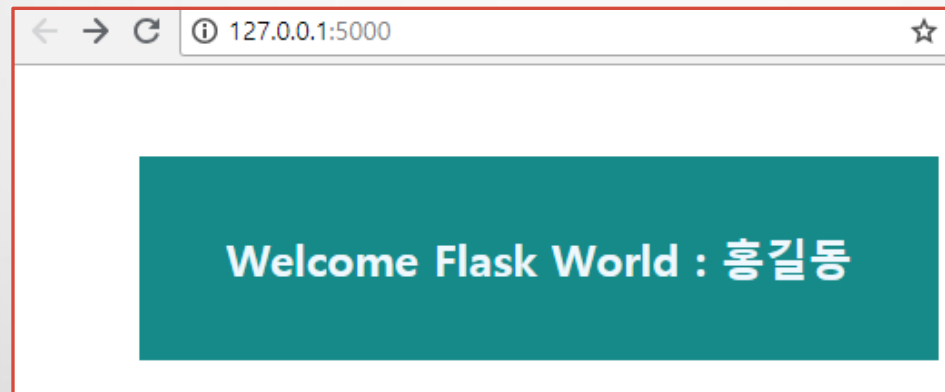
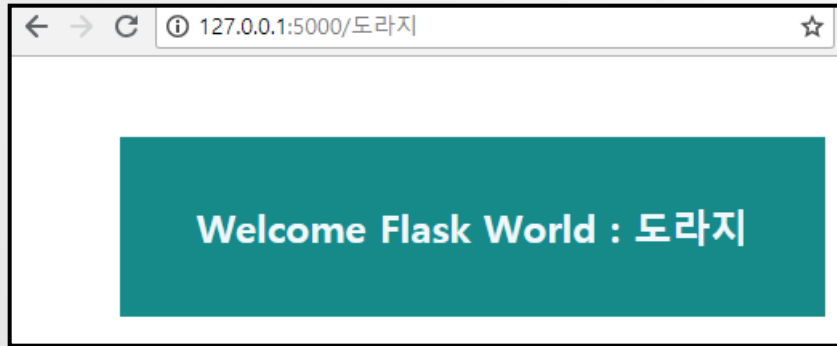
# Jinja Template - 값전달

- template/index\_user.html

```
<html>
<head>
<meta charset='utf-8'>
<title>플라스크월드</title>
<style type="text/css">
    #wrap {
        text-align:center;
        background:darkcyan;
        color:aliceblue;
        width:400px;
        margin:50px auto;
        padding:20px
    }
</style>
</head>
```

```
<body>
    <div id="wrap">
        <h2>Welcome Flask World : {{ user }}</h2>
    </div>
</body>
</html>
```

# Jinja Template - 값전달



# Jinja Template - include

- 파이썬 실행 파일

```
from flask import Flask, render_template
```

```
app = Flask(__name__)
```

```
@app.route('/')  
def home():
```

```
    return render_template('include_main.html')
```

```
if __name__ == '__main__':  
    app.run(debug=True)
```

Sub.html

확인하기 :

<http://127.0.0.1:5000> -> include\_main.html

# Jinja Template - include

- /template/include\_main.html

```
<html>
<head>
<meta charset='utf-8'>
<title>플라스크월드</title>
  <style type="text/css">
    #wrap {
      text-align:center;
      background:red;
      color:aliceblue;
      width:400px;
      margin:50px auto;
      padding:20px
    }
    ul>li { display:inline;padding:10px;
      border:1px solid black}
  </style>
</head>
```

# Jinja Template - include

- /template/include\_main.html

```
<body>
  <!-- sub.html 인클루드 -->
  {% include "sub.html" %}
  <div id="wrap">
    <h2>Welcome Flask World</h2>
  </div>
</body>
</html>
```

# Jinja Template - include

- /template/sub.html

```
<ul>
  <li><a href="#">menu1</a></li>
  <li><a href="#">menu2</a></li>
  <li><a href="#">menu3</a></li>
</ul>
```

Sub.html

확인하기 :

<http://127.0.0.1:5000> -> include\_main.html

# Response

- 파이썬 실행 파일

```
from flask import Flask, render_template, request  
app = Flask(__name__)
```

```
@app.route('/')  
def form():  
    return render_template('form_submit.html')
```

```
#form action  
@app.route('/hello', methods=['POST'])  
def action():  
    firstname = request.form['firstname']  
    lastname = request.form['lastname']  
    email = request.form['email']  
    return render_template('form_result.html', firstname=firstname,  
                           lastname=lastname, email=email)
```

```
if __name__ == '__main__':  
    app.run(debug=True)
```



# Response

- /template/form\_submit.html

```
<!doctype html>
<html lang="ko">
<head>
<meta charset="UTF-8">
<title>form submit</title>
</head>
<body>
  <h3>Flask Request Example</h3>
  <form action="/hello" method="post">
    First name:<br>
    <input type="text" name="firstname">
    <br> Last name:<br>
    <input type="text" name="lastname" <br>
    Email:<br> <input type="text" name="email">
    <br><br> <input type="submit" value="Submit">
  </form>
</body>
</html>
```

# Response

- /template/form\_result.html

```
<!doctype html>
<html lang="ko">
<head>
<meta charset="UTF-8">
<title>Rusult</title>
</head>
<body>
  <h4>반갑습니다.</h4>
  <strong>User infomation</strong><br>
  Fistname : {{ firstname }}<br>
  Lastname: {{ lastname }} <br>
  Email: {{ email }} <br><br>
  <button type="button"
onclick="location.href='/'">돌아가기 </button>
</body>
</html>
```

# Response

- /template/form\_result.html

## Flask Request Example

First name:

Last name:

Email:

반갑습니다.

## User information

Fistname : 홍

Lastname: 길동

Email: test@gmail.com

# 조건문 사용하기

- 실행되는 파이썬 파일에서 전달되는 값에 따라 조건문을 실행하여 출력한다.

```
{% if 조건1}
```

```
실행문1
```

```
{% elif 조건2 %}
```

```
실행문2
```

```
{% else %}
```

```
실행문3
```

```
{% endif %}
```

```
{% if 100 >= score >= 90 %}
```

```
<span>You are genius</span>
```

```
{% elif 90 > score >= 80 %}
```

```
<span>You are top student</span>
```

```
{% else %}
```

```
<span>Fail</span>
```

```
{% endif %}
```

# 조건문 사용하기

- 실행되는 파이썬 파일에서 전달되는 값에 따라 조건문을 실행하여 출력한다.

```
from flask import Flask, render_template
```

```
app = Flask(__name__)
```

```
@app.route('/')  
def home():
```

```
    return render_template('control_main.html')
```

```
@app.route('/controlif/<int:score>')
```

```
def control_if(score):
```

```
    return render_template('control_if.html',score=score)
```

```
if __name__ == '__main__':
```

```
    app.run(debug=True)
```

# 조건문 사용하기

- / : 실행파일

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
</head>
<body>
<h1>Control_main</h1>
<ul>
<li><a href="/controlif/30">if 문으로 이동하기 1</a></li>
<li><a href="/controlif/80">if 문으로 이동하기 2</a></li>
<li><a href="/controlif/95">if 문으로 이동하기 3</a></li>
</ul>
</body>
</html>
```

# 조건문 사용하기

- /controlif/숫자 : 결과값 출력파일

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
</head>
<body>
<h1>Control if</h1>
<ul>
```

```
<li>Score - {{score}}</li>
<li>
    {% if 100 >= score >= 90 %}
    <span>You are genius</span>
    {% elif 90 > score >= 80 %}
    <span>You are top student</span>
    {% else %}
    <span>Fail</span>
    {% endif %}
</li>
```

```
<li><a href="/">Home</a></li>
</ul>
</body>
</html>
```

# 조건문 사용하기 - 퀴즈

- 입력받은 과목의 점수의 총점과 평균 출력.
- 70점 이상시 진급. 그렇지 않다면 다음기회에 도전 출력

## Control\_main2 - form

국어 56

영어 55

수학 100

과학 100

전송

## Score Result

- 국어 56
- 영어 100
- 수학 55
- 과학 100
- 총점 311
- 평균 77.75

**진급**



# 조건문 사용하기 - 퀴즈

## 파이썬 실행파일

```
from flask import Flask, render_template, request
```

```
app = Flask(__name__)
```

```
@app.route('/')  
def home():
```

```
    return render_template('control_main2.html')
```

```
@app.route('/score_result', methods=['POST'])
```

```
def action():
```

```
    kor = request.form['kor']
```

```
    math = request.form['math']
```

```
    eng = request.form['eng']
```

```
    sci = request.form['sci']
```

```
    tot = int(kor)+int(math)+int(eng)+int(sci)
```

```
    avg = tot/4
```

```
    return render_template('score_result.html', kor=kor, eng=eng,  
                           math=math, sci=sci, tot=tot, avg=avg)
```

```
if __name__ == '__main__':
```

```
    app.run(debug=True)
```

# 조건문 사용하기 - 퀴즈

- / : html 실행파일

```
!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
</head>
<body>
<h1>Control_main2 - form</h1>
<form action="/score_result" method="POST">
<label>국어 <input type="text" name="kor"></label>
<label>영어 <input type="text" name="eng"></label>
<label>수학 <input type="text" name="math"></label>
<label>과학 <input type="text" name="sci"></label>
<button>전송</button>
</form>
</body>
</html>
```

# 조건문 사용하기 - 퀴즈

- /score\_result : score\_result.html

```
<h1>Score Result</h1>
<ul>
<li>국어 {{kor}}</li>
<li>영어 {{math}}</li>
<li>수학 {{eng}}</li>
<li>과학 {{sci}}</li>
<li>총점 {{tot}}</li>
<li>평균 {{avg}}</li>
</ul>
```

```
<p>
{% if avg>=70 %}
진급
{% else %}
다음기회에 도전
{% endif %}
</p>
```

# 반복문 사용하기

- 리스트안의 아이템 출력하기

```
{% for 리스트변수 in 리스트 %}
```

실행문

```
{% endfor %}
```

```
{% for i in fruits %}
```

```
<li>{{i}}</li>
```

```
{% endfor %}
```

# 반복문 사용하기

- 파이썬 실행파일

```
from flask import Flask, render_template
```

```
app = Flask(__name__)
```

```
@app.route('/')  
def home():
```

```
    fruits = ['바나나', '포도', '수박', '오렌지', '체리']
```

```
    return render_template('for_list.html', fruits = fruits)
```

```
if __name__ == '__main__':
```

```
    app.run(debug=True)
```

# 반복문 사용하기

- / : for\_list.html

```
<div id="wrap">  
<h1>우리 집 냉장고에는</h1>  
<ul>  
  {% for i in fruits %}  
    <li>{{i}}</li>  
  {% endfor %}  
</ul>  
</div>
```

# 반복문 사용하기 – range

- 파이썬 실행파일

```
from flask import Flask, render_template  
app = Flask(__name__)
```

```
@app.route('/')  
def home():
```

```
    return render_template('for_range.html')
```

```
if __name__ == '__main__':  
    app.run(debug=True)
```

# 반복문 사용하기-range

- / : for\_range.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<style>
#wrap {
    width:500px;
    text-align: center;
    margin:20px auto
}
ul li {
    display:inline-block; padding:10px;
    border:1px solid red; background:yellow;
    margin:10px; width:20px
}
</style>
</head>
```



# 반복문 사용하기-range

- / : for\_range.html

```
<body>
<div id="wrap">
<h1>1~20까지 한줄에 5개씩 출력하기</h1>
<ul>
{% for i in range(1,21) %}
<li>{{i}}</li>
{% if i%5==0 %}
<br>
{% endif %}
{% endfor %}
</ul>
</div>
</body>

</html>
```

# 반복문 사용하기-range

- 퀴즈 - 특정단의 구구단을 출력하여라

```
from flask import Flask, render_template  
app = Flask(__name__)
```

```
@app.route('/')  
def home():  
    return render_template('for_range_quiz.html')
```

```
if __name__ == '__main__':  
    app.run(debug=True)
```

# 반복문 사용하기-range

- 퀴즈 - 특정단의 구구단을 출력하여라

```
<div id="wrap">
<h1>구구단 출력하기</h1>
<ul>
{% for i in range(1,10) %}
<li> 3 X {{i}} = {{ 3*i }}</li>
{% endfor %}
</ul>
</div>
```

# 반복문 사용하기-range

- 퀴즈 - 구구단 모두를 출력하여라

```
from flask import Flask, render_template
app = Flask(__name__)
```

```
@app.route('/')
def home():
    return render_template('for_range_quiz2.html')
```

```
if __name__ == '__main__':
    app.run(debug=True)
```



# 반복문 사용하기-딕셔너리

- 파이썬 실행파일

```
from flask import Flask, render_template
```

```
app = Flask(__name__)
```

```
@app.route('/')  
def home():
```

```
    dic = {'name': 'Jhon', 'nation': 'USA', 'age': 20}
```

```
    return render_template('for_dict.html', dic=dic)
```

```
if __name__ == '__main__':
```

```
    app.run(debug=True)
```

# 반복문 사용하기-딕셔너리

- / : for\_dict.html

```
<div id="wrap">
<h1>딕셔너리 출력하기</h1>
<ul>
{% for key,value in dic.items() %}
<li><span> {{key}} </span>: {{value}} </li>
{% endfor %}
</ul>
</div>
```

# 템플릿 상속

- 부모 html

```
<div id="container">
{% block content %}
{% endblock%}
</div>
```

- 자식 html

```
{% extends "부모html경로" %}
{% block content %}
    내용
{% endblock %}
```



# 템플릿 상속

- 파이썬 실행파일

```
from flask import Flask, render_template
```

```
app = Flask(__name__)
```

```
@app.route('/')  
def home():
```

```
    return render_template('home2.html')
```

```
@app.route('/about')  
def about():
```

```
    return render_template('about.html')
```

```
@app.route('/site')  
def site():
```

```
    return render_template('site.html')
```

```
@app.route('/gallery')  
def gallery():
```

```
    return render_template('gallery.html')
```

```
if __name__ == '__main__':
```

```
    app.run(debug=True)
```

# 템플릿 상속

- /template/layout.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
</head>
<body>
<div id="wrap">
  <h1>Inheritance</h1>
  <nav>
    <ul>
      <li><a href="/">Home</a></li>
      <li><a href="/about">About</a></li>
      <li><a href="/site">Site</a></li>
      <li><a href="/gallery">Gallery</a></li>
    </ul>
  </nav>
```

```
<div id="container">
  {% block content %}
  {% endblock%}
</div>
```

```
</div>
</body>
</html>
```

# 템플릿 상속

- /template/home2.html

```
{% extends "layout.html" %}
```

```
{% block content %}
```

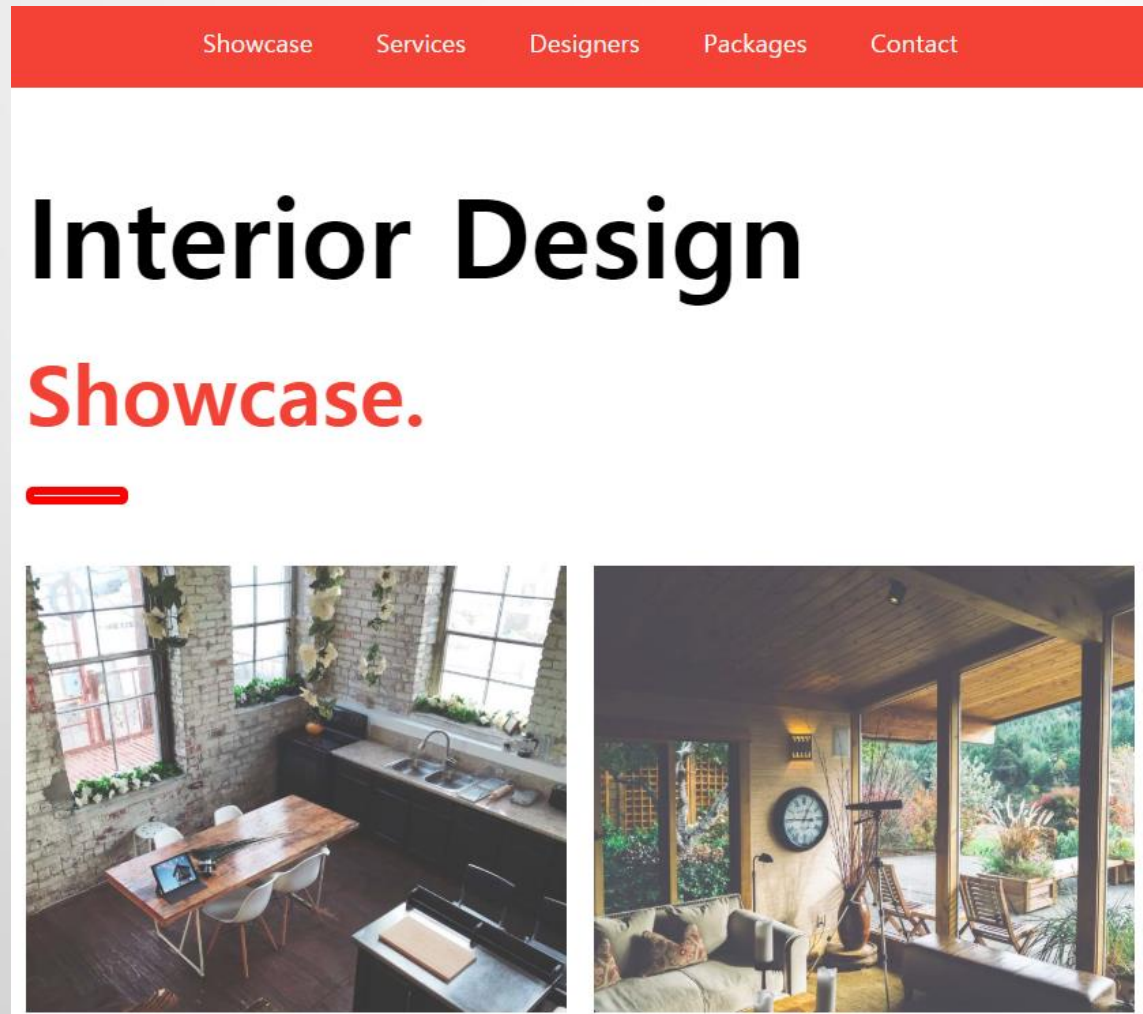
```
<h2>Home</h2>
```

```
<p>시작페이지입니다.</p>
```

```
{% endblock %}
```

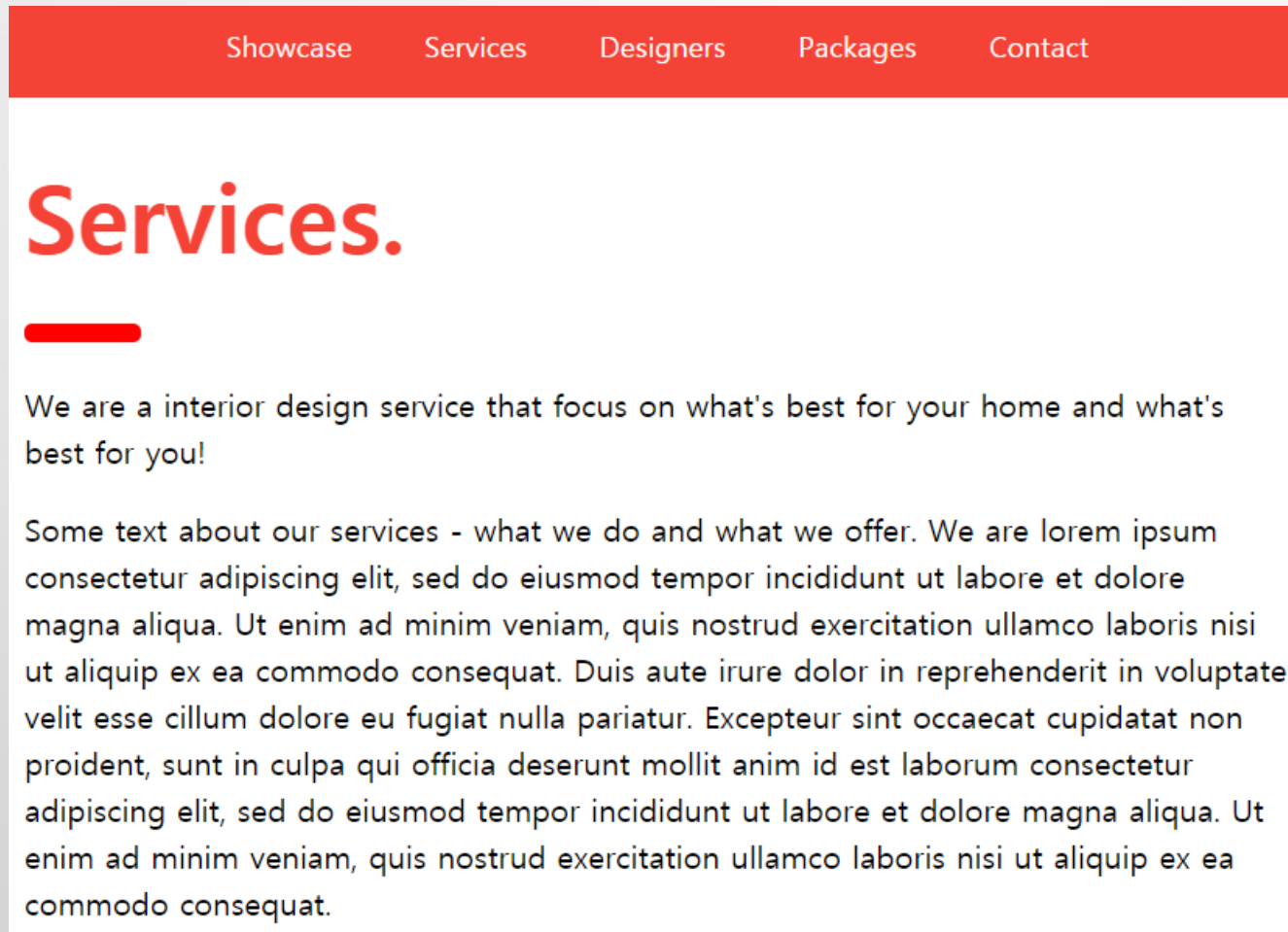
# 템플릿 상속 퀴즈

- /template/layout\_quiz.html



# 템플릿 상속 퀴즈

- /template/layout\_quiz.html



# 템플릿 상속 퀴즈

- /template/layout\_quiz.html

Showcase Services Designers Packages Contact


## Designers.

---


The best team in the world.

We are lorem ipsum consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.


**Our designers are thoughtfully chosen:**



John Doe



Jane Doe



Mike Ross

# 템플릿 상속 퀴즈

- /template/layout\_quiz.html

[Showcase](#) [Services](#) [Designers](#) [Packages](#) [Contact](#)

## Packages.

Some text our prices. Lorem ipsum consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure

Basic	Pro
Floorplanning	Floorplanning
10 hours support	50 hours support
Photography	Photography
20% furniture discount	50% furniture discount
Good deals	GREAT deals

# 템플릿 상속 퀴즈

- /template/layout\_quiz.html

Showcase Services Designers Packages Contact

## Contact.

---

Do you want us to style your home? Fill out the form and fill me in with the details :)  
We love meeting new people!

Name

Email

Message

Send Message



# DB 연결

- DB 접속 - 리스트 생성 - html로 전달

```
import pymysql
from flask import Flask, render_template
conn = pymysql.connect(host='localhost',
                        port=3300, user='root',
                        passwd='1234',
                        db='pythondb',
                        charset='utf8')

cursor = conn.cursor() #커서 생성
cursor.execute('select * from users;')

# 조회한 데이터 불러오기
result_list = cursor.fetchall()
# 데이터 출력하기 - 터미널창
for i in result_list:
    print(i)
```

# DB 연결

- DB 접속 - 리스트 생성 - html로 전달

```
# Flask 연동
app = Flask(__name__)

@app.route('/')
def home():
    return render_template('home.html', result=result_list)

if __name__ == '__main__':
    app.run(debug=True)

# DB 종료
conn.close()
```

# DB 연결

- /template/home.html

```
<html>
<head>
<meta charset='utf-8'>
<title>플라스크월드</title>
</head>
<body>
<div id="wrap">
<h2>전체 리스트 출력</h2>
  print({{ result }})
</div>
</body>
</html>
```

# DB 연결

- /template/home.html

```
<h2>회원정보 출력</h2>
<table border="1"
cellspacing="0">
<thead>
<tr>
    <td>번호</td>
    <td>아이디</td>
    <td>비밀번호</td>
    <td>이름</td>
    <td>가입일</td>
</tr>
</thead>
```

```
<tbody>
    {% for team in result %}
    <tr>
        <td>{{ team[0] }}</td>
        <td>{{ team[1] }}</td>
        <td>{{ team[2] }}</td>
        <td>{{ team[3] }}</td>
        <td>{{ team[4] }}</td>
    </tr>
    {% endfor %}
</tbody>
</table>
```

# DB - 입력

## ■ 실행파일

```
import pymysql
from flask import Flask, render_template, request

def result(num,contents,writer):
    print('*'*30)
    print('After:',num,contents,writer)
    conn = pymysql.connect(
        host='localhost',
        port=3300, user='root',
        passwd='1234',
        db='pythondb',
        charset='utf8')
    print('연결완료')
    cursor = conn.cursor() #커서 생성
    cursor.execute("insert into bbs (num,contents,writer) values
        (%s, %s,%s)" , (num, contents, writer))
    conn.commit()
```

# DB - 입력

## ■ 실행파일

```
# 조회한 데이터 불러오기
cursor.execute('select * from bbs;')
result_list = cursor.fetchall()
print(result_list)
# DB 종료
conn.close()
```

```
# Flask 연동
app = Flask(__name__)
```

```
@app.route('/')
def form():
    return render_template('form_bbs.html')
```

# DB - 입력

## ■ 실행파일

```
#form action
@app.route('/bbs_result', methods=['POST'])
def action():
    num = str(request.form['num'])
    contents = request.form['contents']
    writer = request.form['writer']
    print(num,contents,writer)
    result(num,contents,writer)
    return
    render_template('form_bbs_result.html',num=num,contents=contents,
writer=writer)

if __name__ == '__main__':
    app.run(debug=True)
```

# DB - 입력

- /template/form\_bbs.html

```
<!doctype html>
<html lang="ko">
<head>
<meta charset="UTF-8">
<title>form submit BBS</title>
</head>
<body>
  <h3>BBS Example</h3>
  <form action="/bbs_result" method="post">
    Num:<br>
    <input type="text" name="num" value="">
    <br> Contents : <br>
    <input type="text" name="contents" value="">
    <br> Writer : <br>
    <input type="text" name="writer" value="">
    <br> <br>
    <input type="submit" value="Submit">
  </form>
</body>
</html>
```



# DB - 입력

- /template/form\_bbs\_result.html

```
<!doctype html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>BBS Result</title>
</head>
<body>
  <p>
데이터가 입력되었습니다.<br>
{{num}},{{contents}},{{writer}}
</p>
<br><br>
<button type="button"
onclick="location.href='/'">돌아가기 </button>
</body>
</html>
```

# DB - 입력

- /template/form\_bbs\_result.html

**BBS Example**  
  
Num:  
  
  
Contents :  
  
  
Writer :