SpringBoot整合Redis入门案例

1. 创建项目

2. 添加相关依赖

- redis
- 通用池
- mysql
- mybatis
- 通用mapper
- lombok
- web
- test

```
<!-- boot版本 -->
<parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>2.2.7.RELEASE</version>
</parent>

<dependencies>
    <!-- redis -->
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-data-redis</artifactId>
    </dependency>
    <!-- 通用池 -->
    <dependency>
        <groupId>org.apache.commons</groupId>
        <artifactId>commons-pool2</artifactId>
    </dependency>
    <!-- mysql -->
    <dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
    </dependency>
    <!-- mybatis -->
    <dependency>
        <groupId>org.mybatis.spring.boot</groupId>
        <artifactId>mybatis-spring-boot-starter</artifactId>
        <version>2.1.1</version>
    </dependency>
```

```
    <!-- 通用mapper -->
    <dependency>
        <groupId>tk.mybatis</groupId>
        <artifactId>mapper-spring-boot-starter</artifactId>
        <version>2.1.5</version>
    </dependency>
    <!-- lombok -->
    <dependency>
        <groupId>org.projectlombok</groupId>
        <artifactId>lombok</artifactId>
    </dependency>
    <!-- test -->
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
    </dependency>
</dependencies>
```

3. 配置

- 端口
- mysql数据源
- redis数据源
- log
- mybatis

```
# 端口、mysql数据源、redis、mybatis 、log
server:
  port: 9998

spring:
  datasource:
    username: root
    password: 123456
    url: jdbc:mysql://127.0.0.1:3306/dbtest

  redis:
    host: localhost
    port: 6379
    timeout: 1000
    jedis:
      pool:
        min-idle: 5
        max-idle: 10
        max-wait: -1

mybatis:
  mapper-locations: classpath:/mybatis/mapper/*.xml
  type-aliases-package: cn.kgc.entities
```

```
  configuration:
    map-underscore-to-camel-case: true
```

4. 业务实现

1. 简单实用redis

```
@RestController
public class RedisContoller {
    @Autowired
    private RedisTemplate redisTemplate;

    @GetMapping("/redis/get/{key}")
    public Object get(@PathVariable("key") String key){
       return  redisTemplate.opsForValue().get(key);
    }

    @PostMapping("/redis/set/{key}/{value}")
    public Object set(@PathVariable("key") String key,
                      @PathVariable("value") String value){
        redisTemplate.opsForValue().set(key,value);
        return "set success";
    }
}
@Configuration
public class RedisConfig {

    @Bean
    public RedisTemplate<String,Object> redisTemplate(RedisConnectionFactory factory){
        RedisTemplate<String,Object> redisTemplate = new RedisTemplate<>();
        redisTemplate.setConnectionFactory(factory);

        // 指定kv的序列化方式
        Jackson2JsonRedisSerializer jackson2JsonRedisSerializer = new Jackson2JsonRedisSerializer(Object.class);
        //redisTemplate.setDefaultSerializer(jackson2JsonRedisSerializer);
        redisTemplate.setValueSerializer(jackson2JsonRedisSerializer);
        redisTemplate.setKeySerializer(new StringRedisSerializer());

        return redisTemplate;
    }
}
```

2. 并发缓存测试

```
public  Object getEmpById(Integer id) {
```

```
        // 先从缓存获取数据，如果有则直接返回
        //                   如果无，则查询mysql,并将数据设置到缓存
        String key = "user:" + id;
        Object userObj = redisTemplate.opsForValue().get(key);
        if(userObj == null){
            synchronized (this.getClass()){
                userObj = redisTemplate.opsForValue().get(key);
                if(userObj == null ){
                    log.debug("----> 查询数据库.............");
                    // 查数据库
                    Emp emp = empMapper.selectByPrimaryKey(id);
                    redisTemplate.opsForValue().set(key,emp);
                    return emp;
                }else{
                    log.debug("----> 查询缓存(同步代码块)>>>>>>>>>>>>>>>>>>");
                    return userObj;
                }
            }

        }else{
            log.debug("----> 查询缓存>>>>>>>>>>>>>>>>>");
        }
        return userObj;
    }
```

5. 启动类

6. 测试