



Plugin Architecture for NODE

Limitations

Due to AHK's limitations, the Plugins will have to be compiled with the host application, so there is currently no known way to give NODE Plugins the authority they currently have while keeping them separate from the main exe file.

Overview

The Plugins currently work such that whether one of them is removed or added, NODE can work independently with or without that plugin's features. If an Plugin exists, certain specifically named functions are searched for, and if found NODE adds that Plugin's objects, sub-object database, actions etc. to its searchable domain.

Every plugin can either add a new Object, add new Actions or add actions to existing object.

Following functions are supported:

Function name	Short description
P_PluginName_Actions	This function returns a list of Actions available to the selected Object.
P_PluginName_ConfGUI	If this plugin will have certain user configurable settings, then provide this function. This function receives the GUI_ID of the Configuration window, and a tab will be automatically created by the name of this plugin.
P_PluginName_ConfInfo	This function will allow Plugin Authors to add their contact information and plugin help in a separate function callable from the Info button in settings.
P_PluginName_ConfSave	This function will autorun when the Configuration window's Save button is pressed.
P_PluginName_Execute	This function is called when the user launches an Object+Action combination in NODE.
P_PluginName_KeyWords	This function can directly affect the internal algorithm of scoring the results
P_PluginName_Objects	This function simply returns the pipe separated list of Objects it supports .
P_PluginName_OnExit	This is a function that is automatically called when Node exits.

P_PluginName_RefreshTime	This function, if it exists, simply provides the number of <u>minutes</u> after which the database would be refreshed by calling the P_PluginName_Scan function.
P_PluginName_Scan	This function, if found, will be called every time the database is to be refreshed (for example when NODE starts).
P_PluginName_SearchAnalyze	This is a special type of function for a different set of plugins. If
P_PluginName_ToShow	After a search in NODE when a matching Object is found, NODE asks the plugin through this function as to how the results are to be displayed to the user.

Databases

Each plugin saves its Objects data under databases (.ndb files). The databases created must be named as PluginName1.ndb, PluginName2.ndb, PluginName3.ndb etc. (PluginName.ndb or PluginName0.ndb will not be read)

The format of the database entry is like this:

<Object=File><Plugin=Files><Scan1=Dropbox.Ink><Scan2=C:\Users\RAJAT\Desktop><Path=C:\Users\RAJAT\Desktop\Dropbox.Ink>

The Object tag will be used to identify the Object this database adds to. The Scan1, Scan2, Scan3 etc tags provide the searchable data to NODE (any other tags are not part of the searchable data).

Please note that amongst the Scan# tags, **Scan1** is the most important, as this will be used to rate results and determine its placement in the list of results. Therefore it is very important to keep Scan1 tag data as short as possible to keep performance optimal.

For the remaining Scan tags, NODE only searches whether the search terms exist in them or not. A plugin can add any number of custom tags (like Path above), if needed.

Recognized Functions

A plugin can have any number of functions that it uses internally. This is the list of function names that will be called by host, and has to be standardized across plugins. Not all functions are mandatory, and the plugin can only contain a few, depending on the functionality it adds to NODE.

Below are the set of functions with explanation and examples.

P_PluginName_Scan()
<p>This function, if found, will be called every time the database is to be refreshed (for example when NODE starts).</p> <p>The function will save the database(s) it creates, in the NODE\db folder (more folder name details later). It must be noted that to improve search performance (and providing results with shorter search queries) it is better to break a possible larger database into a few smaller size ones.</p> <p>A plugin can create a tag for its own use by any name, like the Path tag in this example. This tag will be used by this plugin itself when user selects an Object and Action and the control is transferred back to the plugin.</p>

```

P_ControlP_Scan()
{
    RegExMatch(A_ThisFunc, "U)_(?<PlugName>.*)_", 0)    ;get the name of
this plugin

    FileDelete, %A_ScriptDir%\db\%OplugName%.ndb
    ;add a / at the beginning of Scan tag for top level object - this
places it higher up when searched with a /
    ;create DB with plugin's name + numbers starting with 1, else it
won't be read

    CPIItems =
    (Ltrim
    Access.cpl    Accessibility properties
    Appwiz.cpl    Add-Remove Programs properties
    Desk.cpl      Display properties
    Hdwiz.cpl     Add Hardware properties
    Inetcp1.cpl   Internet properties
    Intl.cpl      Regional Settings properties
    Irprops.cpl   Infrared Port properties
    Joy.cpl       Joystick properties
    Main.cpl      Mouse properties
    Mmsys.cpl     Multimedia properties
    Ncpa.cpl      Network Connections properties
    Nusrmgr.cpl   User Accounts properties
    Nwc.cpl       Gateway Services for NetWare properties
    Odbccp32.cpl  Open Database Connectivity (ODBC) properties
    Powercfg.cpl  Power Options properties
    Sysdm.cpl     System properties
    Telephon.cpl  Phone and Modem Options properties
    Timedate.cpl  Time and Date properties
    )

    Loop, Parse, CPIItems, `n
    {
        IfEqual, A_LoopField,
            continue
        StringSplit, Item, A_LoopField, `t
        IfExist, %A_WinDir%\System32\%Item1%
            FileAppend,
<Object=%OplugName%><Plugin=%OplugName%><Scan1=%Item2%><File=%Item1%>`n,
%A_ScriptDir%\db\%OplugName%1.ndb
    }
}

```

P_PluginName_RefreshTime()

This function, if it exists, simply provides the number of minutes after which the database would be refreshed by calling the **P_PluginName_Scan** Fehler! Verweisquelle konnte nicht gefunden werden.function.

```

P_Files_RefreshTime()
{
    Global File_conf

```

```

    RegExMatch(A_ThisFunc, "U)_(?<PlugName>.*)_", 0)
    IniRead, P_Files_C_RefreshTime, %File_conf%, %OplugName%,
RefreshTime, 60
    Return P_Files_C_RefreshTime
}

```

P_PluginName_ConfGUI(GUI_ID)

If this plugin will have certain user configurable settings, then provide this function. This function receives the GUI_ID of the Configuration window, and a tab will be automatically created by the name of this plugin.

The plugin can add any number of controls to this tab within the below available co-ordinates:
x10 y30 w740 h415

The plugin can have global variables for various GUI controls with naming as P_PluginName_C_anything

```

;add plugin specific settings to own tab in settings
P_Files_ConfGUI(GUI_ID)
{
    Global
File_conf,P_Files_C_FolderList,P_Files_C_TypeList,P_Files_C_RefreshTime,P_Files_C_Editor
    RegExMatch(A_ThisFunc, "U)_(?<PlugName>.*)_", 0)

    Gui, %GUI_ID%:Default
    Gui, Tab, %OplugName%, , Exact

    IniRead, P_Files_C_RefreshTime, %File_conf%, %OplugName%, RefreshTime, 30
    IniRead, P_Files_C_FolderList, %File_conf%, %OplugName%, FolderList,
%A_Space%
    IniRead, P_Files_C_TypeList, %File_conf%, %OplugName%, TypeList, %A_Space%
    IniRead, P_Files_C_Editor, %File_conf%, %OplugName%, Editor,
%A_Windir%\Notepad.exe

    StringReplace, P_Files_C_FolderList, P_Files_C_FolderList, |, `n, A
    StringReplace, P_Files_C_TypeList, P_Files_C_TypeList, |, `n, A
    ;plugin's settings here
    Gui, Add, GroupBox, x10 y35 w390 h260 , Folders
    Gui, Add, Text, x20 y80 w110 h100 , Add one folder per row.`n`neg:`nD:\My
Folder`nC:\Pics`nF:\documents
    Gui, Add, Edit, x140 y65 w250 h220 vP_Files_C_FolderList,
%P_Files_C_FolderList%

    Gui, Add, GroupBox, x420 y35 w320 h260 , Extensions
    Gui, Add, Text, x430 y80 w150 h100 , Add one extension per
row.`n`neg:`njpg`npng`nahk
    Gui, Add, Edit, x600 y65 w70 h220 vP_Files_C_TypeList,
%P_Files_C_TypeList%
    Gui, Add, Text, x20 y315 w100 h30 , Refresh in minutes:
    Gui, Add, Edit, x150 y315 w30 h20 vP_Files_C_RefreshTime,
%P_Files_C_RefreshTime%
    Gui, Add, Text, x20 y355 w100 h30 , Path to Text Editor:
    Gui, Add, Edit, x150 y355 w300 h20 vP_Files_C_Editor, %P_Files_C_Editor%

```

```

    Gui, Add, Text, x480 y315 w260 h130 +Border , Help:`n`nThe Start Menu,
Desktop and My Documents folder are already being scanned.`n`nThe below
extensions are also already added.`nxls, doc, ppt, pdf, docx, xlsx, pptx,
pdf, mp3, wma, ogg
}

```

P_PluginName_ConfInfo(ByRef Author, ByRef Info)

This function will allow Plugin Authors to add their contact information and plugin help in a separate function callable from the Info button in settings

```

P_Files_Info(ByRef Author, ByRef Info)
{
    Author = hoppfrosch <hoppfrosch@ahk4.net>
    Info = Custom Date and Time entries plugin.
}

```

P_PluginName_ConfSave()

This function will autorun when the Configuration window's Save button is pressed. This would ideally save down all the settings in global variables that belong to this plugin.

```

P_Files_ConfSave()
{
    Global
File_Conf,P_Files_C_FolderList,P_Files_C_TypeList,P_Files_C_RefreshTime,P_Files_C_Editor
    RegExMatch(A_ThisFunc, "U)_(?<PlugName>.*)_", O)

    StringReplace, P_Files_C_FolderList, P_Files_C_FolderList, `n, |, A
    StringReplace, P_Files_C_FolderList, P_Files_C_FolderList, ||, |, A
    StringReplace, P_Files_C_FolderList, P_Files_C_FolderList, `, ", A
    StringReplace, P_Files_C_TypeList, P_Files_C_TypeList, `n, |, A
    StringReplace, P_Files_C_TypeList, P_Files_C_TypeList, ||, |, A

    IniWrite, %P_Files_C_FolderList%, %File_conf%, %OplugName%, FolderList
    IniWrite, %P_Files_C_TypeList%, %File_conf%, %OplugName%, TypeList
    IniWrite, %P_Files_C_RefreshTime%, %File_conf%, %OplugName%, RefreshTime
    IniWrite, %P_Files_C_Editor%, %File_conf%, %OplugName%, Editor
}

```

P_PluginName_KeyWords(ByRef PlusWords, ByRef MinusWords)

This function can directly affect the internal algorithm of scoring the results. If the match contains a Plus Word, it is scored up and down for Minus Word.

```

P_Files_KeyWords(ByRef PlusWords, ByRef MinusWords)
{
    ;This function is so that the plugin can directly affect the
internal algorithm of scoring the results
    ;if the match contains a Plus Word, it is scored up and down for
Minus Word.
    MinusWords = uninstall,help,readme,read me,setup
}

```

```

PlusWords = google,bing,6ikipedia,firefox,chrome,internet
explorer,opera,office,word,excel,exe,lnk
}

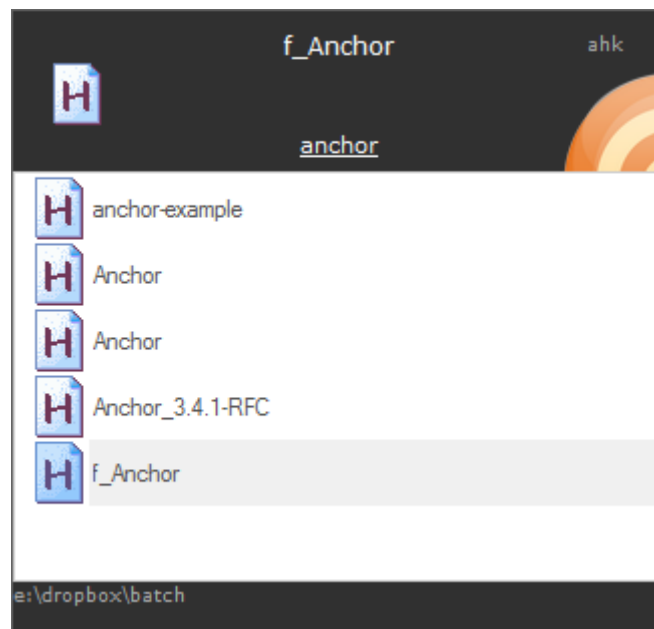
```

P_PluginName_ToShow(SrchStrng, Result, ByRef LVLine, ByRef MainTxt, ByRef SubTxt, ByRef IconIdx, ByRef InfoBar)

After a search in NODE when a matching Object is found, NODE asks the plugin through this function as to how the results are to be displayed to the user.

User's Search String and the matching line from database are received by this function as input and this function returns how the results are to be shown in GUI as main and sub text and icon. Only first 3 are mandatory.

LVLine	The text to be displayed on the ListView row (f_Anchor in example Listview below)
MainTxt	The Main name of result (f_Anchor at top of GUI)
IconIdx	Index of the icon retrieved through GetIcon(FilePath) function
SubTxt	Any secondary text (ahk at top right of GUI)
InfoBar	Another row of descriptive text (path e:\dropbox\batch at bottom of GUI)



;The result from database is provided to this function and this function returns
;how the results are to be shown in GUI as main and sub text and icon
P_Clipboard_ToShow(SrchStrng, Result, **ByRef** LVLine, **ByRef** MainTxt, **ByRef** SubTxt, **ByRef** IconIdx, **ByRef** InfoBar)

```

{
    if !IconIdx
        IconIdx := Int_GetIcon(A_ScriptDir
"\res\icons\Objects\Clipboard.ico")

    MainTxt = /Clipboard
    SubTxt =
    InfoBar =
    LVLine = %MainTxt%
}

```

P_PluginName_Objects()

This function simply returns the pipe separated list of Objects it supports . Only in rarest cases a plugin would support more than one object. Object Name has to be valid variable name.

```

P_Files_Objects()
{
    ;this function simply returns the pipe separated list of Objects
it supports
    ;only in rare cases a plugin would support more than one object

    Objects = File
    Return Objects
}

```

P_PluginName_Actions()

This function returns a list of Actions available to the selected Object. The Actions shown can depend on nature of object or any other variable, as deemed fit by the plugin.

```

P_Screen_Actions(Line)
{
    RegexMatch(A_ThisFunc, "U)_(?<PlugName>.*)_", 0)
    ActionList =
    (LTrim Join `r`n
        <Action=%OPlugName%.Turn Off><Scan1=Turn Off>
        <Action=%OPlugName%.Run Screensaver><Scan1=Run
Screensaver>
    )

    Return ActionList
}

```

P_PluginName_Execute(Line, Action, SrchStrng)

This function is called when the user launches an Object+Action combination in NODE. The

function receives the Object's result line from database, the selected Action and user's search string.

```
P_Screen_Execute(Line, Action, SrchStrng)
{
    Global Hist_DB, File_Conf
    RegExMatch(A_ThisFunc, "U)_(?<PlugName>.*)_", 0)
    UpdateHistory = Y

    IfEqual, Action, %OPlugName%.Turn Off
    {
        Sleep, 500
        WinGet, hwnd, ID, Program Manager
        SendMessage, 0x112, 0xF170, 2,, ahk_id %hwnd%
    }

    IfEqual, Action, %OPlugName%.Run Screensaver
    {
        Sleep, 500
        WinGet, hwnd, ID, Program Manager
        SendMessage, 0x112, 0xF140, 2,, ahk_id %hwnd%
    }

    ;update history
    IfEqual, UpdateHistory, Y
    {
        StringReplace, Hist_DB, Hist_DB, %Line%\r\n,, A
        ;clear history of last occurrence of same line
        NewLine := RegExReplace(Line, "iU)\<Action=.*\>")
        ;remove earlier action

        NewLine = %NewLine%<Action=%Action%>
        Hist_DB = %NewLine%\r\n%Hist_DB% ;add latest item at
        ;beginning of history
    }
}
```

P_PluginName_SearchAnalyze()

This is a special type of function for a different set of plugins. If the plugin's object can be freeform and a database can't be built which can define all the possible sub-objects that are allowed (for example if the plugin can solve mathematical expressions, or post the typed text to Facebook, or search text over internet – then this plugin can make that a recognized object and make available relevant actions. If this function returns a Y, that means a valid input was found by it.

```
P_Calculator_SearchAnalyze(SrchStrng)
{
    ;this function basically tries to analyze the search string and understand
    ;it if it is valid input
    ;for this plugin.
```



```

Match = N
If SrchStrng contains *,-
,/,+,Abs,Ceil,Exp,Floor,Log,Ln,Round,Sqrt,Sin,Cos,Tan,ASin,ACos,ATan,SGN,Fib,
fac
If SrchStrng contains 1,2,3,4,5,6,7,8,9,0
{
    EvalResult := Eval(SrchStrng)
    If EvalResult contains 1,2,3,4,5,6,7,8,9,0
        Match = Y
}
Return Match
}

```

P_PluginName_OnExit()

This is a function that is automatically called when Node exits. This would typically be used for file/memory cleanups. Please ensure that there are not time consuming operations here and no possibility of un-ending loops, which will cause the application to get stuck.

```

P_Files_OnExit()
{
    Global
    DllCall("FreeLibrary", Ptr, P_Files_Int_Storage("FFF1"))
    ; get search library handle from storage and free the library
    GetIcon()      ; uninitialized COM
}

```

Folders

The plugins must utilise the below folders for the along mentioned uses.

NODE\db	To save the database if the plugin creates one
NODE\res\icons\Actions\	To save any icons in 'PluginName.ActionName.ico' format
NODE\res\icons\Objects\	To save any icons in 'ObjectName.ico' format
NODE\tmp	To save any temporary working files