

스프링 시작하기

스프링 API 12장



1.1 application.yml

dev,prod 설정

```
server:
  port: 8080
  servlet:
    context-path: /

spring:
  # todo 개발 데이터베이스 연결
  datasource:
    url: jdbc:h2:mem:test
    username: sa
    password:
    driver-class-name: org.h2.Driver

  jpa:
    hibernate:
      ddl-auto: create
    show-sql: true # 콘솔창에 sql 출력
    properties:
      hibernate:
        format_sql: true # sql 예쁘게 출력
        default_batch_fetch_size: 500 # 일대다 컬렉션 조회 시 성능 최적화
    open-in-view: false # 영속성 컨텍스트의 생존 범위를 트랜잭션 범위로 한정

  servlet:
    multipart:
      max-file-size: 10MB # 파일 업로드 요청 시 하나의 파일 크기를 10MB 제한
      max-request-size: 100MB # 파일 업로드 요청 시 모든 파일 크기합을 100MB 제한
```

1.2 application.yml

```
logging:  
  level:  
    org.hibernate.type: trace # 콘솔창에 조건에 바인딩되는 값 및 조회 결과 출력  
    com.app: debug # todo 패키지 수정
```

SQL 구문에 바인딩 된 값 처리

1.2 Environment 체크

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.core.env.Environment;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;

@Controller
public class ProfileController {

    @Autowired
    private Environment environment;

    @GetMapping("/profile")
    public String showProfile(Model model) {
        String[] activeProfiles = environment.getActiveProfiles();
        model.addAttribute("profiles", activeProfiles);
        return "profile";
    }
}
```



1.3 프로젝트 생성

Edit Config

The screenshot shows the 'Run/Debug Configurations' dialog in IntelliJ IDEA. On the left, a tree view shows 'Spring Boot' expanded, with 'SpringApiAppApplication' selected. The main panel on the right is titled 'Run/Debug Configurations' and contains the following settings:

- Name:** SpringApiAppApplication. There is a checkbox 'Store as project file' which is unchecked.
- Run on:** Local machine (selected from a dropdown). A link 'Manage targets...' is next to it.
- A note: 'Run configurations may be executed locally or on a target: for example in a Docker Container or on a remote host using SSH.'
- Build and run** section, with a 'Modify options' link and a keyboard shortcut icon (⌘M).
- VM options:** java 17 SDK of 'spring-api' (selected from a dropdown) and -cp spring-api-app.main (selected from a dropdown).
- Main class:** com.app.SpringApiAppApplication (with a list icon on the right).
- A hint: 'Press ⌘ for field hints'.
- Active profiles:** An empty text field with a hint 'Comma separated list of profiles' below it.
- Two toggle buttons at the bottom:
 - 'Open run/debug tool window when started' (checked, with a close button).
 - 'Add dependencies with "provided" scope to classpath' (checked, with a close button).

At the bottom of the dialog, there is a bar with a help icon (?), a 'Run' button (with a green play icon), a dropdown arrow, 'Cancel', 'Apply', and 'OK' buttons. In the bottom-left corner of the IDE window, there is a link 'Edit configuration templates...'.

1.4 spring cloud openfeign

```
ext {  
    set('springCloudVersion', "2023.0.0")  
}
```

```
spring:  
  cloud:  
    openfeign:  
      client:  
        config:  
          default:  
            connectTimeout: 5000  
            readTimeout: 5000
```

```
@Service  
@RequiredArgsConstructor  
public class ExampleService {  
    1 usage  
    private final ExampleClient exampleClient;  
  
    1 usage  
    public String getResource() {  
        return exampleClient.getResource();  
    }  
}
```

```
package com.example.mhapi.mh;  
  
import lombok.RequiredArgsConstructor;  
import org.springframework.http.ResponseEntity;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.RestController;  
  
@RestController  
@RequiredArgsConstructor  
public class ExampleController {  
    1 usage  
    private final ExampleService exampleService;  
  
    @GetMapping("/mh/test")  
    public ResponseEntity<String> mhtest(){  
  
        String result = exampleService.getResource();  
  
        return ResponseEntity.ok(result);  
    }  
  
    @GetMapping("/testjson")  
    public ResponseEntity<String> testjson(){  
        return ResponseEntity.ok(body: "qweqwe");  
    }  
}
```

1.4 spring cloud openfeign

```
import org.springframework.cloud.openfeign.FeignClient;
import org.springframework.web.bind.annotation.GetMapping;

1 usage
@FeignClient(name = "test", url = "https://www.dhlottery.co.kr/common.do?method=getLottoNumber&drwNo=861")
public interface ExampleClient {

    1 usage
    @GetMapping("")
    String getResource();
}
```

<https://www.dhlottery.co.kr/common.do?method=getLottoNumber&drwNo=861>

1.4 spring cloud openfeign

2 usages

@Slf4j

```
public class FeignClientExceptionHandler implements ErrorDecoder {
```

1 usage

```
private ErrorDecoder errorDecoder = new Default();
```

@Override

```
public Exception decode(String methodKey, Response response) {
```

```
    log.error("{} 요청 실패, status : {}, reason : {}", methodKey, response.status(), response.reason());
```

```
    FeignException exception = FeignException.errorStatus(methodKey, response);
```

```
    HttpStatus httpStatus = HttpStatus.valueOf(response.status());
```

```
    if(httpStatus.is5xxServerError()) {
```

```
        return new RetryableException(
```

```
            response.status(),
```

```
            exception.getMessage(),
```

```
            response.request().httpMethod(),
```

```
            exception,
```

```
            null,
```

```
            response.request()
```

```
        );
```

```
    }
```

```
    return errorDecoder.decode(methodKey, response);
```

```
}
```

```
}
```


1.4 실행결과

소스 참고

<https://github.com/dron512/feignClient>

```
localhost:8082/mh/test

dron512/git231229 dron512/AI_305_7_... codings

1 // 20240303083154
2 // http://localhost:8082/mh/test
3
4 {
5   "totSellamnt": 81032551000,
6   "returnValue": "success",
7   "drwNoDate": "2019-06-01",
8   "firstWinamnt": 4872108844,
9   "drwtNo6": 25,
10  "drwtNo4": 21,
11  "firstPrzwnerCo": 4,
12  "drwtNo5": 22,
13  "bnusNo": 24,
14  "firstAccumamnt": 19488435376,
15  "drwNo": 861,
16  "drwtNo2": 17,
17  "drwtNo3": 19,
18  "drwtNo1": 11
19 }
```

고맙습니다.

