# Use Case 2 Analysis of a Windows Application with C or Cpp

Analysis of a Windows application with C/C++



In this use case, we will focus now on how to analyze a Microsoft Windows application:

## Sample application

From 'Help – Samples' option, in Visual Studio 13 Community edition, you can download the source code of our sample application.

#### **Database Query From MFC**

 Technologies
 SQL Server, Visual Studio 2012

 Topics
 User Interface, Data Access, Database Synchronization, Database, Database

 Connectivity
 Superior Control C

Platforms Desktop, Data

## Analizing the code

First of all, start 'Kiuwan Local Analyzer' with the {KiuwanLocalAnalyzer\_install\_dir}\kiuwan.cmd script.

Then, create a new application and set the source code directory.

kiuwan		
Specify an application		
Application name *	Database Query From MFC	
Label your analysis		
Analysis Label		
Select your code		
Select folder(s) to analyze *	C:\_dev\vs\Database Query From MFC	
Include patterns		
Exclude patterns	**/Generated Files/**,*/src/test/**,**/MAC	
	Continue X Exit	

In the 'exclude patterns' option, add the '\*\*/Generated Files/\*\*' path. Visual Studio generates temporal files in this directory and we need to exclude them from the analysis because we don't want to analyze files we haven't coded.

Click 'Continue', select 'C++', and analyze...



Once the analysis has finished, you can see the results on Kiuwan:



Everything seems just ok, but if you look closely, only four files were analyzed out of the eight files detected by Kiuwan Local Analyzer.

If we go to the log screen to find out what happened, we can see which files were left out:

UNPARSED FILES (4)	
Batabase Guery From MFC: Database Guery From MFCOlg.h	
Batabase Guery From MFC: Database Guery From MFCOIg opp	
Batabase Guery From MFC:Database Guery From MFC:pp	
Database Query From MFC:Database Query From MFC:h	

## Solving parser problems

Looking in the log directory for this analysis ({KiuwanLocalAnalyzer\_install\_dir}\temp\Database Query From MFC.1444389776635-0\results), we can find the file cpp.unresolved.headers.log.

The content of this file are the header files that Kiuwan couldn't find during the analysis. Most probably because they are not in our project folders:

<SDKDDKVer.h>

<afxwin.h>

<afxext.h>

<afxdisp.h>

<afxdtctl.h>

<afxcmn.h>

<afxcontrolbars.h>

"afxwin.h"

<afxdb.h>

"afxdialogex.h"

In fact, these are Windows and Visual Studio header files our application uses, but are located in external directories. As we can see in the previous post, the resolution of these files is necessary to get a complete and reliable analysis.

With the last version of Kiuwan Local Analyzer, you can autoconfigure the header files paths and macro definitions using the log file that Visual Studio generates on the compilation phase.

To get the default Visual Studio header files paths, run the following commands from a DOS console:

> cd C:\\_dev\vs\Database Query From MFC

> call "C:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\vcvarsall.bat"

> echo %INCLUDE% > include.txt

(update the source code and Visual Studio directories with those of your installation).

To get Visual Studio log file we need to compile the application from the command line:

> cd C:\\_dev\vs\Database Query From MFC

> call "C:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\vcvarsall.bat"

> msbuild.exe "Database Query From MFC.sln" /nologo /target:Rebuild /p:Configuration=Release /p: Platform=Win32 > msbuild.log

Now, we can use the generated 'include.txt' and 'msbuild.log' files to configure Kiuwan Local analyzer.

Before clicking on 'Analyze', open the configuration section:



Copy the content of 'include.txt' file in 'Header directories' option. You need to replace the semicolon separator by a comma separator before the copy.

C:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\INCLUDE,C:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\ATLMFC\INCLUDE,C:\Program Files (x86)\Windows Kits\8.1\include\shared,C: \Program Files (x86)\Windows Kits\8.1\include\winrt

You should also set the 'Use all know macros?' to true.

Analysis Configuration Extensions editor Select block: C++ - O C++ configuration		
Header directories:	C:\Program Files (x86)\Microsoft Vis	
Header extensions:	h,hh,hpp	
Use all known macros?:	true 🗸	
Process conditional directives?:	true 🔹	
Define macros:	🖉 Edit	
Analyze build logfile:	🖻 Details	
	Save 🗙 Cancel	

E.

Now, in this same screen, click 'Analyze build logfile - Details' button, to load the msbuild output logfile:

\_

Extensions editor		
Select block: C++ -		
© C++ configuration		
Header directories: C:\Program Files (x86)\Micr	osoft Vis	
Header extensions: h,hh,hpp		
Use all known macros?:		
Process conditional directives?:	-	
Define macros:		
Analyze build logfile: C Details		
Save	Cancel	
Ruild tool output applying results	×	
Log file C1 dev/wt)Database Queer From MEC1 mshuild L	~	
Application Database Query From MFC (msbuild.log		
Language C++		
0 include directories found:		
e 6 pre-defined macros found:		
#define NDEBUG	-	
#define UNICODE		
#deline wind2		
#define _AFXDLL #define _UNICODE		
<pre>#define wiN22 #define _AFXDLL ' #define _UNICODE ' #define _WINDOWS</pre>		
#define _AFXDLL     #define _UNICODE     #define _WINDOWS     Click "OK" to add the found includes and macros to your current     Click "Cancel" to discard these results.	configuration.	

After these steps, run the analysis again and check the new results on Kiuwan:



Now, all the eight C++ files of this sample application were correctly analyzed.