Babbling With the New Bundle of Joy

BASIS' New Resource Bundle Editor

Many believe that removing the string literals displayed to users from application code is only necessary in applications that will be used in multiple languages. Although this is the common case, there is a major reason to remove the literals even when writing an application for a single language. Maintenance of these string literals becomes very easy when they are separated from the code; so easy in fact, that non-programmers can even change the text. Another benefit to separating strings from the code is that a single update of the string propagates the change throughout the entire application regardless of how many times it appears. If the need ever arises to internationalize the application, simply send the resource bundle off to a translator. In BBj®, a developer can separate the text literals from the code by using the BASIS resource bundle utility called the Resource Bundle Editor (RBE) that facilitates easy creation and maintenance of resource bundles. See **Figure 1**.

By Brian Hipple
Quality Assurance Supervisor

Overview

A resource bundle is simply a collection of one or more "property files" or ASCII files that contain key/value pairs much like a string table in BBj. The names of property files include the name of the resource bundle and, if necessary, locale information. For example, a HelloWorld application might have a HelloWorld.properties file that contains default values in English and a HelloWorld_de_DE.properties file for German. The RBE is a BBj application building block utility program that provides a very intuitive user interface and is a turnkey solution for creating, editing, and updating resource bundles and their associated translation property files.

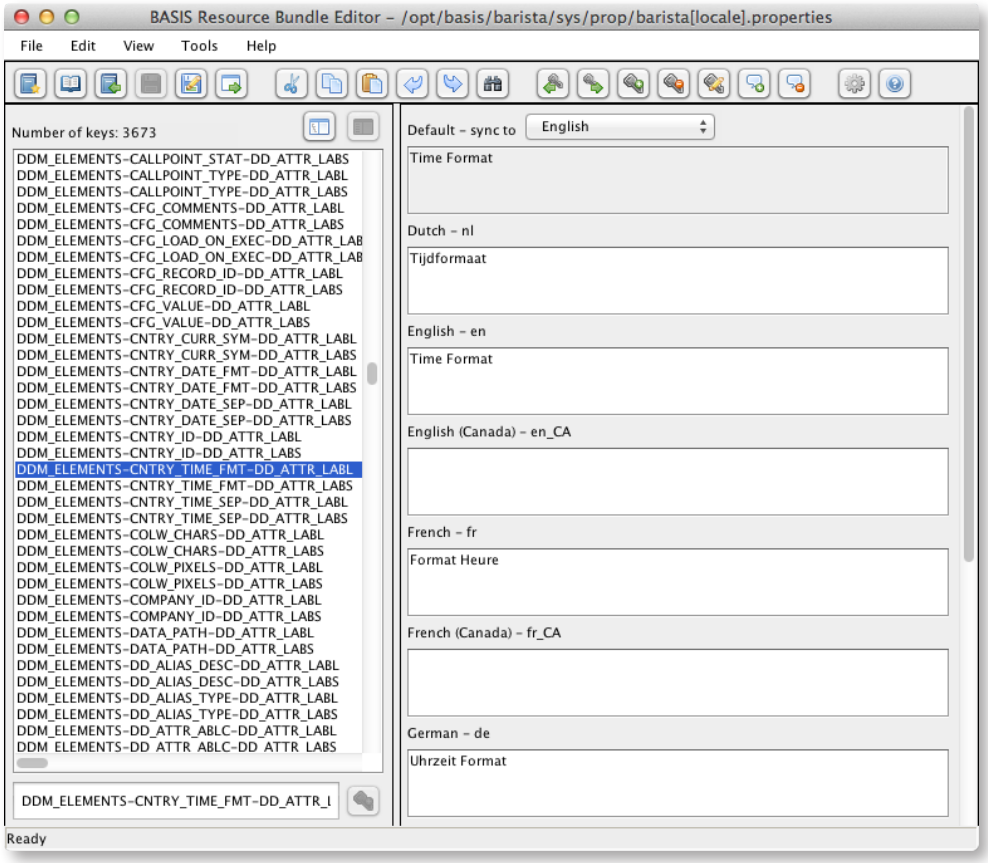

Figure 1. The Resource Bundle Editor displaying a few translations of a particular key

Conceiving

To create or edit a resource bundle, use the menu items or toolbar buttons inside the RBE. A most recently used list is also available from the menu to quickly reopen resource bundles. When creating a new resource bundle, a dialog box like the one shown in **Figure 2** prompts the user for the folder and base name of the resource bundle. A locale is the language/country/variant and is selectable from the list or entered manually. The locale en_US is US English rather than en_GB (Great Britain) English for United Kingdom or en_CA English for Canada.

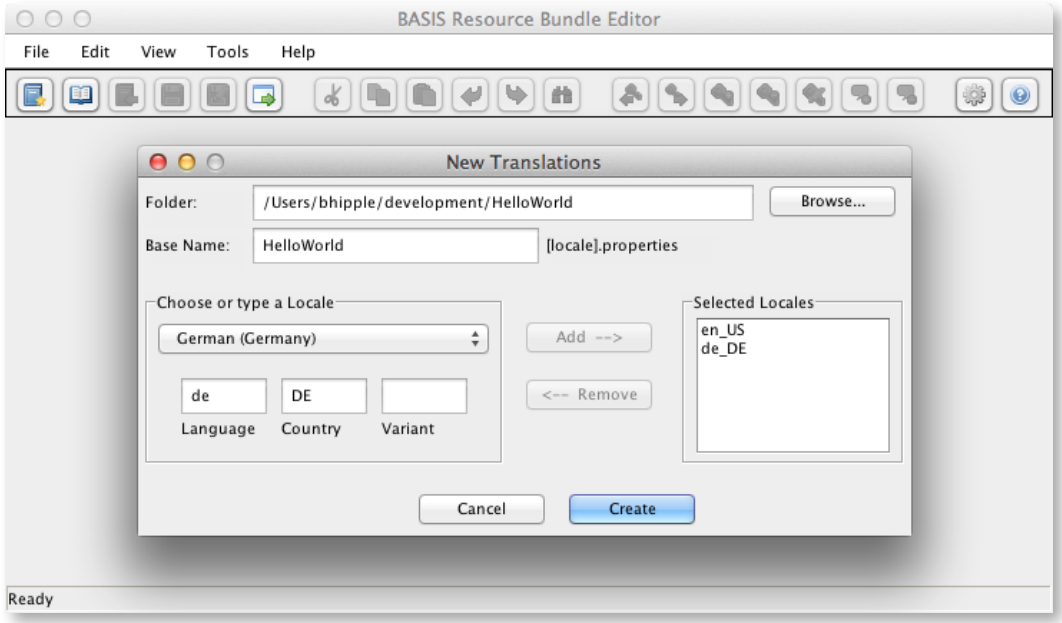

Figure 2. Define a new resource bundle

Reproducing

After creating or opening a resource bundle, users can easily add keys in several ways – the [Add Key] button, toolbar button, menu item; or by right mouse clicking and selecting the Add Menu option from the popup menu. A dialog appears in which the user can enter the new key name as shown in **Figure 3**.

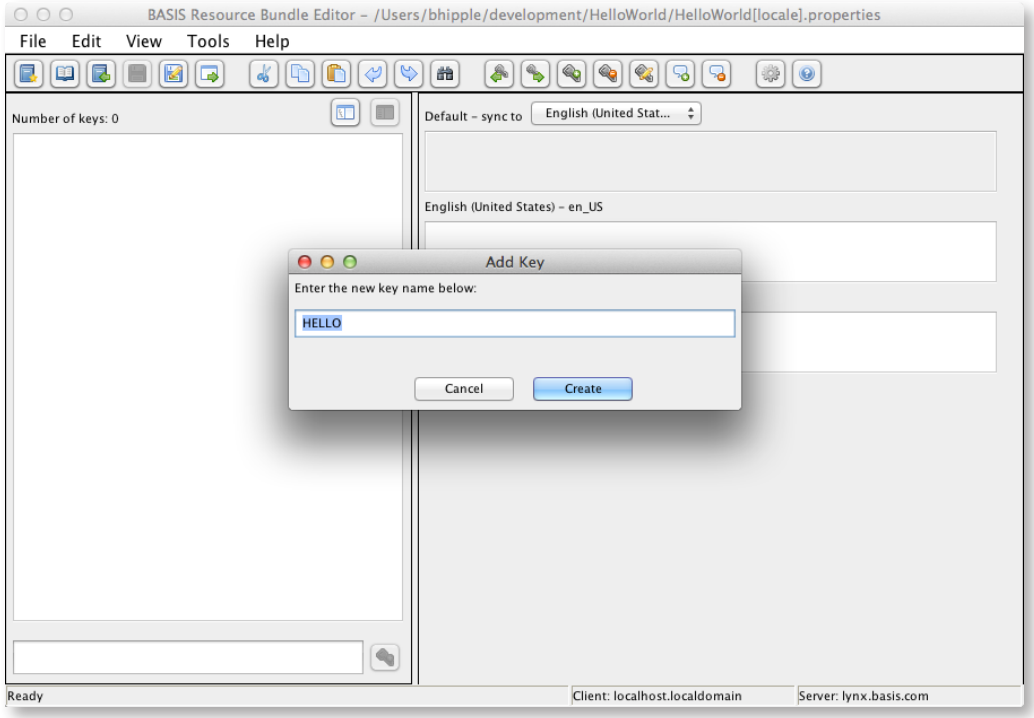

Figure 3. Add a new key to the bundle

Localizing

Next, specify the values for the default and any additional locales. The utility provides the option to sync to a default locale as shown in **Figure 4**.

Extra Capabilities

The keys display one of two views; in a straight list or, if there is a hierarchy in the key name, in a tree view. The utility offers a myriad of tool buttons and menu items to open/close/save resource bundles, copy/cut/paste values, undo and redo values, traverse keys, add/remove locales and keys, and set options.

An advanced "Find" dialog (see **Figure 5**) helps to find keys/values, which is a useful feature when working with very large resource bundles.

Propagating

After creating the resource bundle, the developer can access it from inside a BBj application in one of two ways; using the BASIS-provided BBTranslator utility or the Java ResourceBundle API. The BBJabber utility facilitates the creation of resource bundles from BBx program source and resources. This includes adding the necessary code to access the resource bundle via the BBTranslator utility instead of using string literals. Take a look at some additional *BASIS Advantage* articles – *Can Your App Speak to Your Customer?* and *Parlez-BUI Français?* – to learn more about this BASIS-supplied utility.

BBTranslator Utility

The BBTranslator utility offers several advantages. One benefit is that developers can use BBj code without learning another language. The BBTranslator includes object-

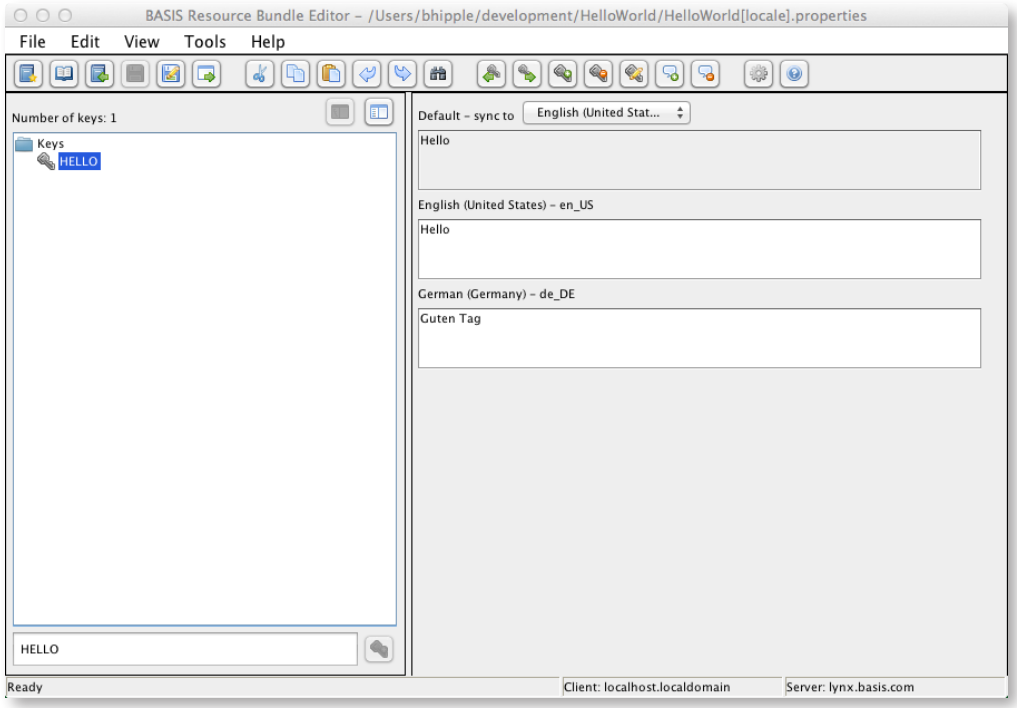

Figure 4. Set values for the new key

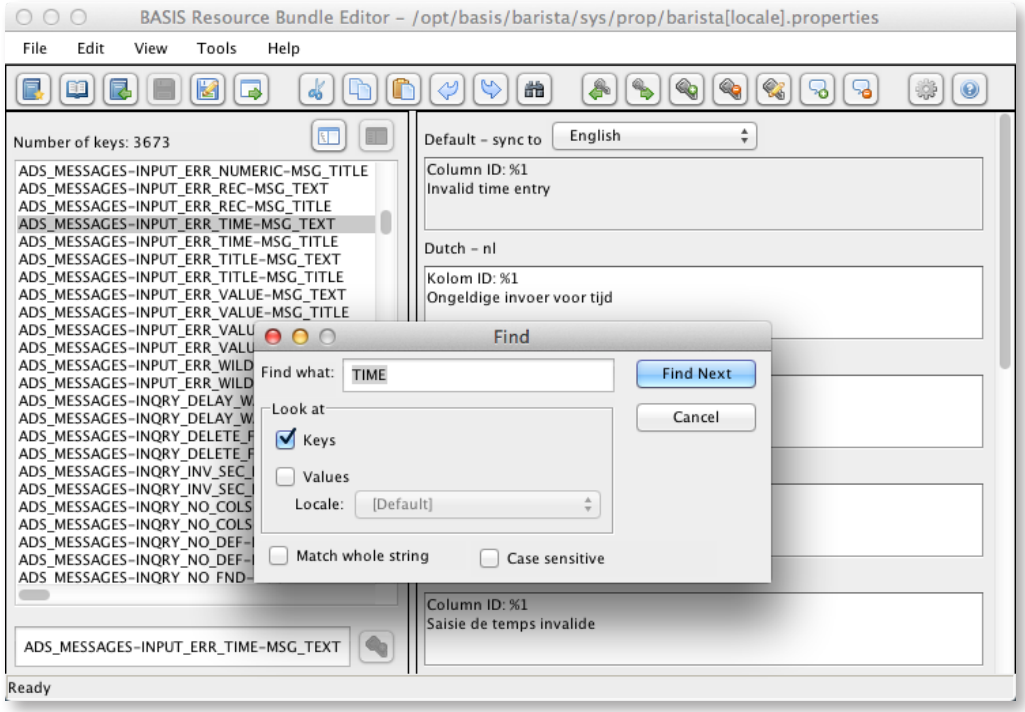

Figure 5. Find a key or value in a bundle

oriented methods (**Figure 6**) and CALL'able subroutines (**Figure 7**) to make the access to the resources bundles from legacy or object-oriented code as seamless as possible. Another significant advantage is that the resource bundle may exist in a directory referenced in the PREFIX.

```
use ::bbtranslator.bbj::BBTranslationBundle
use java.util.Locale

bundle! = BBTranslationBundle.getBundle("HelloWorld")
resourceBundle! = bundle!.getTranslations(new Locale("de","DE"))
resourceBundleValue$ = resourceBundle!.getTranslation("HELLO")
print "Resource bundle value: " + resourceBundleValue$
```

Figure 6. Use the BBTranslator's object-oriented methods to retrieve a translation for a key

```
call "bbtranslator.bbj::getInstance",resourceBundle!,"HelloWorld","de_DE","","",""
resourceBundleValue$ = resourceBundle!.getTranslation("HELLO")
print "Resource bundle value: " + resourceBundleValue$
```

Figure 7. Use the BBTranslator's CALL'able subroutines to retrieve a translation for a key

Developers who prefer programming in Java can use the Java ResourceBundle API to access the resource bundle. However, this method requires that the bundle is in a jar referenced in the BBj Classpath using a session-specific classpath. Developers can localize JasperReports using resource bundles packaged and referred to in this manner. Although Java code is necessary for this option, it is straightforward and almost as easy to implement as the CALL approach. ☺ See **Figure 8**.

```
use java.util.ResourceBundle
use java.util.Locale

resourceBundle! = ResourceBundle.getBundle("HelloWorld", new Locale("de","DE"))
resourceBundleValue$ = resourceBundle!.getString("HELLO")
print "Resource bundle value: " + resourceBundleValue$
```

Figure 8. Use the Java ResourceBundle API to retrieve a translation for a key

Summary

Whether or not you have internationalized your application, it is always a good idea to keep displayed string literals separate from the application code. The RBE is a great way to manage the resource bundles that contain this text. Since the RBE is a BBj utility, it can run in Java Web Start or in a browser (BUI mode) to facilitate direct exchange from professional translators to the resource bundle.

Before RBE, translators relied on a tool like MS Excel, which created an extra step for the developer to manually put the text into the application. Alternatively, translators could have used a resource bundle-aware IDE but most do not provide a consistent manner for handling the resource bundles. Another drawback to the second approach is that installing an IDE is a very heavy-handed requirement for non-programmers to perform translations.

Now, all one needs to do is to send the translator a link to the RBE application! Honestly, how can that be any easier? BASIS uses the RBE to manage resource bundles for the BASIS Custom Installer, the BASIS Product Suite Download page, the Barista RAD tool, and for the AddonSoftware building block. Won't you welcome this new bundle and add it to your family of tools? ■

- For more information
  - BBTranslator in the online documentation at links.basis.com/bbtranslator
- Read about
  - BBJabber in *Can Your App Speak to Your Customer?* at links.basis.com/09apptranslate
  - BUI localization in *Parlez-BUI Français?* at links.basis.com/11builocal