МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ ДНР

ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Институт КНТ

Факультет ФИСП

Кафедра ПИ им. Л.П. Фельдмана

Лабораторная работа №2

Тема: Создание пользовательского интерфейса для задачи имитационного моделирования

Курс: Системное программирование

Выполнил

ст. гр. ПИ-18б

Моргунов А.Г.

Проверил

Филипишин Д.А.

Донецк – 2021

**Задание**

5. Автомат Калашникова.

Автомат имеет спусковой механизм (курок) и рычаг переключения «одиночный выстрел» ‑ «автоматическая стрельба». При нажатии на курок в зависимости от рычага переключения автомат выстреливает:

* 1 пулю при режиме «одиночный выстрел»;
* 5 пуль при режиме «автоматическая стрельба».

Если обойма пуста, то стрельба производиться не может и необходимо сменить обойму. Ёмкость обоймы 30 патронов. Если в обойме осталось менее 5 патронов, то при «автоматической стрельбе» выстреливаются все остающиеся. Две обоймы заправлять одновременно нельзя. Начальное состояние: в обойме 30 патронов, рычаг переключен в состояние «одиночный выстрел».

**Код**

#define UNICODE

#include <iostream>

#include <tchar.h>

#include <windows.h>

#include "MachineGun.hpp"

#define ID\_FIRE\_BUTTON 1

#define ID\_RELOAD\_BUTTON 2

#define ID\_SWITCH\_FIRE\_MODE\_BUTTON 3

MachineGun machineGun{};

LRESULT CALLBACK MainWndProc(HWND, UINT, WPARAM, LPARAM);

WCHAR mainWindowName[] = \_T(L"MainWindow");

WCHAR fireButtonName[] = \_T(L"FireButton");

WCHAR switchFireButtonName[] = \_T(L"SwitchFireButton");

WCHAR reloadButtonName[] = \_T(L"ReloadButton");

HWND labelAmmo;

HWND labelFireMode;

int WINAPI WinMain(HINSTANCE This,

HINSTANCE Prev,

LPSTR cmd,

int mode) {

WNDCLASS wc;

memset(&wc, 0, sizeof(wc));

wc.hInstance = This;

wc.lpszClassName = mainWindowName;

wc.lpfnWndProc = MainWndProc;

wc.style = CS\_HREDRAW | CS\_VREDRAW;

wc.hIcon = LoadIcon(NULL, IDI\_APPLICATION);

wc.hCursor = LoadCursor(NULL, IDC\_ARROW);

// wc.hbrBackground = (HBRUSH)(COLOR\_WINDOW + 1);

RegisterClass(&wc);

auto static mainWindow = CreateWindow(mainWindowName,

\_T(L"AK-47 Simulation"),

WS\_OVERLAPPEDWINDOW,

0,

0,

500,

500,

HWND\_DESKTOP,

NULL,

This,

NULL);

ShowWindow(mainWindow, mode);

labelAmmo = CreateWindow(\_T(L"static"),

\_T(L"Ammo: 30"),

WS\_CHILD | WS\_VISIBLE,

0,

0,

100,

30,

mainWindow,

NULL,

NULL,

NULL);

ShowWindow(labelAmmo, mode);

labelFireMode = CreateWindow(\_T(L"static"),

\_T(L"FireMode: Single"),

WS\_CHILD | WS\_VISIBLE,

105,

0,

200,

30,

mainWindow,

NULL,

NULL,

NULL);

ShowWindow(labelFireMode, mode);

auto fireButton = CreateWindow(\_T(L"button"),

\_T(L"Fire"),

WS\_CHILD | WS\_VISIBLE,

0,

35,

100,

30,

mainWindow,

(HMENU)ID\_FIRE\_BUTTON,

NULL,

NULL);

ShowWindow(fireButton, mode);

auto reloadButton = CreateWindow(\_T(L"button"),

\_T(L"Reload"),

WS\_CHILD | WS\_VISIBLE,

0,

70,

100,

30,

mainWindow,

(HMENU)ID\_RELOAD\_BUTTON,

NULL,

NULL);

ShowWindow(reloadButton, mode);

auto switchFireModeButton = CreateWindow(\_T(L"button"),

\_T(L"Switch fire mode"),

WS\_CHILD | WS\_VISIBLE,

105,

35,

200,

30,

mainWindow,

(HMENU)ID\_SWITCH\_FIRE\_MODE\_BUTTON,

NULL,

NULL);

ShowWindow(switchFireModeButton, mode);

MSG msg;

while(GetMessage(&msg, mainWindow, 0, 0)) {

DispatchMessage(&msg);

}

return 0;

}

LRESULT CALLBACK MainWndProc(HWND hWnd, UINT msg,

WPARAM wParam, LPARAM lParam){

switch(msg){

case WM\_COMMAND:

switch(LOWORD(wParam)) {

case ID\_FIRE\_BUTTON: {

int currentAmmo = machineGun.Fire();

auto currentAmmoString = std::to\_wstring(currentAmmo);

auto newText = std::wstring(L"Ammo: ") + currentAmmoString;

SetWindowText(labelAmmo, newText.c\_str());

}

break;

case ID\_RELOAD\_BUTTON: {

int currentAmmo = machineGun.Reload();

auto currentAmmoString = std::to\_wstring(currentAmmo);

auto newText = std::wstring(L"Ammo: ") + currentAmmoString;

SetWindowText(labelAmmo, newText.c\_str());

}

break;

case ID\_SWITCH\_FIRE\_MODE\_BUTTON:

{

auto currentFireModeString = machineGun.SwitchFireMode();

auto newText = std::wstring(L"Fire mode: ") + currentFireModeString;

SetWindowText(labelFireMode, newText.c\_str());

}

break;

}

break;

case WM\_DESTROY:

PostQuitMessage(0);

break;

default:

return DefWindowProc(hWnd, msg, wParam, lParam);

}

return 0;

}

//

// Created by Arsenii on 14.11.2021.

//

#include "MachineGun.hpp"

#include <math.h>

int MachineGun::Fire() {

if (clip == 0) {

return 0;

}

if (fireMode == FireMode::single) {

--clip;

} else {

clip -= std::min(clip, autoFireCount);

}

return clip;

}

int MachineGun::Reload() {

clip = clipSize;

return clip;

}

std::wstring MachineGun::SwitchFireMode() {

if (fireMode == FireMode::single) {

fireMode = FireMode::automatic;

return std::wstring(L"Automatic");

} else {

fireMode = FireMode::single;

return std::wstring(L"Single");

}

}

//

// Created by Arsenii on 14.11.2021.

//

#ifndef SP\_LAB\_2\_\_MACHINEGUN\_HPP\_

#define SP\_LAB\_2\_\_MACHINEGUN\_HPP\_

#include <iostream>

class MachineGun {

public:

std::wstring SwitchFireMode();

int Reload();

int Fire();

private:

enum FireMode {

single,

automatic

};

int fireMode{FireMode::single};

int clip{30};

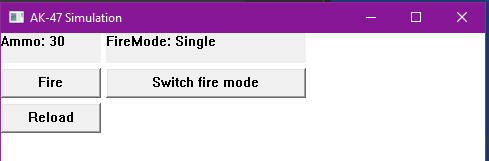
int clipSize{30};

int autoFireCount{5};

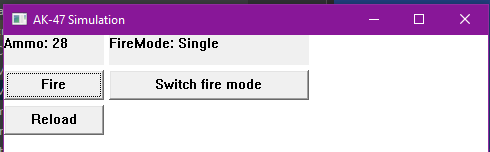
};

#endif //SP\_LAB\_2\_\_MACHINEGUN\_HPP\_

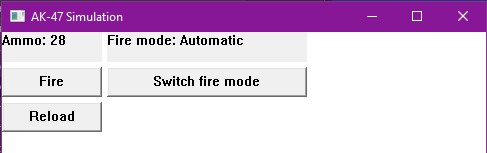
**Скриншоты**



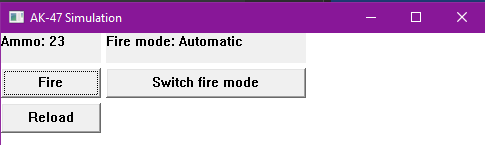
Fire



Switch fire mode



Fire



Reload

