Lotus Developer Domain — The Technical Resource for Lotus software — LDD Today

admin admin/GPDCAM_LAP/IR;
ユーザー1の別名/サンプル組織名/サンプルドメイン名; ユーザー2の別名/サンプル組織名/サンプルドメイン名;
alternate user3/GPDCAM_LAP/IR;
alternate user4/GPDCAM_LAP/IR

Alternate name support in Domino.Doc and Lotus Workflow

Level: Intermediate
Works with: Domino.Doc, Lotus Workflow
Updated: 02-Dec-2002

by
Dan
O'Connor

In R5, Lotus introduced new functionality in Notes and Domino that let system administrators give users a second, or *alternate* name. Since then, alternate names has proven to be a popular feature—so popular, in fact, that we have incorporated alternate name support into other Lotus products. For example, Lotus Workflow 3.0 natively supports this feature. For other products, such as Domino.Doc, you can customize the server code to enable alternate name support.

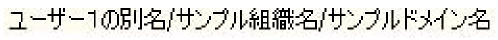
This article provides a brief introduction to alternate names and how they are used. Then it describes enabling alternate names in Notes/Domino and Lotus Workflow applications. Finally, it provides snippets of code that illustrate how you can implement alternate name support in Domino.Doc 3.1. You can download a sample database containing all code examples shown in this article from the **Sandbox**. The code samples include edited versions of Domino.Doc and Lotus Workflow forms and a script library.

This article assumes you're an experienced Notes/Domino developer with Formula language and LotusScript knowledge and are familiar with Lotus Workflow and Domino.Doc features and terminology.

What are alternate names?

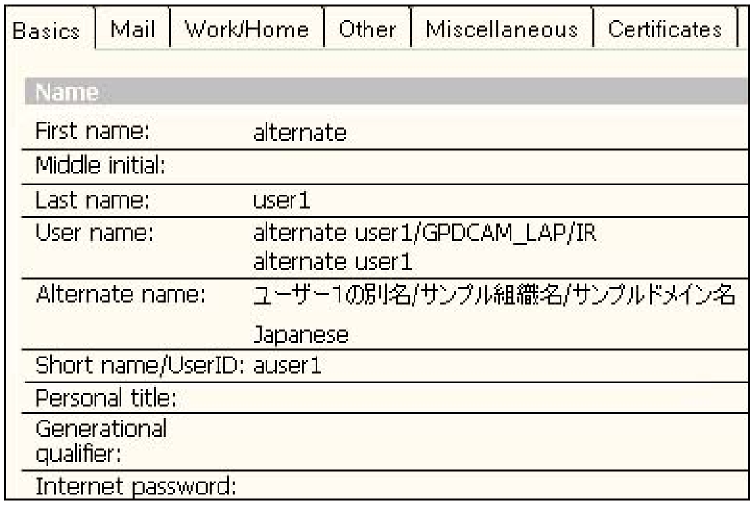
In Notes, there are three types of user names:

- Primary name
  All users have primary names. Notes/Domino only supports ASCII (single-byte) characters in the primary name, for example, Dan O'Connor/org/domain.
- Short name
  This is usually a combination of the first and last name and, as with the primary name, should always be ASCII characters. This name is created at the same time as the primary name. An example of a short name is doconnor.
- Alternate name
  Alternate names are usually non-ASCII names. They allow people to have DBCS (double-byte character set) and accented characters in their names. For example,

  
  ユーザー1の別名/サンプル組織名/サンプルドメイン名

  can be an alternate name.

Alternate names are set up through the Domino Administrator client using the domain certifier. When you, as Domino administrator, create a user name, you can also give that person an alternate name if the domain certifier is set up to do so. You can also assign alternate names to existing user names. The Basics tab in the Person document in the Domino Directory contains alternate name information:



Name

First name:	alternate
Middle initial:	
Last name:	user1
User name:	alternate user1/GPDCAM_LAP/IR alternate user1
Alternate name:	ユーザー1の別名/サンプル組織名/サンプルドメイン名 Japanese
Short name/UserID:	auser1
Personal title:	
Generational qualifier:	
Internet password:	

Why use alternate names?
Notes/Domino does not support non-ASCII characters in a user's primary name. However, many users in Asia and parts of Europe want to use their own names in Notes/Domino. Alternate names provide them a way to do this because they can contain non-ASCII characters.

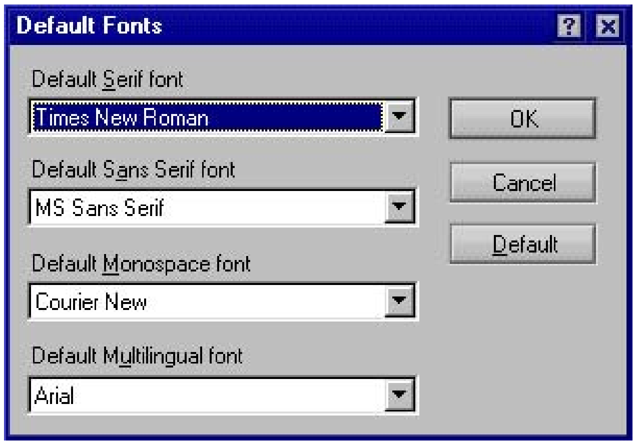
Alternate names are fully authenticated, just like primary names. This distinguishes an alternate name from an alias, which cannot be authenticated, so you can use alternate names in a database's ACL.

Where can alternate names be used?
Alternate names can be used throughout Notes and Domino. They can be displayed virtually everywhere primary names are—author fields, cc lists in mail memos, and so on. But before alternate names can be used, the Notes client must be configured for this feature, and the Notes/Domino application must be designed to use alternate names.

You can implement alternate names in a variety of different ways in an application. You can implement them in the user interface (UI) or in the ACL. Implementing alternate names in the UI is probably the easiest method for an existing (legacy) application. This results in alternate names being used only at display time, not for authentication.

If you are designing a new application that needs alternate names enabled, consider implementing alternate names at the ACL level. This means users' alternate names are added to the ACL, instead of their primary names, so that all components of the application that reference the ACL will not need to convert the user's primary name to the alternate name. (Note that if you take this approach, only the alternate name appears in the ACL, not the primary name.)

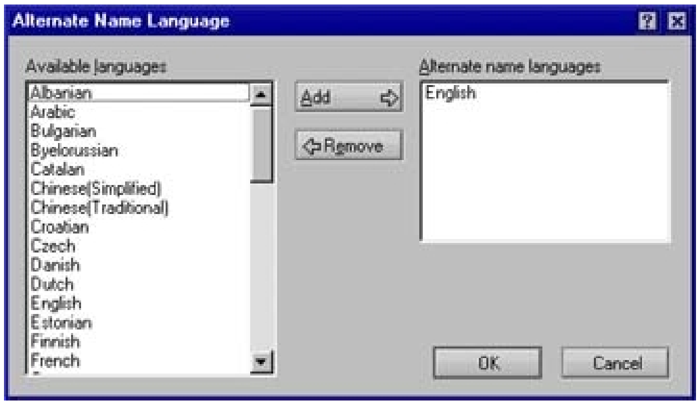
Enabling alternate names in Notes
You enable alternate names in your Notes client through the Default Fonts dialog box in your User Preferences:



Default Serif font: Times New Roman
Default Sans Serif font: MS Sans Serif
Default Monospace font: Courier New
Default Multilingual font: Arial

To do this, proceed as follows:

1. Open your current Location document. On the Basics tab, go to the Default display name field and select Display alternate names. Then click Save & Close.
2. Choose File - Preferences - User Preferences. On the Basics tab, click the Default Fonts button. This displays the Defaut Fonts dialog box, shown above.
3. For the four font fields—Default Serif font, Default Sans Serif font, Default Monospace font, and Default Multilingual font—select a font capable of displaying your alternate name in the appropriate language characters. You need not select this font for all four fields. Which fields you change depends on the applications in which you want to see the alternate names. For example, if the field containing the alternate name is using default sans serif (such as the To field in a mail memo), then you must specify the default sans serif font. Also, be sure to select the font applicable to your platform. For example, Tahoma is a good choice for Windows 2000. If you're unsure which font to select, consult your operating system documentation.
4. Click OK to return to the User Preferences dialog box.
5. In the Additional Options list on the Basics tab, select Enable Unicode display.
6. Click the International tab to move to that tab.
7. Click the Change button for Alternate name language.
8. In the Alternate Name Language dialog box, select the appropriate language in the Available languages list, click the Add button to add it to the Alternate Name languages list, and click OK.



Available languages: Albanian, Arabic, Bulgarian, Byelorussian, Catalan, Chinese(Simplified), Chinese(Traditional), Croatian, Czech, Danish, Dutch, English, Estonian, Finnish, French
Alternate name languages: English

9. Then click OK to close the User Preferences dialog box.
10. Restart Notes so that these changes take affect.

Your Notes client will now support alternate names.

Formula language and LotusScript support for alternate names

You can use LotusScript and @functions to enable alternate name support in a Notes/Domino application. This section outlines a few of these functions and briefly explains what they do. For full documentation about these functions, consult the **Domino Designer Help**.

@LocationGetInfo([NamePreference])
This undocumented function retrieves the user's alternate name preference set in the Location document. (See step 1 of "How do I enable alternate name support in Note? earlier in this article.) When this function is run via the Notes client, it returns 1 if Display alternate names is selected in the Default display name field. Otherwise, it will return 0. For example:

```
var = Evaluate("@LocationGetInfo([NamePreference])")
If(var(0) = "1") Then
    'The user wants to see alternate names, now we should figure out which languages they want to see.
    etc...
```

@NameLookup(look_up_flag; username; item_to_return)
This function performs a lookup in the Person document for the user specified in *username.* The *username* can be either a person's primary or alternate name. This function is very useful because it can return a variety of information. The following code returns the user's primary name:

```
@NameLookup([Exhaustive]; alternate_name; "FullName")
```

where *alternate_name* is a string containing the user's alternate name. This next sample returns the same user's alternate name language:

```
@NameLookup([Exhaustive]; alternate_name; "AltFullNameLanguage")
```

@NameLookup should be used sparingly because it has to perform a lookup every time it is called.

@UserName(x)
When x is 1, @UserName returns the user's alternate name. If the user does not have an alternate name, an empty string is returned. @UserName(0) is the same as @UserName, it returns the current user's primary name.

@UsernameLanguage(x)
When x is 1, @UsernameLanguage returns the language of the user's alternate name.

@LanguagePreference([AlternateName])
When this function is run through the Notes client, it returns a list of languages the user has selected as the preferred alternate name language in User Preferences. When this function is run through a Web browser, it returns the browser's language options.

NotesNameArray = notesSession.UserNameList
The first element of this array is associated with the user's primary name. The second element is associated with the alternate name. If a user only has a primary user name, UserNameList property returns an array of a single NotesName element.

Lotus Workflow

Lotus Workflow 3.0 includes alternate name support "out of the box." You can easily create applications through Lotus Workflow that are alternate name enabled. In this section, we describe how this functionality can be enabled, and what you can do with it. For more information, see the **Lotus Workflow product page** on LDD.

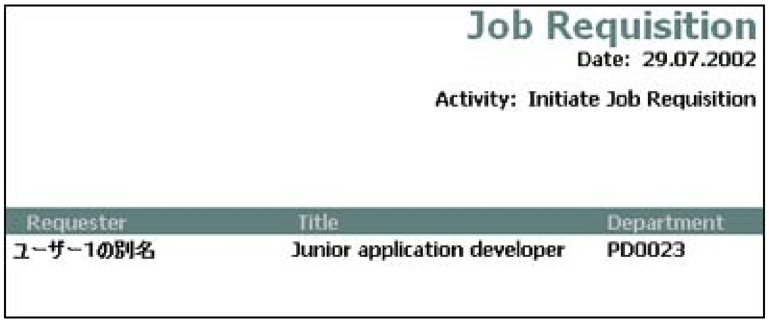
Enabling alternate names in Lotus Workflow
Before you can enable alternate names in Lotus Workflow, they must be enabled in Domino and Notes, as described earlier in this article. To demonstrate alternate names in Workflow, we will enable them in the samples database in the Employee Hiring process. The code provided enables alternate names in the cover document and also in the sample form.

To start, enable alternate names in the Job requisition form. You can use the customized Job requisition form in our sample database or make the edits on the standard form yourself. To do this, open the Job requisition form in Domino Designer. To add alternate name support to an application, you usually change all areas that normally display a person's primary name to also display the alternate name. In the sample Job requisition form, the Initiator field normally displays the primary user name of the person who initiated the process.

To enable alternate names in the Initiator field, make the field computed for display. This enables the field to change its contents dynamically. To do this, add the following @formula code to the Initiator field:

```
altLangsList := @LanguagePreference([AlternateName]);
userLang := @NameLookup([Exhaustive];InitiatorOS; "AltFullNameLanguage");
altLangs := @Implode(altLangsList; " ");
altName := @NameLookup([Exhaustive]; InitiatorOS; "AltFullName");
@If(@Contains(altLangs; userLang);
      @If(altName = "";
            @Name([CN];InitiatorOS);
      @Name([CN]; altName));
@Name([CN];InitiatorOS))
```

This code dynamically displays the user's alternate name, if the user selects this preference. When a user with an alternate name creates a job, the alternate name appears in the Initiator field.



Job Requisition
Date: 29.07.2002
Activity: Initiate Job Requisition

Requester	Title	Department
ユーザー1の別名	Junior application developer	PD0023

You also need to add and modify existing code to enable alternate names in the cover document. First, add a new subroutine to the OS Application Events script library (or if you prefer, use the OS Application Events script library contained in our sample database, which includes the new subroutine). The code for this subroutine is below, with extensive comments that explain what is happening:

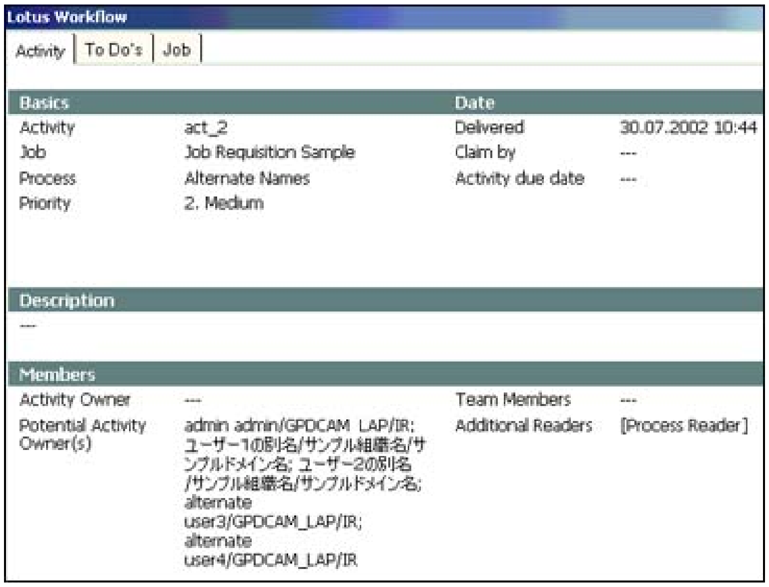
```
Sub DWFUtilUpdateUserInfo(doc As Notesdocument, soption As String, properties As Variant)
%REM
This routine updates the UserInfoOS field in the cover document.
This field contains additional information of participants in the format
username (canonical)#value1#value2#value3...
The routine performs an @NameLoopkup for each of the users for each property that is to be retrieved.
The parameter soption allows specifying a parameter for @NameLookup (see Designer documentation).
If empty, this parameter defaults to "[NOSEARCHING]".
The parameter Properties has to be an array of string denoting the itemnames that shall be retrieved.
If a string is passed, the function retrieves the AlternateLanguage Flag and Alternate Username.
It gets all usernames that are contained in the fields:
ExpandedParticipantAuthorsOS
:ActivityOwnerOS:ExpandedParticipantReadersOS:TeamBackupOS:ProcessReaderExpandedOS:UserInfoInputO
S
Then it checks whether or not information has been retrieved for these users already.
If not, it retrieves information from the Directory and stores it in the UserInfoOS field.
All entries in UserInfoOS that do not refer to an existing user are removed.
%END REM
      Const ScriptName = "DWFUtilUpdateUserInfo"
      On Error Goto Errorhandler
      Dim lProperties As Variant
      Dim lOption As String
      Dim DefaultProperties(1) As String
      Dim users As Variant
      Dim Formula As String
      Dim i As Integer
      Dim j As Integer
      Dim info As Notesitem
      Const UFormula = |
      user1:=@Unique(@Explode(@Explode(tmpUserOS;"#");";"));
      user:=@Trim(@Replace(user1;"[Process Reader]":"NoBodyOS";""));
      user|
	If soption = "" Then loption = "[NOSEARCHING]" Else loption = soption

	If Typename(Properties) = "STRING" Then
		DefaultProperties(0) = "AltFullNameLanguage"
		DefaultProperties(1) = "AltFullName"
		lProperties = DefaultProperties
	Else
		lProperties = Properties
	End If

	If Not doc.HasItem("UserInfoOS") Then
		Set info = New NotesItem(doc, "UserInfoOS","")
		info.isSummary = False
	End If

	' get user
	Const UserFormula = |user1:=ExpandedParticipantAuthorsOS : ActivityOwnerOS:
	ExpandedParticipantReadersOS: TeamBackupOS:
				ProcessReaderExpandedOS: UserInfoInputOS;
				userlist:=@Trim(@Replace(@unique(user1);"[Process Reader]":"NoBodyOS";""));
				UserWithoutInfo:=@Trim(@Replace(Userlist; @Word(UserInfoOS;"#";1);""));
				UserWithInfo:=@Trim(@Replace(Userlist; UserWithoutInfo;""));
				KeepInfo:=@Replace(UserWithInfo;@Word(UserInfoOS;"#";1);UserInfoOS);
				Field UserInfoOS := KeepInfo;
				UserWithoutInfo|

	gwfSession.DebugPrint ScriptName, Userformula

	' assemble formula
	users = Evaluate(UserFormula, doc)
	Formula =""
	For i = 0 To Ubound(users)
		If Not Trim(users(i)) = "" Then
			Formula = Formula + |e_| + Trim(Str$(i)) + |:="| + users(i) + |"|
			For j=0 To Ubound(lProperties)
				gwfSession.DebugPrint ScriptName,
				|@NameLookup(|+loption+|;"|+users(i)+|";"|+lproperties(j)+|") |
				Formula = Formula + |+ "#" + @NameLookup(|+loption+|;"|+users(i)+|";"|+lproperties(j)+|") |
			Next j
			Formula = Formula +"; "
		End If
	Next i

	Formula = Formula + |@SetField("UserInfoOS";@Trim(UserInfoOS:|
	For i = 0 To Ubound(users)
		Formula = Formula + |e_|+ Trim(Str$(i)) + ":"
	Next i
	Formula = Formula + |""));@success|
	gwfSession.DebugPrint ScriptName, formula
	Evaluate Formula, doc
	Exit Sub

Errorhandler:
	gwfSession.DebugPrint ScriptName, "Internal Error: " + Error$ + " (" + Str$(Err) + ")"
	Exit Sub
End Sub
```

The final step to enabling alternate names in the cover document is to add a call from the PostDocumentRouting_ function to the subroutine above. PostDocumentRouting_ should look like the following:

```
Private Function PostDocumentRouting_(RoutedDocument As NotesDocument, ActivityPropertyDocument As
NotesDocument)
```

```
    If Ucase(RoutedDocument.COVERDOCOS(0)) ="YES" Then
        Call DWFUtilUpdateUserInfo(routeddocument, "[ForceUpdate]:[Exhaustive]:[FallbackToUserName]", "" )
    End If
End Function
```

Now every time the document is routed, the alternate name information is updated. This code displays alternate names in the cover document in accordance with the user's Notes/browser settings. For instance, if a user only wants to see Japanese alternate names, the cover document for a job from the sample Hiring process appears like this:

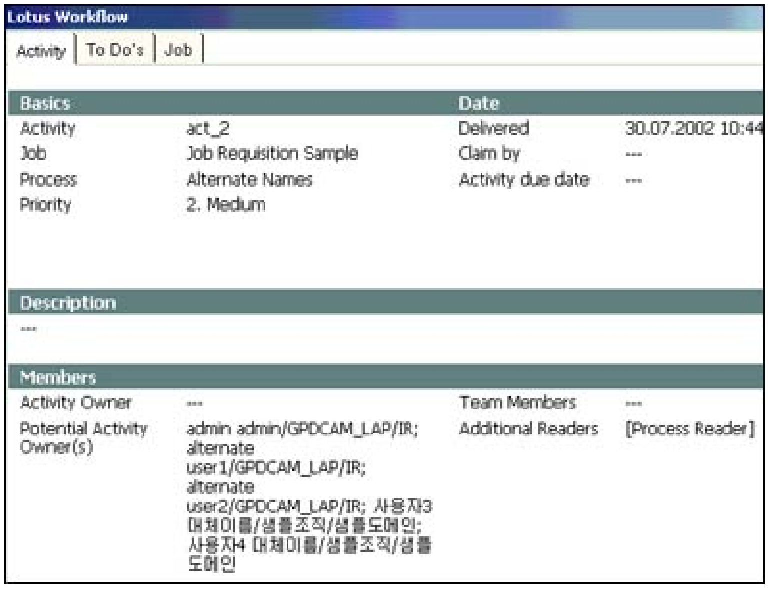


Basics		Date	
Activity	act_2	Delivered	30.07.2002 10:44
Job	Job Requisition Sample	Claim by	---
Process	Alternate Names	Activity due date	---
Priority	2. Medium		

Description

Members			
Activity Owner	---	Team Members	---
Potential Activity Owner(s)	admin admin/GPDCAM_LAP/IR; ユーザー1の別名/サンプル組織名/サンプルドメイン名; ユーザー2の別名/サンプル組織名/サンプルドメイン名; alternate user3/GPDCAM_LAP/IR; alternate user4/GPDCAM_LAP/IR	Additional Readers	[Process Reader]

But if the user wishes to see Korean alternate names, the same cover document looks as follows:



Basics		Date	
Activity	act_2	Delivered	30.07.2002 10:44
Job	Job Requisition Sample	Claim by	---
Process	Alternate Names	Activity due date	---
Priority	2. Medium		

Description

Members			
Activity Owner	---	Team Members	---
Potential Activity Owner(s)	admin admin/GPDCAM_LAP/IR; alternate user1/GPDCAM_LAP/IR; alternate user2/GPDCAM_LAP/IR; 사용자3 대체이름/샘플조직/샘플도메인; 사용자4 대체이름/샘플조직/샘플도메인	Additional Readers	[Process Reader]

As you can see, the code needed to enable alternate name functionality in Lotus Workflow is relatively straightforward. Our sample code can also enable alternate names through the Web interface.

Domino.Doc

Domino.Doc is a document management system that runs on Domino. It allows users to save a variety of types of documents directly from their desktops to a central Domino.Doc server. This can be done through three different clients: Notes, a Web browser, and the ODMA/Explorer client. Virtually all Domino.Doc server code is provided through the templates that make up a Domino.Doc library infrastructure. For more information on Domino.Doc, see its **product page**.

As mentioned earlier, there is no native alternate name support in Domino.Doc This section explains how to add alternate names support to Domino.Doc for Notes and Web users. We describe how to add code to the Domino.Doc server to make it possible to invite users with alternate names to a file cabinet. We then discuss how to customize Domino.Doc forms to display alternate names in a variety of ways.

Enabling alternate names in Domino.Doc
In standard Domino.Doc, if you try to invite a user with an alternate name to a file cabinet, you get an error stating that the user is not registered. To avoid this, you must replace all the alternate names in the invite field with primary name equivalents. To do this, open the Invitation form in the Domino.Doc library template (domdoc.ntf). Click anywhere on the form, then select the QueryClose event. Add the following code to the QueryClose event (or simply copy the Invitation form in our sample database and use it to replace the standard Invitation form in domdoc.ntf):

```
Sub Queryclose(Source As Notesuidocument, Continue As Variant)
    Dim singleName, inviteList, macroStr As String
    Dim length As Integer
    Dim var As Variant
    Dim fullName As NotesName
    Dim inviteListItem As NotesItem
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    '' Find out if the user sees alternate names or not.
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    var = Evaluate("@LocationGetInfo([NamePreference])")
    If(var(0) = "1")Then
        ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
        '' The user sees alternate names, this means when s/he selects a user
        '' who has an alternate name, the alternate name will be copied to the
        '' FCInviteList field. Now we want to check for the primary name.
        ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
        inviteList = Source.FieldGetText("FCInviteList") '' Get all the names, in the field.
                                                        '' It may include alternate names.
        Call Source.FieldSetText("FCInviteList", "")
        Set inviteListItem = Source.Document.ReplaceItemValue("FCInviteList", "")
        length = Instr(inviteList, Chr(10)) '' Check for CR, its presence suggests multiple names.

        If(length = 0)Then '' Probably just one name in the list, no CR found.
            length = Len(inviteList)
        Else
            length = length - 2 '' Skip the carriage return, and line feed (CR/LF).
        End If
        If(length <> 0 And length <> -1)Then '' We have at least one name in the list.
            Do While length <> 0                '' Get each name individually.
                singleName = Trim(Left$(inviteList, length))
                '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
                '' The user sees alternate names, singleName could be either an alternate name
                '' or a primary name. We need to check for the primary name. When we get the
                '' primary name back, place it in the list. Discard the old names.
                '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
                macroStr = {@NameLookup([Exhaustive]; "} + singleName + {"; "FullName")}
                var = Evaluate(macroStr, Source.Document)
                Set fullName = New NotesName(var(0))
                If(fullName.abbreviated <> "")Then          '' The value returned was a primary name
                    inviteListItem.AppendToTextList(fullName.abbreviated + ";")
                Elseif(singleName <> "")Then       '' No primary name returned, implies the original was a
                primary name.
                    inviteListItem.AppendToTextList(singleName + ";")
                End If
                '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
                '' Remove the current name from our temp list and move onto the next name.
                '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
                inviteList = Mid$(inviteList, (length + 3))
                length = Instr(inviteList, Chr(10))
                If(length = 0)Then
                    length = Len(inviteList)
                Else
                    length = length - 2
                End If
            Loop
        End If
    End If
End Sub
```

After you add this code to your Invitation form, you can invite users with alternate names to the cabinet and library. However, it is still not possible to display alternate user names. To do this, you need to customize Domino.Doc. One way to do this is to enable alternate names through the UI. You can add new fields to the forms that currently display the primary names. These new fields are hidden if the user does not have alternate names enabled.

In the following example, we show two ways to enable alternate names in Domino.Doc. Our preceding sample code allows us to invite users with alternate names to a file cabinet and library. Continuing with this theme, we will now illustrate how you can display alternate names in the FileCabAccess form.

Our earlier example changed the Invitation form so it always returns the user's primary name. Thus, even if the user has alternate names enabled, he will only see the user's primary name in the FileCabAccess form. The following code changes the form so it now displays alternate names instead.

First, you must add three new fields that display the alternate names to the FileCabAccess form in domdoc.ntf (or replace this standard form with the version we provide in our sample database). Make these editable text fields. They should use comma and new line as separator values. The fields should also allow multiple values and use the following hide/when formula:

```
@If(@Text(@LocationGetInfo([NamePreference])) = "1"; @False; @True) | @IsMember("$$WebClient";
@UserRoles)
```

For this example, name the fields:

- FCManagerDisplay
- FCAuthorDisplay
- FCReaderDisplay

When new users are invited to a file cabinet, the UpdateACLAndInvite global subroutine is called. We appended code to this subroutine to call a new global subroutine that we created. This new subroutine gets the current list of primary names, and if alternate names is enabled, it updates the alternate names fields.

Add the following to the UpdateACLAndInvite global subroutine (new code in **bold**):

```
' computed Readers/Authors fields.

'If alternate names is enabled, update the alternate name info.
var = Evaluate("@LocationGetInfo([NamePreference])")
If(var(0) = "1")Then
    Call UpdateDisplayedNames(doc)
End If
Call doc.ComputeWithForm(False, False)
```

The code in UpdateDisplayedNames is as follows:

```
Sub UpdateDisplayedNames(doc As NotesDocument)
    Dim allNames, listOfNames, altLangs As String
    Dim singleName As String
    Dim nameList As NotesItem
    Dim singleNotesName As NotesName
    Dim calculationVar(1 To 3), displayVar(1 To 3) As Variant
    Dim var0, var1 As Variant
    Dim nameLen, iterator, upperBound, x, i As Integer
    Dim langSelected As Variant
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    '' First of all, we need to find out which languages the user wishes to see.
    '' Once we have determined that, we next need to find out if the names
    '' in the list have alternate names. If they have an alternate name, we need
    '' to find out what the alternate name's language is. If the user wishes to see
    '' that language, then we display it; otherwise, just display the primary name.
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    iterator = 1
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    '' The following fields are for display purposes only. We will clear the fields first.
    ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    doc.FCManagerDisplay = ""
    doc.FCAuthorDisplay = ""
    doc.FCReaderDisplay = ""
    displayVar(1) = "FCReaderDisplay"
    displayVar(2) = "FCAuthorDisplay"
    displayVar(3) = "FCManagerDisplay"
    calculationVar(1) = "FCReaders"
    calculationVar(2) = "FCAuthors"
    calculationVar(3) = "FCManagers"

    var0 = Evaluate("@LanguagePreference([AlternateName])", doc)
    upperBound = Ubound(var0)
    altLangs = " "
    For i = 0 To upperBound
        altLangs = altLangs + " " + var0(i)
    Next
    Do While iterator < 4
        Set nameList = doc.GetFirstItem(calculationVar(iterator))
        allNames = nameList.Text
        Set nameList = doc.GetFirstItem(displayVar(iterator))
        iterator = iterator + 1
        nameLen = Instr(allNames, ";")
        If(nameLen = 0) Then
            nameLen = Len(allNames)
            If(nameLen <> 0)Then
                nameLen = nameLen + 1
            End If
        End If
        Do While nameLen <> 0 And nameLen <> - 1
            singleName = Left$(allNames, nameLen - 1)
            If(singleName <> "")Then
                Set singleNotesName = New NotesName(singleName)
                If(Instr(altLangs, GetAltLang(singleName, doc)))Then
                    langSelected = True
                Else
                    langSelected = False
                End If
                singleName = GetAltName(singleName, doc)
                If(langSelected = False Or Ucase(singleName) = "NO_ALT")Then
                    Call nameList.AppendToTextList(singleNotesName.Abbreviated + ",")
                Else
                    Call nameList.AppendToTextList(singleName + ",")
                End If
            End If
            allNames = Mid$(allNames, nameLen + 1)
            nameLen = Instr(allNames, ";")
            If(nameLen = 0) Then
                nameLen = Len(allNames)
```

```
                    If(nameLen <> 0 And nameLen <> -1)Then
                        nameLen = nameLen + 1
                    End If
                End If
            Loop
        Loop
End Sub
```

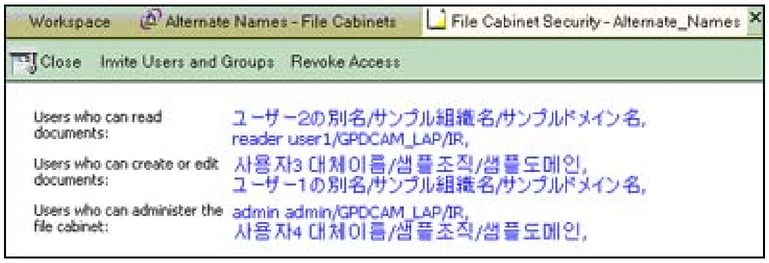
As you can see from the preceding subroutine, it calls a function called GetAltName. GetAltName (as the name suggests) simply returns the user's alternate name. The user's primary name is passed in as one of the function's parameters.

```
Function GetAltName(primName As String, doc As NotesDocument)As String
    Dim macroStr As String
    Dim fullName As NotesName
    Dim var As Variant

    macroStr$ = {@NameLookup([Exhaustive];"} + primName + {"; "AltFullName")}
    var = Evaluate(macroStr, doc)
    Set fullName = New NotesName(var(0))
    If(fullName.Abbreviated = "")Then
        GetAltName = "NO_ALT"
    Else
        GetAltName = fullName.Abbreviated
    End If
End Function
```

Finally, UpdateDisplayedNames calls another new function named GetAltLanguage. This returns the language of the alternate name. If this language is among the ones the user has specified in the Notes/Web browser settings, the alternate name is displayed. Otherwise, the primary name is used.

```
Function GetAltLang(primName As String, doc As NotesDocument) As String
    Dim macroStr As String
    Dim var As Variant

    macroStr$ = {@NameLookup([Exhaustive];"} + primName + {"; "AltFullNameLanguage")}
    var = Evaluate(macroStr, doc)
    GetAltLang = var(0)
End Function
```

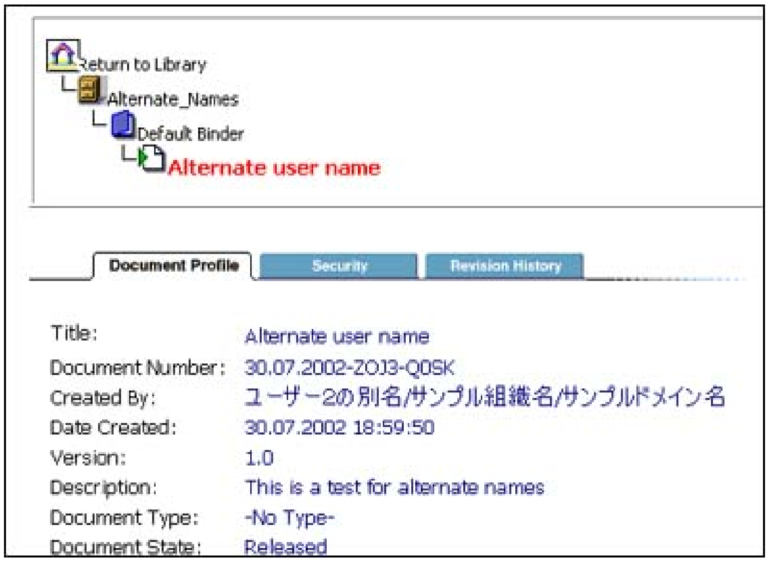
This code displays alternate names in the FileCabAccess form:


Workspace
Alternate Names - File Cabinets
File Cabinet Security - Alternate_Names
Close
Invite Users and Groups
Revoke Access

Users who can read documents:	ユーザー2の別名/サンプル組織名/サンプルドメイン名, reader user1/GPDCAM_LAP/IR,
Users who can create or edit documents:	사용자3 대체이름/샘플조직/샘플도메인, ユーザー1の別名/サンプル組織名/サンプルドメイン名,
Users who can administer the file cabinet:	admin admin/GPDCAM_LAP/IR, 사용자4 대체이름/샘플조직/샘플도메인,

Much of the code required to implement alternate names support in Domino.Doc is resuable, basically designed to get the primary name from the alternate name, or vice versa. And if you use the examples provided, you should be able to enable this functionality at your site relatively quickly.

Customizing alternate names in Domino.Doc for Notes users
Now we'll explain how to customize the Notes version of a Domino.Doc document to display a user's alternate name in the author field. You do this by adding two new hidden fields to the NewDocumentNotes form. These fields contain the author's alternate name and the alternate name language.



Title:	Alternate user name
Document Number:	30.07.2002-ZOJ3-Q0SK
Created By:	ユーザー2の別名/サンプル組織名/サンプルドメイン名
Date Created:	30.07.2002 18:59:50
Version:	1.0
Description:	This is a test for alternate names
Document Type:	-No Type-
Document State:	Released

To get alternate names to display in the Notes document profile, you need to modify the NewDocumentNotes form (or replace this standard form with the modified version in our sample database). Do this by adding a new hidden computed when composed text field named AlternateLanguage. The value of this field is:

```
@UserNameLanguage(1);
```

You then need to add a second field that will also be computed when composed. Name this field AlternateName. The value is:

```
@UserName(1);
```

Save the form and close it. Next, edit the DocumentNotes form (or replace it with our modified sample). Edit the code in the CreatedBy field so it now can dynamically switch between alternate names and full names:

```
primaryName := @Name([Abbreviate] ; DocAuthor);
alternateName := @Name([Abbreviate]; AlternateName);
altLangsPref := @Text(@LocationGetInfo([NamePreference]));
altLangs := @Implode(@LanguagePreference([AlternateName]); " ");

@If((@Contains(altLangs; AlternateLanguage) = @TRUE) & (altLangsPref = "1");
    @If(AlternateLanguage = "";
        primaryName;
        alternateName);
    primaryName);
```

With this code in place, you can dynamically display alternate names, depending on the user's Notes preferences. To display alternate names in a Web browser, you must implement similar changes in the Web versions of the forms mentioned above, for instance NewDocumentWeb. The majority of the @functions used for the Notes UI can also be used for the Web.

Conclusion

Alternate name functionality can obviously be very useful. We recommend using alternate names because it is the only way Lotus products can support non-ASCII user names. And if you follow the examples above, you can customize both Lotus Workflow and Domino.Doc so alternate names appear dynamically.

ABOUT THE AUTHOR
Dan O'Connor is a developer in the Lotus Global Product Development (GPD) team, working on the past two releases of Domino.Doc and Lotus Workflow. He helps ensure both products meet global customer requirements. Dan received his degree in Computer Engineering from the University of Limerick, Ireland.