

사용자 데이터 -> 모델로 꺼내기



2025년 상반기 K-디지털 트레이닝

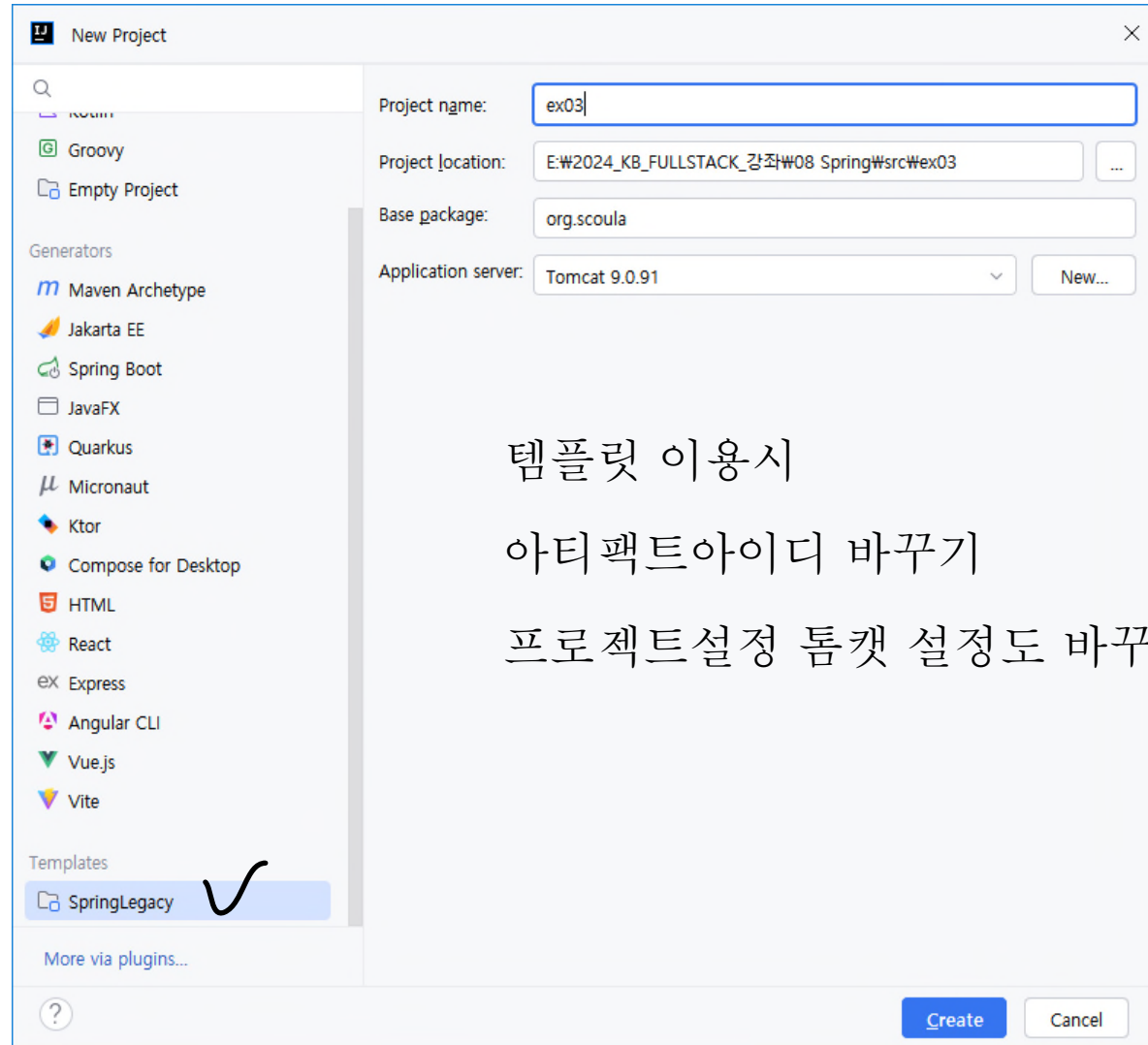
스프링 MVC의 Controller

[KB] IT's Your Life

1 스프링 MVC 프로젝트의 내부 구조

✓ 프로젝트 만들기

- 템플릿: SpringLegacy
- 프로젝트명: ex03



1 @Controller, @RequestMapping

ServletConfig.java 컨트롤러를 찾아내야함

```
package org.scoula.config;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.web.servlet.config.annotation.EnableWebMvc;
import org.springframework.web.servlet.config.annotation.ResourceHandlerRegistry;
import org.springframework.web.servlet.config.annotation.ViewResolverRegistry;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;
import org.springframework.web.servlet.view.InternalResourceViewResolver;
import org.springframework.web.servlet.view.JstlView;
```

@EnableWebMvc ————— @Controller 붙은 클래스 찾는거

```
@ComponentScan(basePackages = {
    "org.scoula.controller",
    "org.scoula.ex03.controller"
})

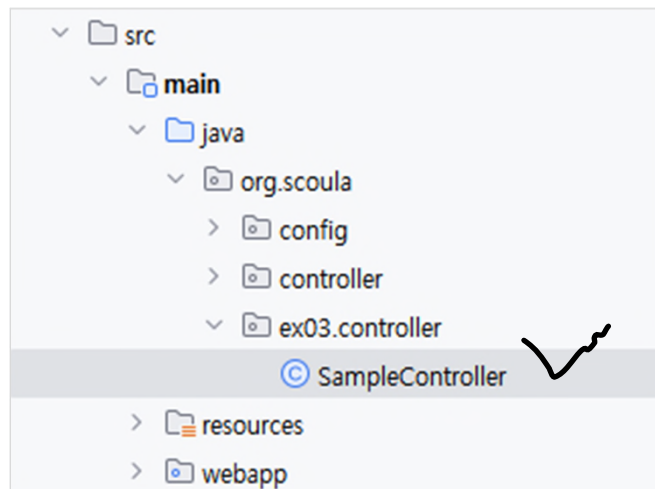
public class ServletConfig implements WebMvcConfigurer {
    ...
}
```

컨트롤러가 있는 패키지 나열 컨트롤러냐 서비스냐 구분해야함

1 @Controller, @RequestMapping

✓ 컨트롤러 추가

- org.scoula.ex03.controller.SampleController



1 @Controller, @RequestMapping

SampleController.java

```
package org.scoula.ex03.controller;
```

```
import org.springframework.stereotype.Controller;
```

```
import org.springframework.web.bind.annotation.RequestMapping;
```

✓ @Controller

@RequestMapping("/sample")

```
public class SampleController {
```

```
}
```

@RequestMapping

공통 URL 지정 어노테이션.

이 컨트롤러는 항상 공통 URL로 설정된다.

이 정보는 핸들러매핑에 전달됨.

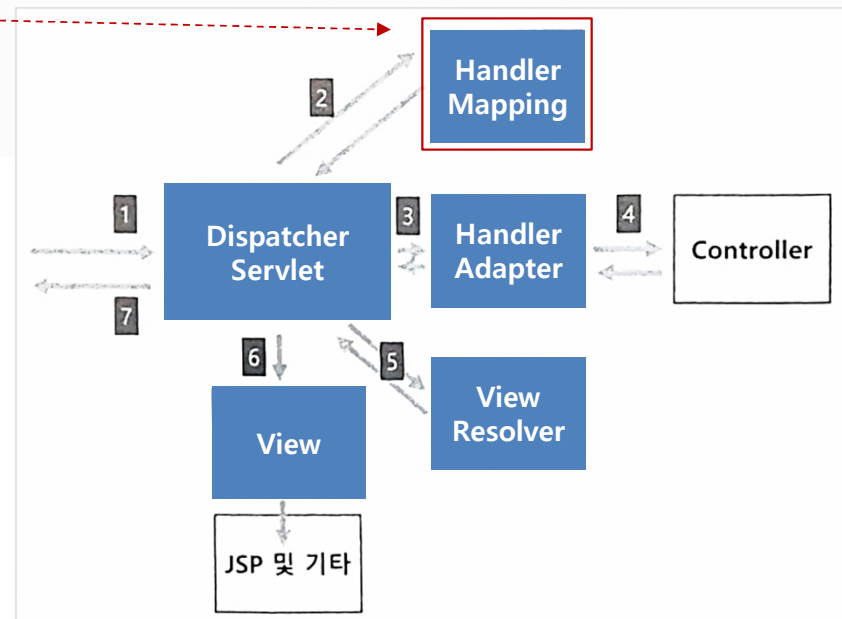
/sample/list

/sample/get

/sample/create ...

현재는 공통만 지정됐으니 모든 메서드를 받음

세부지정시 각 메서드당 많은 컨트롤러 지정가능



1 @Controller, @RequestMapping

SampleController.java

```
package org.scoula.ex03.controller;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;

import lombok.extern.log4j.Log4j2;

@Controller
@RequestMapping("/sample")
@Log4j2
public class SampleController {

    @RequestMapping("") // url : /sample
    public void basic() {
        log.info("basic.....");
    }
}
```

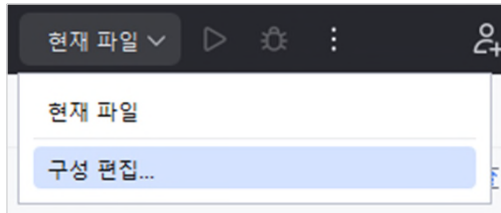
만들어진 최종 url. 메서드에 제한을 두지 않는 url.
get이든 post든

현재 메서드에는 의존성이 보이지 않는다. 매개변수나
내용이나 반환에서 주입할게 없다,

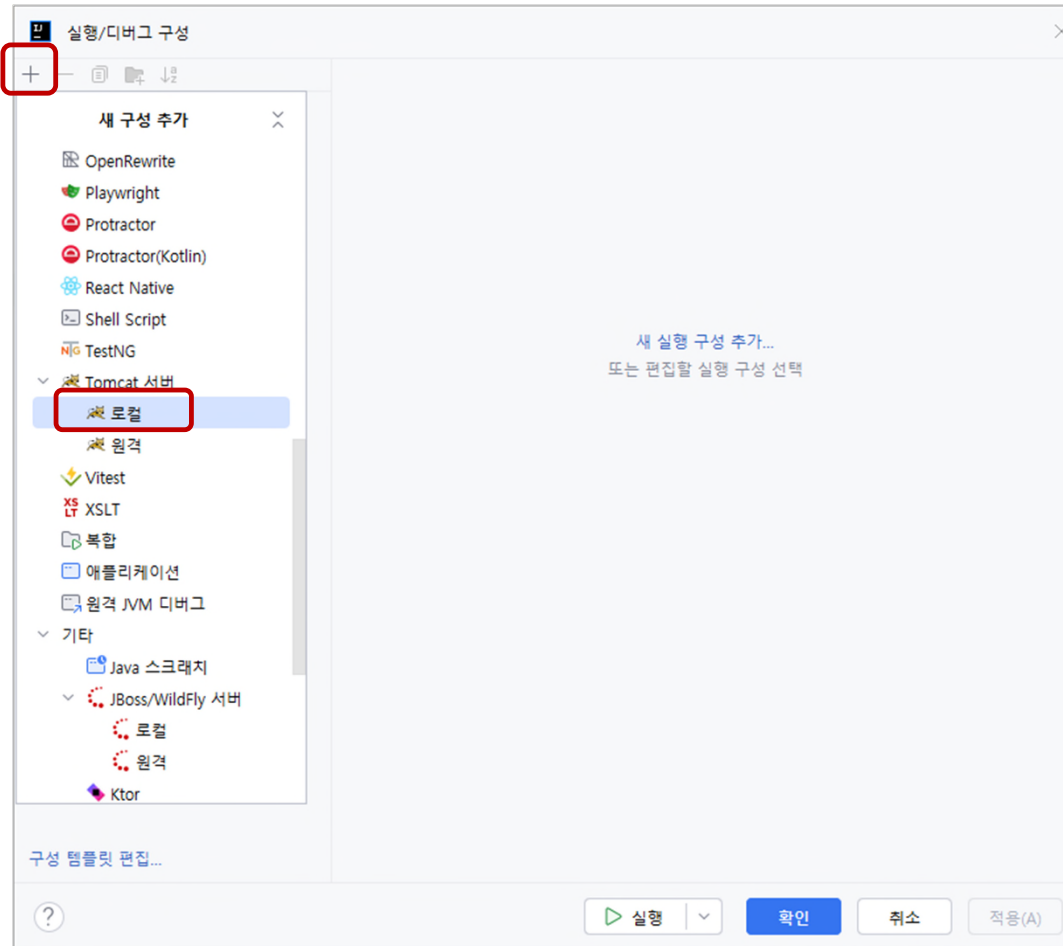
/? 메서드 정보가 없어도 그니까 매핑 정보가 없어도 404에러.

1 @Controller, @RequestMapping

✓ 실행 구성 편집



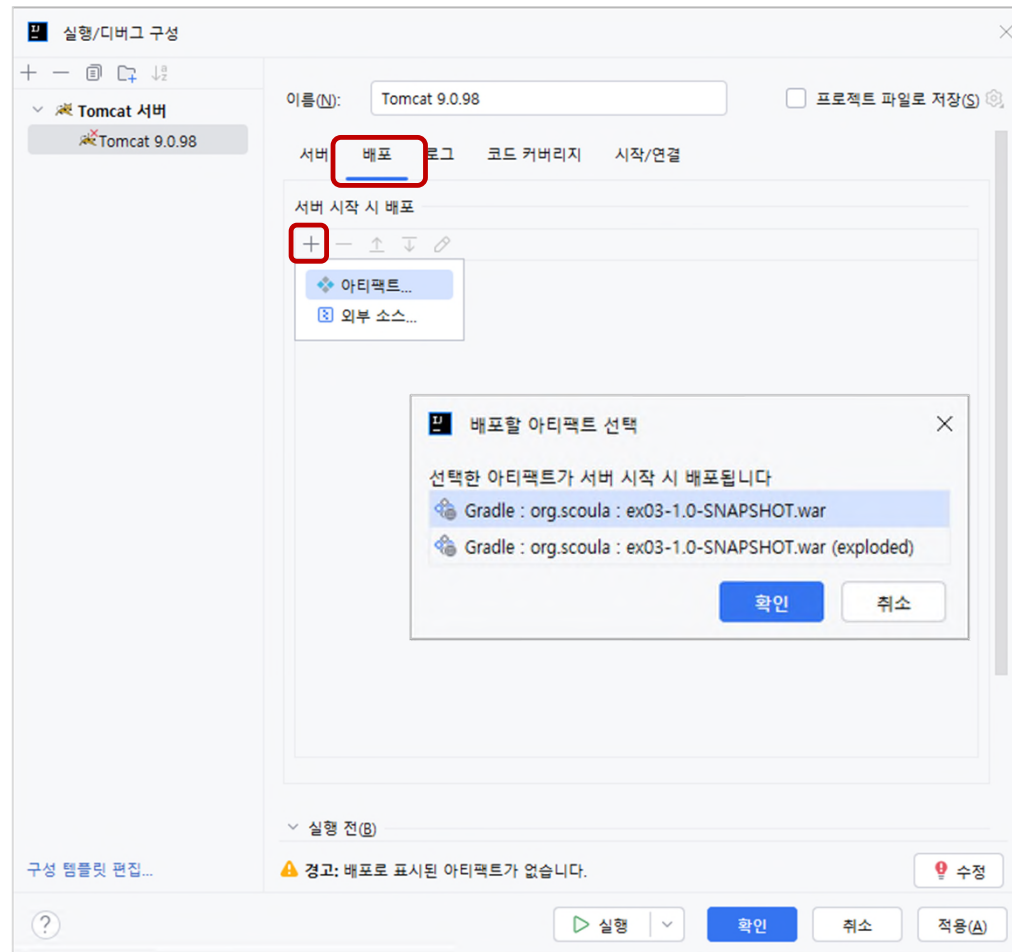
톰캣이
추가되지 않았을때
톰캣 추가



1 @Controller, @RequestMapping

✓ 실행 구성 편집

○ 아티팩트 설정

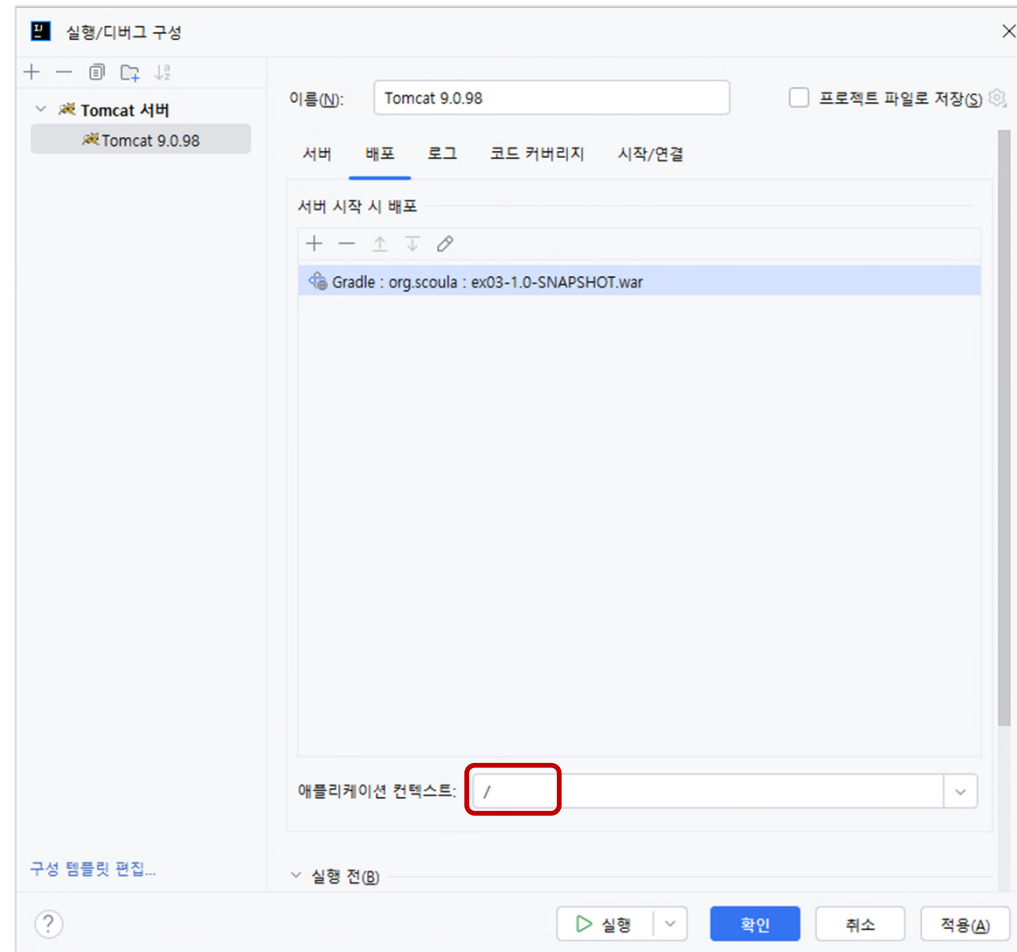


추가하고
아티팩트아이디도 추가!

1 @Controller, @RequestMapping

✓ 실행 구성 편집

○ Context Path 설정



컨텍스트 경로
루트로 지정

1 @Controller, @RequestMapping

✓ 확인

○ <http://localhost:8080/sample> ✓

INFO org.scoula.ex03.controller.SampleController(basic:18) - basic..... 로그메세지 확인



웹브 확인

controller가 아닌 sample.jsp를 찾을수 없대.

/sample 요청시 요청매핑때문에 basic메서드가 호출되었고 이는 반환하지 않음
요청url문자열이 반환되지 않았으면 요청url이 view의 이름이 되고나다. 따라서 없는 sample.jsp를 찾으려다가 못 찾은 거임

○ 브라우저에서는 jsp에 대한 404 에러 출력 → 컨트롤러는 동작했음. ✓

요청을 해석하고 컨트롤러까지 동작했다

에러시 url이 잘못됐는지 컨트롤러가 잘못됐는지 jsp쪽이 잘못됐는지.

1 @Controller, @RequestMapping

✓ 확인

○ <http://localhost:8080/sample>

○ `src/main/resources/log4j2.xml`

`<!-- Logger 설정 -->`

`<Loggers>`

`<Root level="DEBUG">` 로그 레벨을 info에서 debug으로 한단계 낮춰보고 서버를 재기동해보자

`<AppenderRef ref="console"/>`

`</Root>`

DEBUG org.springframework.web.servlet.HandlerMapping.Mappings(detectHandlerMethods:295) -

o.s.c.HomeController:

{GET [/]}: home()

이 녀석이 나오는지 확인해봐라. /url이랑 home메소드랑 연결되어있다

DEBUG org.springframework.web.servlet.HandlerMapping.Mappings(detectHandlerMethods:295) -

o.s.e.c.SampleController:

{[/sample]}: basic()

/sample url이랑 basic메소드랑 연결되어있다

DEBUG org.springframework.web.servlet.HandlerMapping.Mappings(detectHandlers:86) - 'beanNameHandlerMapping' {}

DEBUG org.springframework.web.servlet.HandlerMapping.Mappings(logMappings:177) - 'resourceHandlerMapping' {/resources/**=ResourceHttpRequestHandler [ServletContext [/resources/]]}

INFO org.springframework.web.servlet.DispatcherServlet(initServletBean:547) - Completed initialization in 641 ms

제대로 요청 url이 잘 매핑 등록됐는지 확인해보는 방법

다른 디버그 메세지도 너무 많음. 저 매핑 정보만 확인하고 싶은데. 저걸을 다시 info레벨로 고치고. 저 메세지가 나는 클래스만 따로 지정하면 저 문장만 출력되게 할 수 있다

1 @Controller, @RequestMapping

URL 맵핑 정보만 로그로 출력하기

<!-- Logger 설정 -->

<Loggers>

<Root level="INFO">

<AppenderRef ref="console"/>

</Root>

<Logger name="_org.springframework.web.servlet.HandlerMapping.Mappings" level="DEBUG" additivity="false" >

<AppenderRef ref="console"/>

</Logger>

<Logger name="org.scoula" level="INFO" additivity="false" >

<AppenderRef ref="console"/>

</Logger>

<Logger name="org.springframework" level="INFO" additivity="false">

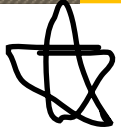
<AppenderRef ref="console"/>

</Logger>

</Loggers>

어노테이션 특 속성 지정 없이 하나 적으면 디폴트 속성 값이 됨. 속성을 2개 이상적은 경우는 생략 없이 다 적어야함. 어노테이션에서 중괄호는 배열

2 @RequestMapping의 변환



✓ @RequestMapping

- 수용할 HTTP method 설정
- GET, POST, PUT, DELETE 등

`@RequestMapping(value="/basic", method= {RequestMethod.GET, RequestMethod.POST})`

- 설정하지 않으면 모두 허용

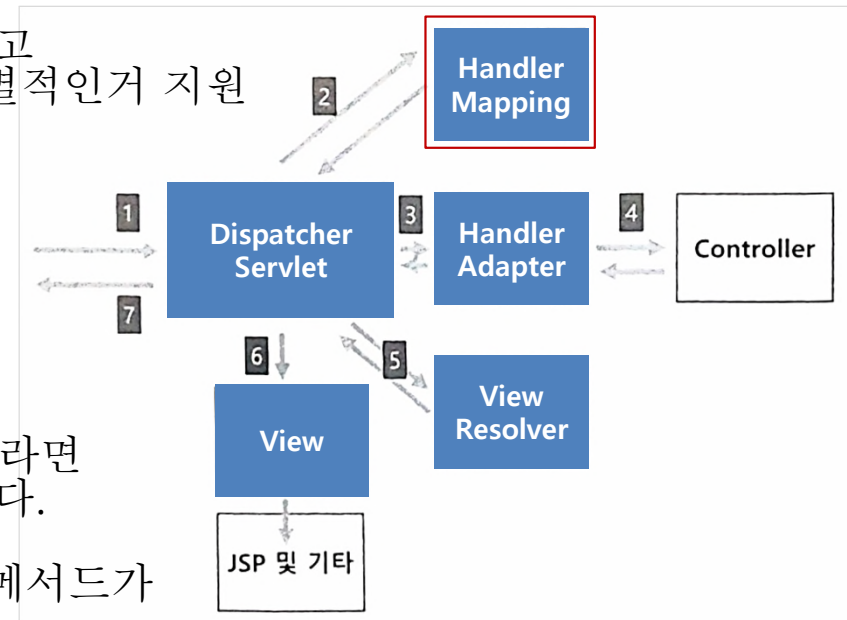
✓ 메서드별 Mapping 어노테이션

- 일반적으로 한 가지 메서드로 매핑함
- @GetMapping
- @PostMapping
- @PutMapping
- @DeleteMapping

`@GetMapping("/basicOnlyGet")` ✓
`public void xxx() { ... }`

이렇게 적기 길고 귀찮으니까 개별적인거 지원

잘못된 메서드라면 405에러가 난다.
메서드오류.
url은 맞는데 메서드가 다를때



2 @RequestMapping의 변환

SampleController.java

```

...
@Controller
@RequestMapping("/sample")
@Log4j
public class SampleController {

    @RequestMapping("") // url: /sample
    public void basic() {
        log.info("basic.....");
    }

    @RequestMapping(value="/basic", method= {RequestMethod.GET, RequestMethod.POST}) // url: /sample/basic
    public void basicGet(){
        log.info("basic get.....");
    }

    @GetMapping("/basicOnlyGet") // url: /sample/basicOnlyGet
    public void basicGet2(){
        log.info("basic get only get.....");
    }

}

```

뷰이름이
sample/basic이 되어버려.

이것도 반환이 없기에

뷰이름이 sample/basiconlyget임.

404jsp에러 나겠지 안만들었으니까

2 @RequestMapping의 변환

✓ **로그 확인** 아까 지정한 로그로 매핑 정보를 확인해보자

```
DEBUG _org.springframework.web.servlet.HandlerMapping.Mappings(detectHandlerMethods:295) -  
    o.s.c.HomeController:  
        {GET [/]}: home()  
DEBUG _org.springframework.web.servlet.HandlerMapping.Mappings(detectHandlerMethods:295) -  
    o.s.e.c.SampleController:  
        { [/sample]}: basic()  
        {[GET, POST] [/sample/basic]}: basicGet()  
        {GET [/sample/basicOnlyGet]}: basicGet2()
```

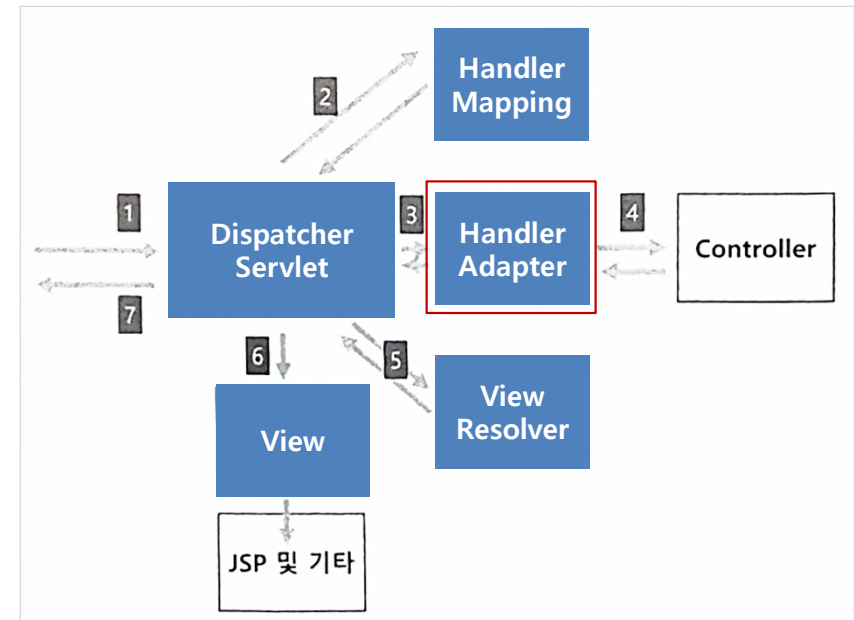
3 Controller의 파라미터 수집

✓ 요청의 쿼리 파라미터 수집

- `request.getParameter()`를 대체 ↓ 기존 방법. 다양한 후처리가 필요했음. 스프링은 간단하게 바뀜
- 컨트롤러 메서드의 매개변수에 필요한 쿼리 파라미터 매핑 지정
 - DTO 객체 사용
 - `HandlerAdapter`가 요청한 DTO 객체를 생성
 - Request의 파라미터/body에서 DTO의 프로퍼티를 설정해서 주입
 - `@RequestParam("파라미터명")` 어노테이션

혹은

이를 핸들러 어댑터가 함.



3 Controller의 파라미터 수집

SampleDTO.java

```
package org.scoula.ex03.dto;
```

```
import lombok.Data;
```

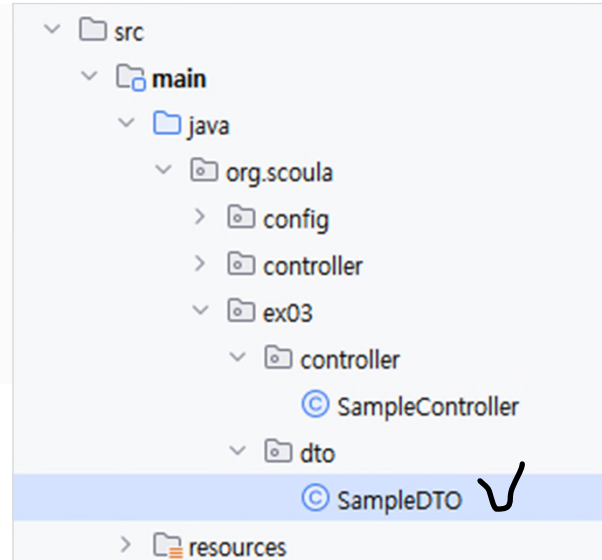
```
@Data ✓
```

```
public class SampleDTO {  
    private String name;  
    private int age;  
}
```

디폴트 생성자
반드시 있어야 한다고
했지?

그리고 세터도!

지금은 자동으로
추가된 디폴트 생성자

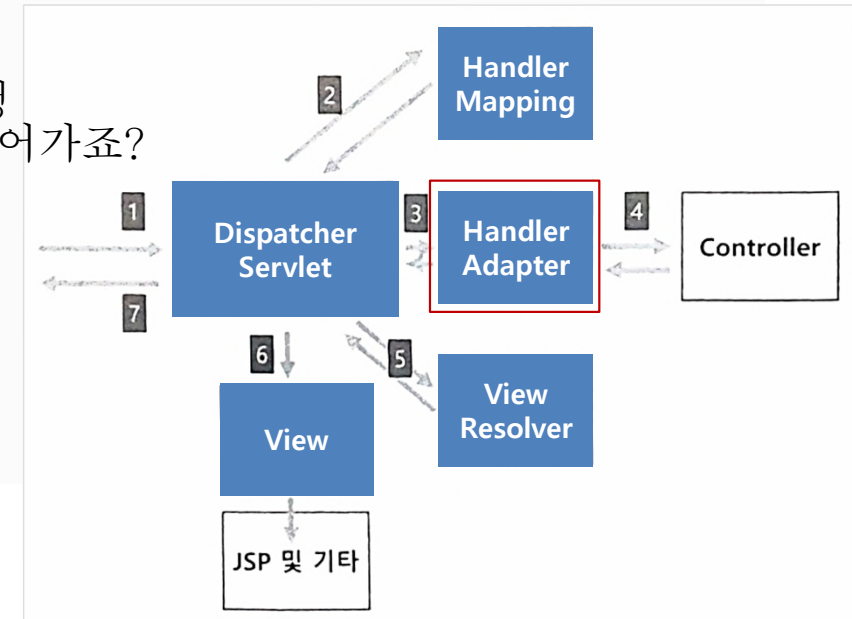


3 Controller의 파라미터 수집

SampleController.java

```
...
public class SampleController {
    ...
    @GetMapping("/ex01")
    public String ex01(SampleDTO dto) {
        log.info("'" + dto);
        return "ex01";
    }
}
```

내가 필요한 객체를
메서드의 매개변수로 요청
DI요청이죠? 자동으로 들어가요?



★ HandlerAdapter가 SampleDTO의 디폴트 생성자를 이용해 생성

- DTO 객체는 반드시 기본 생성자가 있어야 함. ✓

○ SampleDTO의 setter 메서드에서 프로퍼티명 유추

- setName() → name, setAge() → age ✓

★ 프로퍼티와 동일한 이름의 요청 파라미터를 추출

- request.getParameter("name"), request.getParameter("age")

○ 해당 파라미터가 존재하면 setter를 이용해 DTO의 프로퍼티 설정 ✓

○ 컨트롤러의 요청 처리 메서드의 인자로 전달

쿼리가 name=XXX&age=10000

이면

setName, setAge 메소드 없으면 넘어가고 있으면
프로퍼티 값을 지정함.

3 Controller의 파라미터 수집

✓ 확인

http://localhost:8080/sample/ex01

INFO org.scoula.ex03.controller.SampleController(ex01:34) - SampleDTO(name=null, age=0) ✓

http://localhost:8080/sample/ex01?name=AAA&age=10

setName(AAA), setAge(10)

INFO : org.scoula.ex03.controller.SampleController(ex01:34) - SampleDTO(name=AAA, age=10) ✓

개발자의 노고를 줄여주는 기능

3 Controller의 파라미터 수집

SampleController.java

아니 DTO 말고 그냥 변수에다가 담고 싶다면???

```
...
public class SampleController {
```

```
...
```

```
@GetMapping("/ex02")
```

```
public String ex02(
```

```
    @RequestParam("name") String name,
```

```
    @RequestParam("age") int age) {
```

```
    log.info("name: " + name);
```

```
    log.info("age: " + age);
```

```
    return "ex02";
```

```
}
```

```
}
```

매개변수 name, age 변수에
쿼리스트링에서 name 값과 age 값 찾아서 넣어줘

```
http://localhost:8080/sample/ex02?name=AAA&age=10
```

```
INFO org.scoula.ex03.controller.SampleController(ex02:44) - name: AAA
```

```
INFO org.scoula.ex03.controller.SampleController(ex02:45) - age: 10
```

만약 부분만 있다면? 혹은 age에 변환안되는
값이 들어간다면?

여기서 예외가 발생할 수 있음

웬만하면 값이 없어도 null이 나와 예외가
발생되는 것을 피하도록

primitive 타입을 wrapper 타입으로 받아라
int 말고 Integer로

3 Controller의 파라미터 수집

✓ 리스트 배열 처리

- 동일한 이름의 파라미터가 여러 개 전달되는 경우 ✓
- ArrayList<> 나 배열 사용

체크 박스 같은거

3 Controller의 파라미터 수집

SampleController.java

```
...  
public class SampleController {  
    ...  
  
    @GetMapping("/ex02List")  
    public String ex02List(@RequestParam("ids") ArrayList<String> ids) {  
        log.info("ids: " + ids);  
  
        return "ex02List";  
    }  
}
```

값이 없다면 비어있는 arraylist

http://localhost:8080/sample/ex02List?ids=111&ids=222&ids=333

INFO org.scoula.ex03.controller.SampleController(ex02List:54) - ids: [111, 222, 333] ✓

3 Controller의 파라미터 수집

SampleController.java

```
...  
public class SampleController {  
    ...
```

```
    @GetMapping("/ex02Array")  
    public String ex02Array(@RequestParam("ids") String[] ids) {  
        log.info("array ids: " + Arrays.toString(ids));  
  
        return "ex02Array";  
    }  
}
```



arraylist 말고 배열도 가능

http://localhost:8080/sample/ex02Array?ids=111&ids=222&ids=333

INFO org.scoula.ex03.controller.SampleController(ex02Array:62) - array ids: [111, 222, 333]



3 Controller의 파라미터 수집

✓ 객체 리스트

- 객체 리스트를 쿼리 파라미터로 받기

주로 ajax통신할때 form통신 말고.

3 Controller의 파라미터 수집

SampleDTOList.java

```
package org.scoula.ex03.dto;
```

```
import lombok.Data;
```

```
import java.util.ArrayList;
```

```
import java.util.List;
```

```
@Data
```

```
public class SampleDTOList {
```

```
    private List<SampleDTO> list;
```

```
    public SampleDTOList() {
```

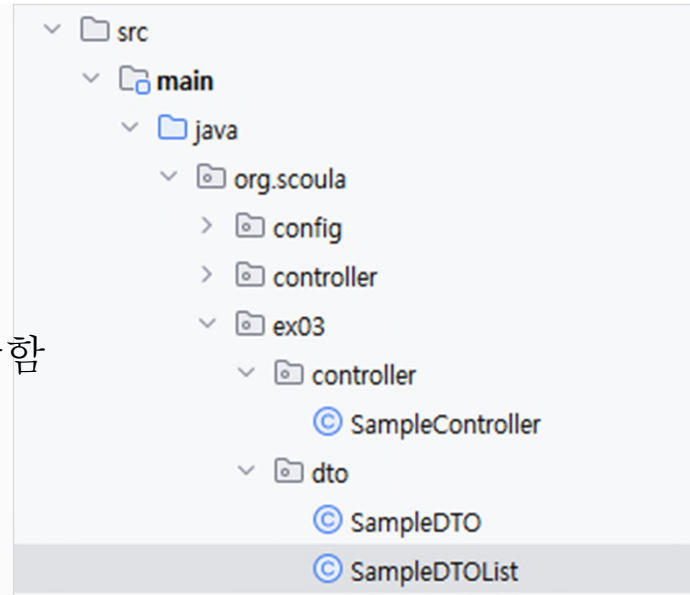
```
        list = new ArrayList<>();
```

```
    }
```

```
}
```

form태그 통신 말고

rest, js통신할때 등장함



3 Controller의 파라미터 수집

SampleController.java

```
...
public class SampleController {
    ...

    @GetMapping("/ex02Bean")
    public String ex02Bean(SampleDTOList list) {
        log.info("list dtos: " + list);

        return "ex02Bean";
    }
}
```

http://localhost:8080/sample/ex02Bean?list[0].name=aaa&list[2].name=bbb

[,]문자 때문에 에러 발생

http://localhost:8080/sample/ex02Bean?list%5B0%5D.name=aaa&list%5B2%5D.name=bbb

INFO org.scoula.ex03.controller.SampleController(ex02Bean:70) - list dtos: SampleDTOList(list=[SampleDTO(name=aaa, age=0), SampleDTO(name=null, age=0), SampleDTO(name=bbb, age=0)])

3 Controller의 파라미터 수집

✓ @DateTimeFormat

- 커스텀 바인딩이 가장 많이 사용되는 타입 → Date ✓
- DTO 객체의 Date 타입 필드에 @DateTimeFormat을 지정

```
@DateTimeFormat(pattern="yyyy-MM-dd")  
private Date reg_date;
```

string으로 된 날짜를
Date로 변환해야하는데

패턴을 제시해야함

이때 사용하는 어노테이션.

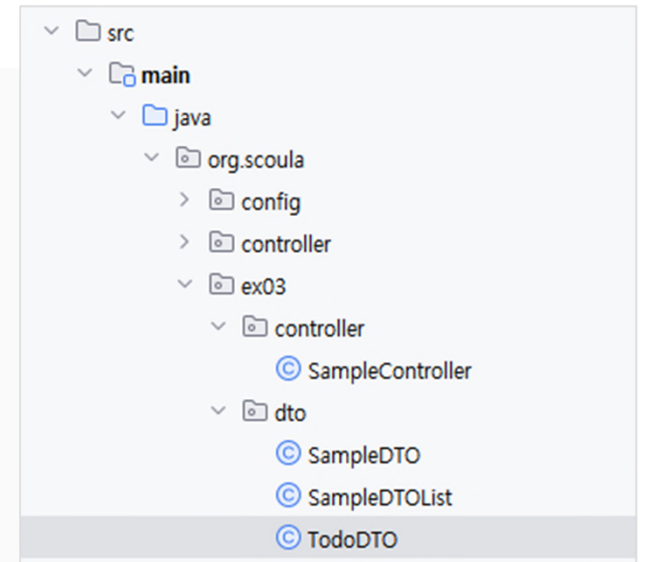
@DateTimeFormat

파싱해서 Date로 변환해줌.

3 Controller의 파라미터 수집

TodoDTO.java

```
package org.scoula.ex03.dto;  
  
import java.util.Date;  
  
import org.springframework.format.annotation.DateTimeFormat;  
  
import lombok.Data;  
  
@Data  
public class TodoDTO {  
    private String title;  
  
    @DateTimeFormat(pattern="yyyy/MM/dd")  
    private Date dueDate;  
  
}
```



3 Controller의 파라미터 수집

SampleController.java

```
...  
public class SampleController {  
    ...  
    @GetMapping("/ex03")  
    public String ex03(TodoDTO todo) {  
        log.info("todo: " + todo);  
        return "ex03";  
    }  
}
```

DI요구함.

setTitle호출됨. setDate(형변환 "2023/01/01")

<http://localhost:8080/sample/ex03?title=test&dueDate=2023/01/01>

INFO org.scoula.ex03.controller.SampleController(ex03:78) - todo: TodoDTO(title=test, dueDate=Sun Jan 01 00:00:00 KST 2023)