# Ejemplo CreateProccess

La función CreateProcess crea un nuevo proceso, que se ejecuta independientemente del proceso de creación. Sin embargo, para simplificar, la relación se conoce como una relación padre-hijo.

El siguiente código muestra cómo crear un process.Process handle no heredable:

#include <windows.h>

#include <stdio.h>

#include <tchar.h>

void \_tmain( int argc, TCHAR \*argv[] )

{

STARTUPINFO si;

PROCESS\_INFORMATION pi;

ZeroMemory( &si, sizeof(si) );

si.cb = sizeof(si);

ZeroMemory( &pi, sizeof(pi) );

if( argc != 2 )

{

printf("Usage: %s [cmdline]\n", argv[0]);

return;

}

// Inicia de proceso

if( !CreateProcess( NULL, //

argv[1], // linea de comando

NULL, // proceso no heredable

NULL, // proceso no heredable

FALSE, // Set FALSE

0, // No se crean Banderas

NULL, // Usa el bloqueo de la función padre

NULL, // Usa el directorio de la función padre

&si, // Apuntador a la estructura STARTUPINFO

&pi ) // Apuntador a la estructura PROCESS\_INFORMATION

)

{

printf( "CreateProcess failed (%d).\n", GetLastError() );

return;

}

// espera hasta que la función hija exista.

WaitForSingleObject( pi.hProcess, INFINITE );

// cierre del proceso

CloseHandle( pi.hProcess );

CloseHandle( pi.hThread );}

Si CreateProcess tiene éxito, devuelve una estructura PROCESS\_INFORMATION que contiene controladores e identificadores para el nuevo proceso y su hilo principal. Los identificadores de subproceso y proceso se crean con derechos de acceso completos, aunque el acceso puede restringirse si especifica descriptores de seguridad. Cuando ya no necesite estas asas, ciérrelas utilizando la función CloseHandle.

También puede crear un proceso utilizando la función CreateProcessAsUser o CreateProcessWithLogonW. Esto le permite especificar el contexto de seguridad de la cuenta de usuario en la que se ejecutará el proceso.