## Spring Boot 中 Redis 的使用

**Redis 介绍**

Redis 是目前业界使用最广泛的内存数据存储。相比 Memcached，Redis 支持更丰富的数据结构，例如 hashes, lists, sets 等，同时支持数据持久化。除此之外，Redis 还提供一些类数据库的特性，比如事务，HA，主从库。可以说 Redis 兼具了缓存系统和数据库的一些特性，因此有着丰富的应用场景。本文介绍 Redis 在 Spring Boot 中两个典型的应用场景。

**如何使用**

1、引入依赖包

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-redis</artifactId>

</dependency>

<dependency>

<groupId>org.apache.commons</groupId>

<artifactId>commons-pool2</artifactId>

</dependency>

Spring Boot 提供了对 Redis 集成的组件包：spring-boot-starter-data-redis，spring-boot-starter-data-redis依赖于spring-data-redis 和 lettuce 。Spring Boot 1.0 默认使用的是 Jedis 客户端，2.0 替换成 Lettuce，但如果你从 Spring Boot 1.5.X 切换过来，几乎感受不大差异，这是因为 spring-boot-starter-data-redis 为我们隔离了其中的差异性。

Lettuce 是一个可伸缩线程安全的 Redis 客户端，多个线程可以共享同一个 RedisConnection，它利用优秀 netty NIO 框架来高效地管理多个连接。

2、添加配置文件

# Redis数据库索引（默认为0）

spring.redis.database=0

# Redis服务器地址

spring.redis.host=localhost

# Redis服务器连接端口

spring.redis.port=6379

# Redis服务器连接密码（默认为空）

spring.redis.password=

# 连接池最大连接数（使用负值表示没有限制） 默认 8

spring.redis.lettuce.pool.max-active=8

# 连接池最大阻塞等待时间（使用负值表示没有限制） 默认 -1

spring.redis.lettuce.pool.max-wait=-1

# 连接池中的最大空闲连接 默认 8

spring.redis.lettuce.pool.max-idle=8

# 连接池中的最小空闲连接 默认 0

spring.redis.lettuce.pool.min-idle=0

3、添加 cache 的配置类

@Configuration

@EnableCaching

public class RedisConfig extends CachingConfigurerSupport{

@Bean

public KeyGenerator keyGenerator() {

return new KeyGenerator() {

@Override

public Object generate(Object target, Method method, Object... params) {

StringBuilder sb = new StringBuilder();

sb.append(target.getClass().getName());

sb.append(method.getName());

for (Object obj : params) {

sb.append(obj.toString());

}

return sb.toString();

}

};

}

}

注意我们使用了注解：@EnableCaching来开启缓存。

3、好了，接下来就可以直接使用了

@RunWith(SpringRunner.class)

@SpringBootTest

public class TestRedis {

@Autowired

private StringRedisTemplate stringRedisTemplate;

@Autowired

private RedisTemplate redisTemplate;

@Test

public void test() throws Exception {

stringRedisTemplate.opsForValue().set("aaa", "111");

Assert.assertEquals("111", stringRedisTemplate.opsForValue().get("aaa"));

}

@Test

public void testObj() throws Exception {

User user=new User("aa@126.com", "aa", "aa123456", "aa","123");

ValueOperations<String, User> operations=redisTemplate.opsForValue();

operations.set("com.neox", user);

operations.set("com.neo.f", user,1, TimeUnit.SECONDS);

Thread.sleep(1000);

//redisTemplate.delete("com.neo.f");

boolean exists=redisTemplate.hasKey("com.neo.f");

if(exists){

System.out.println("exists is true");

}else{

System.out.println("exists is false");

}

// Assert.assertEquals("aa", operations.get("com.neo.f").getUserName());

}

}

以上都是手动使用的方式，如何在查找数据库的时候自动使用缓存呢，看下面；

4、自动根据方法生成缓存

@RestController

public class UserController {

@RequestMapping("/getUser")

@Cacheable(value="user-key")

public User getUser() {

User user=new User("aa@126.com", "aa", "aa123456", "aa","123");

System.out.println("若下面没出现“无缓存的时候调用”字样且能打印出数据表示测试成功");

return user;

}

}

其中 value 的值就是缓存到 Redis 中的 key

**共享 Session**

分布式系统中，Session 共享有很多的解决方案，其中托管到缓存中应该是最常用的方案之一，

Spring Session 官方说明

Spring Session provides an API and implementations for managing a user’s session information.

Spring Session 提供了一套创建和管理 Servlet HttpSession 的方案。Spring Session 提供了集群 Session（Clustered Sessions）功能，默认采用外置的 Redis 来存储 Session 数据，以此来解决 Session 共享的问题。

如何使用

1、引入依赖

<dependency>

<groupId>org.springframework.session</groupId>

<artifactId>spring-session-data-redis</artifactId>

</dependency>

2、Session 配置：

@Configuration

@EnableRedisHttpSession(maxInactiveIntervalInSeconds = 86400\*30)

public class SessionConfig {

}

maxInactiveIntervalInSeconds: 设置 Session 失效时间，使用 Redis Session 之后，原 Spring Boot 的 server.session.timeout 属性不再生效。

好了，这样就配置好了，我们来测试一下

3、测试

添加测试方法获取 sessionid

@RequestMapping("/uid")

String uid(HttpSession session) {

UUID uid = (UUID) session.getAttribute("uid");

if (uid == null) {

uid = UUID.randomUUID();

}

session.setAttribute("uid", uid);

return session.getId();

}

登录 Redis 输入 keys '\*sessions\*'

t<spring:session:sessions:db031986-8ecc-48d6-b471-b137a3ed6bc4

t(spring:session:expirations:1472976480000

其中 1472976480000 为失效时间，意思是这个时间后 Session 失效，db031986-8ecc-48d6-b471-b137a3ed6bc4 为 sessionId,登录 http://localhost:8080/uid 发现会一致，就说明 Session 已经在 Redis 里面进行有效的管理了。

如何在两台或者多台中共享 Session

其实就是按照上面的步骤在另一个项目中再次配置一次，启动后自动就进行了 Session 共享。