# Detailed Design

**Team Code Slayer**

**Version 1.1 December 3, 2016**

# Table of Contents

Purpose……………………………………………………………………………………………..1

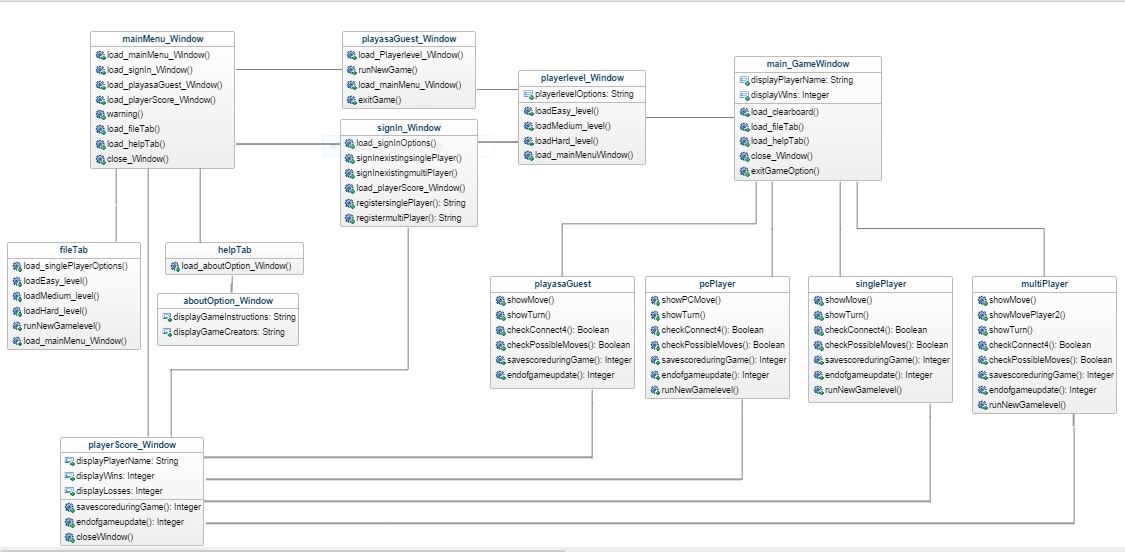
Source Code Descriptions………………………………………………………………….……………………..3 - 29

Awknowledgements……………………………………………….………………………………..30

# Purpose

The purpose of this project is to create a Tic-Tac-Toe with a game board of 5 X 5 where the winner is determined by the first person who connect 4 stone in a line, being horizontally, vertically, or diagonally. The game should be able to play with two players, one which can be the computer. History of User should be able to display at user request. User is also allowed to choose the difficulty when playing AI.

# Class Diagram



# Source Code Detail

***//Global variables***

**const int CELL\_SIZE = 100;**

**HBRUSH hbr1, hbr2, hbr3, hbr4;**

**int board[4] = { 0,1,2,3 };**

**int playerTurn = 0;**

**int mode = 6;**

**int gameBoard[25] = { 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 };**

**int winner = 0;**

**int wins[4];**

**int player1 = 0;**

**int player2 = 0;**

**int guest = 0;**

**const WCHAR \* select[4];**

**WCHAR \* szPlayer1[20];**

**WCHAR \* szPlayer2[20];**

**const WCHAR szPlayer3[20] = L"AI";**

**int index;**

**HWND TextBox, SendButton;**

**WCHAR szInput1[MAX\_PATH], szInput2[MAX\_PATH];**

**int page = 0;**

**ofstream file;**

**int cells[] = { 0,1,2,3, 1,2,3,4, 5,6,7,8, 6,7,8,9, 10,11,12,13, 11,12,13,14, 15,16,17,18, 16,17,18,19, 20,21,22,23, 21,22,23,24,**

**0,5,10,15, 5,10,15,20, 1,6,11,16, 6,11,16,21, 2,7,12,17, 7,12,17,22, 3,8,13,18, 8,13,18,23, 4,9,14,19, 9,14,19,24,**

***//Pause the system making it seem like is thinking***

**void time(HWND hWnd, HDC hdc)**

**{**

**clock\_t endwait;**

**endwait = clock() + 1 \* CLOCKS\_PER\_SEC;**

**while (clock() < endwait) {}**

**}**

***//selecting the correct name to display on the screen***

**void PlayersName(HWND hWnd, HDC hdc)**

**{**

**if (guest == 1)**

**{**

**int ret = MessageBox(hWnd, L"Will player one be playing the AI?", L"Whos playing AI?", MB\_YESNO | MB\_ICONQUESTION);**

**if (IDNO == ret)**

**{**

**szPlayer1[player1] = szInput2;**

**szPlayer2[player2] = szInput1;**

**guest = 2;**

**}**

**if (IDYES == ret)**

**{**

**szPlayer1[player1] = szInput1;**

**szPlayer2[player2] = szInput2;**

**guest = 0;**

**}**

**}**

**else**

**{**

**if (guest == 2)**

**{**

**szPlayer1[player1] = szInput2;**

**szPlayer2[player2] = szInput1;**

**}**

**else**

**{**

**szPlayer1[player1] = szInput1;**

**szPlayer2[player2] = szInput2;**

**}**

**}**

**}**

***//creating the correct screen depending on user selection***

**BOOL GetGameBoard(HWND hWnd, RECT \* pRect)**

**{**

**RECT rc;**

**if (GetClientRect(hWnd, &rc))**

**{**

**int width = rc.right - rc.left;**

**int height = rc.bottom - rc.top;**

**if (mode >= 6)**

**{**

**pRect->left = (width - CELL\_SIZE \* 3) / 2;**

**pRect->top = (height - CELL\_SIZE \* 3) / 2;**

**pRect->right = pRect->left + CELL\_SIZE \* 3;**

**pRect->bottom = pRect->top + CELL\_SIZE \* 3;**

**}**

**else if (mode >= 4)**

**{**

**pRect->left = (width - CELL\_SIZE \* 4) / 2;**

**pRect->top = (height - CELL\_SIZE \* 4) / 2;**

**pRect->right = pRect->left + CELL\_SIZE \* 4;**

**pRect->bottom = pRect->top + CELL\_SIZE \* 4;**

**}**

**else**

**{**

**pRect->left = (width - CELL\_SIZE \* 5) / 2;**

**pRect->top = (height - CELL\_SIZE \* 5) / 2;**

**pRect->right = (pRect->left + CELL\_SIZE \* 5);**

**pRect->bottom = (pRect->top + CELL\_SIZE \* 5);**

**}**

**return TRUE;**

**}**

**SetRectEmpty(pRect);**

**return FALSE;**

**}**

***// Drawing the lines of the tic-tac-toe game ( Horizontally and Vertically)***

**void Drawlines(HDC hdc, int x1, int y1, int x2, int y2)**

**{**

**MoveToEx(hdc, x1, y1, NULL);**

**LineTo(hdc, x2, y2);**

**}**

***// each box has and index value to be able to call upon***

**int GetCellNumberFromPoint(HWND hwnd, int x, int y)**

**{**

**POINT pt = { x, y };**

**RECT rc;**

**if (GetGameBoard(hwnd, &rc))**

**{**

**if (PtInRect(&rc, pt))**

**{**

**//use clicks in the board**

**x = pt.x - rc.left;**

**y = pt.y - rc.top;**

**int column = x / CELL\_SIZE;**

**int row = y / CELL\_SIZE;**

**if (mode >= 4)**

**{**

**int row = (y / CELL\_SIZE);**

**return (row);**

**}**

**return (column + row \* 5);**

**}**

**}**

**//outside tic-tac-toe board**

**return -1;**

**}**

***// Making sure that anything inside the game board is and index of 0-24 while out side is -1***

**BOOL GetCellRect(HWND hWnd, int index, RECT \*pRect)**

**{**

**RECT rcBoard;**

**SetRectEmpty(pRect);**

**if (index < 0 || index > 24)**

**return FALSE;**

**if (GetGameBoard(hWnd, &rcBoard))**

**{**

**int y, x;**

**if (mode == 6)**

**{**

**y = index;**

**x = index / 3;**

**}**

**else if (mode >= 4)**

**{**

**y = index;**

**x = index / 4;**

**}**

**else**

**{**

**y = index / 5;**

**x = index % 5;**

**}**

**if (mode == 6)**

**{**

**pRect->left = rcBoard.left + x \* CELL\_SIZE + 10;**

**pRect->top = rcBoard.top + y \* CELL\_SIZE + 10;**

**pRect->right = pRect->left + CELL\_SIZE + 180;**

**pRect->bottom = pRect->top + CELL\_SIZE - 20;**

**}**

**else if (mode >= 4)**

**{**

**pRect->left = rcBoard.left + x \* CELL\_SIZE + 10;**

**pRect->top = rcBoard.top + y \* CELL\_SIZE + 10;**

**pRect->right = pRect->left + CELL\_SIZE + 280;**

**pRect->bottom = pRect->top + CELL\_SIZE - 20;**

**}**

**else**

**{**

**pRect->left = rcBoard.left + x \* CELL\_SIZE + 10;**

**pRect->top = rcBoard.top + y \* CELL\_SIZE + 10;**

**pRect->right = pRect->left + CELL\_SIZE - 20;**

**pRect->bottom = pRect->top + CELL\_SIZE - 20;**

**}**

**return TRUE;**

**}**

**return FALSE;**

**}**

**//returns the winner**

**int Winner(int wins[4])**

**{**

***//row, verticals, columns***

**for (int i = 0; i < ARRAYSIZE(cells); i += 4)**

**{**

**if (0 != gameBoard[cells[i]] && gameBoard[cells[i]] == gameBoard[cells[i + 1]] && gameBoard[cells[i]] == gameBoard[cells[i + 2]] && gameBoard[cells[i]] == gameBoard[cells[i + 3]])**

**{**

**wins[0] = cells[i];**

**wins[1] = cells[i + 1];**

**wins[2] = cells[i + 2];**

**return gameBoard[cells[i]];**

**}**

**}**

**for (int i = 0; i < ARRAYSIZE(gameBoard); ++i)**

**{**

**if (0 == gameBoard[i])**

**return 0;**

**}**

**return 3;**

**}**

***//Creating a text for when is user turn***

**void ShowTurn(HWND hWnd, HDC hdc)**

**{**

**const WCHAR \* TurnText = NULL;**

**switch (winner)**

**{**

**case 0:**

**if (mode == 0)**

**{**

**TurnText = (playerTurn == 1) ? szPlayer1[player1] : szPlayer2[player2];**

**}**

**if (mode != 0)**

**{**

**TurnText = (playerTurn == 1) ? szPlayer1[player1] : szPlayer3;**

**}**

**break;**

**case 1:**

**TurnText = L"Player 1 is the winnner!";**

**break;**

**case 2:**

**TurnText = L"Player 2 is the winnner!";**

**break;**

**case 3:**

**TurnText = L"It's a draw!";**

**break;**

**case 4:**

**TurnText = L"AI is the winnner!";**

**break;**

**}**

**RECT rc;**

**if (NULL != TurnText && GetClientRect(hWnd, &rc))**

**{**

**rc.top = rc.bottom - 48;**

**FillRect(hdc, &rc, (HBRUSH)GetStockObject(GRAY\_BRUSH));**

**SetTextColor(hdc, RGB(255, 255, 255));**

**SetBkMode(hdc, TRANSPARENT);**

**DrawText(hdc, TurnText, lstrlen(TurnText), &rc, DT\_CENTER);**

**}**

**}**

***//keeping track of user history***

**void Score(HWND hWnd, HDC hdc)**

**{**

**file.open("Score.txt", fstream::app);**

**if (file.is\_open())**

**{**

**if (playerTurn == 1 && winner == 1 && mode == 0)**

**{**

**file.write((char\*)szInput1, 20);**

**file << " Wins \n";**

**file.write((char\*)szInput2, 20);**

**file << " Loses \n";**

**file.close();**

**}**

**else if (playerTurn == 2 && winner == 2 && mode == 0)**

**{**

**file.write((char\*)szInput2, 20);**

**file << " Wins \n";**

**file.write((char\*)szInput1, 20);**

**file << " Loses \n";**

**file.close();**

**}**

**else if (playerTurn == 1 && winner == 1 && mode >= 1)**

**{**

**file.write((char\*)szInput1, 20);**

**file << " Wins \n";**

**file.write((char\*)szPlayer3, 20);**

**file << " Loses \n";**

**file.close();**

**}**

**else if (playerTurn == 4 && winner == 4 && mode >= 1)**

**{**

**file.write((char\*)szPlayer3, 20);**

**file << " Wins \n";**

**file.write((char\*)szInput1, 20);**

**file << " Loses \n";**

**file.close();**

**}**

**else if ( winner == 3)**

**{**

**file.write((char\*)szInput2, 20);**

**file << " Tie \n";**

**file.write((char\*)szInput1, 20);**

**file << " Tie \n";**

**file.close();**

**}**

**}**

**}**

***//Player VS Player mode***

**void Moves(HWND hWnd, HDC hdc, int xPos, int yPos)**

**{**

**if (mode == 0)**

**{**

***//calling function for button***

**index = GetCellNumberFromPoint(hWnd, xPos, yPos);**

***//print out Value of cell clicked on***

**if (NULL != hdc)**

**{**

**/\*WCHAR temp[100];**

**wsprintf(temp, L"Index = %d", index);**

**TextOut(hdc, xPos, yPos, temp, lstrlen(temp));\*/**

**//get cell demensions**

**if (index != -1)**

**{**

**RECT rcCell;**

***//cell is taken no other player can take over***

**if ((0 == gameBoard[index]) && GetCellRect(hWnd, index, &rcCell))**

**{**

**gameBoard[index] = playerTurn;**

**FillRect(hdc, &rcCell, (playerTurn == 2) ? hbr2 : hbr1);**

**//FillRect(hdc, &rcCell, hbr1);**

**winner = Winner(wins);**

**if (winner == 1 || winner == 2)**

**{**

**MessageBox(hWnd, (winner == 1) ? L"Player 1 is the winnner!" : L"Player 2 is the winner!", L"You Win!", MB\_OK | MB\_ICONINFORMATION);**

**Score(hWnd, hdc);**

**playerTurn = 0;**

**}**

**else if (3 == winner)**

**{**

**MessageBox(hWnd, L"No one wins!", L"It's a draw", MB\_OK | MB\_ICONEXCLAMATION);**

**Score(hWnd, hdc);**

**playerTurn = 0;**

**}**

**else if (0 == winner)**

**{**

**playerTurn = (playerTurn == 1) ? 2 : 1;**

**}**

**ShowTurn(hWnd, hdc);**

**}**

**}**

**ReleaseDC(hWnd, hdc);**

**}**

**}**

**}**

***//Player moves when playing AI***

**void Move(HWND hWnd, HDC hdc, int xPos, int yPos)**

**{**

**if (mode >= 1 && playerTurn == 1)**

**{**

***//calling function for button***

**index = GetCellNumberFromPoint(hWnd, xPos, yPos);**

***//print out Value of cell clicked on***

**if (NULL != hdc)**

**{**

**/\*WCHAR temp[100];**

**wsprintf(temp, L"Index = %d", index);**

**TextOut(hdc, xPos, yPos, temp, lstrlen(temp));\*/**

***//get cell demensions***

**if (index != -1)**

**{**

**RECT rcCell;**

***//cell is taken no other player can take over***

**if ((0 == gameBoard[index]) && GetCellRect(hWnd, index, &rcCell))**

**{**

**gameBoard[index] = playerTurn;**

**FillRect(hdc, &rcCell, hbr1);**

**winner = Winner(wins);**

**if (winner == 1)**

**{**

**MessageBox(hWnd, L"Player 1 is the winnner!", L"You Win!", MB\_OK | MB\_ICONINFORMATION);**

**Score(hWnd, hdc);**

**playerTurn = 0;**

**}**

**else if (3 == winner)**

**{**

**MessageBox(hWnd, L"No one wins!", L"It's a draw", MB\_OK | MB\_ICONEXCLAMATION);**

**Score(hWnd, hdc);**

**playerTurn = 0;**

**}**

**else if (0 == winner)**

**{**

**playerTurn = (playerTurn == 1) ? 4 : 1;**

**}**

**ShowTurn(hWnd, hdc);**

**}**

**}**

**ReleaseDC(hWnd, hdc);**

**}**

**}**

**}**

**// AI rules to be able to play agains the computer.**

**int modes(int &index)**

**{**

**//stoped here!**

**int winner = 25;**

**int loser1 = 25;**

**int loser2 = 25;**

**int opt1 = 25;**

**int opt2 = 25;**

**for (int i = 0; i < ARRAYSIZE(cells); i += 4)**

**{**

**if (mode > 0)**

**{**

**if (mode > 1)**

**{**

**if (mode > 2)**

**{**

***//block hard***

**if (1 == gameBoard[cells[i]] && 1 == gameBoard[cells[i + 1]] && 0 == gameBoard[cells[i + 2]] && 0 == gameBoard[cells[i + 3]])**

**loser2 = cells[i + 2];**

**else if (1 == gameBoard[cells[i]] && 1 == gameBoard[cells[i + 2]] && 0 == gameBoard[cells[i + 1]] && 0 == gameBoard[cells[i + 3]])**

**loser2 = cells[i + 1];**

**else if (1 == gameBoard[cells[i]] && 1 == gameBoard[cells[i + 3]] && 0 == gameBoard[cells[i + 2]] && 0 == gameBoard[cells[i + 1]])**

**loser2 = cells[i + 2];**

**else if (1 == gameBoard[cells[i + 1]] && 1 == gameBoard[cells[i + 2]] && 0 == gameBoard[cells[i]] && 0 == gameBoard[cells[i + 3]])**

**loser2 = cells[i];**

**else if (1 == gameBoard[cells[i + 1]] && 1 == gameBoard[cells[i + 3]] && 0 == gameBoard[cells[i]] && 0 == gameBoard[cells[i + 2]])**

**loser2 = cells[i];**

**else if (1 == gameBoard[cells[i + 2]] && 1 == gameBoard[cells[i + 3]] && 0 == gameBoard[cells[i]] && 0 == gameBoard[cells[i + 1]])**

**loser2 = cells[i];**

**}**

***//block medium***

**if (1 == gameBoard[cells[i]] && 1 == gameBoard[cells[i + 2]] && 1 == gameBoard[cells[i + 3]] && 0 == gameBoard[cells[i + 1]])**

**loser1 = cells[i + 1];**

**else if (1 == gameBoard[cells[i]] && 1 == gameBoard[cells[i + 1]] && 1 == gameBoard[cells[i + 3]] && 0 == gameBoard[cells[i + 2]])**

**loser1 = cells[i + 2];**

**else if (1 == gameBoard[cells[i]] && 1 == gameBoard[cells[i + 1]] && 1 == gameBoard[cells[i + 2]] && 0 == gameBoard[cells[i + 3]])**

**loser1 = cells[i + 3];**

**else if (1 == gameBoard[cells[i + 1]] && 1 == gameBoard[cells[i + 2]] && 1 == gameBoard[cells[i + 3]] && 0 == gameBoard[cells[i]])**

**loser1 = cells[i];**

***//medium options***

**else if (4 == gameBoard[cells[i]] && 0 == gameBoard[cells[i + 2]] && 0 == gameBoard[cells[i + 3]] && 0 == gameBoard[cells[i + 1]])**

**opt2 = cells[i + 1];**

**else if (0 == gameBoard[cells[i]] && 0 == gameBoard[cells[i + 2]] && 0 == gameBoard[cells[i + 3]] && 4 == gameBoard[cells[i + 1]])**

**opt2 = cells[i];**

**else if (0 == gameBoard[cells[i]] && 4 == gameBoard[cells[i + 2]] && 0 == gameBoard[cells[i + 3]] && 0 == gameBoard[cells[i + 1]])**

**opt2 = cells[i];**

**else if (0 == gameBoard[cells[i]] && 0 == gameBoard[cells[i + 2]] && 4 == gameBoard[cells[i + 3]] && 0 == gameBoard[cells[i + 1]])**

**opt2 = cells[i];**

***//options to process to win***

**else if (4 == gameBoard[cells[i]] && 4 == gameBoard[cells[i + 1]] && 0 == gameBoard[cells[i + 2]] && 1 != gameBoard[cells[i + 3]])**

**opt1 = cells[i + 2];**

**else if (4 == gameBoard[cells[i]] && 4 == gameBoard[cells[i + 2]] && 0 == gameBoard[cells[i + 1]] && 1 != gameBoard[cells[i + 3]])**

**opt1 = cells[i + 1];**

**else if (4 == gameBoard[cells[i]] && 4 == gameBoard[cells[i + 3]] && 0 == gameBoard[cells[i + 2]] && 1 != gameBoard[cells[i + 1]])**

**opt1 = cells[i + 2];**

**else if (4 == gameBoard[cells[i + 1]] && 4 == gameBoard[cells[i + 2]] && 0 == gameBoard[cells[i]] && 1 != gameBoard[cells[i + 3]])**

**opt1 = cells[i];**

**else if (4 == gameBoard[cells[i + 1]] && 4 == gameBoard[cells[i + 3]] && 0 == gameBoard[cells[i]] && 1 != gameBoard[cells[i + 2]])**

**opt1 = cells[i];**

**else if (4 == gameBoard[cells[i + 2]] && 4 == gameBoard[cells[i + 3]] && 0 == gameBoard[cells[i]] && 1 != gameBoard[cells[i + 1]])**

**opt1 = cells[i];**

**}**

***//win***

**if (4 == gameBoard[cells[i]] && 4 == gameBoard[cells[i + 2]] && 4 == gameBoard[cells[i + 3]] && 0 == gameBoard[cells[i + 1]])**

**winner = cells[i + 1];**

**else if (4 == gameBoard[cells[i]] && 4 == gameBoard[cells[i + 1]] && 4 == gameBoard[cells[i + 3]] && 0 == gameBoard[cells[i + 2]])**

**winner = cells[i + 2];**

**else if (4 == gameBoard[cells[i]] && 4 == gameBoard[cells[i + 1]] && 4 == gameBoard[cells[i + 2]] && 0 == gameBoard[cells[i + 3]])**

**winner = cells[i + 3];**

**else if (4 == gameBoard[cells[i + 1]] && 4 == gameBoard[cells[i + 2]] && 4 == gameBoard[cells[i + 3]] && 0 == gameBoard[cells[i]])**

**winner = cells[i];**

***// easy options***

**else**

**{**

**opt2 = rand() % 25;**

**while (1 == gameBoard[opt2] || 4 == gameBoard[opt2])**

**opt2 = rand() % 25;**

**}**

**}**

**}**

**if (winner != 25)**

**index = winner;**

**else if (loser1 != 25)**

**index = loser1;**

**else if (loser2 != 25)**

**index = loser2;**

**else if (opt1 != 25)**

**index = opt1;**

**else if (opt2 != 25)**

**index = opt2;**

**return index;**

**}**

***//AI moves***

**void PCMoves(HWND hWnd, HDC hdc)**

**{**

**if (4 == playerTurn)**

**{**

***//calling function for button***

***//print out Value of cell clicked on***

**HDC hdc = GetDC(hWnd);**

**if (NULL != hdc)**

**{**

**/\*WCHAR temp[100];**

**wsprintf(temp, L"Index = %d", index);**

**TextOut(hdc, xPos, yPos, temp, lstrlen(temp));\*/**

***//get cell demensions***

**RECT rcCell;**

***//AI***

**modes(index);**

***//cell is taken no other player can take over***

**if ((0 == gameBoard[index]) && GetCellRect(hWnd, index, &rcCell))**

**{**

**gameBoard[index] = playerTurn;**

**FillRect(hdc, &rcCell, hbr3);**

**winner = Winner(wins);**

**if (winner == 4)**

**{**

**MessageBox(hWnd, L"AI is the winner!", L"You Lose!", MB\_OK | MB\_ICONINFORMATION);**

**Score(hWnd, hdc);**

**playerTurn = 0;**

**}**

**else if (3 == winner)**

**{**

**MessageBox(hWnd, L"No one wins!", L"It's a draw", MB\_OK | MB\_ICONEXCLAMATION);**

**Score(hWnd, hdc);**

**playerTurn = 0;**

**}**

**else if (0 == winner)**

**{**

**playerTurn = (playerTurn == 1) ? 4 : 1;**

**}**

**ShowTurn(hWnd, hdc);**

**}**

**}**

**ReleaseDC(hWnd, hdc);**

**}**

**}**

***// How many lines to create when display the options to start playing***

**void Horizon(HDC hdc, int x1, int y1, int x2, int y2)**

**{**

**MoveToEx(hdc, x1, y1, NULL);**

**LineTo(hdc, x2, y2);**

**}**

***// rest depending on user options. To play a new game or go back to the main menu***

**void reset(HWND hWnd, HDC hdc)**

**{**

**if (mode >= 4)**

**playerTurn = 0;**

**else**

**{**

**int ans = MessageBox(hWnd, L"Would you like to make the first move?", L"Player", MB\_YESNO | MB\_ICONQUESTION);**

**if (IDNO == ans)**

**{**

**if (mode == 0)**

**playerTurn = 2;**

**else**

**playerTurn = 4;**

**}**

**else**

**playerTurn = 1;**

**}**

**winner = 0;**

**ZeroMemory(gameBoard, sizeof(gameBoard));**

**InvalidateRect(hWnd, NULL, TRUE);**

**UpdateWindow(hWnd);**

**if (mode > 0)**

**{**

**PCMoves(hWnd, hdc);**

**}**

**PlayersName(hWnd, hdc);**

**}**

***// Giving the user an option to sign in or log in as a Guest***

**void Register(HWND hWnd, HDC hdc)**

**{**

**if (mode >= 5)**

**{**

**if (page == 2)**

**{**

**TextBox = CreateWindowEx(WS\_EX\_CLIENTEDGE, L"edit", L"Guest1",**

**WS\_CHILD | WS\_VISIBLE | WS\_TABSTOP | WS\_BORDER | ES\_LEFT,**

**CW\_USEDEFAULT, CW\_USEDEFAULT, 200, 24, // x, y, w, h**

**hWnd, (HMENU)(101),**

**(HINSTANCE)GetWindowLong(hWnd, GWL\_HINSTANCE), NULL);**

**GetWindowText(GetDlgItem(hWnd, 101), szInput1, MAX\_PATH);**

**DestroyWindow(TextBox);**

**page = 0;**

**}**

**else if (page == 1)**

**{**

**TextBox = CreateWindowEx(WS\_EX\_CLIENTEDGE, L"edit", L"Guest2",**

**WS\_CHILD | WS\_VISIBLE | WS\_TABSTOP | WS\_BORDER | ES\_LEFT,**

**CW\_USEDEFAULT, CW\_USEDEFAULT, 200, 24, // x, y, w, h**

**hWnd, (HMENU)(101),**

**(HINSTANCE)GetWindowLong(hWnd, GWL\_HINSTANCE), NULL);**

**GetWindowText(GetDlgItem(hWnd, 101), szInput2, MAX\_PATH);**

**DestroyWindow(TextBox);**

**page = 0;**

**}**

**else**

**{**

**TextBox = CreateWindowEx(WS\_EX\_CLIENTEDGE, L"edit", L"",**

**WS\_CHILD | WS\_VISIBLE | WS\_TABSTOP | WS\_BORDER | ES\_LEFT,**

**CW\_USEDEFAULT, CW\_USEDEFAULT, 200, 24, // x, y, w, h**

**hWnd, (HMENU)(101),**

**(HINSTANCE)GetWindowLong(hWnd, GWL\_HINSTANCE), NULL);**

**if (mode == 6)**

**{**

**SendButton = CreateWindowEx(BS\_PUSHBUTTON, L"Button", L"Enter",**

**WS\_CHILD | WS\_VISIBLE | WS\_BORDER,**

**200, 34, 200, 34,**

**hWnd, (HMENU)1, NULL, NULL);**

**page = 1;**

**}**

**else if (mode == 5)**

**{**

**SendButton = CreateWindowEx(BS\_PUSHBUTTON, L"Button", L"Enter",**

**WS\_CHILD | WS\_VISIBLE | WS\_BORDER,**

**200, 34, 200, 34,**

**hWnd, (HMENU)2, NULL, NULL);**

**page = 1;**

**}**

**}**

**}**

**}**

***// when clicking on the main screens to get to play the game.***

**void selection(HWND hWnd, HDC hdc, int xPos, int yPos)**

**{**

**RECT rcCell;**

**index = GetCellNumberFromPoint(hWnd, xPos, yPos);**

**if (NULL != hdc)**

**{**

**if ((0 == board[index]) && GetCellRect(hWnd, index, &rcCell))**

**{**

**FillRect(hdc, &rcCell, hbr1);**

**time(hWnd, hdc);**

**if (mode == 6)**

**{**

**Register(hWnd, hdc);**

**}**

**else if (mode == 5)**

**{**

**int ret = MessageBox(hWnd, L"Player 2 would you like to Sing In?", L"Sing In", MB\_YESNO | MB\_ICONQUESTION);**

**if (IDYES == ret)**

**{**

**Register(hWnd, hdc);**

**mode = 5;**

**}**

**else**

**{**

**page = 1;**

**Register(hWnd, hdc);**

**mode = 0;**

**}**

**}**

**else**

**{**

**mode = 1;**

**}**

**reset(hWnd, hdc);**

**}**

**else if ((1 == board[index]) && GetCellRect(hWnd, index, &rcCell))**

**{**

**FillRect(hdc, &rcCell, hbr2);**

**time(hWnd, hdc);**

**if (mode == 6)**

**{**

**page = 2;**

**Register(hWnd, hdc);**

**mode = 5;**

**}**

**else if (mode == 5)**

**mode = 4;**

**else**

**mode = 2;**

**reset(hWnd, hdc);**

**}**

**else if ((2 == board[index]) && GetCellRect(hWnd, index, &rcCell))**

**{**

**FillRect(hdc, &rcCell, hbr4);**

**time(hWnd, hdc);**

**if (mode == 6)**

**DestroyWindow(hWnd);**

**else if (mode == 5)**

**{**

**system("Score.txt");**

**}**

**else if (mode == 4)**

**mode = 3;**

**reset(hWnd, hdc);**

**}**

**else if ((3 == board[index]) && GetCellRect(hWnd, index, &rcCell))**

**{**

**FillRect(hdc, &rcCell, hbr3);**

**time(hWnd, hdc);**

**if (mode == 5)**

**mode = 6;**

**else**

**mode = 5;**

**reset(hWnd, hdc);**

**}**

**}**

**}**

***// In the middle of a game is user wants to quit they will be warn that it will be a lost***

**BOOL warning(HWND hWnd, HDC hdc, int &x)**

**{**

**file.open("Score.txt", fstream::app);**

**if (playerTurn == 1)**

**{**

**int ret = MessageBox(hWnd, L"Will be consider the loser", L"Player 1", MB\_YESNO | MB\_ICONWARNING);**

**if (IDYES == ret)**

**{**

**if (playerTurn == 2)**

**{**

**file.write((char\*)szInput2, 20);**

**file << " Wins \n";**

**}**

**else**

**{**

**file.write((char\*)szPlayer3, 20);**

**file << " Wins \n";**

**}**

**file.write((char\*)szInput1, 20);**

**file << " Loses \n";**

**file.close();**

**x = TRUE;**

**return x;**

**}**

**else**

**{**

**file.close();**

**x = FALSE;**

**return x;**

**}**

**}**

**else if (playerTurn == 2)**

**{**

**int ret = MessageBox(hWnd, L"Will be consider the loser", L"Player 2", MB\_YESNO | MB\_ICONWARNING);**

**if (IDYES == ret)**

**{**

**file.write((char\*)szInput1, 20);**

**file << " Wins \n";**

**file.write((char\*)szInput2, 20);**

**file << " Loses \n";**

**file.close();**

**x = TRUE;**

**return x;**

**}**

**else**

**{**

**file.close();**

**x = FALSE;**

**return x;**

**}**

**}**

**}**

***// Main fuction this create all the GUI for the user***

**LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)**

**{**

**int x;**

**switch (message)**

**{**

***//create a brush to user as player***

**case WM\_CREATE:**

**{**

**hbr1 = CreateSolidBrush(RGB(255, 0, 0));**

**hbr2 = CreateSolidBrush(RGB(0, 0, 255));**

**hbr3 = CreateSolidBrush(RGB(0, 225, 0));**

**hbr4 = CreateSolidBrush(RGB(200, 200, 200));**

**}**

**case WM\_COMMAND:**

**{**

**HDC hdc = GetDC(hWnd);**

**int wmId = LOWORD(wParam);**

***//options for when user want to sing in***

**switch (LOWORD(wParam))**

**{**

**case 1:**

**// Obtains input from the textbox and puts it into the char array**

**GetWindowText(GetDlgItem(hWnd, 101), szInput1, MAX\_PATH);**

**DestroyWindow(TextBox);**

**DestroyWindow(SendButton);**

**mode = 5;**

**page = 0;**

**reset(hWnd, hdc);**

**break;**

**case 2:**

***// Obtains input from the textbox and puts it into the char array***

**GetWindowText(GetDlgItem(hWnd, 101), szInput2, MAX\_PATH);**

**DestroyWindow(TextBox);**

**DestroyWindow(SendButton);**

**mode = 0;**

**page = 0;**

**reset(hWnd, hdc);**

**break;**

**}**

***// Menu selection on the top of the screen***

**switch (wmId)**

**{**

**case ID\_SCORE:**

**{**

**system("Score.txt");**

**}**

**break;**

**case ID\_NEWGAME:**

**{**

**if (page == 0)**

**{**

**int ret = MessageBox(hWnd, L"Are you sure you want to start a new game?", L"New Game", MB\_YESNO | MB\_ICONQUESTION);**

**if (IDYES == ret)**

**{**

**if (winner == 0 && mode < 4)**

**{**

**warning(hWnd, hdc, x);**

**if (x == TRUE)**

**{**

**mode = 0;**

**reset(hWnd, hdc);**

**}**

**}**

**else**

**{**

**mode = 0;**

**reset(hWnd, hdc);**

**}**

**}**

**}**

**}**

**break;**

**case ID\_SINGLEPLAYER\_EASY:**

**{**

**if (page == 0)**

**{**

**int ret = MessageBox(hWnd, L"Are you sure you want to start a new game?", L"New Game", MB\_YESNO | MB\_ICONQUESTION);**

**if (IDYES == ret)**

**{**

**if (mode == 0)**

**guest = 1;**

**if (winner == 0 && mode < 4)**

**{**

**warning(hWnd, hdc, x);**

**if (x == TRUE)**

**{**

**mode = 1;**

**reset(hWnd, hdc);**

**}**

**}**

**else**

**{**

**mode = 1;**

**reset(hWnd, hdc);**

**}**

**}**

**}**

**}**

**break;**

**case ID\_SINGLEPLAYER\_MEDIUM:**

**{**

**if (page == 0)**

**{**

**int ret = MessageBox(hWnd, L"Are you sure you want to start a new game?", L"New Game", MB\_YESNO | MB\_ICONQUESTION);**

**if (IDYES == ret)**

**{**

**if (mode == 0)**

**guest = 1;**

**if (winner == 0 && mode < 4)**

**{**

**warning(hWnd, hdc, x);**

**if (x == TRUE)**

**{**

**mode = 2;**

**reset(hWnd, hdc);**

**}**

**}**

**else**

**{**

**mode = 2;**

**reset(hWnd, hdc);**

**}**

**}**

**}**

**}**

**break;**

**case ID\_SINGLEPLAYER\_HARD:**

**{**

**if (page == 0)**

**{**

**int ret = MessageBox(hWnd, L"Are you sure you want to start a new game?", L"New Game", MB\_YESNO | MB\_ICONQUESTION);**

**if (IDYES == ret)**

**{**

**if (mode == 0)**

**guest = 1;**

**if (winner == 0 && mode < 4)**

**{**

**warning(hWnd, hdc, x);**

**if (x == TRUE)**

**{**

**mode = 3;**

**reset(hWnd, hdc);**

**}**

**}**

**else**

**{**

**mode = 3;**

**reset(hWnd, hdc);**

**}**

**}**

**}**

**}**

**break;**

**case ID\_MAINMENU:**

**{**

**if (page == 0)**

**{**

**int ret = MessageBox(hWnd, L"Are you sure you want to go to the Main Menu?", L"Main Menu", MB\_YESNO | MB\_ICONQUESTION);**

**if (IDYES == ret)**

**{**

**if (winner == 0 && mode < 4)**

**{**

**warning(hWnd, hdc, x);**

**if (x == TRUE)**

**{**

**mode = 6;**

**reset(hWnd, hdc);**

**}**

**}**

**else**

**{**

**mode = 6;**

**reset(hWnd, hdc);**

**}**

**}**

**}**

**}**

**break;**

**case IDM\_ABOUT:**

**if (page == 0)**

**{**

**DialogBox(hInst, MAKEINTRESOURCE(IDD\_ABOUTBOX), hWnd, About);**

**}**

**break;**

**case IDM\_EXIT:**

**if (page == 0)**

**{**

**if (winner == 0 && mode < 4)**

**{**

**warning(hWnd, hdc, x);**

**if (x == TRUE)**

**{**

**DestroyWindow(hWnd);**

**}**

**}**

**else**

**{**

**DestroyWindow(hWnd);**

**}**

**}**

**break;**

**default:**

**return DefWindowProc(hWnd, message, wParam, lParam);**

**}**

**}**

**break;**

**//When user click on the mouse**

**case WM\_LBUTTONDOWN:**

**{**

**if (page == 0)**

**{**

**int xPos = GET\_X\_LPARAM(lParam);**

**int yPos = GET\_Y\_LPARAM(lParam);**

**HDC hdc = GetDC(hWnd);**

**if (mode >= 4)**

**selection(hWnd, hdc, xPos, yPos);**

**else if (mode >= 1)**

**{**

**Move(hWnd, hdc, xPos, yPos);**

**}**

**else**

**{**

**Moves(hWnd, hdc, xPos, yPos);**

**}**

**}**

**}**

**break;**

**case WM\_LBUTTONUP:**

**{**

**if (page == 0)**

**{**

**int xPos = GET\_X\_LPARAM(lParam);**

**int yPos = GET\_Y\_LPARAM(lParam);**

**HDC hdc = GetDC(hWnd);**

**PCMoves(hWnd, hdc);**

**}**

**}**

**break;**

**case WM\_GETMINMAXINFO:**

**{**

**MINMAXINFO \* pMinMax = (MINMAXINFO\*)lParam;**

**pMinMax->ptMinTrackSize.x = (CELL\_SIZE \* 7);**

**pMinMax->ptMinTrackSize.y = (CELL\_SIZE \* 7);**

**}**

**case WM\_PAINT:**

**{**

**PAINTSTRUCT ps;**

**HDC hdc = BeginPaint(hWnd, &ps);**

**RECT rc;**

**if (mode >= 4)**

**{**

**if (GetGameBoard(hWnd, &rc))**

**{**

**Rectangle(hdc, rc.left, rc.top, rc.right, rc.bottom);**

**}**

**for (int i = 0; i < 4; i++)**

**Horizon(hdc, rc.left, rc.top + CELL\_SIZE \*i, rc.right, rc.top + CELL\_SIZE \*i);**

***//Display the first screen***

**if (mode == 6)**

**{**

**select[1] = L"Sing In";**

**select[2] = L"Play as guest";**

**select[3] = L"Exit Game";**

**SetTextColor(hdc, RGB(0, 0, 0));**

**TextOut(hdc, rc.left + CELL\_SIZE \* 1, rc.top + CELL\_SIZE \* .5, select[1], lstrlen(select[1]));**

**TextOut(hdc, rc.left + CELL\_SIZE \* 1, rc.top + CELL\_SIZE \* 1.5, select[2], lstrlen(select[2]));**

**TextOut(hdc, rc.left + CELL\_SIZE \* 1, rc.top + CELL\_SIZE \* 2.5, select[3], lstrlen(select[3]));**

**}**

***//second screen***

**if (mode == 5)**

**{**

**select[1] = L"Multi-Player";**

**select[2] = L"Single Player";**

**select[3] = L"Score Player";**

**select[4] = L"Register";**

**SetTextColor(hdc, RGB(0, 0, 0));**

**TextOut(hdc, rc.left + CELL\_SIZE \*1.5, rc.top + CELL\_SIZE \* .5, select[1], lstrlen(select[1]));**

**TextOut(hdc, rc.left + CELL\_SIZE \*1.5, rc.top + CELL\_SIZE \* 1.5, select[2], lstrlen(select[2]));**

**TextOut(hdc, rc.left + CELL\_SIZE \*1.5, rc.top + CELL\_SIZE \* 2.5, select[3], lstrlen(select[3]));**

**TextOut(hdc, rc.left + CELL\_SIZE \*1.5, rc.top + CELL\_SIZE \* 3.5, select[4], lstrlen(select[4]));**

**}**

***// third if wanting to play with AI***

**if (mode == 4)**

**{**

**select[1] = L"Easy";**

**select[2] = L"Medium";**

**select[3] = L"Hard";**

**select[4] = L"Main Menu";**

**SetTextColor(hdc, RGB(0, 0, 0));**

**TextOut(hdc, rc.left + CELL\_SIZE \*1.5, rc.top + CELL\_SIZE \* .5, select[1], lstrlen(select[1]));**

**TextOut(hdc, rc.left + CELL\_SIZE \*1.5, rc.top + CELL\_SIZE \* 1.5, select[2], lstrlen(select[2]));**

**TextOut(hdc, rc.left + CELL\_SIZE \*1.5, rc.top + CELL\_SIZE \* 2.5, select[3], lstrlen(select[3]));**

**TextOut(hdc, rc.left + CELL\_SIZE \*1.5, rc.top + CELL\_SIZE \* 3.5, select[4], lstrlen(select[4]));**

**}**

**}**

**else**

**{**

**if (GetGameBoard(hWnd, &rc))**

**{**

**RECT rcClient;**

**if (GetClientRect(hWnd, &rcClient))**

**{**

***//display Player's name***

**SetBkMode(hdc, TRANSPARENT);**

**if (mode > 0)**

**{**

**SetTextColor(hdc, RGB(255, 0, 0));**

**TextOut(hdc, 18, 20, szPlayer1[player1], lstrlen(szPlayer1[player1]));**

**SetTextColor(hdc, RGB(0, 255, 0));**

**TextOut(hdc, rcClient.right - 80, 20, szPlayer3, ARRAYSIZE(szPlayer3));**

**}**

**if (mode == 0)**

**{**

**SetTextColor(hdc, RGB(255, 0, 0));**

**TextOut(hdc, 18, 20, szPlayer1[player1], lstrlen(szPlayer1[player1]));**

**SetTextColor(hdc, RGB(0, 0, 255));**

**TextOut(hdc, rcClient.right - 80, 20, szPlayer2[player2], lstrlen(szPlayer2[player2]));**

**}**

**ShowTurn(hWnd, hdc);**

**}**

***//Remove boarders***

**FillRect(hdc, &rc, (HBRUSH)GetStockObject(WHITE\_BRUSH));**

***//Rectangle(hdc, rc.left, rc.top, rc.right, rc.bottom);***

**}**

**for (int i = 1; i < 5; ++i)**

**{**

***//vertical lines***

**Drawlines(hdc, rc.left + CELL\_SIZE \* i, rc.top, rc.left + CELL\_SIZE \*i, rc.bottom);**

***//Horizantal line***

**Drawlines(hdc, rc.left, rc.top + CELL\_SIZE \* i, rc.right, rc.top + CELL\_SIZE \* i);**

**}**

**for (int i = 0; i < 25; i++)**

**{**

**RECT rcCell;**

***//action stays after re-size page***

**if (0 != gameBoard[i] && GetCellRect(hWnd, i, &rcCell) && mode == 0)**

**{**

**FillRect(hdc, &rcCell, (gameBoard[i] == 2) ? hbr2 : hbr1);**

**}**

**else if (0 != gameBoard[i] && GetCellRect(hWnd, i, &rcCell) && mode > 0)**

**{**

**FillRect(hdc, &rcCell, (gameBoard[i] == 4) ? hbr3 : hbr1);**

**}**

**}**

**}**

***// TODO: Add any drawing code that uses hdc here...***

**EndPaint(hWnd, &ps);**

**}**

**break;**

**case WM\_DESTROY:**

**DeleteObject(hbr1);**

**DeleteObject(hbr2);**

**DeleteObject(hbr3);**

**DeleteObject(hbr4);**

**PostQuitMessage(0);**

**break;**

**default:**

**return DefWindowProc(hWnd, message, wParam, lParam);**

**}**

**return 0;**

**}**

***// Message handler for about box.***

**INT\_PTR CALLBACK About(HWND hDlg, UINT message, WPARAM wParam, LPARAM lParam)**

**{**

**UNREFERENCED\_PARAMETER(lParam);**

**switch (message)**

**{**

**case WM\_INITDIALOG:**

**return (INT\_PTR)TRUE;**

**case WM\_COMMAND:**

**if (LOWORD(wParam) == IDOK || LOWORD(wParam) == IDCANCEL)**

**{**

**EndDialog(hDlg, LOWORD(wParam));**

**return (INT\_PTR)TRUE;**

**}**

**break;**

**}**

**return (INT\_PTR)FALSE;**

**}**

# Acknowledgements