SERVLET - HANDLING DATE

One of the most important advantages of using Servlet is that you can use most of the methods available in core Java. This tutorial would take you through Java provided **Date** class which is available in **java.util** package, this class encapsulates the current date and time.

The Date class supports two constructors. The first constructor initializes the object with the current date and time.

```
Date( )
```

The following constructor accepts one argument that equals the number of milliseconds that have elapsed since midnight, January 1, 1970

```
Date(long millisec)
```

Once you have a Date object available, you can call any of the following support methods to play with dates:

SN	Methods with Description
1	**boolean after***Datedate* Returns true if the invoking Date object contains a date that is later than the one specified by date, otherwise, it returns false.
2	**boolean before***Datedate* Returns true if the invoking Date object contains a date that is earlier than the one specified by date, otherwise, it returns false.
3	**Object clone** Duplicates the invoking Date object.
4	**int compareTo***Datedate* Compares the value of the invoking object with that of date. Returns 0 if the values are equal. Returns a negative value if the invoking object is earlier than date. Returns a positive value if the invoking object is later than date.
5	**int compareTo***Objectobj* Operates identically to compareTo*Date* if obj is of class Date. Otherwise, it throws a ClassCastException.
6	**boolean equals***Objectdate* Returns true if the invoking Date object contains the same time and date as the one specified by date, otherwise, it returns false.

7

long getTime

Returns the number of milliseconds that have elapsed since January 1, 1970.

8

int hashCode

Returns a hash code for the invoking object.

9

void setTime*longtime*

Sets the time and date as specified by time, which represents an elapsed time in milliseconds from midnight, January 1, 1970.

10

String toString

Converts the invoking Date object into a string and returns the result.

Getting Current Date & Time

This is very easy to get current date and time in Java Servlet. You can use a simple Date object with *toString* method to print current date and time as follows:

```
// Import required java libraries
import java.io.*;
import java.util.Date;
import javax.servlet.*;
import javax.servlet.http.*;

// Extend HttpServlet class
public class CurrentDate extends HttpServlet {

  public void doGet(HttpServletRequest request,
                    HttpServletResponse response)
            throws ServletException, IOException
  {
      // Set response content type
      response.setContentType("text/html");

      PrintWriter out = response.getWriter();
      String title = "Display Current Date & Time";
      Date date = new Date();
      String docType =
      "<!doctype html public \"-//w3c//dtd html 4.0 " +
      "transitional//en\">\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>");
  }
}
```

Now let us compile above servlet and create appropriate entries in web.xml and then call this servlet using URL http://localhost:8080/CurrentDate. This would produce following result:

DISPLAY CURRENT DATE & TIME

```
Mon Jun 21 21:46:49 GMT+04:00 2010
```

Try to refersh URL http://localhost:8080/CurrentDate and you would find difference in seconds everytime you would refresh.

Date Comparison:

As I mentioned above you can use all the available Java methods in your Servlet. In case you need to compare two dates, following are the methods:

- You can use getTime to obtain the number of milliseconds that have elapsed since midnight, January 1, 1970, for both objects and then compare these two values.
- You can use the methods before, after, and equals. Because the 12th of the month comes before the 18th, for example, new Date99, 2, 12.before*newDate*(99, 2, 18) returns true.
- You can use the compareTo method, which is defined by the Comparable interface and implemented by Date.

Date Formatting using SimpleDateFormat:

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

Let us modify above example as follows:

```
// Import required java libraries
import java.io.*;
import java.text.*;
import java.util.Date;
import javax.servlet.*;
import javax.servlet.http.*;

// Extend HttpServlet class
public class CurrentDate extends HttpServlet {

  public void doGet(HttpServletRequest request,
                    HttpServletResponse response)
            throws ServletException, IOException
  {
      // Set response content type
      response.setContentType("text/html");

      PrintWriter out = response.getWriter();
      String title = "Display Current Date & Time";
      Date dNow = new Date( );
      SimpleDateFormat ft =
      new SimpleDateFormat ("E yyyy.MM.dd 'at' hh:mm:ss a zzz");
      String docType =
      "<!doctype html public \"-//w3c//dtd html 4.0 " +
      "transitional//en\">\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>");
  }
}
```

Compile above servlet once again and then call this servlet using URL

http://localhost:8080/CurrentDate. This would produce following result:

```
DISPLAY CURRENT DATE & TIME

Mon 2010.06.21 at 10:06:44 PM GMT+04:00
```

Simple DateFormat format codes:

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:

Character	Description	Example
G	Era designator	AD
y	Year in four digits	2001
M	Month in year	July or 07
d	Day in month	10
h	Hour in A.M./P.M. 1 12	12
H	Hour in day 0 23	22
m	Minute in hour	30
s	Second in minute	55
S	Millisecond	234
E	Day in week	Tuesday
D	Day in year	360
F	Day of week in month	2 *secondWed. inJuly*
w	Week in year	40
W	Week in month	1
a	A.M./P.M. marker	PM
k	Hour in day 1 24	24
K	Hour in A.M./P.M. 0 11	10
z	Time zone	Eastern Standard Time
'	Escape for text	Delimiter
"	Single quote	`

For a complete list of constant available methods to manipulate date, you can refer to standard Java documentation.