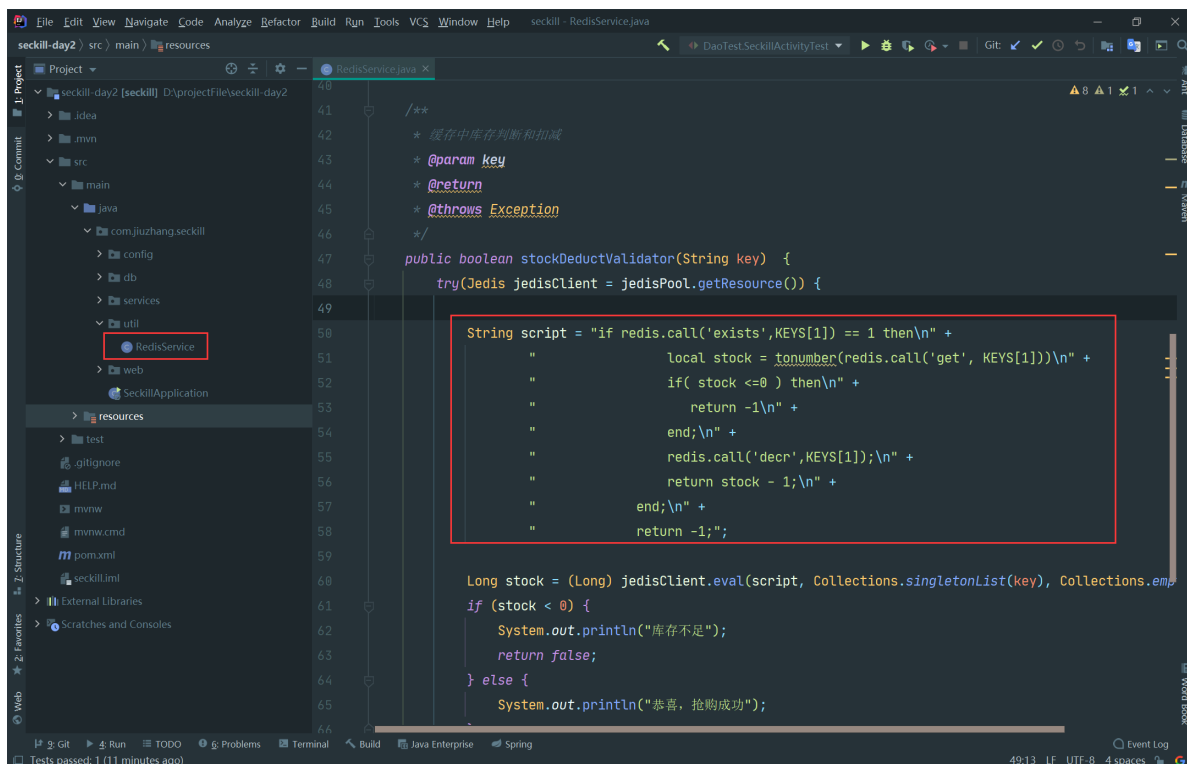


《保姆级教程》第六集 Lua 脚本实现库存扣减

全网一手不加密IT课程，需要请加微信zzj97666九章来offer都有

1. 结合 Redis 和 Lua 脚本，Redis 库存判断方法实现

在 RedisService 中加入 stockDeductValidator 方法



```
/**
 * 缓存中库存判断和扣减
 * @param key
 * @return
 * @throws Exception
 */
public boolean stockDeductValidator(String key) {
    try(Jedis jedisClient = jedisPool.getResource()) {

        String script = "if redis.call('exists',KEYS[1]) == 1 then\n" +
            "                local stock = tonumber(redis.call('get',
KEYS[1]))\n" +
            "                if( stock <=0 ) then\n" +
            "                    return -1\n" +
            "                end;\n" +
            "                redis.call('decr',KEYS[1]);\n" +
            "                return stock - 1;\n" +
            "            end;\n" +
            "            return -1;";

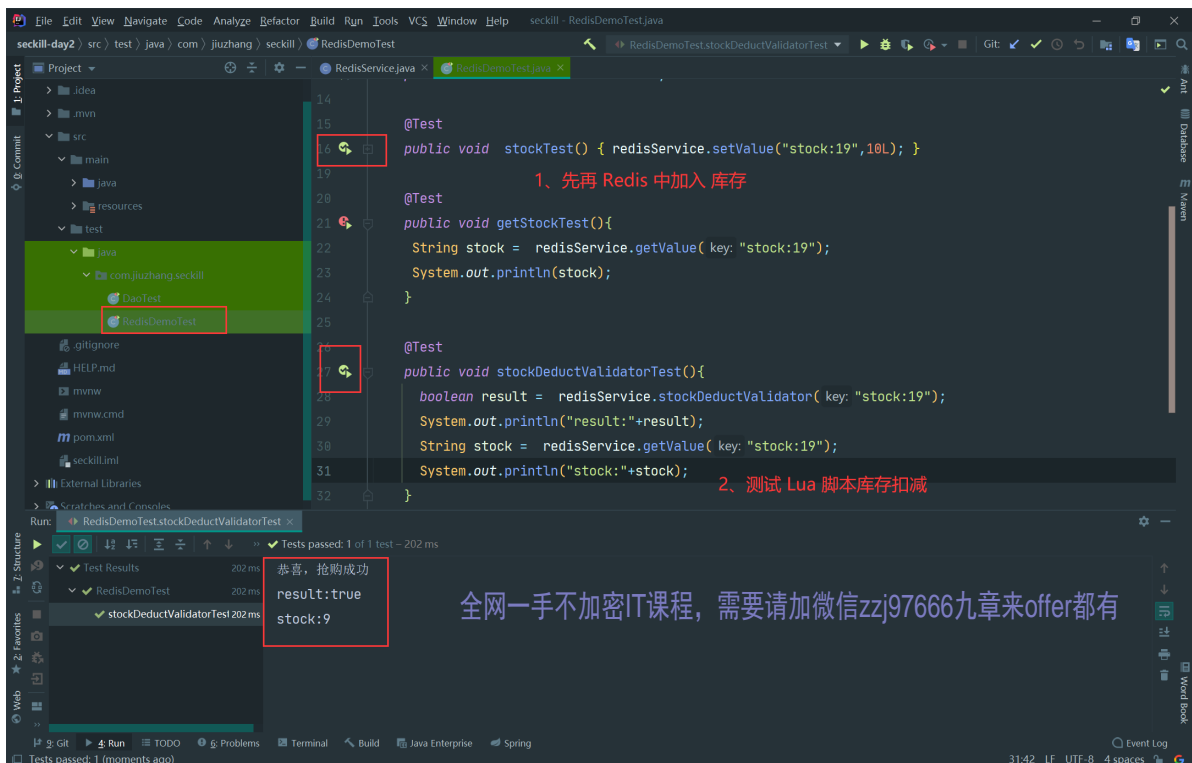
        Long stock = (Long) jedisClient.eval(script,
collections.singletonList(key), Collections.emptyList());
```

```

        if (stock < 0) {
            System.out.println("库存不足");
            return false;
        } else {
            System.out.println("恭喜, 抢购成功");
        }
        return true;
    } catch (Throwable throwable) {
        System.out.println("库存扣减失败: " + throwable.toString());
        return false;
    }
}

```

2. 测试 Lua 脚本扣减库存，在RedisDemoTest中编写测试方法，并进行测试

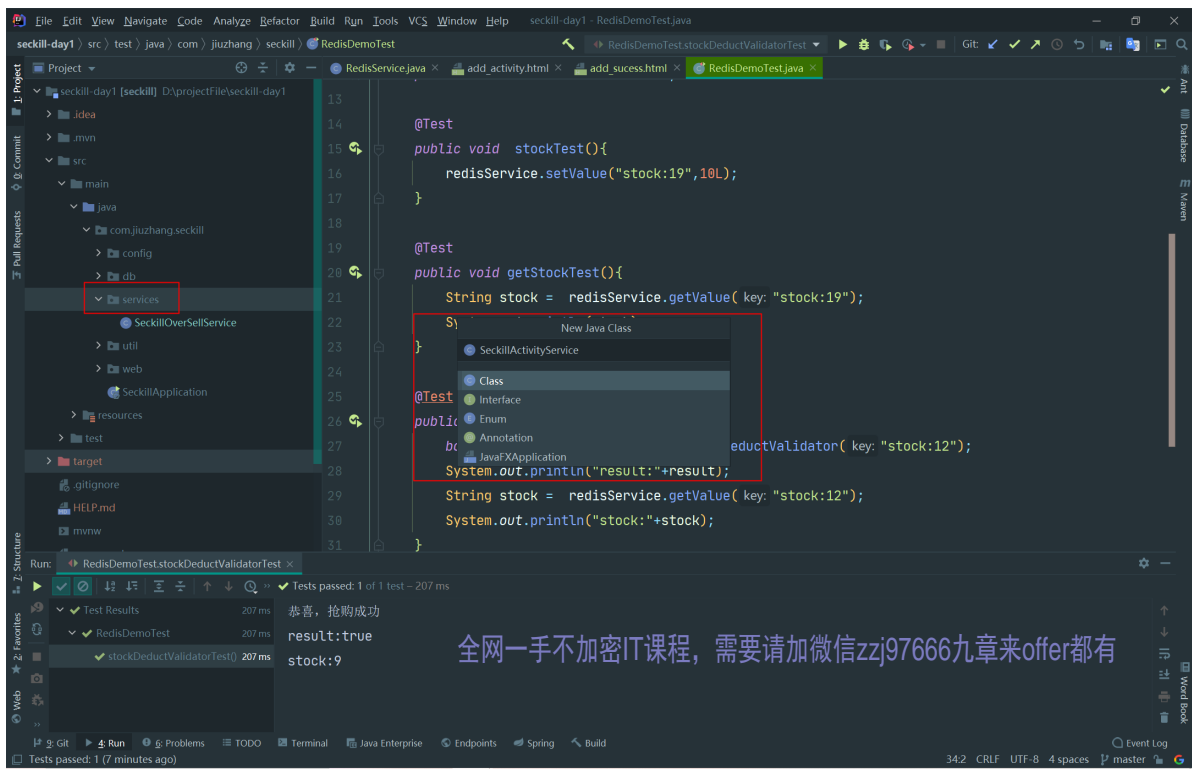


```

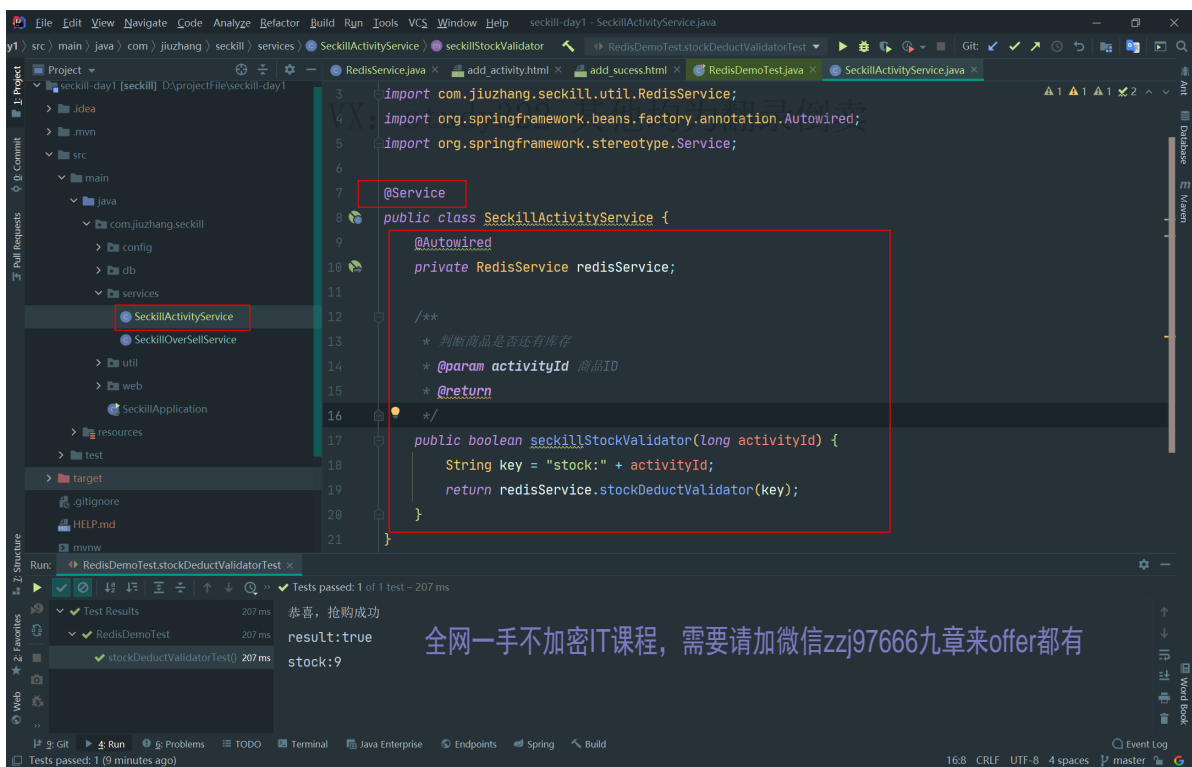
@Test
public void stockDeductValidatorTest(){
    boolean result = redisService.stockDeductValidator("stock:19");
    System.out.println("result:"+result);
    String stock = redisService.getValue("stock:19");
    System.out.println("stock:"+stock);
}

```

3. 新建 SeckillActivityService 类



4. 加上 @Service 注解，注入 RedisService 对象，编写判断库存方法



```
@Service
public class SeckillActivityService {

    @Autowired
    private RedisService redisService;

    /**
     * 判断商品是否还有库存
     * @param activityId 商品ID
```

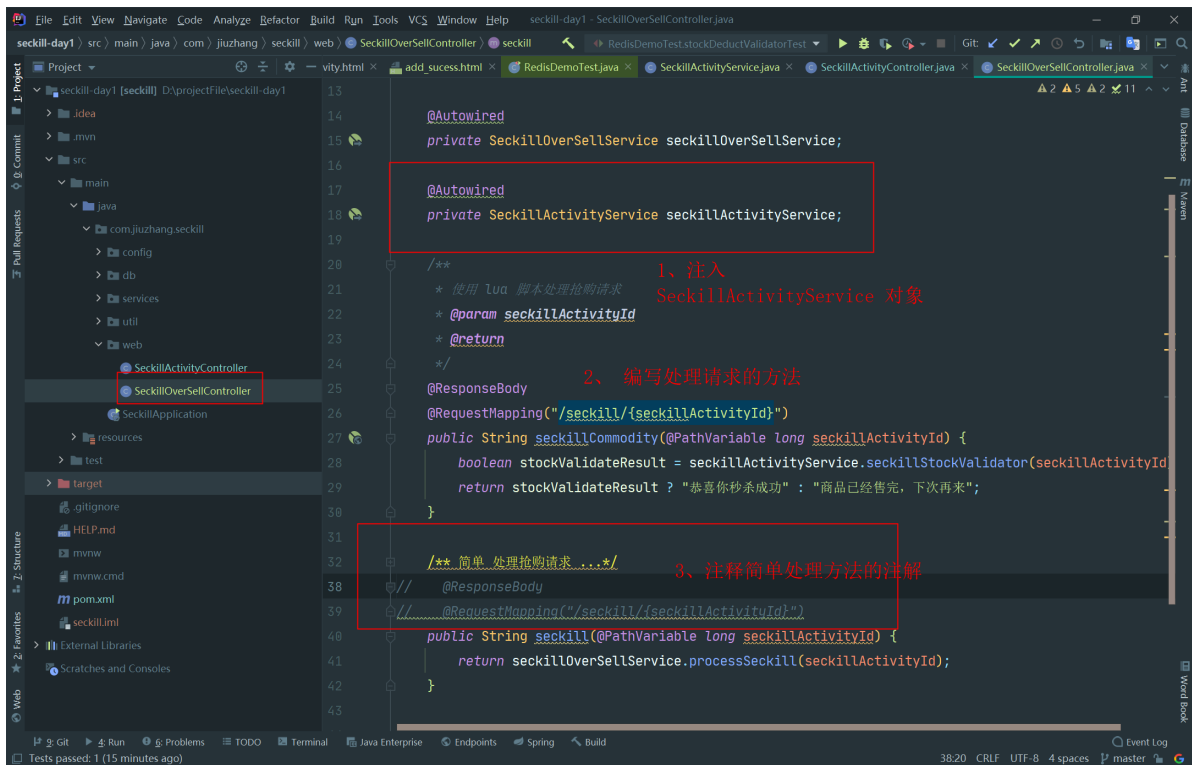
```

        * @return
        */
        public boolean seckillStockValidator(long activityId) {
            String key = "stock:" + activityId;
            return redisService.stockDeductValidator(key);
        }
    }
}

```

5. SeckillOverSellController 控制器编写 使用 lua 脚本处理抢购请求的方法

全网一手不加密IT课程，需要请加微信zzj97666九章来offer都有



```

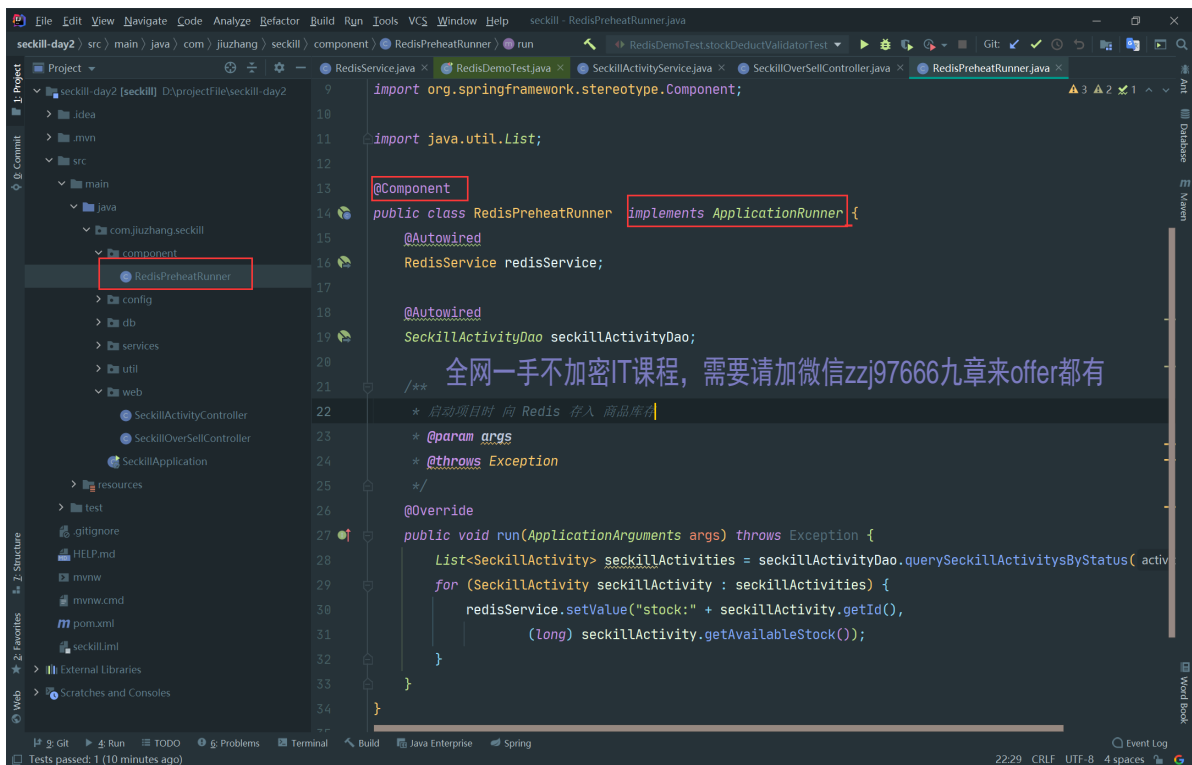
@Autowired
private SeckillActivityService seckillActivityService;

/**
 * 使用 lua 脚本处理抢购请求
 * @param seckillActivityId
 * @return
 */
@ResponseBody
@RequestMapping("/seckill/{seckillActivityId}")
public String seckillCommodity(@PathVariable long seckillActivityId) {
    boolean stockValidateResult =
    seckillActivityService.seckillStockValidator(seckillActivityId);
    return stockValidateResult ? "恭喜你秒杀成功" : "商品已经售完，下次再来";
}

```

6. 启动项目时 向 Redis 存入 商品库存

创建 RedisPreheatRunner 类



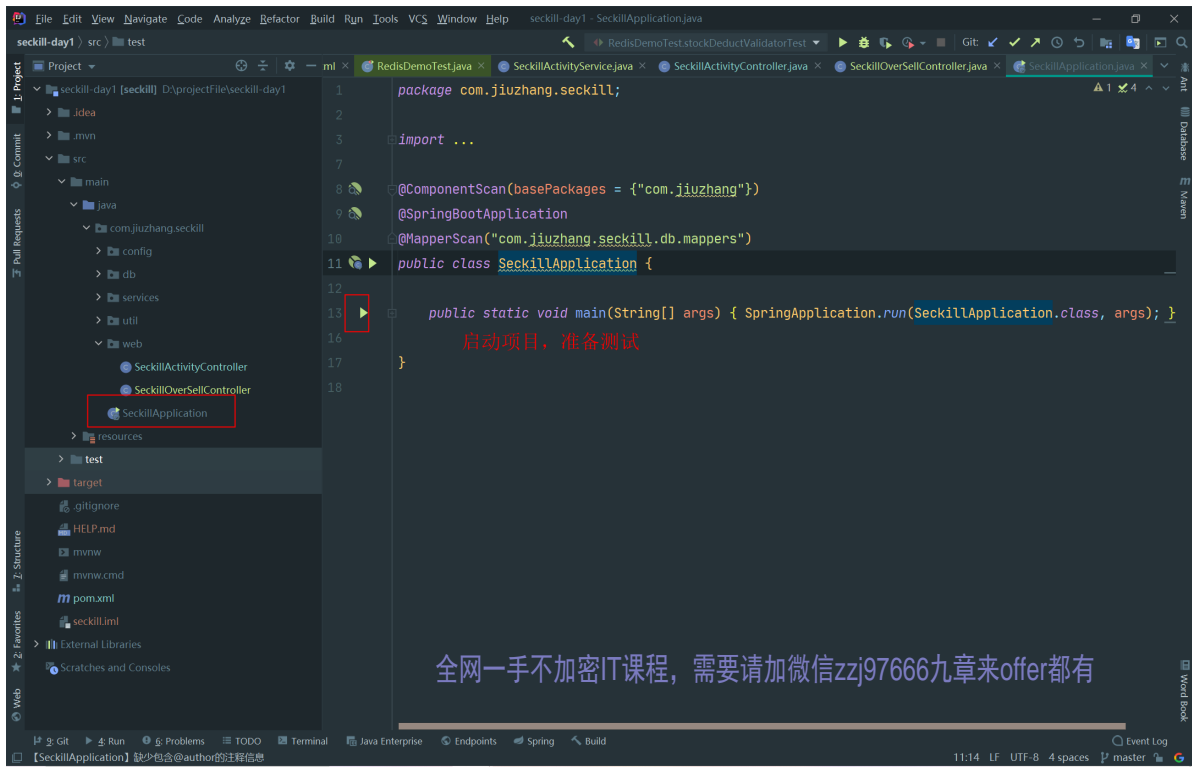
```
@Component
public class RedisPreheatRunner implements ApplicationRunner {
    @Autowired
    RedisService redisService;

    @Autowired
    SeckillActivityDao seckillActivityDao;

    /**
     * 启动项目时 向 Redis 存入 商品库存
     * @param args
     * @throws Exception
     */
    @Override
    public void run(ApplicationArguments args) throws Exception {
        List<SeckillActivity> seckillActivities =
        seckillActivityDao.querySeckillActivitiysByStatus(1);
        for (SeckillActivity seckillActivity : seckillActivities) {
            redisService.setValue("stock:" + seckillActivity.getId(),
                (long) seckillActivity.getAvailablestock());
        }
    }
}
```

全网一手不加密IT课程，需要请加微信zzj97666九章来offer都有

7. 启动项目，准备测试



8. 启动 Jmeter 开始测试

