# DateUtils，Date，Calendar函数用法详解

DateUtils的函数：

/\*\*

\* 注：

\* 1.一下大部分(也可能是全部)date都可以换成calendar也就是日历时间

\* 2.calendar.get(Calendar.MONTH)获取的月比正常月份少一个月

\* 3.calendar.get(Calendar.DAY\_OF\_WEEK)获取到的指定时间是一周的第几天周日为第一天

\* 4.以下注释未必百分之百正确

\*/

SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

Date date = new Date();

Date date2 = new Date();

Date parse = null;

Calendar calendar = Calendar.getInstance();

int i = calendar.get(Calendar.DAY\_OF\_WEEK);

System.out.println(i);

System.out.println(dateFormat.format(date));//2018-08-09 17:31:27

//对指定的日期新增日

date = DateUtils.addDays(date, 1);

System.out.println(dateFormat.format(date));

//对指定的日期减去5小时

date = DateUtils.addHours(date, -5);

System.out.println(dateFormat.format(date));

//对指定的日期加上1个月

date = DateUtils.addMonths(date, 1);

System.out.println(dateFormat.format(date));

//获取一天的毫秒值

long millisPerDay = DateUtils.MILLIS\_PER\_DAY;

System.out.println(millisPerDay);

//获取一小时的毫秒值

millisPerDay = DateUtils.MILLIS\_PER\_HOUR;

System.out.println(millisPerDay);

//月范围，从星期一开始(单个用貌似没什么意义往后看)

int rangeMonthMonday = DateUtils.RANGE\_MONTH\_MONDAY;

System.out.println(rangeMonthMonday);

//检查两个日期对象是否同一天忽略时间

System.out.println(DateUtils.isSameDay(date, date2));

//检查两个日期对象是否代表同一时刻，此方法比较两个日期精确到毫秒数

System.out.println(DateUtils.isSameInstant(date, date2));

//今天在一年中是第几天

long fragmentInDays = DateUtils.getFragmentInDays(date2, Calendar.YEAR);

//今天在一个月中是第几天

long fragmentInDays2 = DateUtils.getFragmentInDays(date2, Calendar.MONTH);

System.out.println(fragmentInDays2);

System.out.println("fragmentInDays========="+fragmentInDays);

//现在是今年的第几个小时

long fragmentInHours = DateUtils.getFragmentInHours(date2, Calendar.YEAR);

System.out.println("fragmentInHours="+fragmentInHours);

//向上舍入日期-年(比如当前日期是2018-08-10 10:25:35那么调用这个函数之后是2019-01-01 00:00:00)

date2 = DateUtils.ceiling(date2, Calendar.YEAR);

System.out.println(dateFormat.format(date2));

Date date3 = new Date();

//向上舍入日期-月(比如当前日期是2018-08-10 10:25:35那么调用这个函数之后是2018-09-01 00:00:00)

date3 = DateUtils.ceiling(date3, Calendar.MONTH);

System.out.println(dateFormat.format(date3));

Date date4 = new Date();

//DateUtils.RANGE\_WEEK\_CENTER--以当前时间为中心找出前3天后3天凑够一周

//DateUtils.RANGE\_MONTH\_MONDAY--我的理解是以周一为起点以这个月为主体生成n个完整的周，且必须将这个月的所有日期都包含进去，如果这个月的1号不是周一那就去上个月去最近的一个周一，如果30号不是周日，去下个月去最近的一个周日

//DateUtils.RANGE\_MONTH\_SUNDAY--我的理解是以周日为起点以这个月为主体生成n个完整的周，且必须将这个月的所有日期都包含进去，如果这个月的1号不是周日那就去上个月去最近的一个周日，如果30号不是周一，去下个月去最近的一个周一

//DateUtils.RANGE\_WEEK\_MONDAY--我的理解是以周一为起点生成这一周的日期DateUtils.RANGE\_WEEK\_SUNDAY就不做多余解释。

//DateUtils.RANGE\_WEEK\_RELATIVE--以今天为起点向后生成一周的日期

Iterator<Calendar> iterator = DateUtils.iterator(date4, DateUtils.RANGE\_WEEK\_RELATIVE);

while (iterator.hasNext()) {

Calendar next = iterator.next();

System.out.println("iterator.next======"+dateFormat.format(next.getTime()));

}

try {

parse = dateFormat.parse("2018-09-02 00:00:00");

} catch (ParseException e) {

e.printStackTrace();

}

calendar.setTime(parse);

//获取指定时间是这周的第几天(周日为第一天)

i = calendar.get(Calendar.DAY\_OF\_WEEK);

System.out.println(i);

Date date5 = new Date();

//这里可定义多种个时以备用

String[] parsePatterns = new String[]{"yyyy-MM-dd HH:mm:ss","yyyyMMdd","yyyy年MM月dd日 HH时mm分ss秒"};

try {

//尝试用parsePatterns中各种不同的日期格式解析代表时间的字符串str,意义在于定义一个格式组,解析字符串时间的时候方便

date5 = DateUtils.parseDate("2018-09-02 00:00:00", parsePatterns);

System.out.println(dateFormat.format(date5));

//尝试用parsePatterns中各种不同的日期格式解析代表时间的字符串str,意义在于定义一个格式组,解析字符串时间的时候方便

date5 = DateUtils.parseDate("2018902",parsePatterns);

System.out.println(dateFormat.format(date5));

//相对于parseDate更严格一些同样是yyyyMMdd格式paresDate输入2018902不会报错但是会转成一个时间而这个会报错，建议用这个

date5 = DateUtils.parseDateStrictly("2018-09-16 12:30:31", parsePatterns);

System.out.println(dateFormat.format(date5));

} catch (ParseException e1) {

e1.printStackTrace();

}

//时间四舍五入-时(分钟超过30或等于30时就进一位比如2018-09-02 23:30:00四舍五入后就是2018-09-03 00:00:00)

date5 = DateUtils.round(date5, Calendar.HOUR);

System.out.println(dateFormat.format(date5));

//时间四舍五入-日(时超过12或等于12时就进一位比如2018-09-02 12:30:31四舍五入后就是2018-09-03 00:00:00和分秒没有关系)

date5 = DateUtils.round(date5, Calendar.DATE);

System.out.println(dateFormat.format(date5));

//时间四舍五入-月(如果超过15时就进一位比如2018-09-16 13:30:31四舍五入后就是2018-10-01 00:00:00和时分秒没有关系)

date5 = DateUtils.round(date5, Calendar.MONTH);

System.out.println("date5===="+dateFormat.format(date5));

Date date6 = new Date();

//日期截取-时2018-08-10 14:00:00其他不多说日月年截取

date6 = DateUtils.truncate(date6, Calendar.HOUR);

System.out.println("date6===="+dateFormat.format(date6));

//先截断，再比较。返回的int的值是多少无所谓，重要的是符号。负数代表date6在date5前，正数代表date6在date5后。0代表相等。

int truncatedCompareTo = DateUtils.truncatedCompareTo(date6, date5, Calendar.MONTH);

System.out.println(truncatedCompareTo);

//相当于判断truncatedCompareTo的结果是否为0。

boolean truncatedEquals = DateUtils.truncatedEquals(date6, date5, Calendar.MONTH);

System.out.println(truncatedEquals);

Date和Calendar：

/\*\*

\* 注：

\* 1.calendar.get(Calendar.MONTH)获取的月比正常月份少一个月

\* 2.calendar.get(Calendar.DAY\_OF\_WEEK)获取到的指定时间是一周的第几天周日为第一天

\* 3.以下注释未必百分之百正确

\*/

SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

//new当前时间

Date date2 = new Date();

//根据毫秒实例一个时间2009-02-14 07:31:30

Date date3 = new Date(1234567890100L);

//创建当前日历时间

Calendar calendar = Calendar.getInstance();

//获取年

int a = calendar.get(Calendar.YEAR);

//获取日

int b = calendar.get(Calendar.DATE);

//获取月的索引--一月是0

int c = calendar.get(Calendar.MONTH);

//获取日是月份中的第几天

int d = calendar.get(Calendar.DAY\_OF\_MONTH);

//获取日在一周中的第几天--周日是第一天

int e = calendar.get(Calendar.DAY\_OF\_WEEK);

//日所在的周是这个月的第几周

int f = calendar.get(Calendar.DAY\_OF\_WEEK\_IN\_MONTH);

//日在年份的第几天

int g = calendar.get(Calendar.DAY\_OF\_YEAR);

//获取当前时

int h = calendar.get(Calendar.HOUR);

//获取时是一天的第几个小时

int i = calendar.get(Calendar.HOUR\_OF\_DAY);

//获取分

int j = calendar.get(Calendar.MINUTE);

//2018-1-7-1-4-1-213-5-17-6-7-7

System.out.println(a+"-"+b+"-"+c+"-"+d+"-"+e+"-"+f+"-"+g+"-"+h+"-"+i+"-"+j+"-");

//给日历时间赋一个时间

calendar.setTime(date3);

//获取日期--也可以使用上面的各种方法

Date time = calendar.getTime();

//给日历的年月日时分秒加数

/\*\*

\* 可用的常量有：

\* MONTH,YEAR,DATE,HOUR,MINUTE,SECOND

\* 这些就够用

\*/

calendar.add(Calendar.SECOND, 10);

System.out.println("秒加10====="+dateFormat.format(calendar.getTime()));

//给日历的年月日时分秒设值

/\*\*

\* 可用的常量有：

\* HOUR\_OF\_DAY,DAY\_OF\_MONTH,YEAR,HOUR\_OF\_DAY,MINUTE,SECOND

\* 其他的没用明白这几个就够了

\*/

calendar.set(Calendar.SECOND, 222);

System.out.println("set后的日期为："+dateFormat.format(calendar.getTime()));

//清空年月日时分秒-清空秒

calendar.clear(Calendar.SECOND);

System.out.println("清空秒："+dateFormat.format(calendar.getTime()));

Calendar instance = Calendar.getInstance();

//是否在某个日期前正数是负数不是

int compareTo = calendar.compareTo(instance);

System.out.println("compareTo====="+compareTo);

//比较两个日期是否一样

boolean equals = calendar.equals(calendar);

System.out.println("equals====="+equals);

//查询当前（年、月、周）拥有的最大值

int actualMaximum = calendar.getActualMaximum(Calendar.MONTH);

System.out.println("actualMaximum====="+actualMaximum);

//查询当前（年、月、周）拥有的最大值

calendar.getActualMinimum(Calendar.MONTH);

//查询（年、月、周）拥有的最大值区别于getActualMinimum就是这个函数不与当前时间相关单纯地获取年月日最大值

int maximum = calendar.getMaximum(Calendar.HOUR);

System.out.println("maximum====="+maximum);

//判断日期/时间的解释是否为宽松的(不懂)

boolean lenient = calendar.isLenient();

System.out.println("lenient====="+lenient);

//不是很懂

String displayName = calendar.getDisplayName(Calendar.MONTH, 1, new Locale("haha", "hehe"));

System.out.println("displayName====="+displayName);

//获取毫秒值1234567890100

long time2 = time.getTime();

System.out.println(time2);

//是否是指定时间后的时间

boolean after = time.after(date2);

System.out.println(after);

//是否是指定时间前的时间

boolean before = time.before(date2);

System.out.println(before);

//常用的时间格式转换类

String format2 = dateFormat.format(date3);

System.out.println(format2);