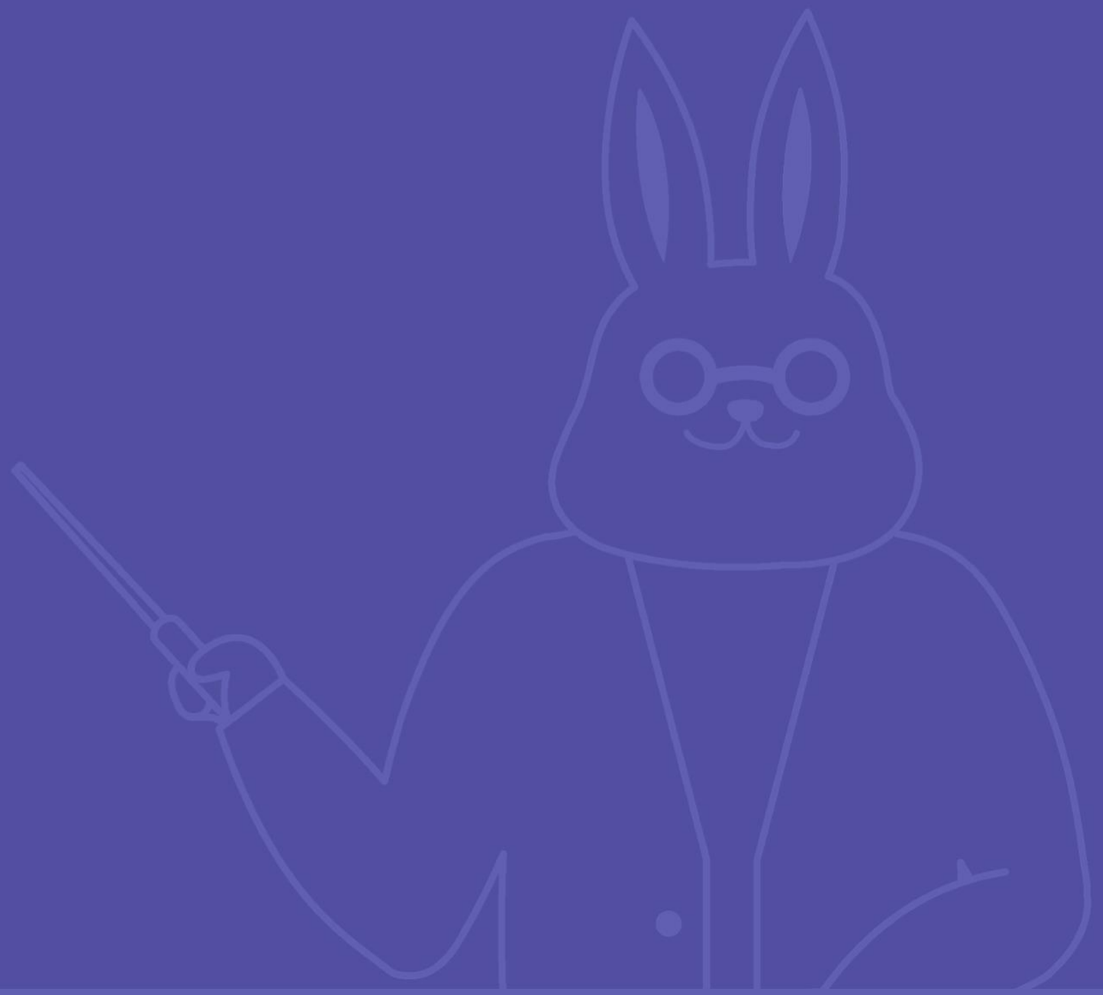
Flask 기초

01 Flask 시작하기



목차

01. 웹 서버의 동작 과정 이해하기

02. Flask Framework란?

03. Flask로 웹 서버 만들기

04. URL을 연결하고 데이터를 화면에 나타내기

05. REST API란?

06. HTTP Method 사용하기

🎯 수강목표

1. 웹 서비스의 흐름을 안다.

포털 사이트에 나오는 화면과 데이터들이 이동하는 과정과 순서들을 이해 할 수 있습니다.

2. 웹 프레임워크의 종류를 알고 Flask를 사용할 줄 안다.

웹 서비스를 만들기 위한 웹 프레임워크의 종류를 배우고, 그 중 Flask를 실행하는 방법을 배웁니다.

3. 서버 통신의 방법을 알고 사용 할 줄 안다.

서버가 어떤 방법으로 통신하는지 알고, 직접 사용해 봅니다.

01

웹 서버의 동작 과정 이해하기

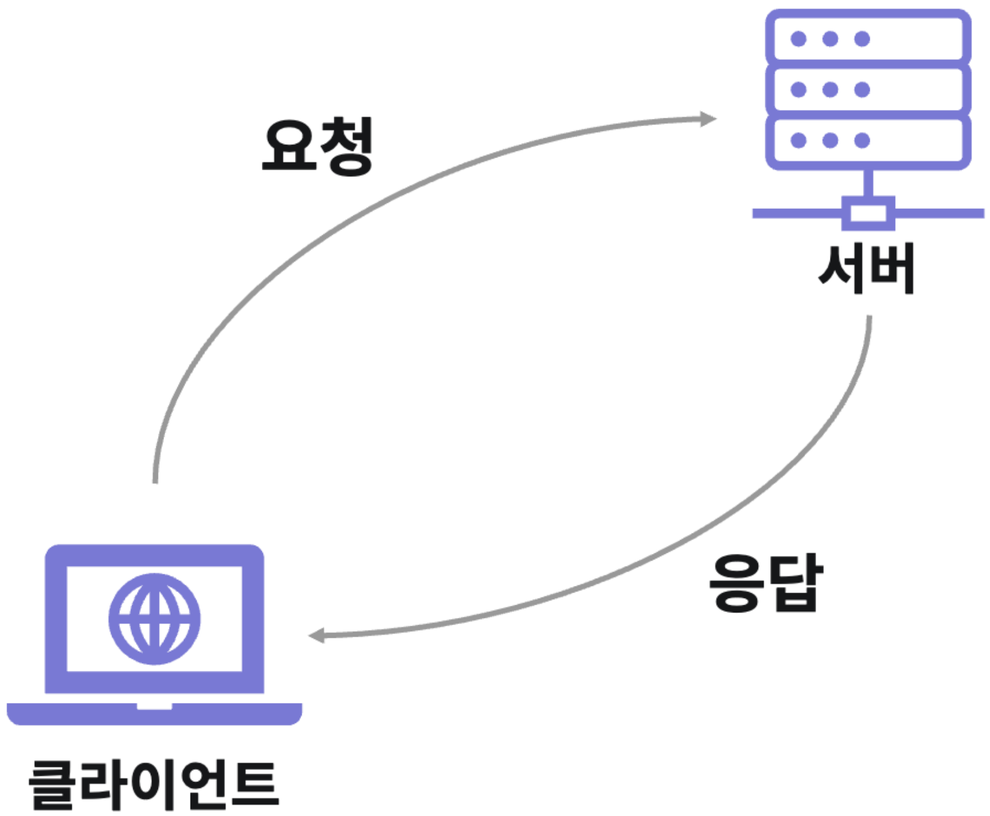
웹 서버가 동작하는 과정을 알아보자

우리가 흔히 정보들을 접하고 사용하는 **많은 서비스는**

'웹 서버'를 통해 우리에게 제공되고 있습니다.

이 **웹 서버의 동작 과정**들을 간단히 알아보도록 하겠습니다.

웹 서버의 동작 과정



클라이언트 : 사용자 / 내 컴퓨터 / 크롬 등 …

서버 : 클라이언트로부터 요청을 받아서 처리해 주고, '응답'으로 데이터를 돌려주는 곳

클라이언트가 서버로 요청을 하고, 요청을 받은 서버는 **요청에 해당하는 데이터를 응답**으로 돌려주게 됩니다.

서버의 응답으로 클라이언트가 보게 되는 데이터는 **HTML, JSON, XML 등 다양한 형태**가 될 수 있습니다.

요청과 응답은 어떻게 하면 될까?

원하는 정보를 아무렇게나 요청하고 받는 것이 아니라

미리 약속한 규칙을 통해서 **요청**하면

정해진 형태의 데이터로 **응답**을 합니다.

웹 서버의 동작 예시 - 은행에서 업무 보기

은행원과 음식과 관련된 대화하지 않습니다.

은행 업무와 관련된 대화를 합니다.

정해진 형식, 데이터의 통로! API

https://www.naver.com

https://www.google.com

네이버와 구글 각각 페이지의 접속 주소는
미리 약속된 주소라고 할 수 있습니다.

미리 약속된 방법으로 요청한 후
해당하는 페이지를 보게 되는 것입니다.

정해진 방식으로 데이터의 통로 역할을 하는
것이 바로 **API** 라고 할 수 있습니다.

02

Flask Framework란?

Framework는 무엇인가?

하나의 결과물을 만들기 위해서 제공하는 '틀'

미리 작성되어 있는 함수 (라이브러리) 이상의 기능을 제공

그렇다면 Flask Framework란?

Flask는 Python을 사용해서

웹 서버를 만들 수 있게 도와주는 Web Framework

파이썬을 사용하는 Framework

파이썬을 사용하는 Framework는 여러 가지가 존재합니다.

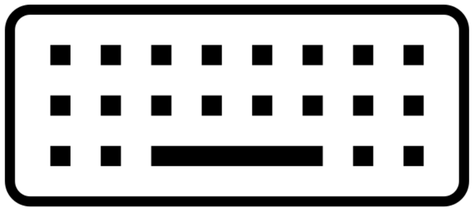
다량의 기능을 미리 제공
- Django (사용 할 수 있는 기능이 많다)

기본적인 기능만 제공
- Flask, Pyramid, Bottle (개발이 자유롭다)

각각의 Framework마다 장/단점이 있습니다.

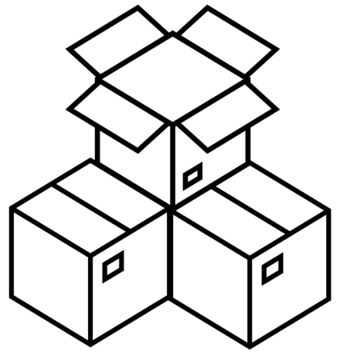
Flask 프레임워크의 장점

나만의 서버를 쉽게
작성 할 수 있다.



app.run()

간단한 코드로 빠르게
실행 할 수 있다.



원하는 기능을 유연하게
확장하기 편리하다.

03

Flask 웹 서버 만들기

Flask 웹 서버를 만들어 보자

웹 서버를 만드는 방법은 복잡할 것 같지만

Flask Framework를 사용해서

간단한 코드만으로 서버를 만들 수 있습니다.

Flask Simple Web Server 만들기

app.py

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

@app.route("/")
def elice():
    return "hello elice"

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

- Flask 패키지에서 Flask를 import합니다.
- @app.route() 는 서버에 접속 할 수 있는 url을 만들어 줍니다.
- @app.route("/") 아래의 def elice()라는 함수는 app.route()의 url에서 실행할 함수입니다.
- If __name__ == "__main__" 은, 파일 이름이 main일 때만 app.run()이 실행되도록 합니다.

04

URL을 연결하고 데이터를 화면에 나타내기

URL을 어떻게 연결할까?

이전 실습에서 보았던 내용과 같이 **app.route()는 url을 만들어 줍니다.**

app.route()와 이어져 있는 함수를 사용해서

HTML과 **JSON** 형식의 데이터를 전달해 보겠습니다.

JSON 형식의 데이터 나타내기

app.py

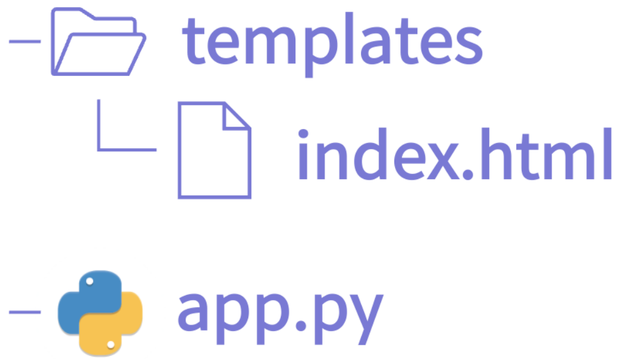
```
from flask import Flask,jsonify
app = Flask(__name__)

@app.route("/")
def elice_json():
    my_data = {"name":"elice"}
    return jsonify(my_data)

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

- Flask 패키지에서 Flask와 jsonify를 import 합니다.
- jsonify 라는 함수는 jsonify() 안에 있는 내용을 화면에 전달합니다.
- @app.route("/") 아래의 def elice_json() 함수는 app.route()의 url에서 실행할 함수이고, {"name": "elice"} 라는 데이터를 화면에 전달 해 줍니다.

HTML 형식의 데이터 나타내기 – 파일 준비하기



- html을 화면에 전달하기 위해서는 html 파일이 필요합니다.
- html 파일은 왼쪽의 그림처럼 templates라는 폴더 아래에 넣어 주어야 합니다.
- templates 폴더에 html파일을 넣어 놓으면 Flask가 자동으로 찾아서 연결해 줍니다.

HTML 형식의 데이터 나타내기 – templates 준비하기

templates/index.html

```
<html>
    <head>
        <title> 나의 첫 html </title>
    </head>
    <body>
        <h4> html파일 띄우기 </h4>
    </body>
</html>
```

- Flask를 통해 가져올 index.html을 templates폴더 아래에 생성합니다.
- 왼쪽의 코드와 같이 간단한 html을 작성합니다.

HTML 형식의 데이터 나타내기 – app.py 준비하기

app.py

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

@app.route("/")
def elice_html():
    return render_template("index.html")

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

- Flask 패키지에서 Flask와 render_template를 import 합니다.
- render_template 라는 함수는 templates라는 폴더 안의 html파일을 불러와 주는 역할을 합니다.
- templates폴더 내의 html 파일의 이름과 render_template()안의 이름이 같아야 화면에 잘 전달할 수 있습니다.
- 우리가 만들었던 index.html을 render_template에 넣어서 elice_html 함수의 return에 넣어주세요.

여러 가지 url 연결하기

app.py

```
@app.route("URL주소")
```

- "URL 주소" 라고 되어있는 부분은 우리가 만들어 줄 수 있는 url의 주소입니다.

- 1개의 @app.route는 1개의 함수와 연결될 수 있습니다.

- @app.route를 여러 개 사용해서 다양한 url을 만들 수 있습니다.

여러 가지 url 연결하기

app.py

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

@app.route("/")
def elice():
    return jsonify("home")

@app.route("/admin")
def elice_admin():
    return jsonify("admin page")
```

```
@app.route("/student")
def elice_student():
    return jsonify("student Page")

@app.route("/student/<name>")
def elice_user(name):
    user = {"name":name}
    return jsonify(user)

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

05

REST API란?

REST API의 개념

HTTP URL을 통해 데이터의 자원을 표현하고

HTTP Method를 통해서 데이터를 다루는 방법을 의미합니다.

Database, 이미지, 텍스트 등의 **다양한 데이터에 적용** 할 수 있습니다.

REST API의 개념

REST API는 아래와 같이 표현 할 수 있습니다.

- HTTP URI(Uniform Resource Identifier)를 통해 자원을 명시하고, HTTP Method(POST, GET, PUT, DELETE)를 통해 해당 자원에 대한 CRUD Operation을 적용하는 것

- 다양한 클라이언트가 생겨남에 따라서 REST API가 필요합니다. (다양한 곳에서 통신해야 하기 때문)
- REST API는 메시지가 의도하는 바를 URL에서 나타내므로, 쉽게 기능을 파악 할 수 있습니다.
- HTTP 표준 프로토콜에 따르는 플랫폼에서 사용 가능합니다.
- 서버와 클라이언트의 구분을 명확하게 할 수 있습니다.
- REST API의 표준이 존재하지 않는 단점이 있습니다.

CRUD

Create	생성
Read	조회
Update	수정
Delete	삭제

HTTP Method

GET	조회
POST	생성 / 수정 / 삭제

06

HTTP Method 사용하기

HTTP Method의 개념 – GET / POST

GET과 **POST**는 HTTP Method 중 하나입니다.

GET은 **데이터를 URL 뒤에 ?와 함께 사용**하고

POST는 **특정 양식(form)에 데이터를 넣어 전송**하는 방법입니다.

HTTP Method의 예시 – GET / POST

GET 방식의 예시

http://사이트의_주소?데이터=123

GET의 예시 -> url 뒤에 물음표를 붙여서 데이터를 서버에 전송

네이버 영화 페이지
-> https://movie.naver.com/movie/bi/mi/basic.nhn

네이버 영화의 해리포터와 마법사의 돌 페이지
-> https://movie.naver.com/movie/bi/mi/basic.nhn?code=30688

HTTP Method의 예시 – GET / POST

POST방식의 예시

http://사이트의_주소

POST의 예시 -> 일정한 양식에 담아 데이터를 숨겨서 서버에 전송

네이버 로그인 페이지 url
-> https://nid.naver.com/nidlogin.login

네이버 로그인 데이터 전송 url
-> https://nid.naver.com/nidlogin.login

app.route()에 method 옵션 사용하기

app.py

```
@app.route('url1', methods=["GET"])
@app.route('url2', methods=["POST"])
@app.route('url3', methods=["GET", "POST"])
```

- app.route()에 methods라는 옵션을 추가해서 해당하는 HTTP Method만 사용 할 수 있도록 적용 할 수 있습니다.

- url1, url2 는 각각 GET, POST 메서드만 사용이 가능합니다.

- url3은 GET, POST 모두 사용이 가능합니다.

GET 요청만 사용하기

app.py

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

@app.route("/" , methods=["GET"] ) # URL 뒤에 ?name=elice를 넣어 GET 요청을 합니다.
def elice():
     name = request.args.get['name']
    result = "hello. " + name
    return result

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

POST 요청만 사용하기 - templates/index.html 만들기

index.html

```
<html>
<body>
    <form action='/login' method='post'>
        <p>
        아이디 : <input type='text' name='id'>
        </p>
        <p>
        비밀번호 : <input type='password' name='pwd'>
        </p>
        <button type='submit' />
    </form>
</body>
</html>
```

여러 가지 url 연결하기

app.py

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

@app.route("/" , methods=["GET"] )
def elice():
    return render_template("index.html")
```

```
@app.route("/login" , methods=["POST"] )
def elice_post():
    id= request.form['id']
    pwd= request.form['pwd']
     if id == 'elice' and pwd = '1234':
        return 'Hi! Elice!"
    else:
        return "ERROR"

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

크레딧

/* elice */

코스 매니저
이재성

콘텐츠 제작자
이바울, 이재성

강사
이바울

감수자
이바울

디자이너
강혜정

연락처

TEL
070-4633-2015

WEB
https://elice.io

E-MAIL
contact@elice.io