**Flask 요약**

* Flask는 웹 응용을 쉽고 간결하게 개발할 수 있는 마이크로 웹 프레임워크임으로 파일 렌더링 및 데이터 전달 기능을 제공함
* Flask 는 웹응용 개발을 위해 필요한 경량화 패키지는 flask 를 import 하며, 패키지 flask의 Flask 클래스를 사용하여 응용을 생성함

from flask import Flask

* Flask 객체를 app에 할당

app = Flask(\_\_name\_\_)

* 주어진 URL규칙에 따라 등록된 함수를 수행하도록 정의하는 URL 라우팅을 사용하며, app 객체를 이용해 라우팅 경로를 설정

|  |
| --- |
| @app.route("/") |
| def hello(): |
| return "Hello World!" |

@app.route('/', methods=['GET'])

def root():

return app.send\_static\_file('index.html')

@app.route('/')

def static\_prox(path):

return app.send\_static\_file(path)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| * 함수 app.run()을 사용하여 Flask 응용을 실제수행 시킴   **if** \_\_name\_\_ == '\_\_main\_\_':  app.run()     * URI를 변수로 사용, 변수에 데이터 타입도 줄 수 있으며, 데이터 타입이 없으면 문자열로 인식함 * int 이외에 float 도 데이터 타입으로 줄 수 있음   @app.route("/message/<int:message\_id>")  def get\_message(message\_id):  return "message id: " + message\_id   * /data URL 요청시 자료를 JSON으로 변환하여 제공함   @app.route('/data') **def data**(): data = {"names": ["John", "Jacob", "Julie", "Jennifer"]} **return**jsonify(data)   * [Flask](http://flask.pocoo.org/)에서는 [request](http://flask.pocoo.org/docs/api/#flask.request)라는 객체를 통해 요청 파라매터 처리 기능을 제공함  |  |  |  |  | | --- | --- | --- | --- | | from flask import request | | | | | @app.route("/reverse") | | | | def reverse(): | | | Message = request.values["message"] | | | | | | return "".join(reversed(message)) |      * URL 요청에 대한 응답으로 단순한 문자열을 사용하여 복잡한 응용개발을 지원하기 위해 Template 를 활용함, 함수에서 HTML 문자열을 반환하는 대신에, render\_template() 함수를 사용해서 HTML 파일을 랜더링함, 이를 위해 Flask 클래스와 함수 render\_template를 가져옴   from flask import Flask, render\_template   * /hello 또는 /hello/<name> 요청에 대해 hello.html이라는 템플릿 파일을 읽어서 처리하도록 함, Flask는 templates 폴더에서 hello.html 파일을 위치하도록 함  |  |  | | --- | --- | | from flask import render\_template | | |  | |  | | @app.route('/hello/') | | | @app.route('/hello/<name>') | | | def hello(name=None): | | | return render\_template('hello.html', name=name) |  * (실습) lights.py 에서는 URL / 요청 후 default() 함수를 호출하여 GPIO 의 핀 상태를 읽어 변수 pin을 통해 HTML lights.html로 전달함   @app.route("/")  def default():  # read pin state  status = GPIO.input(pin)  if status == 0:  return render\_template ('lights.html',pin=1)  else:  return render\_template ('lights.html',pin=0)  <html>  <body>  <h2>Lights on RPi Webserver</h2>  <!-- check pin state -->  {% if pin == 0 %}  <p>Light is <b>On</b></p>  <input type="button" onclick="window.location='/off';" value="Turn off"/>  {% else %}  <p>Ligh is <b>Off</b></p>  <input type="button" onclick="window.location='/on';" value="Turn on"/>  {% endif %}  </body>  </html> |
|  |