

Ascending Order

Source Code Preview

This project was a collaboration between a fellow developer and I that I met online though GAMEDEV.NET. Being relatively new, we decided to do something simple. Here I will provide information from my responsibilities for this project.

Source Files

- main.cpp
- Game.cpp
- Game.h

Responsibilities:

- Programming Flow and Structure
- Audio Programming
- User Interface Design and Programming
- Graphic Art Development
- Documentation

Source File: main.cpp

```
#define USE_CONSOLE
#include <allegro.h>
#include "Game.h"

int main(int argc, char *argv[])
{
    Game* game = new Game();           // Create New Game Object

    game->m_done = false;              // Set m_done variable which controls game loop
    game->initGame();                 // Initialize Graphics and Game Components

    while(game->m_done != true)       // While Not Done
    {
        game->gameUpdate();           // Update Game Logic
    }

    game = NULL;                      // Clear Game Object

    return 0;
}
END_OF_MAIN();
```

Source File: Game.h

```
#ifndef H_GAME
#define H_GAME

class Game {
public:
    // Constructor
    Game();
    // Destructor
    ~Game();

    /*****
    * Function: initGame(void)
    * Purpose: Initializes Allegro. Game Components
    *
    * @return - True, success; False, fail
    *****/
    bool initGame();

    /*****
    * Function: showMessage(int)
    * Purpose: pass 1 to show 'Welcome', pass 2 to show 'Congrat..'
    *****/
    void showMessage(int number);

    /*****
    * Function: gameUpdate(void)
    * Purpose: Game Loop
    *****/
    void gameUpdate();

    /*****
    * Function: loadImages(void)
    * Purpose: Loading images into pointers
    *****/
    void loadImages();

    /*****
    * Function: loadSounds(void)
    * Purpose: Loading sounds into pointers
    *****/
    void loadSounds();

    /*****
    * Function: setupMouse(void)
    * Purpose: Sets up the mouse
    *****/
    void setupMouse();

    /*****
    * Function: detectWin(void);
    * Purpose: Detects when a player wins
    *****/
    void detectWin();
};
```

```

/*****
* Function: detectKey(void)
* Purpose: Detects key presses and executes
*         an action
*****/
void detectKey();

/*****
* Function: detectSelection(void);
* Purpose: Detects when the mouse hovers over
*         a tile
*****/
void detectSelection(int mx, int my);

/*****
* Function: handleMouse(void)
* Purpose: Handles mouse clicks
*****/
void handleMouse(int mx, int my);

/*****
* Function: randomizeTiles(void)
* Purpose: Randomizes all of the tiles
*****/
void randomizeTiles();

/*****
* Function: updateTiles(void)
* Purpose: Redraws all the tiles to their
*         current positions
*****/
void updateTiles();

/*****
* Function: updateScreen();
* Purpose: Updates the screen
*****/
void updateScreen();

/*****
* Function: msgCreditwin(int mx, int my)
* Purpose: To update and keep track of what is
*         going on in the msg window
*****/
void updateMsgWin(int mx, int my);

/*****
* Function: updateCreditwin(int mx, int my)
* Purpose: To update and keep track of what is
*         going on in the credit window
*****/
void updateCreditWin(int mx, int my);

/*****
* Function: updateSoundWin();
* Purpose: Updates the sound window
*****/
void updateSoundWin(int mx, int my);

```

```
    // Class Members
    int mx, my;
    int tiles[9];
    bool m_done;
    bool win;
};

#endif
```

Source File: Game.cpp

```
bool Game::initGame()
{
    // Prepare Allegro
    if(install_allegro(GFX_AUTODETECT, &errno, atexit) != 0)
    {
        allegro_message("Error initializing Allegro Graphics | install_allegro(...)\n\n");
        return false;
    }

    // Graphics
    set_color_depth(32);

    if(set_gfx_mode(GFX_AUTODETECT_WINDOWED, 600, 400, 0, 0) != 0)
    {
        allegro_message("Error initializing Allegro gfx_mode | set_gfx_mode(...)\n\n");
        return false;
    }

    // Load Files for Use
    loadImages();

    set_window_title("Ascending Order v.001");

    clear(screen);

    blit(background, screen, 0, 0, 0, 0, 600, 400);

    // Componets
    if(install_keyboard() != 0)
    {
        allegro_message("Error initializing Allegro Keyboard | install_keyboard(...)\n\n");
        return false;
    }

    setupMouse();

    // Sound
    if(install_sound(DIGI_AUTODETECT, MIDI_AUTODETECT, "") != 0)
    {
        allegro_message("Error initializing Allegro sounds | install_sound(...)\n\n");
        return false;
    }

    // Load Sound files for Use
    loadSounds();

    // Seed
    srand(time(NULL));

    // Randomize Initial Tile Layout
    randomizeTiles();

    // Update Display
    updateTiles();

    // Play Theme
    play_midi(themel, 1);

    // Show Welcome Message
    showMessage(1);

    return true; // success!
}
```

```

void Game::handleMouse(int mx, int my)
{
    // If any other windows are not up, check for mouse collision with buttons
    if(windowOpen == false)
    {
        // Menu Buttons
        if(mx >= 375 && mx <= 573)
        {
            if((soundWinShown == false) && (creditWinShown == false))
            {
                // New Game Button
                if(my >= 88 && my <= 125)
                {
                    blit(msgWindow, screen, 0, 0, 30, 80, 288, 187);
                    play_sample(move, svolume, 128, 800, FALSE);
                    msgWinShown = true;
                    windowOpen = true;
                }
            }
            if((creditWinShown == false) && (msgWinShown == false))
            {
                // Sound Button
                if(my >= 138 && my <= 170)
                {
                    blit(soundWindow, screen, 0, 0, 30, 80, 288, 187);
                    play_sample(move, svolume, 128, 800, FALSE);
                    soundWinShown = true;
                    windowOpen = true;
                }
            }
            if((soundWinShown == false) && (msgWinShown == false))
            {
                // Credits Button
                if(my >= 290 && my <= 324)
                {
                    blit(creditWindow, screen, 0, 0, 30, 80, 288, 187);
                    play_sample(move, svolume, 128, 800, FALSE);
                    creditWinShown = true;
                    windowOpen = true;
                }
            }
        }
    }

    // Exit Button
    if(my >= 340 && my <= 377)
    {
        m_done = true;
    }

    if(msgWinShown == true)
    {
        scare_mouse();
        updateMsgWin(mx, my); // Update Button Click for Message Window
        show_mouse(screen);
    }

    if(creditWinShown == true)
    {
        scare_mouse();
        updateCreditWin(mx, my); // Update Button Click for Credit Window
        show_mouse(screen);
    }
}

```

```
if(soundWinShown == true)
{
    scare_mouse();
    updateSoundWin(mx, my);           // Update Button Click for Sound Window
    show_mouse(screen);
}

rest(100);
}
```

Graphic Design

