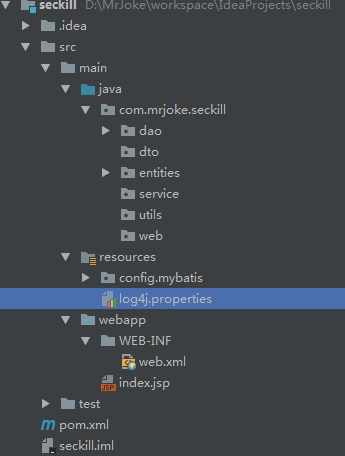
# 秒杀系统开发流程概要设计

## 使用maven命令新建web项目，用maven管理项目

mvn archetype:generate –DarchetypeArtifactId=maven-archetype-webapp –DgroupId=com.mrjoke.seckill –DartifactId=seckill

## 搭建项目的概要目录结构



## 引入项目需要的maven依赖

<!-- 统一管理maven依赖 -->  
<!-- junit4依赖，支持注解方式 -->  
<junit.version>4.12</junit.version>  
<!-- 日志依赖，使用log4j -->  
<commons-logging.version>1.1.3</commons-logging.version>  
<log4j.version>1.2.14</log4j.version>  
<!-- dao层依赖 -->  
<mysql-connector-java.version>5.1.38</mysql-connector-java.version>  
<c3p0.version>0.9.1.2</c3p0.version>  
<mybatis.version>3.4.4</mybatis.version>  
<mybatis-spring.version>1.3.1</mybatis-spring.version>  
<!-- web层依赖 -->  
<servlet-api.version>3.1.0</servlet-api.version>  
<jstl.version>1.2</jstl.version>  
<!-- spring依赖 -->  
<spring.version>4.3.8.RELEASE</spring.version>  
<!-- json依赖 -->  
<fastjson.version>1.2.0</fastjson.version>

## 编写dao层的接口，并使用mybatis实现

### 1）秒杀商品相关操作接口

public interface SeckillDao {  
 */\*\*  
 \* 根据id查询秒杀商品记录  
 \** ***@param*** *seckillId  
 \** ***@return*** *返回指定id的商品  
 \*/* Seckill selectById(@Param("seckillId") int seckillId);  
  
 */\*\*  
 \* 查询offset偏移量开始的limit条记录  
 \** ***@param*** *offset  
 \** ***@param*** *limit  
 \** ***@return*** *返回秒杀商品列表  
 \*/* List<Seckill> selectAll(@Param("offset") int offset,@Param("limit") int limit);  
  
 */\*\*  
 \* 减少指定商品的库存  
 \** ***@param*** *seckillId  
 \** ***@param*** *killTime  
 \** ***@return*** *返回记录修改条数，为0则表示修改失败  
 \*/* int reduceQuantity(@Param("seckillId") int seckillId,@Param("killTime") Date killTime);  
}

### 2）秒杀商品记录

public interface RecordDao {  
 */\*\*  
 \* 插入一条用户秒杀记录，可过滤重复  
 \** ***@param*** *seckillId  
 \** ***@param*** *recordPhone  
 \** ***@return*** *返回记录修改的条数，为0插入失败  
 \*/* int insertRecord(@Param("seckillId") int seckillId,@Param("recordPhone") String recordPhone);  
  
 */\*\*  
 \* 根据联合主键查询Record并携带Seckill秒杀商品对象  
 \** ***@param*** *seckillId  
 \** ***@return*** *\*/* Record selectByIdWithSeckill(@Param("seckillId") int seckillId,@Param("recordPhone") String recordPhone);  
}

## 对dao接口进行单元测试

### 秒杀商品Dao单元测试

public class SeckillDaoTest {  
 //注入Dao实现依赖  
 @Resource  
 private SeckillDao seckillDao;  
 @Test  
 public void selectById() throws Exception {  
 int seckillId = 1;  
 Seckill seckill = seckillDao.selectById(seckillId);  
 System.*out*.println(seckill);  
 }  
  
 @Test  
 public void selectAll() throws Exception {  
 List<Seckill> seckills = seckillDao.selectAll(0, 4);  
 for (Seckill seckill:seckills){  
 System.*out*.println(seckill);  
 }  
 }  
  
 @Test  
 public void reduceQuantity() throws Exception {  
 int i = seckillDao.reduceQuantity(1, new Date());  
 System.*out*.println(i);  
 }  
  
}

### 秒杀记录Dao单元测试

public class RecordDaoTest {  
 @Autowired  
 private RecordDao recordDao;  
 @Test  
 public void insertRecord() throws Exception {  
 int seckillId = 1;  
 String recordPhone = "13544494187";  
 int i = recordDao.insertRecord(seckillId, recordPhone);  
 System.*out*.println(i);  
 }  
  
 @Test  
 public void selectByIdWithSeckill() throws Exception {  
 int seckillId = 1;  
 String recordPhone = "13544494187";  
 Record record = recordDao.selectByIdWithSeckill(seckillId, recordPhone);  
 System.*out*.println(record);  
 }  
  
}

## 编写service层的接口，并实现

## 对service层和dao层接口进行集成测试

## 编写web的RESTful接口，并实现

## 编写前端交互页面

## 10.进行项目部署测试