

ПРАКТИЧНА РОБОТА 6



Тема: Створення додатку Windows Forms в Visual Studio 2019.
Додавання декількох форм.

Мета роботи: вивчити прийоми створення нового додатку Windows Forms в Visual C++. Засвоїти методи компіляції, запуску та відлагоджування проекту Windows Forms.

Методичне, програмне забезпечення і ТЗН: Інструкція до практичної роботи №6, персональний комп'ютер, ОС Windows 7/8.1/10, Microsoft Office 2010-2019, Adobe Reader, Visual Studio 2019.



Короткі теоретичні відомості

Платформа .NET Framework – це інтегрований компонент Windows, який підтримує розробку та виконання різних типів додатків і веб-служб. Основними компонентами .NET Framework є загальномовне середовище виконання (Common Language Runtime, CLR) і бібліотека класів .NET Framework, яка включає набори класів, що підтримують наступні основні технології: ADO.NET, ASP.NET, Windows Forms і Windows Presentation Foundation (WPF).

Середовище виконання CLR можна вважати агентом, який управляє кодом під час виконання і надає основні служби, такі як управління пам'яттю, управління потоками і віддаленою взаємодією.

Інший основний компонент платформи .NET Framework, бібліотека класів, надає засоби, які застосовуються для розроблення додатків, починаючи від звичайних, що запускаються з командного рядка і закінчуючи додатками, що використовують останні можливості розроблення для інтернет (технологія ASP.NET) такі як Web Forms і веб-служби XML.

Технологія ADO.NET має багатий набір компонентів для створення різних додатків, зокрема розподілених додатків, надає доступ до реляційних даних, XML-даних, які використовуються для побудови веб-додатків. ASP.NET – це технологія для розроблення веб-додатків, містить служби, необхідні для їх побудови. За допомогою Windows Forms можна створювати додатки з елементами уніфікованого інтерфейсу Windows. Windows Presentation Foundation забезпечує підтримку високої якості двомірної та тривимірної графіки, анімації та мультимедіа при розробці додатків.

В .NET Framework використовують поняття збірки. Збірки є структурними елементами додатків .NET Framework; вони складають основну одиницю розгортання, керування версіями, повторного використання, областей дії активації та дозволів безпеки. Збірка являє собою колекцію типів і ресурсів, зібраних для спільної роботи, які утворюють логічну функціональну одиницю. Збірка надає середовищу CLR відомості, необхідні для розпізнавання реалізацій типів. Збірка містить код, що виконується середовищем CLR. Кожна збірка може мати тільки одну точку входу (тобто DllMain, WinMain або Main). Маніфест збірки містить метадані, які використовуються для визначення областей дії типів і для виконання запитів, пов'язаних з ресурсами. У маніфесті вказуються типи і ресурси, які експортуються за межі збірки.

Додатки Windows створюють у інтегрованому середовищі розроблення Microsoft Visual Studio. Visual Studio дозволяє розробляти як консольні додатки, так і додатки з графічним інтерфейсом, а також веб-сайти, веб-додатки, веб-служби для платформ: Microsoft Windows, Windows Mobile, Windows CE, .NET Framework, .NET Compact Framework і Microsoft Silverlight (платформа для підтримки графіки). Додаток Windows в Visual Studio будується на основі багатофункціонального набору класів .NET Framework середовища CLR. Додатки Windows можна створювати за допомогою декількох мов програмування .NET Framework (Visual Basic, C#, Visual J #, C++).

Основою додатків є форми (у додатку їх може бути декілька), на яких розміщуються компоненти. Зазвичай це – віконна форма. На форму за допомогою "миші" переносяться і розміщаються компоненти, наявні в бібліотеках середовища програмування. Головна перевага візуального програмування полягає в тому, що під час проектування форми і розміщення на ній компонентів середовище автоматично генерує коди програм, включаючи в них відповідні фрагменти, які описують даний компонент. Розробник створює коди оброблювачів подій, на які реагують розміщені на формі компоненти. Тобто проектування зводиться до розміщення компонентів на формі, задавання їхніх властивостей і написання оброблювачів подій.

Для кожної форми, яка проектується, створюються окремі модулі. Саме в модулях і здійснюється програмування задачі та написання оброблювачів подій. У оброблювачах подій об'єктів (форм і компонентів) розташовуються програми, при цьому вони постійно звертаються до методів (функцій) різноманітних об'єктів.

У Visual Studio реалізовано два види контейнерів для створення додатків: рішення (solutions) і проекти (projects). Об'єкти, що містяться в цих контейнерах, називаються елементами.

Проект – це набір файлів і належних до них метаданих, наприклад, відомостей про з'єднання. Склад файлів в проекті залежить від того, для чого цей проект призначений. Наприклад, проект SQL Server може містити інструкції DDL, що визначають об'єкти в базі даних. Проекти середовища Visual Studio – це контейнери в середині рішення.

Рішення містять елементи, необхідні для створення проектів. Рішення містить один або кілька проектів, а також файли і метадані, які дозволяють визначити рішення як єдине ціле. Visual Studio автоматично створює рішення при створенні нового проекту. У рішенні використовуються проекти логічного керування, побудови та налагодження елементів, які складають додаток. На виході проект, як правило, являє собою виконувану програму (EXE), файл бібліотеки динамічного компонування (DLL) або модуль. Елементами рішень і проектів можуть бути файли та інші складові проекту, наприклад, посилання, підключення до даних або папки.

Форма

Робота над застосуванням починається із створення стартової форми - головного вікна програми.

Спочатку треба встановити необхідні значення властивостей форми, потім - помістити на форму необхідні компоненти (поля введення інформації, командні кнопки, поля відображення тексту та ін.) і виконати.

Налаштування форми (а також компонентів) здійснюється шляхом зміни значень властивостей. Властивості об'єкту (форми, компонента) визначають його вид і поведінку. Наприклад, властивість Text визначає текст заголовка вікна, а властивість StartPosition - положення вікна у момент появи його на екрані.

Основні властивості форми (об'єкту WinForm) приведені в таблицю 1.1.

Таблиця 1.1 – Властивості форми (об'єкту WinForm)

Властивість	Опис
Name	Ім'я форми
Text	Текст в заголовку
Size	Розмір форми. Уточнююча властивість Width визначає ширину, властивість Height - висоту
StartPosition	Положення форми у момент першої появи на екрані. Форма може знаходитися в центрі екрану (CenterScreen), в центрі батьківського вікна (CenterParent). Якщо значення властивості рівне Manual, то положення форми визначається значенням властивості Location

Властивість	Опис
Location	Положення форми на екрані. Відстань від верхньої межі форми до верхньої межі екрану задає уточнюючу властивість Y, відстань від лівої межі форми до лівої межі екрану - уточнюючу властивість X
FormBorderStyle	Тип форми (межі). Форма може бути звичайним вікном (Sizable), вікном фіксованого розміру (FixedSingle, Fixed3D), діалогом (FixedDialog) або вікном без кнопок Згорнути і Розвернути (SizeableToolWindow, FixedToolWindow). Якщо властивості присвоїти значення None, біля вікна не буде заголовка і межі
ControlBox	Управляє відображенням системного меню і кнопок управління вікном. Якщо значення властивості рівне False, то в заголовку вікна кнопка системного меню, а також кнопки Згорнути, Розвернути, Закрити не відображуються
MaximizeBox	Кнопка Розвернути. Якщо значення властивості рівне False, то кнопка, що знаходиться в заголовку вікна, Розвернути недоступна
MinimizeBox	Кнопка Згорнути. Якщо значення властивості рівне False, то кнопка, що знаходиться в заголовку вікна, Згорнути недоступна
Icon	Значок в заголовку вікна
Font	Шрифт, використовуваний за умовчанням компонентами, що знаходяться на поверхні форми. Зміна значення властивості призводить до автоматичної зміни значення властивості Font усіх компонентів форми (за умови, що значення властивості компонента не було задане явно)
ForeColor	Колір, успадкований компонентами форми і використовуваний ними для відображення тексту. Зміна значення властивості призводить до автоматичної зміни відповідної властивості усіх компонентів форми (за умови, що значення властивості Font компонента не було задане явно)
BackColor	Колір фону. Можна задати явно (вибрати на вкладці Custom або Web) або вказати елемент колірної схеми (вибрати на вкладці System)
Opacity	Міра прозорості форми. Форма може бути непрозорою (100%) або прозорою. Якщо значення властивості менше 100%, то крізь форму видно поверхня, на якій вона відображується

Для зміни значень властивостей об'єктів використовується вікно Properties(рисунок 1.1). У лівій колонці вікна перераховані властивості об'єкту, вибраного в даний момент, в правій - вказані значення властивостей. Ім'я вибраного об'єкту відображується у верхній частині вікна Properties.

Деякі властивості є складними. Вони є сукупністю інших (що уточнюють) властивостей. Наприклад, властивістю Size, що визначає розмір форми, є сукупність властивостей Width і Height. Перед іменами складних властивостей стоїть значок ☐,

в результаті клацання на якому розкривається список уточнюючих властивостей. Значення уточнюючої властивості можна задати (змінити) звичайним способом - ввести потрібне значення в поле редагування.

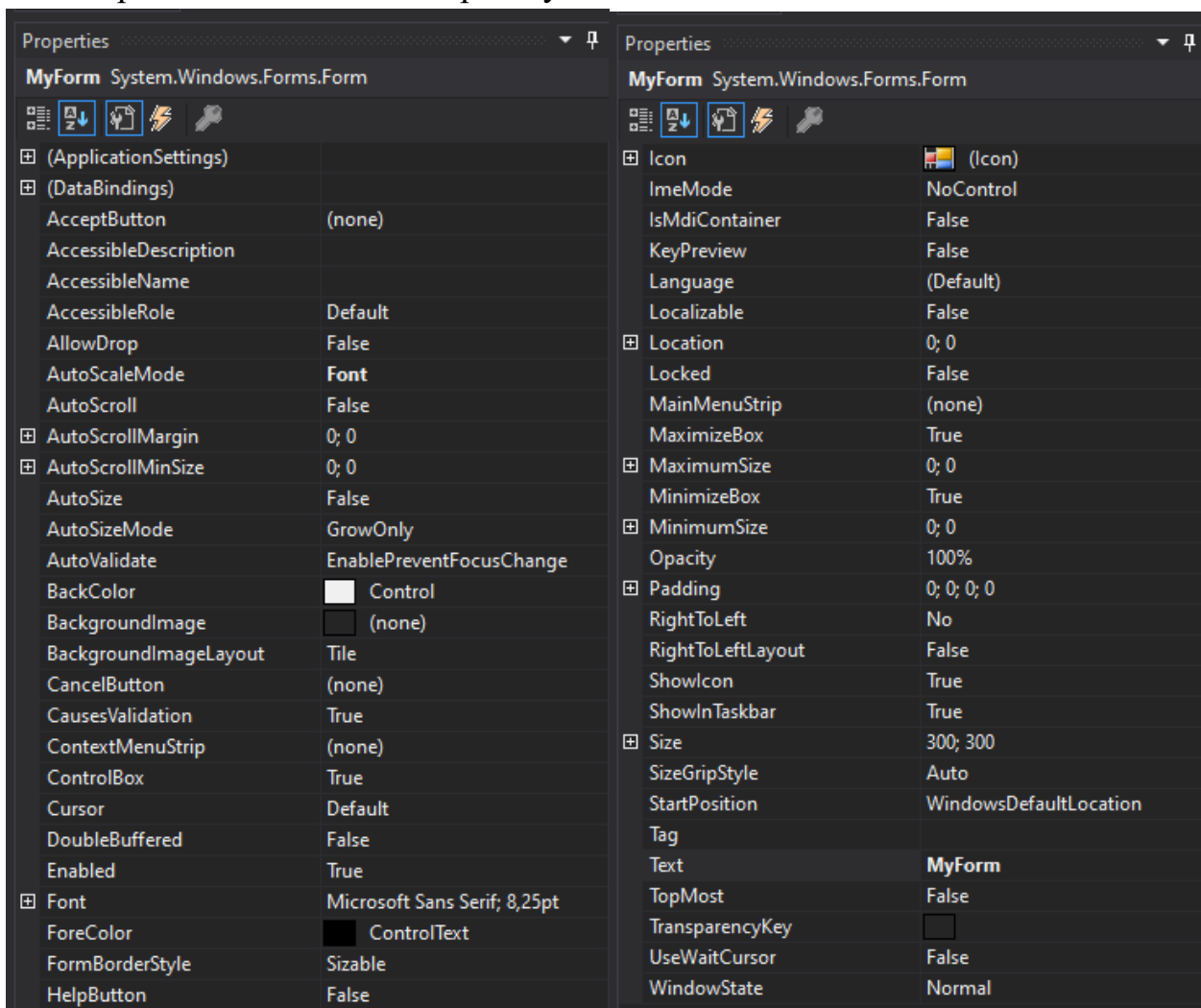


Рисунок 1.1 – вікно «Properties»

Порядок виконання роботи

1.1 Створення нового додатку Windows Forms

Завантажуємо середовище Visual Studio 2019.

При створенні програми на мові Visual C++ в середовищі Visual Studio першим кроком є вибір типу (шаблону) проекту. Для кожного типу проекту Visual Studio задає параметри компілятора і автоматично створює стартовий код.

Створюємо порожній проект. У відмінності від Visual Studio 2010 тут CLR додатки створюються пустими без форми. Вибираємо CLR Empty Project(.NET Framework) (рисунок 1.2).

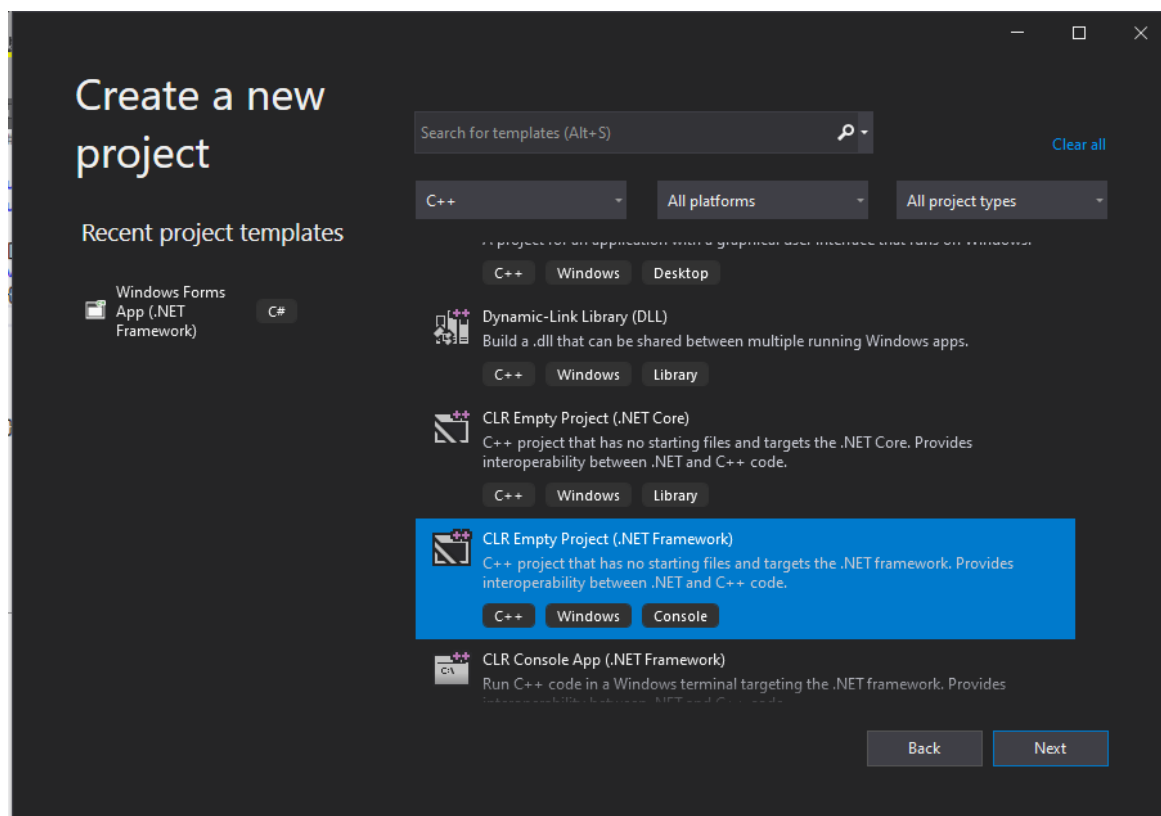


Рисунок 1.2 – Вікно вибору типу проєкту

2 Вказуємо ім'я проєкту та його розташування.

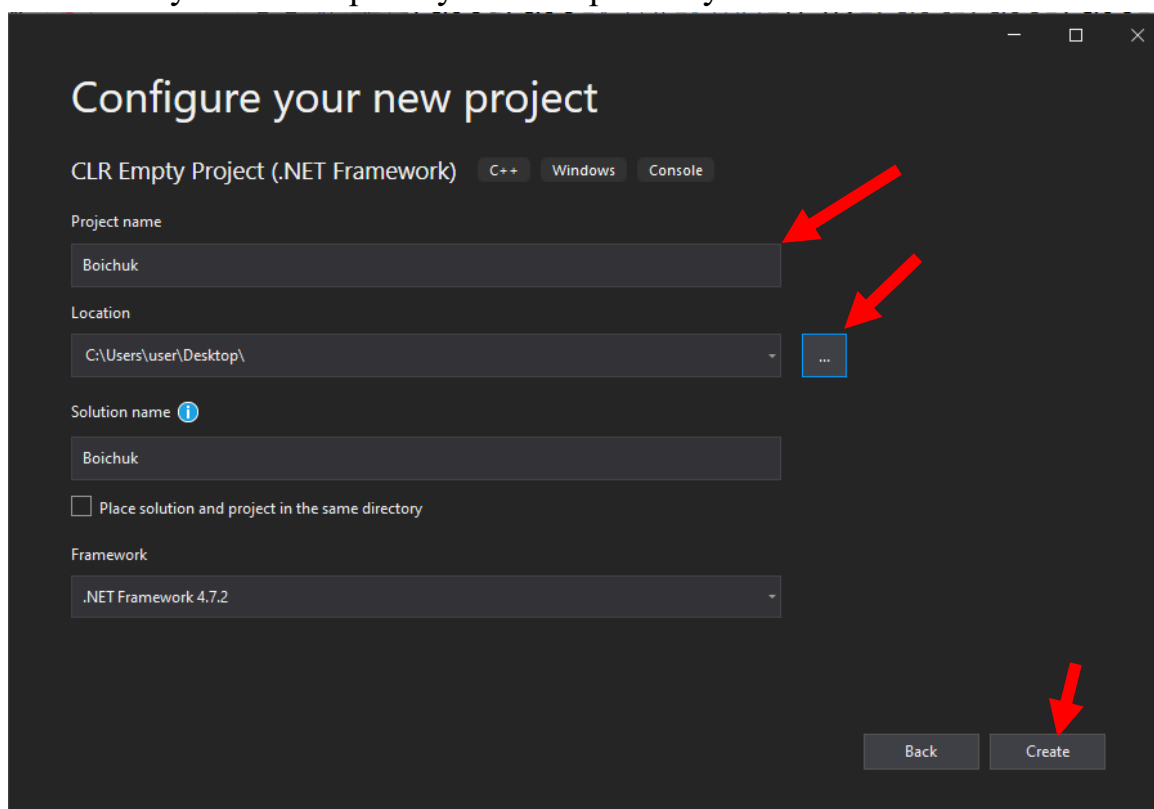


Рисунок 1.3 – Вікно «Налаштування проєкту»

3 Отримуємо свій порожній проєкт

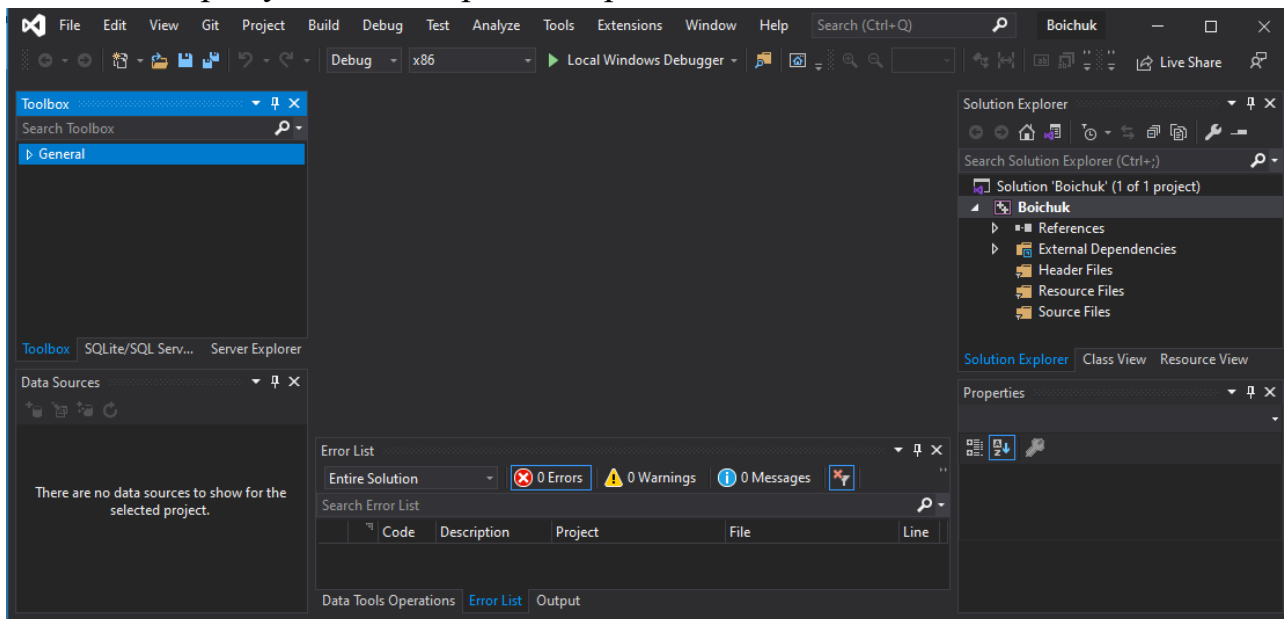


Рисунок 1.4 – Вікно порожнього проєкту у Visual Studio 2019

4 Створюємо новий елемент у проєкті.

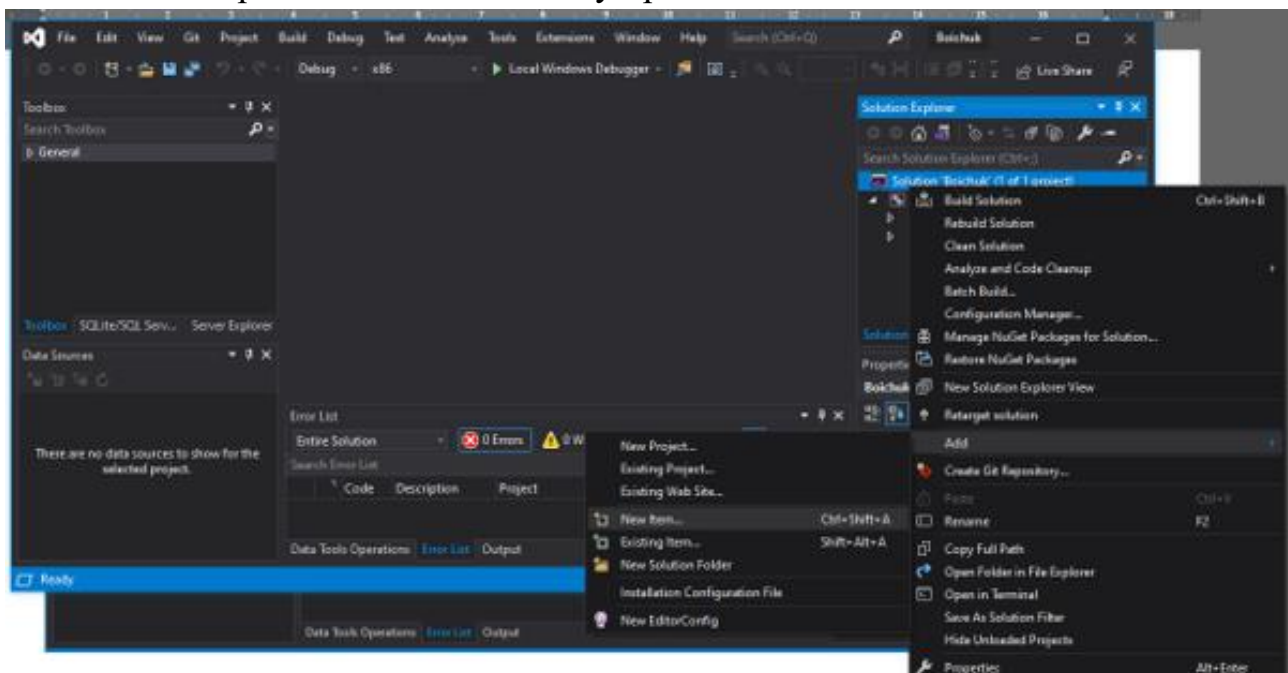


Рисунок 1.5 – Вікно для додавання нового елемента до проєкту

5 Для додавання форми до проекту, тобто створення головного вікна програми оберіть пункти меню **Project, Add New Item**, розкрийте **Visual C++**, оберіть **UI, Windows Form**, в полі **Name** введіть ім'я файлу форми згідно індивідуального завдання (по замовчуванню встановлюється назва **MyForm**).

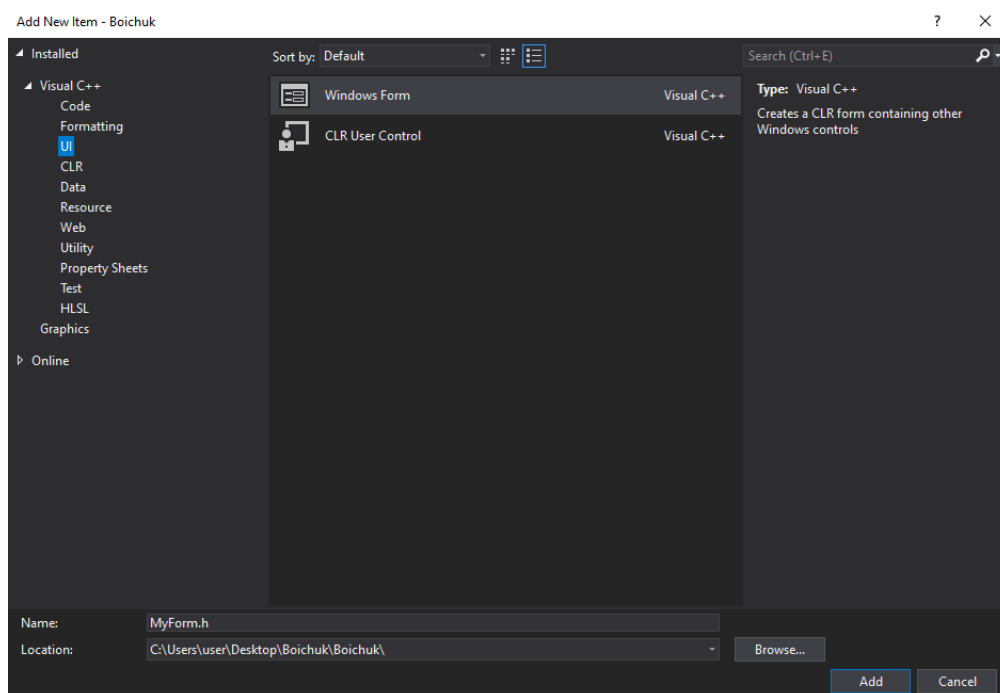


Рисунок 1.6 – Вікно додавання форми до проекту

Якщо у вас немає підтримки UI, потрібно перейти Visual Studio Installer, вибрати команду модифікації проєкту. Переходимо на вкладку «Individual components» знаходимо «C++/CLI support for v142 build tools (14.28)» ставимо галочку та інсталуємо.

6 Скоріш за все, що у вас виникне ось така картинка з помилками.

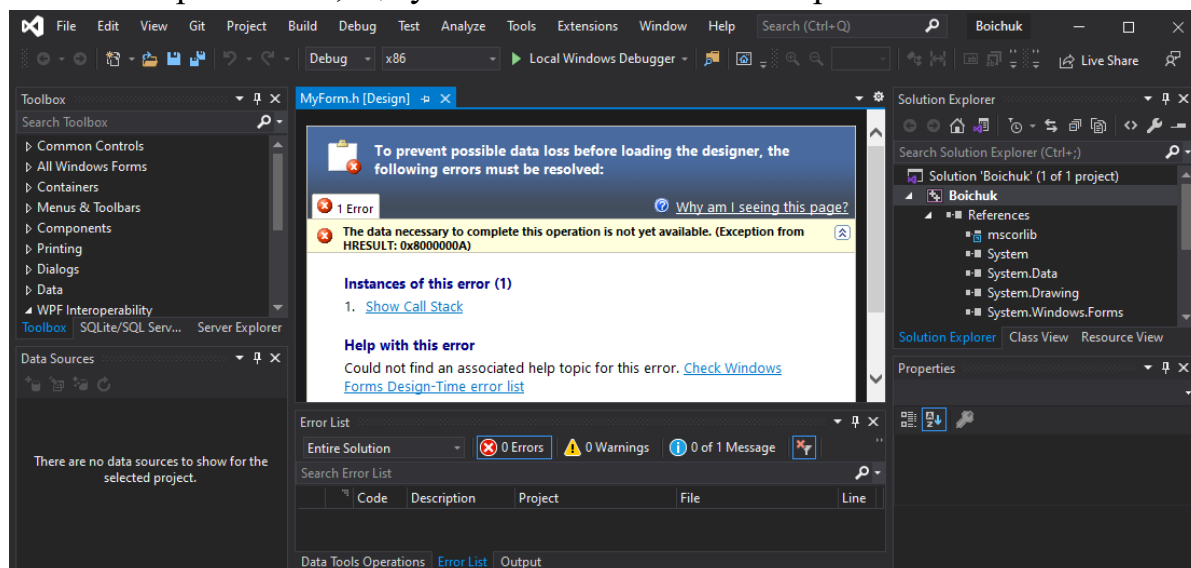


Рисунок 1.7– Вікно з доданою формою

Закриваємо вікно форми.

7 Далі переходимо у файл *.cpp нашої форми. Допишуємо там ось такий код після єдиного рядка (`#include "MyForm.h"`).

```
#include "MyForm.h"
#include <Windows.h>

using namespace Boichuk; //назва проєкту
[STAThreadAttribute]

int WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)
{
    Application::EnableVisualStyles();
    Application::SetCompatibleTextRenderingDefault(false);
    Application::Run(gcnew MyForm);
    return 0;
}
```

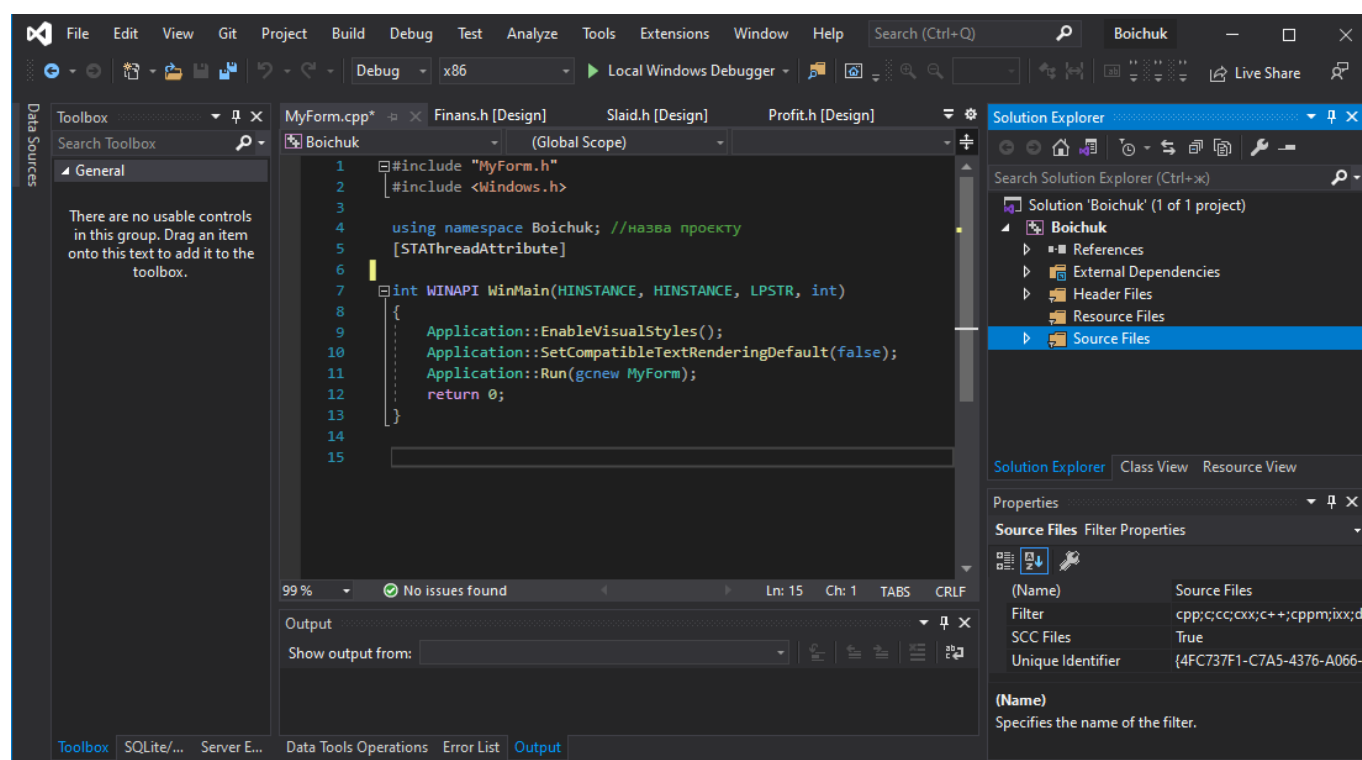


Рисунок 1.8 – Вікно коду проєкту

Зберігаємо код і закриваємо **Visual Studio**.

8 Після завантаження вашого проєкту запускаємо його на виконання. Якщо програма не видає ніяких помилок, отже ви все виконали правильно і ми отримаємо наступний результат.

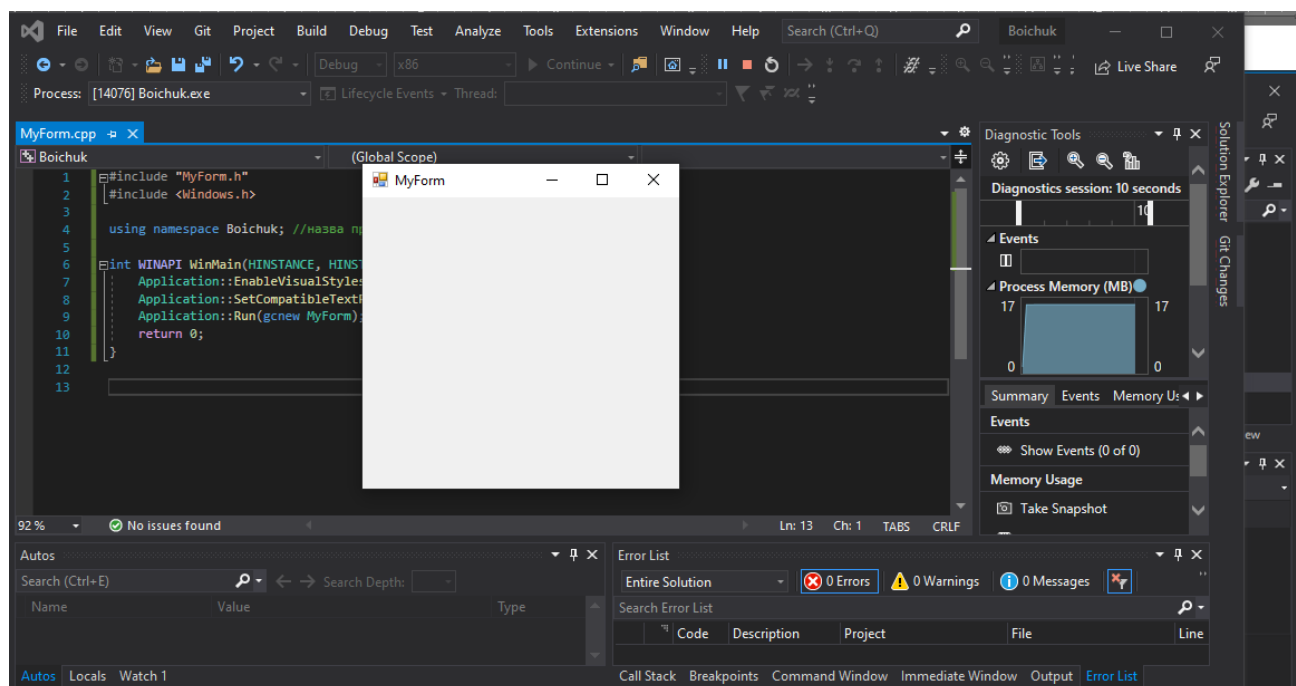


Рисунок 1.9 – Відображення exe-файлу проєкту

9 Для того, щоб відкрити дизайнера форми достатньо клікнути клавiшею миші по файлу MyForm.h.

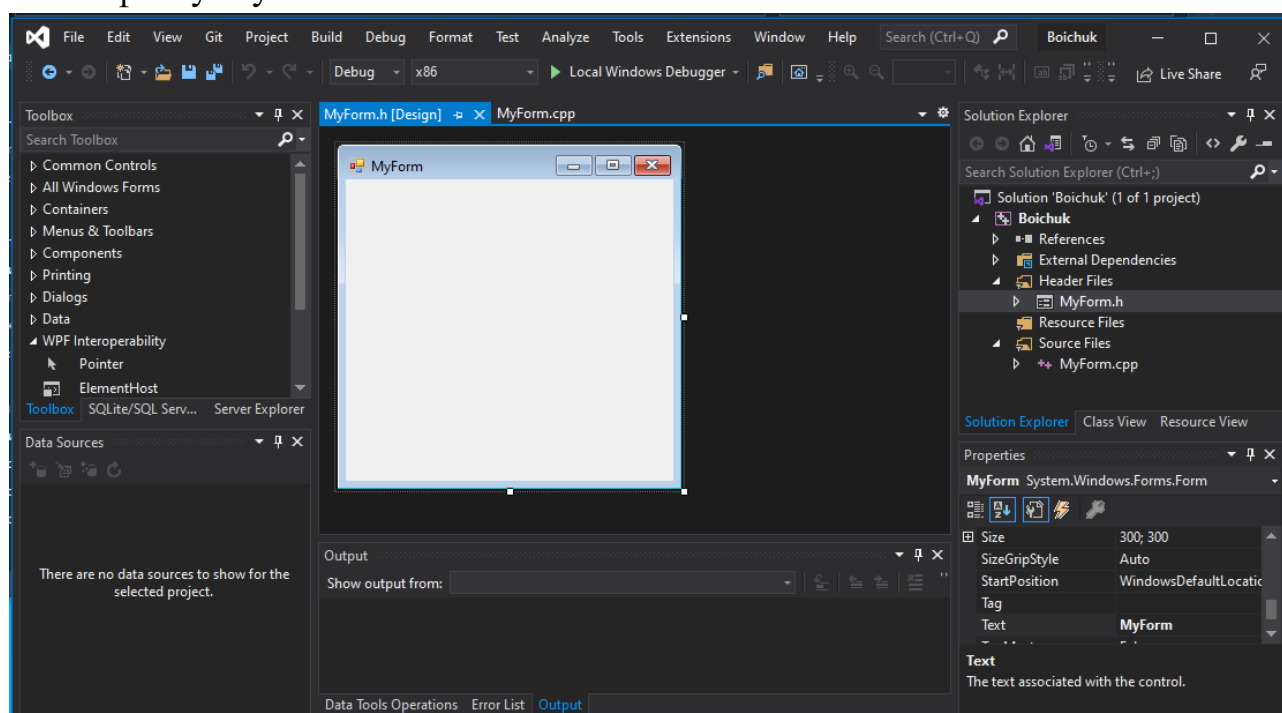
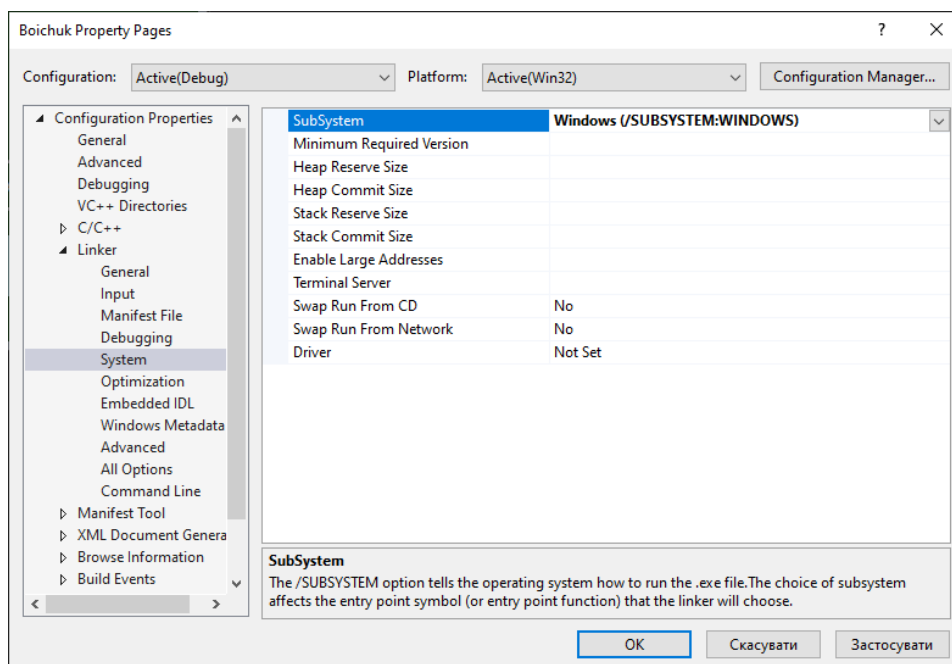


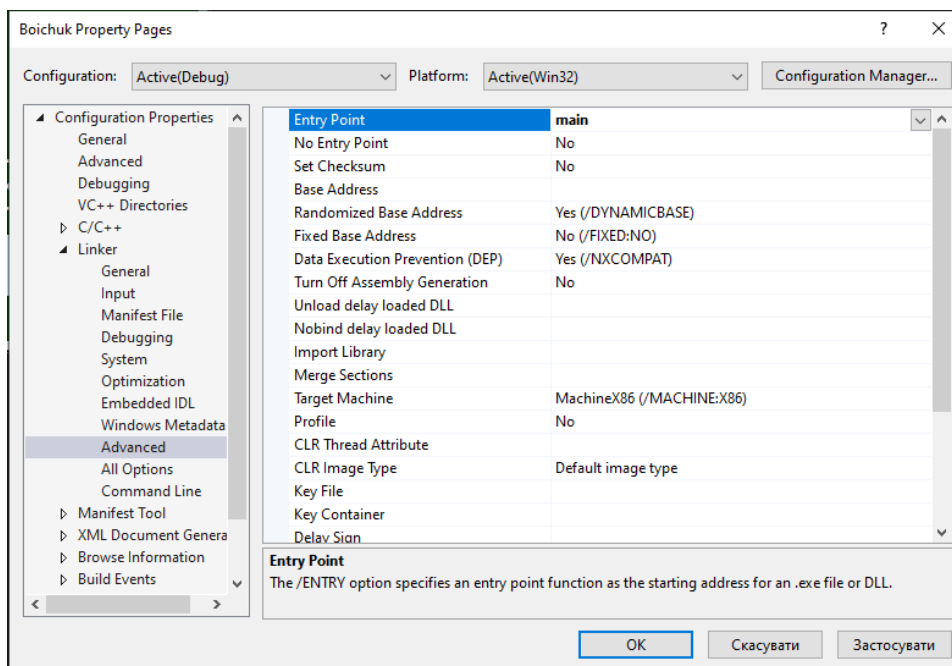
Рисунок 1.10 – Вікно форми в режимі дизайнера

10 Переходимо у вікно властивостей проєкту вибираємо розділ Linker→System→SubSystem – вибираємо із списку **Windows (/SUBSYSTEM:WINDOWS)** (рисуюнок 1.11)



Рисуюнок 1.11 – Вікно властивостей розділ System

11 Далі переходимо в розділ Linker→Advanced→Entry Point – пишемо **main** (рисуюнок 1.12)



Рисуюнок 1.10 – Вікно властивостей розділ Advanced

2.2 Створення нового шаблону додатку

Для того, щоб не повторювати кожного разу даний перелік операцій при створенні нового віконного додатку Windows Forms потрібно створений проект додати до шаблону проектів Visual C++. Для цього виконуємо наступні дії.

1) Обираємо пункт меню Проект \Rightarrow Експорт шаблону... (Project \Rightarrow Export Template...).

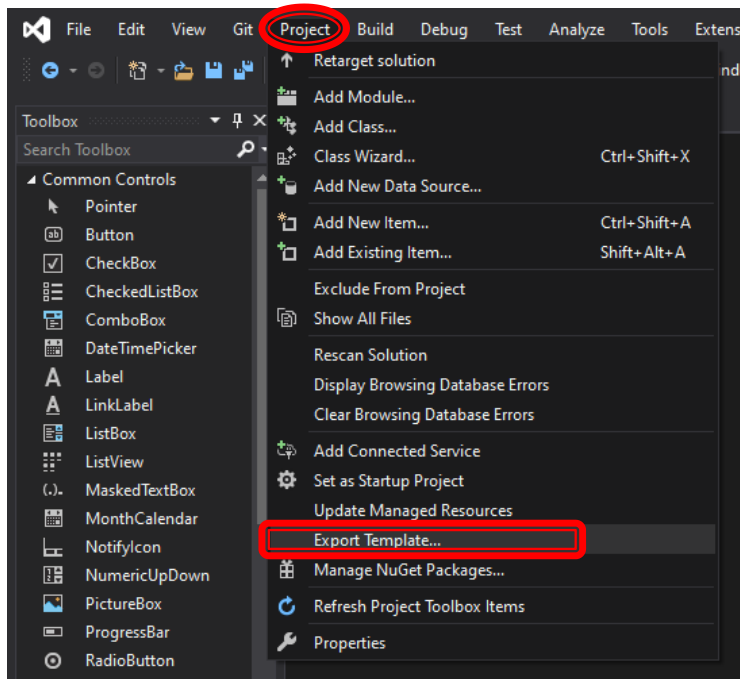


Рисунок 1.13 – Вікно вибору команди Project \Rightarrow Export Template...

2) З'явиться Майстер експорту шаблонів (Export Template Wizard). У вікні майстра Вибір типу шаблону (Choose Template Type) залишаємо все за замовчуванням та натискаємо кнопку Далі (Next) як показано на рисунку 1.14.

3) В другому вікні майстра Вибір параметрів шаблону (Export Template Wizard) додаємо наступне:

- a) у полі Ім'я шаблону: (Template name:) записуємо Windows Forms;
- b) у полі Опис шаблону: (Template description:) записуємо текст пояснення «Створення віконного додатку Windows Forms (CLR)»;
- c) у полі Зображення значка: (Icon Image:) за бажанням для можливості швидкого знаходження шаблону проекту серед переліку шаблонів Visual C++ додаємо графічне зображення;
- d) перевіряємо вибір опції Автоматично імпортувати шаблон у Visual Studio (Automatically import the template into Visual Studio) та натискаємо кнопку Готово (Finish) як зображено на рисунку 1.15.

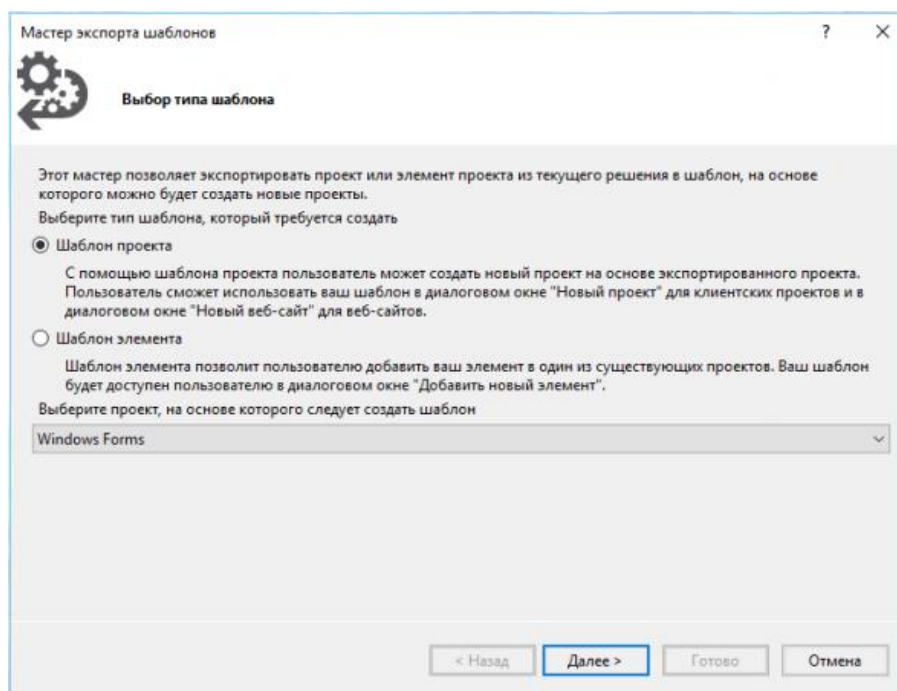


Рисунок 1.14 – Вікно Вибір типу шаблону майстра додавання шаблонів

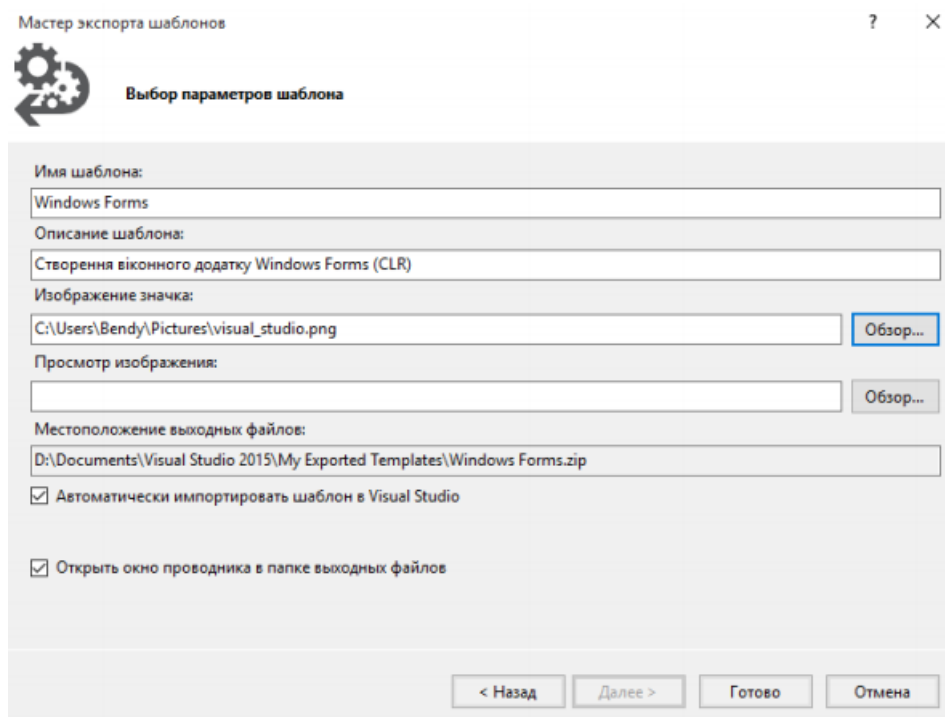


Рисунок 1.15 – Вікно Вибір параметрів шаблону майстра додавання шаблонів

Після додавання шаблону, для створення нового проекту Windows Forms необхідно буде виконати наступний перелік дій.

1) Обрати в головному меню пункт Файл ⇒ Створити ⇒ Проект... (File ⇒ New ⇒ Project...).

2) В діалоговому вікні Створення проекту (New Project) в лівій частині вікна обрати Встановлені ⇒ Шаблони ⇒ Visual C++ (Installed ⇒ Templates ⇒ Visual C++).

3) В діалоговому вікні Створення проекту (New Project) в центральній частині вікна з'явиться пункт створеного нами шаблону Windows Forms, який і потрібно обрати (рисунок 1.16).

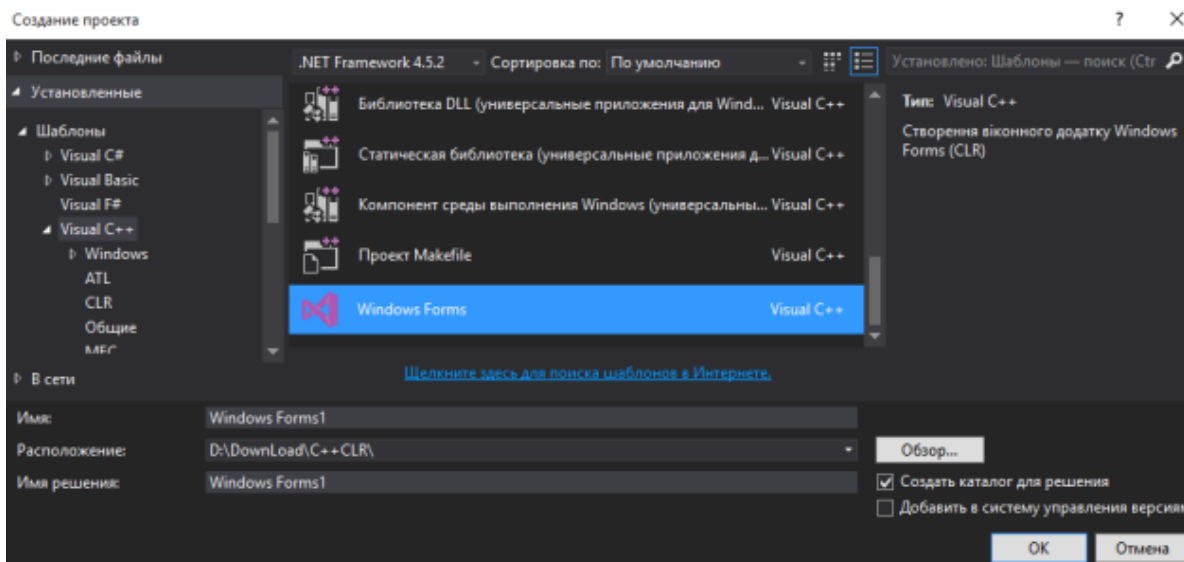


Рисунок 1.16 – Діалогове вікно створення нового проекту з доданням шаблоном Windows Forms

Після цього в IDE відкриється новий проект, який вже міститиме пусту форму (рисунок 1.17).

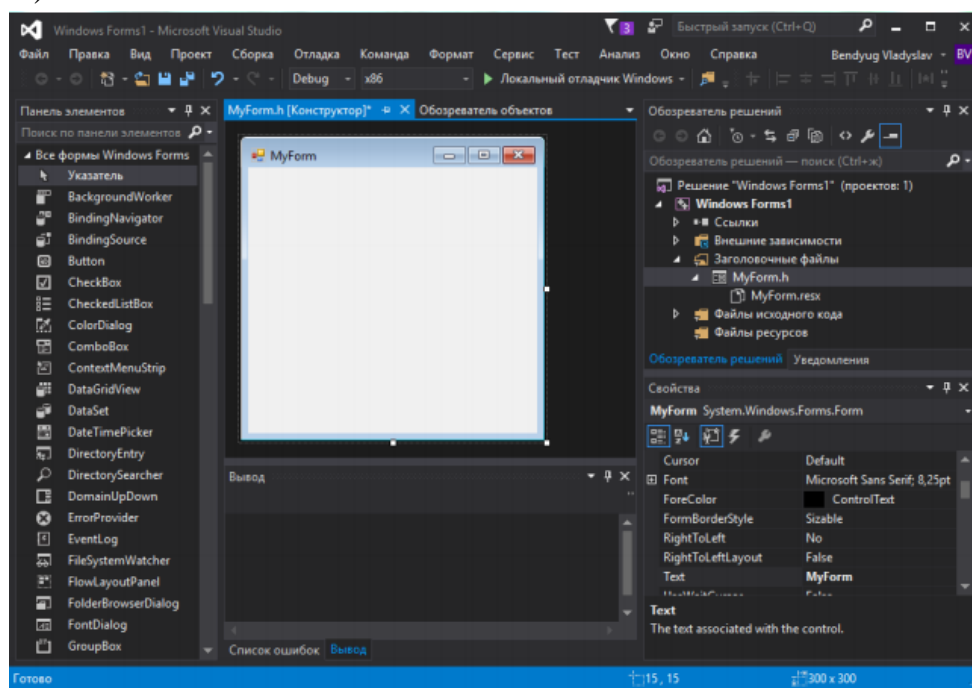


Рисунок 1.17 – Вікно IDE з відкритим проектом Windows Forms

3 Приклад програмної реалізації

3.1 Додаємо до проєкту ще 11 форм(аналогічно до стартової форми, але код писати вже не потрібно). При створенні форм даємо їм назви відповідно до назв завдань для наступної ЛР(рисунок 1.18). Всі файли рішення і проєкту відобразяться на панелі перегляду елементів додатку Solution Explorer в об'єктах Header Files, Resource Files, Source Files.

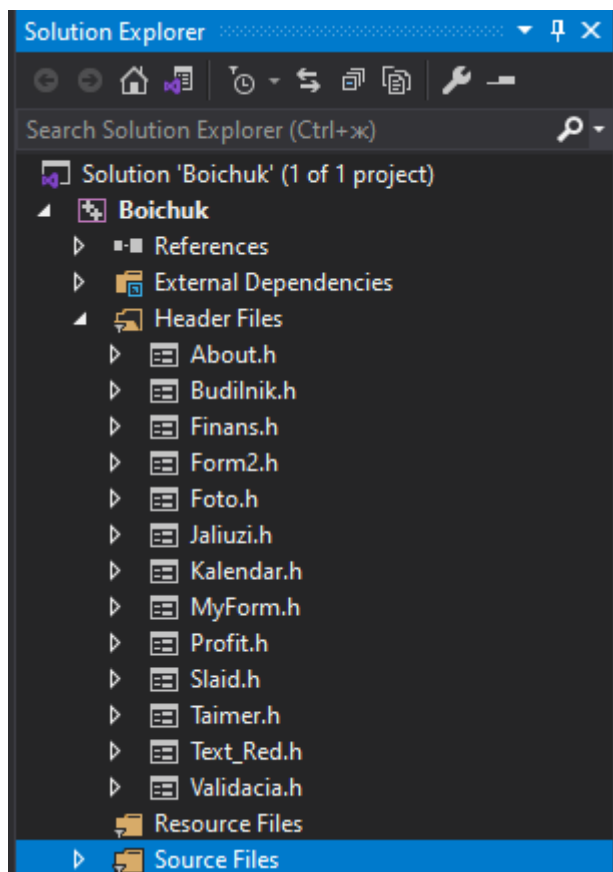


Рисунок 1.18 – Вікно Solution Explorer з доданими Windows Form

3.2 Встановлення властивостей компонента форма MyForm.

- обрати конструктор Windows Forms, що відображає форму MyForm створеного проєкту.
- вибрати за допомогою «миші» компонент **MyForm**. Властивості цього компонента з'являться на панелі **Properties**;
- в групі **Appearance** встановити властивість **Text** «СТАРТОВА ФОРМА»;
- в групі **Layout** встановити властивість **WindowState**, яка б дорівнювала значенню **Maximized**; властивість WindowState визначає розмір вікна: Maximized – максимальний, – Minimized – зменшений до розмірів заголовка, Normal – довільний, визначений у процесі проектування;

- *в групі **VisualStyle** встановити властивість **Icon**, обравши кнопку з крапками. Властивість **Icon** дає можливість задавати піктограму заголовку вікна. Зазвичай, усім вікнам додатка належить мати єдину піктограму. За допомогою діалогової панелі, що з'явилася, треба вибрати файл піктограми (розширення .ico).
- аналогічно задати властивості інших форм.

3.3 Встановити на стартовій формі 10 кнопок для відкривання інших форм і кнопку для завершення роботи програми(рисунок 1.19).

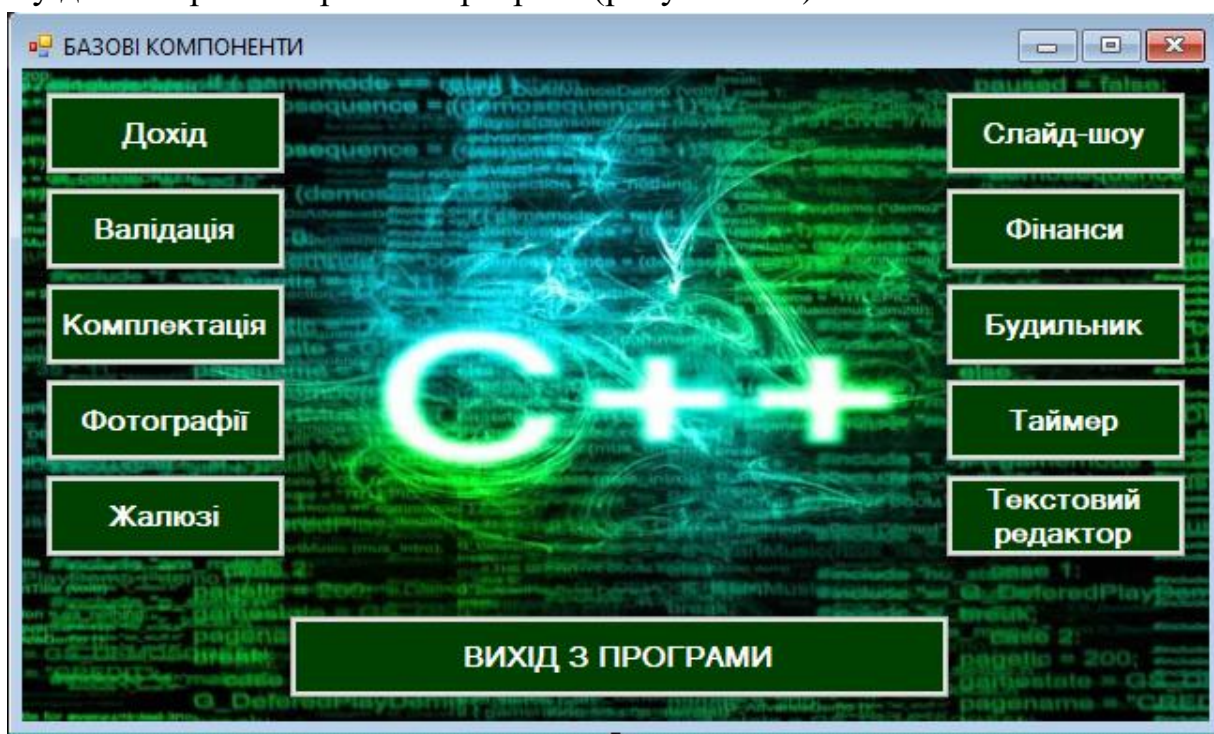


Рисунок 1.19 – Стартова форма

3.4 Для того щоб «Стартова форма» бачила інші форми, підключаємо їх за допомогою директиви `include`.

```
1  #pragma once
2  #include "Profit.h"
3  #include "Validacia.h"
4  #include "Kalendar.h"
5  #include "Foto.h"
6  #include "Finans.h"
7  #include "Budilnik.h"
8  #include "Jaliuzi.h"
9  #include "Taimer.h"
10 #include "Text_Red.h"
11 #include "Slaid.h"
```

3.5 Щоб створити метод Click для кнопок, потрібно клікнути клавiшею миші по обраній кнопці, у редакторі коду буде створено шаблон методу.

```
private: System::Void button1_Click(System::Object^ sender, System::EventArgs^ e) {
}

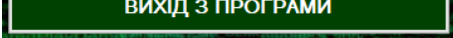
```

Код методу кнопки для відкриття форми «Дохід»(наприклад) наступний:

```
private: System::Void button1_Click(System::Object^ sender, System::EventArgs^ e) {
    Profit^ form = gcnew Profit();
    form->ShowDialog();
}

```

Аналогічно створюємо методи для інших кнопок.

Метод кнопки  наступний:

```
private: System::Void button11_Click(System::Object^ sender, System::EventArgs^ e) {
    Close();
}

```

4 Компіляція і виконання додатку.

1) вибрати пункт меню **Debug, Start Debugging**; ця операція компілює, редагує зв'язки та виконує додаток;

2) додаток виконано – відкрилося основне вікно додатка з заголовком «СТАРТОВА ФОРМА» вказаних розмірів і з обраною піктограмою в заголовку;

3) перевірити відкриття інших форм;

4) для завершення роботи додатка слід використати кнопку закриття вікна.

5 Завершення роботи над рішенням і проектом. Обрати пункт меню **File, Close Solution**.

Завдання до самостійної роботи

- 1 Створити найпростіший додаток.
- 2 Встановити довільні розміри форми(ширину і висоту).
- 3 *Встановити позиціювання форми при запуску.
- 4 Встановити кольори фону форм.
- 5 *Встановити фоновий рисунок на формі.
- 6 **Оформити стиль форм.

Контрольні запитання

1. У чому переваги об'єктно-орієнтованого візуального середовища розробки?
2. Назвіть елементи інтегрованого середовища розробки.
3. Що таке проект об'єктно-орієнтованого візуального середовища розробки?
4. Що таке рішення?
5. Які файли створюються в процесі створення рішення, проекту?
6. Що визначають властивості об'єкту?
7. Що таке складені властивості?
8. Що таке додаток Windows Forms?
9. Як створити новий проект Windows Forms?
10. Як додати нову форму у проект Windows Forms?
11. Як створити шаблон проекту?
12. Що таке Конструктор форми, як додавати нові елементи інтерфейсу у форму?
13. Для чого потрібне вікно Редоктора?
14. Для чого потрібен Оглядач рішень?
15. Як змінити властивості об'єкту?
16. Як додати реакцію на подію для об'єкту?