《Spring Boot 基础教程》

第 20 节

使用 Caching-Redis

一、添加依赖

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

二、配置文件：

```
spring.cache.type=redis
```

三、缓存使用优先级问题

1.默认按照 spring boot 的加载顺序来实现

2.配置文件优先于默认

四、自定义缓存管理器

```
/**
 * redis 自定义缓存管理器
 *
 * @author wujing
 */
@Configuration
public class RedisCacheConfiguration extends CachingConfigurerSupport {

    /**
     * 自定义缓存管理器.
     *
     * @param redisTemplate
     * @return
     */
    @Bean
    public CacheManager cacheManager(RedisTemplate<?, ?> redisTemplate) {
        RedisCacheManager cacheManager = new RedisCacheManager(redisTemplate);
        // 设置默认的过期时间
        cacheManager.setDefaultExpiration(20);
        Map<String, Long> expires = new HashMap<String, Long>();
        // 单独设置
        expires.put("roncooCache", 200L);
        cacheManager.setExpires(expires);
        return cacheManager;
```

```
    }
```

自定义 key 的生成策略

```
/**
     * 自定义 key. 此方法将会根据类名+方法名+所有参数的值生成唯一的一个 key,即使@Cacheable 中的 value 属性一样，key 也会不一样。
     */
    @Override
    public KeyGenerator keyGenerator() {
        return new KeyGenerator() {
            @Override
            public Object generate(Object o, Method method, Object... objects) {
                StringBuilder sb = new StringBuilder();
                sb.append(o.getClass().getName());
                sb.append(method.getName());
                for (Object obj : objects) {
                    sb.append(obj.toString());
                }
                return sb.toString();
            }
        };
    }


}
```

更多课程信息，请关注 龙果学院 官方网站 http://www.roncoo.com/
或关注 龙果 微信公众号 RonCoo_com