01 日期类

JAVAEE高级

一：System类

> 1、System类在java.lang.System包下；
>
> 2、System类提供的public static long currentTimeMillis()用来返回当前时间与1970年1月1日0时0分0秒之间以毫秒为单位的时间差
>
> 3、此方法适于计算时间差

```
    @Test
    public void getTest01() throws InterruptedException {
        long currentTimeMillis = System.currentTimeMillis();
        // 休眠100毫秒
        Thread.sleep(100);
        long currentTimeMillis01 = System.currentTimeMillis();
        System.out.println(currentTimeMillis);
        System.out.println(currentTimeMillis01);
        System.out.println(currentTimeMillis01 - currentTimeMillis);
    }
```

二：Date类

> 1、Date类在java.util.Date包下；
>
> 2、特定的瞬间，精确到毫秒；
>
> 3、注意java.sql.Date是继承了java.util.Date

```
    @Test
    public void getTest04() {
        // java.util.Date
```

```
        //参数为0是1970年1月1日以来通过00:00:格林威治时间0毫秒数
        Date date = new Date(0);
        //为什么打印出来的是01 08:00:00呢?
        //<东八区的标准时间为08:00>
        System.out.println("date:" + date.toString());
    }


    @Test
    public void getTest02(){
        // java.util.Date
        Date date = new Date();
        System.out.println("date:"+date.toString());
       //java.sql.Date
        java.sql.Date date2 = new java.sql.Date(12312312311l);
        System.out.println("date2:"+date2.toString());
    }
```

> 4、东八区的标准时间为08:00

1、常用的构造方法

①、Date( )

> 1、使用Date类的无参数构造方法创建的对象可以获取本地当前时间

②、Date(long date)

2、常用方法

①、getTime()

> 1、返回自 1970 年 1 月 1 日 00:00:00 GMT 以来此 Date 对象表示的毫秒数

②、toString()

> 1、把此 Date 对象转换为以下形式的 String (dow mon dd hh:mm:ss zzz yyyy)
>
> 2、dow 是一周中的某一天 (Sun, Mon, Tue, Wed, Thu, Fri, Sat)，zzz是时间标准

```java
@Test
    public void getTest02(){
        // java.util.Date
        Date date = new Date();
        System.out.println("date:"+date.toString());
        long time = date.getTime();
        System.out.println("获取Long型的值:"+time);
        Date date3 = new Date(time);
        System.out.println("date3:"+date3);

        java.sql.Date date2 = new java.sql.Date(12312312311l);
        System.out.println("date2:"+date2.toString());
    }
```

3、不易于国际化

> 1、Date类的API不易于国际化，大部分被废弃了
>
> 2、java.text.SimpleDateFormat类

三：SimpleDateFormat类

> 1、java.text.SimpleDateFormat类
>
> 2、是用于格式化和解析日期的具体类
>
> 3、允许进行格式化（日期->文本）解析（文本->日期）
>
> 4、DateFormat类很少使用

```java
//DateFormat是抽象类，不能进行实例化
     DateFormat dateFormat = new DateFormat();
    DateFormat dateInstance = DateFormat.getDateInstance();
    //使用的
     SimpleDateFormat simpleDateFormat = new SimpleDateFormat();
```

1、格式化

1、SimpleDateFormat()
默认的模式和语言环境创建对象
2、public String format(Date date)
方法格式化时间对象date

```
    @Test
    public void getTest03(){
        //格式化一：
          SimpleDateFormat simpleDateFormat = new SimpleDateFormat();
          String format = simpleDateFormat.format(new Date());
        System.out.println("format:"+format);
    }
```

3、public SimpleDateFormat(String pattern)
该构造方法可以用参数指定的格式创建一个对象，该对象调用

```
    @Test
    public void getTest03() {
        // 格式化二：
        SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
        String format = simpleDateFormat.format(new Date());
        System.out.println("format:" + format);
    }
```

其他格式化的参数(可查API)

Date and Time Pattern	Result
`"yyyy.MM.dd G 'at' HH:mm:ss z"`	`2001.07.04 AD at 12:08:56 PDT`
`"EEE, MMM d, ''yy"`	`Wed, Jul 4, '01`
`"h:mm a"`	`12:08 PM`
`"hh 'o''clock' a, zzzz"`	`12 o'clock PM, Pacific Daylight Time`
`"K:mm a, z"`	`0:08 PM, PDT`

Date and Time Pattern	Result
`"yyyyy.MMMMM.dd GGG hh:mm aaa"`	`02001.July.04 AD 12:08 PM`
`"EEE, d MMM yyyy HH:mm:ss Z"`	`Wed, 4 Jul 2001 12:08:56 -0700`
`"yyMMddHHmmssZ"`	`010704120856-0700`
`"yyyy-MM-dd'T'HH:mm:ss.SSSZ"`	`2001-07-04T12:08:56.235-0700`
`"yyyy-MM-dd'T'HH:mm:ss.SSSXXX"`	`2001-07-04T12:08:56.235-07:00`
`"YYYY-'W'ww-u"`	`2001-W27-3`

2、解析

> 1、public Date parse(String source)
> 给定字符串的开始解析文本，以生成一个日期

```
@Test
    public void getTest03() throws ParseException {
        SimpleDateFormat simpleDateFormat = new SimpleDateFormat();
        String format = simpleDateFormat.format(new Date());
        System.out.println("format:" + format);
        // 解析
        Date parse = simpleDateFormat.parse(format);
        System.out.println("parse:" + parse);
        System.out.println("----------------------");
        SimpleDateFormat sdf = new SimpleDateFormat("EEE, d MMM yyyy HH:mm:ss Z");
        String formatsdf = sdf.format(new Date());
        System.out.println("format:" + formatsdf);
        // 解析
        Date sdfDate = sdf.parse(formatsdf);
        System.out.println("sdfDate:" + sdfDate);
    }
```

3、案例：<你来的世界的天数>

> 1、求出你出生到现在来到的这个世界的目前一共有多少天

```
    @Test
    public void getDaysNumber() throws ParseException{
        //出生日期和今天的日期
        String buffer="2017-1-1";
        String end="2018-1-1";
        //格式化时间
        SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyy-MM-dd");
        //进行转换为日期对象
        Date bufferDate = simpleDateFormat.parse(buffer);
        Date endDate = simpleDateFormat.parse(end);

        long tmie=endDate.getTime()-bufferDate.getTime();
        System.out.println(tmie/1000/3600/24);
    }
```

4、案例：<渔夫打鱼>

> 1、<渔夫三天打鱼两天晒网>:请求某天是打鱼还是晒网 ？

```
    /**
     * 案例 <渔夫三天打鱼两天晒网>:请求某天是打鱼还是晒网
     */
    @Test
    public void getVoid() throws ParseException {
        String date1 = "2018-12-24";
        String date2 = "2018-12-25";
        int days = getDays(date1, date2);
        if (days % 5 == 0 || days % 5 == 4) {
            System.out.println("晒网");
        } else {
            System.out.println("打鱼");
        }
    }

    public int getDays(String date1, String date2) throws ParseException {
        SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyy-
```

```
MM-dd");
            Date parse1 = simpleDateFormat.parse(date1);
            Date parse2 = simpleDateFormat.parse(date2);
            long lo = parse2.getTime() - parse1.getTime();
            return (int) lo / 1000 / 3600 / 24 + 1;
        }
```

四：Calendar(日历)类

> 1、Calendar在java.util.Calendar包下
>
> 2、 Calendar是一个抽象基类，主用用于完成日期字段之间相互操作的功能

1、获取Calendar实例

> 1、Calendar.getInstance()方法
>
> 2、子类GregorianCalendar的构造器
>
> 3、一个Calendar的实例是系统时间的抽象表示
>
> 4、通过get(int field)方法来取得想要的时间信息如:（YEAR、MONTH、HOUR_OF_DAY 、MINUTE、SECOND、DAY_OF_MONTH ）

```
    @Test
    public void getVoid02() throws ParseException {
        Calendar instance = Calendar.getInstance();
        int j = instance.get(Calendar.YEAR);
        System.out.println("年:" + j);
        int k = instance.get(Calendar.MONTH + 1);
        System.out.println("月:" + k);
        int l = instance.get(Calendar.HOUR_OF_DAY);
        System.out.println("小时:" + l);
        int m = instance.get(Calendar.MINUTE);
        System.out.println("分:" + m);
        int i = instance.get(Calendar.DAY_OF_MONTH);
        System.out.println("当月的第几天:" + i);
        instance.add(Calendar.DAY_OF_MONTH, 2);
        i = instance.get(Calendar.DAY_OF_MONTH);
```

```
            System.out.println("当月的第几天:" + i);
            instance.set(Calendar.DAY_OF_MONTH,20);
            Date time = instance.getTime();
            System.out.println("time:"+time);
        }
```

5、获取某天的是星期几

```
    @Test
    public void getVoid03() throws ParseException {
        Calendar instance = Calendar.getInstance();
        //返回的每周的第一天:周日是第一天，周六是最后一天
        int i = instance.get(Calendar.DAY_OF_WEEK);
        System.out.println(i);
        String [] arr={"星期日","星期一","星期二","星期三","星期四","星期五",
"星期六"};
        System.out.println(arr[i-1]);
    }
```

1-1、Calendar Date SimpleDateFormat转换

1、Calendar转换成Date

```
Calendar cal = Calendar.getInstance();
Date date = cal.getTime;
```

2、Date 转换成 Calendar

```
Date date = new Date();
Calendar cal = Calendar.getInstance();
Cal.setTime(date );
```

3、SimpleDateFormat

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

```

```
String date = sdf.format(new Date());
```

3、SimpleDateFormat与Calendar

```
Calendar calendat = Calendar.getInstance();
SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
String dateStr = sdf.format(calendar.getTime());
```

2、案例<获取任意年份是平年还是闰年>

1、任意输入一个年份,进行判断该年份是闰年还是平年

```
@Test
    public void getVoid04() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("输入年份,进行判断是平年还是闰年.....");
        String nextLine = scanner.nextLine();
        int parseInt = Integer.parseInt(nextLine);
        boolean bool = getBool(parseInt);
        if (bool) {
            System.out.println("该年份为润年");
        } else {
            System.out.println("该年份为平年");
        }

    }

    public boolean getBool(int year) {
        Calendar instance = Calendar.getInstance();
        // 设那一年的3月1日
        instance.set(year, 2, 1);
        // 将日减一1
        instance.add(Calendar.DAY_OF_MONTH, -1);
        // 进行判断2月的天数
        return instance.get(Calendar.DAY_OF_MONTH) == 29;

    }
```