

# Access Europe – Peter Cole (32 to 64-bit Conversion)

We are going to look at updating Access so it will run in 64-bit.

Most of you will already have done the upgrade but for the benefit of those viewing the video we will recap the problem and solution

We open the getusernames sample and look at the code. We see RED

Open the free Win32ApiViewer open declares and paste GetUserNameA into the Alias.

After pressing enter the Declare details are shown and instantly added to the clipboard.

Note in a Declare if only the Function name appears there will not be an Alias visible but the Alias will be identical to the function name.

Alias is preferable for searching as developers could have used different function names.

We go back to the DB and paste the new Declare underneath then comment the original.

No more red!

The only difference is that PtrSafe has been added! But in most cases, it's not that simple!

Full details of manual upgrades can be found here



32to64Video.url

The question is how much work is required to update?

We can get an idea using our Free AccessAPIScanner database.

We will select the Clipboard example and Process. We see there are 10 Knowns and 0 Unknowns.

What are known? The 1550 declares from the Microsoft file Plus, currently 40+ others.

If we look at the tables in the win32apiviewer we can see tblWIN32APIDeclares and

tblWIN32APIDeclaresOthers. They are combined using a union query.

There are a number of fields plus the full Declare in a LongText (memo) field.

Yes, they are not normalised but easy to read and make updates manually. More about updates later.

We could update our DB manually as we demonstrated earlier using the viewer but in addition to the Declares we would have to make changes to the calling code.

We will look to see how much red there is in Clipboard db!

We close the Clipboard DB and open the (not free) Access32264Updater.  
We select the Clipboard db.  
The default is "Analyse and Process" so we just click Continue Analyse and Process  
We see a message box when complete.  
We Open the clipboardDB we see 3 test forms have been created.

We open set and add some text and test.  
Then we open Get and test, out test text is returned.

If we "Clear" then "Get" again an error message shows as there was nothing to retrieve!

You will be thinking. What has been updated?  
If we click the Analysis tab and refresh F5 then we see a list of the declare names their alias's and module where Declared.  
Clicking a module will open the file comparison program "WinMerge" Showing the differences between the original and the updated version.

Winmerge is opensource available from  
[WinMerge - You will see the difference...](#)

You can all see why adding comments regarding changes would not be a good Idea!

We can see that in the Declares PrtSafe has been added and some Long types have changed to LongPtr.  
Note. hWnd has been changed to hwnd.

Name changes in parameters are no problem, however Alias Names must be the correct case as they refer to functions within the library dll which are case sensitive.

What else has changed?  
In Subs where Calls are made the variable names have had prefixes added  
Lng for Long values  
Str for String

Variables that are LongPtr have been prefixed lpt

By using a prefixe it is easy to compare the original Dim type with the type in the parameters of a call and adjust if required.  
You can see there a lot of changes to Lpt!  
Not hard to do manually as compiling will reveal type mis matches but is slow and tedious.

Types are prefixed with udt which means that len can be adjusted to lenB when a len applies to a type.

We will run the scanner on AllSamples [ *Not provided in Samples for download*]  
50 Knowns and 0 unknowns

We will come back to the individual modules in All samples later.

How can we make unknowns known?

We will look at an example from Colin.

Running AccessAPIScanner shows 4 unknowns and 16 known.

Clicking View Results opens a folder containing 2 text files

DeclaresIn-OnScreenKeyboardDEMO\_v2-1-accdb and

ModulesIn-OnScreenKeyboardDEMO\_v2-1-accdb

Opening Declaresin- shows the unknown declare definitions

The separator is | pipe since it will not normally be used in code.

ModulesIn- contains the code modules containing the Declares.

This will assist in creating test functions for the calls.

Since the Api functions are very simple they are often used in combination to achieve the desired functionality so just providing the declarations does not facilitate testing. Think back to the Clipboard example it had 10 Declares to provide the functionality.

You can edit out any code, in the text file, that you do not wish to share or does not contain calls.

Noting that some functionality builds on multiple subs containing calls. More on that in a later example.

In this case the module is not actually needed since Colin has tested this on 32 and 64 bit.

When happy with the module text "Zip and send" will open an email with an attachment.

Please leave the subject line as presented.

The body of the email should be left as displayed but you can add any further details at the end.

The message will appear in my Outlook in "DeclaresIn" sub folder.

We now look at how the exported text files are processed at ThemeMyDatabase using

FindMissingFunctions.accdb

In Outlook in the folder 32to64 we see a "Declaresin" subfolder here incoming files are stored using a rule.

Exit scanner and open

In FindMissingFunctions clicking "Get Folders with Declares" will read the email and unzip into a folder.

The Declares New form will open with the First unknown declare. It will show both PtrSafe and old versions if available.

Additionally we see the return type for functions.

If the function is specific to 32 or 64 bit then that should be entered into 64or32. Note virtually all of functions the work in both 32 and 64 bit. We will see an example later where there are 32 and 64 bit versions.

Status 0 indicates unchecked 1 is for checked. Colin has verified these so should be set as 1.

We normally have to check the parameters are correct as the source may have just had PtrSafe added with no other adjustments.

Source indicates where the additional Declares came from.

Status 1 is entered then again on the next record and "continue" pressed.  
A Types form will open and pressing continue will move to a blank Lib Names form.  
If there are DLL's that are not in an available list then they are recorded. They can be flagged as available or not available in 64 bit. Some old Dll files don't have 64 bit versions available.

We now "Copy New Declares and Types to Others"  
This step allows for multiple sets of Declares in files to be collected.

When all input has been collected we  
"Prepare Updates"  
An update number and location displays, with date time appended to the name  
Close FindMissingFunctions

We will check in the api viewer for the Wow64DisableWow64FsRedirection Declare statement.  
It is not found.  
We now update using the update file just created then search for  
Wow64DisableWow64FsRedirection  
The Declare statement is shown, and loaded onto the Clipboard ready to paste.

We go back to the scanner and apply the new update

Checking the OnScreenKeyboard db we see  
0 Unknown Total Known 20!

Recapping we can scan databases or collections of databases for Declare statements.

A count will be provided for known functions and those not currently known.  
For Unknowns details can be easily sent for incorporation into the known list.

Example Updates [ *May not be in order in the Video!*]

4CheckModRegistry

Run the updater  
Look at the differences

Open Check Registry it compiles with no problems

Open the test form  
We see 3 parameters  
KeyToGet which is a LongPtr  
We use -2147483647 which is the value of HKEY\_CURRENT\_USER  
KeyNames string  
Software\Microsoft\Office\16.0\Common\Toolbars\Access

Then Value the name as string  
QuickAccessToolbarStyle

We Click test and see the result 20! The update has worked.

Reviewing the Analysis in the updater and clicking the module name we see significant changes to the calling code!

5pointtest

We open the db and move the mouse and an error displays

Update

Analysis

WindowFromPoint one of the few where there are 32 and 64 bit versions.

Updater adds

#If Win64

The 64 bit version uses one LongLong parameter

Whereas the 32 bit uses 2 Longs.

When called there is another #If Win64

The 64 bit uses a function PointToLongLong that takes a Point parameter and returns LongLong

The updater adds the PointToLongLong Function to the end of the Module

DON'T OPEN FORM Alt+F11

Compile

Minor problems in the code update found

Udt twice and lngIC should be lptIC twice.

( Note These minor problems have been corrected in the samples)

3FileFindModules.accdb

Ancient and complex file listing with attributes

Problems

lpthFind not defined

Updater Options

Analysis Only. Shows modules with Declare statements on the exclusions tab.

Options to remove module types from processing, only update Declares not code, don't create test forms and comment options.

Special record count feature where old code uses an Integer for the RecordCount of a record set.

Why Not to include pre VBA7.

Versions of Access before 2013 no longer receive security updates so are at higher risk if connected to the internet. Since I will not know if a user of the updater is aware of the heightened risk the safest approach is not to supply pre VBA 7 code. Obviously a custom version of a database could be built if required.

My reading of advice from GCHQ is that machines that have software that cannot have security updates should have that software removed or be disconnected from the internet.

I believe GCHQ know a bit more about security than me.

See

[Cyber-Essentials-Requirements-for-Infrastructure-v3-0-January-2022.pdf \(ncsc.gov.uk\)](#)

The relevant part is



The Applicant must ensure all in scope software is kept up to date. All software on in scope devices must be:

- licensed and supported
- removed from devices when it becomes un-supported or removed from scope by using a defined "subset" that prevents all traffic to / from the internet

Using unsupported software also runs the risk of invalidating insurance that customers may have against cyber-attacks.

In some cases, the customer may be able to use a free newer runtime version.

If people only have mde versions of old databases they may be able to have them converted to mdb by EverythingAccess.

See

[MDE to MDB Conversion - Retrieve VBA code from your MDE/ADE Databases \(everythingaccess.com\)](#)

A unique service from the developer of vbWatchdog

I have recently seen the results of a recovery and it is as good as the original!

For details of the updater please contact

[support@thememydatabase.co.uk](mailto:support@thememydatabase.co.uk)