Guidelines for how to write code that can be easily translated and that can be run in multiple languages in Dexterity in Microsoft Dynamics GP

CONFIDENTIAL ARTICLE

(The information in this article is provided to you in accordance with your Confidentiality Agreement)

Partner Only Article

Article ID : 942749
Last Review : N/A
Revision : 1.0

INTRODUCTION

This article contains guidelines for how to write code that can be easily translated in Dexterity in Microsoft Dynamics GP and in Microsoft Business Solutions - Great Plains. Additionally, this article contains guidelines for how to write code that can be run in multiple languages.

When Dexterity was originally designed, one principle of the design was to allow for applications to be easily translated into other languages. Another principle of the design was to allow for the terminology and the format to be easily localized for specific countries.

MORE INFORMATION

Guidelines for how to write Dexterity code that can be easily translated

To write Dexterity code that can be easily translated, use the following guidelines:

- Do not use hard-coded strings. Use messages instead.
- Do not concatenate messages. For example, use the *%1* placeholder, use the *%2* placeholder, or use substitution.
- Do not use a message resource if it should not be translated. Use a constant instead.
- Do not assume anything about the size of a message resource. Overestimate the possible length of a message resource.
- Do not use a single message to do the work of many messages. Create separate messages for each use.
- Do not use strings that end in spaces or messages that end in spaces. Trailing spaces cannot be seen, and trailing spaces are lost.
- Do not use messages to assign key values in tables. Use constants.
- Do not use text in bitmaps. Resources can have both text and pictures assigned to the resources.
- Maximize the size of the fields for prompts. Leave space for prompts to be longer after they are translated.
- When you substitute data into messages, only substitute data from tables or from calculations.
- Do not assume that all letters are between a and z or between A and Z. Also consider extended characters. For example, consider the following extended characters:
  - ç
  - è
  - é
  - ê
  - Ö

Guidelines for how to write Dexterity code that can be run in multiple languages

Even if an application can easily be translated, you may still encounter problems when you run the application in multiple languages on the same system. To write Dexterity code that can be run in multiple languages, use the following guidelines:

- Do not use messages to store data or to retrieve data. Use constants to store data or to retrieve data.
- Do not use sorted lists unless it is absolutely necessary. However, you can use sorted lists if the user types the items in the list.

- Do not store a string in a table unless the user types the string or unless the string has a Language ID in the key. However, you can store a string in a table if the string is a constant value that the user cannot see. Or, you can store a string in a table if the table is a true temporary table.
- Trigger the **syLanguage** procedure to add third-party resources that need translation.
- Include the Language ID when you design new tables.
- Do not use messages where constants are more appropriate.

REFERENCES

For more information, click the following article number to view the article in the Microsoft Knowledge Base:

942751 Description of the Language ID global system variable in Microsoft Dynamics GP

APPLIES TO

- Dexterity, when used with:
  Microsoft Dynamics GP 10.0
  Microsoft Dynamics GP 9.0
  Microsoft Business Solutions–Great Plains 8.0

Keywords: kbexpertiseadvanced kbmbspartner kbhowto kbinfo kbmbsmigrate KB942749