

SpringCache

参考视频或文章：

- <https://juejin.cn/post/7507548342508371983>

一、技术介绍

1.概述

- SpringCache是Spring框架提供的一套声明式缓存抽象层（**只提供接口，不提供实现**），通过在方法上添加注解简化缓存操作，无需手动编写缓存逻辑。
- SpringCache支持多种缓存实现，如Caffeine、Redis、EhCache等等，并统一了缓存访问的API。

2.核心特点

- 基于注解的声明式缓存
- 支持SpEL(Spring Expression Language)表达式
- 自动与Spring生态集成
- 支持条件缓存

3.常用缓存注解

缓存注解	功能
@EnableCaching	开启注解方式的缓存功能，通常加在项目启动类上。
@Cacheable	标记方法，将方法的返回值缓存，下次调用直接从缓存中读取，无需重新执行方法。
@CachePut	标记方法，将方法的返回值缓存。
@CacheEvict	标记方法，用于清除缓存，通常配合数据删除操作使用。
@Caching	组合多个缓存注解，支持在同一个方法上同时配置多种缓存行为。
@CacheConfig	标记类，为类中所有方法指定统一的缓存配置，减少重复配置。

二、项目应用

涉及到的文件如下：

```
1 sky-take-out: pom.xml
2
3 sky-server:
4   pom.xml
5   src/main/java/com.sky:
6     SkyApplication
7
```

1.导入和Redis和SpringCache的Maven依赖坐标

1.1 sky-take-out: pom.xml

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <project xmlns="http://maven.apache.org/POM/4.0.0"
3          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4          xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
5      <modelVersion>4.0.0</modelVersion>
6      <parent>
7          <artifactId>spring-boot-starter-parent</artifactId>
8          <groupId>org.springframework.boot</groupId>
9          <version>2.7.3</version>
10     </parent>
11
12     <groupId>com.sky</groupId>
13     <artifactId>sky-take-out</artifactId>
14     <packaging>pom</packaging>
15     <version>1.0-SNAPSHOT</version>
16
17     <modules>
18         <module>sky-common</module>
19         <module>sky-pojo</module>
20         <module>sky-server</module>
21     </modules>
22
23     <properties>
24         <druid>1.2.1</druid>
25     </properties>
26
27     <dependencyManagement>
28         <dependencies>
29             <dependency>
30                 <groupId>com.alibaba</groupId>
31                 <artifactId>druid-spring-boot-starter</artifactId>
32                 <version>${druid}</version>
33             </dependency>
34         </dependencies>
35     </dependencyManagement>
36 </project>
```

1.2 sky-server: pom.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0"
3         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
5     <parent>
6         <artifactId>sky-take-out</artifactId>
7         <groupId>com.sky</groupId>
8         <version>1.0-SNAPSHOT</version>
9     </parent>
10
11     <modelVersion>4.0.0</modelVersion>
12     <artifactId>sky-server</artifactId>
13
14     <dependencies>
15         <dependency>
16             <groupId>com.sky</groupId>
17             <artifactId>sky-common</artifactId>
18             <version>1.0-SNAPSHOT</version>
19         </dependency>
20         <dependency>
21             <groupId>com.sky</groupId>
22             <artifactId>sky-pojo</artifactId>
23             <version>1.0-SNAPSHOT</version>
24         </dependency>
25
26         <dependency>
27             <groupId>org.springframework.boot</groupId>
28             <artifactId>spring-boot-starter</artifactId>
29         </dependency>
30         <dependency>
31             <groupId>org.springframework.boot</groupId>
32             <artifactId>spring-boot-starter-test</artifactId>
33             <scope>test</scope>
34         </dependency>
35         <dependency>
36             <groupId>org.springframework.boot</groupId>
37             <artifactId>spring-boot-starter-web</artifactId>
38             <scope>compile</scope>
39         </dependency>
40
41         <dependency>
```

```
42         <groupId>mysql</groupId>
43         <artifactId>mysql-connector-java</artifactId>
44         <scope>runtime</scope>
45     </dependency>
46
47     <dependency>
48         <groupId>com.alibaba</groupId>
49         <artifactId>druid-spring-boot-starter</artifactId>
50     </dependency>
51
52     <dependency>
53         <groupId>org.springframework.boot</groupId>
54         <artifactId>spring-boot-starter-data-redis</artifactId>
55     </dependency>
56
57     <dependency>
58         <groupId>org.springframework.boot</groupId>
59         <artifactId>spring-boot-starter-cache</artifactId>
60     </dependency>
61 </dependencies>
62
63 <build>
64     <plugins>
65         <plugin>
66             <groupId>org.springframework.boot</groupId>
67             <artifactId>spring-boot-maven-plugin</artifactId>
68         </plugin>
69     </plugins>
70 </build>
71
72 </project>
```

2.在项目启动类 `SkyApplication` 上开启缓存注解功能

```
1 @SpringBootApplication
2 @EnableTransactionManagement// 开启注解方式的事务管理
3 @EnableCaching// 开启注解方式的缓存功能
4 public class SkyApplication {
5     public static void main(String[] args) {
6         SpringApplication.run(SkyApplication.class, args);
7     }
8 }
```

3.编写 `user/SetmealController`，使用 `@Cacheable` 更新缓存

```
1 /**
2  * 用户端-套餐接口
3  */
4 @RestController("userSetmealController")
5 @RequestMapping("/user/setmeal")
6 public class SetmealController {
7
8     @Autowired
9     private SetmealService setmealService;
10    @Autowired
11    private DishService dishService;
12
13    // 根据分类id查询已启用的套餐
14    @GetMapping("/list")
15    @Cacheable(cacheNames = "setmeal", key = "#categoryId")// key名称:
    setmeal:::{categoryId}
16    public Result<List<Setmeal>> getByCategoryId(Long categoryId) {
17        List<Setmeal> setmeals =
18        setmealService.getByCategoryId(categoryId);
19        return Result.success(setmeals);
20    }
21 }
```

4.编写 `admin/SetmealController`，使用 `@CacheEvict` 清除缓存

```
1  /**
2   * 套餐管理模块
3   */
4  @RestController("adminSetmealController")
5  @RequestMapping("/admin/setmeal")
6  public class SetmealController {
7
8      @Autowired
9      private SetmealService setmealService;
10
11     // 新增套餐和对应的菜品
12     @PostMapping
13     @CacheEvict(cacheNames = "setmeal", key =
"#setmealDTO.categoryId")// 清理缓存
14     public Result saveWithDish(@RequestBody SetmealDTO setmealDTO) {
15         setmealService.saveWithDish(setmealDTO);
16         return Result.success();
17     }
18
19
20     // 批量删除套餐
21     @DeleteMapping
22     @CacheEvict(cacheNames = "setmeal", allEntries = true)// 清理全部缓
存
23     public Result deleteBatch(@RequestParam List<Long> ids) {
24         setmealService.deleteBatch(ids);
25         return Result.success();
26     }
27
28
29     // 修改套餐
30     @PutMapping
31     @CacheEvict(cacheNames = "setmeal", allEntries = true)// 清理全部缓
存
32     public Result update(@RequestBody SetmealDTO setmealDTO) {
33         setmealService.updateWithDishes(setmealDTO);
```

```
34         return Result.success();
35     }
36
37
38     // 起售停售套餐
39     @PostMapping("/status/{status}")
40     @CacheEvict(cacheNames = "setmeal", allEntries = true)// 清理全部缓
存
41     public Result changeStatus(@PathVariable Integer status, Long id) {
42         setmealService.changeStatus(status, id);
43         return Result.success();
44     }
45
46 }
```