**具体如何在项目中操作数据，请看项目代码。最主要的看子模块redis的springbootredis相关代码，例子展示了如何从redis存取基本数据类型和对象。**

**第一步：导入依赖**

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

**第二步：配置redis**

创建一个redis配置类（本例配置类名称为RedisConfig），具体代码在最后。该类的作用：将存到redis的数据序列化。  
 如果不配置这个类，使用“redis desktop manager”查看redis缓存的数据时，数据会乱码（想看效果的话，可以试着将本类的代码都注释了，做个对比）

**第三步：引入RedisTemplate**

在serviceImpl层引入，通过RedisTemplate对redis进行操作

@Autowired  
**private** RedisTemplate **redisTemplate**;

**第四步：直接使用redisTeplate即可**

**附：RedisConfig.class代码**@Configuration  
@EnableCaching  
**public class** RedisConfig **extends** CachingConfigurerSupport {//自定义缓存key生成策略@Bean  
 **public** KeyGenerator keyGenerator() {  
 **return new** KeyGenerator(){  
 @Override  
 **public** Object generate(Object target, java.lang.reflect.Method method, Object... params) {  
 StringBuffer sb = **new** StringBuffer();  
 sb.append(target.getClass().getName());  
 sb.append(method.getName());  
 **for**(Object obj:params){  
 sb.append(obj.toString());  
 }  
 **return** sb.toString();  
 }  
 };  
 }  
 @Bean  
 **public** CacheManager cacheManager(RedisConnectionFactory connectionFactory) {  
// 基于java代码实现数据库查询同步缓存到reids，不会出现乱码，的//cachemanager的配置RedisCacheManager redisCacheManager = RedisCacheManager.*create*(connectionFactory);  
 **return** redisCacheManager;  
 }@Bean  
 **public** RedisTemplate<String, Object> redisTemplate(RedisConnectionFactory factory){

// 基于java代码的查询数据库同步更新缓存的代码，不会让存到缓存的数据出// 现乱码。可用RedisTemplate<String, Object> template = **new** RedisTemplate<>();  
// 配置连接工厂template.setConnectionFactory(factory);  
//使用Jackson2JsonRedisSerializer来序列化和反序列化redis的value值（默认使用JDK的序列化方式）Jackson2JsonRedisSerializer jacksonSeial = **new** Jackson2JsonRedisSerializer(Object.**class**);  
  
 ObjectMapper om = **new** ObjectMapper();  
// 指定要序列化的域，field,get和set,以及修饰符范围，ANY是都有包括//private和publicom.setVisibility(PropertyAccessor.***ALL***, JsonAutoDetect.Visibility.***ANY***);  
// 指定序列化输入的类型，类必须是非final修饰的，final修饰的类，比如//String,Integer等会跑出异常om.enableDefaultTyping(ObjectMapper.DefaultTyping.***NON\_FINAL***);  
 jacksonSeial.setObjectMapper(om);  
 // 值采用json序列化template.setValueSerializer(jacksonSeial);  
 //使用StringRedisSerializer来序列化和反序列化redis的key值template.setKeySerializer(**new** StringRedisSerializer()); *//原*  // 设置hash key 和value序列化模式template.setHashKeySerializer(**new** StringRedisSerializer());  
 template.setHashValueSerializer(jacksonSeial);  
 template.afterPropertiesSet();  
 **return** template;  
 }}