Flask

web development,
one drop at a time

IndiFlex 시니어코딩

Flask

web development,
one drop at a time

0

화면(UI) 작성

IndiFlex 시니어코딩

Flask

web development,
one drop at a time

1

Python Flask

IndiFlex 시니어코딩

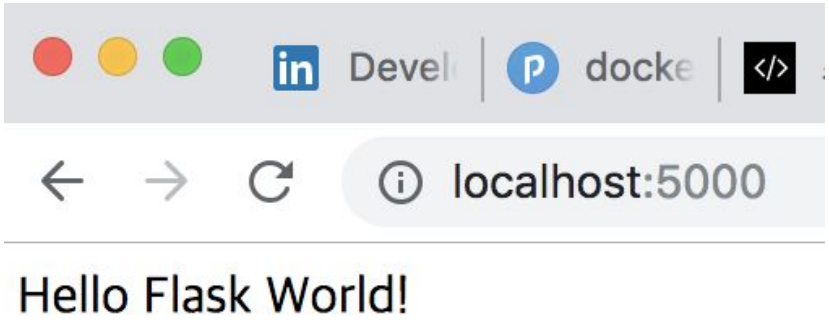
Install Flask & Setup Flask Project

```
# Install flask module
$> pip install flask

# Setup Flask Project
$> mkdir webapp
    /flaskapp (helloflask)
        - /static
            - /css
            - /images
            - /js
        - /templates
        - __init__.py

    start_helloflask.py
```

Hello Flask World

```
# flaskapp/__init__.py
from flask import Flask

app = Flask(__name__)

@app.route("/")
def helloworld():
    return "Hello Flask World!"


# ../start_helloflask.py
from helloflask import app

app.run(host='0.0.0.0')
```

```
* Serving Flask app "hif" (lazy loading)
* Environment: production
  WARNING: Do not use the development server in a production en
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
```



Global Object : g

```
from flask import g

app = Flask(__name__)
app.debug = True      # use only debug!!

@app.before_request
def before_request():
    print("before_request!!!")
    g.str = "한글"

@app.route("/")
def helloworld():
    return "Hello World!" + getattr(g, 'str', '111')
```

Response Objects

```
from flask import Response, make_response

# response_class
custom_res = Response("Custom Response", 200, {'test': 'ttt'})
return make_response(custom_res)

# str : Simple String (HTML, JSON)
return make_response("custom response")
```

Response Objects (Cont'd) : WSGI

```
from flask import make_response

# WSGI(WebServer Gateway Interface)
@app.route('/test_wsgi')
def wsgi_test():
    def application(environ, start_response):
        body = 'The request method was %s' % environ['REQUEST_METHOD']
        headers = [ ('Content-Type', 'text/plain'),
                    ('Content-Length', str(len(body))) ]
        start_response('200 OK', headers)
        return [body]

    return make_response(application)
```

Request Event Handler

```
@app.before_first_request
def …

@app.before_request
def …

@app.after_request
def …(response)
    return response

@app.teardown_request
def …(exception)

@app.teardown_appcontext
def …(exception)
```

Routing

```
@app.route('/test')
def …

@app.route('/test', methods=[ 'POST', 'PUT' ])
def …

@app.route('/test/<tid>')
def test3(tid):
    return "tid is %s" % tid

@app.route('/test', defaults={'page': 'index'})
@app.route('/test/<page>')
def xxx(page):

@app.route('/test', host='abc.com')
@app.route('/test', redirect_to='/new_test')
```

Routing (Cont'd) : subdomain

```
app.config['SERVER_NAME'] = 'local.com:5000'

@app.route("/")
def helloworld_local():
    return "Hello Local.com!"

@app.route("/", subdomain="g")
def helloworld():
    return "Hello G.Local.com!!!"
```

Request Parameter

```
# MultiDict Type
...get('<param name>', <default-value>, <type>)
methods: get, getlist, clear, etc

# GET
request.args.get('q')

# POST
request.form.get('p', 123)

# GET or POST
request.values.get('v')

# Parameters
request.args.getlist('qs')
```

Request Parameter Custom Function Type

```
from datetime import datetime, date

# request 처리 용 함수
def ymd(fmt):
    def trans(date_str):
        return datetime.strptime(date_str, fmt)
    return trans


@app.route('/dt')
def dt():
    datestr = request.values.get('date', date.today(), type=ymd('%Y-%m-%d'))
    return "우리나라 시간 형식: " + str(datestr)
```

request.environ

```
return ('REQUEST_METHOD: %(REQUEST_METHOD) s <br>'
        'SCRIPT_NAME: %(SCRIPT_NAME) s <br>'
        'PATH_INFO: %(PATH_INFO) s <br>'
        'QUERY_STRING: %(QUERY_STRING) s <br>'
        'SERVER_NAME: %(SERVER_NAME) s <br>'
        'SERVER_PORT: %(SERVER_PORT) s <br>'
        'SERVER_PROTOCOL: %(SERVER_PROTOCOL) s <br>'
        'wsgi.version: %(wsgi.version) s <br>'
        'wsgi.url_scheme: %(wsgi.url_scheme) s <br>'
        'wsgi.input: %(wsgi.input) s <br>'
        'wsgi.errors: %(wsgi.errors) s <br>'
        'wsgi.multithread: %(wsgi.multithread) s <br>'
        'wsgi.multiprocess: %(wsgi.multiprocess) s <br>'
        'wsgi.run_once: %(wsgi.run_once) s') % request.environ
```

request

```
request.is_xhr

request.url

request.path

request.endpoint

request.get_json()

app.config.update(MAX_CONTENT_LENGTH=1024*1024)
request.max_content_length
```

Response Object

```
from flask import Response

# Response Attributes
 - headers
 - status
 - status_code
 - data
 - mimetype

ex)
res = Response("Test")
res.headers.add('Program-Name', 'Test Response')
res.set_data("This is Test Program.")
res.set_cookie("UserToken", "A12Bc9")
```

Cookie

```
from flask import Response

# Cookie __init__ Arguments
 - key
 - value
 - max_age
 - expires
 - domain
 - path

ex)
res = Response("Test")
res.set_cookie("UserToken", "A12Bc9")

# other request
request.cookies.get('UserToken', 'default token')
```

Try This: Cookie

1) 다음 형태로 요청했을때 해당 key로 Cookie를 굽는 코드를 작성하시오.
   http://localhost:5000/wc?key=token&val=abc

2) 다음과 같이 요청했을때 해당 key의 Cookie Value를 출력하는 코드를 작성하시오.
   http://localhost:5000/rc?key=token

Session

```
from flask import session

app.secret_key = 'X1243yRH!mMwf'

OR

app.config.update(
    SECRET_KEY='X1243yRH!mMwf',
    SESSION_COOKIE_NAME='pyweb_flask_session',
    PERMANENT_SESSION_LIFETIME=timedelta(31)      # 31 days  cf. minutes=30
)

* Save to Memory, File or DB
```

Session (Cont'd)

```
from flask import session

@app.route('/setsess')
def setsess():
    session['Token'] = '123X'
    return "Session이 설정되었습니다!"

@app.route('/getsess')
def getsess():
    return session.get('Token')

@app.route('/delsess')
def delsess():
    if session.get('Token'):
        del session['Token']
    return "Session이 삭제되었습니다!"
```

Flask 2

web development,
one drop at a time

Templates

IndiFlex 시니어코딩

Templates (Jinja)

```
# Jinja2 Librury: Flask Template Engine  (http://jinja.pocoo.org)
# Types
String, XML, HTML, JSON, Image, Video, etc

# example
{% extends "application.html" %}
{% block body %}
     <ul>
          {% for song in songs %}
               <li><a href="{{song.url}}"> {{song.title}} </a></li>
          {% endfor %}
     </ul>
{% endblock %}

render_template("xx.html", username="Jade")

{# comment #}
```

trim_blocks

```
from flask import render_template

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

# ./templates/index.html
<pre>
ttt 한글
{% if True  %}
    TTT
{% endif %}qqq
</pre>

# trim_blocks app config
app.jinja_env.trim_blocks = True
```

trim_blocks (Cont'd)

```
# ./templates/index.html
<pre>
ttt 한글
{%- if True -%}
    TTT
{%- endif -%}qqq
</pre>


# invalid
{% - if True - %}

# Tip: nodemon watching the html
nodemon start_helloflask.py -w helloflask/__init__.py -w helloflask/templates/index.html
```

escape

```
# quotation escape
{{ abc {ef} ghi }}  ⇒ {{ "abc {ef} ghi" }}
{{ "}}>> <strong>Strong</strong>"}}    or   {{ '}}>> <strong>Strong</strong>' | escape }}

# cf. safe string & striptags
{{ "<strong>Strong</strong>" | safe}}
{{ "<strong>Strong</strong>" | striptags}}

# {% raw %} ~ {% endraw %} : display source code
{% raw %}
    {% if True  %}
        TTT
    {% endif %}
{% endraw %}
```

Markup

```
# from flask import Markup
return render_template("index.html", markup=Markup("<b>B</b>"))

# Example: Markup()
mu = Markup("<h1>iii = <i>%s</i></h1>")
h = mu % "Italic"
print("h=", h)
return render_template("index.html", markup=Markup(h))

# Markup.escape() & unescape()
bold = Markup("<b>Bold</b>")
bold2 = Markup.escape("<b>Bold</b>")
bold3 = bold2.unescape()

print(bold, bold2, bold3)
⇒ <b>Bold</b> &lt;b&gt;Bold&lt;/b&gt; <b>Bold</b>
```

FOR loop

```
# {% for var in iter %}  …  {% endfor %}
{% for item in items %}
     ...item 처리..
{% endfor %}

# Example
lst = [ ("만남1", "김건모"), ("만남2", "노사연") ]
return render_template("index.html", lst=lst)

<ul>
    {% for item in lst %}
        <li>{{item[0]}}: {{item[1]}}</li>
    {% endfor %}
</ul>

{% for title, name in lst %}
    <li>{{title}}: {{name}}</li>
{% endfor %}
```

loop object

 # for loop 속에서 기본으로 제공되는 object : `현재 for loop 의 self`

- loop.**index**: 1부터 시작하는 index 값 (cf. loop.**index0**)
- loop.**revindex**: n~1 내림차순 index값 (cf. loop.**revindex0**)
- loop.**first**: boolean(isThis**First**Item), loop의 첫번째인지의 여부
- loop.**last**: boolean(isThis**Last**Item), loop의 마지막인지의 여부
- loop.**length**: size
- loop.**depth** : loop 깊이

 # loop.cycle (특정 주기로 수행)

```
<ul>
    {% for item in lst %}
        <li class="{{loop.cycle('aaa', 'bbb')}}">{{item[0]}}: {{item[1]}}</li>
    {% endfor %}
</ul>
```

for loop Filtering

```
# data의 세번째 인자로 숨김 여부 추가
lst = [ (1, "만남", "김건모", False), (2, "만남", "노사연", True), (3, "만남", "익명", False) ]
return render_template("index.html", lst=lst)

# index.html
<ul>
    {% for rank, title, name, hide in lst if not hide %}
        <li class="{{loop.cycle('aaa', 'bbb')}}">{{title}}: {{name}}</li>
    {% endfor %}
</ul>

# for else
{% for title, name, isShow in lst %}
    <li class="{{loop.cycle('aaa', 'bbb', '')}}">{{title}}: {{name}}</li>
{% else %}
    <li>There is no data.</li>
{% endfor %}
```

for recursion

```
# loop(data)
a = (1, "만남1", "김건모", False, [])
b = (2, "만남2", "노사연", True, [a])
c = (3, "만남3", "익명", False, [a,b])
d = (4, "만남4", "익명", False, [a,b,c])

return render_template("index.html", lst=[a,b,c,d])

# index.html
<ul>
   {% for rank, title, name, hide, ref in lst recursive %}
        <li>
            {{title}}: {{name}}
            <ul class="sub">{{ loop(ref) }}</ul>
        </li>
    {% endfor %}
</ul>
```

{% if condition %} ... {% endif %}

```
# grammar
{% if <Condition> %}
     ....
{ % endif %}

# Example
{% if lst %}
     {% for .... %}
{ % endif %}

# if else (elif)
{% if <Condition> %}
     ....
{% elif <Other Condition> %}
     ...
{% else %}
     ...
{ % endif %}
```

set & parent's loop object

```
# set value
{% set title = 'ABC' %}

# access parent(outer) loop
<ul>
    {% for rank, title, name, hide, ref in lst2 recursive %}
    <li>
        {{loop.index}} - <small>{{title}}</small>: {{name}}
        {%- if ref -%}
            {% set outer_loop = loop %}
            {% for ref_song in ref %}
                <p>{{outer_loop.index}} - {{loop.index}} : {{ ref_song[1] }}</p>
            {% endfor %}
        {%- endif %}
    </li>
    {% endfor %}
</ul>
```

Try This: Recursive For

1) 다음과 같은 형태를 갖는 메뉴(NavigationBar)를 for recursive를 이용하여 HTML로 출력하시오.(title, url, children)

- 프로그래밍 언어
  - 파이썬
  - 자바
- 웹 프레임워크
  - 플라스크
    - Jinja
    - Genshi Cheetah
  - 스프링
  - 노드JS
- 기타
  - 나의 일상
  - 이슈 게시판

URL & Link

```
# url_for('folder', filename='filename.ext')
<link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}" >

# url_for('router-link')
Copyright <a href="/tmpl">IndiFlex Senior Coding</a>
Copyright <a href="{{ url_for('tmpl') }}">IndiFlex Senior Coding</a>
```

Template Extends (block ~ endblock)

```
# Base Template: 구조 및 자리 잡기용 (layout.html)
<body>
     <h1>{% block <block-name> %}{% endblock %} - Layout Title </h1>
</body>

# extends the base layout html (main.html)
{% extends "layout.html" %}

# mapping block : block 사용, 순서 무관! (in main.html)
{% block <block-name> %}AAAAA{% endblock %}

# super() : import html from same block-name
{% block <block-name> %}
     {{ super() }}
     <p>TTT</p>
{% endblock %}
```

Duplicate Blocks

```
# layout.html
{% block outer_block %}
     <h3>
          {% block inner_block %}{% endblock inner_block %}
     </h3>
{% endblock outer_block %}

# main.html
{% block outer_block %}
    <p>1111111111</p>
    {% block inner_block %}BBBBBB{% endblock inner_block %}
{% endblock outer_block %}
```

Use blocks in For loop

```
# layout.html
<h2>{% block title2 %}{% endblock %} - Layout Title2</h2>

# main.html
{% for item in [1,2,3] %}
    {% block title2 scoped %} <p>XXXXXXXXXX {{item}}</p> {% endblock %}
{% endfor %}
```

Try This: Site 구조 잡기

1) application.html의 경로를 모두 url_for()를 사용하여 변경하시오.

2) 실제로 만들 Site의 Markup된 HTML(application.html)의 구조를 잡고, block을 구성하시오.

Macro

```
# {% macro macro_name(args…) %} ~ {% endmacro %}
{% macro test_macro(type) -%}
    <h3>
        TEST MACRO: {{type}} - {{test_macro.caller}}        # False
    </h3>
{%- endmacro %}

# main.html
{% block … %}
    <p>{{ test_macro('password') }}</p>
{% endblock %}
```

Callable Macro

```
# {% call macro_name(args…) %} ~ {% endcall %}
{% macro test_macro2(name, class='red') -%}
     <h3 class="{{class}}">
          TEST MACRO2: {{name}} - {{test_macro2.caller}}    # True cf. hasBlock
          <div> {{caller()}} </div>
     </h3>
{%- endmacro %}

# main.html
{% block … %}
    {% call test_macro2('Hong') %}
       <p>This is main.macro.call</p>
    {% endcall %}
{% endblock %}

# call with args
{% call(x) test_macro('password') $}  {{x}} {% endcall %}
  ⇐  in macro: caller(x=200)
```

Import Macro Module

```
# {% import "macro_file_path" as <macro-alias> [with context] %}
{% import "macro/commons.html" as cm %}
     <h3>
          TEST MACRO2: {{cm.test_macro2()}}
     </h3>

# {% import "macro_file_path" as <macro-alias> with context %}
  (macro.html에서 main.html의 변수를 사용할 수 있음)

# 특정 매크로만 import 하기!!
# {% from "file_path" import <macro1>, <macro2> %}
{% block … %}
    <p>{{ test_macro('password') }}</p>
{% endblock %}

# macro_name이 _ (underscore)로 시작하면 private(import 불가)!!
```

Try This: macro

1) 아래와 같은 form에서 사용되는 태그들을 macro로 작성하시오.

input

textarea

radio

checkbox

select

Try This: macro - modal

1) modal창을 macro로 작성하시오.

header

body - caller()

footer - isShowFooter

Auto Versioning for Static files

```
@app.context_processor
def override_url_for():
    return dict(url_for=dated_url_for)

def dated_url_for(endpoint, **values):
    if endpoint == 'static':
        filename = values.get('filename', None)
        if filename:
            file_path = os.path.join(app.root_path,
                                     endpoint, filename)
            values['q'] = int(os.stat(file_path).st_mtime)
    return url_for(endpoint, **values)
```

```
<link href="/static/css/bootstrap.min.css?q=1548957792" rel=
"stylesheet">
<!-- Material Design Bootstrap -->
<link href="/static/css/mdb.min.css?q=1548957792" rel="stylesheet">
<!-- Your custom styles (optional) -->
<link href="/static/css/style.css?q=1550035841" rel="stylesheet">
```

Include

```
# {% include "include_file_path" %}
<div>
     {% include "inc/navbars.html" %}
</div>

# {% include "include_file_path" ignore missing %}

# {% include ["a.html", "b.html"] %}
<div>
     {% include ["inc/navbars.html", "inc/menus.html"] ignore missing %}
</div>

# {% include "include_file_path" without context %}  # default: with context!!
<div>
     {% include "inc/navbars.html" without context %}
</div>
```

Template Filters

```
@app.template_filter('ymd')                    # cf. Handlebars' helper
def datetime_ymd(dt, fmt='%m-%d'):
    if isinstance(dt, date):
        return dt.strftime(fmt)
    else:
        return dt

# in template
{{ today | ymd }}  or {{ today | ymd('%m/%d') }}  or {{today | ymd('%m-%d') | safe}}

# basic filters
safe, striptags, abs, escape, filesizeformat, replace, int, round, trim, truncate, wordwrap,
indent, center

# batch filter : batch(div size, str to fill)
{% for row in range(-2, 32) | batch(7, 'TT') %}
    <p>{{row}}</p>
{% endfor %}
```

Try This: include, template_filter

1) 반복해서 사용되는 HTML 부분을 include를 사용하여 분리하시오.

2) 시간을 받아서 오늘 날짜면 '시:분'을, 오늘 이전이면 '월/일'로 출력하는 template_filter를 작성하시오.

Try This: batch filter

1) 2019년 2월 달력을 출력하시오.
2) 2019년 전체 달력을 출력하시오.

(참고) **timedelta** vs **relativedelta** **pip install python-dateutil**

```
from datetime import datetime, timedelta
from dateutil.relativedelta import relativedelta

# timedelta units: days, hours, minutes, seconds, microseconds
# relativedelta units: months, years
d = datetime.strptime('2019-01-01', '%Y-%m-%d')
nextMonth = d + relativedelta(months=1)
cf>> (now - d2).days

# d.day, d.month, d.weekday() : {0:월, 1:화, 2:수, 3:목, 4:금, 5:토, 6:일}
```

Flask

web development,
one drop at a time

요청경 의

Static file versioning

IndiFlex 시니어코딩

Flask 3

web development,
one drop at a time

SQLAlchemy

IndiFlex 시니어코딩

Modularize Source

```
# helloflask package

# __init__.py : app

# views.py : router

# filters.py : template_filters

# utils.py : utility functions

# classes.py : classes

# models.py : Data Models
```

SQLAlchemy

```
# RDBMS ORM Manipulate in python                    (cf. MongoKit)

# python lib base in ORM (Object-Relational Mapping)

# 설치: pip install sqlalchemy


# modules
from sqlalchemy import create_engine, Table, Column
from sqlalchemy import Integer, String, Boolean, Date, Time, Float, BigInt, Binary,
LargeBinary, Blob, Clob, DateTime, TIMESTAMP
from sqlalchemy.orm import scoped_session, sessionmaker
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.exc import SQLAlchemyException
```

Initialize MySQL Connection

```
# Declare connection
mysql_url = "mysql+pymysql://<user>:<password>@<ip>/<dbname>?charset=utf8"
engine = create_engine(mysql_url, echo=True, convert_unicode=True)

# Declare & create Session
db_session = scoped_session( sessionmaker(autocommit=False, autoflush=False, bind=engine) )

# Create SqlAlchemy Base Instance
Base = declarative_base()
Base.query = db_session.query_property()

def init_database():
    Base.metadata.create_all(bind=engine)
```

Connection Management

```
# Auto Managed Connection, But db_session create every request

# initialize connection
@app.before_first_request
def beforeFirstRequest():
    init_database()

# close connection
@app.teardown_appcontext
def teardown(exception):
    db_session.remove()
```

Data Model

```
# Data Model is Value Object(DTO)

# Data model mapped Table

# extends SqlAlchemy Base Class and __tablename__
class User(Base):
     __tablename__ = 'User'

# Member variable is Column
    id = Column(Integer, primary_key=True)
    email = Column(String, unique=True)
    nickname = Column(String)

    def __init__(self, email=None, nickname='손님'):

    def __repr__(self):
        return 'User %r, %r' % (self.email, self.nickname)
```

Data Manipulates

```
# Create(insert)
u = User('abc@efg.com', 'hong')
db_session.add(u)

# Read(select)
u = User.query.filter(User.id == 2).first()    # cf. db_session.query(User).filter...
lst = User.query.all()

# Update
u.email = 'qqq@ppp.com'
db_session.merge(u)  # auto-insert if not exists id

# Delete
db_session.delete(u)

# Commit & Rollback (SQLAlchemyError)
db_session.commit()
db_session.rollback()
```

Send Query

```
# db_session().execute()
s = db_session()

s.execute("update User set nickname=:nickname where id = :id", {'id': 3,
'nickname': 'hong3'})
result = s.execute('select id, nickname from User where id > :id', {'id': 1})

from collections import namedtuple
Record = namedtuple('User', result.keys())
records = [Record(*r) for r in result.fetchall()]
for r in records:
    print(r, r.nickname, type(r))

s.close()
```

ORM Query

```
# query
User.query.filter(id == 100)
db_session.query(User).filter...

# String Filter
User.query.filter("id < :val").params(val=10)

# count
User.query.all()
User.query.one()
User.query.first()
User.query.count()

# order-by
User.query.order_by(User.id)
User.query.order_by(User.id.desc()).limit(10)
```

1:1 (Join, References) : Song.album - Album

```
# import
from sqlalchemy import ForeignKey
from sqlalchemy.orm import relationship, backref

class Album(Base):
    __tablename__ = 'Album'
    albumid = Column(String, primary_key=True)

class Song(Base):
    __tablename__ = 'Song'
    songno = Column(String, primary_key=True)
    albumid = Column(String, ForeignKey('Album.albumid'), nullable=False)
    album = relationship('Album', lazy='joined')

# in views.py
ret = Song.query.filter(Song.likecnt < 10000)
OR
ret = Song.query.join(Album, Song.albumid == Album.albumid).filter(Song.likecnt < 10000)
```

join vs subqueryload vs joinedload

```
# join
db_session.query(Song).join(Album, Album.id == Song.album)

# import
from sqlalchemy.orm import subqueryload, joinedload

# subqueryload
ret = db_session.query(Song).options(
          subqueryload(Song.album)).filter(Song.likecnt < 10000)

# joinedload
ret = db_session.query(Song).options(
          joinedload(Song.album)).filter(Song.likecnt < 10000)
```

Save with References

```
# exists ref
    a1 = Album.query.filter_by(albumid = '10242994').one()
    song1 = Song(songno='TTT2', title='TTT2 Title')
    song1.album = a1
    db_session.add(song1)
    db_session.commit()

# create with ref
    a1 = Album(albumid='TTT-a1', title='TTT-a1')
    song1 = Song(songno='TTT2', title='TTT2 Title')
    song1.album = a1
    db_session.add(song1)
    db_session.commit()
```

1:n References (Album.songs - Song)

```
# just relationship
class Album(Base):
    __tablename__ = 'Album'
    albumid = Column(String, primary_key=True)
    songs = relationship('Song', lazy='joined')


# (참고) Multi-column Primary Key
class SongArtist(Base):
    __tablename__ = 'SongArtist'
    songno = Column(String, ForeignKey('Song.songno'), nullable=False)
    artistid = Column(String, ForeignKey('Artist.artistid'))
    atype = Column(Integer)

    __table_args__ = ( PrimaryKeyConstraint('songno', 'artistid', 'atype'), {} )
```

n:n Reference (Song - SongArtist - Artist)

```
# Artist
class Artist(Base):
    __tablename__ = 'Artist'
    songartists = relationship('SongArtist', lazy='joined')

# Song
class Song(Base):
    __tablename__ = 'Artist'
    songartists = relationship('SongArtist', lazy='joined')

# SongArtist (Mapping Table)
class SongArtist(Base):
    __tablename__ = 'SongArtist'
    songno = Column(String, ForeignKey('Song.songno'), nullable=False)
    artistid = Column(String, ForeignKey('Artist.artistid'))
    atype = Column(Integer)
    song = relationship('Song', lazy='joined')
    artist = relationship('Artist', lazy='joined')
    __table_args__ = ( PrimaryKeyConstraint('songno', 'artistid', 'atype'), {} )
```

n:n Reference (Song - SongArtist - Artist)

```
# Song list (with Album info)
Song.query.options(joinedload(Song.album)).filter(Song.likecnt < 10000)

# (Song + Album) → (SongArtist + Artist)
Song.query.options(joinedload(Song.album)).filter(Song.likecnt < 10000)
          .options(joinedload(Song.songartists, SongArtist.artist))

# (Song + Album + SongArtist) + Artist
Song.query.options(joinedload(Song.album)).filter(Song.likecnt < 10000)
          .options(joinedload(Song.songartists))
          .options(subqueryload(Song.songartists, SongArtist.artist))
```

Flask 4

web development,
one drop at a time

웹사이트 만들기

IndiFlex 시니어코딩

SiteMap

```
# Server Side Template
/ Home (app.html) : 가요 Top 100 목록 (일별, 실시간)
/myalbum My Album (app.html) : 나만의 앨범 (앨범 생성 - 앨범에 곡 담기)
/regist Sign Up(regist.html) : 가입하기(GET: form, POST: regist → /login)
/login Sign In(login.html) : 로그인하기(GET: form, POST: login)
/logout Sign Out(redirect to /) : 로그아웃하기


# AJAX
/songinfo/<songno> : 노래별(아티스트) 정보
```

Worklist

1) home(app.html) - live top 100
2) home(app.html) - today top 100
3) home - songinfo ajax
4) login
5) regist
6) login info
7) myalbum - 앨범 생성
8) myalbum - 노래 담기 (home)
9) myalbum - 담긴 곡 list

view decorator

1) home(app.html) - live top 100
2) home(app.html) - today top 100
3) home - songinfo ajax
4) login
5) regist
6) login info
7) myalbum - 앨범 생성
8) myalbum - 노래 담기 (home)
9) myalbum - 담긴 곡 list

Flask

web development,
one drop at a time

5

Flask + Handlebars

IndiFlex 시니어코딩

Flask

web development,
one drop at a time

6

File Uploader

IndiFlex 시니어코딩

Upload & Download File (on HTTP)

```
upfile = request.files['file']

# safe upload (eg. ../../aaa)
from werkzeug.utils import secure_filename
filename = secure_filename(upfile.filename)



# Download file
send_file(path, as_attachment=True)
```

(참고) http://flask.pocoo.org/docs/1.0/patterns/fileuploads/

Javascript FormData & AJAX

```
var form = $('#frm_' + id)[0],
            formData = new FormData(form),
            file = $("#file_" + id)[0].files[0];

  formData.append("file", file);
  formData.append("myalbumid", id);

  $.ajax({
      url: '/upload',
      processData: false,
      contentType: false,
      data: formData,
      type: 'POST',

      success: function (res) {\
          console.log("res>>", res);
          if (res && res.path)
              $('#img_' + id).attr('src', res.path)
      }
  });
```

Rename duplicate filename

```
def rename(path):

    while True:

        if os.path.isfile(path):
            idx = path.rindex('.')

            if idx == -1:
                path += '1'
            else:
                path = path[:idx] + '1' + path[idx:]

        else:
            return path
```

(Tip) Change the flask server port

```
# start_helloflask.py

app.run(host='0.0.0.0', port=80)
```

Flask

web development,
one drop at a time

7

이메일 발송하기

IndiFlex 시니어코딩

Sendmail by Gmail

```
# import library
import smtplib
from email import encoders
from email.mime.text import MIMEText
from email.mime.multipart import MIMEMultipart
from email.mime.base import MIMEBase

# import library & login for SMTP
smtp = smtplib.SMTP('smtp.gmail.com', 587)
smtp.ehlo()
smtp.starttls()
smtp.login('indiflex1@gmail.com', os.environ['GMAIL_PASSWD'])

# message
msg = MIMEMultipart()
msg['Subject'] = 'Test Title'
content = MIMEText('SMTP로 메일 보내기 본문 메시지입니다.')
msg.attach(content)
```

Sendmail by Gmail (Cont'd)

```
# attach files
filepath = "./indiflex.png"
with open(filepath, 'rb') as f:
    part = MIMEBase("application", "octet-stream")
    part.set_payload(f.read())
    encoders.encode_base64(part)
    part.add_header('Content-Disposition', 'attachment', filename=filepath)
    msg.attach(part)

# send & quit
addr = "jeonseongho@naver.com"
msg["To"] = addr
smtp.sendmail("indiflex1@gmail.com", addr, msg.as_string())
smtp.quit()
```

Flask

web development,
one drop at a time

8

OAuth2 Login

IndiFlex 시니어코딩

Google API & OAuth2

```
# https://console.cloud.google.com
# console > 프로젝트 선택 > API 개요 > 사용자 인증정보 (사용자 인증 정보 만들기)
# Set the URL & Callback URL  (http://localhost:5000,  http://localhost:5000/oauth2callback)
# Download json secret file

# use flask_util
from oauth2client.contrib.flask_util import UserOAuth2

# set API key
app.config['GOOGLE_OAUTH2_CLIENT_SECRETS_FILE'] = 'secret.json'
app.config['GOOGLE_OAUTH2_CLIENT_ID'] = os.environ['OAUTH_CLIENT']
app.config['GOOGLE_OAUTH2_CLIENT_SECRET'] = os.environ['OAUTH_SECRET']

# apply app to oauth2
oauth2 = UserOAuth2(app)

# /logout
session.modified = True
oauth2.storage.delete()
```

Google API & OAuth2 (Cont'd)

```
# google oauth login  (oauth2.email, oauth2.user_id)
@app.route('/google_oauth')
@oauth2.required
def google_oauth():
    u = User.query.filter('email = :email').params(email=oauth2.email).first()
    if u is not None:
        session['loginUser'] = {'userid': u.id, 'name': u.nickname}
        return redirect('/')

    else:
        flash("해당 사용자가 없습니다!!")
        return render_template("login.html", email=oauth2.email)
```

Flask

web development,
one drop at a time

Unit Test

IndiFlex 시니어코딩

Flask

web development,
one drop at a time

Jenkins 배포하기

IndiFlex 시니어코딩