Practices for Lesson 14

Practices Overview

In these practices, you create a date application that is similar to the example used in the lesson. For each practice, a NetBeans project is provided for you. Complete the project as indicated in the instructions.

Practice 14-1: Summary Level: Creating a Localized Date Application

Overview

In this practice, you create a text-based application that displays dates and times in a number of different ways. Create the resource bundles to localize the application for French, Simplified Chinese, and Russian.

Assumptions

You have attended the lecture for this lesson. You have access to the JDK7 API documentation.

Summary

Create a simple text-based date application that displays the following date information for today:

- Default date
- Long date
- Short date
- Full date
- Full time
- Day of the week
- And a custom day and time that displays: day of the week, long date, era, time, and time zone

Localize the application so that it displays this information in Simplified Chinese and Russian. The user should be able to switch between the languages.

The application output in English is shown here.

```
=== Date App ===
Default Date is: Aug 1, 2011
Long Date is: August 1, 2011
Short Date is: 8/1/11
Full Date is: Monday, August 1, 2011
Full Time is: 10:13:56 AM MDT
Day of week is: Monday
My custom day and time is: Monday August 1, 2011 AD 10:13:56
Mountain Daylight Time

--- Choose Language Option ---
1. Set to English
2. Set to French
3. Set to Chinese
4. Set to Russian
q. Enter q to quit
Enter a command:
```

Tasks

Open the `Localized-Practice01` project in NetBeans and make the following changes:

1. Edit the `DateApplication.java` file.
2. Create a message bundle for Russian and Simplified Chinese.
   - The translated text for the menus can be found in the `MessagesText.txt` file in the `practices` directory.
3. Add code to display the specified date formats (indicated with comments) and localized text.
4. Add code to change the `Locale` based on the user input.
5. Run the `DateApplication.java` file and verify that it operates as described.

Practice 14-1: Detailed Level: Creating a Localized Date Application

Overview

In this practice, you create a text-based application that displays dates and times in a number of different ways. Create the resource bundles to localize the application for French, Simplified Chinese, and Russian.

Assumptions

You have attended the lecture for this lesson. You have access to the JDK7 API documentation.

Summary

Create a simple text-based date application that displays the following date information for today:

- Default date
- Long date
- Short date
- Full date
- Full time
- Day of the week
- And a custom day and time that displays: day of the week, long date, era, time, and time zone

Localize the application so that it displays this information in Simplified Chinese and Russian. The user should be able to switch between languages.

The application output in English is shown here.

```
=== Date App ===
Default Date is: Aug 1, 2011
Long Date is: August 1, 2011
Short Date is: 8/1/11
Full Date is: Monday, August 1, 2011
Full Time is: 10:13:56 AM MDT
Day of week is: Monday
My custom day and time is: Monday August 1, 2011 AD 10:13:56
Mountain Daylight Time

--- Choose Language Option ---
1. Set to English
2. Set to French
3. Set to Chinese
4. Set to Russian
q. Enter q to quit
Enter a command:
```

Tasks

Open the `Localized-Practice01` project in NetBeans and make the following changes:

1. Edit the `DateApplication.java` file.
2. Open the `MessagesText.txt` file found in the `practices` directory for this practice in a text editor.
3. Create a message bundle file for Russian text named `MessagesBundle_ru_RU.properties`.
   - Right-click the project and select New > Other > Other > Properties File.
   - Click Next.
   - Enter `MessagesBundle_ru_RU` in the File Name field.
   - Click Browse.
   - Select the `src` directory.
   - Click Select Folder.
   - Click Finish.
   - Paste the localized Russian text into the file and save it.
4. Create a message bundle file for Simplified Chinese text named `MessagesBundle_zh_CN.properties`.
   - Right-click the project and select New > Other > Other > Properties File.
   - Click Next.
   - Enter `MessagesBundle_zh_CN` in the File Name field.
   - Click Finish.
   - Paste the localized Simplified Chinese text into the file and save it.
5. Update the code that sets the locale based on user input.

```
    public void setEnglish(){
        currentLocale = Locale.US;
        messages = ResourceBundle.getBundle("MessagesBundle",
currentLocale);
    }

    public void setFrench(){
        currentLocale = Locale.FRANCE;
        messages = ResourceBundle.getBundle("MessagesBundle",
currentLocale);
    }

    public void setChinese(){
        currentLocale = Locale.SIMPLIFIED_CHINESE;
```

```
        messages = ResourceBundle.getBundle("MessagesBundle",
currentLocale);
    }

    public void setRussian(){
        currentLocale = ruLocale;
        this.messages =
ResourceBundle.getBundle("MessagesBundle", currentLocale);
    }
```

6. Add the code that displays the date information to the `printMenu` method.

```
        df = DateFormat.getDateInstance(DateFormat.DEFAULT,
currentLocale);
        pw.println(messages.getString("date1") + " " +
df.format(today));
        df = DateFormat.getDateInstance(DateFormat.LONG,
currentLocale);
        pw.println(messages.getString("date2") + " " +
df.format(today));
        df = DateFormat.getDateInstance(DateFormat.SHORT,
currentLocale);
        pw.println(messages.getString("date3") + " " +
df.format(today));
        df = DateFormat.getDateInstance(DateFormat.FULL,
currentLocale);
        pw.println(messages.getString("date4") + " " +
df.format(today));
        df = DateFormat.getTimeInstance(DateFormat.FULL,
currentLocale);
        pw.println(messages.getString("date5") + " " +
df.format(today));
        sdf = new SimpleDateFormat("EEEE", currentLocale);
        pw.println(messages.getString("date6")  + " " +
sdf.format(today));
        sdf = new SimpleDateFormat("EEEE MMMM d, y G kk:mm:ss
zzzz", currentLocale);
        pw.println(messages.getString("date7") + " " +
sdf.format(today));
```

7. Run the `DateApplication.java` file and verify that it operates as described.

Practice 14-2: Summary Level: Localizing a JDBC Application (Optional)

Overview

In this practice, you localize the JDBC application that you created in the practices for the "Building Database Applications with JDBC" lesson.

Assumptions

You have attended the lecture for this lesson. You have completed the practices for the "Building Database Applications with JDBC" lesson.

Summary

Localize the JDBC application from the previous lesson. Identify any object that displays menu or object information and change them so that localized messages are displayed instead of static text.

Localize the application so that it displays this information in English, French, and Russian. The user should be able to switch between languages.

Tasks

You have a couple of project choices in this lab. First you can use the project files from Lesson 13, Practice 2, "Using the Data Access Object Pattern," and just continue with that project. Alternatively, open the `Practice02` project for this lesson. Perform the following steps:

1. Open the `EmployeeTestInteractive.java` file. Examine the source code and determine which messages printed to the console should be converted into resource bundles. Notice that not all text output is included in this class file.

   Note: You do not have to include error messages in the bundle. Only prompt and informational messages should be included.

2. A slight change to the user interface is required.

   Currently the main interface looks like this:

   ```
   [C]reate | [R]ead | [U]pdate | [D]elete | [L]ist | [Q]uit:
   ```

   Changing the user interface to the following makes it easier to translate just the words in the menu.

   ```
   [C] - Create | [R] - Read | [U] - Update | [D] - Delete | [L] -
   List | [S] – Set Language | [Q] - Quit:
   ```

   This separates the single character commands from the words. For the solution only the words were translated. You could, of course, translate both. Notice a new option has been added to set the language.

3. Create a message bundle for English, French, and Russian.

   - The translated text for the menus can be found in the `MessagesText02.txt` file in the `practices` directory.

4. Add a `ResourceBundle` object to any object that displays menu-related information. Replace the static text with a call to the resource bundle and get the appropriate string message.

5. Examine all the date-related source code. Make sure that date information will print in the appropriate localized format.
6. When you have finished, run `EmployeeTestInteractive.java` and make sure that all the menus have been localized.
7. Additional improvements you could make:
   - Localize all the error messages in the application.
   - Localize the single character options for the main menu.