República Bolivariana de Venezuela
Universidad Nacional Experimental Politécnica
"Antonio José de Sucre"
Vice Rectorado Barquisimeto
Departamento de Ingeniería Electrónica

Practica 4
laboratorio de diseño de sistemas de computación

Integrantes:
Gerardo Alfonzo Campos Fonseca
V. 27085179
José Andrés Cortez Teran
V. 26540824

Barquisimeto, Julio del 2021

Índice

Índice de figuras

1. Programa DOS

El código comentado se adjunta en el archivo (dos/lectura.asm)

2. Programa consola

El código comentado se adjunta en el archivo (consola/lectura.asm)

inicialmente es directivas e inclusión de librerías, luego viene el prototipo de funciones y el macro que permiten el controlar la entrada y salida de datos. Cabe resaltar que se utilizo como modelo la practica 2, posteriormente, vienen los segmentos de data y código:

En el segmento de data inicializada "**.DATA**" se encuentran:

- "Msg1" el cual es un mensaje que indica al usuario cuando debe ingresar un path o el nombre del archivo.
- "Msg4" mensaje que indica la salida del programa.
- "CRLF" string dedicada a dejar una linea en blanco.

En el segmento de data sin inicializar "**.DATA?**" se encuentran:

- "inbuf" buffer para la lectura de datos.
- "hfile" "puntero" a la dirección de la fila que se leerá.
- "FileSize" espacio para el tamaño, en bytes, del fichero.
- "hMem" "puntero" a la memoria creada con el contenido del fichero.
- "BytesRead" cantidad de bytes que se leyeron.

Posteriormente, comienza el código que básicamente es ejecutar "syscall" a través de la directiva "INVOKE".

Primero se pide por consola y se lee por la misma el nombre o path de la fila que se abrirá. Una vez recibido se usa la api de windows con "CreateFile" para abrir la fila Microsoft, 2017, este devuelve la dirección en "eax" por lo tanto se mueve a hfile. Luego, se utiliza la api nuevamente con "GetFileSize" Microsoft, 2021c y se mueve el valor devuelto de "eax" a "FileSize", se incrementa el valor de "eax" para al pedir memoria dinámicamente, necesario si la fila es muy grande, poder agregar el carácter de terminación y usar de forma segura la función de escritura.

Se pide la memoria dinámica con la api "GlobalAlloc" Microsoft, 2021d, se guarda en "hMem", se le da el valor del tamaño de la fila de nuevo a "eax" para poder usarlo como puntero y darle el valor "0" (carácter de terminación) a "hMem" (espacio para el contenido de la fila + carácter terminación).

Finalmente se lee el contenido con la función de la api ReadFile Microsoft, 2021f se cierra la fila con “CloseHandle” Microsoft, 2021a luego se imprime por pantalla el contenido de la fila. Se libera la memoria con “GlobalFree” y para que se visualice correctamente el resultado, se imprime un salto de linea y se manda a leer esperando por un enter para finalizar.

El programa termina con “Invoke ExitProcess” para terminar la ejecución correctamente.

3. Programa ventana

El código comentado se adjunta en el archivo (ventana/DIALOG.asm y ventana/recursos.rc)

Para este programa se utilizo como base los tutoriales de “Iczelion” Iczelion, s.f. mas específicamente el 11-1, ya que en la carpeta 11-3 estaba prácticamente el código hecho y se quería intentar hacer a cuenta propia.

Esta estructura da acceso a una ventana principal con un submenu superior, desde el cual se puede cerrar el programa y se puede abrir una ventana emergente con un cuadro de texto y 2 botones. Aquí es en donde se basaron todas las modificaciones.

Para el funcionamiento del código, se incluyo la librería “masm32” y se declararon variables adicionales en la sección “.data?”

- “inbuf” buffer para la lectura de datos.
- “hfile” “puntero” a la dirección de la fila que se leerá.
- “FileSize” espacio para el tamaño, en bytes, del fichero.
- “hMem” “puntero” a la memoria creada con el contenido del fichero.
- “BytesRead” cantidad de bytes que se leyeron.

Cabe resaltar que el programa cuenta de estructuras que no se modificaron ya que parecían apropiadas, estas son constantes y el nombre de las ventanas. El segmento de código hace uso de 4 invoke, el primero es para poder manejar los módulos “GetModuleHandle”, el segundo “GetCommandLine” permite hacer uso de una “CMD”, el tercero, “WinMain” le da el control del programa a un subproceso descrito posteriormente y el final para cerrar los procesos.

“Invoke WinMain” le da el control del programa a un subproceso y le pasa como parámetro las variables hInstance, NULL, CommandLine, SW_SHOWDEFAULT, con las cuales se creara la ventana main, estas se inicializan y posteriormente se crea un objeto, struct, y a el se le asignan los, valores, variables y procedimientos requeridos, el tamaño de la ventana, el color, los nombres de los objetos gráficos estilo de cursor entre otros, el mas importante es “mov wc.lpfnWndProc, OFFSET WndProc”, aquí se le da a la clase la dirección de otro subproceso definido y explicado posteriormente, en el cual se hará todo el proceso de búsqueda, lectura y escritura del fichero.

Después de estas definiciones se usan las directivas “INVOKE ShowWindow” y “INVOKE UpdateWindow” que crean una ventana y la actualizan con toda la información proveída desde la clase.

Posteriormente, se entra en un ciclo while true el cual hace la función de una especie de “framework” ya que mantiene el programa en ejecución a la espera de que un evento suceda y sea necesario realizar alguna acción, en este caso “GetMessage” se encarga de verificar si ocurre algún evento, “eax” guarda cualquier posible cambio y si es 0 indica la finalización del programa.

“WindProc” es otro proceso construido y es llamado automáticamente por la clase (ya que se le asigno su dirección, y es llamado cuando es requerido) y, básicamente, se encarga de revisar si algún evento sucedió, y si sucedió enviar los mensajes con “PostQuitMessage” si se destruyo la ventana, para finalizar el proceso mencionado anteriormente (asigna el valor 0 a “eax”), cuando este no es el caso, se trata de una serie de condicionales anidados para ir revisando todos los posibles eventos y tomar las medidas necesarias en cada caso. Estos son:

- Si “uMsg==WM_DESTROY”, condición de terminación, se envía el mensaje 0 (null) en “eax” al “framework” (“.while true”) para terminar su ejecución.
- Si no y “uMsg==WM_COMMAND”, hace la comparación de la entrada al submenu.
    - Si “ax==IDM_ABOUT” crea la nueva ventana y le da la dirección a otro proceso definido posteriormente donde se realizan todo lo relacionado a el el “InputBox” y los botones que permiten ingresar el nombre del fichero. Cabe resaltar que “IDM_ABOUT” no es mas que el identificador de la ventana emergente creada que contendrá todos los elementos anteriormente nombrados, este nombre puede ser cambiado, requeriría modificar aquí y en el archivo “recursos.rc” ya que aquí se define el “InputBox”.
    - Si no, la única otra opción disponible implica cerrar, así que llama al proceso “DestroyWindow”
- En el caso que no se haya cumplido nada de esto, llama a “DefWindowProc” para definir la ventana y regresa el control.

El proceso “DlgProc” es igualmente una secuencia de condicionales anidados:

- “.if iMsg==WM_INITDIALOG” se inicializa la ventana de diálogos y se centra la atención en “eax”. “inputbox”
- “.elseif iMsg==WM_CLOSE” el mensaje es para cerrar la pestaña. se clickeo el salir de la ventana.
- “.elseif iMsg==WM_COMMAND” Ha ocurrido algún evento dentro de la ventana, particularmente se trabajan los botones:
    - “.if eax==IDC_EXIT” se presiono el botón “salir” y se cierra la ventana, mandando un mensaje de terminación con “SendMessage”.
    - “.elseif eax==IDC_BUTTON” se presiono el botón para la lectura del texto. entonces se hace:
        - “GetDlgItemText” para obtener el texto en el “InputBox” y guardarlo en “inbuf”.
        - “CreateFile” para abrir la fila especificada en “inbuf”.
        - “GetFileSize” tomar el tamaño de la fila.

- “GlobalAlloc” reservar memoria dinámica. Se hacen las mismas consideraciones que en el programa consola.
- “ReadFile” lee la fila.
- “CloseHandle” cierra la fila.
- “StdOut” escribe el contenido en “hMem”.
- “MessageBox” Manda una ventana emergente con el contenido del fichero.
- “GlobalFree” libera la memoria pedida.

- Si no se ejecuta ninguna de estas opciones, se devuelve “False” en “eax”.

Si algo de eso ocurrió, devuelve “True” en “eax”.

Con ese ultimo procedimiento finaliza todo el código.

Referencias Bibliográficas

Iczelion. (s.f.). http://www.movsd.com/icz.htm

Irvine, K. R. (2002). Assembly Language for Intel Assembly Language for Intel-Based Computers, 4 Computers, 4th Edition Edition [fecha de consulta: 19/7/2021]. https://www.csie.ntu.edu.tw/~acpang/course/asm_2004/slides/chapt_08Solve.pdf

Microsoft. (2017). CreateFileA function (fileapi.h) [fecha de consulta: 22/07/2021]. https://docs.microsoft.com/en-us/windows/win32/api/fileapi/nf-fileapi-createfilea

Microsoft. (2021a). CloseHandle function (handleapi.h) [fecha de consulta: 22/7/2021]. https://docs.microsoft.com/en-us/windows/win32/api/handleapi/nf-handleapi-closehandle

Microsoft. (2021b). GetDlgItemTextA function (winuser.h) [fecha de consulta: 24/7/2021]. https://docs.microsoft.com/en-us/windows/win32/api/winuser/nf-winuser-getdlgitemtexta

Microsoft. (2021c). GetFileSize function (fileapi.h) [fecha de consulta: 22/7/2021]. https://docs.microsoft.com/en-us/windows/win32/api/fileapi/nf-fileapi-getfilesize

Microsoft. (2021d). GlobalAlloc function (winbase.h) [fecha de consulta: 22/7/2021]. https://docs.microsoft.com/en-us/windows/win32/api/winbase/nf-winbase-globalalloc

Microsoft. (2021e). How to get the string value of editbox in the variable in win32 application? [fecha de consulta: 24/7/2021]. https://social.msdn.microsoft.com/Forums/vstudio/en-US/17c2d97a-011b-4fb1-9563-4f095d9321e4/how-to-get-the-string-value-of-editbox-in-the-variable-in-win32-application?forum=vcgeneral

Microsoft. (2021f). ReadFile function (fileapi.h) [fecha de consulta: 22/7/2021]. https://docs.microsoft.com/en-us/windows/win32/api/fileapi/nf-fileapi-readfile

OVERFLOW, S. (2015). Getting string input and displaying input with DOS interrupts MASM [fecha de consulta: 19/7/2021]. https://stackoverflow.com/questions/29504516/getting-string-input-and-displaying-input-with-dos-interrupts-masm

to 2012, T. M. F. A. 2. (s.f.). how to read from file in masm [fecha de consulta: 22/7/2021]. http://www.masmforum.com/board/index.php?topic=16266.0