# 第2章：SpringBoot集成Redis

## 课程内容

|  |
| --- |
| 1. springboot集成redis缓存  1. 环境准备  2. 使用springboot操作redis  3. 案例 |

## 环境准备

### 添加jar依赖

|  |
| --- |
| <!-- springboot集成redis -->  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-data-redis</artifactId>  </dependency> |

### 配置redis的连接信息

|  |
| --- |
| #redis  spring.redis.host=127.0.0.1  spring.redis.port=6379 |

## springboot操作redis

### 注入RedisTemplate

|  |
| --- |
|  |

### 操作字符串类型

|  |
| --- |
|  |

### 类型结果分析

#### 存在问题

|  |
| --- |
|  |

#### 原因

|  |
| --- |
| 1. spring-data-redis的RedisTemplate<K, V>模板类在操作redis时默认使用JdkSerializationRedisSerializer来进行序列化 2. 因为spring操作redis是在jedis客户端基础上进行的，而jedis客户端与redis交互的时候协议中定义是用byte类型交互，jedis中提供了string类型转为byte[]类型 |

#### 解决办法

##### 添加依赖

|  |
| --- |
| <dependency>  <groupId>com.fasterxml.jackson.core</groupId>  <artifactId>jackson-annotations</artifactId>  </dependency> |

##### 编写配置类

|  |
| --- |
| @Configuration  **public** **class** RedisConfig {  @Bean  **public** RedisTemplate<Object,Object> redisTemplate(RedisConnectionFactory redisConnectionFactory){  RedisTemplate<Object,Object> redisTemplate = **new** RedisTemplate<>();  // 设置redis连接  redisTemplate.setConnectionFactory(redisConnectionFactory);  // 使用Jackson2JsonRedisSerialize 替换默认序列化  Jackson2JsonRedisSerializer jackson2JsonRedisSerializer = **new** Jackson2JsonRedisSerializer(Object.**class**);  ObjectMapper objectMapper = **new** ObjectMapper();  objectMapper.setVisibility(PropertyAccessor.***ALL***, JsonAutoDetect.Visibility.***ANY***);  objectMapper.enableDefaultTyping(ObjectMapper.DefaultTyping.***NON\_FINAL***);  jackson2JsonRedisSerializer.setObjectMapper(objectMapper);  // 设置value的序列化规则和 key的序列化规则  redisTemplate.setHashValueSerializer(jackson2JsonRedisSerializer);  redisTemplate.setValueSerializer(jackson2JsonRedisSerializer);  // 将redisTemplate的序列化方式更改为StringRedisSerializer  redisTemplate.setHashKeySerializer(**new** StringRedisSerializer());  redisTemplate.setKeySerializer(**new** StringRedisSerializer());  redisTemplate.afterPropertiesSet();  **return** redisTemplate;  }  } |

### 操作哈希类型hash

|  |
| --- |
|  |

### 操作集合类型Set

|  |
| --- |
|  |

### 操作列表类型list

|  |
| --- |
|  |

### 操作有序集合类型sortedSet

|  |
| --- |
|  |

## 使用springboot改造分类查询案例

### 需求

|  |
| --- |
| 1. 提供index.html页面，页面中有一个一级分类下拉列表  2. 当页面加载完成后 发送ajax请求，加载所有一级分类 |

### 添加jar依赖

|  |
| --- |
| <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-web</artifactId>  </dependency>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-tomcat</artifactId>  <scope>provided</scope>  </dependency>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-test</artifactId>  <scope>test</scope>  </dependency>  <!-- springboot集成redis -->  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-data-redis</artifactId>  </dependency>    <!-- MySQL驱动包 -->  <dependency>  <groupId>mysql</groupId>  <artifactId>mysql-connector-java</artifactId>  <scope>runtime</scope>  </dependency>  <!-- MyBatisPlus驱动包 -->  <dependency>  <groupId>com.baomidou</groupId>  <artifactId>mybatis-plus-boot-starter</artifactId>  <version>3.1.1</version>  </dependency>  <!-- druid 数据库连接池 -->  <dependency>  <groupId>com.alibaba</groupId>  <artifactId>druid</artifactId>  <version>1.0.9</version>  </dependency>  <!-- 以下是代码生成器的jar依赖 -->  <!-- 代码生成器核心jar依赖 -->  <dependency>  <groupId>com.baomidou</groupId>  <artifactId>mybatis-plus-generator</artifactId>  <version>3.1.1</version>  </dependency>    <!-- 使用默认的velocity模板 -->  <dependency>  <groupId>org.apache.velocity</groupId>  <artifactId>velocity-engine-core</artifactId>  <version>2.0</version>  </dependency>  <!-- sfl4j -->  <dependency>  <groupId>org.slf4j</groupId>  <artifactId>slf4j-api</artifactId>  </dependency>  <dependency>  <groupId>org.slf4j</groupId>  <artifactId>slf4j-log4j12</artifactId>  </dependency>    <!-- fastjson -->  <dependency>  <groupId>com.alibaba</groupId>  <artifactId>fastjson</artifactId>  <version>1.2.47</version>  </dependency>    <dependency>  <groupId>com.fasterxml.jackson.core</groupId>  <artifactId>jackson-annotations</artifactId>  </dependency> |

#### application.properties

|  |
| --- |
| #加载驱动  spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver  #数据库连接路径  spring.datasource.url=jdbc:mysql://localhost:3306/travel?useUnicode=true&characterEncoding=UTF-8&serverTimezone=UTC  #数据库用户名  spring.datasource.username=root  #数据库密码  spring.datasource.password=root  #数据源类型（阿里巴巴）  spring.datasource.type=com.alibaba.druid.pool.DruidDataSource  #mybatis取别名  mybatis-plus.type-aliases-package=com.redis.entiy  #日志  logging.level.com.redis.dao=debug |

#### 业务接口

|  |
| --- |
|  |

#### 业务实现类

|  |
| --- |
|  |

#### 控制器

|  |
| --- |
|  |

#### 页面

|  |
| --- |
|  |

### 总结

|  |
| --- |
| 注意：使用redis缓存，对一些不经常发生变化的数据进行缓存，数据库的数据一旦发生改变，则需要更新缓存。  \* 数据库的表执行 增删改的相关操作，需要将redis缓存数据情况，再次存入  \* 在service对应的增删改方法中，将redis数据删除。 |