Función GetProcAddress (libloaderapi.h)

Artículo • 05/24/20222 minutos para leer

Recupera la dirección de una función exportada (también conocida como procedimiento) o variable de la biblioteca de vínculos dinámicos (DLL) especificada.

Sintaxis

```
C++

FARPROC GetProcAddress(
  [in] HMODULE hModule,
  [in] LPCSTR  lpProcName
);
```

Parámetros

`[in] hModule`

Identificador del módulo DLL que contiene la función o variable. La función LoadLibrary , LoadLibraryEx , LoadPackagedLibrary o GetModuleHandle devuelve este identificador.

La función **GetProcAddress** no recupera direcciones de módulos que se cargaron con el indicador **LOAD_LIBRARY_AS_DATAFILE** . Para obtener más información, consulte LoadLibraryEx .

`[in] lpProcName`

El nombre de la función o variable, o el valor ordinal de la función. Si este parámetro es un valor ordinal, debe estar en la palabra de orden inferior; la palabra de orden superior debe ser cero.

Valor de retorno

Si la función tiene éxito, el valor devuelto es la dirección de la función o variable exportada.

Si la función falla, el valor devuelto es NULL. Para obtener información de error ampliada, llame a GetLastError .

Observaciones

La ortografía y las mayúsculas y minúsculas de un nombre de función señalado por *lpProcName* deben ser idénticos a los de la instrucción **EXPORTS** del archivo de definición de módulo (.def) de la DLL de origen. Los nombres de las funciones exportados pueden diferir de los nombres que usa al llamar a estas funciones en su código. Esta diferencia está oculta por las macros utilizadas en los archivos de encabezado del SDK. Para obtener más información, consulte Convenciones para prototipos de funciones .

The *lpProcName* parameter can identify the DLL function by specifying an ordinal value associated with the function in the **EXPORTS** statement. **GetProcAddress** verifies that the specified ordinal is in the range 1 through the highest ordinal value exported in the .def file. The function then uses the ordinal as an index to read the function's address from a function table.

If the .def file does not number the functions consecutively from 1 to *N* (where *N* is the number of exported functions), an error can occur where **GetProcAddress** returns an invalid, non-NULL address, even though there is no function with the specified ordinal.

If the function might not exist in the DLL module—for example, if the function is available only on Windows Vista but the application might be running on Windows XP—specify the function by name rather than by ordinal value and design your application to handle the case when the function is not available, as shown in the following code fragment.

C++

```
typedef void (WINAPI *PGNSI)(LPSYSTEM_INFO);

// Call GetNativeSystemInfo if supported or GetSystemInfo otherwise.

   PGNSI pGNSI;
   SYSTEM_INFO si;

   ZeroMemory(&si, sizeof(SYSTEM_INFO));

   pGNSI = (PGNSI) GetProcAddress(
      GetModuleHandle(TEXT("kernel32.dll")),
      "GetNativeSystemInfo");
   if(NULL != pGNSI)
   {
      pGNSI(&si);
   }
   else
   {
```

```
        GetSystemInfo(&si);
    }
```

For the complete example that contains this code fragment, see Getting the System Version.

Examples

For an example, see Using Run-Time Dynamic Linking.

Requirements

Minimum supported client	Windows XP [desktop apps | UWP apps]
Minimum supported server	Windows Server 2003 [desktop apps | UWP apps]
Target Platform	Windows
Header	libloaderapi.h (include Windows.h)
Library	Kernel32.lib
DLL	Kernel32.dll

See also

Dynamic-Link Library Functions

FreeLibrary

GetModuleHandleGetModuleHandle

Cargar biblioteca

CargarLibreríaEx

LoadPackagedLibrary

Vinculación dinámica en tiempo de ejecución