How to Build DDAddin

2018-03-25, DDAddin Version 2.0.4

This document describes, how to build your own DDAddin version based on the sources supplied with the OEM license.

Content

1	I	Requirements				
2	Download Source Code					
3	Compile 32bit DLL					
	3.1 Adapt Build Step for Digitally Sign the DLL					
	3.2	2 Ente	r License Key in Build.cpp	2		
	3.3	B Com	pile	2		
4	Compile 64bit DLL					
5	(Compile Setups				
6	I	Install				
7	I	Debugging				
	7.1 Detect Outlook Bitness		ect Outlook Bitness	4		
	7.2 Register Debug Artifact of DDAddin		ster Debug Artifact of DDAddin	4		
	7.3 Debug Outlook with DDAddin		4			
	7.4	Poss	ibly Interesting Lines when Debugging	4		
	-	7.4.1	Add-in Initialization	4		
	-	7.4.2	Drag&Drop Handling	5		

1 Requirements

- Microsoft Visual Studio 2017, C++ Projects.
- Microsoft Visual Studio 2017 Installer Projects, <u>https://marketplace.visualstudio.com/items?itemName=VisualStudioProductTeam.Microsoft</u> <u>VisualStudio2017InstallerProjects</u>

2 Download Source Code

In your order fulfillment mail, a link is supplied from where you can download the ZIP archive that contains all sources of DDAddin.

• Download the ZIP and extract it into a working directory. We refer to this directory as "InstDir".

3 Compile 32bit DLL

- Open Visual Studio solution InstDir/ddaddin32.sln.
- Three projects are listed in Solution Explorer:

- o ddaddin: COM Add-in DLL that is loaded by Outlook to enabled D&D to web pages.
- TestDDAddin: Test application which can be used to install and uninstall DDAddin.
- MergeModule32: Merge Module used to build an MSM file which contains the setup steps for DDAddin and is referred by the setup project introduced later.

3.1 Adapt Build Step for Digitally Sign the DLL

- In Solution Explorer window, right-click on project "ddaddin" and select "Properties".
- Remove or adapt the "Post Build Event" that calls "signtool" to digitally sign the DLL.

ddaddin Property Pages ? X						
Configuration: Release	✓ <u>P</u> latform:	Win32 ~	Configuration Manager			
 Configuration Properties General Debugging VC++ Directories C/C++ Linker Manifest Tool Resources MIDL XML Document Generator Browse Information Build Events Pre-Build Event Pre-Link Event Post-Build Event Code Analysis 	Command Line Description Use In Build	"c:\Program Files (x86)\Microsoft SDKs\Windows\v7.1A\Bir Yes	n\signtool.exe" sign /a /t (

• Repeat the previous step for "Platform x64".

3.2 Enter License Key in Build.cpp

- Open file "InstDir/Build.cpp"
- Replace wstring DDADDIN_LICENSE = L"DEMO";
- By your license: wstring DDADDIN_LICENSE = L"your license key";

3.3 Compile

• Set solution configuration and platform to Release/Win32:

bug	Tea <u>m</u>		<u>T</u> ools	Te <u>s</u> t	A <u>n</u> aly	/ze
-	Release	•	Win32		-	►

- From Visual Studio menu bar, select "Build Build Solution".
- Close Visual Studio
- The artifacts built from Visual Studio can be found in InstDir/Release_86.

4 Compile 64bit DLL

- As a requirement, the section 3 has to be completed.
- Open Visual Studio solution ddaddin64.sln
- Set solution configuration and platform to Release/x64
- From Visual Studio menu bar, select "Build Build Solution".

- Close Visual Studio
- The artifacts built from Visual Studio can be found in InstDir/Release_64.

5 Compile Setups

- Open Visual Studio Solution InstDir/Setup/OEM/oem.sln
- Adapt properties of Setup32 and Setup64 projects:

Properties 👻 🕂 🗙								
Setup32 Deployment Project Properties								
□ Misc								
AddRemoveProgramsIcon	(None)							
Author	Your Name							
BackwardCompatibleIDGeneration	False							
Description	This add-in allows to drag-drop mails from Outloc							
DetectNewerInstalledVersion	True							
InstallAllUsers	True							
Keywords								
Localization	English (United States)							
Manufacturer	Your Company							
ManufacturerUrl	http://www.your-url.com							
PostBuildEvent								
PreBuildEvent								
ProductCode	{E04974DD-47BE-4E05-84F1-16F88894E75D}							
ProductName	Your DDAddin OEM Product							
RemovePreviousVersions	True							
RunPostBuildEvent	On successful build							
SearchPath								
Subject								
SupportPhone	· · · · · · · · · · · · · · · · · · ·							
Misc								

- Set solution configuration and platform to Release/Default
- From Visual Studio menu bar, select "Build Build Solution".
- Close Visual Studio
- The artifacts built from Visual Studio can be found in InstDir/Setup/OEM/Setup32/Release and InstDir/Setup/OEM/Setup64/Release.
- Digitally sign the artifacts with your certificate.

6 Install

Uninstall DDAddin version from WILUTIONS, if there is installed one – the DEMO version might be installed.

On Windows 32bit, install InstDir/Setup/OEM/Setup32/Release/ddaddin32.msi.

On Windows 64bit, install InstDir/Setup/OEM/Setup64/Release/ddaddin64.msi. This setup includes the COM DLLs for Outlook 32bit and Outlook 64bit.

7 Debugging

- 7.1 Detect Outlook Bitness
 - Detect the Bitness of your Outlook installation, e.g. see <u>https://stackoverflow.com/questions/2203980/detect-whether-office-is-32bit-or-64bit-via-the-registry</u>
 - For Outlook 32bit, use ddaddin32.sln in the following sections.
 - For Outlook 64bit, use ddaddin64.sln in the following sections.

7.2 Register Debug Artifact of DDAddin

- Start Visual Studio as Administrator.
- Load solution ddaddin32.sln resp. ddaddin64.sln.
- Set solution configuration and platform to Debug/Win32 resp. Debug/x64.
- In Solution Explorer, right-click on project "TestDDAddin" and select "Set as Startup Project".
- Open file "CheckInstall.cpp".
- Lookup line "... Uninstall(L""); ..." and comment out this line.
- Start project "TestDDAddin" by "Debug Start Debugging".
- Close Visual Studio

7.3 Debug Outlook with DDAddin

- Open Visual Studio as Non-Administrator and load ddaddin32.sln resp. ddaddin64.sln.
- Set solution configuration and platform to Debug/Win32 resp. Debug/x64.
- In Solution Explorer, right-click on project "ddaddin" and select "Set as Startup Project".
- In Solution Explorer, right-click on project "ddaddin" and select properties.
- Enter Outlook program file at:

ddaddin Property Pages ? X						
Configuration: Debug	 Configuration Manager 					
Configuration Properties General Debugging	Debugger to launch: Local Windows Debugger		~			
VC → Directones ▷ C/C++	Command Command Arguments	\$(TargetPath)	✓			
Linker Manifest Tool Resources	Working Directory Attach	\$(ProjectDir) No				
 ▷ MIDL ▷ XML Document Generator 	Debugger Type Environment	Auto				
Browse Information Build Events	Merge Environment SQL Debugging	Yes No				
 Custom Build Step Code Analysis 	Amp Default Accelerator	WARP software accelerator				

e.g.: C:\Program Files %28x86%29\Microsoft Office\root\Office16\OUTLOOK.EXE

7.4 Possibly Interesting Lines when Debugging

7.4.1 Add-in Initialization

The Add-in class registered with outlook is CWIDragDropAddin, defined in files WIDragDropAddin.h/.cpp. The first function called by Outlook is OnConnection.

7.4.2 Drag&Drop Handling

Replacement of Drag&Drop functions is performed in HookDD::EnableDragDrop, see file HookDD.cpp.

When a drag-operation is started, function HookDD::MyDoDragDrop (file HookDD.cpp) is called. It invokes FileDrop::handleDataObject (FileDrop.cpp) to construct an IDataObject with file drop information.

At this point, only file names are created and packed into DDItem objects. The files are empty yet. If the dragged item is a mail (not an attachment), a MailItem object is created for the item. Mail properties are optionally read, formatted as JSON text, and saved as text entity into the IDataObject (see MailItem::toJson, MailToEml.cpp).

On drop, the program goes through MyDropSource::QueryContinueDrag (MyDataObject.cpp). If the operation is successful, the prepared file items are saved.