

Servlet 处理日期

使用 Servlet 的最重要的优势之一是，可以使用核心 Java 中的大多数可用的方法。本章将讲解 Java 提供的 `java.util` 包中的 `Date` 类，这个类封装了当前的日期和时间。

`Date` 类支持两个构造函数。第一个构造函数初始化当前日期和时间的对象。

```
Date( )
```

下面的构造函数接受一个参数，该参数等于 1970 年 1 月 1 日午夜以来经过的毫秒数。

```
Date(long millisec)
```

一旦您有一个可用的 `Date` 对象，您可以调用下列任意支持的方法来使用日期：

序号	方法 & 描述
1	boolean after(Date date) 如果调用的 <code>Date</code> 对象中包含的日期在 <code>date</code> 指定的日期之后，则返回 <code>true</code> ，否则返回 <code>false</code> 。
2	boolean before(Date date) 如果调用的 <code>Date</code> 对象中包含的日期在 <code>date</code> 指定的日期之前，则返回 <code>true</code> ，否则返回 <code>false</code> 。
3	Object clone() 重复调用 <code>Date</code> 对象。
4	int compareTo(Date date) 把调用对象的值与 <code>date</code> 的值进行比较。如果两个值是相等的，则返回 0。如果调用对象在 <code>date</code> 之前，则返回一个负值。如果调用对象在 <code>date</code> 之后，则返回一个正值。
5	int compareTo(Object obj) 如果 <code>obj</code> 是 <code>Date</code> 类，则操作等同于 <code>compareTo(Date)</code> 。否则，它会抛出一个 <code>ClassCastException</code> 。
6	boolean equals(Object date) 如果调用的 <code>Date</code> 对象中包含的时间和日期与 <code>date</code> 指定的相同，则返回 <code>true</code> ，否则返回 <code>false</code> 。
7	long getTime() 返回 1970 年 1 月 1 日以来经过的毫秒数。
8	int hashCode() 为调用对象返回哈希代码。

9	void setTime(long time) 设置 time 指定的时间和日期，这表示从 1970 年 1 月 1 日午夜以来经过的时间（以毫秒为单位）。
10	String toString() 转换调用的 Date 对象为一个字符串，并返回结果。

获取当前的日期和时间

在 Java Servlet 中获取当前的日期和时间是很容易的。您可以使用一个简单的 Date 对象的 *toString()* 方法来输出当前的日期和时间，如下所示：

```
package com.runoob.test;

import java.io.IOException;
import java.io.PrintWriter;
import java.util.Date;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * Servlet implementation class CurrentDate
 */
@WebServlet("/CurrentDate")
public class CurrentDate extends HttpServlet {
    private static final long serialVersionUID = 1L;

    public CurrentDate() {
        super();
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        String title = "显示当前的日期和时间";
        Date date = new Date();
        String docType = "<!DOCTYPE html> \n";
        out.println(docType +
            "<html>\n" +
            "<head><title>" + title + "</title></head>\n" +
            "<body bgcolor=\"#f0f0f0\">\n" +
```

```
"<h1 align=\"center\">" + title + "</h1>\n" +  
"<h2 align=\"center\">" + date.toString() + "</h2>\n" +  
"</body></html>");  
  
}  
  
}
```

现在，让我们来编译上面的 Servlet，并在 web.xml 文件中创建适当的条目：

```
<?xml version="1.0" encoding="UTF-8"?>  
<web-app>  
  <servlet>  
    <servlet-name>CurrentDate</servlet-name>  
    <servlet-class>com.runoob.test.CurrentDate</servlet-class>  
  </servlet>  
  <servlet-mapping>  
    <servlet-name>CurrentDate</servlet-name>  
    <url-pattern>/TomcatTest/CurrentDate</url-pattern>  
  </servlet-mapping>  
</web-app>
```

然后通过访问 <http://localhost:8080/TomcatTest/CurrentDate> 来调用该 Servlet。这将会产生如下的结果：

显示当前的日期和时间

Mon Aug 29 10:42:25 CST 2016

尝试刷新 URL <http://localhost:8080/TomcatTest/CurrentDate>，每隔几秒刷新一次您都会发现显示时间的差异。

日期比较

正如上面所提到的，您可以在 Servlet 中使用所有可用的 Java 方法。如果您需要比较两个日期，以下是方法：

- 您可以使用 `getTime()` 来获取两个对象自 1970 年 1 月 1 日午夜以来经过的时间（以毫秒为单位），然后对这两个值进行比较。
- 您可以使用方法 `before()`、`after()` 和 `equals()`。由于一个月里 12 号在 18 号之前，例如，`new Date(99, 2, 12).before(new Date(99, 2, 18))` 返回 `true`。
- 您可以使用 `compareTo()` 方法，该方法由 `Comparable` 接口定义，由 `Date` 实现。

使用 SimpleDateFormat 格式化日期

`SimpleDateFormat` 是一个以语言环境敏感的方式来格式化和解析日期的具体类。`SimpleDateFormat` 允许您选择任何用户定义的日期时间格式化的模式。

让我们修改上面的实例，如下所示：

```
package com.runoob.test;

import java.io.IOException;
import java.io.PrintWriter;
import java.text.SimpleDateFormat;
import java.util.Date;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * Servlet implementation class CurrentDate
 */
@WebServlet("/CurrentDate")
public class CurrentDate extends HttpServlet {
    private static final long serialVersionUID = 1L;

    public CurrentDate() {
        super();
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

        response.setContentType("text/html;charset=UTF-8");

        PrintWriter out = response.getWriter();
        String title = "显示当前的日期和时间";
        Date dNow = new Date( );
        SimpleDateFormat ft =
            new SimpleDateFormat ("yyyy.MM.dd hh:mm:ss E a ");
        String docType = "<!DOCTYPE html> \n";
        out.println(docType +
            "<html>\n" +
            "<head><title>" + title + "</title></head>\n" +
            "<body bgcolor=\"#f0f0f0\">\n" +
            "<h1 align=\"center\">" + title + "</h1>\n" +
            "<h2 align=\"center\">" + ft.format(dNow) + "</h2>\n" +
            "</body></html>");
    }

}
```

再次编译上面的 Servlet，然后通过访问 `http://localhost:8080/TomcatTest/CurrentDate` 来调用该 Servlet。这将会产生如下的结果：

显示当前的日期和时间

2016.08.29 10:48:52 星期一 上午

简单的日期格式的格式代码

使用事件模式字符串来指定时间格式。在这种模式下，所有的 ASCII 字母被保留为模式字母，这些字母定义如下：

字符	描述	实例
G	Era 指示器	AD
y	四位数表示的年	2001
M	一年中的月	July 或 07
d	一月中的第几天	10
h	带有 A.M./P.M. 的小时（1~12）	12
H	一天中的第几小时（0~23）	22
m	一小时中的第几分	30
s	一分中的第几秒	55
S	毫秒	234
E	一周中的星期几	Tuesday
D	一年中的第几天	360
F	所在的周是这个月的第几周	2 (second Wed. in July)
w	一年中的第几周	40
W	一月中的第几周	1
a	A.M./P.M. 标记	PM
k	一天中的第几小时（1~24）	24
K	带有 A.M./P.M. 的小时（0~11）	10

z	时区	Eastern Standard Time
'	Escape for text	Delimiter
"	单引号	`

如需查看可用的处理日期方法的完整列表，您可以参考标准的 Java 文档。

← Servlet 文件上传

Servlet 网页重定向 →

✎ 点我分享笔记