Key：venderId\_flightCode\_clazzCode，如："621906\_FM9531\_B"

Value：{

"id": 11609147,

"activityId": 1800, 活动id

"price": 300, 票面价

"reducePrice": 20, 立减金额

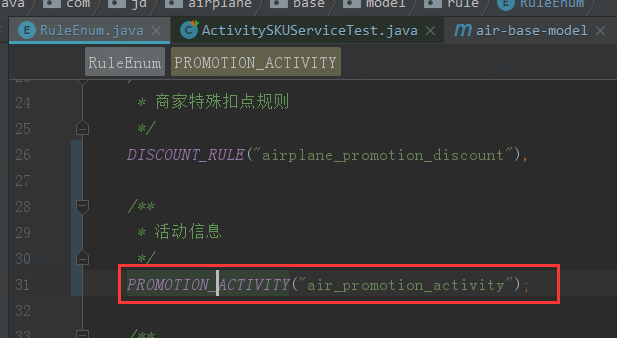
"Type": "0", 促销类型

"Beans": 0, 京豆数量

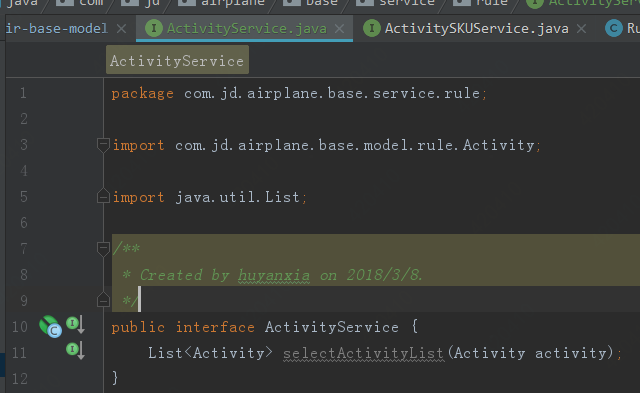
"Key": "621906\_FM9531\_B" 缓存key

}

1. 将air-base-model中的activity和activitySku定义好，自测没有问题，打jar包
2. 在air-base-service中的RuleEnum.java添加*PROMOTION\_ACTIVITY*("air\_promotion\_activity");



1. 创建接口ActivityService



1. 接口RuleFacadeService中继承ActivityService
2. 将model和service打成jar包给宾哥
3. package com.jd.airplane.base.background.service.promotion;  
     
   import com.jd.airplane.base.background.service.BaseService;  
   import com.jd.airplane.base.model.rule.Activity;  
   import com.jd.airplane.base.model.rule.ActivitySKU;  
   import com.jd.airplane.base.service.repository.rule.ActivityRepository;  
   import com.jd.airplane.base.service.repository.rule.ActivitySKURepository;  
   import com.jd.airplane.infra.bean.config.CacheKeyConfig;  
   import com.jd.airplane.infra.cache.JdCacheUtils;  
   import com.jd.airplane.infra.response.Page;  
   import com.jd.fastjson.JSON;  
   import org.apache.commons.collections.CollectionUtils;  
   import org.apache.commons.lang3.StringUtils;  
   import org.slf4j.Logger;  
   import org.slf4j.LoggerFactory;  
   import org.springframework.beans.factory.annotation.Autowired;  
   import org.springframework.beans.factory.annotation.Qualifier;  
     
   import java.text.ParseException;  
   import java.util.\*;  
     
   */\*\*  
    \* 促销缓存索引命令  
    \* Created by yangyanping on 2018/2/5.  
    \*/*public class PromotionCacheService implements BaseService<Object,Boolean> {  
    private static final Logger *LOGGER* = LoggerFactory.*getLogger*(PromotionCacheService.class);  
    //活动  
    @Autowired  
    private ActivityRepository activityRepository;  
    @Autowired  
    private ActivitySKURepository activitySKURepository;  
    @Autowired  
    @Qualifier("jdCacheUtils")  
    private JdCacheUtils jdCacheUtils;  
     
    */\*\*  
    \* activitySKU缓存-key  
    \*/* final static String *PROM\_ACTIVITY\_SKU\_CACHE\_KEY* = "air\_base\_prom\_activity\_sku";  
     
    final static long *cacheTime* = 24 \* 3600;  
     
    public Boolean execute(Object obj) {  
    *LOGGER*.info("promotion cache data start !");  
    //获取有效活动  
    List<Activity> activities = null;  
    //查询当前时间在活动时间范围内的活动  
    try {  
    activities = getActivities();  
    } catch (Exception e) {  
    *LOGGER*.error("get activities is error ! ", e);  
    return Boolean.*FALSE*;  
    }  
    for (Activity act : activities) {  
    for (int pageNo = 1; ; pageNo++) {  
    Page<ActivitySKU> activitySKUS = activitySKURepository.selectPage(pageNo, 100, new ActivitySKU.ActivitySKUBuilder().activityId(act.getActivityId()).build());  
     
    if (activitySKUS == null || CollectionUtils.*isEmpty*(activitySKUS.getData())) {  
    *LOGGER*.info("the last page, has no activitySku");  
    break;  
    }  
     
    List<ActivitySKU> activitySKUlist = new ArrayList<ActivitySKU>();  
    for (ActivitySKU activitySKU : activitySKUS.getData()) {  
    String promActSKUPostFix= String.*valueOf*(activitySKU.getVenderId()) + "\_" + activitySKU.getFlightCode() + "\_" + activitySKU.getClazzCode();  
     
    String promActSKUPostFixKey = activitySKU.getVenderId()+ "\_" + activitySKU.getFlightCode().substring(0,2);  
    Set<String> activitySkuCacheKeySet = new HashSet<String>();  
    String activitySkuCacheKeyStr = jdCacheUtils.get(promActSKUPostFixKey);  
    if (StringUtils.*isNotBlank*(activitySkuCacheKeyStr)) {  
    activitySkuCacheKeySet = JSON.*parseArray*(activitySkuCacheKeyStr);  
    }  
    activitySkuCacheKeySet.add(promActSKUPostFix);  
    jdCacheUtils.set(promActSKUPostFixKey,JSON.*toJSONString*(activitySkuCacheKeySet));  
    *LOGGER*.info("write cache key：" + promActSKUPostFixKey + ", value:" + JSON.*toJSONString*(activitySKUlist));  
    jdCacheUtils.set(promActSKUPostFixKey, JSON.*toJSONString*(activitySKUlist));  
     
     
     
     
    String activitySkuCacheKey = CacheKeyConfig.*AIR\_BASE\_RULE\_PROM\_ACT\_SKU\_PREFIX*.concat(promActSKUPostFix) ;  
    //先从缓存中取该key的activitySku信息（三个维度下的数据可能多条）  
    String activitySKUStr = jdCacheUtils.get(activitySkuCacheKey);  
    if (StringUtils.*isNotBlank*(activitySKUStr)) {  
    activitySKUlist = JSON.*parseArray*(activitySKUStr, ActivitySKU.class);  
    }  
    activitySKUlist.add(activitySKU);  
    *LOGGER*.info("write cache key：" + activitySkuCacheKey + ", value:" + JSON.*toJSONString*(activitySKUlist));  
    jdCacheUtils.set(activitySkuCacheKey, JSON.*toJSONString*(activitySKUlist));  
    }  
    }  
    }  
    //设置缓存时间  
    //jdCacheUtils.expire(PROM\_ACTIVITY\_SKU\_CACHE\_KEY, cacheTime);  
    //redis的rename方法  
    //增量和全量缓存，全量不需要rename  
    return Boolean.*TRUE*;  
    }  
     
    private List<Activity> getActivities() throws ParseException {  
    //获得当前的活动数据  
    Date date = new Date();  
    List<Activity> activities = activityRepository.selectValidActivityList(new Activity.ActivityBuilder().beginTime(date.getTime()).endTime(date.getTime()).build());  
     
    return CollectionUtils.*isEmpty*(activities) ? Collections.*EMPTY\_LIST* : activities;  
    }  
   }



