Feign Client

▼ 1. Feign이란?

- Fegin Client Binder로 Netflix에서 개발
- 스프링 환경에서 간편하게 외부 api를 호출할 수 있는 라이브러리
  - RestTemplate 보다 훨씬 더 간편하게 api를 호출 할 수 있다.

- Feign을 사용하기 위해서는 annotation을 선언하고 interface를 작성
  - Spring Data JPA 에서 실제 쿼리를 작성하지 않고 Interface만 작성하여 구현체를 자동으로 만들어주는 것과 유사
  - Interface 선언을 통해 자동으로 Http Client를 생성

> Feign은 Inteface 그 자체로 ***클라이언트(사용자) 역할***을 한다.

- Document

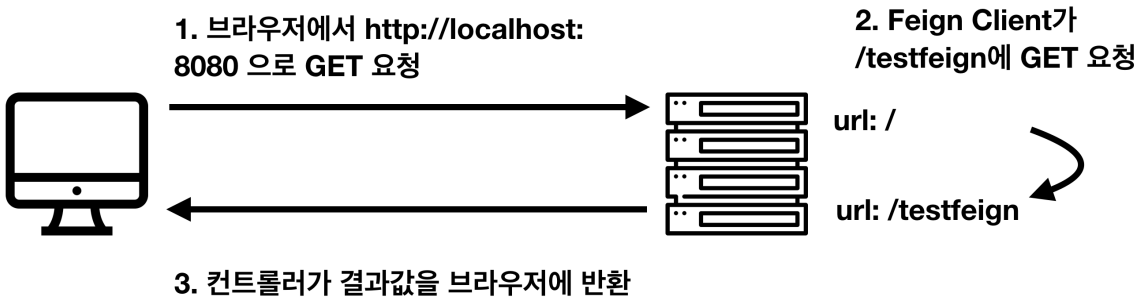
Spring Cloud OpenFeign

```
@SpringBootApplication
@EnableFeignClients public class
WebApplication { public static void
```

https://spring.io/projects/spring-cloud-openfeign

https://cloud.spring.io/spring-cloud-netflix/multi/multi_spring-cloud-feign.html

▼ 2. Feign Client 간단 동작 순서



> 하나의 프로젝트에서 진행 했을 경우

▼ 3. Feign Client 세팅

(예시. Spring boot에서 진행)

- build.gradle setting

```
ext {
  set('springCloudVersion', "2021.0.3")
}

dependencyManagement {
  imports {
    mavenBom "org.springframework.cloud:spring-cloud-dependencies:${springCloudVersion}"
  }
}

apply plugin: "io.spring.dependency-management"

dependencies {
  // feign client 버전 확인
  // https://spring.io/projects/spring-cloud 의 Release Trains
  implementation 'org.springframework.cloud:spring-cloud-starter-openfeign'
}
```

- ProjectApplication.java (Main Method 가 포함된 Class)

```
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.cloud.openfeign.EnableFeignClients;

@EnableFeignClients
@SpringBootApplication
public class DemoApplication {

  public static void main(String[] args) {
    SpringApplication.run(DemoApplication.class, args);
  }

}
```

▼ 4. Feign Client 실습 진행

localhost:8080

- MainController

```
@RestController
@RequestMapping("/math")
public class MainController {

    private int randomNumber = 0;

    @GetMapping("/random")
    public int getRandomNumber() {
        this.randomNumber = (int)(Math.random() * 9 + 1);
        return this.randomNumber;
    }

    @GetMapping("/random/sum/{number}")
    public int sumRandomNumber(@PathVariable int number) {
        return this.randomNumber + number;
    }

}
```

- FeignController

```
@RestController
@RequestMapping("/feign")
@RequiredArgsConstructor
public class FeignController {

    final FeignInterface feignInterface;
```

```
    @GetMapping("/random")
    public int getRandomNumber() {
        return feignInterface.getRandomNumber();
    }

    @GetMapping("/random/sum/{number}")
    public int sumRandomNumber(@PathVariable int number) {
        return feignInterface.sumRandomNumber(number);
    }

}
```

- Feign Interface

```
@FeignClient(name = "number", url = "localhost:8080")
public interface FeignInterface {

    @RequestMapping("/math/random")
    int getRandomNumber();

    @RequestMapping("/math/random/sum/{number}")
    int sumRandomNumber(@PathVariable int number);
}
```

Localhost:9090

- Controller

```
@RestController
@RequestMapping("/member")
@RequiredArgsConstructor
public class MainController {

    @GetMapping("/number")
    public int getNumber() {
        return 123;
    }

    @GetMapping("/Info")
    public MemberDTO getMemberInfo() {
        return new MemberDTO("plitche", "yskwon", 31);
    }

}
```

- MemberDTO

```
@Data
public class MemberDTO {

    private String id;
    private String name;
    private int age;

    public MemberDTO(String id, String name, int age) {
        this.id = id;
        this.name = name;
        this.age = age;
    }
}
```

- MemberService

```
@Service
public class MemberService {

    private List<MemberDTO> members = new ArrayList<>();

    public MemberService() {
        this.members.add(new MemberDTO("kim25", "kim", 25));
        this.members.add(new MemberDTO("lee31", "lee", 31));
        this.members.add(new MemberDTO("park35", "park", 35));
        this.members.add(new MemberDTO("kwon50", "kwon", 50));
    }

    public MemberDTO getMember(String id) {
        return this.members.stream()
                .filter(m -> m.getId().equals(id))
                .findFirst()
                .orElseThrow(() -> new IllegalArgumentException());
    }
}
```

▼ 5. Quiz

1. localhost:8080 port의 Project에서 Feign CLient Interface를 통해 localhost:9090 port의 Controller에 접근했을때에 Network 통신이 몇번 발생할까?

   ▼ 정답

   1번! 처음 8080의 Controller에 접근했을때의 통신만 발생한다.

2. localhost:8080에서 localhost:9090에 Controller에 접근 시 9090에서 오류가 발생하면 몇번대의 status가 return 될까?

   ▼ 정답

   500번대 Server Error

3. localhost:8080에서 localhost:9090에 없는 Mapping값의 Controller로 접근 시 몇번대의 status가 return 될까?

   ▼ 정답

   client Error status 400번대가 아닌 500번대 Server Error 발생
