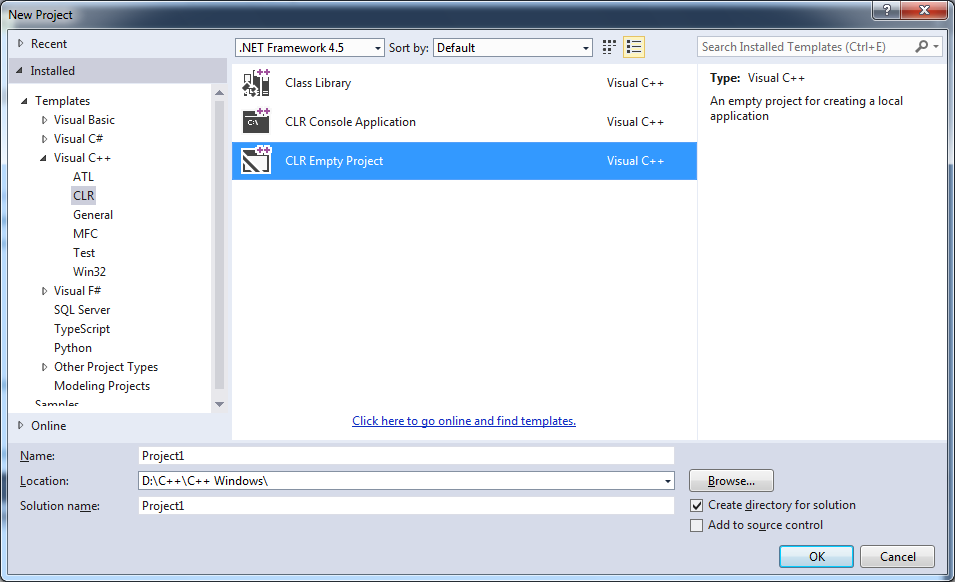
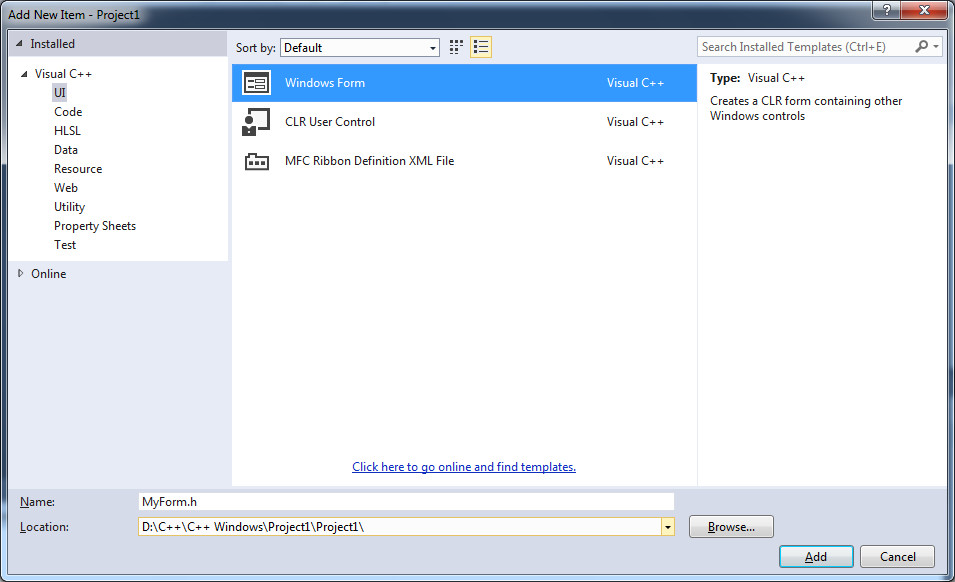
**Лабораторная работа 7**. Наследование и полиморфизм в ООП на примере языка программирования С++

Создание прилодения Windows Form

1. Создать пустой проект CLR.

2. К созданному проекту добавляем новый элемент – форму: Add → New Item. В открывшемся меню в разделе UI выбираем Windows Forms. Имя формы можно изменить.

3. В файле формы MyForm.cpp находится единственной строчкой кода:

#include "MyForm.h"

В данный файл необходимо добавить следующий код:

using namespace System;

using namespace System::Windows::Forms;

[STAThreadAttribute]

void Main(array<String^>^ args) {

Application::EnableVisualStyles();//позволяет задать современный внешний вид для некоторых элементов управления

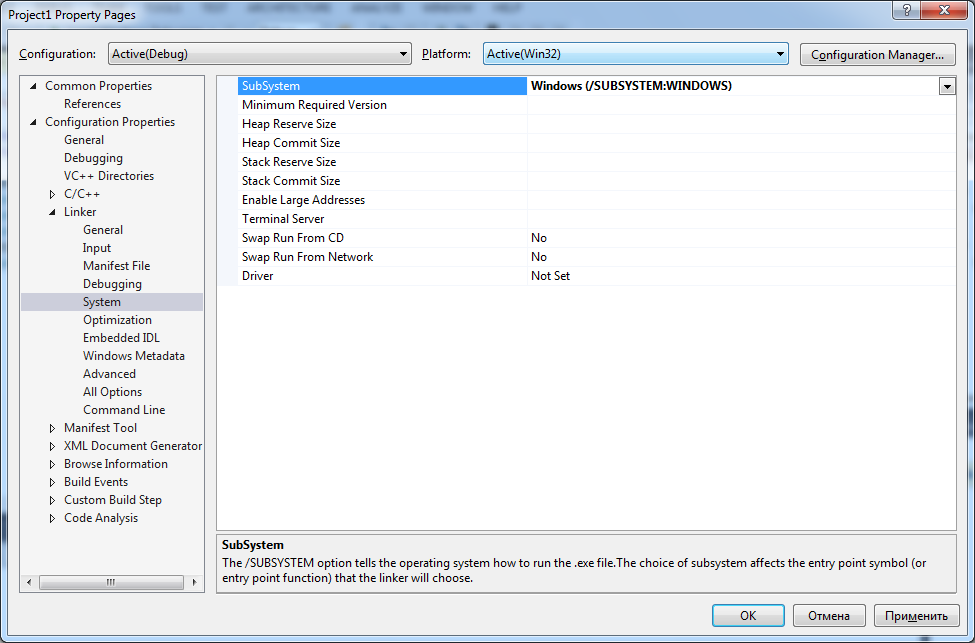
Application::SetCompatibleTextRenderingDefault(false);

Project1::MyForm form;

Application::Run(%form);

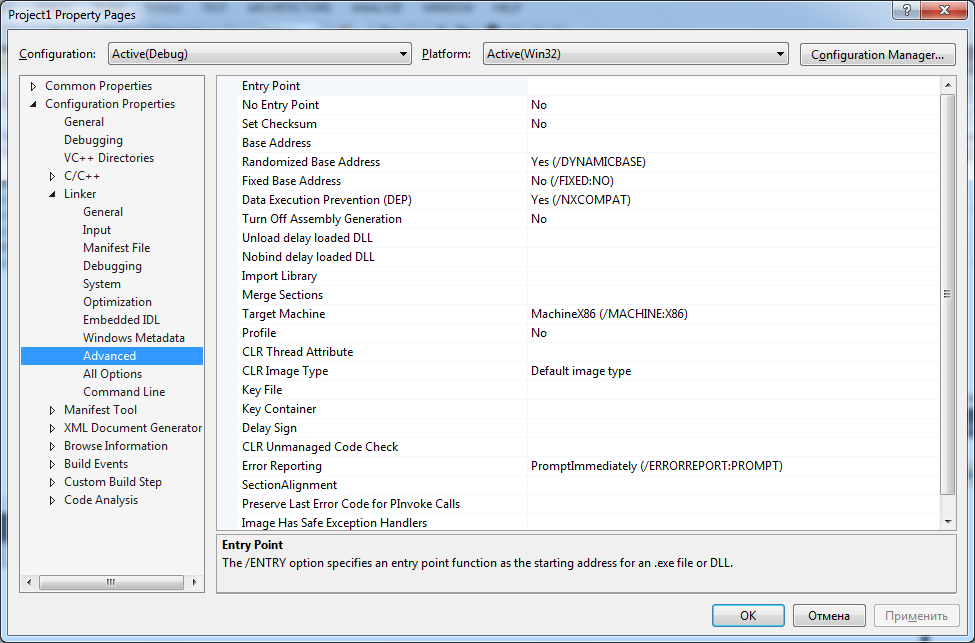
}

4. В **свойствах проекта** в разделе *Linker* (*Компоновщик*) выбираем подраздел *System* (*Система*). В строке *SubSystem* (*Подсистема*) из выпадающего меню выбираем *Windows (/SUBSYSTEM:WINDOWS)*. Нажимаем *Применить.*



В подразделе *Advanced* (*Дополнительно*) в строке *Entry Point* (*Точка входа*) пишем *Main*. Нажимаем **ОК.**

Main – имя функции void Main(array<String^>^ args) в фале MyForm.cpp.



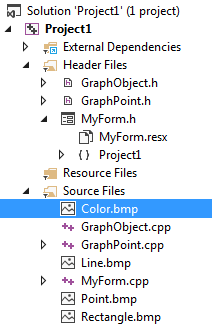
5. Создание приложения для рисования

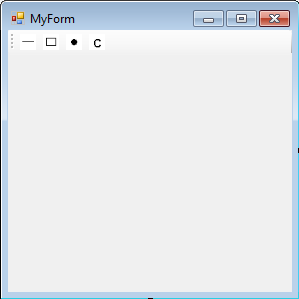
**Дизайн формы**

Положите на форму компонент toolStrip1 и добавьте кнопки для выбора фигур рисования и изменения цвета.

Добавьте фалы рисунков основных элементов: Add -> New Item…

Среди файлов раздела Resource выберите Bitmap File (.bmp).

Проследите за тем, чтобы размер файла совпал с размером кнопки на панели toolStrip1.



**Описание кода**

1) Создайте абстрактный класс GraphObject.

Описание файла GraphObject.h

#pragma once

using namespace System::Drawing; //подключения пространства имен для рисования

ref class GraphObject

{

protected:

Color color; //поле цвет

Graphics^ field; //поле для рисования

public:

GraphObject(){color = Color::Black;} //конструктор

GraphObject(Graphics^ \_field) { field = \_field; color = Color::Black; } //конструктор

Color GetColor(){ return color; } //взятие звета объекта

void SetColor(Color c){ color=c; } //задание цвета для рисования объекта

virtual void Drow(Graphics^) = 0;//чистоабстрактный метод для рисования - должен быть переопределен в потомке

};

2) Создайте класс-наследник для рисования точки GraphPoint

Описание файла GraphPoint.h

#pragma once

#include "GraphObject.h"

ref class GraphPoint :

public GraphObject

{

private:

int X, Y;//координаты точки

public:

GraphPoint(Graphics^ \_field) :GraphObject(\_field){ X = 0; Y = 0; }//конструктор

GraphPoint(Graphics^ \_field, int, int);//конструктор

void Drow(Graphics^ ) override; //перегруженный метод рисования

};

Описание файла GraphPoint.cpp

#include "GraphPoint.h"

GraphPoint::GraphPoint(Graphics^ \_field, int \_X, int \_Y):GraphObject(\_field)

{

X = \_X;

Y = \_Y;

}

void GraphPoint::Drow(Graphics^ \_field)

{

SolidBrush^ b = gcnew SolidBrush(color);

\_field->FillEllipse(b, X, Y, 5, 5);

}

3) Дополнение класса формы MyForm

Добавьте заголовочный файл "GraphPoint.h" в файл MyForm.h.

До описания класса формы добавьте описание перечисления

enum TypeFigure { point, line, rectangle };

В классе формы объявите три переменные:

public:

Graphics^ G;// для рисования на форме

Color color;//цвет рисования

TypeFigure Figure;//тип фигуры

В конструкторе формы после инициализации компонент инициализируйте объявленные переменные:

G = Graphics::FromHwnd(this->Handle);

color = Color::Black;

Figure = point;

В обработчиках события Click кнопок toolStripButton выбора вида фигуры необходимо выбрать одно из следующих присваиваний:

Figure = point;

Figure = rectangle;

Figure = line;

В обработчиках события Click кнопки toolStripButton выбора цвета:

if (colorDialog1->ShowDialog() == System::Windows::Forms::DialogResult::OK)

{

color = colorDialog1->Color;

}

В обработчике события MouseDown для формы необходимо создать объект, выбранный для рисования. Например, для точки:

GraphPoint A(G, e->X, e->Y);

A.SetColor(color);

A.Drow(G);

Задания для самостоятельного выполнения:

1. Опишите классы построения линии, закрашенного прямоугольника, закрашенного эллипса, полого прямоугольника, полого эллипса.

2. Добавьте кнопку выбора толщины линии и соответствующее поле в класс GraphObject.

3. В обработчике события MouseDown реализуйте рисование выбранной линии. Например, с помощью оператора выбора:

switch (Figure)

{

case point:

GraphPoint A(G, e->X, e->Y);

A.SetColor(color);

A.Drow(G);

break;

case Project1::MyForm::line:

break;

case Project1::MyForm::rectangle:

break;

default:

break;

}

4. Добавьте кнопки очистки поля.

5. Добавьте резинку.

6. Добавьте кнопку для сохранения (открытия) изображения.