# 1、项目使用

## 1、从前台接受的日期来到后台，我们要对它进行日期的比较



@Override  
 public List<SignetApplyRecord> getVerifySignetData(Long admId, Pageable pageable, SignetQuery query) {  
  
 query.setStartDate(DateHelper.getDateFirstTime(query.getStartDate()));  
 query.setEndDate(DateHelper.getDateLastTime(query.getEndDate()));  
 query.setOffset(pageable.getOffset());  
 query.setLimit(pageable.getPageSize());  
  
 Integer authType = sysAuthService.getAuthUserDataType(admId, MenuConstans.***VERIFY\_SIGNET***);  
 if (authType.compareTo(UserBindAuth.***DATATYPE\_INHERIT***) == 0) {  
  
 query.setCurrAdmId4Auth(admId);  
 } else {  
   
 }  
 List<SignetApplyRecord> list = signetMapper.findVerifySignetList(query);  
 if (CollectionUtils.isNotEmpty(list)) {  
 for (SignetApplyRecord record : list){  
 record.setAttachsList(signetApplyRecordAttachsDAO.findByPid(record.getId()));  
 }  
 }  
  
 return list;  
 }

## 2、获取一段时间内的日期，（在曲线统计的时候用到过）

List<Date> lDate = findDates(

DateHelper.convertString2Date(startDate,DateHelper.***YYYY\_MM\_DD***), DateHelper.convertString2Date(endDate,DateHelper.YYYY\_MM\_DD));

/\*\*  
 \* 获取一段时间之内的日期  
 \* @param dBegin  
 \* @param dEnd  
 \* @return  
 \*/  
private List<Date> findDates(Date dBegin, Date dEnd) {  
 List lDate = new ArrayList();  
 lDate.add(dBegin);  
 Calendar calBegin = Calendar.getInstance();  
 calBegin.setTime(dBegin);  
 Calendar calEnd = Calendar.getInstance();  
 calEnd.setTime(dEnd);  
 // 测试此日期是否在指定日期之后  
 while (dEnd.after(calBegin.getTime()))  
 {  
 // 根据日历的规则，为给定的日历字段添加或减去指定的时间量  
 calBegin.add(Calendar.***DAY\_OF\_MONTH***, 1);  
 lDate.add(calBegin.getTime());  
 }  
 return lDate;  
}

## 3、取得某天中最早时间和最晚时间

for (Date date : lDate) {  
 xlist.add(DateHelper.convertDate2String(date, DateHelper.***YYYY\_MM\_DD***));  
 String sdate = DateHelper.convertDate2String(date, DateHelper.***YYYY\_MM\_DD***) + " 00:00:00";  
 String edate = DateHelper.convertDate2String(date, DateHelper.***YYYY\_MM\_DD***) + " 23:59:59";  
 Date stime = DateHelper.convertString2Date(sdate, DateHelper.***YYYY\_MM\_DD\_HH\_MM\_SS***);  
 Date etime = DateHelper.convertString2Date(edate, DateHelper.***YYYY\_MM\_DD\_HH\_MM\_SS***);  
 List<UserAppsSpreadData> spreadIdsList = userAppMapper.getSpreadIdsList(4l, stime, etime);  
 y1list.add(Long.valueOf(spreadIdsList.size()));  
}

## 4、取得当月的第一天 时间String （crm 项目/sale/曲线）

private String getMonthFirstDay() {  
 Calendar calendar = Calendar.getInstance();  
 calendar.set(Calendar.***DAY\_OF\_MONTH***,  
 calendar.getActualMinimum(Calendar.***DAY\_OF\_MONTH***));  
 SimpleDateFormat firstDay= new SimpleDateFormat(DateHelper.***YYYY\_MM\_DD***);  
 return firstDay.format(calendar.getTime());  
}

# 5、取得前面几个月的日期

### 1、getYm(-i)当前月份的接触上减去（6，5，4，3，2，1） 也就是显示8，9，10，11，12，1 月的曲线

#### 比如当前月份为2月 参数为-6结果为2018-08

private String getYm(int num) {  
 Date date = new Date();  
 Calendar calendar = Calendar.getInstance();  
 calendar.setTime(date);  
 calendar.add(Calendar.***MONTH***, num);  
 return DateHelper.convertDate2String(calendar.getTime(), DateHelper.***YYYY\_MM***);  
}

## 2、取得上个月的第一天，和上个月的最后一天（这里是28号，不过也差不多就是个这） 减去几就是上几个月的值

#### 比如 当前月为2月 参数为-1 就是为1月份的第一天和1月份的最后一天

private Date getFirstYMDHMS(int num) {  
 Calendar calendar = Calendar.getInstance();  
 calendar.set(Calendar.***DAY\_OF\_MONTH***,  
 calendar.getActualMinimum(Calendar.***DAY\_OF\_MONTH***));  
 calendar.add(Calendar.***MONTH***, num);  
 String time = DateHelper.convertDate2String(calendar.getTime(), DateHelper.***YYYY\_MM\_DD***)+" 00:00:00";  
 return DateHelper.convertString2Date(time, DateHelper.***YYYY\_MM\_DD\_HH\_MM\_SS***);  
}  
  
private Date getLastYMDHMS(int num) {  
 Date date = new Date();  
 Calendar calendar = Calendar.getInstance();  
 calendar.set(Calendar.***DAY\_OF\_MONTH***,  
 calendar.getActualMaximum(Calendar.***DAY\_OF\_MONTH***));  
 calendar.add(Calendar.***MONTH***, num);  
 String time = DateHelper.convertDate2String(calendar.getTime(), DateHelper.***YYYY\_MM\_DD***)+" 23:59:59";  
 return DateHelper.convertString2Date(time, DateHelper.***YYYY\_MM\_DD\_HH\_MM\_SS***);  
}

# 6、将long转化为date或者string类型的date

public Map<String,Object> stampToDate(String s){  
 String res;  
 SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyy-MM-dd");  
 long lt = new Long(s);  
 Date date = new Date(lt);  
 Map<String,Object> map = new HashMap<>();  
 map.put("date",date)；  
 res = simpleDateFormat.format(date);  
 map.put("dateString",res)；  
  
 return map;  
}

\* Created by j.sh on 2015/5/15.

public class DateHelper extends DateUtils {  
  
 private static final Logger ***logger*** = LoggerFactory.getLogger(DateHelper.class);  
  
 public static final String ***YYYY\_MM*** = "yyyy-MM";  
 public static final String ***YYYY\_MM\_DD*** = "yyyy-MM-dd";  
 public static final String ***YYYY\_MM\_DD\_HH\_MM\_SS*** = "yyyy-MM-dd HH:mm:ss";  
 public static final String ***YYYY\_MM\_DD\_HH\_MM*** = "yyyy-MM-dd HH:mm";  
 public static final String ***YYYY\_MM\_DD\_HH*** = "yyyy-MM-dd HH";  
  
 public static final String ***YYYYMMDD*** = "yyyyMMdd";  
 public static final String ***YYYYMMDDHH*** = "yyyyMMddHH";  
 public static final String ***YYYYMMDD\_SLANT*** = "yyyy/MM/dd";  
 public static final String ***YYYYMM*** = "yyyyMM";  
 public static final String ***YYYY*** = "yyyy";  
 public static final String ***MMDD*** = "MMdd";  
 public static final String ***HHMMSS*** = "HHmmss";  
  
  
 public static final String ***YYYYMMDDHHMMSS*** = "yyyyMMddHHmmss";  
  
 public static final String ***HH\_MM*** = "HH:mm";  
 public static Date convertString2Dat;  
  
 /\*\*  
 \* 格式化日期为字符串  
 \* @param date  
 \* @param pattern  
 \* @return  
 \*/  
 public static String convertDate2String(Date date,String pattern){  
 if (date == null)  
 return null;  
  
 SimpleDateFormat dateFormat = new SimpleDateFormat(pattern);  
 return dateFormat.format(date);  
 }  
  
 public static String convertDate2String(Date date){  
 return convertDate2String(date, ***YYYY\_MM\_DD\_HH\_MM\_SS***);  
 }  
  
 /\*\*  
 \* 转换字符串为日期  
 \* @param pattern 日期格式  
 \* @param dateStr 日期字符串  
 \* @return date  
 \*/  
 public static Date convertString2Date(String dateStr,String pattern ){  
 if(StringUtils.isBlank(dateStr)){  
 throw new NullPointerException("dateStr is null");  
 }  
 SimpleDateFormat df = new SimpleDateFormat(pattern);  
 try {  
 return new Date(df.parse(dateStr).getTime());  
 } catch (ParseException pe) {  
 ***logger***.error(pe.getMessage(),pe);  
 throw new RuntimeException("date parse error"+dateStr);  
 }  
 }  
  
  
 public static String formatDateString(String dateStr,String pattern){  
 if(StringUtils.isBlank(dateStr)){  
 throw new NullPointerException("dateStr is null");  
 }  
 SimpleDateFormat df = new SimpleDateFormat(pattern);  
 try {  
 Date date = new Date(df.parse(dateStr).getTime());  
 return df.format(date);  
 } catch (ParseException pe) {  
 ***logger***.error(pe.getMessage(),pe);  
 throw new RuntimeException("date parse error"+dateStr);  
 }  
 }  
  
 /\*\*  
 \* 滚动日期,  
 \* @param point 要滚动的时间  
 \* @param field 要滚动的类别,传入参数为Calendar.YEAR,Calendar.MONTH,DATE...等  
 \* @param num 滚动数量,正负数都可以,负数为向前滚动,  
 \* @return 结果  
 \* 例: DateHelper.rollDate(new Date(),Calendar.DATE,-12); 当前日期的前12天  
 \* 例: DateHelper.rollDate(new Date(),Calendar.MONTH,2); 向后滚动两个月  
 \*/  
 public static Date rollDate(Date point,int field,int num){  
 Calendar cal = Calendar.getInstance();  
 cal.setTime(point);  
 cal.add(field,num);  
 return cal.getTime();  
 }  
  
  
 /\*\*  
 \* 获取当前小时最初的时间  
 \* @param date  
 \* @return  
 \*/  
 public static Date getHourFirstTime(Date date){  
 if(date == null) return null;  
 Calendar cal = Calendar.getInstance();  
 cal.setTime(date);  
 cal.set(Calendar.***MINUTE***, 0);  
 cal.set(Calendar.***SECOND***, 0);  
 cal.set(Calendar.***MILLISECOND***, 0);  
 return new Date(cal.getTimeInMillis());  
 }  
  
 /\*\*  
 \* 获取当前小时最后的时间  
 \* @param date  
 \* @return  
 \*/  
 public static Date getHourLastTime(Date date){  
 if(date == null) return null;  
 Calendar cal = Calendar.getInstance();  
 cal.setTime(date);  
 cal.set(Calendar.***MINUTE***, 59);  
 cal.set(Calendar.***SECOND***, 59);  
 cal.set(Calendar.***MILLISECOND***,0);  
 return new Date(cal.getTimeInMillis());  
 }  
  
 /\*\*  
 \* 得到一天的最早的时间  
 \* @param date one day  
 \* @return date  
 \*/  
 public static Date getDateFirstTime(Date date){  
 if(date == null) return null;  
 Calendar cal = Calendar.getInstance();  
 cal.setTime(date);  
 cal.set(Calendar.***HOUR\_OF\_DAY***, 0);  
 cal.set(Calendar.***MINUTE***, 0);  
 cal.set(Calendar.***SECOND***, 0);  
 cal.set(Calendar.***MILLISECOND***, 0);  
  
 return new Date(cal.getTimeInMillis());  
 }  
  
 /\*\*  
 \* 获取一天最晚的时间  
 \* @param date one day  
 \* @return date  
 \*/  
 public static Date getDateLastTime(Date date){  
 if(date == null) return null;  
 Calendar cal = Calendar.getInstance();  
 cal.setTime(date);  
 cal.set(Calendar.***HOUR\_OF\_DAY***, 23);  
 cal.set(Calendar.***MINUTE***, 59);  
 cal.set(Calendar.***SECOND***, 59);  
 cal.set(Calendar.***MILLISECOND***, 0);  
 return new Date(cal.getTimeInMillis());  
 }  
  
  
 /\*\*  
 \* 获取传入日期所在月份的开始日期  
 \* @param date date  
 \* @return 日期所在月份的开始日期  
 \*/  
 public static Date getMonthStartTime(Date date){  
 Calendar cal = Calendar.getInstance();  
 cal.setTime(date);  
 cal.set(Calendar.***DATE***,cal.getActualMinimum(Calendar.***DATE***));  
 return getDateFirstTime(cal.getTime());  
 }  
  
 /\*\*  
 \* 获取传入日期所在月份的结束日期  
 \* @param date date  
 \* @return 日期所在月份的结束日期  
 \*/  
 public static Date getMonthEndTime(Date date){  
 Calendar cal = Calendar.getInstance();  
 cal.setTime(date);  
 cal.set(Calendar.***DATE***,cal.getActualMaximum(Calendar.***DATE***));  
 return getDateLastTime(cal.getTime());  
 }  
  
 /\*\*  
 \* 获取当年最后时间  
 \* @param date  
 \* @return  
 \*/  
 public static Date getYearLastTime(Date date){  
 Calendar calendar = Calendar.getInstance();  
 calendar.setTime(date);  
 calendar.set(Calendar.***MONTH***,Calendar.***DECEMBER***);  
 calendar.set(Calendar.***DAY\_OF\_YEAR***,calendar.getActualMaximum(Calendar.***DAY\_OF\_YEAR***));  
 return getDateLastTime(calendar.getTime());  
 }  
  
 /\*\*  
 \* 获取传入日期的上个月的开始日期  
 \* @param date date  
 \* @return 上一个月的开始日期  
 \*/  
 public static Date getLastMonthStartTime(Date date){  
 Calendar cal = Calendar.getInstance();  
 cal.setTime(date);  
 cal.add(Calendar.***MONTH***, -1);  
 //设置为1号,当前日期既为本月第一天  
 cal.set(Calendar.***DATE***,cal.getActualMinimum(Calendar.***DATE***));  
 return getDateFirstTime(cal.getTime());  
 }  
  
 /\*\*  
 \* 获取传入日期上个月的结束日期  
 \* @param date date  
 \* @return 日期所在月份的结束日期  
 \*/  
 public static Date getLastMonthEndTime(Date date){  
 Calendar cal = Calendar.getInstance();  
 cal.setTime(date);  
 cal.add(Calendar.***MONTH***, -1);  
 //设置为1号,当前日期既为本月第一天  
 cal.set(Calendar.***DATE***,cal.getActualMaximum(Calendar.***DATE***));  
 return getDateLastTime(cal.getTime());  
 }  
  
 /\*\*  
 \* 获取传入日期的上个周的开始日期  
 \* @param date date  
 \* @return 上一个周的开始日期 认为是周一  
 \*/  
 public static Date getLastWeekStartTime(Date date){  
 Calendar calendar = Calendar.getInstance();  
 calendar.setTime(date);  
 calendar.add(Calendar.***WEEK\_OF\_YEAR***,-1);  
 return getThisWeekStartTime(calendar.getTime());  
 }  
  
 /\*\*  
 \* 获取传入日期的昨天的日期  
 \* @param date date  
 \* @return 昨天  
 \*/  
 public static Date getYesterDayStartTime(Date date){  
 Calendar cal = Calendar.getInstance();  
 cal.setTime(date);  
 cal.add(Calendar.***DATE***, -1);  
 //设置为1号,当前日期既为本月第一天  
 return getDateFirstTime(cal.getTime());  
 }  
  
 /\*\*  
 \* 获取传入日期上个周的结束日期  
 \* @param date date  
 \* @return 日期所在周的结束日期  
 \*/  
 public static Date getLastWeekEndTime(Date date){  
 Calendar calendar = Calendar.getInstance();  
 calendar.setTime(date);  
 calendar.add(Calendar.***WEEK\_OF\_YEAR***,-1);  
 return getThisWeekEndTime(calendar.getTime());  
 }  
  
  
 /\*\*  
 \* 获取传入日期的本周的开始日期  
 \* @param date date  
 \* @return 上一个周的开始日期 认为是周一  
 \*/  
 public static Date getThisWeekStartTime(Date date){  
 Calendar calendar = Calendar.getInstance();  
 calendar.setFirstDayOfWeek(Calendar.***MONDAY***);  
 calendar.setTime(date);  
 calendar.set(Calendar.***DAY\_OF\_WEEK***,  
 calendar.getFirstDayOfWeek());  
 return getDateFirstTime(calendar.getTime());  
 }  
  
 /\*\*  
 \* 获取传入日期本周的结束日期  
 \* @param date date  
 \* @return 日期所在周的结束日期  
 \*/  
 public static Date getThisWeekEndTime(Date date){  
 Calendar calendar = Calendar.getInstance();  
 calendar.setFirstDayOfWeek(Calendar.***MONDAY***);  
 calendar.setTime(date);  
 calendar.set(Calendar.***DAY\_OF\_WEEK***,  
 calendar.getFirstDayOfWeek() + 6);  
 return getDateLastTime(calendar.getTime());  
 }  
  
 /\*\*  
 \* 获取hour  
 \* @param date one day  
 \* @return date  
 \*/  
 public static int getDateHour(Date date){  
 if(date == null) return 0;  
 Calendar cal = Calendar.getInstance();  
 cal.setTime(date);  
 return cal.get(Calendar.***HOUR\_OF\_DAY***);  
 }  
  
 /\*\*  
 \* 获取hour  
 \* @param date one day  
 \* @return date  
 \*/  
 public static int getDateMinute(Date date){  
 if(date == null) return 0;  
 Calendar cal = Calendar.getInstance();  
 cal.setTime(date);  
 return cal.get(Calendar.***MINUTE***);  
 }  
  
 /\*\*  
 \* 通过 long 得到Date  
 \* @param date one day  
 \* @return date  
 \*/  
 public static Date getDateByTime(Long date){  
 if(date == null) return null;  
 Calendar cal = Calendar.getInstance();  
 cal.setTimeInMillis(date);  
 cal.set(Calendar.***SECOND***, 0);  
 cal.set(Calendar.***MILLISECOND***, 0);  
 return new Date(cal.getTimeInMillis());  
 }  
  
 /\*\*  
 \* 第二个时间相对第一个时间的天数差  
 \* @param date1  
 \* @param date2  
 \* @return  
 \*/  
 public static long differ(Date date1, Date date2){  
 return date2.getTime() / 86400000 - date1.getTime() / 86400000;  
 }  
  
 public static long differDays(Date date1, Date date2){  
 return Days.daysBetween(new DateTime(date1).withTimeAtStartOfDay(),new DateTime(date2).withTimeAtStartOfDay()).getDays();  
 }  
  
 public static long differSeconds(Date date1, Date date2){  
 return Seconds.secondsBetween(new DateTime(date1),new DateTime(date2)).getSeconds();  
 }  
  
 public static long differMinutes(Date date1, Date date2){  
 return Minutes.minutesBetween(new DateTime(date1).withSecondOfMinute(0),new DateTime(date2).withSecondOfMinute(0)).getMinutes();  
 }  
  
 public static long differHours(Date date1, Date date2){  
 return Hours.hoursBetween(new DateTime(date1),new DateTime(date2)).getHours();  
 }  
  
 public static double differDoubleHours(Date date1, Date date2){  
 return BigDecimal.valueOf(date2.getTime() - date1.getTime()).divide(BigDecimal.valueOf(3600000),1,BigDecimal.***ROUND\_HALF\_UP***).doubleValue();  
 }  
  
 public static boolean isSameDay(Date date1, Date date2) {  
 return StringUtils.equals(convertDate2String(date1,DateHelper.***YYYYMMDD***),convertDate2String(date2,DateHelper.***YYYYMMDD***));  
 }  
  
 /\*\*  
 \* 得到一个理论上软件生命周期无法到达的时间  
 \* @return date  
 \*/  
 public static Date getRemoteDate(){  
 Calendar cal = Calendar.getInstance();  
 cal.set(Calendar.***YEAR***, 2999);  
 cal.set(Calendar.***MONTH***, 11);  
 cal.set(Calendar.***DATE***, 31);  
 cal.set(Calendar.***HOUR\_OF\_DAY***, 23);  
 cal.set(Calendar.***MINUTE***, 59);  
 cal.set(Calendar.***SECOND***, 59);  
 cal.set(Calendar.***MILLISECOND***, 0);  
  
 return cal.getTime();  
 }  
  
 /\*\*  
 \* 得到一个多点广告开始的时间  
 \* @return date  
 \*/  
 public static Date getInitDate(){  
 Calendar cal = Calendar.getInstance();  
 cal.set(Calendar.***YEAR***, 2016);  
 cal.set(Calendar.***MONTH***, 11);  
 cal.set(Calendar.***DATE***, 1);  
 cal.set(Calendar.***HOUR\_OF\_DAY***, 0);  
 cal.set(Calendar.***MINUTE***, 0);  
 cal.set(Calendar.***SECOND***, 0);  
 cal.set(Calendar.***MILLISECOND***, 0);  
  
 return cal.getTime();  
 }  
  
  
 /\*\*  
 \* 获取两个日期之间的所有日期  
 \* @param pattern 日期格式  
 \* @param startDate 开始日期  
 \* @param endDate 结束日期  
 \* @return Set<String>  
 \*/  
 public static List<String> getDates(String pattern, Date startDate, Date endDate) {  
 List<String> result = new ArrayList<>();  
 Calendar startCalendar = Calendar.getInstance();  
 Calendar endCalendar = Calendar.getInstance();  
 SimpleDateFormat df = new SimpleDateFormat(pattern);  
 startCalendar.setTime(startDate);  
 endCalendar.setTime(endDate);  
 result.add(df.format(startCalendar.getTime()));  
 while(true){  
 startCalendar.add(Calendar.***DAY\_OF\_MONTH***, 1);  
 if(startCalendar.getTimeInMillis() <= endCalendar.getTimeInMillis()){  
 result.add(df.format(startCalendar.getTime()));  
 }else{  
 break;  
 }  
 }  
 return result;  
 }  
  
 /\*\*  
 \* 获取两个日期之间的所有周的周一日期  
 \* @param pattern 日期格式  
 \* @param startDate 开始日期  
 \* @param endDate 结束日期  
 \* @return Set<String>  
 \*/  
 public static List<String> getWeeksMonday(String pattern, Date startDate, Date endDate) {  
 List<String> result = new ArrayList<>();  
 Calendar startCalendar = Calendar.getInstance();  
 Calendar endCalendar = Calendar.getInstance();  
 SimpleDateFormat df = new SimpleDateFormat(pattern);  
 startCalendar.setTime(getThisWeekStartTime(startDate));  
 startCalendar.setFirstDayOfWeek(Calendar.***MONDAY***);  
 endCalendar.setTime(endDate);  
 result.add(df.format(getThisWeekStartTime(startDate)));  
 while(true){  
 startCalendar.add(Calendar.***WEEK\_OF\_YEAR***, 1);  
 if(startCalendar.getTimeInMillis() <= endCalendar.getTimeInMillis()){  
 result.add(df.format(startCalendar.getTime()));  
 }else{  
 break;  
 }  
 }  
 return result;  
 }  
  
 /\*\*  
 \* 根据ios回传的时间获取正常时间  
 \* @param apptime  
 \* @return  
 \*/  
 public static Date getValidDateFromIOS(Long apptime){  
 try {  
 return new Date((Long.valueOf(apptime)+978307200)\*1000L);  
 } catch (Exception e) {  
 return null;  
 }  
 }  
  
 /\*\*  
 \* 获取传入日期对应月份 总共跨越了几个周  
 \* @param date  
 \* @return  
 \*/  
 public static int getMaxWeekNumOfMonth(Date date){  
 if(date == null) return 0;  
 Calendar cal = Calendar.getInstance();  
 cal.setTime(date);  
 cal.setFirstDayOfWeek(Calendar.***MONDAY***);  
 return cal.getActualMaximum(Calendar.***WEEK\_OF\_MONTH***);  
 }  
  
 /\*\*  
 \* 日期在一年中第几周  
 \* @param date  
 \* @return  
 \*/  
 public static int getWhichWeek(Date date) throws Exception {  
 if(date == null) return 0;  
  
 Calendar calendar1 = Calendar.getInstance();  
 calendar1.setTime(date);  
 calendar1.setFirstDayOfWeek(Calendar.***MONDAY***);  
 return calendar1.get(Calendar.***WEEK\_OF\_YEAR***);  
  
 }  
  
 /\*\*  
 \* 获取两个日期 跨越了几个周  
 \* @param startDate  
 \* @param endDate  
 \* @return  
 \*/  
 public static int getWeekNum(Date startDate, Date endDate){  
 if(startDate == null || endDate == null) return 0;  
  
 Calendar calendar1 = Calendar.getInstance();  
 calendar1.setTime(startDate);  
 calendar1.setFirstDayOfWeek(Calendar.***MONDAY***);  
 Integer w1 = calendar1.get(Calendar.***WEEK\_OF\_YEAR***);  
  
 Calendar calendar2 = Calendar.getInstance();  
 calendar2.setTime(endDate);  
 calendar2.setFirstDayOfWeek(Calendar.***MONDAY***);  
 Integer w2 = calendar2.get(Calendar.***WEEK\_OF\_YEAR***);  
  
 return w2 - w1 + 1;  
 }  
  
 /\*\*  
 \* 检查当前时间是否为工作日（周一至周五）  
 \* @return  
 \*/  
 public static boolean checkCurrentDayIsWorkDay(){  
 boolean flag = false;  
 Calendar curr\_calendar = Calendar.getInstance();  
 int weeks = curr\_calendar.get(Calendar.***DAY\_OF\_WEEK***);  
  
// Integer start\_minute = 9 \* 60;  
// Integer end\_minute = 18 \* 60;  
  
// Integer current\_minute = curr\_calendar.get(Calendar.HOUR\_OF\_DAY) \* 60 + curr\_calendar.get(Calendar.MINUTE);  
  
 if(weeks >=2 && weeks <=6){  
// if(current\_minute >= start\_minute && current\_minute <= end\_minute && ( weeks >=2 && weeks <=6 )){  
 flag = true;  
// System.out.println(String.format("当前时间: [%s] 在工作日9点~18点范围内",curr\_calendar.getTime()));  
 }  
 return flag;  
 }  
  
 public static void main(String[] args) {  
// Calendar calendar = Calendar.getInstance();  
// int date = calendar.get(Calendar.DATE);  
//  
// System.out.println(date);  
  
// System.out.println(differDays(convertString2Date("2017-07-16 00:00:00",YYYY\_MM\_DD\_HH\_MM\_SS),convertString2Date("2017-07-17 23:59:59",YYYY\_MM\_DD\_HH\_MM\_SS)));  
  
// System.out.println(getDateHour(DateHelper.rollDate(convertString2Date("2017-07-16 00:00:00",YYYY\_MM\_DD\_HH\_MM\_SS), Calendar.HOUR\_OF\_DAY, 1)));  
  
 System.***out***.println(getWeeksMonday(***YYYY\_MM\_DD***,convertString2Date("2017-10-01 00:00:00",***YYYY\_MM\_DD\_HH\_MM\_SS***),convertString2Date("2017-10-31 23:00:00",***YYYY\_MM\_DD\_HH\_MM\_SS***)));  
// System.out.println(convertDate2String(getThisWeekStartTime(convertString2Date("2017-10-01",YYYY\_MM\_DD))));  
  
 */\* List<String> list = FileUtils.readLines(new File("/Users/bianjingshuai/Downloads/1212.csv"),"UTF-8");  
  
 for (String c : list) {  
 String[] split = c.split(",");  
  
  
 String sql = "insert into lgzx\_member (userid,userpwd,mobile,cityid,status,score) values ('"+split[0]+"','e10adc3949ba59abbe56e057f20f883e','"+split[0]+"',253,1,10);";  
  
 System.out.println(sql);  
  
 }\*/* }

# 7、js日期格式转化

function timestampToTime(timestamp) {

        var date = new Date(timestamp \* 1000);//时间戳为10位需\*1000，时间戳为13位的话不需乘1000

        Y = date.getFullYear() + '-';

        M = (date.getMonth()+1 < 10 ? '0'+(date.getMonth()+1) : date.getMonth()+1) + '-';

        D = date.getDate() + ' ';

        h = date.getHours() + ':';

        m = date.getMinutes() + ':';

        s = date.getSeconds();

        return Y+M+D+h+m+s;

    }

    timestampToTime(1403058804);

    console.log(timestampToTime(1403058804));//2014-06-18 10:33:24

# 8、获取最近前多少天的开始时间

@Override  
public List<CouponExpireChange> getRecentlyDayData(Integer day) {  
 Date startDate = getRecentlyDayFirstTime(-day);  
 Date endDate = DateHelper.getDateLastTime(new Date());  
 return couponExpireChangeMapper.findByRecentlyDay(startDate,endDate);  
}

private Date getRecentlyDayFirstTime(Integer num) {  
 Date date = new Date();  
 Calendar calendar = Calendar.getInstance();  
 calendar.setTime(date);  
 calendar.add(Calendar.***DATE***, num);  
 String time = DateHelper.convertDate2String(calendar.getTime(), DateHelper.***YYYY\_MM\_DD***)+" 00:00:00";  
 return DateHelper.convertString2Date(time,DateHelper.***YYYY\_MM\_DD\_HH\_MM\_SS***);  
}