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：

```
sky-take-out: pom.xml

sky-server:
    pom.xml
    src/main/java/com.sky:
        SkyApplication

```

1.导入和Redis和SpringCache的`Maven`依赖坐标

1.1 `sky-take-out: pom.xml`

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <artifactId>spring-boot-starter-parent</artifactId>
        <groupId>org.springframework.boot</groupId>
        <version>2.7.3</version>
    </parent>

    <groupId>com.sky</groupId>
    <artifactId>sky-take-out</artifactId>
    <packaging>pom</packaging>
    <version>1.0-SNAPSHOT</version>

    <modules>
        <module>sky-common</module>
        <module>sky-pojo</module>
        <module>sky-server</module>
    </modules>

    <properties>
        <druid>1.2.1</druid>
    </properties>

    <dependencyManagement>
        <dependencies>
            <dependency>
                <groupId>com.alibaba</groupId>
                <artifactId>druid-spring-boot-starter</artifactId>
                <version>${druid}</version>
            </dependency>
        </dependencies>
    </dependencyManagement>
</project>
```

1.2 `sky-server: pom.xml`

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <parent>
        <artifactId>sky-take-out</artifactId>
        <groupId>com.sky</groupId>
        <version>1.0-SNAPSHOT</version>
    </parent>

    <modelVersion>4.0.0</modelVersion>
    <artifactId>sky-server</artifactId>

    <dependencies>
        <dependency>
            <groupId>com.sky</groupId>
            <artifactId>sky-common</artifactId>
            <version>1.0-SNAPSHOT</version>
        </dependency>
        <dependency>
            <groupId>com.sky</groupId>
            <artifactId>sky-pojo</artifactId>
            <version>1.0-SNAPSHOT</version>
        </dependency>

        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
            <scope>compile</scope>
        </dependency>

        <dependency>
```

```xml
                <groupId>mysql</groupId>
                <artifactId>mysql-connector-java</artifactId>
                <scope>runtime</scope>
            </dependency>

            <dependency>
                <groupId>com.alibaba</groupId>
                <artifactId>druid-spring-boot-starter</artifactId>
            </dependency>

            <dependency>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-starter-data-redis</artifactId>
            </dependency>

            <dependency>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-starter-cache</artifactId>
            </dependency>
        </dependencies>

        <build>
            <plugins>
                <plugin>
                    <groupId>org.springframework.boot</groupId>
                    <artifactId>spring-boot-maven-plugin</artifactId>
                </plugin>
            </plugins>
        </build>

    </project>
```

2.在项目启动类`SkyApplication`上开启缓存注解功能

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

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

```java
/**
 * 用户端-套餐接口
 */
@RestController("userSetmealController")
@RequestMapping("/user/setmeal")
public class SetmealController {

    @Autowired
    private SetmealService setmealService;
    @Autowired
    private DishService dishService;

    // 根据分类id查询已启用的套餐
    @GetMapping("/list")
    @Cacheable(cacheNames = "setmeal", key = "#categoryId")// key名称：setmeal::{categoryId}
    public Result<List<Setmeal>> getByCategoryId(Long categoryId) {
        List<Setmeal> setmeals = setmealService.getByCategoryId(categoryId);
        return Result.success(setmeals);
    }

}
```

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

```java
/**
 * 套餐管理模块
 */
@RestController("adminSetmealController")
@RequestMapping("/admin/setmeal")
public class SetmealController {

    @Autowired
    private SetmealService setmealService;

    // 新增套餐和对应的菜品
    @PostMapping
    @CacheEvict(cacheNames = "setmeal", key = "#setmealDTO.categoryId")// 清理缓存
    public Result saveWithDish(@RequestBody SetmealDTO setmealDTO) {
        setmealService.saveWithDish(setmealDTO);
        return Result.success();
    }


    // 批量删除套餐
    @DeleteMapping
    @CacheEvict(cacheNames = "setmeal", allEntries = true)// 清理全部缓存
    public Result deleteBatch(@RequestParam List<Long> ids) {
        setmealService.deleteBatch(ids);
        return Result.success();
    }


    // 修改套餐
    @PutMapping
    @CacheEvict(cacheNames = "setmeal", allEntries = true)// 清理全部缓存
    public Result update(@RequestBody SetmealDTO setmealDTO) {
        setmealService.updateWithDishes(setmealDTO);
        return Result.success();
    }


    // 起售停售套餐
    @PostMapping("/status/{status}")
    @CacheEvict(cacheNames = "setmeal", allEntries = true)// 清理全部缓存
    public Result changeStatus(@PathVariable Integer status, Long id) {
        setmealService.changeStatus(status, id);
        return Result.success();
    }

}
```