

# 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Spring Cache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
5             http://maven.apache.org/xsd/maven-4.0.0.xsd">
6     <modelVersion>4.0.0</modelVersion>
7     <parent>
8         <artifactId>spring-boot-starter-parent</artifactId>
9         <groupId>org.springframework.boot</groupId>
10        <version>2.7.3</version>
11    </parent>
12
13    <groupId>com.sky</groupId>
14    <artifactId>sky-take-out</artifactId>
15    <packaging>pom</packaging>
16    <version>1.0-SNAPSHOT</version>
17
18    <modules>
19        <module>sky-common</module>
20        <module>sky-pojo</module>
21        <module>sky-server</module>
22    </modules>
23
24    <properties>
25        <druid>1.2.1</druid>
26    </properties>
27
28    <dependencyManagement>
29        <dependencies>
30            <dependency>
31                <groupId>com.alibaba</groupId>
32                <artifactId>druid-spring-boot-starter</artifactId>
33                <version>${druid}</version>
34            </dependency>
35        </dependencies>
36    </dependencyManagement>
</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
5             http://maven.apache.org/xsd/maven-4.0.0.xsd">
6     <parent>
7         <artifactId>sky-take-out</artifactId>
8         <groupId>com.sky</groupId>
9         <version>1.0-SNAPSHOT</version>
10    </parent>
11
12    <modelVersion>4.0.0</modelVersion>
13    <artifactId>sky-server</artifactId>
14
15    <dependencies>
16        <dependency>
17            <groupId>com.sky</groupId>
18            <artifactId>sky-common</artifactId>
19            <version>1.0-SNAPSHOT</version>
20        </dependency>
21        <dependency>
22            <groupId>com.sky</groupId>
23            <artifactId>sky-pojo</artifactId>
24            <version>1.0-SNAPSHOT</version>
25        </dependency>
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        <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:
16    setmeal::{categoryId}
17    public Result<List<Setmeal>> getByCategoryId(Long categoryId) {
18        List<Setmeal> setmeals =
19        setmealService.getByCategoryId(categoryId);
20        return Result.success(setmeals);
21    }
22 }
```

## 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 =
14         "#setmealDTO.categoryId")// 清理缓存
15     public Result saveWithDish(@RequestBody SetmealDTO setmealDTO) {
16         setmealService.saveWithDish(setmealDTO);
17         return Result.success();
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     // 修改套餐
29     @PutMapping
30     @CacheEvict(cacheNames = "setmeal", allEntries = true)// 清理全部缓存
31     public Result update(@RequestBody SetmealDTO setmealDTO) {
32         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 }
```