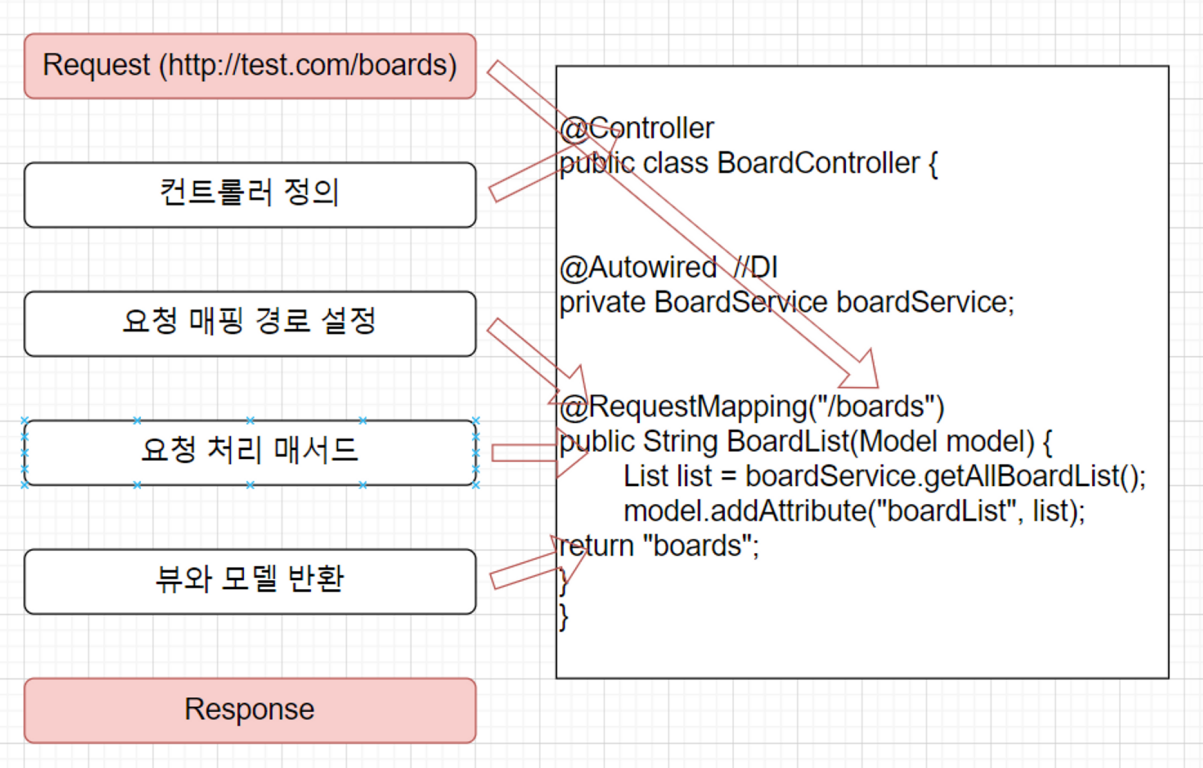
Class 230228

Spring - 2일차

Controller



- 스프링 컨트롤러는 웹 어플리케이션에서 클라이언트의 요청을 처리하는 역할을 담당하는 클래스
- 컨트롤러는 `@Controller` 어노테이션을 사용하여 스프링에서 해당 클래스가 컨트롤러임을 인식하도록 해야 함.
- 컨트롤러 클래스는 일반적인 자바 클래스와는 달리, 클라이언트로부터 들어온 요청을 처리하는 매서드를 포함.
- servlet-context.xml의 component-scan에 해당 컨트롤러가 있는 패키지를 명시해줘야 함.

RequestMapping

```
@Controller
@RequestMapping(value="/board", method=RequestMethod.GET)  // 클래스에 RequestMapping
public class BoardController {                             // GET 요청 시

  @Autowired
  private com.carshop.service.BoardService BoardService;

  @RequestMapping("/list")                                 // 메서드에 RequestMapping
  public String BoardList(Model model) {
    List<BoardDTO> list = BoardService.getAllBoardList();
    model.addAttribute("boardList",list);
    return "/board/list";
  }

  @RequestMapping("/test")                                 // 메서드에 RequestMapping
  public String test(Model model) {
    return "/board/test";
  }
}
```

- 클래스 단위의 매핑과 매서드 단위의 매핑을 적절히 사용하면 가독성에 이점이 있음

GetMapping / PostMapping …

스프링 4.3부터 지원

```
@GetMapping("/test")

@PostMapping("/test")

@PutMapping("/test")

@DeleteMapping("/test")

@PatchMapping("/test")
```

→ `@RequestMapping(value="/test", method=RequestMethod.GET)` 와 같은 방식은 가독성이 떨어짐

Medel / ModelMap / ModelAndView

Model	인터페이스 데이터만 저장
ModelMap	클래스 데이터만 저장
ModelAndView	데이터와 이동하고자 하는 View Page를 같이 저장

- Model (인터페이스) / ModelMap (구현 클래스)
  - model.addAttribute("변수명");
  - modelMap.addAttribute("변수명");
  - `addAttribute`를 사용
  - Model / ModelMap에 데이터만 저장 후 View에서 사용목적

```
@GetMapping("/test1")
  public String test1(Model model) {
    model.addAttribute("data1", "모델입니다.1");
    model.addAttribute("data2", "모델입니다.2");

    return "/study/test";
  }

  @GetMapping("/test2")
  public String test2(ModelMap model) {
    model.addAttribute("data1", "모델맵입니다.1");
    model.addAttribute("data2", "모델맵입니다.2");

    return "/study/test";
  }
```

- ModelAndView
  - `addObject`를 통해 데이터를 저장
  - `setViewName`을 통해 이동하고자 하는 View 경로를 저장
  - ModelAndView 객체를 반환

```
@GetMapping("/test3")
  public ModelAndView test3(ModelAndView mv) {
    mv.addObject("data1", "모델앤뷰입니다.1");
    mv.addObject("data2", "모델앤뷰입니다.2");
```

```
    mv.setViewName("/study/test");

    return mv;
  }
```

경로 변수와 @PathVariable

웹 요청 url에 포함된 파라미터 값을 전달 받는 경로 변수와 이를 처리하는 @PathVariable 어노테이션

```
@RequestMapping("/{ccate}")
public String requestCarById(@PathVariable("ccate") String ccate, Model model) {
  List<CarDTO> carscate = carService.getCarListByCategoty(ccate);
  model.addAttribute("carList", carscate);
  return "cars";
}
```

스프링 한글 처리 필터

- web.xml 파일에 다음 필터 추가

```
<!-- 한글 처리 -->
<filter>
<filter-name>encodingFilter</filter-name>
<filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
<init-param>
<param-name>encoding</param-name>
<param-value>UTF-8</param-value>
</init-param>
<init-param>
<param-name>forceEncoding</param-name>
<param-value>true</param-value>
</init-param>
</filter>
<filter-mapping>
<filter-name>encodingFilter</filter-name>
<url-pattern>/*</url-pattern>
</filter-mapping>
```

Python Adv - Crawling 2일차

▼ 네이버 웹툰 제목 크롤링

```
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)

'''
네이버 웹툰 Title
'''
html = requests.get('https://comic.naver.com/webtoon/weekday')
bsObj = bs(html.text, "html.parser")

titles = bsObj.select('.title')

data = []
for i in range(len(titles)):
    title = titles[i].text
    data.append([title])
df = pd.DataFrame(data, columns=['타이틀'])
df
```

▼ 네이버 웹툰 이미지 크롤링 및 다운

```
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   # 이미지 다운을 위한 모듈
import re                                # 정규식 사용

'''
네이버 웹툰 Thumbnail 다운
'''
html = requests.get('https://comic.naver.com/webtoon/weekday')
bsObj = bs(html.text, "html.parser")

titles = bsObj.select('.title')
thumbs = bsObj.findAll('div', {'class': 'thumb'})

data = []
for i in range(len(titles)):
    img = thumbs[i].find('img')['src']
    title = re.sub('[^0-9a-zA-Zㄱ-힣]', '', titles[i].text)
```

```
    data.append([titles[i].text, img])
    urlretrieve(img, './img/'+title+'.jpg')

df = pd.DataFrame(data, columns=['title', 'img'])
df
```

▼ 네이버 현재 날씨

```
'''
네이버 날씨
'''
html = requests.get('https://weather.naver.com/')
bsObj = bs(html.text, "html.parser")

weather = bsObj.find('strong', {'class': 'current'})

print(weather.text[6:])
```

▼ 비트코인 가격

```
'''
비트코인 가격 조회
'''
driver = webdriver.Chrome('chromedriver.exe')           # 크롬을 이용해 불러옴
driver.get('https://www.bithumb.com/trade/order/BTC_KRW')

html = driver.page_source
bsObj = bs(html)

bit = bsObj.select('span#assetRealBTC_KRW')

print(bit[0].text)
```