

Using the Resource String Wizard

NOTE: This supplemental material may be used at the end of Chapter 14, "Internationalization," in Introduction to Java Programming with JBuilder 4. The Resource String wizard is available in JBuilder Professional and JBuilder Enterprise.

JBuilder provides a resource string wizard to help you develop internationalized applications and applets. First you develop the project without worrying about locale. Once the project is done, use the resource string wizard to create resource bundles to move hard-coded strings to Resource Bundles. JBuilder automatically generates the code for creating a resource bundle object and for retrieving objects from the resource bundle.

To demonstrate using the resource string wizard, we will modify the NumberFormatDemo program in Example 14.3 so that it displays messages, title, and button labels in English, Chinese, and French, as shown in Figures 1. You will learn how to use the Resource string wizard that automatically generates the code for handling resources.

Number Formatting Demo

Choisir la localite

French (France)

inscrire le taux d'interet, les annees, et le montant du pret

le taux d'interet	6.7	6,70%
annees	15	15
Le montant du pret	107000	107 000,00 F

paiement

versement mensuel	943,89 F
reglement total	169 900,10 F

Calculer l'hypothèque

Number Formatting Demo

選擇國家

Chinese (China)

輸入利率, 年限, 貸款總額

利率	6.7	6.70%
年限	15	15
貸款額度	107000	¥107,000.00

付息

月付	¥943.89
總額	¥169,900.10

計算貸款利息

Figure 1

The program displays the strings in French or in Chinese.

To see the development in action, follow the steps below.

1. To keep `NumberFormatDemo` intact, select `NumberFormatDemo.java` in the project pane, and choose File, Save As to save it as `ResourceBundleDemo.java` in the same folder. Replace all `NumberFormatDemo` with

ResourceBundleDemo in ResourceBundleDemo.java and compile the program. The program should run fine.

2. With ResourceBundleDemo.java selected in the project pane, choose Wizards, Resource Strings to display Step 1 of 2 of the resource string wizard, as shown in Figure 2.
3. Check "Generate keys from the string value" and click New to display the Create ResourceBundle dialog box, as shown in Figure 3. You can choose either ListResourceBundle or PropertyResourceBundle. For simplicity, since all locale-specific resource are strings in this applet, choose PropertyResourceBundle. Click OK to close the dialog box. You will see the Resource string wizard Step 2 of 2, as shown in Figure 4.

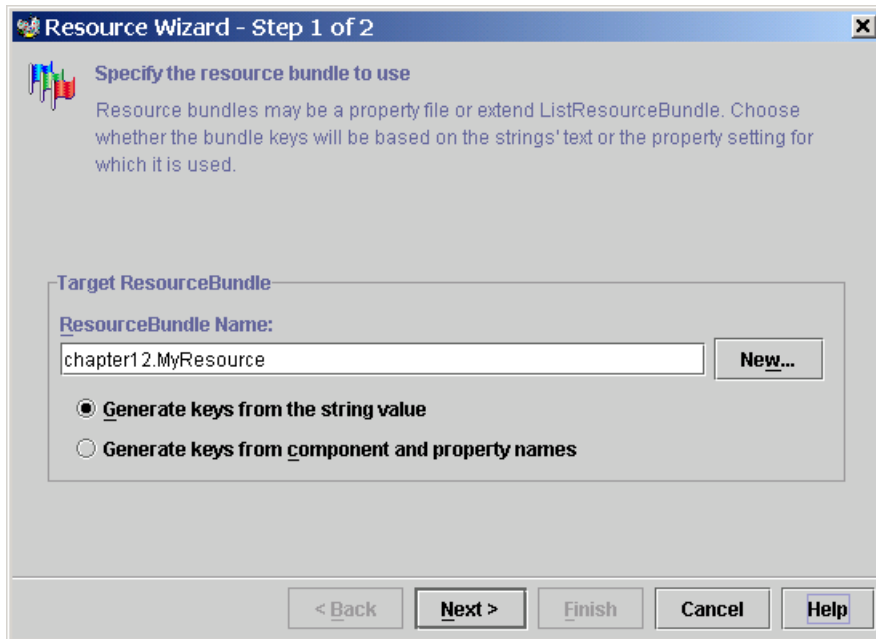


Figure 2

The Resource string wizard automatically generates the resource bundle and the code for retrieving resource.

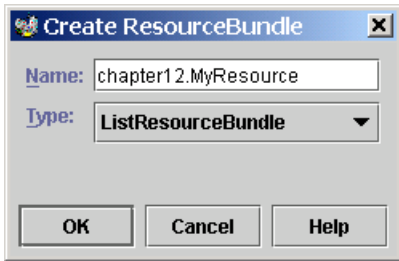


Figure 3

You can choose ListResourceBundle or PropertyResourceBundle in the Create ResourceBundle dialog box.

3. Click Next to display Step 2 of 2 of the Resource string wizard, as shown in Figure 14.11. Press Finish to move the strings currently displayed in the Key field to a ResourceBundle.

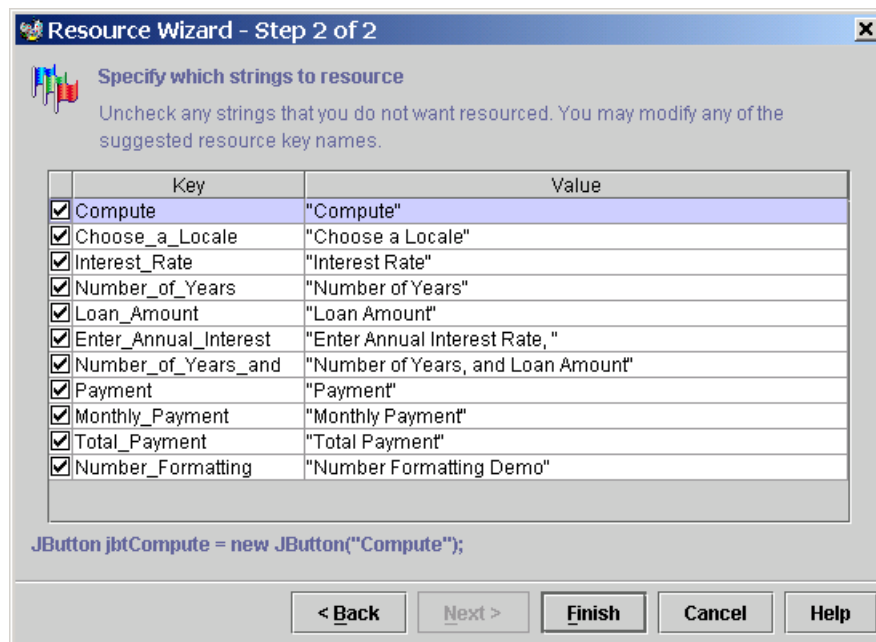


Figure 5

You can specify which strings to resource in Step 2 of the Resource string wizard.

4. JBuilder automatically generated the code for creating a resource bundle object and for retrieving resources. To update the locale-sensitive strings, create a method named updateString() and invoke it in

the handler for selecting locales. When locale changes, this method is invoked to update strings.

The source code for ResourceBundleDemo.java is shown below:

```
// ResourceBundleDemo.java: Demonstrate resource bundle
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.border.*;
import java.util.*;
import java.text.*;

public class ResourceBundleDemo extends JApplet
    implements ItemListener, ActionListener
{
    // Combo box for selecting available locales
    JComboBox jcbLocale = new JComboBox();
    ResourceBundle res = ResourceBundle.getBundle("MyResource");

    // Create labels
    JLabel jlblInterestRate =
        new JLabel(res.getString("Annual Interest Rate"));
    JLabel jlblNumOfYears =
        new JLabel(res.getString("Number Of Years"));
    JLabel jlblLoanAmount = new JLabel(res.getString("Loan Amount"));
    JLabel jlblMonthlyPayment =
        new JLabel(res.getString("Monthly Payment"));
    JLabel jlblTotalPayment = new JLabel(res.getString("Total Payment"));

    // Create titled borders
    TitledBorder comboBoxTitle =
        new TitledBorder(res.getString("Choose a Locale"));
    TitledBorder inputTitle = new TitledBorder
        (res.getString("Enter Interest Rate"));
    TitledBorder paymentTitle =
        new TitledBorder(res.getString("Payment"));

    // Text fields for interest rate, year, loan amount,
    JTextField jtfInterestRate = new JTextField(10);
    JTextField jtfNumOfYears = new JTextField(10);
    JTextField jtfLoanAmount = new JTextField(10);
    JTextField jtfFormattedInterestRate = new JTextField(10);
    JTextField jtfFormattedNumOfYears = new JTextField(10);
    JTextField jtfFormattedLoanAmount = new JTextField(10);

    // Text fields for monthly payment and total payment
    JTextField jtfTotalPayment = new JTextField();
    JTextField jtfMonthlyPayment = new JTextField();

    // Compute Mortgage button
    JButton jbtCompute = new JButton(res.getString("Compute"));

    // Current locale
    Locale locale = Locale.getDefault();

    // Declare locales to store available locales
    Locale locales[] = Calendar.getAvailableLocales();

    // Initialize the combo box
    public void initializeComboBox()
    {
        // Add locale names to the combo box
        for (int i=0; i<locales.length; i++)
            jcbLocale.addItem(locales[i].getDisplayName());
    }

    // Initialize the applet
    public void init()
    {
        // Load native fonts. Uncomment the following two statements.
        // if native fonts such as Chinese fonts are used
        // GraphicsEnvironment ge =
        //     GraphicsEnvironment.getLocalGraphicsEnvironment();
        // ge.getAllFonts();

        // Panel pl to hold the combo box for selecting locales
    }
}
```

```

    JPanel p1 = new JPanel();
    p1.setLayout(new FlowLayout());
    p1.add(jcboLocale);
    initializeComboBox();
    p1.setBorder(comboBoxTitle);

    // Panel p2 to hold the input for annual interest rate,
    // number of years and loan amount
    JPanel p2 = new JPanel();
    p2.setLayout(new GridLayout(3, 3));
    p2.add(jlblInterestRate);
    p2.add(jtfInterestRate);
    p2.add(jtfFormattedInterestRate);
    p2.add(jlblNumOfYears);
    p2.add(jtfNumOfYears);
    p2.add(jtfFormattedNumOfYears);
    p2.add(jlblLoanAmount);
    p2.add(jtfLoanAmount);
    p2.add(jtfFormattedLoanAmount);
    p2.setBorder(inputTitle);

    // Panel p3 to hold the payment
    JPanel p3 = new JPanel();
    p3.setLayout(new GridLayout(2, 2));
    p3.setBorder(paymentTitle);
    p3.add(jlblMonthlyPayment);
    p3.add(jtfMonthlyPayment);
    p3.add(jlblTotalPayment);
    p3.add(jtfTotalPayment);

    // Set text field alignment
    jtfFormattedInterestRate.setHorizontalAlignment
        (JTextField.RIGHT);
    jtfFormattedNumOfYears.setHorizontalAlignment(JTextField.RIGHT);
    jtfFormattedLoanAmount.setHorizontalAlignment(JTextField.RIGHT);
    jtfTotalPayment.setHorizontalAlignment(JTextField.RIGHT);
    jtfMonthlyPayment.setHorizontalAlignment(JTextField.RIGHT);

    // Set editable false
    jtfFormattedInterestRate.setEditable(false);
    jtfFormattedNumOfYears.setEditable(false);
    jtfFormattedLoanAmount.setEditable(false);
    jtfTotalPayment.setEditable(false);
    jtfMonthlyPayment.setEditable(false);

    // Panel p4 to hold result payments and a button
    JPanel p4 = new JPanel();
    p4.setLayout(new BorderLayout());
    p4.add(p3, BorderLayout.CENTER);
    p4.add(jbtCompute, BorderLayout.SOUTH);

    // Place panels to the applet
    getContentPane().add(p1, BorderLayout.NORTH);
    getContentPane().add(p2, BorderLayout.CENTER);
    getContentPane().add(p4, BorderLayout.SOUTH);

    // Register listeners
    jcboLocale.addItemListener(this);
    jbtCompute.addActionListener(this);
}

// Main method
public static void main(String[] args)
{
    // Create an instance of the applet
    ResourceBundleDemo applet = new ResourceBundleDemo();

    // Create a frame with a resource string
    JFrame frame = new JFrame(
        applet.res.getString("Number Formatting"));

    // Add the applet instance to the frame
    frame.getContentPane().add(applet, BorderLayout.CENTER);

    // Invoke init() and start()
    applet.init();
    applet.start();

    // Display the frame
    frame.setSize(300, 300);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}

```

```

    frame.setVisible(true);
}

// Handle locale selection
public void itemStateChanged(ItemEvent e)
{
    if (e.getSource() == jcboLocale)
    {
        locale = locales[jcboLocale.getSelectedIndex()];
        updateStrings();
        computeMortgage();
    }
}

// Handle button action
public void actionPerformed(ActionEvent e)
{
    if (e.getSource() == jbtCompute)
        computeMortgage();
}

// Compute payments and display results locale-sensitive format
private void computeMortgage()
{
    // Retrieve input from user
    double loan = new Double(jtfLoanAmount.getText()).doubleValue();
    double interestRate =
        new Double(jtfInterestRate.getText()).doubleValue()/1200;
    int numOfYear = new Integer(jtfNumOfYears.getText()).intValue();

    // Calculate payments
    double monthlyPayment = loan*interestRate/
        (1-(Math.pow(1/(1+interestRate), numOfYear*12)));
    double totalPayment = monthlyPayment*numOfYear*12;

    // Get formatters
    NumberFormat percentFormatter =
        NumberFormat.getPercentInstance(locale);
    NumberFormat currencyForm =
        NumberFormat.getCurrencyInstance(locale);
    NumberFormat numberForm = NumberFormat.getNumberInstance(locale);
    percentFormatter.setMinimumFractionDigits(2);

    // Display formatted input
    jtfFormattedInterestRate.setText(
        percentFormatter.format(interestRate*12));
    jtfFormattedNumOfYears.setText(numberForm.format(numOfYear));
    jtfFormattedLoanAmount.setText(currencyForm.format(loan));

    // Display results in currency format
    jtfMonthlyPayment.setText(currencyForm.format(monthlyPayment));
    jtfTotalPayment.setText(currencyForm.format(totalPayment));
}

// Update resource strings
private void updateStrings()
{
    res = ResourceBundle.getBundle("MyResource", locale);
    jlblInterestRate.setText(res.getString("Annual Interest Rate"));
    jlblNumOfYears.setText(res.getString("Number Of Years"));
    jlblLoanAmount.setText(res.getString("Loan Amount"));
    jlblTotalPayment.setText(res.getString("Total Payment"));
    jlblMonthlyPayment.setText(res.getString("Monthly Payment"));
    jbtCompute.setText(res.getString("Compute"));
    comboBoxTitle.setTitle(res.getString("Choose a Locale"));
    inputTitle.setTitle(res.getString("Enter Interest Rate"));
    paymentTitle.setTitle(res.getString("Payment"));

    // Make sure the new labels are displayed
    repaint();
}
}

```

The resource bundle for the English language is given as follows:

```

#Thu May 06 07:48:10 EST 1999
Number Of Years=Years
Total Payment=French Total\ Payment

```

```

Enter Interest Rate=Enter\ Interest\ Rate,\ Years,\ and\ Loan\ Amount
Payment=Payment
Compute=Compute
Annual Interest Rate=Interest\ Rate
Number Formatting=Number\ Formatting\ Demo
Loan Amount=Loan\ Amount
Choose a Locale=Choose\ a\ Locale
Monthly Payment=Monthly\ Payment
=
=

```

The resource bundle for the Chinese language is given as follows:

```

#MyResource_zh.properties for Chinese language
Choose a Locale = \u9078\u64c7\u570b\u5bb6
Enter Interest Rate =
\u8f38\u5165\u5229\u7387,\u5e74\u9650,\u8cb8\u6b3e\u7e3d\u984d
Annual Interest Rate = \u5229\u7387
Number Of Years = \u5e74\u9650
Loan Amount = \u8cb8\u6b3e\u984d\u5ea6
Payment = \u4ed8\u606f
Monthly Payment = \u6708\u4ed8
Total Payment = \u7e3d\u984d
Compute = \u8a08\u7b97\u8cb8\u6b3e\u5229\u606f
=
=

```

The resource bundle for the French language is given as follows:

```

#MyResource_fr.properties for French language
#Thu May 06 07:48:10 EST 1999
Number Of Years=annees
Annual Interest Rate=le taux d'interet
Loan Amount=Le montant du pret
Enter Interest Rate=inscrire le taux d'interet, les annees, et le montant du pret
Payment=paiement
Compute=Calculer l'hypothèque
Number Formatting=demonstration du formatting des chiffres
Choose a Locale=Choisir la localite
Monthly Payment=versement mensuel
Total Payment=reglement total
=
=

```

Example Review

Property resource bundles are implemented as text files with a `.properties` extension, and are placed in the same location as the class files for the application or applet. `ListResourceBundles` are provided as Java source files. Because they are implemented as Java source code, new and modified `ListResourceBundles` need to be recompiled for deployment. With `PropertyResourceBundles`, there is no need for recompilation when translations are modified or added to the application. Nevertheless, `ListResourceBundles` provide considerably better performance than `PropertyResourceBundles`.

If the resource bundle is not found or a resource object is not found in the resource bundle, a `MissingResourceException` is raised. Since `MissingResourceException` is a subclass of

RuntimeException, you do not need to catch the exception explicitly in the code.

This example is the same as Example 14.3 except that the program contains the code for handling resource strings. The updateString method is responsible for displaying the locale-sensitive strings. This method is invoked when a new locale is selected in the combo box. Since the variable res of the ResourceBundle class is an instance variable in ResourceBundleDemo, it cannot be directly used in the main method, because the main method is static. To fix the problem, create applet as an instance of ResourceBundleDemo and you will then be able to reference res using applet.res.