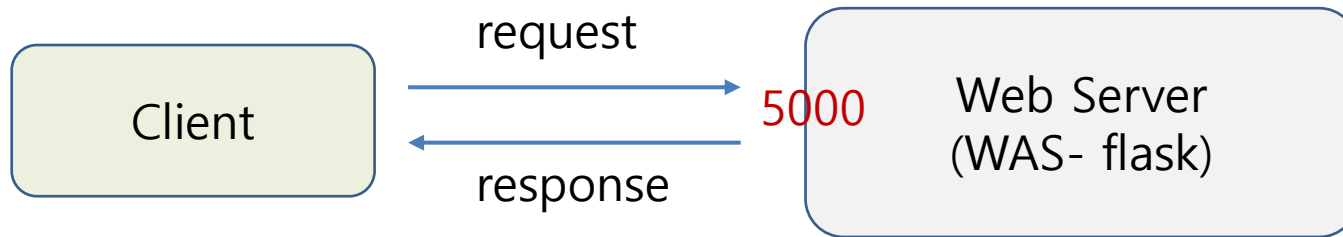


# 10장. 서버(Server) 및 네트워크



# 클라이언트 & 서버

- 요청(request) 및 응답(response)



**WAS(Web Application Server)** : 웹 애플리케이션 서버로 함수를 통한 제어, DB 연동 업무를 수행하고 클라이언트에 응답하는 역할을 한다. 파이썬의 플라스크(flask)와 장고(django), 자바의 스프링(spring)이 웹 프레임워크라 할 수 있다.

# flask(플라스크)

- **플라스크(flask)란?**

파이썬으로 제작된 마이크로 웹 프레임워크의 하나이며, 웹 sever를 만들고, 웹 애플리케이션을 제작할 수 있다.



# Flask

web development,  
one drop at a time

Flask 문서에 오신 것을 환영합니다. 시작하기 [설치](#) 한 다음에 대한 개요 얻을 [빠른 시작](#)을 . Flask를 사용하여 작지만 완전한 애플리케이션을 만드는 방법을 보여주는 더 자세한 [자습서](#) 도 있습니다 . 일반적인 패턴은 [Flask 용 패턴](#) 섹션에 설명되어 있습니다. 나머지 문서에서는 [API](#) 섹션 의 전체 참조와 함께 Flask의 각 구성 요소를 자세히 설명 합니다.

# flask(플라스크)

- 플라스크 설치

- flask documentation 검색 및 설치

## Install Flask

Within the activated environment, use the following command to install Flask:

```
$ pip install Flask
```

**pip install Flask**

# flask 웹서버 만들기

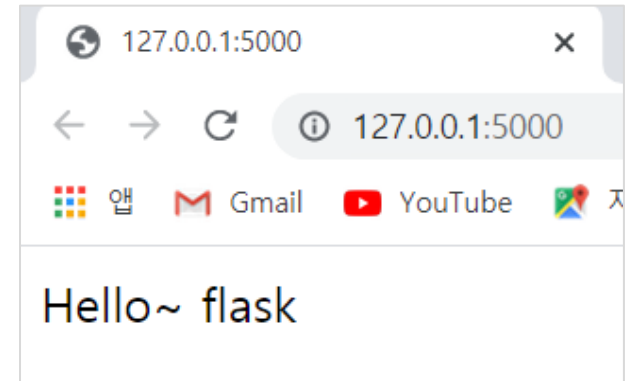
## ■ 웹 서버 만들기

```
from flask import Flask

app = Flask(__name__) # app - Flask 객체 생성

@app.route('/') # 라우트 ('/'는 루트 경로)
def index():
    return "Hello~ flask"

if __name__ == "__main__":
    app.run()
```



배포용이 아닌 개발용이다.  
WSGI(WebServer GateWay Interface)

\* Serving Flask app 'app'

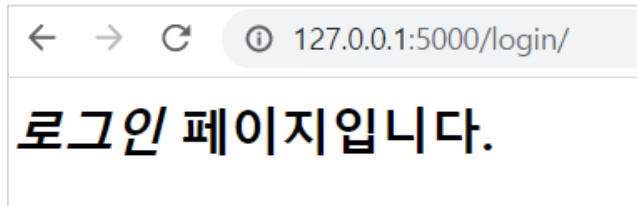
\* Debug mode: off

**WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.**

\* Running on http://127.0.0.1:5000

# flask 웹서버 만들기

## ■ 페이지 라우팅



```
from flask import Flask

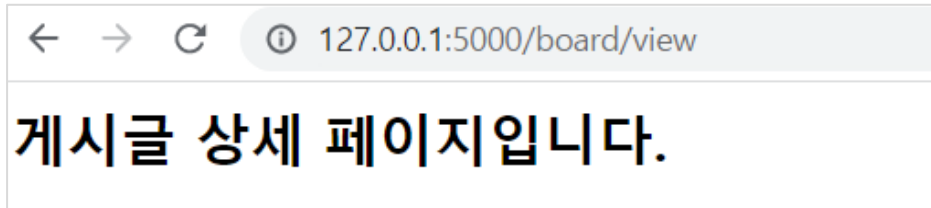
app = Flask(__name__) #app 객체 생성

@app.route('/') #http://127.0.0.1:5000/
def index():
    return "Hello~ Flask"

@app.route('/login') #http://127.0.0.1:5000/login
def login():
    return "<h2><i>로그인</i> 페이지입니다.</h2>"
```

# flask 웹서버 만들기

## ■ 페이지 라우팅



```
@app.route('/register')
def register():
    return "<h2>회원가입 페이지입니다.</h2>"

@app.route('/board/view')
def view():
    return "<h2>게시글 상세 페이지입니다.</h2>"
```

# flask 웹서버 만들기

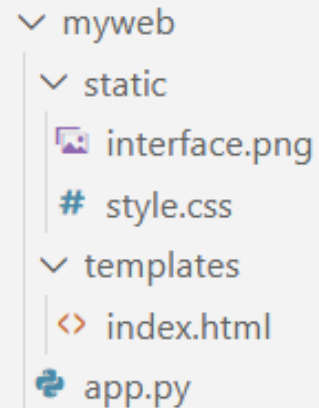
## ▪ Flask 웹 프레임워크

### 웹 프레임워크 구조

**templates** 폴더 -> html 파일

**static** 폴더 -> css, image 파일

**start\_app.py** -> 실행 파일



```

  myweb
  ├── static
  │   ├── interface.png
  │   └── style.css
  ├── templates
  │   └── index.html
  └── app.py

```



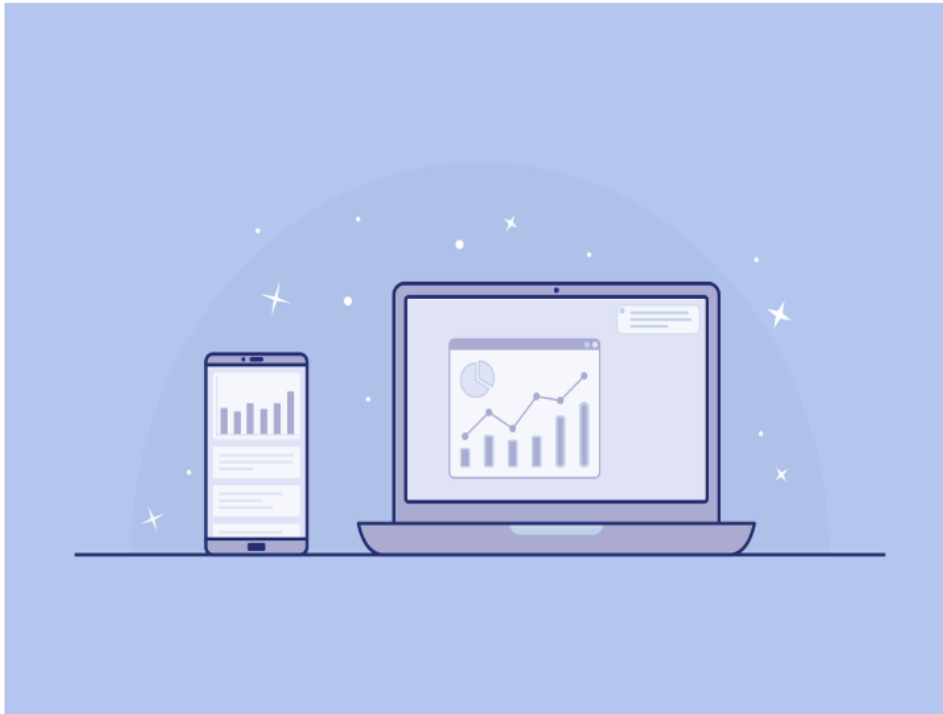
# flask 웹서버 만들기

## 웹 기초 기술

HTML - 웹 페이지의 구조를 담당

CSS - 웹페이지의 디자인을 담당

Javascript(자바스크립트) - 웹 페이지의 인터랙티브를 담당



[픽사베이로 이동](#)

# flask 웹서버 만들기

- 인덱스 페이지 서버에 띄우기

app.py

```
from flask import Flask, render_template

app = Flask(__name__) #app 객체 생성

@app.route('/') #http://127.0.0.1:5000/
def index():
    # return "<h1>Welcome~ My Home!!</h1>"
    return render_template('index.html') #인덱스 페이지 렌더링

app.run()
```

# flask 웹서버 만들기

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>인덱스 페이지</title>
  <link rel="stylesheet" href={{ url_for('static', filename='style.css')}}>
</head>
<body>
  <div id="container">
    <h1>웹 기초 기술</h1>
    <hr>
    <p>HTML - 웹 페이지의 구조를 담당</p>
    <p>CSS - 웹페이지의 디자인을 담당</p>
    <p>Javascript(자바스크립트) - 웹 페이지의 인터랙티브를 담당</p>

    <!-- 이미지 삽입 -->
    <img src={{ url_for('static', filename='interface.png')}}
alt="인터페이스" width="640" height="480"><br><br>

    <!-- 하이퍼 링크 -->
    <a href="https://pixabay.com/ko/" target="_blank">
      Pixabay로 이동</a>
  </div>
</body>
</html>
```

index.html

# flask 웹서버 만들기

- 인덱스 페이지 서버에 띄우기

```
/* sytle.css */  
  
/* 페이지 가운데 정렬, 위.아래 여백 30px */  
#container{ width: 80%; margin: 30px auto;}  
  
/* 제목 글자색 - 파랑 */  
#container h1{color: blue;}
```