Basic functions of ncurses (Before menu.h)

Funciones iniciales

Deshabilitar espera del terminal respecto al buffer

En C, el terminal almacena informacion dentro del buffer: todos los caracteres que el usuario ingrese hasta que detecte un salto de linea (tecla enter) o un “carriage return”. Para desactivar esa opcion de la terminal, existen dos maneras importantes:

- **Raw():** Los caracteres se pasan directamente a la terminal sin general ninguna alerta o esperar una tecla especial.
- **Cbreak():** Los caracteres que espera la terminal para pasar a la siguiente parte del programa (tecla enter y “carriage return”), son interpretados como cualquier otro caracter.

Se recomienda trabajar con raw por permitir mayor control con lo que el usuario ingresa.

Echoing

Se encarga de mostrar en la terminal cada caracter que el usuario ingrese en el programa. Existen dos funciones principales para trabajar con el echo() && noecho().

- **Echo()** = Esta activada por defecto en C, se encarga de mostrar en la terminal todo lo que el usuario esta escribiendo en la terminal.

- **Noecho()** = Desactiva el echo, es decir, el usuario ya no vera lo que escriba en la terminal.

Lectura de teclas especiales

En C, en la mayoria de los casos, teclas de funcion como F1, F2, Arrow keys, etc. No estan disponibles para ser detectadas e interpretadas en la terminal.

Keypad() = Se encarga de este aspecto del programa. Permite la lectura de estas teclas especificas dentro del programa. Casi todos los menus interactivos en este lenguaje, requieren que Keypad ese

activada. Una forma practica para ponerlo en funcion es: keypad(stdscr, TRUE).

Control parcial del tiempo

Muy similar a como funciona cbreak(): halfdelay() se encarga de recibir todo tipo de caracter dentro del sistema con la diferencia de esperar x cantidad de segundos para pasar a la siguiente parte del codigo. Si no recibe ningun tipo de dato en lo que pasa el tiempo, devuelve el valor de tiempo que transcurrio. Es util cuando se solicita alguna entrada al usuario y, si este no responde en el tiempo esperado, el programa puede pasar a seguir con el resto del codigo.

Algunas otras funciones:

- **#include <ncurses.h>** = Libreria principal con la que se trabajara.
- **initscr();** = Activa el modo curses (activando todas las funciones de la biblioteca ncurses.h).
- **Printw()** = Imprime informacion en cualquier parte de la terminal donde el desarrollador le indique, se diferencia de printf ya que esta ultima lo muestra segun la terminal indique.
- **Refresh()**= Se actualiza la terminal modificada con Ncurses y muestra la posicion nueva en donde se encuentre el usuario.
- **Attron()** = Activa atributos visuales al texto que se imprime en la terminal.
- **Attroff()** = Desactiva los atributos visuales al texto que se imprime en la terminal.
- **Getch** = Espera la entrada de algun caracter del usuario.
- **Endwin** = Finaliza el modo curses.

Codigo de referencia:

```
#include <ncurses.h>

int main(){
    int ch;
```

```
    initscr();              /* Activamos el modo curses          */
    raw();                  /* Se desactiva el buffer de terminal    */
    keypad(stdscr, TRUE);       /* Activamos teclas de funcion       */
    noecho();           /*No se muestra nada en la pantalla */


        printw("Imprime cualquier cosa en negrita\n");
    ch = getch();           /* Si no activamos raw, tendremos que oprimir enter antes de mandarlo al programa*/
    if(ch == KEY_F(1))      /* Sin keypad no se activa*/
        printw("F1 Key pressed");/* Le dice de manera educada al usuario que esta bien baboson */
                    /* Sin noecho, algunos molestos caracteres podrian haber sido impresos en diferentes partes de la terminal    */
    else
    {   printw("The pressed key is ");
        attron(A_BOLD); //Aplica negrita al caracter
        printw("%c", ch);
        attroff(A_BOLD);//Desactiva la negrita al caracter


    }
    refresh();          /* Lo imprime en la pantalla real*/
        getch();            /*Espera la entrada del usuario */
    endwin();           /* Apaga el modo curses              */

    return 0;
}
```

Funciones de salida

Funciones de impresion

La familia curses, permite tres salidas de impresion disponibles a las que se les puede aplicar formato de salida

- **addch() class:** Imprime un solo caracter, con atributos aplicados de manera preestablecida.

- **printw() class:** Imprime una salida con formato muy similar a printf() y con atributos.
- **addstr() class:** Imprime strings (solo imprime cadenas ya construidas) con atributos.

Cada una se puede usar indistintamente ya que manejan sintaxis similares.

Funciones de posicion con addch():

mvaddch() = Posiciona el cursor en un punto establecido por el desarrollador para luego imprimir en en esa posicion.

Ejemplo:

move(row,col); /* Mueve la posicion de impresion en fila, columna*/

addch(ch); //Imprime el caracter en esa posicion.

Otra forma de hacer mas “pro” es:

mvaddch(row,col,ch); //Hace todo en la misma linea de codigo

waddch() = Es muy similar a addch(), solo que, imprime el caracter en una ventana especificada, mientras que addch lo hace en la ventana (comunmente terminal) por default.

mvwaddch() = Es una combinacion tanto de waddch como de mvaddch. Lo que hace es imprimer el caracter en: 1 La coordenadas predefinidas y 2 la ventana especificada.

Esta serie de parametros nos sirven para imprimer solo un caracter, para hacerlo mediante strings ya sea que el usuario dio, o que fueron preestablecidos en primer lugar, podemos seguir otras reglas.

Funciones de posicion con printw():

mvprintw() = Imprime la cadena de caracteres en un punto especifico de la terminal, se usa como: mvprintw(FILA, COL, "string");

Wprintw() = indica en que ventana se imprimira el string.

Mvwprintw() = indica el punto y la ventana en donde se imprime el string.

Funciones con addstr():

Como se menciono antes, addstr() se emplea para imprimir un string preestablecido por el desarrollador. Funciona de manera similar a addch().

Las funciones mas notorias de este parametro son:
mvaddstr() = Imprime el string en un punto en especifico.

Waddstr() = Imprime el string en una ventana en especifico

Mvwaddstr() = Imprime el string en un punto y ventana en especifico

Addnstr() = Imprime un string limitado a n cantidad de caracteres.

Codigo de referencia:

```
#include <ncurses.h>

#include <string.h>

int main() {

 char mesg[]="Just a string"; // Mensaje que aparece en la pantalla

 int row,col;   //indica el numero de filas y columnas

 initscr();  //Inicia el modo curses

 getmaxyx(stdscr,row,col);      /* Recibe el numero de filas y columnas en la terminal principal
```

```
 mvprintw(row/2,(col-strlen(mesg))/2,"%s",mesg); //imprime el mensaje al centro de la pantalla

 mvprintw(row-2,0,"This screen has %d rows and %d columns\n",row,col);

 printw("Try resizing your window(if possible) and then run this program again");

 refresh();

 getch();

 endwin();


 return 0;

}
```

Funciones de entrada

Para la entrada de informacion al programa, existen tres funciones principales que cuentan con sus sub-funciones derivadas segun sea el caso. Estas son:

- getch() class: Recibe un solo caracter //Esta se utilizara para el menu
- scanw() class: Recibe una cadena de caracteres
- getstr() class: Recibe strings

Existen importantes consideraciones a tomar en cuenta. Por ejemplo, si no usas cbreak, ncurses no va a leer y guardar la informacion que el usuario ingrese al programa hasta que detecte un enter o la tecla ajustada a esa funcion. Por ello se recomienda utilizar cbreak para poder estar enviando teclas al programa y que este las pueda ir interpretando en tiempo real. Esto ira muy de la mano con el menu interactivo, en donde para controlar desplazamientos se requerira:

1. Que el programa lea la tecla y la utilice en el programa.
2. Que no se muestre lo que se presiono, pero que si se desplace en el menu.
3. Permitir que el usuario pueda ver lo que escribe y desactivar teclas de funcion para que el usuario sea capaz de escribir un string sin utilizar teclas de funcion.

En terminos de ncurses se traduce a:

- Activar keypad, noecho y cbreak. //Para utilizar el menu
- Desactivarlos y activar parametros de stdio.h /*Para ingresar informacion y guardarla en el programa */

Es importante aclarar que getstr es igual al gets estandar de C, por ello se recomienda mejor utilizar “getnstr()”, para recibir un string por parte del usuario y limitarlo a n caracteres.

La sintaxis de getnstr() es:

char nombre[10];

getnstr(nombre, 10); /*Si el usuario excede los diez caracteres, los demas no se leen */

Codigo de referencia:

```
#include <ncurses.h>                /* ncurses.h includes stdio.h */
#include <string.h>

int main() {
 char mesg[]="Ingresa un string: ";          /* Mensaje que aparece en la pantalla*/
 char str[80];
 int row,col;
 initscr();
```

```
 getmaxyx(stdscr,row,col);
 mvprintw(row/2,(col-strlen(mesg))/2,"%s",mesg);
 getstr(str); //Recibe el string
 mvprintw(LINES - 2, 0, "You Entered: %s", str);
 getch(); //Recibe un caracter
 endwin();

 return 0;
}
```

Menu library

Empezamos con la parte divertida de ncurses: La seccion de Menus.

Un menu muestra en la pantalla una serie de opciones disponibles para elegir, permitiendo al usuario mostrarle de manera ordenada y atractiva aquello que pueda elegir.

La estructura de un menu es sencilla, es la coleccion de items (ya sea uno o n) los cuales estan disponibles para seleccion en el programa.

El flujo de un menu es el siguiente:

- Inicializamos curses
- Creamos items utilizando: new_item(). Puedes especificar el nombre y descripcion del item.
- Creamos el menu con: new_menu(). Es importante especificar los items adjuntos al menu
- Subimos el menu con: menu_post() y refrescamos la pantalla.
- Se procesan las selecciones del usuario con un bucle y se realizan las necesarias actualizaciones constantes con: menu_driver.
- Sacamos el menu con: menu_unpost().
- Liberamos la memoria dedidaca con: free_menu().
- Liberamos la memoria dedicada de cada item con: free_item().
- Finalizamos curses.

Aparte de utilizar la biblioteca de ncurses.h, tambien es necesario utilizar menu.h (misma que viene incluida en la familia curses).

Para compilar se hace de esta manera: gcc <programa.c> -lmenu -lncurses

Codigo de ejemplo

```
#include <curses.h> //Biblioteca principal de ncurses para manejo de la pantalla
#include <menu.h> //Biblioteca de ncurses para menús
#define ARRAY_SIZE(a) (sizeof(a) / sizeof(a[0])) // Macro para calcular el número de elementos de un arreglo
#define CTRLD    4 // Macro que representa Ctrl+D (ASCII 4), útil para detectar Ends of File

char *choices[] = {            // Arreglo de punteros a cadenas que representan las opciones del menú
                        "Choice 1",
                        "Choice 2",
                        "Choice 3",
                        "Choice 4",
                        "Exit",
                  };

int main(){
    ITEM **my_items; // Puntero a arreglo de punteros de tipo ITEM, para almacenar cada opción del menú
    int c; // Variable para capturar la tecla presionada por el usuario
    MENU *my_menu; // Puntero al menú
    int n_choices, i; // n_choices = número de opciones, i = índice para bucles
    ITEM *cur_item; // Puntero al item actualmente seleccionado (no se usa en este código)

    // ---------------- Inicialización de ncurses ----------------
    initscr(); // Inicializa ncurses y prepara la pantalla
    cbreak(); // Activa el modo cbreak: los caracteres se leen inmediatamente sin esperar Enter
    noecho(); // No muestra en pantalla lo que el usuario escribe
    keypad(stdscr, TRUE); // Permite leer teclas especiales (flechas, F1, etc.)
```

```
    // ---------------- Crear items ----------------
    n_choices = ARRAY_SIZE(choices); // Calcula el número de opciones
    my_items = (ITEM **)calloc(n_choices + 1, sizeof(ITEM *));
// Reserva memoria dinámica para los punteros de los items (+1 para NULL)
    for(i = 0; i < n_choices; ++i){
                my_items[i] = new_item(choices[i], choices[i]); // Crea cada ITEM con nombre y descripción
        }
    my_items[n_choices] = (ITEM *)NULL; // El último puntero debe ser NULL según la documentación de ncurses


    // ---------------- Crear y mostrar menú ----------------
    my_menu = new_menu((ITEM **)my_items); // Crea el menú asociando los items
    mvprintw(LINES - 2, 0, "F1 to Exit"); // Muestra un mensaje de ayuda en la penúltima fila
    post_menu(my_menu); // Publica el menú en la pantalla
    refresh(); // Actualiza la pantalla


    // ---------------- Bucle de interacción ----------------
    while((c = getch()) != KEY_F(1)){ // Espera hasta que el usuario presione F1
        switch(c){
            case KEY_DOWN: // Si se presiona la flecha hacia abajo
                menu_driver(my_menu, REQ_DOWN_ITEM); // Mueve el cursor del menú hacia abajo
                break;
            case KEY_UP: // Si se presiona la flecha hacia arriba
                menu_driver(my_menu, REQ_UP_ITEM); // Mueve el cursor del menú hacia arriba
                break;
        }
    }
```

```
    // ---------------- Liberación de memoria ----------------
    free_item(my_items[0]);   // Libera la memoria del primer item
    free_item(my_items[1]);   // Libera la memoria del segundo item
    free_menu(my_menu);   // Libera la memoria del menú
    endwin(); // Finaliza ncurses y devuelve la terminal a su estado normal
}
```

Este ejemplo muestra los conceptos basicos relacionados en la creacion de un menu utilizando la biblioteca de menus, propia de la familia curses.

Primero creamos los items usando new_item().

Luego los adjuntamos al menu con la funcion new_menu().

El codigo lee las entradas del usuario correspondientes a cada accion.

Se “refresca” la pantalla del menu de acuerdo al movimiento del usuario.

Por ultimo se libera la memoria dedicada a cada item dentro del menu.

La funcion menu_driver, es una pieza importante en dentro de la funcion menus, porque dictamina **que** se hara dentro del codigo. Su estructura es asi: menu_driver(“Parametro 1”, “Parametro 2”), el primer parametro indica en donde se trabajara mientras que el segundo indica la instruccion que debera hacer el codigo. El valor del parametro dos puede ser:

- A menu navigational request.
- Un caracter en ascii.
- KEY_MOUSE special key (asociada con eventos del raton).

Por defecto solo lee “menu navigational request”, sin embargo, para que lea caracteres en ascii y/o KEY_MOUSE, se requiere habilitar el soporte correspondiente a cada una.

Las instrucciones que son permitidas en “Menu navigational request” son:

- REQ_LEFT_ITEM = Se mueve al item de la izquierda.
- REQ_RIGHT_ITEM= Se mueve al item de la derecha.
- REQ_UP_ITEM= Se mueve al item de arriba.
- REQ_DOWN_ITEM= Se mueve al item de abajo.
- REQ_SCR_ULINE = El scroll del raton se mueve a la linea de arriba.
- REQ_SCR_DLINE = El scroll del raton se mueve a la linea de abajo.
- REQ_SCR_DPAGE = El scroll del raton se regresa una pagina.
- REQ_SCR_UPAGE = El scroll del raton avanza una pagina.

- REQ_FIRST_ITEM = Se desplaza al primer item.
- REQ_LAST_ITEM = Se desplaza al ultimo item.
- REQ_NEXT_ITEM = Se desplaza al proximo item.
- REQ_PREV_ITEM = Se desplaza al item anterior.
- REQ_TOGGLE_ITEM = Se selecciona o deselecciona un item..
- REQ_CLEAR_PATTERN = Limpia el buffer de patrones del menu.
- REQ_BACK_PATTERN = Elimina el caracter anterior al buffer de patrones..
- REQ_NEXT_MATCH = Pasa al siguiente elemento que coincida con el buffer de patrones.
- REQ_PREV_MATCH = Regresa al anterior elemento que coincida con el buffer de patrones.