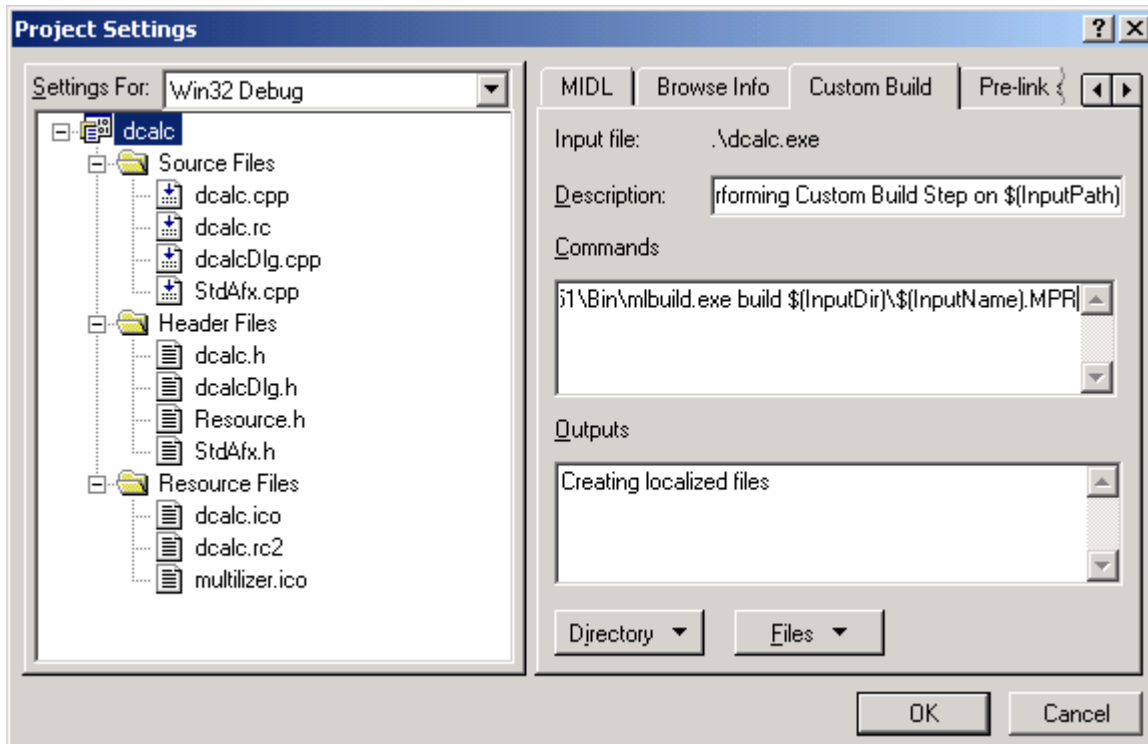


Visual C++ 6.0 lets configure the build settings for a VC++ project. Using this feature, it's easy to integrate localization in the build process. Thanks to Multilizer command-line functionality (included in Multilizer 5.1 Enterprise), you can fully integrate Multilizer in the build process.

Choose **Project**→**Settings...**, and choose **Custom Build** tab to start customization.



In Commands add the following command-line statement:

```
D:\BIN\ml51\Bin\mlbuild.exe build $(InputDir)\$(InputName).MPR
```

<multilizerpath>/mlbuild.exe is the path to Multilizer command-line utility. It can scan, translate, build, add targets, import/export, and create Exchange packages, from command-line<sup>1</sup>. In this case, it builds localized versions of the software, based on the settings in the Multilizer project (MPR-file).

In abovementioned statement, path to Multilizer project is specified in a generic way. Of course any qualified path will do the build.

With these customizations, build process will include localization in the process, and the build log includes also messages related to the localization.

```
Deleting intermediate files and output files for project 'dcalc - Win32 Debug'.
-----Configuration: dcalc - Win32 Debug-----
Compiling resources...
Compiling...
StdAfx.cpp
Compiling...
dcalc.cpp
dcalcDlg.cpp
Generating Code...
Linking...
```

<sup>1</sup> To see the command-line syntax and help, run mlbuild.exe without parameters.

```
Performing Custom Build Step on .\dcalc.exe
MLBuild Version 5.1.31 Copyrights (c) 1995-2002 Multilizer Oy
Opening the project file ... OK
Start building the localized items
Creating 'D:\BIN\ml51\VCPP\Samples\binary\dcalc\en\dcalc.exe', the English version of
'dcalc.exe' ... OK
Creating 'D:\BIN\ml51\VCPP\Samples\binary\dcalc\fi\dcalc.exe', the Finnish version of
'dcalc.exe' ... OK
Localized items succesfully built

dcalc.exe - 0 error(s), 0 warning(s)
```

In this particular project Multilizer binary localization was used. So in practice the non-localized EXE was built with VC++, and custom build used the non-localized EXE and Multilizer project (MPR) to create the localized versions.

This arrangement ensures that both original EXE and the localized versions are identical.