**Simulation of User Input Program: Part 1**

I am Alvin Alexander, eighth standard student of Salvation Army HSS, Kowdiar, Thiruvananthapuram.

In the previous session, we learned about The Windows API and its functional areas. […] This session and subsequent sessions will focus on developing simple application programs using WinAPI. The application program that we will develop uses the User Interface functional area of WinAPI.

The basic idea of the program is to automatically type sentences in notepad. This basic idea can be extended to make a program that includes more features, though we won’t cover that here. Our aim, right now, is to get familiar with WinAPI and its standards.

How can we develop this?

We can use two different ways to develop this program: by using SendInput() to synthesize keystrokes or by using SendMessage() / PostMessage() to send / post WM\_SETTEXT window message to the notepad edit window. The only difference between SendMessage() function and PostMessage() function is, the former waits for the window to process the message before returning, while the latter returns immediately (placing the message in a queue, without waiting for it to be processed). Here I am going with the second option and I am using SendMessage() function. This function has 4 parameters: Window Handle, Message to send, WPARAM and LPARAM. Here, message to send is WM\_SETTEXT and for this, we can omit the third parameter. We have to give the text to send in the fourth parameter. The Window Handle has to be the edit window of notepad. So we have to use FindWindow() to get a window handle of notepad, then we have to use FindWindowEx() to get the child window, that is the edit window. Once we have that sorted, we need logic to handle the “Automatic Typing”. Because, the SendMessage function won’t automatically type our sentence. Here, we need the ‘for’ loop that increments ‘i’ until our sentence is fully typed.

The logic is pretty simple. We need two variables of LPSTR type (CHAR type will also work just fine). The first variable will be our sentence to type and the second variable is used to store the transferred characters from the first var., mimicking a user typing the sentence. So, every repetition will include the next character and the previous character; it appears as if it is being typed automatically.

The code to the program is as follows (doesn’t include the main function where this is implemented in)**:**

HWND hWnd = FindWindow("Notepad", NULL);

HWND editWnd = FindWindowEx(hWnd, NULL, "Edit", NULL);

LPSTR message = "Hello! How Are You? Hope You Are Well.";

INT len = lstrlen(message);

LPSTR msg = HeapAlloc(GetProcessHeap(), HEAP\_ZERO\_MEMORY, len + 1);

for (INT j = 0; j < (len + 1); j++) {

msg[j] = message[j];

Sleep(200);

SendMessage(editWnd, WM\_SETTEXT, 0, msg);

}

To be continued in the next session…