# Messages And Notifications

From Notepad++ Wiki

Jump to: [navigation](http://npp-wiki.tuxfamily.org/index.php?title=Messages_And_Notifications#mw-navigation), [search](http://npp-wiki.tuxfamily.org/index.php?title=Messages_And_Notifications#p-search)

Messages and notifications Notepad++ exchanges with or forwards to plugins for cooperative purposes

### Why messages *and* notifications?

Basically, a message may have a return value, and is usually thought as a query. A notification simply informs of some event, and is more usually thought as a command. However, a notification is brought by a Windows message, WM\_NOTIFY.

While the meaning of the two parameters a message can carry – known as wParam and lParam – depend on the message, things are much clearer for a notification. The WM\_NOTIFY message holds the sender's control ID in its wParam, and a pointer to a notification parameter block in its lParam. The first three integers of the latter are as follows:

* Handle of sender
* Control ID of sender (duplicates wParam for historical reasons)
* Notification code

Extra information, dependent on the notification code, may follow in the parameter block pointed to by lParam. Such information is documented in the Description field, if present.

### Notepad++ messages

|  |  |  |
| --- | --- | --- |
| NPPM\_GETCURRENTSCINTILLA | wParam: | 0 |
| lParam: | [out] int \* currentEdit |
| currentEdit indicates the current Scintilla view :   * 0 is the main Scintilla view * 1 is the second Scintilla view. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETCURRENTLANGTYPE | wParam: | 0 |
| lParam: | [out] int \* langType |
| langType indicates the language type of current Scintilla view document : please see the enum LangType in "PluginInterface.h" for all possible value. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_SETCURRENTLANGTYPE | wParam: | 0 |
| lParam: | [in] int langTypeToSet |
| langTypeToSet is used to set the language type of current Scintilla view document : please see the enum LangType in "PluginInterface.h" for all possible value. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETFULLCURRENTPATH | wParam: | [in] size\_t fullPathLen |
| lParam: | [out] TCHAR \* fullPath |
| fullPath receives the full path of current Scintilla view document. User is responsible to allocate (or use automatic variable) a buffer with an enough size. MAX\_PATH is suggested to use. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETCURRENTDIRECTORY | wParam: | [in] size\_t directoryPathLen |
| lParam: | [out] TCHAR \* directoryPath |
| directoryPath receives the directory path of current Scintilla view document. User is responsible to allocate (or use automatic variable) a buffer with an enough size. MAX\_PATH is suggested to use. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETFILENAME | wParam: | [in] size\_t fileNameLen |
| lParam: | [out] TCHAR \* fileName |
| fileName receives the file name of current Scintilla view document. User is responsible to allocate (or use automatic variable) a buffer with an enough size. MAX\_PATH is suggested to use. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETNAMEPART | wParam: | [in] size\_t namePartLen |
| lParam: | [out] TCHAR \* namePart |
| namePart receives the part of name (without extension) of current Scintilla view document. User is responsible to allocate (or use automatic variable) a buffer with an enough size. MAX\_PATH is suggested to use. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETEXTPART | wParam: | [in] size\_t extensionLen |
| lParam: | [out] TCHAR \* extension |
| extension receives the part of extension of current Scintilla view document. User is responsible to allocate (or use automatic variable) a buffer with an enough size. MAX\_PATH is suggested to use. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETCURRENTWORD | wParam: | [in] size\_t currentWordLen |
| lParam: | [out] TCHAR \* currentWord |
| currentWord receives the word on which cursor is currently of current Scintilla view document. User is responsible to allocate (or use automatic variable) a buffer with an enough size. MAX\_PATH is suggested to use. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETNPPDIRECTORY | wParam: | [in] size\_t nppDirLen |
| lParam: | [out] TCHAR \* nppDir |
| nppDir receives the full path of directory where located Notepad++ binary. User is responsible to allocate (or use automatic variable) a buffer with an enough size. MAX\_PATH is suggested to use. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETNBOPENFILES | wParam: | 0 |
| lParam: | [in] int nbType |
| The return value depends on nbType :   |  |  | | --- | --- | | **nbType** | **Meaning** | | ALL\_OPEN\_FILES | the total number of files opened in Notepad++ | | PRIMARY\_VIEW | the number of files opened in the primary view | | SECOND\_VIEW | the number of files opened in the second view | | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETOPENFILENAMES | wParam: | [out] TCHAR \*\* fileNames |
| lParam: | [in] int nbFile |
| nbFile is the size of your fileNames array. You should get this value by using NPPM\_NBOPENFILES message with constant ALL\_OPEN\_FILES, then allocate fileNames array with this value.  fileNames receives the full path names of all the opened files in Notepad++. User is responsible to allocate fileNames array with an enough size.  The return value is the number of file full path names copied in fileNames array. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETOPENFILENAMESPRIMARY | wParam: | [out] TCHAR \*\* fileNames |
| lParam: | [in] int nbFile |
| nbFile is the size of your fileNames array. You should get this value by using NPPM\_NBOPENFILES message with constant PRIMARY\_VIEW, then allocate fileNames array with this value.  fileNames receives the full path names of the opened files in the primary view. User is responsible to allocate fileNames array with an enough size. The return value is the number of file full path names copied in fileNames array. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETOPENFILENAMESSECOND | wParam: | [out] TCHAR \*\* fileNames |
| lParam: | [in] int nbFile |
| nbFile is the size of your fileNames array. You should get this value by using NPPM\_NBOPENFILES message with constant SECOND\_VIEW, then allocate fileNames array with this value.  fileNames receives the full path names of the opened files in the second view. User is responsible to allocate fileNames array with an enough size.  The return value is the number of file full path names copied in fileNames array. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_DOOPEN | wParam: | 0 |
| lParam: | [in] TCHAR \* fullPathName2Open |
| fullPathName2Open indicates the full file path name to be opened.  The return value is TRUE (1) if the operation is successful, otherwise FALSE (0). | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_MODELESSDIALOG | wParam: | [in] int op |
| lParam: | [in] HWND hDlg |
| For each created dialog in your plugin, you should register it (and unregister while destroy it) to Notepad++ by using this message. If this message is ignored, then your dialog won't react with the key stroke messages such as TAB key. For the good functioning of your plugin dialog, you're recommended to not ignore this message.  hDlg is the handle of the dialog to be registed.  op : the operation mode. MODELESSDIALOGADD is to register; MODELESSDIALOGREMOVE is to unregister. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_SAVECURRENTSESSION | wParam: | 0 |
| lParam: | [in] const TCHAR \*sessionFileName |
| You can save the current opened files in Notepad++ as a group of files (session) by using this message. Notepad++ saves the current opened files' full pathe names and their current stats in a xml file. The xml full path name is provided by sessionFileName. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_SAVESESSION | wParam: | 0 |
| lParam: | [in] sessionInfo sessionInfomation |
| This message let plugins save a session file (xml format) by providing an array of full files path name. sessionInfomation is a structure defined as follows:    TCHAR\* sessionFilePathName;  the full path name of session file to save    int nbFile;  the number of files in the session    TCHAR\*\* files;  files' full path | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETNBSESSIONFILES | wParam: | 0 |
| lParam: | [in] const TCHAR \* sessionFileName |
| This message return the number of files to load in the session sessionFileName. sessionFileName should be a full path name of an xml file. 0 is returned if sessionFileName is NULL or an empty string | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETSESSIONFILES | wParam: | [out] TCHAR \*\* sessionFileArray |
| lParam: | [in]const TCHAR \* |
| Send this message to get files' full path name from a session file.  sessionFileName is the session file from which you retrieve the files.  sessionFileArray : the array in which the files' full path of the same group are written. You should send message NPPM\_GETNBSESSIONFILES before to allocate this array with the proper size. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_LOADSESSION | wParam: | 0 |
| lParam: | [in] const TCHAR \* sessionFileName |
| Open all files of same session in Notepad++ via a xml format session file sessionFileName. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_CREATESCINTILLAHANDLE | wParam: | 0 |
| lParam: | [in] HWND pluginWindowHandle |
| A plugin can create a Scintilla for its usage by sending this message to Notepad++. The return value is created Scintilla handle. The handle should be destroyed by NPPM\_DESTROYSCINTILLAHANDLE message while exit the plugin. If pluginWindowHandle is set (non NULL), it will be set as parent window of this created Scintilla handle, otherwise the parent window is Notepad++. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_DESTROYSCINTILLAHANDLE | wParam: | 0 |
| lParam: | [in] HWND scintillaHandle2Destroy |
| If plugin called NPPM\_DESTROYSCINTILLAHANDLE to create a Scintilla handle, it should call this message to destroy this handle while it exit. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETCURRENTDOCINDEX | wParam: | 0 |
| lParam: | [in] int iView |
| Sending this message to get the current index in the view that you indicates in iView : MAIN\_VIEW or SUB\_VIEW. Returned value is -1 if the view is invisible (hidden), otherwise is the current index. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_ACTIVATEDOC | wParam: | [in] int iView |
| lParam: | [in] int index2Activate |
| When Notepad++ receives this message, it switches to iView (MAIN\_VIEW or SUB\_VIEW) as current view, then it switches to index2Activate from the current document. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETMENUHANDLE | wParam: | [in] int whichMenu |
| lParam: | 0 |
| This message help plugins to get the plugins menu handle of Notepad++, whichMenu must be NPPPLUGINMENU (0), or NPPMAINMENU (1) to return handle to the menu br in the main window. 0 is return on any other inut. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_RELOADFILE | wParam: | [in] BOOL withAlert |
| lParam: | [in] TCHAR \*filePathName2Reload |
| This Message reloads the file indicated in filePathName2Reload. If withAlert is TRUE, then an alert message box will be launched. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_SWITCHTOFILE | wParam: | 0 |
| lParam: | [in] TCHAR \*filePathName2switch |
| When this message is received, Notepad++ switches to the document which matches with the given filePathName2switch. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETWINDOWSVERSION | wParam: | 0 |
| lParam: | 0 |
| The return value is windows version of enum winVer. The possible value is WV\_UNKNOWN, WV\_WIN32S, WV\_95, WV\_98, WV\_ME, WV\_NT, WV\_W2K, WV\_XP, WV\_S2003, WV\_XPX64 and WV\_VISTA | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_SAVECURRENTFILE | wParam: | 0 |
| lParam: | 0 |
| Send this message to Notepad++ to save the current document. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_SAVEALLFILES | wParam: | 0 |
| lParam: | 0 |
| Send this message to Notepad++ to save all opened document. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETPLUGINSCONFIGDIR | wParam: | [in] int strLen |
| lParam: | [out] TCHAR \*pluginsConfDir |
| pluginsConfDir receives the directory path of plugin config files. User is responsible to allocate (or use automatic variable) a buffer with an enough size. MAX\_PATH is suggested to use. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_SETMENUITEMCHECK | wParam: | [in] int cmdID |
| lParam: | [in] BOOL doCheck |
| Use this message to set/remove the check on menu item. cmdID is the command ID which corresponds to the menu item. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_LAUNCHFINDINFILESDLG | wParam: | [in] TCHAR \* dir2Search |
| lParam: | [in] TCHAR \* filtre |
| This message triggers the Find in files dialog. The fields Directory and filters are filled by respectively dir2Search and filtre if those parameters are not NULL or empty. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_DMMREGASDCKDLG | wParam: | 0 |
| lParam: | [in] tTbData \* dockingData |
| From v4.0, Notepad++ supports the dockable dialog feature for the plugins. This message passes the necessary data dockingData to Notepad++ in order to make your dialog dockable. Here is tTbData looks like this:    HWND hClient;  your dockable dialog handle.    TCHAR \*pszName;  the name of your plugin dialog.    int dlgID;  index of menu entry where the dialog in question will be triggered.    UINT uMask;  contains the behaviour informations of your dialog. It can be one of the following value : DWS\_DF\_CONT\_LEFT, DWS\_DF\_CONT\_RIGHT, DWS\_DF\_CONT\_TOP, DWS\_DF\_CONT\_BOTTOM and DWS\_DF\_FLOATING combined (optional) with DWS\_ICONTAB, DWS\_ICONBAR and DWS\_ADDINFO.    HICON hIconTab;  handle to the icon to display on the dialog's tab    TCHAR \*pszAddInfo;  pointer to a string joined to the caption using " - ", if not NULL    RECT rcFloat;  Used internally, do not set    int iPrevCont;  Used internally, do not set    const TCHAR \*pszModuleName;  the name of your plugin module (with extension .dll).  Minimum informations you need to fill out before sending it by NPPM\_DMMREGASDCKDLG message is hClient, pszName, dlgID, uMask and pszModuleName. Notice that rcFloat and iPrevCont shouldn't be filled. They are used internally. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_DMMSHOW | wParam: | 0 |
| lParam: | [in] HWND hDlg |
| This message is used for your plugin's dockable dialog. Send this message to show the dialog. hDlg is the handle of your dialog to be shown. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_DMMHIDE | wParam: | 0 |
| lParam: | [in] HWND hDlg |
| This message is used for your plugin's dockable dialog. Send this message to hide the dialog. hDlg is the handle of your dialog to be hidden. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_DMMUPDATEDISPINFO | wParam: | 0 |
| lParam: | [in] HWND hDlg |
| This message is used for your plugin's dockable dialog. Send this message to update (redraw) the dialog. hDlg is the handle of your dialog to be updated. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_DMMGETPLUGINHWNDBYNAME | wParam: | [in] const TCHAR \* windowName |
| lParam: | [in] const TCHAR \* moduleName |
| This message returns the dialog handle corresponds to the windowName and moduleName. You may need this message if you want to communicate with another plugin "dockable" dialog, by knowing its name and its plugin module name. If moduleName is NULL, then return value is NULL. If windowName is NULL, then the first found window handle which matches with the moduleName will be returned. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_MSGTOPLUGIN | wParam: | [in] TCHAR \* destModuleName |
| lParam: | [in][out] CommunicationInfo \* info |
| This message allows the communication between 2 plugins.  For example, plugin X can execute a command of plugin Y if plugin X knows the command ID and the file name of plugin Y. destModuleName is the complete module name (with the extesion .dll) of plugin with which you want to communicate (plugin Y). communicationInfo is a poniter of structure type :    long internalMsg;  an integer defined by plugin Y, known by plugin X, identifying the message being sent.    TCHAR \* srcModuleName;  the complete module name (with the extesion .dll) of caller(plugin X).    void \* info;  defined by plugin, the informations to be exchanged between X and Y. It's a void pointer so it should be defined by plugin Y and known by plugin X.  The returned value is TRUE if Notepad++ found the plugin by its module name (destModuleName), and pass the info (communicationInfo) to the module. The returned value is FALSE if no plugin with such name is found. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_MENUCOMMAND | wParam: | 0 |
| lParam: | [in] int commandID |
| This message allows plugins to call all the Notepad++ menu commands.  commandID are the command ID used in Notepad++. All the command ID are defined in menuCmdID.h.` | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_TRIGGERTABBARCONTEXTMENU | wParam: | [in] int whichView |
| lParam: | [in] int index2Activate |
| This message switches to iView (MAIN\_VIEW or SUB\_VIEW) as current view, and it switchs to index2Activate from the current document. Finally it triggers the tabbar context menu for the current document. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETNPPVERSION | wParam: | 0 |
| lParam: | 0 |
| You can get Notepad++ version via this message. The return value is made up of 2 parts : the major version (the high word) and minor version (the low word). For example, the 4.7.5 version will be :    HIWORD(version) == 4   LOWORD(version) == 75 Note that this message is supported by the v4.7 or higher version. Earlier versions return 0. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_HIDETABBAR | wParam: | 0 |
| lParam: | [in] BOOL hideOrNot |
| If hideOrNot == TRUE, then this message will hide the tabbar, otherwise it makes tabbar shown. The returned value is the previous status before this operation. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_ISTABBARHIDDEN | wParam: | 0 |
| lParam: | 0 |
| By sending this message, a plugin is able to tell the current status of tabbar from the returned value: TRUE if the tabbar is hidden, FALSE otherwise. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_HIDETOOLBAR | wParam: | 0 |
| lParam: | [in] BOOL hideOrNot |
| If hideOrNot == TRUE, then this message will hide the toolbar, otherwises it makes tabbar shown. The returned value is the previous staus before this operation. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_ISTOOLBARHIDDEN | wParam: | 0 |
| lParam: | 0 |
| Via this message plugin is able to know the current status of toolbar.  TRUE if the toolbar is hidden, FALSE otherwise. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_HIDEMENU | wParam: | 0 |
| lParam: | [in] BOOL hideOrNot |
| If hideOrNot == TRUE, then this message will hide the menu bar, otherwises it makes tabbar shown. The returned value is the previous staus before this operation. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_ISMENUHIDDEN | wParam: | 0 |
| lParam: | 0 |
| Via this message plugin is able to know the current status of menu bar.  TRUE if the menbar is hidden, FALSE otherwise. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_HIDESTATUSBAR | wParam: | 0 |
| lParam: | [in] BOOL hideOrNot |
| If hideOrNot == TRUE, then this message will hide the status bar, otherwises it makes tabbar shown. The returned value is the previous staus before this operation. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_ISSTATUSBARHIDDEN | wParam: | 0 |
| lParam: | 0 |
| Via this message plugin is able to know the current status of status bar.  TRUE if the status bar is hidden, FALSE otherwise. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETSHORTCUTBYCMDID | wParam: | [in] int cmdID |
| lParam: | [out] ShortcutKey \* sk |
| Get your plugin command current mapped shortcut into sk via cmdID. You may need it after getting NPPN\_READY notification.  Returned value : TRUE if this function call is successful and shorcut is enable, otherwise FALSE | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETPOSFROMBUFFERID | wParam: | [in] int bufferID |
| lParam: | 0 |
| Get 0-based document position from given buffer ID, which is held in the 30 lowest bits of the return value on success.  If bufferID doesn't exist, -1 is returned. Otherwise, the index part is valid, and bit 30 indicates which view has the buffer (clear for main view, set for sub view). | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETFULLPATHFROMBUFFERID | wParam: | [in] int bufferID |
| lParam: | [out] TCHAR \* fullFilePath |
| Get full path file name from a given buffer ID.  Return -1 if the bufferID non existing, otherwise the number of TCHAR copied/to copy. User should call it with fullFilePath be NULL to get the number of TCHAR (not including the nul character), allocate fullFilePath with the return values + 1, then call it again to get full path file name | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETBUFFERIDFROMPOS | wParam: | [in] int position |
| lParam: | [in] int view |
| Get document buffer ID from given position.  position is 0 based index, view should be MAIN\_VIEW or SUB\_VIEW. Return value : 0 if given position is invalid, otherwise the document buffer ID. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETCURRENTBUFFERID | wParam: | 0 |
| lParam: | 0 |
| Returns active document buffer ID | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_RELOADBUFFERID | wParam: | [in] int bufferID |
| lParam: | [in] BOOL doAlertOrNot |
| Reload the document by given buffer ID.  if doAlertOrNot is TRUE, then a message box will display to ask user to reload the document, otherwise document will be loaded without asking user. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETBUFFERLANGTYPE | wParam: | 0 |
| lParam: | 0 |
| Get document's language type from given buffer ID.  Returns value : if error -1, otherwise language type (see LangType). [in] int bufferID | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_SETBUFFERLANGTYPE | wParam: | [in] int bufferID |
| lParam: | [in] LangType langType2Set |
| Set language type of given buffer ID's document.  Returns TRUE on success, FALSE otherwise. L\_USER and L\_EXTERNAL are not supported. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETBUFFERENCODING | wParam: | [in] int bufferID |
| lParam: | 0 |
| Get document's encoding from given buffer ID.  Returns value : if error -1, otherwise encoding number. enum UniMode - uni8Bit 0, uniUTF8 1, uni16BE 2, uni16LE 3, uniCookie 4, uni7Bit 5, uni16BE\_NoBOM 6, uni16LE\_NoBOM 7 | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_SETBUFFERENCODING | wParam: | [in] int bufferID |
| lParam: | [in] UniMode encoding |
| Set given buffer ID's document encoding.  Can only be done on new, unedited files | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETBUFFERFORMAT | wParam: | [in] int bufferID |
| lParam: | 0 |
| Get document's format from given buffer ID.  Returns value : if error -1, otherwise document's format (see formatType). | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_SETBUFFERFORMAT | wParam: | [in] int bufferID |
| lParam: | [in] formatType format |
| Set format of given buffer ID's document.  Returns TRUE on success, FALSE otherwise | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETCURRENTLINE | wParam: | 0 |
| lParam: | 0 |
| Returns the caret current position 0-based line | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETCURRENTCOLUMN | wParam: | 0 |
| lParam: | 0 |
| Returns the caret current position 0-based column | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_SAVECURRENTFILEAS | wParam: | [in] 0 to Save As, 1 to Save a Copy As |
| lParam: | [in] TCHAR\* filename |
| Performs a Save As (wParam == 0) or Save a Copy As (wParam == 1) on the current buffer, outputting to *filename*. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETCURRENTNATIVELANGENCODING | wParam: | 0 |
| lParam: | 0 |
| Returns the code page associated with the current localisation of Notepad++. As of v6.6.6, returned values are 1252 (ISO 8859-1), 437 (OEM US) or 950 (Big5). | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_ALLOCATESUPPORTED | wParam: | 0 |
| lParam: | 0 |
| Returns TRUE if NPPM\_ALLOCATECMDID is supported. Use it to identify if subclassing is necessary. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_ALLOCATECMDID | wParam: | [in] Requested number of IDs |
| lParam: | [out] Pointer to allocated range as an int |
| Allows a plugin to obtain a number of consecutive meni item IDs for creating menus dynamically, with the guarantee of these IDs not clashing with any other plugin's. Returns 0 on failure, nonzero on success. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_ALLOCATEMARKER | wParam: | [in] Requested number of IDs |
| lParam: | [out] Pointer to allocated range as an int |
| Allows a plugin to obtain a number of consecutive arker IDs dynamically, with the guarantee of these IDs not clashing with any other plugin's. Returns 0 on failure, nonzero on success. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETLANGUAGENAME | wParam: | [in] The LangType identifier - an int actually |
| lParam: | [in/out] TCHAR\* Pointer to language name string. |
| Returns the number of characters needed or copied. If lParam is null, size of the language name is copied. Use this to allocate a buffer to pass as lParam and get the language name copied therein. The terminating \0 isn't counted in the returned length. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETLANGUAGEDESC | wParam: | [in] The LangType identifier - an int actually |
| lParam: | [in/out] TCHAR\* Pointer to language description string. |
| Returns the number of characters needed or copied. If lParam is null, size of the language name is copied. Use this to allocate a buffer to pass as lParam and get the language description copied therein. The terminating \0 isn't counted in the returned length. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_SHOWDOCSWITCHER | wParam: | 0 |
| lParam: | [in] BOOL showItIfTrue |
| Shows document switcher if lparam is true, hide it if false. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_ISDOCSWITCHERSHOWN | wParam: | 0 |
| lParam: | 0 |
| Returns 0 if the document switcher is not currently shown, else non zero. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETAPPDATAPLUGINSALLOWED | wParam: | 0 |
| lParam: | 0 |
| Returns true if loading plugins from %APPDATA is allowed, else returns false. | | |

|  |  |  |
| --- | --- | --- |
| NPPM\_GETGETCURRENTVIEW | wParam: | 0 |
| lParam: | 0 |
| Returns 0 when primary view is active, and 1 instead if secondary view is. | | |

### Notepad++ notifications

|  |  |  |
| --- | --- | --- |
| Notepad++ notifications | | |
| **Notification code** | **hwndFrom** | **idFrom** |
| NPPN\_READY | |  |  | | --- | --- | | hwndNpp | 0 | | Notifies plugins that all the procedures of launching notepad++ completed succesfully. | | | |
| NPPN\_TBMODIFICATION | |  |  | | --- | --- | | hwndNpp | 0 | | Notifies plugins that toolbar icons can be registered. | | | |
| NPPN\_FILEBEFORECLOSE | |  |  | | --- | --- | | hwndNpp | 0 | | Notifies plugins that the current file is about to be closed | | | |
| NPPN\_FILECLOSED | |  |  | | --- | --- | | hwndNpp | 0 | | Notifies plugins that the current file is just closed | | | |
| NPPN\_FILEBEFOREOPEN | |  |  | | --- | --- | | hwndNpp | 0 | | Notifies plugins that a file is being opened | | | |
| NPPN\_FILEOPENED | |  |  | | --- | --- | | hwndNpp | 0 | | Notifies plugins that the current file just opened | | | |
| NPPN\_FILEBEFORESAVE | |  |  | | --- | --- | | hwndNpp | 0 | | Notifies plugins that the current file is about to be saved | | | |
| NPPN\_FILESAVED | |  |  | | --- | --- | | hwndNpp | 0 | | Notifies plugins that the current file wass just saved | | | |
| NPPN\_SHUTDOWN | |  |  | | --- | --- | | hwndNpp | 0 | | Notifies plugins that Notepad++ is about to shut down. | | | |
| NPPN\_BUFFERACTIVATED | |  |  | | --- | --- | | hwndNpp | activatedBufferID | | Ntifies plugins that a buffer was activated (put to foreground). | | | |
| NPPN\_LANGCHANGED | |  |  | | --- | --- | | hwndNpp | currentBufferID | | Notifies plugins that the language in the current doc just changed. | | | |
| NPPN\_WORDSTYLESUPDATED | |  |  | | --- | --- | | hwndNpp | currentBufferID | | Notifies plugins that user initiated a WordStyleDlg change. | | | |
| NPPN\_SHORTCUTREMAPPED | |  |  | | --- | --- | | ShortcutKeyStructurePointer | cmdID | | Notifies plugins that a plugin command shortcut is remapped.  ShortcutKeyStructurePointer is type ShortcutKey \*, cmdID has type int. | | | |
| NPPN\_FILEBEFORELOAD | |  |  | | --- | --- | | hwndNpp | 0 | | Notifies plugins that the current file is about to be loaded | | | |
| NPPN\_FILELOADFAILED | |  |  | | --- | --- | | hwndNpp | bufferID | | Notifies plugins that file open operation failed | | | |
| NPPN\_DOCORDERCHANGED | |  |  | | --- | --- | | newIndex | bufferID | | Notifies plugins that document order is changed, bufffer *bufferID* having index *newIndex*. | | | |

Retrieved from "<http://npp-wiki.tuxfamily.org/index.php?title=Messages_And_Notifications&oldid=6369>"