초격차 패키지 Online.

실무를 고급지게 만들어주는 기능들

PART1 | Lombok 입문

아니 이걸 몰라?

PART2 | Spring Configuration Processor

설정값을 문서화해보자

PART3 | Spring Cache Abstraction

프로젝트 속도를 높이는 보편적 방법

PART4 | Vault Configuration

프로젝트 암호를 더 안전하게 보관하는 최신 트렌드

실무를 고급지게 만들어주는 기능들

3 Spring Cache Abstraction

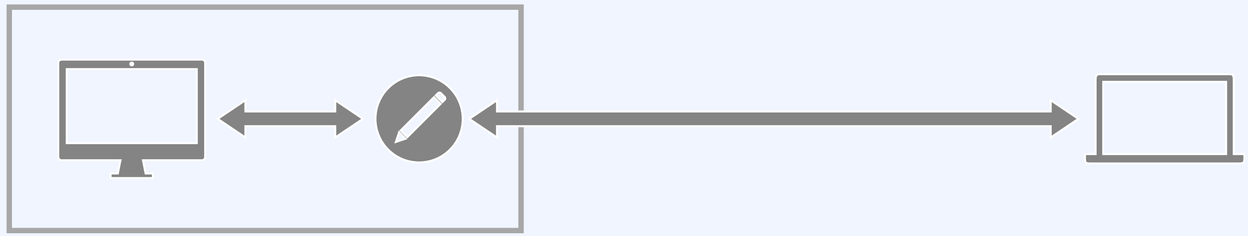
Spring Cache Abstraction

- 애플리케이션에 "투명하게(transparently)" 캐시를 넣어주는 기능
- 메소드, 클래스에 적용 가능
- 캐시 인프라는 스프링 부트 자동 설정으로 세팅되고, 프로퍼티로 관리 가능

> The Spring Framework provides support for transparently adding caching to an application. At its core, the abstraction applies caching to methods, thus reducing the number of executions based on the information available in the cache. The caching logic is applied transparently, without any interference to the invoker. Spring Boot auto-configures the cache infrastructure as long as caching support is enabled via the @EnableCaching annotation.

캐시가 시스템, 애플리케이션에 투명하게 자리잡는다는 말은..

- 데이터를 통신하는 시스템 쌍방이 캐시의 존재를 모른다는 의미
- "캐시가 있건 없건, 시스템의 기대 동작은 동일해야 한다."
- 캐시의 목표: 오로지 "성능"
- 캐시의 개념과 목적에 부합하는 성질이자, 조건



반복 작업이라면 고려해 보세요

- 잘 바뀌지 않는 정보를 외부 저장소에서 반복적으로 읽어온다면
- 기대값이 어차피 같다면
- 캐싱해서 성능 향상, I/O 감소

Spring Boot Starter Cache - 너무 쉬운 사용법


```
@RequiredArgsConstructor
@EnableCaching
@ConfigurationPropertiesScan
@SpringBootApplication
public class FastcampusSpringBootPracticeApplication {
```

```
@RequiredArgsConstructor
@Repository
public class StudentRepository {

    private final Map<String, Student> storage;

    @Cacheable("student")
    public Student getStudent(String name) {
        System.out.println("[repo] 나의 통행료는 무척 비싸다!");
        return storage.get(name);
    }

    public Student enroll(String name, Integer age, Student.Grade grade) {
        return storage.put(name, Student.of(name, age, grade));
    }

}
```

Spring Boot Starter Cache - 너무 쉬운 사용법


```
    @EventListener(ApplicationReadyEvent.class)
    public void init() {
//        System.out.println("내 키는: " + myProperties.getHeight());
        studentService.printStudent( name: "jack");
        studentService.printStudent( name: "jack");
        studentService.printStudent( name: "jack");
        studentService.printStudent( name: "fred");
        studentService.printStudent( name: "cassie");
        studentService.printStudent( name: "cassie");
    }
```

```
[repo] 나의 통행료는 무척 비싸다!
찾는 학생: Student(name=jack, age=15, grade=B)
찾는 학생: Student(name=jack, age=15, grade=B)
찾는 학생: Student(name=jack, age=15, grade=B)
[repo] 나의 통행료는 무척 비싸다!
찾는 학생: Student(name=fred, age=14, grade=C)
[repo] 나의 통행료는 무척 비싸다!
찾는 학생: Student(name=cassie, age=18, grade=A)
찾는 학생: Student(name=cassie, age=18, grade=A)
```

캐싱에서 생각해야 하는 것들

- 무엇을 캐시할까?
- 얼마나 오랫동안 캐시할까?
- 언제 캐시를 갱신할까?

주요 기능들

- @EnableCaching
- @Cacheable
- @CacheEvict
- @CachePut

Redis

Redis - 설정


```
dependencies {
    implementation 'org.springframework.boot:spring-boot-starter-cache'
//    implementation 'org.springframework.boot:spring-boot-starter-cache'
    implementation 'org.springframework.boot:spring-boot-starter-data-redis'
    implementation 'org.springframework.boot:spring-boot-starter-actuator'
    implementation 'org.springframework.boot:spring-boot-starter-web'
    testImplementation 'org.springframework.boot:spring-boot-starter-test'
    compileOnly 'org.projectlombok:lombok'
    annotationProcessor 'org.projectlombok:lombok'
    annotationProcessor 'org.springframework.boot:spring-boot-configuration-processor'
}
```

Redis - 설정

- 프로퍼티와 자바 코드를 이용한 설정

```
spring.cache.type=redis
spring.cache.redis.time-to-live=10s
spring.cache.redis.cache-null-values=false
spring.cache.redis.use-key-prefix=false

spring.redis.username=my
spring.redis.password=pass
spring.redis.host=localhost
spring.redis.port=6379
```

```
@Bean
public RedisCacheConfiguration redisCacheConfiguration() {
    return RedisCacheConfiguration.defaultCacheConfig()
            .computePrefixWith(name -> name + ":")
            .entryTtl(Duration.ofSeconds(10))
            .serializeValuesWith(SerializationPair.fromSerializer(new GenericJackson2JsonRedisSerializer()));
}
```

Reference

- https://docs.spring.io/spring-boot/docs/current/reference/html/features.html#features.caching
- https://spring.io/guides/gs/caching/