Displaying date and time in Java ME

This code snippet demonstrates different ways to display date and time in Java ME.

Overview

Java ME contains time and date in the `Date` class. There are different ways to display date and time:

1. The main way is to use the DateField UI control (see the `outputDateUsingDateField` method in this snippet). This control gets an instance of the Date class and displays its date and time in a formatted way, also allowing the changing of particular parts of this date.
2. The other way is to use the `Date.toString()` method. This method returns a textual representation of the date in string form, which can be used to display it or for other purposes. See the `outputDateUsingToString` method in this snippet.
3. If custom formatting of the date string is needed, the `Calendar` class can be used. For example, the `outputDateUsingCalendar` method of this snippet constructs a textual representation of the specified date, using the Calendar class to get particular parts of the date and using the StringBuffer class to construct a string from these parts of the date. It is not possible to create a textual representation of the date from a pattern string (such as "dd/mm/yyyy") in J2ME, but it can be easily implemented using the Calendar class.

Source file: DateAndTimeMidlet.java

```
    /**
     * Sets up the main form.
     */
    private void setupMainForm() {
        mainForm = new Form("Date and time");

        // Get current date and output it by several ways.
        Date date = new Date();

        outputDateUsingDateField(date);
        outputDateUsingToString(date);
        outputDateUsingCalendar(date);

        mainForm.addCommand(EXIT_COMMAND);
        mainForm.setCommandListener(this);
    }

    /**
     * Shows specified date using DateField UI control.
     * @param date - date for showing
     */
    private void outputDateUsingDateField(Date date) {
        DateField field = new DateField("Using DateField",
            DateField.DATE_TIME);
        field.setDate(date);
        mainForm.append(field);
    }

    /**
     * Converts specified date to string using Date.toString() method and
     * shows it using StringItem.
     * @param date
     */
    private void outputDateUsingToString(Date date) {
        StringItem item = new StringItem("Using .toString()", date.toString());
```


(C) Copyright Nokia 2014. All rights reserved.

```
        mainForm.append(item);
    }

    /**
     * Constructs textual representation of specified date using Calendar class
     * for getting particular parts of date and using StringBuffer class
     * to construct string from these parts of date.
     * @param date
     */
    private void outputDateUsingCalendar(Date date) {
        Calendar calendar = Calendar.getInstance();
        calendar.setTime(date);
        StringBuffer sb = new StringBuffer();

        int day = calendar.get(Calendar.DAY_OF_MONTH);
        sb.append(numberToString(day));
        sb.append("-");
        int month = calendar.get(Calendar.MONTH) + 1;
        sb.append(numberToString(month));
        sb.append("-");
        sb.append(calendar.get(Calendar.YEAR));

        StringItem item = new StringItem("Using Calendar", sb.toString());
        mainForm.append(item);
    }

    /**
     * Utility method. Converts number to string and adds '0' before it if this
     * number is less than 10.
     * @param value number to convert.
     * @return textual representation of value.
     */
    private String numberToString(int value) {
        String valStr = Integer.toString(value);
        return (value < 10) ? "0" + valStr: valStr;
    }
```

Postconditions

After launching the snippet - several controls are placed on the form of this snippet demonstrating different ways to create a textual representation of date and time and to display it for the user.

Supplementary material

This code snippet is part of the stub concept, which means that it has been patched on top of a template application in order to be more useful for developers. The version of the Java ME stub application used as a template in this snippet is v1.1.

- The patched, executable application that can be used to test the features described in this snippet is available for download at Media:DisplayingDateAndTime.zip.
- You can view all the changes that are required to implement the above-mentioned features. The changes are provided in unified diff and colour-coded diff (HTML) formats in Media:DisplayingDateAndTime.diff.zip.
- For general information on applying the patch, see Using Diffs.
- For unpatched stub applications, see Example app stubs with logging framework.

Version Hint

(C) Copyright Nokia 2014. All rights reserved.

Windows Phone: [[Category:Windows Phone]]
[[Category:Windows Phone 7.5]]
[[Category:Windows Phone 8]]

Nokia Asha: [[Category:Nokia Asha]]
[[Category:Nokia Asha Platform 1.0]]

Series 40: [[Category:Series 40]]
[[Category:Series 40 1st Edition]] [[Category:Series 40 2nd Edition]]
[[Category:Series 40 3rd Edition (initial release)]] [[Category:Series 40 3rd Edition FP1]] [[Category:Series 40 3rd Edition FP2]]
[[Category:Series 40 5th Edition (initial release)]] [[Category:Series 40 5th Edition FP1]]
[[Category:Series 40 6th Edition (initial release)]] [[Category:Series 40 6th Edition FP1]] [[Category:Series 40 Developer Platform 1.0]] [[Category:Series 40 Developer Platform 1.1]] [[Category:Series 40 Developer Platform 2.0]]

Symbian: [[Category:Symbian]]
[[Category:S60 1st Edition]] [[Category:S60 2nd Edition (initial release)]] [[Category:S60 2nd Edition FP1]] [[Category:S60 2nd Edition FP2]] [[Category:S60 2nd Edition FP3]]
[[Category:S60 3rd Edition (initial release)]] [[Category:S60 3rd Edition FP1]] [[Category:S60 3rd Edition FP2]]
[[Category:S60 5th Edition]]
[[Category:Symbian^3]] [[Category:Symbian Anna]] [[Category:Nokia Belle]]

(C) Copyright Nokia 2014. All rights reserved.