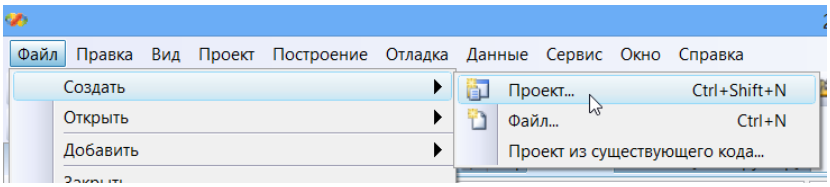
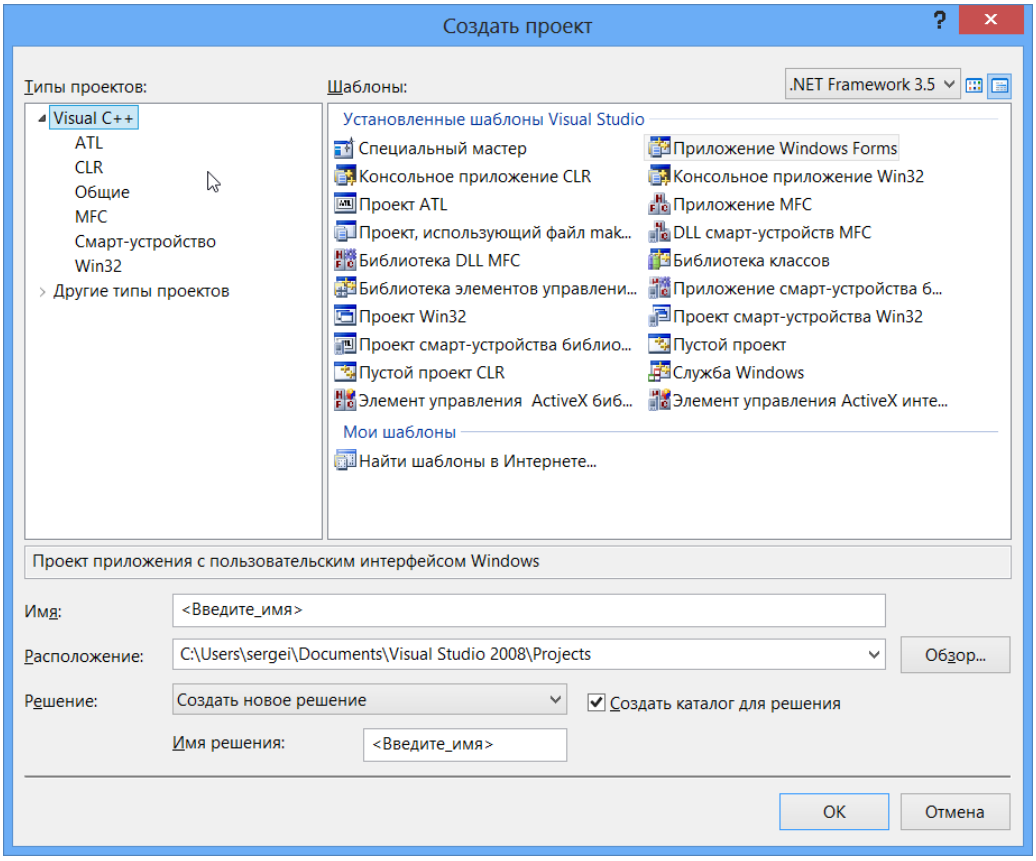
1 Простое рисование на форме

Для того, чтобы создать новый проект, выберите пункт меню *Файл→Создать→Проект*.

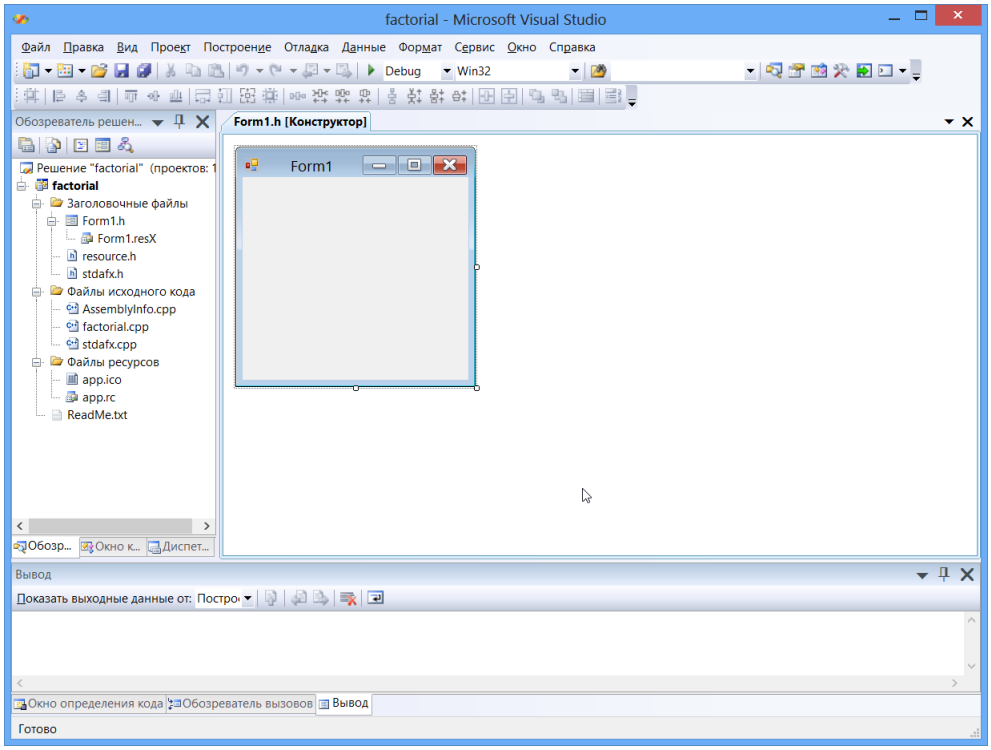


В появившемся окне

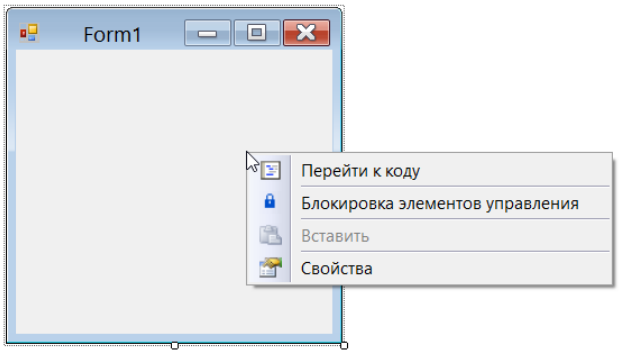


выберите тип проекта «Visual C++», среди установленных шаблонов Visual Studio выберите шаблон проекта «*Приложение Windows Forms*», после чего, в нижней части того же окна задайте имя создаваемого проекта. Обратите внимание на месторасположение будущего проекта (поле «Расположение») и, если необходимо, поменяйте его.

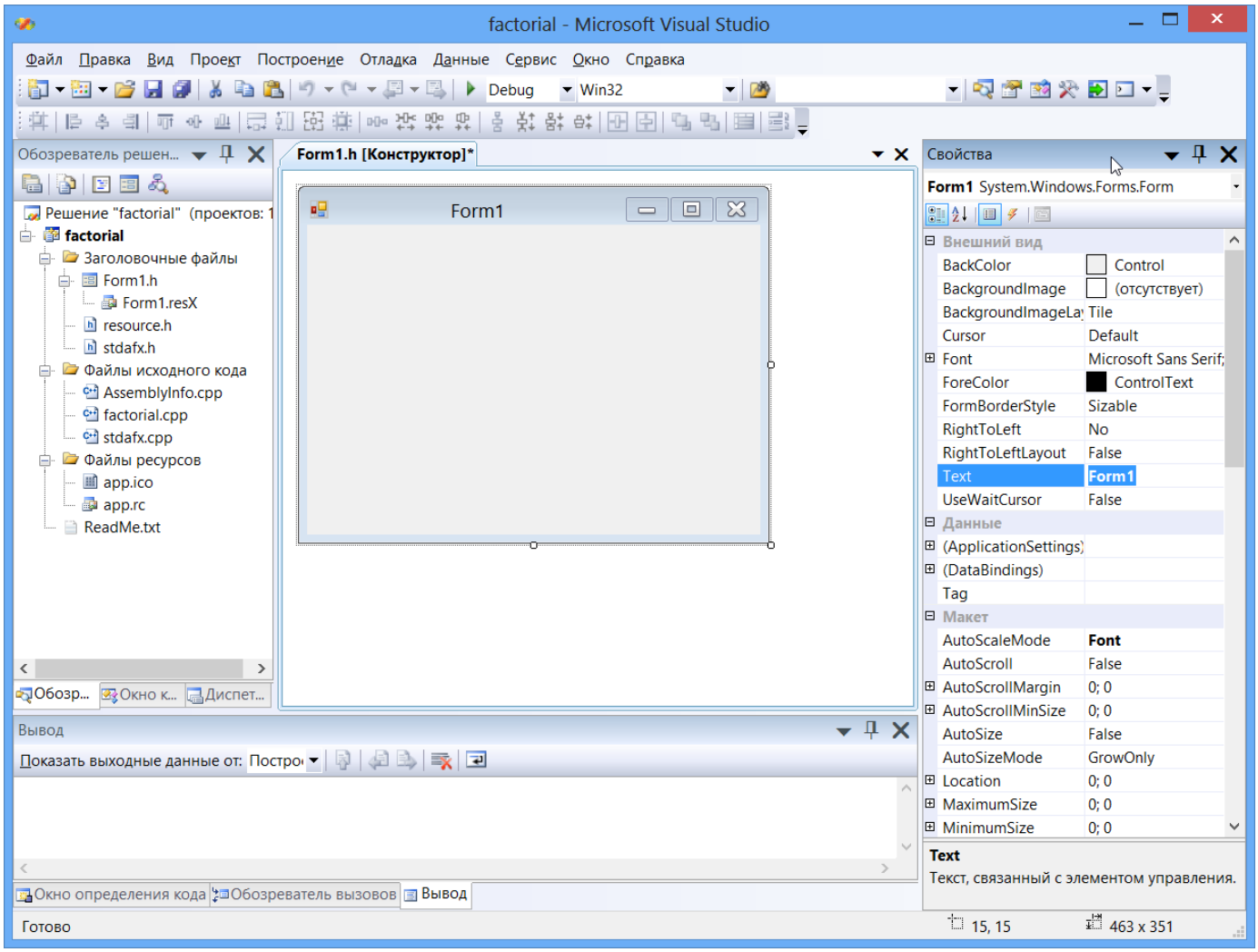
Visual Studio создаст проект с пустой формой. Окно проекта будет выглядеть примерно так:



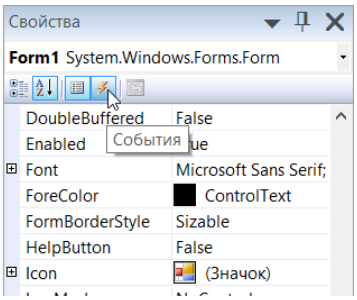
Нажмите правую кнопку мыши на форме и в появившемся меню выберите пункт *Свойства*.



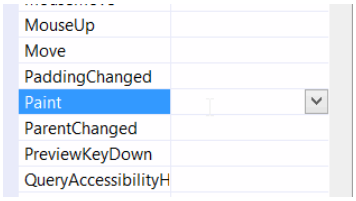
Появится окно диспетчера свойств, в котором отражены все параметры созданной формы.



Перейдите на вкладку «События» диспетчера свойств.



Среди перечисленных событий найдите событие *Paint* (Рисование формы).



Для того, чтобы добавить обработчик данного события дважды кликните мышью на пустом поле правее имени события *Paint*. В файле, описывающем форму (по умолчанию `Form1.h`) будет создана процедура — обработчик события отрисовки формы и вы будете перенаправлены для редактирования этой процедуры.

```
private: System::Void Form1_Paint(System::Object^  sender, System::Windows::Forms::PaintEventArgs^  e) {
         }
```

Выбрав соответствующий элемент из списка можно разместить (нарисовать) его на форме. Сразу нажмем на клавишу Enter (чтобы вставить пустую строку в теле процедуры) и приступим к изменению данной процедуры.

Воспользуемся аргументом `e` процедуры `Form1_Paint` (в дальнейшем здесь будем указывать имена, используемые по умолчанию) типа `System::Windows::Forms::PaintEventArgs^` для извлечения графической области формы (области для закрашивания). Для этого опишем переменную `g` типа `System::Drawing::Graphics^`. Т. к. пространство имен `System::Drawing` уже подключено (в начале файла `Form1.h`), достаточно описать:

```
Graphics^ g
```

Присвоим этой переменной значение `e->Graphics`. Теперь переменная `g` ссылается на область рисования в нашей форме.

Закрасим (очистим) область `g` цветом «Аквамарин». Для этого дадим команду:

```
g->Clear(Color::Aquamarine);
```

Для извлечения текущего размера формы опишем переменную `rect` прямоугольник, заполняющий область рисования.

```
Rectangle rect = Form::ClientRectangle;
```

Опишем переменную `redPen` — перо для рисования (вычерчивания) красного цвета.

```
Pen^ redPen = gcnew Pen(Color::Red);
```

Установим толщину красного пера — 4:

```
redPen->Width = 4;
```

Начертим красным пером линию от левого верхнего угла (координаты `(0,0)`) до правого нижнего угла (координаты — ширина и высота прямоугольника `rect`) формы.

```
g->DrawLine( redPen, 0, 0, rect.Width, rect.Height );
```

Опишем переменную `bluePen` — синее перо, толшиной 10.

```
Pen^ bluePen = gcnew Pen(Color::Blue);
bluePen->Width = 10;
```

Начертим синим пером линию от координат `(50,70)` до точки, касающейся правой стороны окна, на высоте `80`.

```
g->DrawLine( bluePen, 50, 70, rect.Width, 80);
```

Опишем переменную `drawFont` — экземпляр шрифта Arial размером 10.

```
System::Drawing::Font^ drawFont = gcnew System::Drawing::Font( "Arial",10 );
```

Опишем переменную `drawBrush` — экземпляр кисти для письма на форме черного цвета.

```
SolidBrush^ drawBrush = gcnew SolidBrush(Color::Black);
```

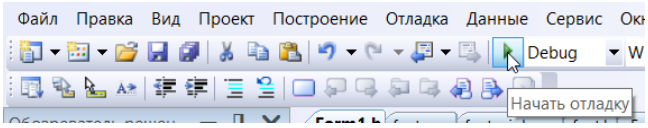
Выведем надпись «Надпись на форме» по координатам `(40,100)` созданной ранее кистью черного цвета и шрифтом Arial размером 10.

```
g->DrawString("Надпись на форме", drawFont,drawBrush, 40, 100);
```

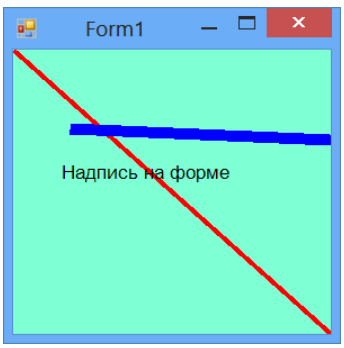
На этом завершим редактирование процедуры. После редактирования процедура будет выглядеть так:

```
private: System::Void Form1_Paint(System::Object^  sender, System::Windows::Forms::PaintEventArgs^  e) {
             Graphics^ g = e->Graphics;
             g->Clear(Color::Aquamarine);
             Rectangle rect = Form::ClientRectangle;
             Rectangle smallRect;

             Pen^ redPen = gcnew Pen(Color::Red);
             redPen->Width = 4;
             g->DrawLine( redPen, 0, 0, rect.Width, rect.Height );

             Pen^ bluePen = gcnew Pen(Color::Blue);
             bluePen->Width = 10;
             g->DrawLine( bluePen, 50, 70, rect.Width, 80);

             System::Drawing::Font^ drawFont = gcnew System::Drawing::Font( "Arial",10 );
             SolidBrush^ drawBrush = gcnew SolidBrush(Color::Black);
             g->DrawString("Надпись на форме", drawFont,drawBrush, 40, 100);
         }
```

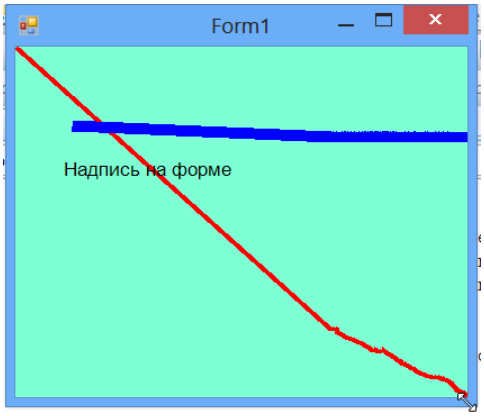
Полученное приложение можно скомпилировать. Для этого можно на панели инструментов нажать соответствующую кнопку.



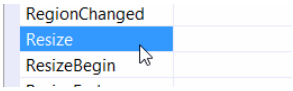
После компиляции приложение будет запущено:



Все отрисовывается именно так, как мы задумали. Но при изменении размера формы рисунок не обновляется, а на форме появляется разный мусор.



Для того, чтобы это исправить, сначала закроем запущенное приложение, после чего в окне свойств формы найдем событие *Resize*.



Дважды кликните мышкой на пустом поле справа от наименования *Resize*. Будет создана процедура — обработчик события, срабатывающая при изменении размера формы.

```
private: System::Void Form1_Resize(System::Object^  sender, System::EventArgs^  e) {
         }
};
```

В теле этой процедуры дадим всего лишь одну команду:

```
this->Refresh();
```

После редактирования процедура будет выглядеть так:

```
private: System::Void Form1_Resize(System::Object^  sender, System::EventArgs^  e) {
             this->Refresh();
         }
```

Теперь после перекомпиляции и запуска изменение размера формы приведет к полному обновлению рисунка на форме.