**Java simple date format**

[**JAVA DATE**](http://beginnersbook.com/category/technology/java-guide/java-date/)

The Utility class SimpleDateFormat extends DateFormat . SimpleDateFormat  is a concrete class for formatting and parsing dates in a locale-sensitive manner and normalization. It allows you to start by choosing any user-defined patterns for date-time formatting.

Do you wanna know more about “How to format dates in any format”? Read my post : [**Date format in Java**](http://beginnersbook.com/2013/04/java-date-format/)

**Time Format Syntax:**

To specify the time format use a time pattern string. In this pattern, all ASCII letters are reserved as pattern letters, which are defined as the following:

|  |  |  |  |
| --- | --- | --- | --- |
| Symbol | Meaning | Presentation | Example |
| G | era designator | Text | AD |
| Y | year | Number | 2003 |
| M | month in year | Text & Number | July & 07 |
| D | day in month | Number | 25 |
| H | hour in am/pm (1~12) | Number | 12 |
| H | hour in day (0~23) | Number | 0 |
| M | minute in hour | Number | 45 |
| S | second in minute | Number | 55 |
| S | millisecond | Number | 958 |
| E | day in week | Text | Tuesday |

|  |  |  |  |
| --- | --- | --- | --- |
| D | day in year | Number | 289 |
| F | day of week in month | Number | 2 (2nd Wed in July) |
| W | week in year | Number | 47 |
| W | week in month | Number | 4 |
| A | am/pm marker | Text | AM |
| K | hour in day (1~24) | Number | 24 |
| K | hour in am/pm (0~11) | Number | 0 |
| Z | time zone | Text | Pacific Standard Time |
| ‘ | escape for text | Delimiter | ‘hour’ h -> “hour 9″ |
| ” | single quote | Literal | ss”SSS -> “45’876″ |

The count of pattern letters determine the format.  
Text:  4 or more pattern letters–use full form, for < 4 – use short or abbreviated form if one exists.

Number:  The minimum number of digits. Shorter numbers are zero-padded to this amount. Year is handled specially; that is, if the count of ‘y’ is 2, the Year will be truncated to 2 digits.

Text & Number:  3 or over, use text, otherwise use number.

Any characters in the pattern that are not in the ranges of [‘a’..’z’] and [‘A’..’Z’] will be treated as quoted text. For instance, characters like ‘#’ and ‘@’ will appear in the resulting time text even they are not embraced within single quotes.

A pattern containing any invalid pattern letter will result in a thrown exception during formatting or parsing.

**Examples:**

*Format Pattern Result*  
*“yyyy.MM.dd G ‘at’ hh:mm:ss z” 2003.07.10 AD at 15:48:56 PDT*  
*“EEE, MMM d, ”yy” Tue, July 10, ’03*  
*“h:mm a” 12:48 PM*  
*“hh ‘o””clock’, zzzz” 12 o’clock PM, Pacific Daylight Time*  
*“K:mm a, z” 0:00 PM, PST*  
*“yyyyy.MMMMM.dd GGG hh:mm aaa” 2003.July.10 AD 12:48 PM*

**Java Code Sample:**

[code language=”java”]  
/\* This function in example below returns the current  
date time in format similar to “04/25/2013 12:47:56 AM”\*/

import java.text.SimpleDateFormat;  
import java.util.Date;  
public static String getCurrentTime(){

/\*This will return current date time like  
this example:11/25/2013 12:47:56 AM\*/  
String TimeFormat = "MM/dd/yyyy hh:mm:ss a"  
Date currDate = new Date();  
SimpleDateFormat smpDateFormatter = new SimpleDateFormat(TimeFormat);  
String currTime= smpDateFormatter.format(currDate);  
return currTime;  
}[/code]