

转 MFC的多国语言界面的实现

2013年10月15日 21:25:24 阅读数：5801

目前很多软件都是要出口到多个国家，因此，为软件提供多国语言支持就成为了一个基本条件。为软件提供多国语言的支持的具体实现方法有很多，但基本原理都差不多，就是实现代码和语言包的独立，代码根据设定的语言选择语言包。

其中，MFC的资源文件就提供了对多国不同语言的支持功能，如果使用MFC开发，直接用资源文件自带的多国语言支持，可以省去不少的麻烦。

下面就介绍给MFC程序添加中英文的支持，开发环境为VS2010。

1. 新建工程

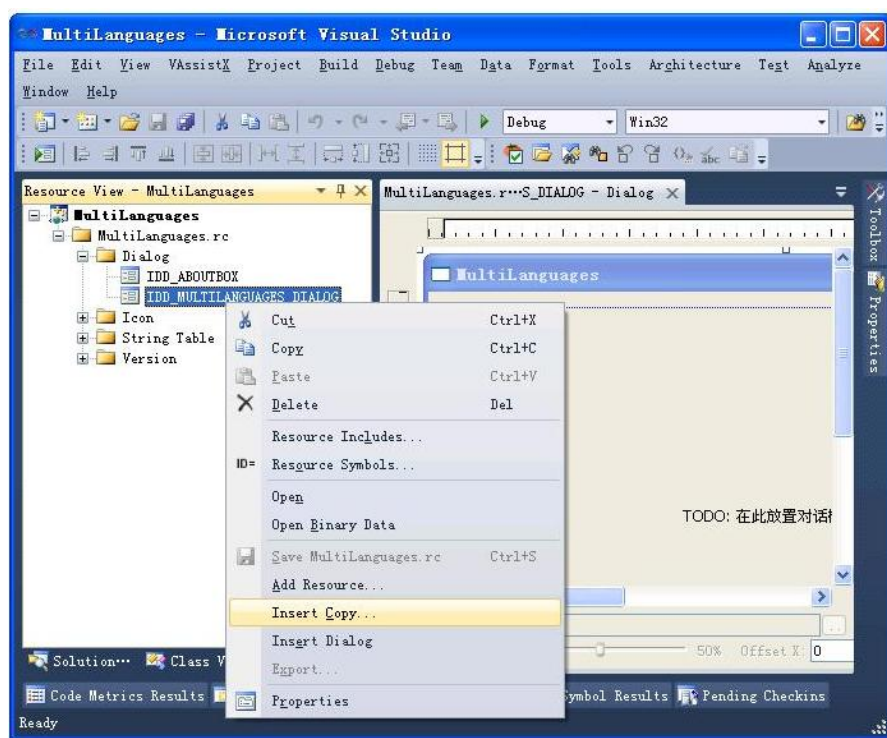
新建了一个对话框工程，工程名称为MultiLanguages，默认语言选择是“中文”。

2. 添加多国语言的资源

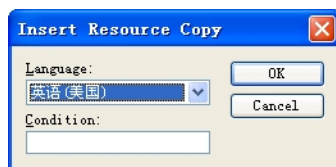
在创建工程后，工程会添加默认的资源，如主对话框，都是“中文”资源。现在我们需要添加相应的英文的资源文件。

为主窗口IDD_MULTILANGUAGES添加英文资源的方法为：

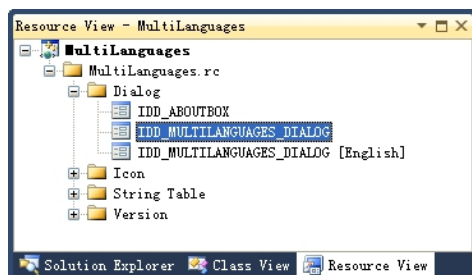
- (1) 打开Resource View窗口。
- (2) 右键IDD_MULTILANGUAGES，点击弹出菜单中的“Insert Copy”菜单，如下图所示。



- (3) 弹出窗口资源复制语言选择窗口，选择语言为“英语（美国）”，如下图所示。



- (4) 点击OK，即完成英文版对话框的添加。完成添加后，IDD_MULTILANGUAGES就对应于两个不同语言版本的对话框了，如下图所示。



使用同样的方法，也可以为其他资源添加多国语言版本的支持。主要需要多国版本需要支持的有对话框、菜单和字符串。

添加多国语言的资源后，要对这些资源进行不同语言的定制，根据资源对应的语言，设置对话框和控件的标题等。

3. Locale

程序的语言选择跟操作系统语言(System Locale)、用户设置语言(User Locale)和线程语言(Thread Locale)有关。程序运行时,是根据线程语言来选择资源的。如果程序中未对线程语言进行设置，线程语言默认采用用户设置语言。设置线程语言的函数是SetThreadLocale。

设置线程语言为“中文”的代码如下：

```
SetThreadLocale(MAKELCID(MAKELANGID(LANG_CHINESE, SUBLANG_CHINESE_SIMPLIFIED), SORT_DEFAULT));
```

设置线程语言为“英语（美国）”的代码如下：

```
SetThreadLocale(MAKELCID(MAKELANGID(LANG_ENGLISH, SUBLANG_ENGLISH_US), SORT_DEFAULT));
```

注：在新的系统中上述代码可能不起作用，比如我的WIN7+VC2010系统中，设置语言应该使用如下代码
SetThreadUILanguage(MAKELCID(MAKELANGID(LANG_ENGLISH, SUBLANG_ENGLISH_US), SORT_DEFAULT));

设置线程语言要在对话框创建之前，否则无法更改对话框的资源。可以在CMultiLanguagesApp::InitInstance函数中的对话框初始化之前添加线程语言设置，分别设置为中文和英文语言，就可以查看到对话框界面的不同。

4. 字符串处理

程序的多国语言的支持，不仅包括界面的多国语言支持，也要包括各类字符串的多国语言支持，如弹出的提示信息。因此，在弹出提示信息时，也要为提示信息创建多个语言版本，并根据当前线程的语言来选择不同的提示信息。

例子：实现不同语言版本中按钮的点击次数的统计。

(1) 在String Table中分别添加中英文的IDS_STRING_SAMPLE资源，内容如下表所示。

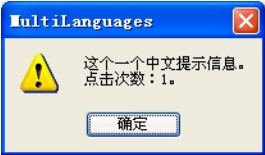
中文	这个一个中文提示信息。\\n点击次数：%d。
英文	This is a prompt message in English.\\nClick Times:%d.

(2) 在主窗口控件中添加一个控件Button1, 控件的中文名为“提示”, 英文名称为“Prompt”。为该控件添加一个左键单击消息响应函数, 该函数的内容如下：

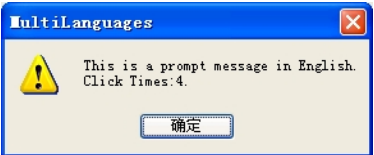
```
[cpp]
1. void CMultiLanguagesDlg::OnBnClickedButton1()
2. {
3.     // TODO: Add your control notification handler code here
4.     static int s_iClickTime = 0;
5.     s_iClickTime++;
6.     CString strPrompt = _T("");
7.     CString strFormat = _T("");
8.     strFormat.LoadString(IDS_STRING_SAMPLE);
9.     strPrompt.Format(strFormat, s_iClickTime);
10.    AfxMessageBox(strPrompt);
11. }
```

(3) 分别在CMultiLanguagesApp::InitInstance添加设置线程语言为中文和英文的代码，然后多次点击按钮进行测试。

中文版本弹出的提示框如下图所示：



英文版本弹出的提示框如下图所示：



5. 语言切换

窗口在初始化时候就导入了资源文件，在通过SetThreadLocale更换了线程语言后，窗口的资源并不会更改，必须要通过代码来重新装载资源。因为窗口中存在多种与线程语言相关的资源，重新启动软件一种叫快捷的更新语言环境的方法。

例：通过菜单来进行语言切换，切换语言后重启软件。

(1) 为程序添加中英文菜单选项ID_LANGUAGE_SWITCH,并为该菜单添加消息响应函数,其中,m_bRestartFlag使用判断关闭窗口时是否需要重启程序的标识。代码如下。

```
[cpp]
1. void CMultiLanguagesDlg::OnLanguageSwitch()
2. {
3.     // TODO: Add your command handler code here
4.     // 读取当前线程的语言,并根据当前线程语言进行语言切换
5.     LCID lcidNew = GetThreadLocale();
6.     if (LANG_ENGLISH == PRIMARYLANGID(LANGIDFROMLCID(lcidNew)))
7.     {
8.         lcidNew = MAKELCID(MAKELANGID(LANG_CHINESE, SUBLANG_CHINESE_SIMPLIFIED), SORT_DEFAULT);
9.     }
10.    else
11.    {
12.        lcidNew = MAKELCID(MAKELANGID(LANG_ENGLISH, SUBLANG_ENGLISH_US), SORT_DEFAULT);
13.    }
14.
15.    // 把语言设置写入配置文件
16.    CFile file;
17.    file.Open(_T("Language.ini"), CFile::modeWrite | CFile::modeCreate | CFile::typeBinary);
18.    file.Write(&lcidNew, sizeof(lcidNew));
19.    file.Close();
20.
21.    // 关闭窗口
22.    m_bRestartFlag = TRUE;
23.    PostMessage(WM_CLOSE, 0, 0);
24. }
```

(2) 在关闭窗口时,重启动该程序。即在窗口响应WM_CLOSE时,重启程序。代码如下：

```
[cpp]
1. void CMultiLanguagesDlg::OnClose()
2. {
3.     // TODO: Add your message handler code here and/or call default
4.     // 判断是否需要重新启动窗口
5.     if (m_bRestartFlag)
6.     {
7.         CString strFileName = _T("");
8.         GetModuleFileName(NULL, strFileName.GetBuffer(MAX_PATH), MAX_PATH);
9.         ShellExecute(NULL, _T(""), strFileName, NULL, NULL, SW_SHOWNORMAL);
10.        strFileName.ReleaseBuffer();
11.    }
12.
13.    CDialogEx::OnClose();
14. }
```

(3) 在启动软件时,根据当前软件的配置文件中语言来设置线程语言,即在CMultiLanguagesApp::InitInstance函数中创建对话框之前设置线程语言,代码如下：

```
[cpp]
1. // 判断你是否存在配置文件,如果存在,从配置文件中读取语言设置
2. CString strFileName = _T("Language.ini");
3. if (PathFileExists(strFileName))
4. {
5.     LCID lcidThread = 0;
6.     CFile file;
7.     file.Open(strFileName, CFile::modeRead | CFile::typeBinary);
8.     file.Read(&lcidThread, sizeof(LCID));
9.     file.Close();
10.    SetThreadLocale(lcidThread);
11. }
```

(4) 通过点击菜单来测试软件的语言切换。

6. MessageBox的问题

由于MessageBox中的按钮的语言是跟操作系统相关的,要想实现MessageBox按钮的多语言化是有很有一定难度的。我现在还没有查到好的解决方

法，很多网友的建议是抛弃MessageBox，自己建立对话框。

7. 源代码

源代码的下载链接如下：

<http://files.cnblogs.com/xianyunhe/MultiLanguages.rar>

8. 相关函数和类型

与本地化相关的函数和类型如下：

GetSystemDefaultLCID

GetSystemDefaultLocaleName

GetUserDefaultLCID

GetUserDefaultLocaleName

SetThreadLocale

GetThreadLocale

MAKELCID

MAKELANGID

LCIDToLocalName

LocalNameToLCID

LANGIDFROMLCID

PRIMARYLANGID

LCID

LANGID

LANG_CHINESE 0x04

LANG_ENGLISH 0x09

原文地址：<http://www.cnblogs.com/xianyunhe/archive/2011/09/02/2163842.html>

文章标签：

界面

语言

VC

MFC

个人分类：[纯编程](#)

此PDF由spygg生成,请尊重原作者版权!!!

我的邮箱:liushidc@163.com