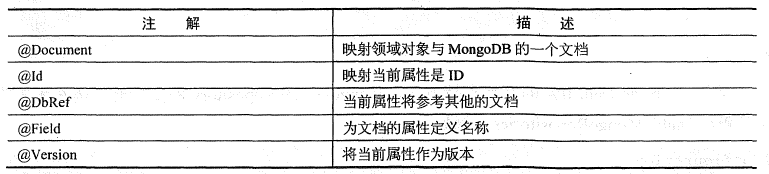
# Spring Boot核心-非关系型数据库NoSQL

## MongoDB

MongoDB 是一个基于文档（ Document ）的存储型的数据瘁，使用面向对象的思想，每一条数据记录都是文档的对象。Spring 对MongoDB 的支持主要是通过Spring Data MongoDB 来实现的， Spring Data MongoDB 为我们提供了如下功能。

**Object/Document 映射注解支持**

JPA 提供了一套Object/Relation 映射的注解（＠Entity 、＠Id ），而Spring Data MongoDB 也提供了下图所示的注解。



**（2）MongoTemplate**

像JdbcTemplate 一样， Spring Data MongoDB 也为我们提供了一个MongoTemplate，MongoTemplate 为我们提供了数据访问的方法。我们还需要为MongoClient 以及MongoDbFactory 来配置数据库连接属性。

|  |
| --- |
| **public class** MongoConfigSample {  @Bean  **public** MongoClient client () **throws** Exception {  **return new** MongoClient(**new** ServerAddress(**"127.0.0.1"**, **27017**));  }   @Bean  **public** MongoDbFactory mongoDbFactory() **throws** Exception {  String database = **new** MongoClientURI(**"mongo://localhost/test"**).toString();  **return new** SimpleMongoDbFactory(client(), database);  }    @Bean  **public** MongoTemplate mongoTemplate() **throws** Exception {  **return new** MongoTemplate(mongoDbFactory());  } } |

( 3 ) Repository 的支持

类似于Spring Data JPA, Spring Data MongoDB 也提供了Repository 的支付，使用方式和Spring Data JPA一致。

|  |
| --- |
| **public interface** PersonRepository **extends** MongoRepository<Person, String>{  Person findByName(String name);   @Query(**"{'age': ?0}"**)  List<Person> withQueryFindByAge(Integer age); } |

类似于Spring Data JPA 的开启文持方式， MongoDB的Repository 的支持开启需要在配置类上注解@EnableMongoRepositories 。

## Spring Boot 的支持

Spring Boot 对MongoDB 的支持，分别位于org.springframework.boot.autocorifigure.mongo。主要配置数据库连接、Mongo Template 。我们可以使用以“ spring.data.mongodb ”为前缀的属性来配置MongoDB 相关的信息。Spring Boot 为我们提供了一些默认属性，如默认MongoDB 的端口为27017、默认服务器为local host、默认数据库为testa Spring Boot 的主要配置如下：

|  |
| --- |
| *# MONGODB (*[MongoProperties](https://github.com/spring-projects/spring-boot/tree/v1.5.2.RELEASE/spring-boot-autoconfigure/src/main/java/org/springframework/boot/autoconfigure/mongo/MongoProperties.java" \t "_top))  spring.data.mongodb.authentication-database= *# Authentication database name.*  spring.data.mongodb.database=test *# Database name.*  spring.data.mongodb.field-naming-strategy= *# Fully qualified name of the FieldNamingStrategy to use.*  spring.data.mongodb.grid-fs-database= *# GridFS database name.*  spring.data.mongodb.host=localhost *# Mongo server host. Cannot be set with uri.*  spring.data.mongodb.password= *# Login password of the mongo server. Cannot be set with uri.*  spring.data.mongodb.port=27017 *# Mongo server port. Cannot be set with uri.*  spring.data.mongodb.repositories.enabled=true *# Enable Mongo repositories.*  spring.data.mongodb.uri=mongodb://localhost/test *# Mongo database URI. Cannot be set with host, port and credentials.*  spring.data.mongodb.username= *# Login user of the mongo server. Cannot be set with uri.* |

## Redis

Red is 是一个基于键值对的开源内存数据存储，当然Redis 也可以做数据缓存。

## Spring 的支持

（1） 配置

Spring 对Redis 的支持也是通过Spring Data Redis 来实现的， Spring Data JPA 为我们提供了连接相关的ConnectionFactory 和数据操作相关的RedisTemplate 。在此特别指出， Spring Data Redis 只对Redis 2.6 和2.8 版本做过测试。根据Redis 的不同的Java 客户端， Spring Data Redis 提供了如下的ConnectionFactory:

JedisConnectionF actory ：使用Jedis 作为Redis 客户端。

JredisConnectionF actory：使用Jredis 作为Redis 客户端。

LcttuceConnectionFactory ：使用Lettuce 作为Redis 客户端。

SrpConnectionFactory：使用Spullara/redis-protocal 作为Redis 客户端。

配置方式如下：

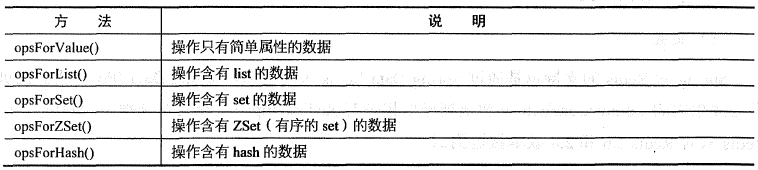
|  |
| --- |
| @Bean **public** RedisConnectionFactory redisConnectionFactory() {  **return new** JedisConnectionFactory(); } |

RedisTemplate 配置方式如下

|  |
| --- |
| @Bean **public** RedisTemplate<Object, Object> redisTemplate() {  RedisTemplate<Object, Object> template = **new** RedisTemplate<>();  template.setConnectionFactory(redisConnectionFactory());  **return** template; } |

（2）使用

Spring Data Redis 为我们提供了RedisTemplate 和StringRedisTemplate 两个模板来进行数据操作，其中， StringRedisTemplate 只针对键值都是字符型的数据进行操作。RedisTemplate 和StringRedisTemplate 提供的主要数据访问方法如下。



（2）定义Serializer

当我们的数据存储到Redis 的时候，我们的键（ key ）和值（ value ）都是通过Spring 提供的Serializer 序列化到数据库的。RedisTemplate 默认使用的是JdkSerializationRedi'sSerializer,StringRedisTemplate 默认使用的是StringRedisSerializero。

Spring Data JPA 为我们提供了下面的Serializer:

GenericToStringSerializer 、Jackson2JsonRedisSerializer 、JacksonJsonRedisSerializer 、J dkSerializationRedisSerializer、OxmSerializer 、StringRedisSerializer。

## Spring Boot 的支持

Spring Boot 对Redis的支持， org.springframework.boot.autoconfigure.redis。RedisAutoConfiguration 为我们默认配置了JedisConnectionFactory 、RedisTemplate 以及StringRedisTemplate ，让我们可以直接使用Redis 作为数据存储。RedisProperties 向我们展示了可以使用以“ spring.redis ”为前缀的属性在application.prope此时中配置Redis ，主要属性如下：

|  |
| --- |
| *# REDIS (*[RedisProperties](https://github.com/spring-projects/spring-boot/tree/v1.5.2.RELEASE/spring-boot-autoconfigure/src/main/java/org/springframework/boot/autoconfigure/data/redis/RedisProperties.java" \t "_top))  spring.redis.cluster.max-redirects= *# Maximum number of redirects to follow when executing commands across the cluster.*  spring.redis.cluster.nodes= *# Comma-separated list of "host:port" pairs to bootstrap from.*  spring.redis.database=0 *# Database index used by the connection factory.*  spring.redis.url= *# Connection URL, will override host, port and password (user will be ignored), e.g. redis://user:password@example.com:6379*  spring.redis.host=localhost *# Redis server host.*  spring.redis.password= *# Login password of the redis server.*  spring.redis.ssl=false *# Enable SSL support.*  spring.redis.pool.max-active=8 *# Max number of connections that can be allocated by the pool at a given time. Use a negative value for no limit.*  spring.redis.pool.max-idle=8 *# Max number of "idle" connections in the pool. Use a negative value to indicate an unlimited number of idle connections.*  spring.redis.pool.max-wait=-1 *# Maximum amount of time (in milliseconds) a connection allocation should block before throwing an exception when the pool is exhausted. Use a negative value to block indefinitely.*  spring.redis.pool.min-idle=0 *# Target for the minimum number of idle connections to maintain in the pool. This setting only has an effect if it is positive.*  spring.redis.port=6379 *# Redis server port.*  spring.redis.sentinel.master= *# Name of Redis server.*  spring.redis.sentinel.nodes= *# Comma-separated list of host:port pairs.*  spring.redis.timeout=0 *# Connection timeout in milliseconds.* |