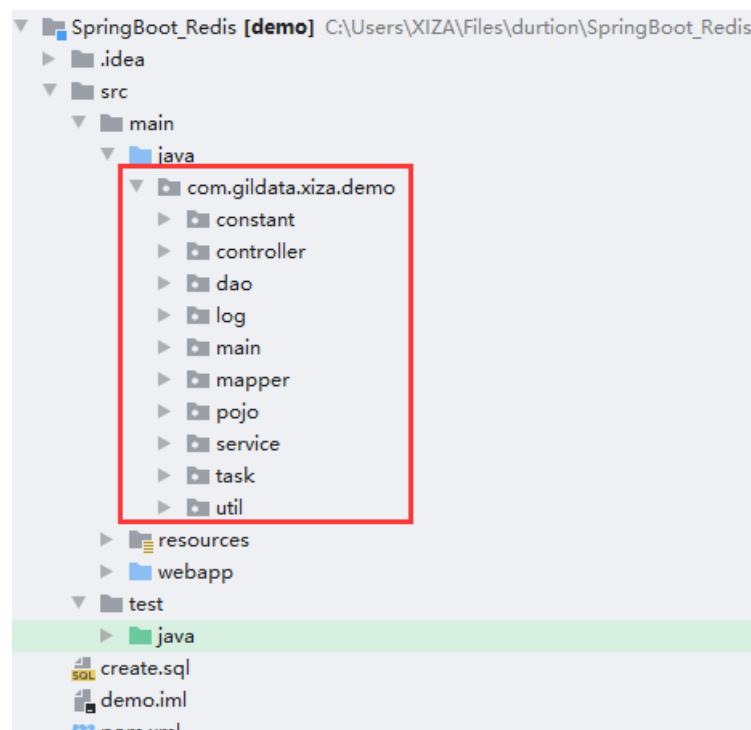


需求说明

- 需求标题：高并发情况下简单秒杀系统设计
- 需求内容：
 - 在高并发场景下，单一热点消息需求极具上升，此时若要保证服务器数据正常以及系统负载的稳定性，需采取Redis缓存将单一热点信息缓存出来。
 - 借助Lua脚本的原子性执行，保证Redis操作不会被其他线程抢占。
 - 当Redis完成商品数量削减后，及时同步更新Mysql的库存信息以及订单记录信息。
 - 模拟不采用Redis的情况下，仅靠CAS操作是否能保证服务的高响应。

代码位置

- 代码在: \src\main\java\com\gilda\demo\demo下



程序主入口

- 入口位置为: \src\main\java\com\gilda\demo\main\DemoApplication.java

```
8  /**
9   * @Description SpringBoot启动类
10  * @Author xiza@gilda.com
11  * @Date 2021/3/8
12  */
13  @SpringBootApplication(scanBasePackages = "com.gilda.demo")
14  @MapperScan(annotationClass = Mapper.class, basePac
15  @EnableScheduling
16
17  public class DemoApplication {
18
19      public static void main(String[] args) { Spring
```

The image shows the code for the `DemoApplication` class. A red arrow points from the package declaration in the left sidebar to the class definition in the code editor. The code includes annotations for `@SpringBootApplication`, `@MapperScan`, and `@EnableScheduling`.