Lotus Developer Domain
The Technical Resource for Lotus software
LDD Today

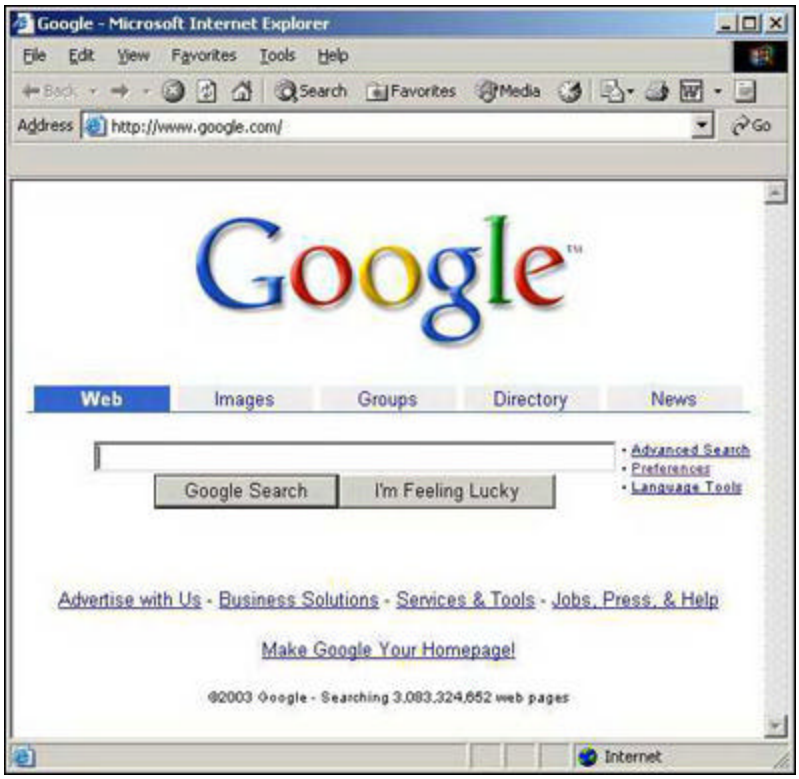
Level: Intermediate
Works with: Notes/Domino
Updated: 01-Jul-2003

by
Dan
O'Connor

Your employees speak only one language; unfortunately, they do not speak the same language. However, they all use the same database, so how do you support more than one language? In Notes/Domino R5, if you needed a multilanguage application, you had to use specialized tools to create it then had to translate the application yourself. As a result, many customers deployed single language applications because multilanguage applications were too complex to create and maintain. Often this also forced the customer to dedicate a server per language for maintenance reasons. However, new features in Notes/Domino 6 let you create a single application that can support more than one language with out-of-the-box functionality. In this article, we introduce you to the new multilanguage application features in Notes and Domino 6. This article tells you how to create a multilanguage and multilingual application using Notes/Domino 6 and Domino Global Workbench (DGW). This article assumes that you are familiar with Notes and Domino and have some familiarity with Notes template design.

What is a multilanguage application?

To get up and running, it is important to define what a multilanguage application is. A multilanguage application is one in which the same code is used in the back-end, but the front-end can have any one of a set of translated user interfaces (that is, a multilanguage application provides a single back-end, but multiple front-end interfaces). In the case of Notes and Domino, this means a template (NTF file) or a database (NSF file) that contains a common back-end, but has a multilanguage user interface. Multilanguage applications are quite common, even if they are not always apparent to the user. Multilanguage applications have become increasingly popular with the emergence of the World Wide Web. It is ever more important for companies to reach customers in different countries, and one of the best ways of doing this is by having their company Web site in multiple languages. There are dozens, if not hundreds of large corporations that currently provide this functionality on their corporate Web sites, but as it is such a subtle feature very few ever notice it. For example, **Google** has automatic language detection built into its Web site. If your browser's language preference is set to English, the Web site looks like this:



Now, change your browser's language setting to Japanese. For Internet Explorer users, follow these steps:

1. In Internet Explorer, choose Tools - Internet Options.
2. Click the Languages button.
3. In the Language Preference dialog box, click Add.
4. In the Add Language dialog box, choose Japanese, then click OK.
5. In the Language Preference dialog box, select Japanese, then click Move Up.
6. Click OK to save.
7. Click OK to close the Internet Options dialog box.
8. Click the Refresh button.

You see the following site:

As you can see, this is a powerful way of informing customers speaking different languages of your products. In this article, we show how this functionality can be attained in Notes and Domino 6.

What is a multilingual application?

There is a vital difference between multilingual applications and multilanguage applications. Multilingual applications have synchronized language content, whereas the content in a multilanguage application is only in one single language. In multilanguage applications, the design of the template/database is translated into multiple languages, but in a multilingual application both the design and the content of the application is translated into multiple languages.

What new features exist in Domino 6 for multilanguage applications?

With Notes and Domino 6, you receive a host of new globalization features. This article pays particular attention to the multilanguage applications that are shipped with Notes and Domino. In the earlier releases of Notes/Domino, if you wanted to develop a multilanguage application, you needed to create the application using a tool like Domino Global Workbench (DGW). It did not matter if you were working on a custom application or on one of the standard applications that ship with Notes, like the discussion database, you needed DGW.

The creation of multilanguage applications has been greatly simplified in Notes and Domino 6 with the Language Pack Installer, an application that merges different languages packs into the existing, standard Notes application templates. A language pack is a set of installable files that have been translated. In the case of Domino, the language pack contains the translated version of some of the templates that are shipped with the product. The introduction of "mergable" templates makes it much simpler for you to create multilanguage applications. This feature greatly reduces TCO (Total Cost of Ownership) for those who need to support more than one language. Having more than one language in a single template reduces the overhead associated with supporting multiple servers for multiple languages, or indeed multiple templates for multiple languages. You now have several different languages coexisting on a single server.

The following table lists the standard Notes application templates that you can merge with your existing application templates.

Template Name	File name	Install Type
Domino Directory	pubnames.ntf	Replace only
Extended Mail (N/D 6)	mail6ex.ntf	Replace, Add
Discussion - Notes and Web (6.0)	discsw6.ntf	Replace, Add
Team Room (6.0)	teamrm6.ntf	Replace, Add
Personal Address Book	pernames.ntf	Replace, Add
Bookmarks (N/D 6)	bookmark.ntf	Replace, Add
Mail (N/D 6)	mail6.ntf	Replace, Add
Doc Library - Notes & Web (N/D 6)	doclbw6.ntf	Replace, Add
Personal Journal (N/D 6)	journal6.ntf	Replace, Add
Resource Reservations (6.0)	resrc60.ntf	Replace, Add
Subscription (N/D 6)	headline.ntf	Replace only
Directory Catalog	dircat5.ntf	Replace only
DOLS Resource Template	dolres.ntf	Replace only
iNotes Web Access (R5)	inotes5.ntf	Replace only
iNotes Web Access (R6.0)	inotes6.ntf inotes60.ntf	Replace, Add

Another major advantage of having multiple languages stored in one template is that you do not have to create multilanguage databases from the multilanguage template. For example, if your template contains 12 languages, your database can have one, two, or more languages (up to 12) depending on your preference. As the application developer, you have two options:

- *Create multiple single language databases*
  Each language database provides a single language user interface.
- *Create a single database that supports multiple languages*
  This database provides multiple language user interfaces.

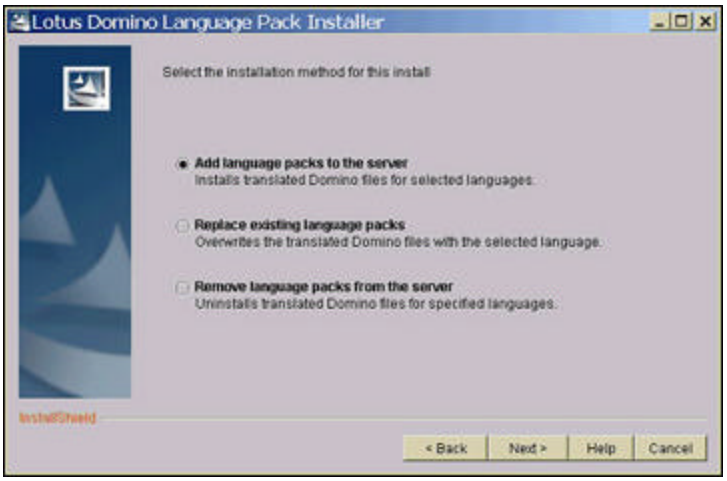
A single language database can inherit its design from a template with multiple languages. For example, from a discussion database template that contains English, French, and German, you can create a database that contains just one language or a database that contains any combination of those languages.

Merging multiple languages into one multilanguage template is a huge improvement over the functionality that was available in Domino R5. This functionality was not available "out-of-the-box" in Domino R5, but it is in Domino 6. In Domino R5, you had to take the English discussion database application and translate it using DGW. In Domino R5, we had translated versions of the standard application templates, but we did not have the functionality to automatically merge those language templates into one template in previous releases of Domino.

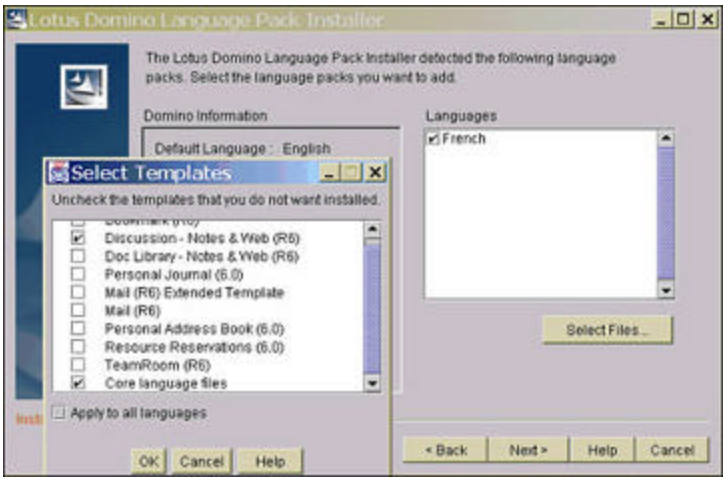
Installing the language packs on a Domino 6 server

For the purpose of this article, we install three language packs on an existing English Domino 6 server. This functionality works equally well if you install one, multiple, or all of the language packs on a single server. For this article, we install French, Spanish, and Japanese. Shut down your Domino 6 server, and proceed with the install. You can find the Lotus Notes/Domino Language Pack Installers on CD; they are also available for download the language pack installers from the Web.

1. Double-click the file W32DomLP60_GUI.EXE to launch the installation program.
2. Read the license agreement and accept it to continue installation. Click Next.
3. Choose the "Add language packs to the server" option, then click Next.



4. The installation program automatically detects the location of your Domino server. If the location is correct, simply click Next. Otherwise, enter the correct location and click Next.
5. The installation program examines the system requirements, then displays the required disk space and the amount of temporary disk space available. Click Next to proceed to the next screen.
6. Select the files and languages you want to install. Typically, there is one available language per language pack, but this may change in future releases.
7. To select the templates that you want to make multilanguage templates, click the Select Files button. Be default, all templates are selected. Deselect any template any templates that you do not want to install. For this article, make sure to keep the Discussion - Notes & Web (R6) template selected.
8. Click OK, and then click Next.

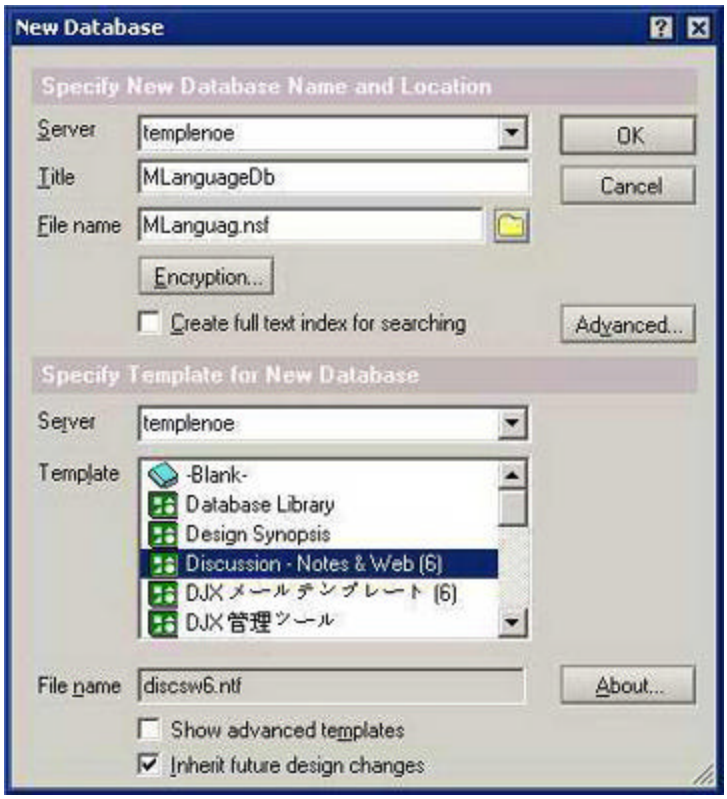


9. The installation program then tells you where it will install the files and asks that you verify that the information is correct. Click Next to install the language templates.
10. Repeat these steps (1 through 10) for as many languages as you need to install.

Congratulations—you now have all the elements in place to create a multilanguage discussion database!

Creating a multilanguage discussion database

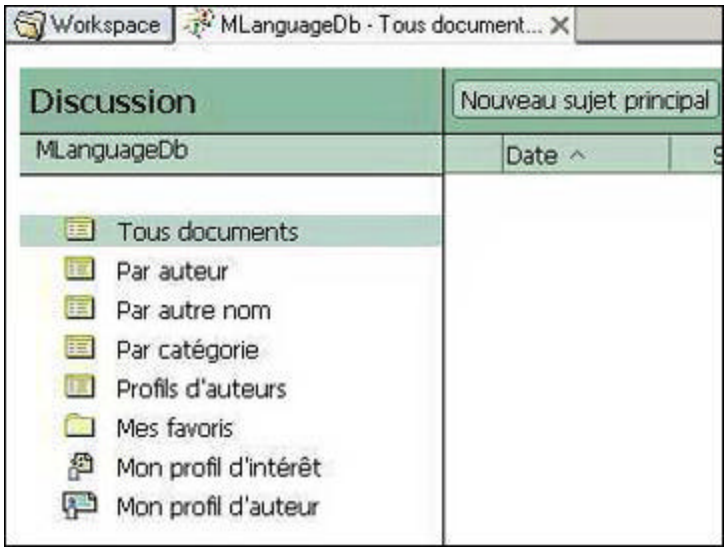
The task of creating a multilanguage discussion database is actually quite simple after you install the language packs. When you create a new discussion database, you are prompted by Notes to select which languages you want in the database. To see this, create a new discussion database on the server where you just installed the language packs. Choose File - Database - New. Select the server where you installed the language templates as your template server. Select the discussion template and create a new database based on that:



After you click OK in the New Database dialog box, you are prompted by the Select Template Languages dialog box to choose which languages you want to include in this database. You have two options. You can either choose to have one language per database, or you can alternatively have multiple or all languages in a single database. For this article, we demonstrate how to include all languages in the database. English was the default language, so select all of the languages that you installed, then click Add. Finally, click OK to create the database.

At this point, we have a fully functional multilanguage database. You can view this database through Notes or over the Web. The language that the database displays corresponds with your international setting in the Notes user preferences. Check the Content Languages dialog box for your language preferences (Choose File - Preferences - User Preferences. Choose the International tab, then click Change beside the Content Language field.)

Reorder the languages so that French is the first one in the list, restart Notes, and open the multilanguage database that you created. Notice that all of the UI is in French.



A Spanish UI is displayed if you change the user preferences to Spanish.

As you can see from the above example, it is extremely easy to create a multilanguage discussion database in Notes/Domino 6.

Creating a multilanguage application
Translating applications can be a tricky task at the best of times. If it were not for Domino Global Workbench (DGW), this task would be nearly impossible for some larger Domino-based applications. Luckily, DGW is one of the tools that ships with Domino Designer, so if you already have the Domino Designer CD, you can install DGW from there. This article assumes that you already have DGW installed and running on your system.

DGW is a tool developed by Lotus/IBM to help both ourselves and our customers build translated applications. It is best described as a set of software tools that help you to manage the localization (translation) of Domino databases, especially Web site databases. DGW also contains tools for managing the ongoing translation of the content (as opposed to the design) of a multilingual application. It provides services for detecting translation work as new content becomes available and a translation workflow process to route content through translation and approval automatically, making the newly translated content available in the application as it becomes ready . The Workbench automatically extracts terminology from your database application and stores it in glossaries ready for translation. Localized versions of the database are built automatically using the translated glossaries.

And if your database changes, you can use the Workbench's update features to transmit the changes easily through the localized versions. DGW can also add translation workflow features to existing databases. This article does not go into great detail about how DGW works or about the mechanics of the databases it create; this article is only meant as a quick reference to get you up and running with DGW, from which point you should have the confidence and knowledge to explore the product further.

There are essentially four main components in the DGW project:

- The Source application (database)
- The Glossary database
- The tagged database
- The Translated application (database)

The actual DGW process is fairly straightforward and is explained in great detail in "**Template Globalization Best Practices**." We will create a multilanguage application in this article. To start, create a blank database with one form and one agent. The form asks for a customer's name and address. The agent pops up a dialog box with some text for illustrative purposes. You can download the database from the **Sandbox**.

In the form, create five fields according to this table:

Field	Type
Name	Text
Address	Text
City	Text
State	Dialog list
Zip	Number

Here is a screen shot of the form:

Create a shared action to Save and Close the document and add the following code to the action:

```
@Command([FileSave]);
@Command( [RunAgent] ; "MessagePopup") ;
@Command([FileCloseWindow])
```

Next, create an agent called MessagePopup and add the following code to the agent:

```
Sub Initialize
      Msgbox "Document Saved.", , "Information"
End Sub
```

To translate the database, launch DGW. If this is the first time you launch DGW, you are asked if you want to

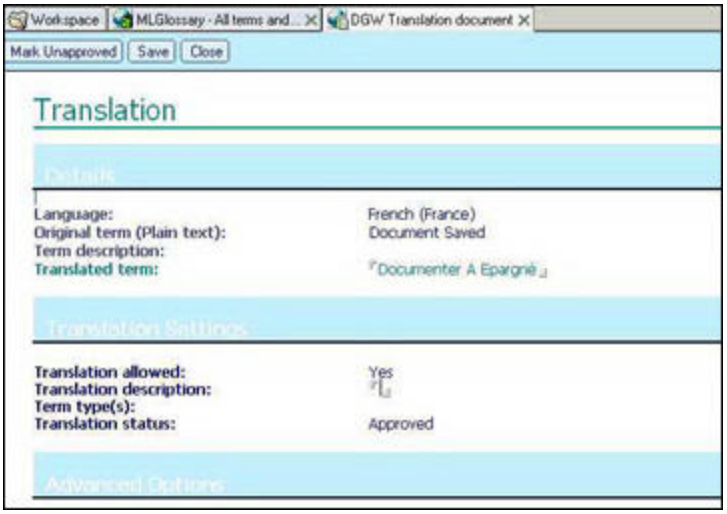
create a project database. Click Yes to continue. Click the New Project icon on the workspace to create a new DGW project. Give the project a name. Follow these step-by-step instructions to set up the project:

1. Choose the source database to be localized, choose the database that you created, and click Select.
2. Click Create to create a glossary for the application selected.
3. Click OK to proceed to the Language Selection dialog box. Select French (France), Japanese, and Spanish (Spain). English is the reference language. Click OK to create the glossary.
4. Click the glossary database in the project navigator, then click Create Tagged Database.
5. Name the database MLTag.nsf, and click OK. Right-click the glossary database in the project navigator and open the database in Notes. It opens the default view which is All Terms and Translations. In this view, you can see all the terms from your source database.
6. After you open the glossary, you can see all the translated and untranslated terms. When you first open the glossary, all terms are untranslated. A glossary element (a term) can be in any one of four states:

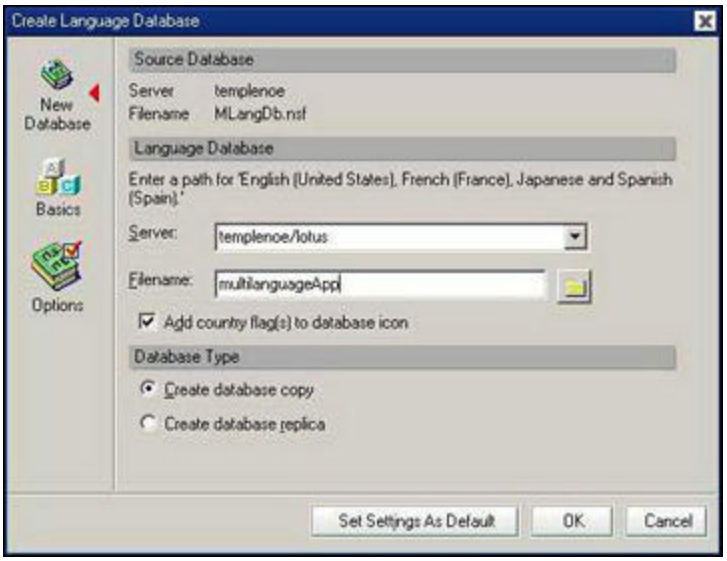
State	Description
Untranslated	This element is an untranslated term still in the reference language.
Translated & Unapproved	The element has been translated by an individual, but it has not been reviewed for accuracy. In your DGW build options, you can choose to exclude or include unapproved terms, depending on the translation process you use.
Translated & Approved	The element has been translated, and someone has reviewed the translation for accuracy and decided that the translation is sufficiently accurate.
Do Not Translate (DNT)	You cannot translate that element. DGW locks the element and will only use the reference language version of the term when building the language database. DNT terms are typically strings that are hardcoded, so changing them causes a bug in the code.

7. When the glossary is open, you can see all the translatable terms in the glossary. Expand each term until you get to the translatable terms.

These documents (terms) are editable, so opening the document and entering into edit mode allows you to enter translated text.



8. After you translate the text, you can mark it as translated and as approved. Repeat this for all translatable terms in the glossary. After the glossary is translated, it is time to build the language databases.
9. Return to DGW and to the project you created earlier.
10. Click the source database to refresh the project. Next click the tagged database. When you click this, the dialog box to add languages to the destination database opens. Add all the available languages, then click OK.
11. At this point, clicking the tagged database provides the option to Build Language Database. Click the Build Language Database button.



12. Fill in the information as appropriate. Choosing to build the database on a server is preferable. Click OK to build the database.
13. The database builds quickly because it is small, and the log is displayed at the bottom of the DGW screen.
14. Change your Notes user preferences to specify Japanese as your preferred content language as described earlier in this article, then restart Notes.
15. In DGW, right-click the built database, and open it in Notes. Go to the Create menu and choose cust_info. The customer information form appears. Fill in the information, and click Save and Close (translated to Japanese). Notice that all text is translated:

16. Go to your Notes user preferences, and change your preferred content language to Spanish. Restart Notes.
17. Reopen the database in Notes, and open the document you just created. All the text is in Spanish:

There is one important point to note from the previous example, and that is that the user entered information is not translated. That information is static once it is entered into the database; it is not translated by the application. This is the primary difference between a multilanguage and a multilingual application. The multilanguage application does not have translated content.

Conclusion

As you can see from the previous example, DGW is a very powerful tool that can greatly decrease the amount of time required to translate an application and the cost of maintenance of the application. The DGW project remembers the current state of the source database, so if changes are made to the source and DGW is loaded after that, DGW detects the changes in the source and applies them appropriately to the language database. The advantage is that DGW preserves all translation work and can share glossaries between projects. If you want to add support for extra languages at any subsequent time, you can easily do that. It is also worth noting that DGW works with Notes R5. It is possible to create multilanguage applications in R5 as well, but the out-of-the-box multilanguage templates do not ship with R5.

ABOUT THE AUTHOR

Dan O'Connor is a developer in the WebSphere Studio Application Developer Web tools team, currently developing tools to make Web development easier. Before joining that team, Dan was a developer with the Lotus Global Product Development (GPD) team, where he worked on the past two releases of Domino.Doc and Lotus Workflow. While there, he helped to ensure

both products met global customer requirements. Dan received his degree in Computer Engineering from the University of Limerick, Ireland.

