Win32 多言語 IME API

Win32 Multilingual IME Application Programming Interface

Version 1.41

日本語訳：片山博文 MZ

翻訳日：2016 年 4 月 14 日

この文書は、入力方式エディタ（input method editor; IME）開発の API リファレンスである。以下の関数は IME が使うものとして意図されている。

IMM UI 関数たち

以下は、UI ウィンドウからアクセス可能な入力方式管理（input method manager; IMM）関数たちである。これらは、IME 状態を変化させるためにアプリからも使うことができる。

- ImmGetCompositionWindow
- ImmSetCompositionWindow
- ImmGetCandidateWindow
- ImmSetCandidateWindow
- ImmGetCompositionString
- ImmSetCompositionString
- ImmGetCompositionFont
- ImmSetCompositionFont
- ImmGetNumCandidateList
- ImmGetCandidateList
- ImmGetGuideLine
- ImmGetConversionStatus
- ImmGetConversionList
- ImmGetOpenStatus
- ImmSetConversionStatus
- ImmSetOpenStatus
- ImmNotifyIME
- ImmCreateSoftKeyboard
- ImmDestroySoftkeyboard
- ImmShowSoftKeyboard

これらの関数に関しては、Platform SDK にある IME 関数たちを参照されたい。

IMM サポート関数たち

以下のトピックは、IME によってサポートされて使われる、IMM 関数たちである。

ImmGenerateMessage 関数

IME は hIMC の hWnd にメッセージを送信するために ImmGenerateMessage 関数を使う。送信されたメッセージは hIMC の hMsgBuf に格納される。

```
BOOL WINAPI
ImmGenerateMessage(HIMC hIMC)
```

引数

引数名	説明
HIMC	hMsgBufを所有する入力コンテキストのハンドル。

戻り値

関数が成功すれば戻り値は TRUE である。さもなければ FALSE になる。

コメント

これは汎用の関数だ。IMM の ImmNotifyIME を通じてコンテキストの更新について通知されるときには IME はたいていこの関数を使う。この場合、IME がアプリにメッセージを提供する必要があるときでも、アプリのメッセージキューにキーストロークはない。IME UI は、UI の見た目だけを更新したいときにこの関数を使うべきではない。IME が更新された入力コンテキストについて IME に伝えられたときは、IME UI が更新されているべきである。キーストロークがなく、変更をアプリに通知する必要があるときで、IME が入力コンテキストを変更したときでのみ、この関数を使うことが推奨される。

ImmRequestMessage 関数

ImmRequestMessage 関数は、アプリに WM_IME_REQUEST メッセージを送信するのに使われる。

```
LRESULT WINAPI ImmRequestMessage(
      HIMC   hIMC,
      WPARAM wParam,
      LPARAM lParam)
```

引数

引数名	説明
hIMC	ターゲットの入力コンテキストのハンドル。
wParam	WM_IME_REQUEST メッセージの wParam。
lParam	WM_IME_REQUEST メッセージの lParam。

戻り値

戻り値は、WM_IME_REQUEST メッセージの戻り値。

コメント

この関数は、Windows 98 および Window 2000 で新しく導入されたもので、IME によって WM_IME_REQUEST をアプリへ送信するのに使われる。IME は、候補ウィンドウやコンポジションウィンドウの位置を定義するときに、アプリからガイドラインを取得したいかもしれない。しかし、IME を完全に意識したアプリ（インライン）では、アプリはたいていコンポジションウィンドウの位置をセットしない。IME がアプリにリクエストをするとき、アプリは WM_IME_REQUEST を受け取る。IME は ImmRequestMessage 関数の呼び出しにより、リクエストをアプリに送るべきであり、SendMessage を呼ぶべきではない。

次は、ImmRequestMessage 関数を通じて IME がアプリに送ることができるサブメッセージのリストである:

- IMR_COMPOSITIONWINOW
- IMR_CANDIDATEWINDOW
- IMR_COMPOSITIONFONT
- IMR_RECONVERTSTRING
- IMR_CONFIRMRECONVERTSTRING
- IMR_QUERYCHARPOSITION
- IMR_DOCUMENTFEED

それらのメッセージの情報については、Platform SDK の入力方式エディタ関数を参照されたい。

HIMC および HIMCC の管理関数たち

以下のトピックは、HIMC と HIMCC の管理関数たちである。

ImmLockIMC 関数

ImmLockIMC 関数は、IMC に対するロックカウントを1だけ増やす。IME が INPUTCONTEXT を参照する必要があるとき、IME は INPUTCONTEXT 構造体のポインタを取得するためにこの関数を呼び出す。

```
LPINPUTCONTEXT WINAPI ImmLockIMC(HIMC hIMC)
```

引数

引数名	説明
hIMC	入力コンテキストのハンドル。

戻り値

関数が成功すれば、INPUTCONTEXT 構造体へのポインタを返す。さもなければ NULL を返す。

ImmUnlockIMC 関数

ImmUnlockIMC 関数は、IMC に対するロックカウントを1だけ減らす。

```
BOOL WINAPI ImmUnlockIMC(HIMC hIMC)
```

引数

引数名	説明
hIMC	入力コンテキストのハンドル。

戻り値

ロックカウントがゼロになったら、戻り値は FALSE である。さもなければ戻り値は TRUE である。

ImmGetIMCLockCount 関数

ImmGetIMCLockCount 関数は、IMC のロックカウントを取得するのに使われる。

```
HIMCC WINAPI ImmGetIMCLockCount(HIMC hIMC)
```

引数

引数名	説明
hIMC	入力コンテキストのハンドル。

戻り値

関数が成功すれば、戻り値は IMC のロックカウントである。さもなければ NULL である。

ImmCreateIMCC 関数

ImmCreateIMCC 関数は、IMC のメンバーとしての新しいコンポーネントを作成する。

```
HIMCC WINAPI ImmCreateIMCC(DWORD dwSize)
```

引数

引数名	説明
dwSize	新しい IMC コンポーネントのサイズ。

戻り値

関数が成功すれば、戻り値は IMC コンポーネントのハンドル（HIMCC）である。さもなければ、NULL である。

コメント

この関数で作成された IMC コンポーネントは、ゼロで初期化される。

ImmDestroyIMCC 関数

ImmDestroyIMCC 関数は、IMC のメンバーとして作成された IME コンポーネントを破棄するために IME によって使われる。

```
HIMCC WINAPI ImmDestroyIMCC(HIMCC hIMCC)
```

引数

引数名	説明
hIMCC	IMC コンポーネントのハンドル。

戻り値

関数が成功すれば戻り値は NULL。さもなければ戻り値は、hIMCC に等しい。

ImmLockIMCC 関数

ImmLockIMCC 関数は、IMC のメンバーとして作成された IMC コンポーネントへのポインタを取得するために IME によって使われる。ImmLockIMC は、IMCC に対するロックカウントを1だけ増加させる。

```
LPVOID WINAPI ImmLockIMCC(HIMCC hIMCC)
```

引数

引数名	説明
hIMCC	IMC コンポーネントのハンドル。

戻り値

関数が成功すれば、戻り値は、IMC コンポーネントへのポインタ。さもなければ戻り値は NULL である。

ImmUnlockIMCC 関数

ImmUnlockIMC 関数は、IMCC に対するロックカウントを1だけ減少させる。

```
BOOL WINAPI ImmUnlockIMCC(HIMCC hIMCC)
```

引数

引数名	説明
hIMCC	IMC コンポーネントのハンドル。

戻り値

もしロックカウントがゼロになれば、戻り値は FALSE である。さもなければ戻り値は TRUE である。

ImmReSizeIMCC 関数

ImmReSizeIMCC 関数は、コンポーネントのサイズを変更する。

```
HIMCC WINAPI ImmReSizeIMCC(
      HIMCC hIMCC,
      DWORD dwSize)
```

引数

引数名	説明
hIMCC	IMC コンポーネントのハンドル。
dwSize	IMC コンポーネントの新しいサイズ。

戻り値

関数が成功すれば、戻り値は HIMCC の新しい値である。さもなければ戻り値は NULL である。

ImmGetIMCCSize 関数

ImmGetIMCCSize 関数は、IMCC のサイズを取得するのに使われる。

```
DWORD WINAPI ImmGetIMCCSize(HIMCC hIMCC)
```

引数

引数名	説明
hIMCC	IMC コンポーネントのハンドル。

戻り値

IMCC のサイズ。

ImmGetIMCCLockCount 関数

ImmGetIMCCLockCount 関数は、IMCC のロックカウントを取得するのに使われる。

```
DWORD WINAPI ImmGetIMCCLockCount(HIMCC hIMCC)
```

引数

引数名	説明
hIMCC	IMC コンポーネントのハンドル。

戻り値

関数が成功すれば、戻り値は IMCC のロックカウントである。さもなければ、戻り値はゼロである。

IME ホットキーとホットキー関数たち

IME ホットキーは、IME の入力モードを変更したり、IME を切り替えたりするのに使われる。IME を直接切り替えるのに使われた IME ホットキーは、直接切り替えホットキー（direct switching hot key）と呼ばれる。

直接切り替えホットキーは、IME_HOTKEY_DSWITCH_FIRST から IME_HOTKEY_DSWITCH_LAST までの範囲になる。それは IME や末端ユーザーがそのようなホットキーがほしいときに、IME やコントロールパネルによって登録される。IME ホットキーは、どの IME がアクティブかに関わらず、すべての IME において効率がいい。

IMM において、いくつかの定義済みホットキー機能が存在する。IMM はそれ自体がそれらのホットキー関数の機能（異なる扱いルーチン）を提供する。すべてのホットキー機能は、IMM において異なるホットキー ID を持ち、それぞれの ID は、それぞれの国の特定の必要性に応じた機能を所有する。アプリが別の定義済みホットキー ID をシステムに追加できないことに注意しておく。

以下は定義済みホットキー識別子である。

ホットキー ID	説明
IME_CHOTKEY_IME_NONIME_TOGGLE	中国語簡体字エディションに対するホットキー。このホットキーは IME と非 IME を切り替える。

ホットキー ID	説明
IME_CHOTKEY_SHAPE_TOGGLE	中国語簡体字エディションに対するホットキー。このホットキーは IME の変換モードを切り替える。
IME_CHOTKEY_SYMBOL_TOGGLE	中国語簡体字エディションに対するホットキー。このホットキーは、IME のシンボル変換モードを切り替える。シンボルモードは、ユーザーが中国語の句読点とシンボル(全角文字)を、キーボードの句読点とシンボルキーストロークを対応付けることで入力できることを示す。
IME_JHOTKEY_CLOSE_OPEN	日本語エディションに対するホットキー。このホットキーは、閉じた状態と開いた状態を切り替える。
IME_THOTKEY_IME_NONIME_TOGGLE	中国語繁体字に対するホットキー。このホットキーは IME と非 IME を切り替える。
IME_THOTKEY_SHAPE_TOGGLE	中国語繁体字に対するホットキー。このホットキーは、IME シェイプ変換モードを切り替える。
IME_THOTKEY_SYMBOL_TOGGLE	中国語繁体字に対するホットキー。このホットキーは IME のシンボル変換モードを切り替える。

この他の種類のホットキーは、IME のプライベートなホットキーであるが、この種類のホットキーに対する機能は存在しない。それはホットキーの値に対する単なるプレースホルダーである。IME は ImmGetHotKey を呼び出すことにより、その値を取得できる。もし IME が1つのホットキー ID に対するこの機能をサポートしたら、このキー入力が検出されるたびに機能を行うだろう。

以下は、現在定義済みのプライベートな IME ホットキー ID である。

ホットキー ID	説明
IME_ITHOTKEY_RESEND_RESULTSTR	中国語繁体字エディションに対するホットキー。このホットキーは、IME がアプリへ以前の結果文字列を再送信するように誘導すべきである。もし IME がこのホットキーが押されていることを検出したら、このアプリへ以前の結果文字列を再送信すべきである。
IME_ITHOTKEY_PREVIOUS_COMPOSITION	中国語繁体字エディションに対するホットキー。このホットキーは IME が以前のコンポジション文字列をアプリへ提出するように誘導すべきである。
IME_ITHOTKEY_UISTYLE_TOGGLE	中国語繁体字エディションに対するホットキー。このホットキーはキャレット関連の UI とキャレットに関係しない UI の UI スタイルを切り替えるように IME UI を誘導すべきである。
IME_ITHOTKEY_RECONVERTSTRING	中国語繁体字エディションに対するホットキー。このホットキーは、IME に再変換を行うように誘導すべきである。これは Windows 98 と Windows 2000 の新しい ID である。

ImmGetHotKey 関数

ImmGetHotKey 関数は、IME ホットキーの値を取得する。

```
BOOL WINAPI ImmGetHotKey(
      DWORD  dwHotKeyID,
      LPUINT lpuModifiers,
      LPUINT lpuVKey,
      LPHKL  lphKL)
```

引数

引数名	説明
dwHotKeyID	ホットキー識別子。
lpuModifiers	ホットキーの組み合わせキー。ALT (MOD_ALT)、CTRL (MOD_CONTROL)、SHIFT (MOD_SHIFT)、左側 (MOD_LEFT)、右側 (MOD_RIGHT)を含む。キーアップフラグ (MOD_ON_KEYUP) はホットキーがキーの上がっているとき有効であることを示す。モディファイアー無視フラグ (MOD_IGNORE_ALL_MODIFIER) は、モディファイアーの組み合わせがホットキーマッチングにおいて無視されることを示す。
lpuVKey	このホットキーの仮想キーコード。
lphKL	IME の HKL。もしこの引数の戻り値が NULL でなければ、ホットキーは、この HKL の IME に切り替えることができる。

戻り値

関数が成功すれば、戻り値は TRUE。さもなければ戻り値は FALSE である。

コメント

この関数はコントロールパネルから呼ばれる。

ImmSetHotKey 関数

ImmSetHotKey 関数は、IME ホットキーの値をセットする。

```
BOOL WINAPI ImmSetHotKey(
      DWORD dwHotKeyID,
      UINT  uModifiers,
      UINT  uVKey,
      HKL   hKL)
```

引数

引数名	説明
dwHotKeyID	ホットキーの識別子。
uModifiers	ホットキーの組み合わせキー。ALT (MOD_ALT)、CTRL (MOD_CONTROL)、SHIFT (MOD_SHIFT)、左側 (MOD_LEFT)、右側 (MOD_RIGHT)を含む。キーアップフラグ (MOD_ON_KEYUP) はホットキーがキーの上がっているとき有効であることを示す。モディファイアー無視フラグ (MOD_IGNORE_ALL_MODIFIER) は、モディファイアーの組み合わせがホットキーマッチングにおいて無視されることを示す。

引数名	説明
uVKey	このホットキーの仮想キーコード。
hKL	IME の HKL。この引数が指定されると、ホットキーはこの HKL の IME に切り替えることができる。

戻り値

関数が成功すれば、戻り値は TRUE。さもなければ戻り値は FALSE である。

コメント

この関数は、コントロールパネルから呼ばれる。特定のキーボードの側を指定しないキーについては uModifiers は両側（MOD_LEFT | MODE_RIGHT）を指定すべきだ。

IMM ソフトキーボード関数たち

以下のトピックは、ソフトキーボードを操作するために IME によって使われる IMM 関数たちである。

ImmCreateSoftKeyboard 関数

ImmCreateSoftKeyboard 関数は、ソフトキーボードウィンドウの一種を作成する。

```
HWND WINAPI ImmCreateSoftKeyboard(
      UINT uType,
      UINT hOwner,
      int x, int y)
```

引数

<table>
<tr><th>引数名</th><th>説明</th></tr>
<tr><td>uType</td><td>ソフトキーボードの種類を指定する。
<table>
<tr><th>uType</th><th>説明</th></tr>
<tr><td>SOFTKEYBOARD_TYPE_T1</td><td>タイプ T1 ソフトキーボード。この種類のソフトキーボードは、IMC_SETSOFTKBDDATA によって更新されるべきだ。</td></tr>
<tr><td>SOFTKEYBOARD_TYPE_C1</td><td>タイプ C1 ソフトキーボード。この種類のソフトキーボードは 2 セットの 256 ワード配列データつきの IMC_SETSOFTKBDDATA によって更新されるべきだ。最初のセットは非シフト状態、二番目はシフト状態である。</td></tr>
</table></td></tr>
<tr><td>hOwner</td><td>ソフトキーボードの所有者を指定する。これは UI ウィンドウでなければならない。</td></tr>
<tr><td>x</td><td>ソフトキーボードの初期水平位置を指定する。</td></tr>
<tr><td>y</td><td>ソフトキーボードの初期垂直位置を指定する。</td></tr>
</table>

戻り値

この関数は、ソフトキーボードのウィンドウハンドルを返す。

ImmDestroySoftKeyboard 関数

ImmDestroySoftKeyboard 関数は、ソフトキーボードウィンドウを破棄する。

```
BOOL WINAPI ImmDestroySoftKeyboard(HWND hSoftKbdWnd)
```

引数

引数名	説明
hSoftKbdWnd	破棄するソフトキーボードのウィンドウハンドル。

戻り値

関数が成功すれば、戻り値は TRUE である。さもなければ戻り値は FALSE である。

ImmShowSoftKeyboard 関数

ImmShowSoftKeyboard 関数は、与えられたソフトキーボードを表示するか、または隠す。

```
BOOL WINAPI ImmShowSoftKeyboard(
      HWND hSoftKbdWnd,
      int nCmdShow)
```

引数

引数名	説明
hSoftKbdWnd	ソフトキーボードのウィンドウハンドル。
nCmdShow	ウィンドウの状態を表す。以下の値が与えられる。 **nCmdShow** / **意味** SW_HIDE / ソフトキーボードを隠す。 SW_SHOWNOACTIVATE / ソフトキーボードを表示する。

戻り値

関数が成功すれば、戻り値は TRUE である。さもなければ戻り値は FALSE である。

メッセージたち

以下のトピックは UI ウィンドウが受け取るメッセージたちである。

WM_IME_SETCONTEXT メッセージ

WM_IME_SETCONTEXT メッセージは、アプリのウィンドウがアクティブになったとき、アプリに送信される。もしアプリがアプリケーション IME ウィンドウを持たなければ、アプリはこのメッセージを DefWindowProc に渡

し、DefWindowProc の戻り値を返さねばならない。もしアプリがアプリケーション IME ウィンドウを所有していれば、アプリは ImmIsUIMessage を呼ぶべきである。

```
WM_IME_SETCONTEXT
fSet = (BOOL) wParam;
lISCBits = lParam;
```

引数

引数名	説明
fSet	入力コンテキストがアプリに対してアクティブになったとき fSet は TRUE である。もし FALSE なら、入力コンテキストはアプリに対して非アクティブになる。
lISCBits	lISCBits は以下のビット組み合わせからなる。 値 / 説明 ISC_SHOWUICOMPOSITIONWINDOW / コンポジションウィンドウを表示する。 ISC_SHOWUIGUIDWINDOW / ガイドウィンドウを表示する。 ISC_SHOWUICANDIDATEWINDOW / インデックス 0 の候補ウィンドウを表示する。 (ISC_SHOWUICANDIDATEWINDOW << 1) / インデックス 1 の候補ウィンドウを表示する。 (ISC_SHOWUICANDIDATEWINDOW << 2) / インデックス 2 の候補ウィンドウを表示する。 (ISC_SHOWUICANDIDATEWINDOW << 3) / インデックス 3 の候補ウィンドウを表示する。

戻り値

戻り値は、DefWindowProc か ImmIsUIMessage の戻り値である。

コメント

アプリが WM_IME_SETCONTEXT をつけて DefWindowProc か ImmIsUIMessage を呼んだ後で UI ウィンドウは WM_IME_SETCONTEXT を受け取る。もしビットが ON ならば、UI ウィンドウはコンポジション、ガイド、候補ウィンドウを lParam のビット状態として表示する。もしアプリ自体がコンポジションウィンドウを描画するなら、UI ウィンドウは、コンポジションウィンドウを表示する必要はない。そのときアプリは、lParam の ISC_SHOWUICOMPOSITIONWINDOW ビットをクリアして DefWindowProc か ImmIsUIMessage を呼ぶ必要がある。

WM_IME_CONTROL メッセージ

WM_IME_CONTROL メッセージは、IME UI を制御するために使われるサブメッセージのグループである。アプリは、アプリによって作成された IME ウィンドウと対話するためにこのメッセージを使うことができる。

```
WM_IME_CONTROL
wSubMessage = wParam;
```

```
lpData = (LPVOID)lParam;
```

引数

引数名	説明
wSubMessage	サブメッセージの値。
lpData	wSubMessage に依存する。

以下のトピックは、wSubMessage の値によって分類されるサブメッセージたちである。

IMC_GETSOFTKBDSUBTYPE、IMC_SETSOFTKBDSUBTYPE、IMC_SETSOFTKBDDATA、IMC_GETSOFTKBDFONT、IMC_SETSOFTKBDFONT、IMC_GETSOFTKBDPOS、および IMC_SETSOFTKBDPOS を除いて、アプリは、IME ウィンドウと通信するために、IMC メッセージの代わりに IMM API を使うことが推奨される。

IMC_GETCANDIDATEPOS

IMC_GETCANDIDATEPOS メッセージは、候補ウィンドウの位置を取得するために、アプリによって IME ウィンドウへ送信される。IME は、スクリーンの境界に応じて候補ウィンドウの位置を補正できる。さらにアプリは、候補ウィンドウを他の位置に動かすかどうか決定するために候補ウィンドウの本当の位置を取得することができる。

```
WM_IME_CONTROL
wSubMessage = IMC_GETCANDIDATEPOS;
lpCANDIDATENFORM = (LPCANDIDATEFORM) lParam;
```

引数

引数名	説明
lpCANDIDATENFORM	候補ウィンドウの位置を得るためのバッファ。

戻り値

メッセージが成功すれば、戻り値はゼロ。さもなければ戻り値は非ゼロ。

コメント

戻るときに、IME は、アプリのフォーカスウィンドウのクライアント座標を伴った lpCANDIDATENFORM によって指し示す CANDIDATEFORM 構造体を埋めるだろう。UI ウィンドウは、このメッセージを受け取る。アプリは別の候補ウィンドウの位置を lpCANIDATEFORM->dwIndex にゼロから 3 までの値を指定しなければならない（例えば、インデックス 0 は、トップレベルの候補ウィンドウである）。

IMC_GETCOMPOSITONFONT

IMC_GETCOMPOSITONFONT メッセージは、候補ウィンドウの未確定文字列の表示において、使われるフォントを取得するために、アプリによって IME ウィンドウへ送信される。

```
WM_IME_CONTROL
wSubMessage = IMC_GETCOMPOSITIONFONT;
lpLogFont = (LPLOGFONT)lParam;
```

引数

引数名	説明
lpLogFont	LOGFONT 構造体を受け取るバッファ。

戻り値

メッセージが成功すれば、戻り値はゼロ。さもなければ戻り値は非ゼロ。

コメント

UI ウィンドウはこのメッセージを受け取らない。

IMC_GETCOMPOSITONWINDOW

IMC_GETCOMPOSITONWINDOW メッセージは、コンポジションウィンドウの位置を取得するために、アプリによって IME ウィンドウへ送信される。IME は、コンポジションウィンドウの位置を補正でき、またアプリは他の位置に動かすかどうかを決定するために、コンポジションウィンドウの本当の位置を取得できる。

```
WM_IME_CONTROL
wSubMessage = IMC_GETCOMPOSITIONWINDOW;
lpCOMPOSITIONFORM = (LPCOMPOSITIONFORM) lParam;
```

引数

引数名	説明
lpCOMPOSITIONFORM	コンポジションウィンドウの位置を取得するためのバッファ。

戻り値

メッセージが成功すれば、戻り値はゼロ。さもなければ、戻り値は非ゼロ。

コメント

戻るときに、IME はアプリのフォーカスウィンドウのクライアント座標を伴った lpCANDIDATENFORM によって指し示す CANDIDATEFORM 構造体を埋めるだろう。UI ウィンドウはこのメッセージを受け取る。

IMC_GETSOFTKBDFONT

IMC_GETSOFTKBDFONT メッセージは、ソフトキーボードウィンドウで表示する文字に使うフォントを取得するために、IME によってソフトキーボードへ送信される。

```
WM_IME_CONTROL
wSubMessage = IMC_GETSOFTKBDFONT;
lpLogFont = (LPLOGFONT)lParam;
```

引数

引数名	説明
lpLogFont	LOGFONT 構造体を受け取るバッファ。

戻り値

メッセージが成功すれば、戻り値はゼロ。さもなければ戻り値は非ゼロ。

IMC_GETSOFTKBDPOS サブメッセージ

IMC_GETSOFTKBDPOS サブメッセージは、ソフトキーボードウィンドウの位置を取得するために、IME によってソフトキーボードウィンドウへ送信される。

```
WM_IME_CONTROL
wSubMessage = IMC_GETSOFTKBDPOS;
lParam = 0;
```

引数

引数名	説明
lParam	使用されない。

戻り値

戻り値は、スクリーン座標系でソフトキーボードの位置の x および y 座標を含む POINTS 構造体を指定する。

コメント

POINTS 構造体は次の形式を持つ。

```
typedef struct tagPOINTS { /* pts */
      SHORT x;
      SHORT y;
} POINTS;
```

IMC_GETSOFTKBDSUBTYPE

IMC_GETSOFTKBDSUBTYPE メッセージは、IMC_SETSOFTKBDSUBTYPE によってセットされたソフトキーボードウィンドウのサブタイプを取得するために、IME によってソフトキーボードウィンドウへ送信される。

```
WM_IME_CONTROL
wSubMessage = IMC_GETSOFTKBDSUBTYPE;
lParam = 0;
```

引数

引数名	説明
lParam	使用されない。

戻り値

戻り値は、IMC_SETSOFTKBDSUBTYPE によってセットされた、ソフトキーボードのサブタイプである。-1 の戻り値は、失敗を意味する。

IMC_GETSTATUSWINDOWPOS

IMC_GETSTATUSWINDOWPOS メッセージは、状態ウィンドウの位置を取得するために、アプリによって IME ウィンドウへ送信される。

```
WM_IME_CONTROL
wSubMessage= IMC_GETSTATUSWINDOWPOS;
```

```
lParam = 0;
```

引数名	説明
lParam	使用されない。

戻り値

戻り値は、スクリーン座標系で、状態ウィンドウの位置の x および y 座標を含む POINTS 構造体を指定する。

コメント

POINTS 構造体は次の形式を持つ。

```
typedef struct tagPOINTS { /* pts */
      SHORT x;
      SHORT y;
} POINTS;
```

UI ウィンドウはこのメッセージを受け取る。

IMC_SETCANDIDATEPOS

IMC_SETCANDIDATEPOS メッセージは、候補ウィンドウの表示位置を指定するために、アプリによって IME ウィンドウへ送信される。特にこのメッセージは、コンポジション文字列をアプリ自身が表示するが、候補を表示するために IME UI を使うアプリに適用される。

```
WM_IME_CONTROL
wSubMessage = IMC_SETCANDIDATEPOS;
lpCANDIDATEFORM = (LPCANDIDATEFORM)lParam;
```

引数

引数名	説明
lpCANDIDATEFORM	候補ウィンドウの位置情報を含むバッファ。

戻り値

メッセージが成功すれば、戻り値はゼロ。さもなければ戻り値は非ゼロ。

コメント

UI ウィンドウはこのメッセージを受け取らない。

IMC_SETCOMPOSITONFONT

IMC_SETCOMPOSITONFONT メッセージは、コンポジションウィンドウで未確定文字列の表示に使うフォントを指定するために、アプリによって IME ウィンドウへ送信される。

```
WM_IME_CONTROL
wSubMessage= IMC_SETCOMPOSITIONFONT;
lpLogFont= (LPLOGFONT) lParam;
```

引数

引数名	説明

lpLogFont	セットする LOGFONT 構造体データを含むバッファ。

戻り値

メッセージが成功すれば、戻り値はゼロ。さもなければ戻り値は非ゼロ。

コメント

UI ウィンドウはこのメッセージを受け取らない。

IMC_SETCOMPOSITONWINDOW

IMC_SETCOMPOSITONWINDOW メッセージは、現在アクティブな入力コンテキストのコンポジションウィンドウのスタイルをセットするために、アプリによって IME ウィンドウへ送信される。一度スタイルをセットすれば、IME UI は入力コンテキストで指定されたスタイルに従う。

```
WM_IME_CONTROL
wSubMessage = IMC_SETCOMPOSITIONWINDOW;
lpCOMPOSITIONFORM = (LPCOMPOSITIONFORM)lParam;
```

引数

引数名	説明
lpCOMPOSITIONFORM	コンポジションウィンドウに対する新しいスタイルを含む COMPOSITIONFORM 構造体。

戻り値

メッセージが成功されば、戻り値はゼロ。さもなければ戻り値は非ゼロ。

コメント

IME UI は、コンポジションウィンドウに対する既定のスタイルを使う。それは CFS_POINT スタイルに等しい。もしアプリがその入力コンテキストにおいて、コンポジションスタイルが指定しなければ、IME UI は、アプリがコンポジションウィンドウを開くときに、クライアント座標系で、現在のキャレット位置とウィンドウクライアント領域を受け取る。UI ウィンドウはこのメッセージを受け取らない。

IMC_SETSOFTKBDDATA

IMC_SETSOFTKBDDATA サブメッセージは、ソフトキーボードウィンドウにおける表示文字列に使う文字コードを指定するために、IME によってソフトキーボードウィンドウへ送信される。

```
WM_IME_CONTROL
wSubMessage = IMC_SETSOFTKBDDATA;
lpSoftKbdData = (LPSOFTKBDDATA)lParam;
```

引数

引数名	説明
lpSoftKbdData	表示文字列に使われる文字コードを指定するためのバッファを指し示す。

戻り値

メッセージが成功されば、戻り値はゼロ。さもなければ戻り値は非ゼロ。

コメント

UI ウィンドウはこのメッセージを受け取らない。

IMC_SETSOFTKBDSUBTYPE

IMC_SETSOFTKBDSUBTYPE サブメッセージは、IME によって、ソフトキーボードウィンドウにおける表示文字列に使うサブタイプを指定するために、ソフトキーボードウィンドウへ送信される。これは IME 特有の目的でも使える。

```
WM_IME_CONTROL
wSubMessage = IMC_SETSOFTKBDSUBTYPE;
lSubType = lParam;
```

引数

引数名	説明
lSubType	セットするサブタイプ。

戻り値

戻り値は、サブタイプである。-1 の戻り値は、失敗を意味する。

コメント

UI ウィンドウはこのメッセージを受け取らず、SOFTKEYBOARD_TYPE_T1 はこの情報を使わない。IME は、ソフトキーボードが表示された読みの文字列を変更しようとしないように、このメッセージを送信する。IME は、このメッセージの意味を定義するために、SOFTKEYBOARD_TYPE_T1 ソフトキーボードを使い、IMC_GETSOFTKBDSUBTYPE を使ってこのデータを取得できる。

IMC_SETSOFTKBDFONT

IMC_SETSOFTKBDFONT メッセージは、ソフトキーボードウィンドウにおける表示文字列で使うフォントを指定するために、IME によってソフトキーボードウィンドウへ送信される。

```
WM_IME_CONTROL
wSubMessage = IMC_SETSOFTKBDFONT;
lpLogFont = (LPLOGFONT)lParam;
```

引数

引数名	説明
lpLogFont	セットする LOGFONT 構造体を指し示す。

戻り値

メッセージが成功すれば、戻り値はゼロ。さもなければ戻り値は非ゼロ。

コメント

UI ウィンドウは、このメッセージを受け取らない。

IMC_SETSOFTKBDPOS

IMC_SETSOFTKBDPOS メッセージは、ソフトキーボードウィンドウの位置をセットするために、UI ウィンドウによってソフトキーボードウィンドウへ送信される。

```
WM_IME_CONTROL
wSubMessage = IMC_SETSOFTKBDPOS;
ptsPt = (POINTS)lParam;
```

引数

引数名	説明
ptsPt	スクリーン座標系でソフトキーボードウィンドウの位置の x および y 座標を含む POINTS 構造体を指定する。

戻り値

メッセージが成功すれば、戻り値はゼロ。さもなければ戻り値は非ゼロ。

コメント

POINTS 構造体は次の形式を持つ。

```
typedef struct tagPOINTS { /* pts */
      SHORT x;
      SHORT y;
} POINTS;
```

IMC_SETSTATUSWINDOWPOS

IMC_SETSTATUSWINDOWPOS メッセージは、状態ウィンドウの位置をセットするために、アプリによって IME ウィンドウへ送信される。

```
WM_IME_CONTROL
wSubMessage = IMC_SETSTATUSWINDOWPOS;
ptsPt = (POINTS)lParam;
```

引数

引数名	説明
ptsPt	スクリーン座標系で状態ウィンドウの位置の x および y 座標を含む POINTS 構造体を指定する。

戻り値

メッセージが成功すれば、戻り値はゼロ。さもなければ戻り値は非ゼロ。

コメント

POINTS 構造体は次の形式を持つ。

```
typedef struct tagPOINTS { /* pts */
      SHORT x;
```

```
        SHORT y;
    } POINTS;
```

WM_IME_COMPOSITION

WM_IME_COMPOSITION メッセージは、IME コンポジション状態がユーザーによって変更されるときに、アプリへ送信される。メッセージはコンポジション文字の 2 バイトから構成される。IME UI ウィンドウは、それがこのメッセージを処理するときに、見た目を変える。アプリは、新しいコンポジション状態を取得するために、ImmGetCompositionString を呼び出すことができる。

```
WM_IME_COMPOSITION
wChar = wParam;
lAttribute = lParam;
```

引数

<table>
<tr><th>引数名</th><th>説明</th></tr>
<tr><td>wChar</td><td>コンポジション文字の最新の変更点の 2 バイト。</td></tr>
<tr><td>lAttribute</td><td>以下のフラグ組み合わせからなる。基本的に、フラグはどのようにコンポジション文字列または文字が変更されたかを示す。アプリは必要な情報を取得するためにこれをチェックする。
<table>
<tr><th>値</th><th>説明</th></tr>
<tr><td>GCR_ERRORSTR</td><td>エラー文字列を更新する。</td></tr>
<tr><td>GCR_INFORMATIONSTR</td><td>情報文字列を更新する。</td></tr>
<tr><td>GCS_COMPATTR</td><td>コンポジション文字列の属性を更新する。</td></tr>
<tr><td>GCS_COMPCLAUSE</td><td>コンポジション文字列の文節情報を更新する。</td></tr>
<tr><td>GCS_COMPREADATTR</td><td>現在のコンポジションの読みの文字列の属性を更新する。</td></tr>
<tr><td>GCS_COMPREADCLAUSE</td><td>コンポジション文字列の読みの文字列の文節情報を更新する。</td></tr>
<tr><td>GCS_COMPREADSTR</td><td>現在のコンポジションの読みの文字列を更新する。</td></tr>
<tr><td>GCS_COMPSTR</td><td>現在のコンポジション文字列を更新する。</td></tr>
<tr><td>GCS_CURSORPOS</td><td>コンポジション文字列中のカーソル位置を更新する。</td></tr>
<tr><td>GCS_DELTASTART</td><td>コンポジション文字列の変更の開始位置を更新する。</td></tr>
<tr><td>GCS_RESULTCLAUSE</td><td>結果文字列の文節情報を更新する。</td></tr>
<tr><td>GCS_RESULTREADCLAUSE</td><td>読みの文字列の文節情報を更新する。</td></tr>
<tr><td>GCS_RESULTREADSTR</td><td>読みの文字列を更新する。</td></tr>
<tr><td>GCS_RESULTSTR</td><td>コンポジション結果の文字列を更新する。</td></tr>
</table>
以下のスタイルビット値が WM_IME_COMPOSITION に与えられる。
<table>
<tr><th>値</th><th>説明</th></tr>
<tr><td>CS_INSERTCHAR</td><td>IME は現在の挿入位置へ挿入されるべきコンポジション文字を wParam で指定するとき、この値を指定する。もしアプリがこのビッ</td></tr>
</table>
</td></tr>
</table>

引数名	説明	
		トフラグを処理するなら、アプリはコンポジション文字を表示すべきだ。
	CS_NOMOVECARET	IME は、WM_IME_COMPOSITION の処理結果としてキャレット位置をアプリによって移動してもらいたくないときにこの値を指定する。例えば、もし IME が CS_INSERTCHAR と CS_NOMOVECARET の組み合わせを指定したら、アプリが wParam によって与えられる文字が現在のキャレット位置に挿入すべきであるが、キャレットを移動すべきではない。GCS_RESULTSTR フラグを含む、付随の WM_IME_COMPOSITION メッセージたちは、この文字を置換する。

戻り値

なし。

コメント

アプリ自体がコンポジション文字列を表示したいとき、このメッセージをアプリケーション IME UI ウィンドウや、DefWindowProc へ渡すべきではない。DefWindowProc 関数は、このメッセージを既定の IME ウィンドウに渡すために処理する。IME は、IME が現在のコンポジションをキャンセルするだけであっても、このメッセージをアプリへ送信すべきだ。このメッセージは、現在のコンポジション文字列を消すために、アプリや IME UI に通知するために使われるべきだ。

参照

ImmGetCompositionString

WM_IME_COMPOSITIONFULL

WM_IME_COMPOSITIONFULL メッセージは、IME UI ウィンドウがコンポジションウィンドウのサイズを増やせないときに、アプリに送信される。アプリは、このメッセージを受け取ったときに、どのように IME UI ウィンドウを表示するか指定すべきである。

```
WM_IME_COMPOSITIONFULL
wParam = 0
lParam = 0
```

引数

引数名	説明
wParam	使われない。
lParam	使われない。

戻り値

なし。

コメント

このメッセージは、IME UI ウィンドウによってアプリに送信される通知である。IME 自身によるものではない。IME は、この通知を送信するために SendMessage を使う。

参照

IMC_SETCOMPOSITONWINDOW

WM_IME_ENDCOMPOSITION メッセージ

WM_IME_ENDCOMPOSITION メッセージは、IME がコンポジションを終了したときに、アプリに送信される。

```
WM_IME_ENDCOMPOSITION
wParam = 0
lParam = 0
```

引数

引数名	説明
wParam	使われない。
lParam	使われない。

戻り値

なし。

コメント

アプリ自身がコンポジション文字列を表示したいとき、このメッセージをアプリケーション IME UI ウィンドウや DefWindowProc へ渡すべきではない。DefWindowProc は、既定の IME ウィンドウにそれを渡すために、このメッセージを処理する。

WM_IME_SELECT メッセージ

WM_IME_SELECT メッセージは、システムが現在の IME を変更しようとしているときに、UI ウィンドウに送信される。

```
WM_IME_SELECT
fSelect = (BOOL)wParam;
hKL = lParam;
```

引数

引数名	説明
fSelect	IME が新しく選択されたら TRUE。IME が選択を解除されたら FALSE。
hKL	IME の入力言語ハンドル。

戻り値

なし。

コメント

システム IME クラスは、新しい UI ウィンドウを作成、並びにアプリやシステムに対する古い UI ウィンドウを破棄するためにこのメッセージを使う。DefWindowProc は、既定の IME ウィンドウへ情報を渡すために、このメッセージを処理する。そのとき既定の IME ウィンドウは、UI ウィンドウへこのメッセージを送信する。

WM_IME_STARTCOMPOSITION メッセージ

WM_IME_STARTCOMPOSITION メッセージは、ユーザーのキーストロークの結果として、IME がコンポジション文字列を生成する直前に送信される。UI ウィンドウは、このメッセージを受け取るときに、コンポジションウィンドウを開く。

```
WM_IME_STARTCOMPOSITION
wParam = 0
lParam = 0
```

引数

引数名	説明
wParam	使われない。
lParam	使われない。

戻り値

なし。

コメント

アプリ自体がコンポジション文字列を表示したいとき、このメッセージをアプリケーション IME ウィンドウや DefWindowProc へ渡すべきではない。DefWindowProc 関数は、既定の IME ウィンドウへこのメッセージを渡すためにこのメッセージを処理する。

WM_IME_NOTIFY メッセージ

WM_IME_NOTIFY メッセージは、アプリや、IME 状態の UI ウィンドウへ通知するサブメッセージのグループである。

```
WM_IME_NOTIFY
wSubMessage = wParam; // submessage ID
lParam = lParam; // depends on the submessage
```

以下のトピックは、wSubMessage の値によって分類されるサブメッセージたちである。

IMN_CLOSESTATUSWINDOW

IMN_CLOSESTATUSWINDOW メッセージは、IME が状態ウィンドウが閉じようとしているときに、送信される。

```
WM_IME_NOTIFY
wSubMessage = IMN_CLOSESTATUSWINDOW;
lParam = 0;
```

引数

引数名	説明
lParam	使われない。

戻り値

なし。

コメント

UI ウィンドウは、このメッセージを受け取ったときに、状態ウィンドウを閉じる。

IMN_OPENSTATUSWINDOW

IMN_OPENSTATUSWINDOW メッセージは、IME が状態ウィンドウを作成しようとしているときに送信される。そのときアプリはこのメッセージを処理し、IME 自身のシステムウィンドウを表示する。アプリは、ImmGetConversionStatus 関数を呼ぶことでシステムウィンドウに関する情報を取得できる。

```
WM_IME_NOTIFY
wSubMessage = IMN_OPENSTATUSWINDOW;
lParam = 0;
```

引数

引数名	説明
lParam	使われない。

戻り値

なし。

コメント

UI ウィンドウは、このメッセージを受け取ったときに状態ウィンドウを作成する。

参照

ImmGetConversionStatus

IMN_OPENCANDIDATE

IMN_OPENCANDIDATE サブメッセージは、IME が候補ウィンドウを開こうとしているときに送信される。そしてアプリは、このメッセージを処理し、候補ウィンドウ自身を表示するために、ImmGetCandidateCount と ImmGetCandidateList を呼び出す。

```
WM_IME_NOTIFY
wSubMessage = IMN_OPENCANDIDATE;
lCandidateList = lParam;
```

引数

引数名	説明
lCandidateList	更新すべき候補リストがどれかを表す。例えば、ビット 0 が 1 なら、最初の候補リストが更新されるべきだ。もし、ビット 31 が 1 なら、32 番目の候補リストが更新されるべきだ。

戻り値

なし。

コメント

UI ウィンドウはこのメッセージを受け取ったときに候補ウィンドウを作成する。

参照

ImmGetCandidateListCount、ImmGetCandidateList、WM_IME_CHANGECANDIDATE

IMN_CHANGECANDIDATE

IMN_CHANGECANDIDATE メッセージは、IME が候補ウィンドウの内容を変更しようとしているときに送信される。そしてアプリは、候補ウィンドウ自体を表示するために、このメッセージを処理する。

```
WM_IME_NOTIFY
wSubMessage = IMN_CHANGECANDIDATE;
lCandidateList = lParam;
```

引数

引数名	説明
lCandidateList	更新すべき候補リストがどれかを表す。例えば、ビット 0 が 1 なら、最初の候補リストが更新されるべきだ。もし、ビット 31 が 1 なら、32 番目の候補リストが更新されるべきだ。

戻り値

なし。

コメント

UI ウィンドウは、このメッセージを受け取ったとき、候補ウィンドウを再描画する。

参照

ImmGetCandidateCount、ImmGetCandidateList

IMN_CLOSECANDIDATE

IMN_CLOSECANDIDATE メッセージは、IME が候補ウィンドウを閉じようとしているときに送信される。アプリは、候補処理の終わりに関する情報を取得するために、このメッセージを処理する。

```
WM_IME_NOTIFY
wSubMessage = IMN_CLOSECANDIDATE;
lCandidateList = lParam;
```

引数

引数名	説明
lCandidateList	更新すべき候補リストがどれかを表す。例えば、ビット 0 が 1 なら、最初の候補リストが更新されるべきだ。もし、ビット 31 が 1 なら、32 番目の候補リストが更新されるべきだ。

戻り値

なし。

コメント

UI ウィンドウは、このメッセージを受け取ったときに、候補ウィンドウを破棄する。

IMN_SETCONVERSIONMODE

IMN_SETCONVERSIONMODE メッセージは、入力コンテキストの変換モードが更新されたときに送信される。アプリまたは UI ウィンドウがこのメッセージを受け取ったとき、どちらかは、状態ウィンドウに関する情報を取得するために、ImmGetConversionStatus を呼び出すことができる。

```
WM_IME_NOTIFY
wSubMessage = IMN_SETCONVERSIONMODE;
lParam = 0;
```

引数

引数名	説明
lParam	使われない。

戻り値

なし。

コメント

UI ウィンドウは、状態ウィンドウが変換モードを指定したら、状態ウィンドウを再描画する。

IMN_SETSENTENCEMODE

IMN_SETSENTENCEMODE メッセージは入力コンテキストのセンテンスモードが更新されたときに送信される。アプリまたは UI ウィンドウがこのメッセージを受け取ったとき、どちらかは、状態ウィンドウに関する情報を取得するために、ImmGetConversionStatus を呼び出すことができる。

```
WM_IME_NOTIFY
wSubMessage = IMN_SETSENTENCEMODE;
lParam = 0;
```

引数

引数名	説明
lParam	使われない。

戻り値

なし。

コメント

UI ウィンドウは、状態ウィンドウがセンテンスモードを指定したとき、状態ウィンドウを再描画する。

IMN_SETOPENSTATUS

IMN_SETOPENSTATUS メッセージは、入力コンテキストのオープン状態が更新されたときに送信される。アプリまたは UI ウィンドウがこのメッセージを受け取ったとき、どちらかは、情報を取得するために ImmGetOpenStatus を呼び出すことができる。

```
WM_IME_NOTIFY
wSubMessage = IMN_SETOPENSTATUS;
lParam = 0;
```

引数

引数名	説明
lParam	使われない。

戻り値

なし。

コメント

UI ウィンドウは、状態ウィンドウが開いたり閉じたりした状態を指定したら、状態ウィンドウを再描画する。

IMN_SETCANDIDATEPOS

IMN_SETCANDIDATEPOS メッセージは、IME が候補ウィンドウを動かそうとしているときに送信される。アプリは、候補処理の終了に関する情報を取得するためにこのメッセージを処理する。

```
WM_IME_NOTIFY
wSubMessage = IMN_SETCANDIDATEPOS;
lCandidateList = lParam;
```

引数

引数名	説明
lCandidateList	更新すべき候補リストがどれかを表す。例えば、ビット 0 が 1 なら、最初の候補リストが更新されるべきだ。もし、ビット 31 が 1 なら、32 番目の候補リストが更新されるべきだ。

戻り値

なし。

コメント

UI ウィンドウは、このメッセージを受け取ったときに、候補ウィンドウを移動する。

IMN_SETCOMPOSITIONFONT

IMN_SETCOMPOSITIONFONT メッセージは、入力コンテキストのフォントが更新されたときに送信される。アプリまたは UI ウィンドウがこのメッセージを受け取ったとき、どちらかは、コンポジションフォントに関する情報を受け取るために、ImmGetCompositionFont を呼び出すことができる。

```
WM_IME_NOTIFY
wSubMessage = IMN_SETCOMPOSITIONFONT;
lParam = 0;
```

引数

引数名	説明
lParam	使われない。

戻り値

なし。

コメント

UI ウィンドウのコンポジションコンポーネントは、コンポジション文字列のテキストを描画するために ImmGetCompositionFont を呼び出すことで、フォント情報を使うことができる。

IMN_SETCOMPOSITIONWINDOW

IMN_SETCOMPOSITIONWINDOW メッセージは、入力コンテキストのコンポジションフォームが更新されたときに送信される。UI ウィンドウがこのメッセージを受け取ったら、入力コンテキストの cfCompForm は、新しい変換モードを取得するために参照されることができる。

```
WM_IME_NOTIFY
wSubMessage = IMN_SETCOMPOSITIONWINDOW;
lParam = 0;
```

引数

引数名	説明
lParam	使われない。

戻り値

なし。

コメント

UI ウィンドウのコンポジションコンポーネントは、コンポジションウィンドウを表示するために cfCompForm を使う。

IMN_GUIDELINE

IMN_GUIDELINE メッセージは、IME がエラーや情報を表示しようとしているときに送信される。アプリまたは UI ウィンドウがこのメッセージを受け取ったとき、どちらかはガイドラインに関する情報を取得するために、

ImmGetGuideLine を呼び出すことができる。

```
WM_IME_NOTIFY
wSubMessage = IMN_GUIDELINE;
lParam = 0;
```

引数

引数名	説明
lParam	使われない。ゼロでなければならない。

戻り値

なし。

コメント

UI ウィンドウは、このメッセージを受け取ったときに情報ウィンドウを作成し、情報文字列を表示する。

参照

ImmGetGuideLine、GUIDELINE 構造体

IMN_SOFTKBDDESTROYED

IMN_SOFTKBDDESTROYED メッセージは、ソフトキーボードが破棄されたときに、UI ウィンドウに送信される。

```
WM_IME_NOTIFY
wSubMessage = IMN_SOFTKBDDESTROYED;
lParam = 0;
```

引数

引数名	説明
lParam	使われない。ゼロでなければならない。

戻り値

なし。

WM_IME_KEYDOWN と WM_IME_KEYUP

WM_IME_KEYDOWN と WM_IME_KEYUP メッセージは、IME が WM_KEYDOWN か WM_KEYUP メッセージを生成する必要があるときに、アプリに送信される。送信される値は、オリジナルな Windows の WM_KEYDOWN と WM_KEYUP の値と同じである（英語版）。

```
WM_IME_KEYDOWN / WM_IME_KEYUP
nVirtKey = (int)wParam; // virtual-key code
lKeyData = lParam; // key data
```

引数

引数名	説明
nVirtKey	wParam の値。システムキーではない仮想キーコードを指定する。
lKeyData	lParam の値。リピートカウント、スキャンコード、拡張キーフラグ、コンテキストコード、直前のキー状態フラグ、そしてトランジション状態フラグを指定する。それは、オリジナルの Windows の WM_KEYDOWN と WM_KEYUP メッセージに対するものと同じである。

戻り値

なし。

コメント

アプリは、WM_KEYDOWN と WM_KEYUP メッセージと同じ方法でこのメッセージを扱うことができる。さもなければ、DefWindowProc が、同じ wParam と lParam 引数をつけて WM_KEYDOWN か WM_KEYUP メッセージを生成するために、このメッセージを処理する。このメッセージは、メッセージ順を管理するために、たいてい IME によって生成される。

WM_IME_CHAR

WM_IME_CHAR メッセージは、IME が変換結果の文字を取得するときに、アプリに送信される。送信される値は、オリジナルの Windows の WM_CHAR に似ている（英語版）。違うのは、wParam が文字の 2 バイトを含むことができることだ。

```
WM_IME_CHAR
wCharCode = wParam;
lKeyData = lParam;
```

引数

引数名	説明
wCharCode	極東文字（FE character）に対する 2 バイトを含む。NT Unicode アプリについては、1 文字の Unicode 文字を含む。
lKeyData	オリジナルの Windows の WM_CHAR のものと同じ（英語版）。以下が利用可能なビットとその説明である。

値	説明
0 – 15	繰り返しカウント。最初のバイトと二番目のバイトは連続だから、これは常に 1 である。
16 – 23	スキャンコード。完全な極東文字のためのスキャンコード。
24 – 28	使われない。
29	コンテキストコード。
31	変換状態。

戻り値

なし。

コメント

アプリがこのメッセージを扱わなければ、DefWindowProc 関数が、WM_CHAR メッセージを生成するためにこのメッセージを処理する。もしアプリが Unicode ではなく、wCharCode が 2 バイトの DBCS 文字を含むなら、DefWindowProc 関数は、2 個の WM_CHAR メッセージを生成し、各メッセージは、DBCS 文字の 1 バイトを所有する。もしメッセージがただ SBCS 文字を含むなら、DefWindowProc は、1 個の WM_CHAR メッセージのみを生成する。

VK_PROCESSKEY

VK_PROCESSKEY メッセージは、WM_KEYDOWN か WM_KEYUP の wParam としてアプリへ送信される。この仮想キーが生成されるとき、本当の仮想キーは入力コンテキストに保存されるか、あるいは IME によって生成されたメッセージが入力コンテキストに格納される。システムは、本当の仮想キーを復元するか、入力コンテキストのメッセージバッファに格納されるメッセージを投稿するかのいずれかである。

```
WM_KEYDOWN / WM_KEYUP
wParam = VK_PROCESSKEY;
lParam = 1;
```

引数

引数名	説明
lParam	1 でなければならない。

NDICM_SETIMEICON

このメッセージは、IME がシステムペンアイコンに対するアイコンを変更したいときに、インディケータウィンドウに送信される。このメッセージは、フォーカスウィンドウの選択中の hKL が送信元の IME と同じであるときに、受け入れられる。

```
INDICM_SETIMEICON
nIconIndex = wParam;
hKL = lParam;
```

引数

引数名	説明
nIconIdex	IME ファイルのアイコンリソースのインデックス。もしこの値が(-1)であれば、インディケータはシステムによって提供されたオリジナルのアイコンを復元する。
hKL	送信元の IME である。

戻り値

非ゼロの値は失敗を表す。さもなければゼロが返される。

コメント

タスクバー管理の内部設計の要件のため、IME は、INDICM_xxx メッセージに対して PostMessage を使わなければならない。

INDICM_SETIMETOOLTIPS

このメッセージは、IME がシステムペンアイコンのツールチップ文字列を変更したいときに、インディケータウィンドウへ送信される。このメッセージは、フォーカスウィンドウの選択中の hKL が送信元 IME と同じであるときに、受け入れられる。

```
INDICM_SETIMETOOLTIPS
hAtom = wParam;
hKL = lParam;
```

引数

引数名	説明
hAtom	ツールチップ文字列に対するグローバル ATOM の値。もしこの値が(-1)であれば、インディケータは、システムによって提供されたオリジナルのチップを復元する。
hKL	送信元の IME である。

戻り値

非ゼロの値は失敗を表す。さもなければゼロが返される。

コメント

タスクバー管理の内部設計の要件のため、IME は INDICM_xxx メッセージに対して PostMessage を使わなければならない。グローバル ATOM は、GlobalAddAtom か GlobalFindAtom で取得しなければならない。

INDICM_REMOVEDEFAULTMENUITEMS

このメッセージは、IME がシステムペンアイコンの既定のメニュー項目たちを除去したいときに、インディケータウィンドウへ送信される。

```
INDICM_REMOVEDEFAULTMENUITEMS
wValue = wParam;
hKL = lParam;
```

引数

<table>
<tr><th>引数名</th><th>説明</th></tr>
<tr><td>wValue</td><td>wValue は次のビットたちの組み合わせである。

<table><tr><th>値</th><th>説明</th></tr><tr><td>RDMI_LEFT</td><td>左クリックメニューのメニュー項目たちを除去する。</td></tr><tr><td>RDMI_RIGHT</td><td>右クリックメニューのメニュー項目たちを除去する。</td></tr></table>
もし wValue がゼロであれば、すべての既定のメニュー項目たちが復元される。</td></tr>
<tr><td>hKL</td><td>送信元の IME である。</td></tr>
</table>

戻り値

非ゼロの値は失敗を表す。さもなければゼロが返される。

コメント

タスクバー管理の内部設計の要件のため、IME は INDICM_xxx メッセージに対して PostMessage を使わなければならない。

IME インターフェイス関数

IME は動的リンクライブラリ(DLL)として提供される。入力方式管理(IMM)は、すべてのインストールされた IME を扱わなければならない。IME は再起動することなく実行時に変更可能であるので、IMM は、それぞれの IME のすべてのエントリポイントを管理するための構造体を持つだろう。

以下のトピックは、すべての共通 IME 関数たちである。これらの関数は、アプリによって直接呼び出すべきではない。

ImeInquire 関数

ImeInquire 関数は、IME の初期化を扱う。また、IMEINFO 構造体や IME の UI クラス名も返す。

Windows 95、Windows 98、そして Windows NT 3.51 について:

```
BOOL ImeInquire(
      LPIMEINFO lpIMEInfo,
      LPTSTR    lpszWndClass,
      LPCTSTR   lpszData)
```

引数

引数名	説明
lpIMEInfo	IME 情報構造体へのポインタ。
lpszWndClass	IME によって埋められるべき、ウィンドウクラス名。この名前は IME の UI クラスである。
lpszData	IME のオプションブロック。このバージョンでは NULL。

Windows NT 4.0 および Windows 2000 について:

```
BOOL ImeInquire(
      LPIMEINFO lpIMEInfo,
      LPTSTR    lpszWndClass,
      DWORD     dwSystemInfoFlags)
```

引数

<table>
<tr><th>引数名</th><th>説明</th></tr>
<tr><td>lpIMEInfo</td><td>IME 情報構造体へのポインタ。</td></tr>
<tr><td>lpszWndClass</td><td>IME によって埋められるべき、ウィンドウクラス名。この名前は IME の UI クラスである。</td></tr>
<tr><td>dwSystemInfoFlags</td><td>システムによって与えられるさまざまなシステム情報。以下のフラグが与えられる。
<table>
<tr><th>フラグ</th><th>説明</th></tr>
<tr><td>IME_SYSINFO_WINLOGON</td><td>クライアントプロセスが、Winlogon プロセスであることを IME に伝える。IME は、このフラグ</td></tr>
</table>
</td></tr>
</table>

引数名	説明
	フラグ / 説明
	が指定されたときに、IME はユーザに IME の設定の変更を許可すべきではない。
	IME_SYSINFO_WOW16 / クライアントプロセスが、16 ビットアプリであることを IME に伝える。

戻り値

関数が成功すれば、戻り値は TRUE である。さもなければ戻り値は FALSE である。

ImeConversionList 関数

ImeConversionList 関数は、別の文字や文字列から変換後の結果リストを取得する。

```
DWORD IMEConversionList(
      HIMC              hIMC,
      LPCTSTR           lpSrc,
      LPCANDIDATELIST   lpDst,
      DWORD             dwBufLen,
      UINT              uFlag)
```

引数

引数名	説明
hIMC	入力コンテキスト。
lpSrc	変換元の文字列。
lpDst	変換先バッファへのポインタ。
dwBufLen	変換先バッファの長さ。
uFlag	現在、以下の 3 つのフラグのうち 1 つを選択できる。

フラグ	説明
GCL_CONVERSION	引数 lpSrc に読みの文字列を指定する。IME は結果文字列を引数 lpDst に返す。
GCL_REVERSECONVERSION	結果文字列を引数 lpSrc に指定する。IME は読みの文字列を引数 lpDst に返す。
GCL_REVERSE_LENGTH	結果文字列を引数 lpSrc に指定する。IME は、GCL_REVERSECONVERSION で扱える長さを返す。例えば、IME は読みの文字列に句点の結果文字列を変換できない。結果としてそれは句点のない文字列の長さバイト数として返す。

戻り値

戻り値は、結果文字列リストのバイト数である。

コメント

この関数は、アプリ、あるいは IME 関連のメッセージを生成していない IME によって呼び出されることを想定している。したがって、この関数において、IME は IME 関連のメッセージを生成すべきではない。

ImeConfigure 関数

ImeConfigure 関数は、IME のオプショナルな情報を要求するために使うダイアログを提供する。

```
BOOL ImeConfigure(
      HKL hKL,
      HWND hWnd,
      DWORD dwMode,
      LPVOID lpData)
```

引数

引数名	説明
hKL	IME の入力言語ハンドル。
hWnd	親ウィンドウのハンドル。
dwMode	ダイアログのモード。以下のフラグが与えられる。 フラグ / 説明 IME_CONFIG_GENERAL / 汎用のダイアログ。 IME_CONFIG_REGWORD / 単語登録のダイアログ。 IME_CONFIG_SELECTDICTIONARY / IME 辞書選択のダイアログ。
lpData	VOID へのポインタで、それは(dwMode == IME_CONFIG_REGISTERWORD)のときのみ、REGISTERWORD 構造体へのポインタになるだろう。さもなければ lpData は単に無視されるべきだ。また、もし初期文字列情報が与えられなければ、IME_CONFIG_REGISTER モードで NULL を指定できる。

戻り値

関数が成功すれば、戻り値は TRUE である。さもなければ戻り値は FALSE である。

コメント

IME は次のような疑似コードで lpData をチェックする。

```
if (dwmode != IME_CONFIG_REGISTERWORD)
{
      // オリジナルの実行をする。
}
else if (IsBadReadPtr(lpdata, sizeof(REGISTERWORD)) == FALSE)
{
      if (IsBadStringPtr(PREGISTERWORD(lpdata)->lpReading, (UINT)-1) == FALSE)
      {
            // 読みの文字列を単語登録ダイアログにセットする。
      }
      if (IsBadStringPtr(PREGISTERWORD(lpdata)->lpWord, (UINT)-1) == FALSE)
      {
```

```
                // 単語文字列を単語登録ダイアログにセットする。
        }
    }
```

ImeDestroy 関数

ImeDestroy 関数は IME 自身を終了する。

```
BOOL ImeDestroy(UINT uReserved)
```

引数

引数名	説明
uReserved	予約済み。現在はゼロであるべき。このバージョンでは、それがゼロでなければ IME は FALSE を返すべき。

戻り値

関数が成功すれば、戻り値は TRUE である。さもなければ戻り値は FALSE である。

ImeEscape 関数

ImeEscape 関数は、他の IMM 関数を通じては直接利用可能ではない特定の IME の能力へアプリがアクセスすることを許可する。これは主に、国特有の機能や IME 内部の機能に対して必要である。

```
LRESULT ImeEscape(
      HIMC   hIMC,
      UINT   uEscape,
      LPVOID lpData)
```

引数

引数名	説明
hIMC	入力コンテキストのハンドル。
uEscape	実行したいエスケープ機能を指定する。
lpData	エスケープに必要なデータを指し示す。

ImeEscape 関数は以下のエスケープ機能をサポートする。

uEscape	意味
IME_ESC_QUERY _SUPPORT	実装をチェックする。もしこのエスケープが実装されていなければ戻り値はゼロである。
IME_ESC_RESERVED_FIRST	IME_ESC_RESERVED_FIRST と IME_ESC_RESERVED_LAST の間のエスケープはシステムによって予約済みである。
IME_ESC_RESERVED_LAST	IME_ESC_RESERVED_FIRST と IME_ESC_RESERVED_LAST の間のエスケープはシステムによって予約済みである。
IME_ESC_PRIVATE_FIRST	IME_ESC_PRIVATE_FIRST と IME_ESC_PRIVATE_LAST の間のエスケープは IME によって予約済みである。IME は IME 固有の目的でこれらのエスケープ機能を自由に使うことができる。
IME_ESC_PRIVATE_LAST	IME_ESC_PRIVATE_FIRST と IME_ESC_PRIVATE_LAST の間のエ

uEscape	意味
	スケープは IME によって予約済みである。IME は IME 固有の目的でこれらのエスケープ機能を自由に使うことができる。
IME_ESC_SEQUENCE_TO_INTERNAL	中国語特有のエスケープ。すべての極東のプラットフォームで実行したいアプリはこれを使うべきではない。これは中国語 EUDC エディタのためのものである。*(LPWORD)lpData はシーケンスコードで、戻り値はこのシーケンスコードに対する文字コードである。典型的に中国語 IME は読み文字コードを 1 から n までのシーケンスへエンコードする。
IME_ESC_GET_EUDC_DICTIONARY	中国語特有のエスケープ。すべての極東のプラットフォームで実行したいアプリはこれを使うべきではない。これは中国語 EUDC エディタのためのものである。この機能から戻るときに、(LPTSTR)lpData は EUDC 辞書のフルパスファイル名で埋められる。lpData によって指し示されるこのバッファのサイズは、MAX_PATH * sizeof(TCHAR)以上でなければならない。注意 : Windows 95/98 と Windows NT 4.0 の EUDC エディタは、80*sizeof(TCHAR)までのバッファを使うことを想定している。
IME_ESC_SET_EUDC_DICTIONARY	EUDC 辞書ファイルをセットする。入力時に、引数 lpData は、フルパスを指定するゼロ終端文字列のポインタである。中国語 EUDC エディタで使うときは、他のアプリで使うべきではない。
IME_ESC_MAX_KEY	中国語特有のエスケープ。すべての極東のプラットフォームで実行したいアプリはこれを使うべきではない。これは中国語 EUDC エディタのためのものである。戻り値は、EUDC 文字に対するキーストロークの最大個数である。
IME_ESC_IME_NAME	中国語特有のエスケープ。すべての極東のプラットフォームで実行したいアプリはこれを使うべきではない。これは中国語 EUDC エディタのためのものである。この機能から戻るとき、(LPTSTR)lpData は、EUDC エディタで表示される IME の名前である。lpData によって指し示されるこのバッファのサイズは、16 * sizeof(TCHAR)以上でなければならない。
IME_ESC_SYNC_HOTKEY	繁体字中国語特有のエスケープ。すべての極東のプラットフォームで実行したいアプリはこれを使うべきではない。これは、異なる IME の間の同期のためにある。入力引数*(LPDWORD)lpData は、IME のプライベートなホットキー ID である。もしこの ID がゼロなら、この IME は関連するすべてのプライベートなホットキーをチェックすべきある。
IME_ESC_HANJA_MODE	韓国語特有のエスケープ。すべての極東のプラットフォームで実行したいアプリはこれを使うべきではない。これは Hangeul から Hanja への変換のためにある。入力引数(LPSTR)lpData は、Hangeul 文字列で埋められ、それは Hanja とゼロ終端文字列に変換されるだろう。コンポジション文字が存在するときで、アプリが Hangeul 文字を Hanja 変換と同じ方式で Hanja 文字に変換したいとき、アプリはこの機能をリクエストするだけでいい。そのとき IME 自体が Hanja 変換モードとして設定する。

uEscape	意味
IME_ESC_GETHELPFILENAME	IME のヘルプファイルの名前のエスケープ。この機能から戻るとき、(LPTSTR)lpData は IME のヘルプファイルのフルパスファイル名である。パス名は、MAX_PATH * sizeof(TCHAR)未満でなければならない。これは Windows 98 と Windows 2000 に追加された。注意: Windows 98 はパスの長さが 80 個未満の TCHAR であることを想定している。
IME_ESC_PRIVATE_HOTKEY	lpData は、IME_HOTKEY_PRIVATE_FIRST から IME_HOTKEY_PRIVATE_LAST までの範囲のホットキー ID を含む DWORD を指し示す。システムがこの範囲でホットキーリクエストを受け取った後、IMM は ImeEscape 関数を使って、それを IME に発送（dispatch）する。注意:Windows 95 はこのエスケープをサポートしない。

戻り値

関数が失敗したら、戻り値はゼロ。さもなければ戻り値は、各エスケープ機能に依存する。

コメント

パラメータの正当性の実証は、強靭性（robustness）のために、各エスケープ機能の内部で行われるべきだ。uEscape が IME_ESC_QUERY_SUPPORT であるとき、lpData は、IME のエスケープの値を含む変数へのポインタである。次は、現在の IME が IME_ESC_GETHELPFILENAME をサポートするかどうかを決定するために使える例である。

```
IME_ESC_GETHELPFILENAME.
DWORD dwEsc = IME_ESC_GETHELPFILENAME;
LRESULT lRet = ImmEscape(hKL, hIMC, IME_ESC_QUERYSUPPORT, (LPVOID)&dwEsc);
```

参照

ImmEscape

ImeSetActiveContext 関数

ImeSetActiveContext 関数は、現在の IME のアクティブな入力コンテキストを通知する。

```
BOOL ImeSetActiveContext(
      HIMC hIMC,
      BOOL fFlag)
```

引数

引数名	説明
hIMC	入力コンテキストのハンドル。
fFlag	2 つのフラグが与えられる。TRUE はアクティブにされたことを示し、FALES は非アクティブにされたことを示す。

戻り値

関数が成功したら、戻り値は TRUE である。さもなければ戻り値は FALSE である。

コメント

IME は、新しく選択された入力コンテキストについてこの関数によって通知される。IME は初期化を実行できるが、それは必ずしも必要ではない。

参照

ImeSetActiveContext

ImeProcessKey 関数

ImeProcessKey 関数は、IMM を通じて与えられたすべてのキーストロークを前処理し、与えられた入力コンテキストについてキーが IME に必要であれば TRUE を返す。

```
BOOL ImeProcessKey(
      HIMC          hIMC,
      UINT          uVirKey,
      DWORD         lParam,
      CONST LPBYTE  lpbKeyState)
```

引数

引数名	説明
hIMC	入力コンテキストのハンドル。
uVirKey	処理すべき仮想キー。
lParam	キーメッセージの lParam。
lpbKeyState	現在のキーボードの状態を含む 256 バイトの配列を指し示す。IME はこのキー状態の内容を変更すべきではない。

戻り値

関数が成功したら、戻り値は TRUE である。さもなければ戻り値は FALSE である。

コメント

システムはこの関数を呼び出すことで、キーが IME によって扱うか、そうでないかを決定する。もし、アプリがキーメッセージを取得する前にこの関数が TRUE を返せば、IME はそのキーを扱うだろう。そのときシステムは、ImeToAsciiEx を呼びだすだろう。もしこの関数が FALSE を返せば、システムは、そのキーが IME によって扱わないことを認識し、キーメッセージはアプリに送信されるだろう。

Windows 2000 において、IME_PROP_ACCEPT_WIDE_VKEY をサポートする IME に関しては、ImeProcessKey は、VK_PACKET を通じて SendInput API を使ってインジェクト (inject) された uVirKey の全部の 32 ビット値を受け取るだろう。uVirKey は、IME が ANSI 版であっても 16 ビット Unicode を上位ワードに含むだろう。

IME_PROP_ACCEPT_WIDE_VKEY をサポートしない IME に関しては、Unicode IME の ImeProcessKey は、上位ワードがゼロの VK_PACKET を受け取るだろう。それでも Unicode IME は、TRUE を返すことができ、ImeToAsciiEx はインジェクトされた Unicode と共に呼び出される。

ANSI IME の ImeProcessKey は何も受け取らないだろう。インジェクトされた Unicode は、ANSI IME が開かれていれば、破棄される。ANSI IME が閉じられていれば、インジェクトされた Unicode メッセージは、アプリの

キューにすぐに投稿される。

NotifyIME 関数

NotifyIME 関数は、与えられた引数に従って IME の状態を変更する。

```
BOOL NotifyIME(
      HIMC  hIMC,
      DWORD dwAction,
      DWORD dwIndex,
      DWORD dwValue)
```

引数

<table>
<tr><th>引数名</th><th>説明</th></tr>
<tr><td>hIMC</td><td>入力コンテキストのハンドル。</td></tr>
<tr><td>dwAction</td><td>以下は、アプリが引数 dwAction に指定できるコンテキスト項目である。
<table>
<tr><th>コンテキスト項目</th><th>説明</th></tr>
<tr><td>NI_OPENCANDIDATE</td><td>アプリは IME に候補リストを開かせる。もし IME が候補リストを開いたら、IME は WM_IME_NOTIFY（サブ関数は IMN_OPENCANDIDATE）メッセージを送信する。
<table>
<tr><th>引数</th><th>説明</th></tr>
<tr><td>dwIndex</td><td>開く候補リストのインデックス。</td></tr>
<tr><td>dwValue</td><td>使われない。</td></tr>
</table></td></tr>
<tr><td>NI_CLOSECANDIDATE</td><td>アプリは IME に候補リストを閉じさせる。もし IME が候補リストを閉じたら、IME は WM_IME_NOTIFY（サブ関数は IMN_CLOSECANDIDATE）メッセージを送信する。
<table>
<tr><th>引数</th><th>説明</th></tr>
<tr><td>dwIndex</td><td>閉じる候補リストのインデックス。</td></tr>
<tr><td>dwValue</td><td>使われない。</td></tr>
</table></td></tr>
<tr><td>NI_SELECTCANDIDATESTR</td><td>アプリは候補の一つを選択する。
<table>
<tr><th>引数</th><th>説明</th></tr>
<tr><td>dwIndex</td><td>選択する候補リストのインデックス。</td></tr>
<tr><td>dwValue</td><td>選択する候補リスト中の候補文字列のインデックス。</td></tr>
</table></td></tr>
<tr><td>NI_CHANGECANDIDATELIST</td><td>アプリは現在選択中の候補リストを変更する。</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>引数名</th><th>説明</th></tr>
<tr><td></td><td>
<table>
<tr><th>コンテキスト項目</th><th>説明</th></tr>
<tr><td></td><td>
<table>
<tr><th>引数</th><th>説明</th></tr>
<tr><td>dwIndex</td><td>選択する候補リストのインデックス。</td></tr>
<tr><td>dwValue</td><td>使われない。</td></tr>
</table>
</td></tr>
<tr><td>NI_SETCANDIDATE_PAGESTART</td><td>アプリは候補リストのページ開始インデックスを変更する。
<table>
<tr><th>引数</th><th>説明</th></tr>
<tr><td>dwIndex</td><td>変更する候補リストのインデックス。</td></tr>
<tr><td>dwValue</td><td>新しいページ開始インデックス。</td></tr>
</table>
</td></tr>
<tr><td>NI_SETCANDIDATE_PAGESIZE</td><td>アプリは、候補リストのページサイズを変更する。
<table>
<tr><th>引数</th><th>説明</th></tr>
<tr><td>dwIndex</td><td>変更する候補リストのインデックス。</td></tr>
<tr><td>dwValue</td><td>新しいページサイズ。</td></tr>
</table>
</td></tr>
<tr><td>NI_CONTEXTUPDATED</td><td>アプリかシステムが入力コンテキストを更新する。
<table>
<tr><th>引数</th><th>説明</th></tr>
<tr><td>dwIndex</td><td>dwValue の値が IMC_SETCONVERSIONMODE のとき、dwIndex は前の変換モードである。dwValue の値が IMC_SETSENTENCEMODE のとき、dwIndex は前のセンテンスモードである。その他の dwValue については dwIndex は使用されない。</td></tr>
<tr><td>dwValue</td><td>WM_IME_CONTROL メッセージで使われる以下の値のうちの一つ:
IMC_SETCANDIDATEPOS
IMC_SETCOMPOSITIONFONT
IMC_SETCOMPOSITIONWINDOW
IMC_SETCONVERSIONMODE
IMC_SETSENTENCEMODE
IMC_SETOPENSTATUS</td></tr>
</table>
</td></tr>
<tr><td>NI_COMPOSITIONSTR</td><td>アプリはコンポジション文字列を変更する。このアクションは、コンポジション文字列が入力コンテキストに存在するときのみ有効である。</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>引数名</th><th>説明</th></tr>
<tr><td></td><td>
<table>
<tr><th>コンテキスト項目</th><th>説明</th></tr>
<tr><td></td><td>
<table>
<tr><th>引数</th><th>説明</th></tr>
<tr><td>dwIndex</td><td>以下の値が dwIndex に与えられる:
CPS_COMPLETE
結果文字列としてコンポジション文字列を決めるために。
CPS_CONVERT
コンポジション文字列を変換するために。
CPS_REVERT
コンポジション文字列を元に戻すために。現在のコンポジション文字列はキャンセルされ、変換が解除された文字列がコンポジション文字列としてセットされる。
CPS_CANCEL
コンポジション文字列をクリアし、コンポジション文字列がない状態をセットするために。</td></tr>
<tr><td>dwValue</td><td>使われない。</td></tr>
</table>
</td></tr>
</table>
</td></tr>
<tr><td>dwIndex</td><td>uAction に依存する。</td></tr>
<tr><td>dwValue</td><td>uAction に依存する。</td></tr>
</table>

戻り値

関数が成功すれば、戻り値は TRUE である。さもなければ戻り値は FALSE である。

参照

ImmNotifyIME

ImeSelect 関数

ImeSelect 関数は、IME のプライベートなコンテキストを初期化・逆初期化する。

```
BOOL ImeSelect(
      HIMC hIMC,
      BOOL fSelect)
```

引数

引数名	説明
hIMC	入力コンテキストのハンドル。
fSelect	2 つのフラグが与えられる。TRUE は、初期化を表し、FALSE は逆初期化を表す (リソースの解放)。

戻り値

関数が成功すれば、戻り値は TRUE である。さもなければ戻り値は FALSE である。

ImeSetCompositionString 関数

ImeSetCompositionString 関数は、lpComp または lpRead 引数に含まれるデータとともに、IME のコンポジション文字列構造体をセットするために、アプリによって使われる。そのとき IME は WM_IME_COMPOSITION メッセージを生成する。

```
BOOL WINAPI ImeSetCompositionString(
      HIMC     hIMC,
      DWORD    dwIndex,
      LPCVOID  lpComp,
      DWORD    dwCompLen,
      LPCVOID  lpRead,
      DWORD    dwReadLen);
```

引数

<table>
<tr><th>引数名</th><th>説明</th></tr>
<tr><td>hIMC</td><td>入力コンテキストのハンドル。</td></tr>
<tr><td>dwIndex</td><td>以下の値が dwIndex に与えられる。
<table>
<tr><th>値</th><th>説明</th></tr>
<tr><td>SCS_SETSTR</td><td>アプリは、コンポジション文字列、読みの文字列、またはそれらの両方をセットする。少なくとも lpComp と lpRead 引数 の一つは、有効な文字列を指し示さなければならない。もしどちらかの文字列が長すぎれば、IME は切り捨てる。</td></tr>
<tr><td>SCS_CHANGEATTR</td><td>アプリは、コンポジション文字列、読みの文字列、またはそれらの両方の属性をセットする。少なくとも lpComp と lpRead 引数の一つは、有効な属性配列を指し示さねばならない。</td></tr>
<tr><td>SCS_CHANGECLAUSE</td><td>アプリは、コンポジション文字列、読みの文字列、またはそれらの両方の文節情報をセットする。少なくとも lpComp と lpRead 引数の一つは、有効な文節情報配列を指し示さねばならない。</td></tr>
<tr><td>SCS_QUERYRECONVERTSTRING</td><td>アプリは IME に RECONVERTSTRING 構造体を調節することを問い合わせる。もしアプリがこの値つきで ImeSetCompositionString 関数を呼び出したら、IME は、RECONVERTSTRING 構造体を調節する。そのときアプリは、調節した RECONVERTSTRING 構造体を SCS_RECONVERTSTRING つきでこの関数に渡すことができる。IME は WM_IMECOMPOSITION メッセージを生成しないだろう。</td></tr>
<tr><td>SCS_SETRECONVERTSTRING</td><td>アプリは IME に RECONVERTSTRING 構造体に含まれている文字列を再変換するか問い合わせる。</td></tr>
</table>
</td></tr>
</table>

引数名	説明
lpComp	更新されたコンポジション文字列を含むバッファへのポインタ。文字列の種類は、dwIndex の値によって決定される。
dwCompLen	コンポジションバッファの長さ(バイト単位)。
lpRead	更新された読みの文字列を含むバッファへのポインタ。　文字列の種類は、dwIndex の値によって決定される。もし dwIndex の値が、SCS_SETRRECONVERTSTRING か SCS_QUERYRECONVERTSTRING であれば、lpRead は、更新された読みの文字列を含む RECONVERTSTRING 構造体へのポインタになるだろう。もし選択中の IME が SCS_CAP_MAKEREAD の値を持っていれば、これは NULL でもよい。
dwReadLen	読みバッファの長さ(バイト単位)。

コメント

Unicode においては、dwCompLen と dwReadLen は、たとえ SCS_SETSTR が指定され、バッファが Unicode 文字列を含んでいても、バイト数でバッファの長さを指定する。

SCS_SETRECONVERTSTRING か SCS_QUERYRECONVERTSTRING のいずれかは SCS_CAP_SETRECONVERTSTRING プロパティを持つ IME に対してのみ使うことができる。このプロパティは、ImmGetProperty 関数を使うことで取得できる。

ImeToAsciiEx 関数

ImeToAsciiEx 関数は、引数 hIMC に基づき、IME 変換エンジンを通じて変換結果を生成する。

```
UINT ImeToAsciiEx(
      UINT             uVirKey,
      UINT             uScanCode,
      CONST LPBYTE     lpbKeyState,
      LPTRANSMSGLIST   lpTransMsgList,
      UINT             fuState,
      HIMC             hIMC)
```

引数

引数名	説明
uVirKey	翻訳する仮想キーコードを指定する。プロパティビット IME_PROP_KBD_CHAR_FIRST が ON であるとき、仮想キーの上位バイトは、エイド(aid)文字列コードである。Unicode においては、IME_PROP_KBD_CHAR_FIRST ビットが ON であるとき、uVirKey の上位ワードは、Unicode 文字コードを含む。
uScanCode	翻訳するキーのハードウェアスキャンコード
lpbKeyState	現在のキーボード状態を含む 256 バイトの配列を指し示す。IME はキー状態の内容を変更すべきではない。
lpTransMsgList	翻訳されたメッセージ結果を受け取るための TRANSMSGLIST バッファを指し示す。これは Windows 95/98 および Windows NT 4.0 IME の文書ではダブルワードバッファとして定義されていて、ダブルワードバッファバッファの形式は、 [翻訳されたメッセージバッファの長さ] [メッセージ 1] [wParam1] [lParam1] {[メッセージ 2] [wParam2] [lParam2] {...{...{...}}}} である。

引数名	説明
fuState	アクティブなメニューフラグ。
hIMC	入力コンテキストのハンドル。

戻り値

戻り値はメッセージの個数を表す。その個数が翻訳されたメッセージバッファより大きければ、翻訳されたメッセージバッファは充分ではない。システムは、翻訳メッセージを取得するために、hMsgBuf をチェックする。

コメント

Windows 2000 においては、wParam の下位バイトに VK_PACKET を使い、上位ワードには Unicode を使った、新しい 32 ビットの幅の仮想キーコードが SendInput を使ってインジェクトされうる。

IME_PROP_ACCEPT_WIDE_VKEY をサポートする ANSI IME については、ImeToAsciiEx は、1 文字に対して 16 ビットまでの ANSI コードを受け取るかもしれない。それは次のようにパックされる。文字は、VK_PACKET を通じて SendInput API からインジェクトされる。

ビット 24 – 31	ビット 16 – 23	ビット 8 – 15	ビット 0 – 7
予約済み	Trailing DBCS byte（もしあれば）	Leading byte	VK_PACKET

参照

ImmToAsciiEx

ImeRegisterWord 関数

ImeRegisterWord 関数は、この IME の辞書へ文字列を登録する。

```
BOOL WINAPI ImeRegisterWord(
      LPCTSTR  lpszReading,
      DWORD    dwStyle,
      LPCTSTR  lpszString)
```

引数

<table>
<tr><th>引数名</th><th>説明</th></tr>
<tr><td>lpszReading</td><td>登録される文字列の読みの文字列。</td></tr>
<tr><td>dwStyle</td><td>登録される文字列のスタイル。以下の値が与えられる。
<table>
<tr><th>値</th><th>説明</th></tr>
<tr><td>IME_REGWORD_STYLE_EUDC</td><td>文字列は EUDC の範囲内である。</td></tr>
<tr><td>IME_REGWORD_STYLE_USER_FIRST から IME_REGWORD_STYLE_USER_LAST まで</td><td>IME_REGWORD_STYLE_USER_FIRST から IME_REGWORD_STYLE_USER_LAST までの範囲の定数は、IME ISV のプライベートなスタイルとして使われる。IME ISV は自由に固有のスタイルを定義できる。例えば:
#define MSIME_NOUN \
(IME_REGWORD_STYLE_USER_FIRST)</td></tr>
</table>
</td></tr>
</table>

引数名	説明	
	値	説明
		#define MSIME_VERB \ (IME_REGWORD_STYLE_USER_FISRT +1)
lpszString	登録される文字列。	

戻り値

関数が成功すれば、戻り値は TRUE である。さもなければ、戻り値は FALSE である。

ImeUnregisterWord 関数

ImeUnregisterWord 関数は、この IME の辞書から登録済みの文字列を除去する。

```
BOOL WINAPI ImeUnregisterWord(
      LPCTSTR  lpszReading,
      DWORD    dwStyle,
      LPCTSTR  lpszString)
```

引数

引数名	説明
lpszReading	登録済みの文字列の読みの文字列。
dwStyle	登録済みの文字列のスタイル。dwStyle の説明については ImeRegisterWord 関数を参照されたい。
lpszString	登録を解除する文字列。

戻り値

関数が成功すれば、戻り値は TRUE である。さもなければ、戻り値は FALSE である。

ImeGetRegisterWordStyle 関数

ImeGetRegisterWordStyle 関数は、この IME において利用可能なスタイルを取得する。

```
UINT WINAPI ImeGetRegisterWordStyle(
      UINT        nItem,
      LPSTYLEBUF  lpStyleBuf)
```

引数

引数名	説明
nItem	バッファが所有するスタイルの最大個数。
lpStyleBuf	埋められるバッファ。

戻り値

戻り値は、バッファへコピーされたスタイルの個数である。もし nItems がゼロであれば、戻り値は、この IME において、利用可能なすべてのスタイルを受け取るのに必要なバッファサイズ（配列要素数）である。

ImeEnumRegisterWord 関数

ImeEnumRegisterWord 関数は、指定された読みの文字列、スタイル、および登録済み文字列データと共に登録済み文字列の情報を列挙する。

```
UINT WINAPI ImeEnumRegisterWord(
      HKL                   hKL,
      REGISTERWORDENUMPROC  lpfnEnumProc,
      LPCTSTR               lpszReading,
      DWORD                 dwStyle,
      LPCTSTR               lpszString,
      LPVOID                lpData)
```

引数

引数名	説明
hKL	入力言語ハンドル。
lpfnEnumProc	コールバック関数のアドレス。
lpszReading	列挙される読みの文字列を指定する。もし lpszReading が NULL であれば、ImeEnumRegisterWord は、指定された dwStyle と lpszString 引数にマッチする、すべての利用可能な読みの文字列を列挙する。
dwStyle	列挙対象のスタイルを指定する。もし dwStyle が NULL ならば、ImeEnumRegisterWord は、指定された lpszReading と lpszString にマッチする、すべての利用可能なスタイルを列挙する。
lpszString	列挙対象の登録済み文字列を指定する。もし lpszString が NULL であれば、ImeEnumRegisterWord は、指定された lpszReading と dwStyle 引数にマッチする、すべての登録済み文字列を列挙する。
lpData	アプリが供給するデータのアドレス。

戻り値

関数が成功したら、戻り値は、コールバック関数が返した最後の値である。その意味は、アプリによって定義される。

コメント

もし lpszReading、dwStyle、そして lpszString のすべてが NULL であれば、ImeEnumRegisterWord は、IME 辞書のすべての登録済み文字列を列挙する。もし入力引数のうち2つが NULL であれば、ImeEnumRegisterWord は、第三の引数にマッチするすべての登録済み文字列を列挙する。

ImeGetImeMenuItems 関数

ImeGetImeMenuItems 関数は、IME メニューに登録済みのメニュー項目たちを取得する。

```
DWORD WINAPI ImeGetImeMenuItems(
      HIMC                hIMC,
      DWORD               dwFlags,
      DWORD               dwType,
      LPIMEMENUITEMINFO   lpImeParentMenu,
      LPIMEMENUITEMINFO   lpImeMenu,
      DWORD               dwSize)
```

引数

<table>
<tr><th>引数名</th><th>説明</th></tr>
<tr><td>hIMC</td><td>lpMenuItem は、入力コンテキストに関連するメニュー項目たちを含む。</td></tr>
<tr><td>dwFlags</td><td>以下のビット組み合わせからなる。

<table><tr><th>ビット</th><th>説明</th></tr><tr><td>IGIMIF_RIGHTMENU</td><td>もしこのビットが 1 なら、この関数は、右クリックコンテキストメニューに対するメニュー項目たちを返す。</td></tr></table></td></tr>
<tr><td>dwType</td><td>以下のビット組み合わせからなる。

<table><tr><th>ビット</th><th>説明</th></tr><tr><td>IGIMII_CMODE</td><td>変換モードに関するメニュー項目たちを返す。</td></tr><tr><td>IGIMII_SMODE</td><td>センテンスモードに関するメニュー項目たちを返す。</td></tr><tr><td>IGIMII_CONFIGURE</td><td>IME の設定に関するメニュー項目たちを返す。</td></tr><tr><td>IGIMII_TOOLS</td><td>IME ツールに関するメニュー項目たちを返す。</td></tr><tr><td>IGIMII_HELP</td><td>IME ヘルプに関するメニュー項目たちを返す。</td></tr><tr><td>IGIMII_OTHER</td><td>その他に関するメニュー項目たちを返す。</td></tr><tr><td>IGIMII_INPUTTOOLS</td><td>文字入力の拡張した方法を提供する IME の入力ツールに関するメニュー項目たちを返す。</td></tr></table></td></tr>
<tr><td>lpImeParentMenu</td><td>fType に MFT_SUBMENU があるとき、IMEMENUINFO 構造体へのポインタ。ImeGetImeMenuItems は、このメニュー項目のサブメニュー項目たちを返す。もしこれが NULL ならば、lpImeMenu はトップレベルのメニュー項目たちを含む。</td></tr>
<tr><td>lpImeMenu</td><td>メニュー項目たちの内容を受け取るバッファへのポインタ。このバッファは、IMEMENUITEMINFO 構造体の配列である。もしこれが NULL ならば、ImeGetImeMenuItems は、登録済みメニュー項目の個数を返す。</td></tr>
<tr><td>dwSize</td><td>IMEMENUITEMINFO 構造体を受け取るバッファのサイズ。</td></tr>
</table>

戻り値

戻り値は、lpImeMenu の中へセットされたメニュー項目の個数である。もし lpImeMenu が NULL であれば、ImeGetImeMenuItems は指定された hKL に登録済みのメニュー項目の個数を返す。

コメント

ImeGetImeMenuItems は Windows 98 と Windows 2000 で登場した新しい関数である。