GetIText

Obtain a copy of the text of an editText item

#include <Dialogs.h> **Dialog Manager**

```
void      GetIText(iHandle, textStr );
Handle    iHandle ;     Handle of an editText or statText item
Str255    textStr ;     address of a 256-byte buffer to hold text
```

GetIText obtains a copy of the text currently stored in an editText or statText item in a dialog. It can be used to obtain a Pascal-style string from a TextEdit text handle.

iHandle is handle obtained from a previous call to **GetDItem**. It is actually the hText field of a TERec, as used by TextEdit.

textStr is the address of a buffer to hold the returned text. Upon return, it will contain a pascal-style length-prefixed string of the current value of the dialog item.

Returns: none

Notes: **GetIText** lets you know the result when a user edits an editText item. Precede this with a call to **GetDItem** to obtain a valid value for *iHandle* .

Use **SetIText** to initialize the value before calling **ModalDialog** or **DialogSelect**.

The following example assumes you have a resource that prompts for a new window title and that item 5 of that dialog is an editText item.

Example

```
#include <Dialogs.h>

DialogPtr   myDlg;
Handle      iHndl;
short       iType, itemHit;
Rect        iRect;
Str255      theTitle;                /* a.k.a: char theTitle[256] */

myDlg = GetNewDialog( MYDLG_ID,. 0, (WindowPtr)-1);
GetDItem( myDlg, 5, &iType, &iHndl, &iRect);
SetIText( iHndl, "\pUntitled" );
SelIText ( myDlg, 5, 0, 32767 );  /* pre-select all for convenience */

/* ----- loop to handle the dialog ----- */
do {
    ModalDialog( 0, &itemHit );
    switch ( itemHit ) {
    case 5:
        GetDItem( myDlg, 5, &iType, &iHndl, &iRect);
        GetIText( iHndl, theTitle );
        SetWTitle( myWindow, theTitle );
```

```
            break;
        case 6:

            /*  ...etc... */

    } while ( (itemHit != ok) && (itemHit != cancel) );


    DisposDialog( myDlg);
```