Localising resource bundles in Java ME

Overview

This code snippet is one of the series of snippets that demonstrate how to implement multi-language applications in Java ME. There are several techniques for localizing a MIDlet:

- Using application attributes
- Using text files
- Using resource bundles

This snippet describes the third technique; using resource bundles.

Simply put, this technique uses Java classes for storing the localized resources. Every such class extends the `ResourceBundle` class, which handles searching the specified resource for the particular locale. Each bundle (a class that extends the `ResourceBundle` class) also has a parent bundle from which a resource is searched in case it is not found from the bundle itself. This mechanism can be referred to as the parent chain of bundles.

Even though the localized resources are strings in this snippet, this technique applies to all kinds of resources, not just strings - unlike the other two techniques. Another benefit of using resource bundles is that there is no need for file handling. On the other hand, resource bundles may be difficult to utilize for non-technical translators, since localizable resources are mixed with code. Resource bundles also require more storage space and memory on the device than, for example, text files.

Source: ResourceBundle.java

```
package com.nokia.example.l10nMIDlet;

import java.util.Vector;

public abstract class ResourceBundle {
    protected ResourceBundle parent;

    /**
     * Returns a resource bundle using the specified base name and target
     * locale.
     */
    public static ResourceBundle getBundle(String baseName, Locale targetLocale) {
        ResourceBundle parentBundle = null;
        ResourceBundle bundle = null;
        // Get a list of candidate locales for which resource bundles are
        // searched
        Vector candidateLocales = getCandidateLocales(targetLocale);
        // Go through every candidate locale and try to instantiate a
        // ResourceBundle using the base name and the candidate locale
        for (int i = candidateLocales.size() - 1; i >= 0; i--) {
            Locale locale = (Locale)candidateLocales.elementAt(i);
            // Bundle name consists of the base name plus an underscore and
            // locale if there is a locale. Otherwise, it will consist only of
            // the base name.
            String bundleName = baseName +
                ((locale.toString().equals("")) ? "" : ("_" + locale));
            try {
                // Try to instantiate a resource bundle using the name
                // constructed above
                Class bundleClass = Class.forName("com.nokia.example.l10nMIDlet." + bundleName);
                bundle = (ResourceBundle)bundleClass.newInstance();
                // Set the parent bundle for this bundle. For the base bundle
                // (the one with the root locale, Locale.ROOT), the parent is
                // null.
                bundle.setParent(parentBundle);
                parentBundle = bundle;
            } catch (Exception ex) {
                // No need to do anything, just continue to the next bundle
            }
        }
        // If bundle is null even here, no resource bundle could be found.
        // This is an error situation.
        if (bundle == null) {
            throw new RuntimeException(
                "Can't find resource bundle for base name " + baseName + ".");
        }
        return bundle;
    }

    /**
     * Returns a list of Locales as candidate locales used in bundle
     * instantiation.
     * @param locale the locale for which a resource bundle is desired
     */
    private static Vector getCandidateLocales(Locale locale) {
        String language = locale.getLanguage();
        String country = locale.getCountry();

        Vector locales = new Vector(3);
        if (!country.equals("")) {
            locales.addElement(locale);
        }
        if (!language.equals("")) {
            locales.addElement(
                (locales.size() == 0) ? locale : new Locale(language, ""));
        }
        locales.addElement(Locale.ROOT);
        return locales;
    }
```


(C) Copyright Nokia 2013. All rights reserved.

```
    protected void setParent(ResourceBundle parent) {
        this.parent = parent;
    }

    public final String getString(String key) {
        String string = handleGetString(key);
        if (string == null) {
            if (parent != null) {
                string = parent.getString(key);
            }
            if (string == null) {
                throw new RuntimeException("Can't find resource for bundle " +
                    this.getClass().getName() + " and key " + key + ".");
            }
        }
        return string;
    }

    /**
     * Gets a string for the given key from this resource bundle. Returns null
     * if this resource bundle does not contain a string for the given key.
     */
    protected abstract String handleGetString(String key);
}
```

Source: Resources.java

```
package com.nokia.example.l10nMIDlet;

import java.util.Hashtable;

/**
 * Default resource bundle (English / United States).
 */
public class Resources extends ResourceBundle {
    private Hashtable strings;

    public Resources() {
        strings = new Hashtable(30);

        strings.put("title", "Localization example");
        strings.put("exit", "Exit");
        strings.put("localeLbl", "Locale");
        strings.put("textLbl", "Text");
        strings.put("text", "Here's some text.");
    }

    public String handleGetString(String key) {
        return (String)strings.get(key);
    }
}
```

Source: Resources_fr.java

```
package com.nokia.example.l10nMIDlet;

import java.util.Hashtable;

/**
 * Resource bundle for French, no specific country.
 */
public class Resources_fr extends Resources {
    private Hashtable strings;

    public Resources_fr() {
        strings = new Hashtable(30);

        strings.put("exit", "Quitter");
        strings.put("textLbl", "Texte");
        strings.put("text", "Voici du texte.");
    }

    public String handleGetString(String key) {
        return (String)strings.get(key);
    }
}
```

Note: In the resource bundle above, the French title is omitted to demonstrate the parent chain of bundles when retrieving the resources. In other words, the title is searched from the superclass (`Resources`), because it cannot be found from the class above (`Resources_fr`). Note also that because of the parent chain mechanism, you don't need to supply a value if the parent bundle handles the same key with the same value. This is the case with the `localeLbl` key above.

Source: Locale.java

```
package com.nokia.example.l10nMIDlet;

public class Locale {
    public static final Locale ROOT = new Locale("", "");
    private String language;
    private String country;
    private static Locale defaultLocale;

    /**
     * Constructs a locale using the specified language and country.
     */
    public Locale(String language, String country) {
        this.language = language;
        this.country = country;
        defaultLocale = null;
    }

    /**
```


(C) Copyright Nokia 2013. All rights reserved.

```
     * Constructs a locale by parsing the language and country from the given
     * string.
     */
    public Locale(String locale) {
        defaultLocale = null;
        // Locale en_US will do if the given locale is invalid
        if (locale == null) {
            language = "en";
            country = "US";
            return;
        }
        language = locale;
        country = "";
        // Some devices separate locale components (language and country) from
        // each other by a hyphen instead of an underscore. Let's convert every
        // hyphen into an underscore for consistency.
        locale = locale.replace('-', '_');
        int separatorPos = locale.indexOf('_');
        if (separatorPos != -1) {
            language = locale.substring(0, separatorPos);
            locale = locale.substring(separatorPos + 1);
            separatorPos = locale.indexOf('_');
            if (separatorPos != -1) {
                country = locale.substring(0, separatorPos);
            } else {
                country = locale;
            }
        }
    }

    public String getLanguage() {
        return language;
    }

    public String getCountry() {
        return country;
    }

    public static Locale getDefault() {
        if (defaultLocale == null) {
            defaultLocale = new Locale(System.getProperty(
                "microedition.locale"));
        }
        return defaultLocale;
    }

    /**
     * Returns this locale as a string representation, with the language and
     * country separated by underscores.
     * Examples: "en", "de_CH", "_US"
     */
    public String toString() {
        if (language.equals("") && country.equals("")) {
            return "";
        }
        StringBuffer localeString = new StringBuffer("");
        if (!country.equals("")) {
            localeString.append(language).append("_").append(country);
        } else if (!language.equals("")) {
            localeString.append(language);
        }
        return localeString.toString();
    }
}
```

Source: L10nMIDlet.java

```
package com.nokia.example.l10nMIDlet;

import javax.microedition.lcdui.Command;
import javax.microedition.lcdui.CommandListener;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Displayable;
import javax.microedition.lcdui.Form;
import javax.microedition.lcdui.StringItem;
import javax.microedition.midlet.MIDlet;

public class L10nMIDlet extends MIDlet implements CommandListener {
    private Form mainForm;
    private Command exitCommand;
    private Locale locale;
    private ResourceBundle resources;

    /**
     * Constructor. Constructs the object and initializes displayables.
     */
    public L10nMIDlet() {
        // Obtain the system locale
        locale = Locale.getDefault();

        resources = ResourceBundle.getBundle("Resources", locale);

        initUI();
    }

    public void initUI() {
        // Initialize the form
        mainForm = new Form(resources.getString("title"));

        // Display the locale
        StringItem localeItem = new StringItem(resources.getString("localeLbl"),
            locale.toString());
        mainForm.append(localeItem);

        // Display a localized text string
        StringItem textItem = new StringItem(resources.getString("textLbl"),
            resources.getString("text"));
        mainForm.append(textItem);

        // Create the exit command
        exitCommand = new Command(resources.getString("exit"), Command.EXIT, 1);
        mainForm.addCommand(exitCommand);
        mainForm.setCommandListener(this);
    }
```


(C) Copyright Nokia 2013. All rights reserved.

```
    /**
     * From MIDlet.
     * Called when the MIDlet is started.
     */
    public void startApp() {
        // The initial display is the first form
        Display.getDisplay(this).setCurrent(mainForm);
    }

    /**
     * From MIDlet.
     * Called to signal the MIDlet to enter the Paused state.
     */
    public void pauseApp() {
        // No implementation required
    }

    /**
     * From MIDlet.
     * Called to signal the MIDlet to terminate.
     * @param unconditional whether the MIDlet has to be unconditionally
     * terminated
     */
    public void destroyApp(boolean unconditional) {
        // No implementation required
    }

    /**
     * From CommandListener.
     * Called by the system to indicate that a command has been invoked on a
     * particular displayable.
     * @param command the command that was invoked
     * @param displayable the displayable where the command was invoked
     */
    public void commandAction(Command command, Displayable displayable) {
        if (command == exitCommand) {
            // Exit the MIDlet
            notifyDestroyed();
        }
    }
}
```

Postconditions

A MIDlet is localized using resource bundles.

See also

- ResourceBundle class in the Java SE 6 API specification.
- Localising app attributes in Java ME
- Localising text files in Java ME

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.