Class 230302

Spring - 3일차

@PathVariable

`경로/변수1/변수2` 형태로 사용

▼ 변수 1개

```
@Controller
@RequestMapping("/vari")
public class VariableController {

  @GetMapping("/names/{name}")
  public String names(@PathVariable String name, Model model) {
    model.addAttribute("name", name);
    return "/study/name";
  }
}
```

▼ 변수 2개 이상

```
@Controller
@RequestMapping("/vari")
public class VariableController {

  @GetMapping("/names/{name}/{gender}")
  public String names(@PathVariable String name, @PathVariable String gender, Model model) {
    model.addAttribute("name", name);
    model.addAttribute("gender", gender);
    return "/study/name";
  }
}
```

@MatrixVariable

`경로;변수명=값` 형태로 사용

⚠️ **servlet-context.xml 설정 필요**

```
<!-- MatrixVariable -->
<annotation-driven enable-matrix-variables="true"/>
```

▼ 코드

```
@GetMapping("/names/car/{carId}")
  public String names2(@PathVariable String carId, @MatrixVariable String cate, Model model) {
    model.addAttribute("id", carId);
    model.addAttribute("cate", cate);
    return "/study/car";
  }
```

- http://localhost:8080/vari/names/car/c0001;cate=중형 형식으로 사용

@RequestParam

`경로?변수명1=값1&변수명2=값2` 의 쿼리스트링 형태로 사용

▼ 코드

```
@Controller
@RequestMapping("/param")
public class ParamController {

  @GetMapping("/exam1")
  public String exam1(@RequestParam String id, Model model) {
    model.addAttribute("id", id);

    return "/study/car";
  }
}
```

CarShop 차량 상세보기

▼ CarRepository.java

```
CarDTO getCarById(String carId);
```

▼ CarRepositoryImpl.java

```
@Override
  public CarDTO getCarById(String carId) {

    CarDTO carInfo = null;

    for (int i = 0 ; i < listOfCars.size() ; i++) {
      CarDTO carDTO = listOfCars.get(i);
      if(carDTO != null && carDTO.getCid() != null && carDTO.getCid().equals(carId)) {
        carInfo = carDTO;
      }
    }

    if (carInfo == null) {
      throw new IllegalArgumentException("자동차 ID 가 " + carId + "인 자동차는 없습니다. ");
    }

    return carInfo;
  }
```

▼ CarService.java

```
CarDTO getCarById(String carId);
```

▼ CarServiceImpl.java

```
@Override
  public CarDTO getCarById(String carId) {
    return carRepository.getCarById(carId);
  }
```

▼ CarController.java

```
@GetMapping("/car")
  public String requestCarById(@RequestParam("id") String carId , Model model) {
    CarDTO car = carService.getCarById(carId);
    model.addAttribute("car", car);
    return "/car/car";
  }
```

▼ view

```
  <%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>차 상세보기</title>
</head>
<%@ include file="../include/header.jsp" %>
<body>
<%@ include file="../include/navbar.jsp" %>
  <div class="alert alert-dark">
    <div class="container">
      <h1>차량 상세보기</h1>
```

```
    </div>
  </div>
  <div class="container">
    <h3>${car.cid}</h3>
    <p>${car.cname}</p>
    <p>${car.cprice} 만원</p>
    <p>${car.ccate}</p>
    <p>${car.cdesc}</p>
    <p><a href="<c:url value="/cars/list" />" class="btn btn-secondary btn-sm">목록으로</a></p>
  </div>
<%@ include file="../include/footer.jsp" %>
</body>
</html>
```

CarShop 차량 등록

▼ CarRepository.java

```
void setNewCar(CarDTO car);
```

▼ CarRepositoryImpl.java

```
@Override
  public void setNewCar(CarDTO car) {
    listOfCars.add(car);
  }
```

▼ CarService.java

```
void setNewCar(CarDTO car);
```

▼ CarServiceImpl.java

```
@Override
  public void setNewCar(CarDTO car) {
    carRepository.setNewCar(car);
  }
```

▼ CarController.java

```
@GetMapping("/add")
  public String add(@ModelAttribute("car") CarDTO car) {
    return "/car/addCar";
  }
  @PostMapping("/add")
  public String addCar(CarDTO car, Model model) {
    carService.setNewCar(car);
    List<CarDTO> list = carService.getAllCarList();
    model.addAttribute("carList",list);
```

```
    return "/car/cars";
  }
```

▼ view

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://www.springframework.org/tags/form" prefix="form"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>차 등록</title>
</head>
<%@ include file="../include/header.jsp" %>
<body>
<%@ include file="../include/navbar.jsp" %>
  <div class="alert alert-dark">
    <div class="container">
      <h1>차량 등록</h1>
    </div>
  </div>


  <div class="container">
    <form:form modelAttribute="car" method="POST" class="form-horizontal">
    <fieldset>
    <legend>${ addTitle }</legend>
      <p>차량 ID : <form:input path="cid" class="form-control"/></p>
      <p>차량 이름 : <form:input path="cname" class="form-control"/></p>
      <p>차량 가격 : <form:input path="cprice" class="form-control"/></p>
      <p>차량 카테고리 : <form:input path="ccate" class="form-control"/></p>
      <p>차량 소개 : <form:textarea path="cdesc" cols="50" rows="2" class="form-control"/></p>

      <button class="btn btn-sm btn-secondary">등록</button>
    </fieldset>
    </form:form>
  </div>
<%@ include file="../include/footer.jsp" %>
</body>
</html>
```

▼ **modelAttribute 옵션**

- **modelAttribute** 옵션은 폼에 있는 요소들의 값을 채우기 위해서 사용될 객체를 request 로부터 찾을때 사용할 이름을 지정

⚠️ commandName은 5버전에서는 다음과 같은 에러 발생

```
Unable to find setter method for attribute: [commandName]
```

이 때에는 commandName 대신에 modelAttribute 옵션을 사용

- 일반적으로 폼에 사용되어지는 id와 name 속성은 modelAttribute와 별도로 지정하는 것이 편리할 때가 많음

```
<form:form id="testForm" name="testForm" modelAttribute="testVO"></form:form>


결과
HTML:<form id="testForm" name="testForm" action="/testForm.do" method="post"></form>
```

form 태그에서 사용될 모델 객체를 설정하는 방법

- form태그의 modelAttribute에서 지정되는 폼의 요소들에 값을 제공할 객체의 지정은 보통 컨트롤러에서 함

- 컨트롤러 메소드의 인자에 @ModelAttribute 어노테이션을 사용하는 방법.

```
@GetMapping("/testForm.do")
  public String testForm(@ModelAttribute TestVO testVO, Model model) throws Exception {
  testVO.setTitle("제목 입니다.");
  return "testForm";
}
```

- @ModelAttribute("TestVO") 처럼 이름을 지정하면 변수명(vo) 과 다른 이름을 사용할 수 있음

```
@GetMapping("/testForm.do")
public String boardForm(@ModelAttribute("testVO") TestVO vo, Model model) throws Exception {
  vo.setTitle("제목 입니다.");
  return "testForm";
}
```

- form 태그에 modelAttribute 속성을 사용하고 또한 폼요소를 사용하였는데, 컨트롤러에서 모델 객체를 설정하지 않으면 에러 발생

```
form:input 요소 사용 :
<form:form modelAttribute="testVO">  <form:input path="title" /></form:form>

에러발생 :
ERROR: org.springframework.web.servlet.tags.form.InputTag -Neither BindingResult nor plain tar
get object for bean name 'testVO' available as request attribute
```

Python Adv - Crawling 3일차

▼ 모듈

```
import requests                           # 웹 문서를 text형태로 가져오기 위해 필요
from bs4 import BeautifulSoup as bs       # 가져온 text형태의 문서의 태그에 접근할 수 있게 정제
import pandas as pd                       # 데이터 분석
from selenium import webdriver            # 웹 브라우저 컨트롤 (막힌 사이트 스크래핑 용)
from urllib.request import urlretrieve    # 파일 다운로드 관련
import re                                 # 정규식 사용
import time
from selenium.webdriver.common.by import By     # 브라우저 조작 관련
from selenium.webdriver.common.keys import Keys # 브라우저 key 입력 관련

from pykrx import stock                   # 주가 조회 모듈
import yfinance as yf                     # 해외 주가 조회 모듈
import pybithumb as pbit                  # 빗썸 API
```

▼ `pykrx` 모듈 사용 (국내 주식)

```
'''
기간별 주가 조회
'''
stock.get_market_ohlcv('20230201', '20230228', '005930')
```

▼ `yfinance` 모듈 사용 (해외 주식)

```
'''
애플
'''
yf.download('AAPL', start='2023-02-28')
'''
태슬라
'''
yf.download('TSLA', start='2023-02-28')
```

▼ `pybithumb` 빗썸 API 모듈 사용

```
'''
빗썸 API
'''
tickers = pbit.get_tickers()    # 종목 이름들을 가져옴
#print(tickers)

bitcoin_price = pbit.get_current_price(("BTC")) # BTC 가격을 가져옴
#print(bitcoin_price)
```

```
# 비트코인 가격 반복 조회 (1초 간격)
while True:
```

```
    bitcoin_price = pbit.get_current_price(("BTC"))
    print(bitcoin_price)
    time.sleep(1)
```

```
# 모든 코인 가격 조회
for ticker in pbit.get_tickers():
    print(ticker, ":", pbit.get_current_price((ticker)))
```

```
# 자세한 코인 가격 조회 (시가 / 고가 / 저가 / 종가(현재가) / 거래량)
print(pbit.get_market_detail("BTC"))
```

```
# 모든 코인 detail 가격 조회
try:
    for ticker in pbit.get_tickers():
        print(ticker, ":", pbit.get_market_detail((ticker)))
except:
    print('이게 터지네;;')
```

```
# 자동으로 이동평균 계산

btc = pbit.get_ohlcv()
close = btc['close']

roll5 = close.rolling(5)
mean5 = roll5.mean()
print(mean5)

# 5일간의 평균을 계산하기 때문에 최초 4일은 계산 값이 없이 NaN으로 표시
```

```
# 상승장 하락장 파악
btc = pbit.get_ohlcv()
close = btc['close']

roll5 = close.rolling(5)
mean5 = roll5.mean()
last_mean5 = mean5[-2]

# 비트 코인 현재 값
price = pbit.get_current_price('BTC')

print('상승장') if price > last_mean5 else print('하락장')
```

스타벅스 매장 검색해서 가져오기

▼ 코드

```
'''
스타벅스
```

```
'''

driver = webdriver.Chrome('chromedriver.exe')                    # 크롬을 이용해 불러옴
driver.get('https://www.starbucks.co.kr/store/store_map.do')

time.sleep(3)                                                    # 페이지 로드 전에 하위 코드 실행 방지

area_btn = driver.find_element(By.CSS_SELECTOR, '.loca_search h3 a')        # 검색어 창 찾기
area_btn.click()
time.sleep(1)

area_btn = driver.find_element(By.CSS_SELECTOR, '.set_sido_cd_btn')        # 검색어 창 찾기
area_btn.click()
time.sleep(1)

area_btn = driver.find_element(By.CSS_SELECTOR, '.set_gugun_cd_btn')        # 검색어 창 찾기
area_btn.click()
time.sleep(3)

html = driver.page_source
bsObj = bs(html)

roc = bsObj.select('.quickResultLstCon strong')
address = bsObj.select('.quickResultLstCon .result_details')

storeData = []
for i in range(len(roc)):
    storeData.append([roc[i].text, address[i].text])
df = pd.DataFrame(storeData, columns=['위치', '주소'])
df
```