63从零开始学Java63之时间格式化怎么实现?

作者：孙玉昌，昵称【**一一哥**】，另外【**壹壹哥**】也是我哦

千锋教育高级教研员、CSDN博客专家、万粉博主、阿里云专家博主、掘金优质作者

前言

在上一篇文章中，**壹哥**给大家讲解了Java里的Date和Calendar类，大家应该学到了如何创建时间日期对象，并利用该对象进行一些关于时间的操作。但是在上一篇文章中，我们知道，默认情况下构造出来的时间对象，它的时间格式并不适合我们阅读。并且在开发时，pc端、Android端、iOS端等展示的时间格式可能也并不完全一样，那么我们有没有办法进行时间格式的自定义设置呢？这个需求当然是可以实现的，这就需要用到我们今天要学习的内容，这就是时间的格式化！

----------前戏已做完，精彩即开始----------

全文大约【3200】字，不说废话，只讲可以让你学到技术、明白原理的纯干货！本文带有丰富的案例及配图视频，让你更好地理解和运用文中的技术概念，并可以给你带来具有足够启迪的思考.....

配套开源项目资料

Github：

GitHub – SunLtd/LearnJava

Gitee：

一一哥/从零开始学Java

一. 时间模式字符串

在进行时间格式化之前，我们先来了解一下时间模式字符串。所谓的时间模式字符串，就是可以用来指定时间格式的一种模式，在此模式中，一些ASCII字母被保留为模式字母，具有特别的含义，如下表所示：

字母	描述	示例
y	四位年份	2020
M	月份	July or 07
d	一个月的日期	10
h	A.M./P.M. (1~12)格式小时	12
H	一天中的小时 (0~23)	22
m	分钟数	30
s	秒数	55
S	毫秒数	234
E	星期几	Tuesday

D	一年中的日子	360
F	一个月中第几周的周几	2 (second Wed. in July)
w	一年中第几周	40
W	一个月中第几周	1
a	A.M.(上午)/P.M.(下午)	PM
k	一天中的小时(1~24)	24
K	A.M./P.M. (0~11)格式小时	10
z	时区	Eastern Standard Time
'	文字定界符	Delimiter

上面表格中的这些符号，我们尽量要记住，因为开发时经常会用到这些符号。掌握了这些符号之后，我们就可以继续学习日期与时间的格式化技能了。

二. printf()格式化方法

如果我们只是在项目的个别某个地方，想要简单地实现日期时间的格式化，其实使用printf()方法就可以。printf()方法格式化时间和日期时，需要使用两个字母进行格式化操作，一般是以 %t 开头，且以下面表格中的某个字母结尾。

转 换 符	说　明	示　例
c	包括全部日期和时间信息	星期六 十月 27 14:21:20 CST 2007
F	"年-月-日"格式	2007-10-27
D	"月/日/年"格式	10/27/07
r	"HH:MM:SS PM"格式（12时制）	02:25:51 下午
T	"HH:MM:SS"格式（24时制）	14:28:16
R	"HH:MM"格式（24时制）	14:28

接下来我们通过一个案例来看看printf()方法是怎么进行格式化时间的。

```java
import java.util.Date;

/**
 * @author ——哥Sun
 */
public class Demo07 {

    public static void main(String[] args) {
        // 初始化 Date 对象
        Date date = new Date();
        //c：全部日期和时间
        System.out.printf("全部日期和时间信息：%tc%n", date);
        //f：年-月-日
        System.out.printf("年-月-日的格式：%tF%n", date);
        //d：月/日/年
        System.out.printf("月/日/年的格式：%tD%n", date);
        //r：HH:MM:SS PM
        System.out.printf("HH:MM:SS PM的格式(12时制)：%tr%n", date);
        //t：HH:MM:SS格式（24时制）
        System.out.printf("HH:MM:SS的格式(24时制)：%tT%n", date);
        //R：HH:MM格式（24时制）
        System.out.printf("HH:MM的格式(24时制)：%tR", date);

        //用格式化字符串,指出要被格式化的参数索引，索引必须紧跟在%后面，而且必须以$结束
        //使用toString()显示日期和时间
        System.out.printf("%1$s %2$tB %2$td, %2$tY","Due date:", date);
    }
}
```

printf()方法进行格式化时，需要结合上表中的几个符号。但是说实话，这种格式化方式并不是很常用，主要适用于少量的个别地方。如果我们的项目中有多处需要进行格式化的地方，尽量还是不要使用这种方式。因为如果你需要重复提供日期，利用这种方式来格式化时间就有点复杂，且不好进行后期的维护。

三. DateFormat类

其实我们在Java项目中对日期进行格式化，主要是利用一些日期格式化类，比如DateFormat及其子类。

1. 简介

DateFormat是负责日期/时间格式化的抽象类，它可以用与语言无关的方式格式化并解析日期或时间。它的子类(如SimpleDateFormat)允许进行日期的格式化，将日期转为文本；也可以进行文本的解析，将文本转为日期。
我们在创建 DateFormat对象时不能使用 new关键字，要使用 DateFormat类中的getDateInstance()静态方法，如下所示：

```java
DateFormat df = DateFormat.getDatelnstance();
```

2. 静态常量

DateFormat中给我们提供了几个常用的静态常量，用于方便我们进行格式化样式的设置，如下所示：

- SHORT：纯数字，如12.5.10 或 5:30pm；
- MEDIUM：较长，如May 10，2023；
- LONG：更长，如May 12，2023 或 11:15:32am；
- FULL：完全指定，如Tuesday、May 10、2022 AD 或 11:l5:42am CST。

3. 常用方法

我们在创建了一个DateFormat对象后，就可以使用该对象中的方法来对日期/时间进行格式化了，DateFormat中的常用方法如下表所示：

方法	描述
String format(Date date)	将 Date 格式化日期/时间字符串
Calendar getCalendar()	获取与此日期/时间格式相关联的日历
static DateFormat getDateInstance()	获取具有默认格式化风格和默认语言环境的日期格式
static DateFormat getDateInstance(int style)	获取具有指定格式化风格和默认语言环境的日期格式

static DateFormat getDateInstance(int style, Locale locale)	获取具有指定格式化风格和指定语言环境的日期格式
static DateFormat getDateTimeInstance()	获取具有默认格式化风格和默认语言环境的日期/时间格式
static DateFormat getDateTimeInstance(int dateStyle,int timeStyle)	获取具有指定日期/时间格式化风格和默认语言环境的日期/时间格式
static DateFormat getDateTimeInstance(int dateStyle,int timeStyle,Locale locale)	获取具有指定日期/时间格式化风格和指定语言环境的日期/时间格式
static DateFormat getTimeInstance()	获取具有默认格式化风格和默认语言环境的时间格式
static DateFormat getTimeInstance(int style)	获取具有指定格式化风格和默认语言环境的时间格式
static DateFormat getTimeInstance(int style, Locale locale)	获取具有指定格式化风格和指定语言环境的时间格式
void setCalendar(Calendar newCalendar)	为此格式设置日历
Date parse(String source)	将给定的字符串解析成日期/时间

4. 基本使用

接下来我们通过一个案例来看看DateFormat的用法。这个案例，主要是给大家介绍DateFormat类的方法与静态常量该如何使用，对日期进行不同风格的格式化。

```java
import java.text.DateFormat;
import java.util.Date;
import java.util.Locale;

/**
 * @author ——哥Sun
 */
public class Demo08 {

    public static void main(String[] args) {
        // 获取不同格式化风格和中国环境的日期
        DateFormat df1 = DateFormat.getDateInstance(DateFormat.SHORT, Locale.CHINA);
        //DateFormat df2 = DateFormat.getDateInstance(DateFormat.FULL, Locale.CHINA);
        //DateFormat df3 = DateFormat.getDateInstance(DateFormat.MEDIUM, Locale.CHINA);
        //DateFormat df4 = DateFormat.getDateInstance(DateFormat.LONG, Locale.CHINA);

        // 将不同格式化风格的日期格式化为日期字符串
        String date = df1.format(new Date());

        // 获取不同格式化风格和中国环境的时间
        DateFormat df2 = DateFormat.getTimeInstance(DateFormat.SHORT, Locale.CHINA);
        // 将不同格式化风格的时间格式化为时间字符串
        String time = df2.format(new Date());

        // 输出日期和时间
        System.out.println("SHORT格式:" + date+" "+time);
    }

}
```

四. SimpleDateFormat类

1. 简介

虽然我们已经有了DateFormat，但有时候这个类并不能满足我们的实际开发需求。此时我们可以进一步使用它的子类，比如SimpleDateFormat来进行更多的操作。

SimpleDateFormat是一个以与语言环境有关的方式来格式化和解析日期的具体类，它具有格式化(日期转文本)、解析(文本转日期)和规范化的功能。相对DateFormat来说，SimpleDateFormat具有更高的灵活性，可以让我们选择任何自定义的日期/时间格式，进行个性化设置。

2. 构造方法

SimpleDateFormat是一个具体的子类，所以我们是可以通过new的方式来创建对象的。
SimpleDateFormat类为我们提供了如下4个构造方法：

> - SimpleDateFormat()：用默认的格式和语言环境，构造一个SimpleDateFormat对象；
> - SimpleDateFormat(String pattern)：用指定的格式和默认的语言环境，构造一个SimpleDateF ormat对象；
> - SimpleDateFormat(String pattern,Locale locale)：用指定的格式和指定的语言环境，构造一个 SimpleDateFormat对象；
> - SimpleDateFormat(String pattern,DateFormatSymbols formatSymbols)：用指定的格式和指定的格式化语法来构造一个SimpleDateFormat对象。

3. 自定义格式化常用字母

SimpleDateFormat自定义格式中常用的字母及含义如表 2 所示。

字母	含义	示例
y	年份。一般用 yy 表示两位年份，yyyy 表示 4 位年份	使用 yy 表示的年扮，如 11； 使用 yyyy 表示的年份，如 2011
M	月份。一般用 MM 表示月份，如果使用 MMM，则会根据语言环境显示不同语言的月份	使用 MM 表示的月份，如 05； 使用 MMM 表示月份，在 Locale.CHINA 语言环境下，如“十月”；在 Locale.US 语言环境下，如 Oct
d	月份中的天数。一般用 dd 表示天数	使用 dd 表示的天数，如 10
D	年份中的天数。表示当天是当年的第几天， 用 D 表示	使用 D 表示的年份中的天数，如 295

E	星期几。用 E 表示，会根据语言环境的不同， 显示不同语言的星期几	使用 E 表示星期几，在 Locale.CHINA 语言环境下，如“星期四”；在 Locale.US 语言环境下，如 Thu
H	一天中的小时数（0~23)。一般用 HH 表示小时数	使用 HH 表示的小时数，如 18
h	一天中的小时数（1~12)。一般使用 hh 表示小时数	使用 hh 表示的小时数，如 10 (注意 10 有可能是 10 点，也可能是 22 点)
m	分钟数。一般使用 mm 表示分钟数	使用 mm 表示的分钟数，如 29
s	秒数。一般使用 ss 表示秒数	使用 ss 表示的秒数，如 38
S	毫秒数。一般使用 SSS 表示毫秒数	使用 SSS 表示的毫秒数，如 156

4. 基本使用

接下来我们通过一个案例，来展示SimpleDateFormat的格式化和解析用法。

```java
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;

/**
 * @author ——哥Sun
 */
public class Demo09 {

    public static void main(String[] args) {
        //设置日期时间格式化模式，这个模式是根据需求自定义展示的，也可以是"yyyy年MM月dd日 E HH点 mm分 ss秒"等格式
        SimpleDateFormat format=new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
        //进行日期格式化
        String date = format.format(new Date());
        System.out.println("格式后的结果:" + date);

        try {
            //解析日期，将一个时间字符串解析为Date类型，这里有可能会产生解析异常
            String time="2022-02-12 17:30:39";
            Date date2 = format.parse(time);
            System.out.println("解析后的结果:" + date2);
        } catch (ParseException e) {
            e.printStackTrace();
        }
    }
}
```

在上述案例中，format()方法用于将Date格式化为String字符串，parse()方法用于将String字符串解析为Date类型。其中yyyy是完整的公元年，MM是月份，dd是日期，HH:mm:ss 是时、分、秒。这里有的格式大写，有的格式小写，例如MM是月份，mm是分，HH是24小时制，而hh则是12小时制。

5. 配套视频

与本节内容配套的视频链接如下：

https://player.bilibili.com/player.html?bvid=BV1Ja411x7XB&p=139&page=139

----------正片已结束，来根事后烟----------

五. 结语

至此，**壹哥**就把日期的格式化操作给大家讲解完毕了。今天的内容其实并不难，大家只需要把一些常用的构造方式及方法、常量记一下即可。如果你独自学习觉得有很多困难，可以加入**壹哥**的学习互助群，大家一起交流学习。