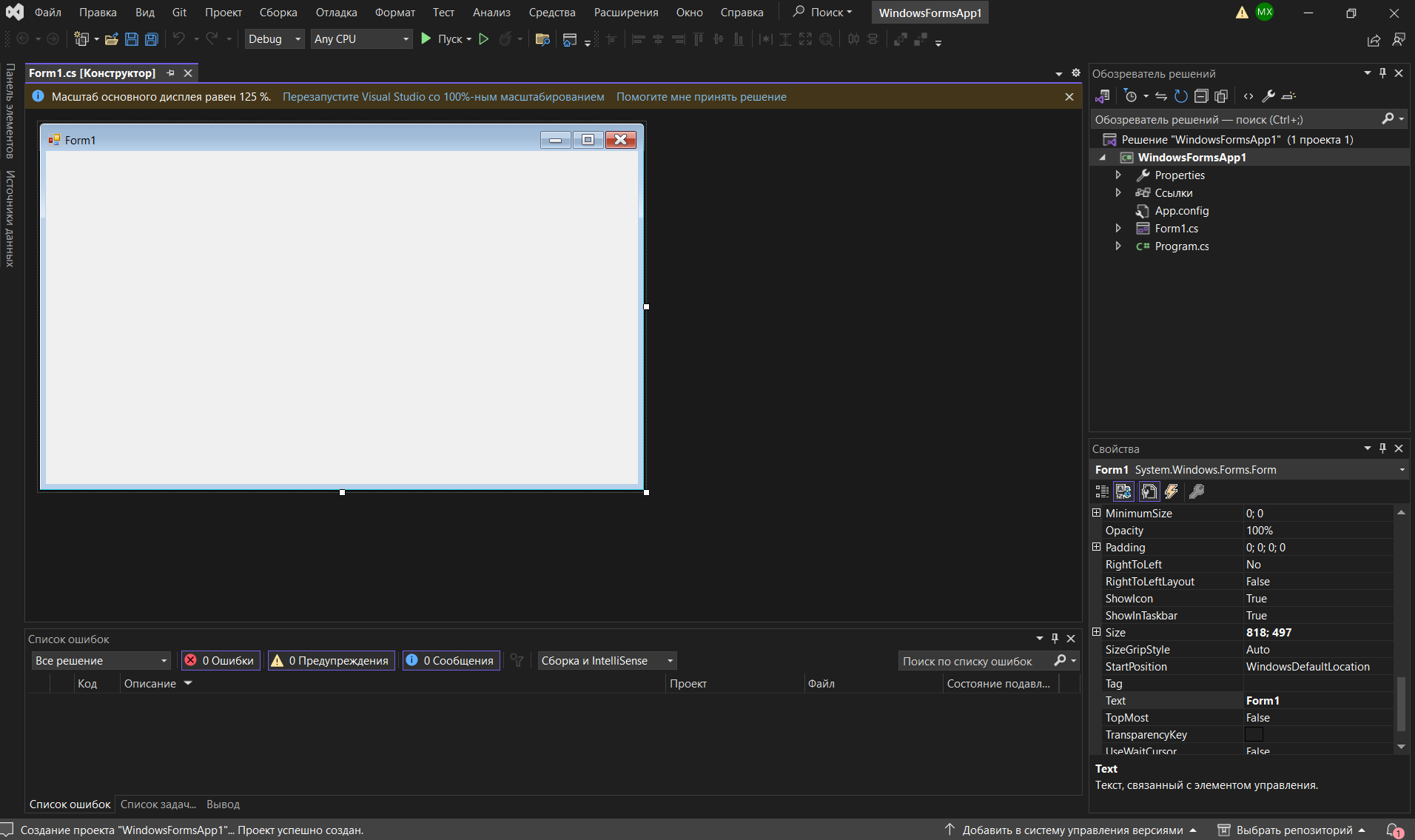
**Лабораторная работа 1. Введение в разработку форм**

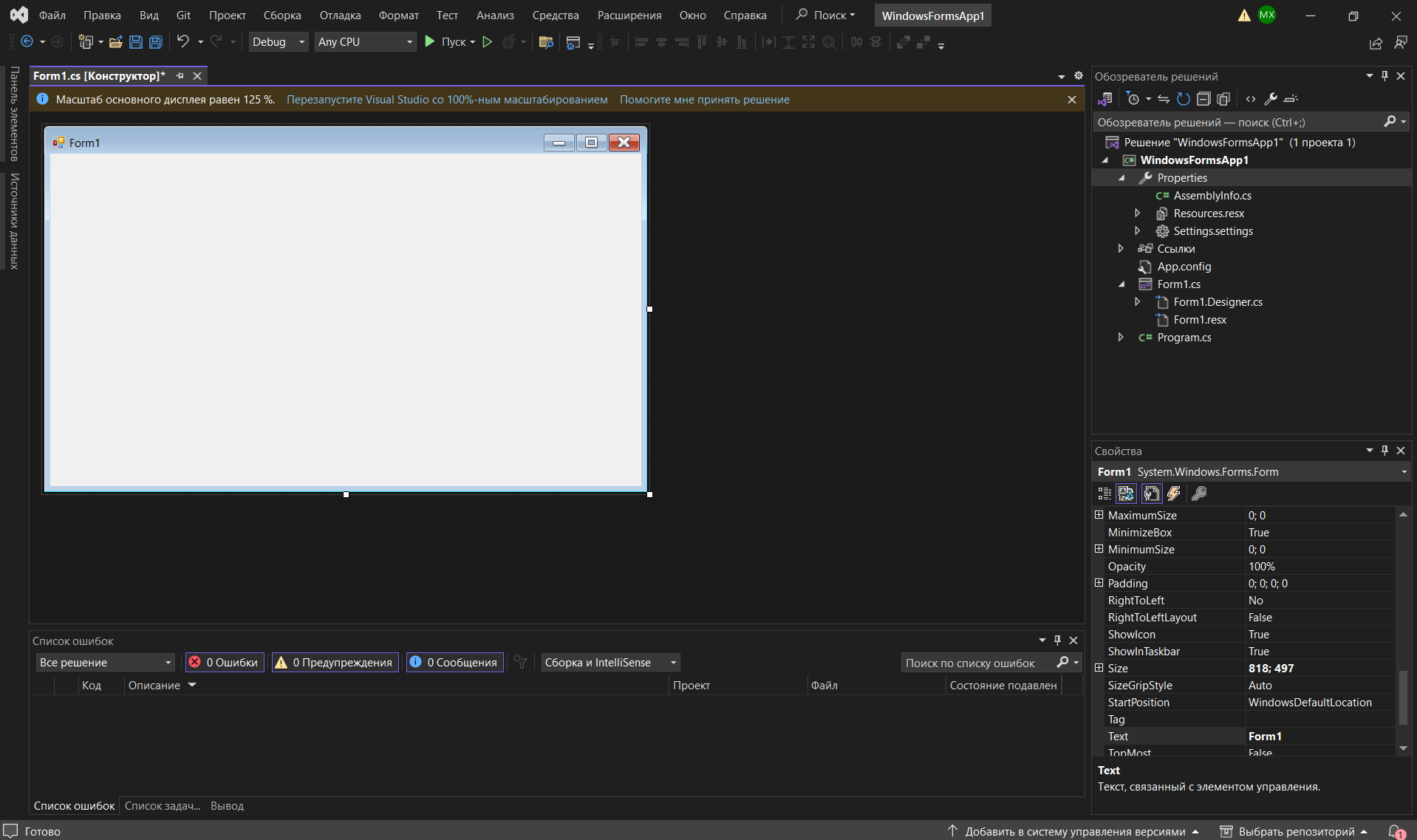
**Цель работы:** Изучение методов построения форм Windows и получение навыков по настройке форм, созданию непрямоугольных и наследуемых (производных) форм.

**Упражнение 1. Настройка прямоугольной формы Windows**

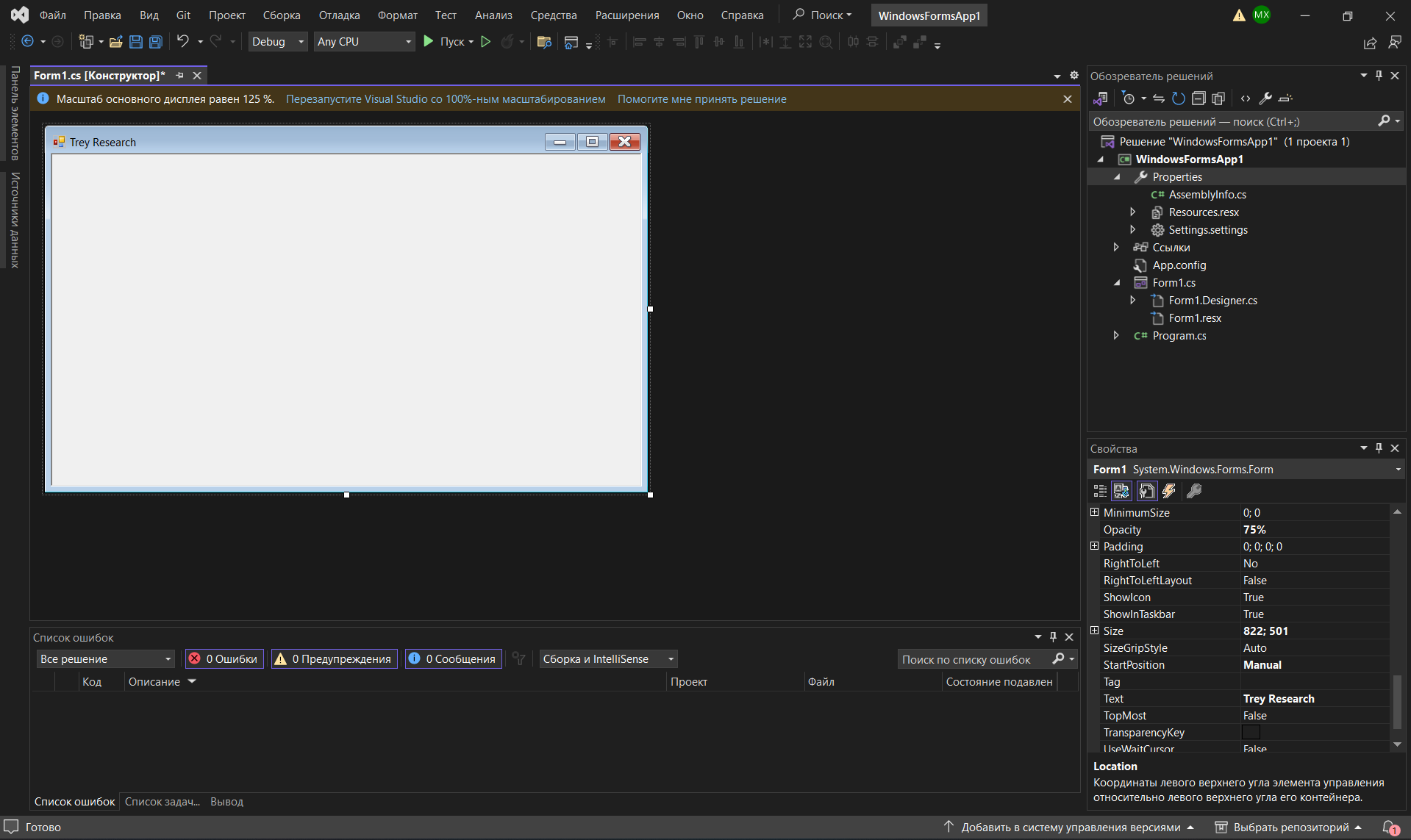
1. **Создание нового проекта**



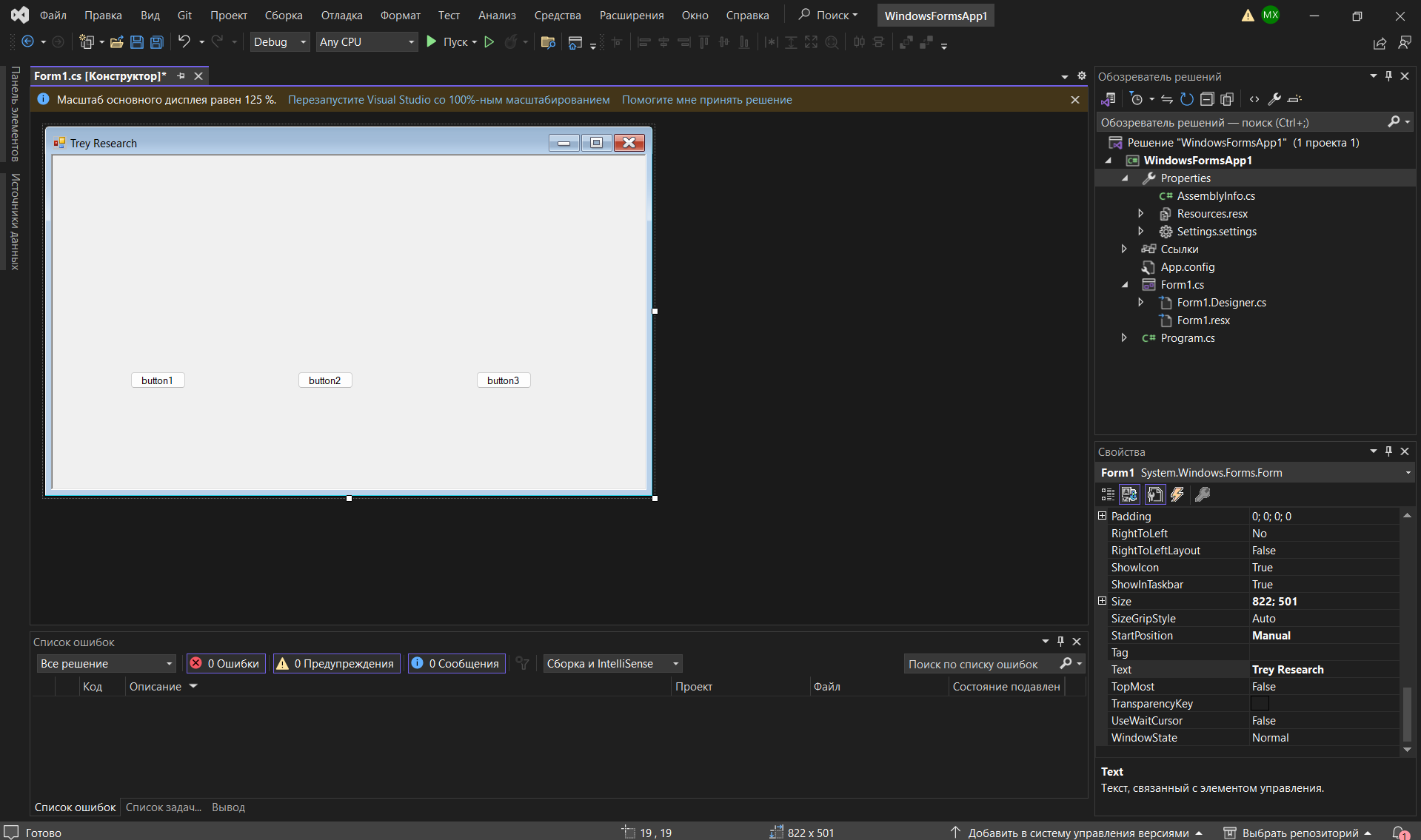
Откройте Visual Studio и создайте новый проект Windows Forms. Проект откроется с формой по умолчанию с именем **Form1** в конструкторе.



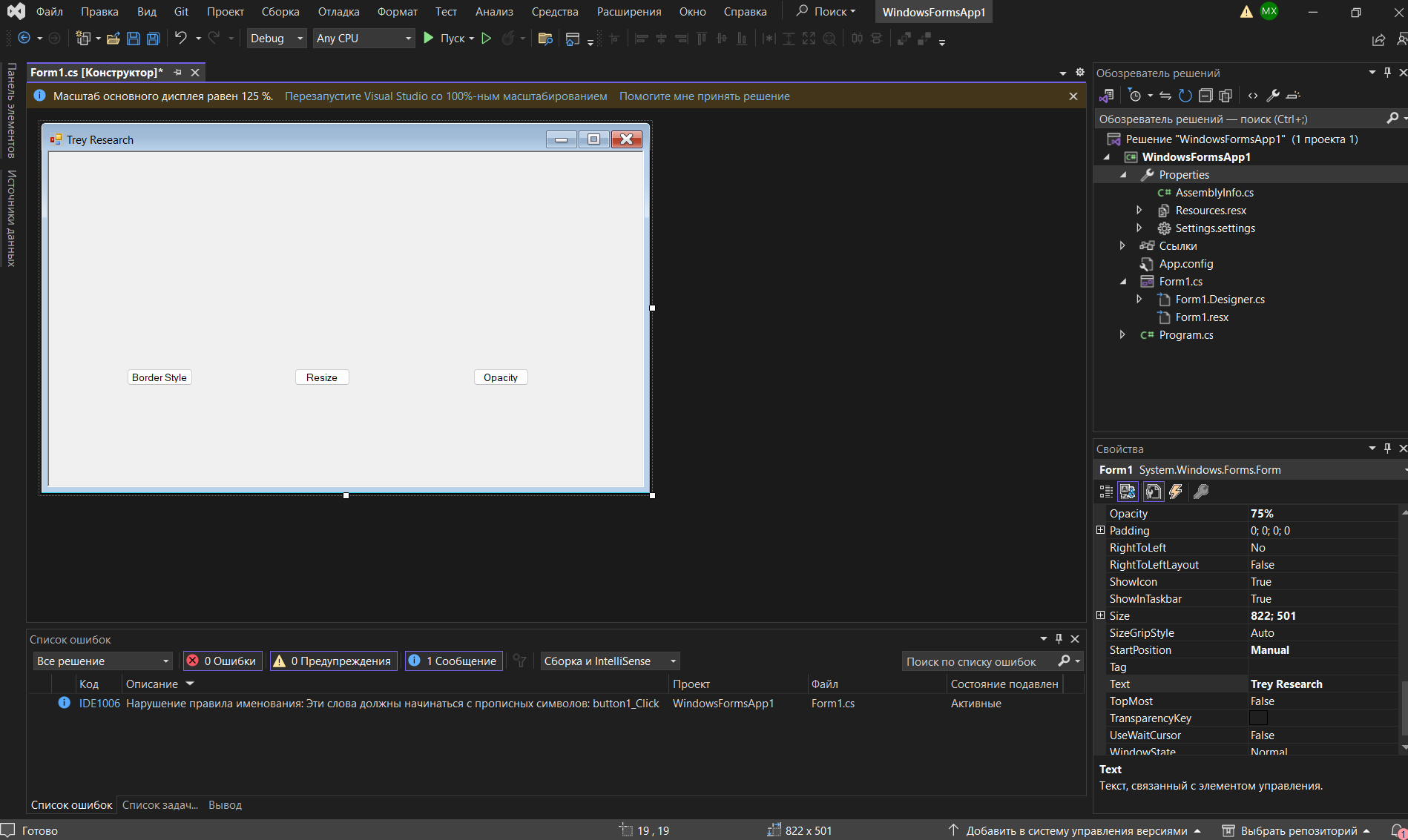
Выберите форму в конструкторе. Свойства формы отображаются в окне **Properties**.



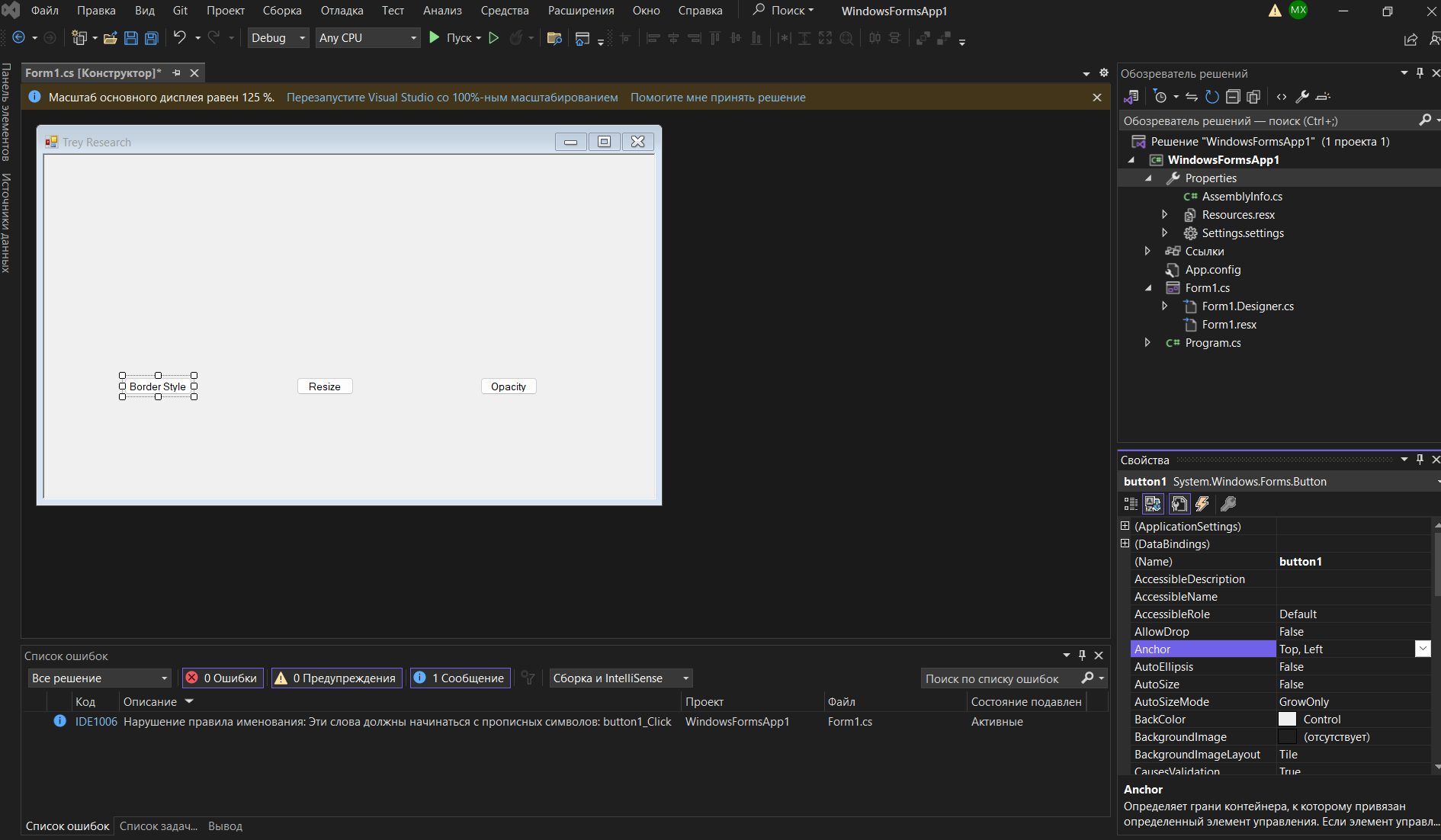
В окне **Properties** задайте свойства значения, как указано ниже.



Перетащите три кнопки из **Toolbox** (Панель элементов) в форму и разместите их так, как вам будет удобно.

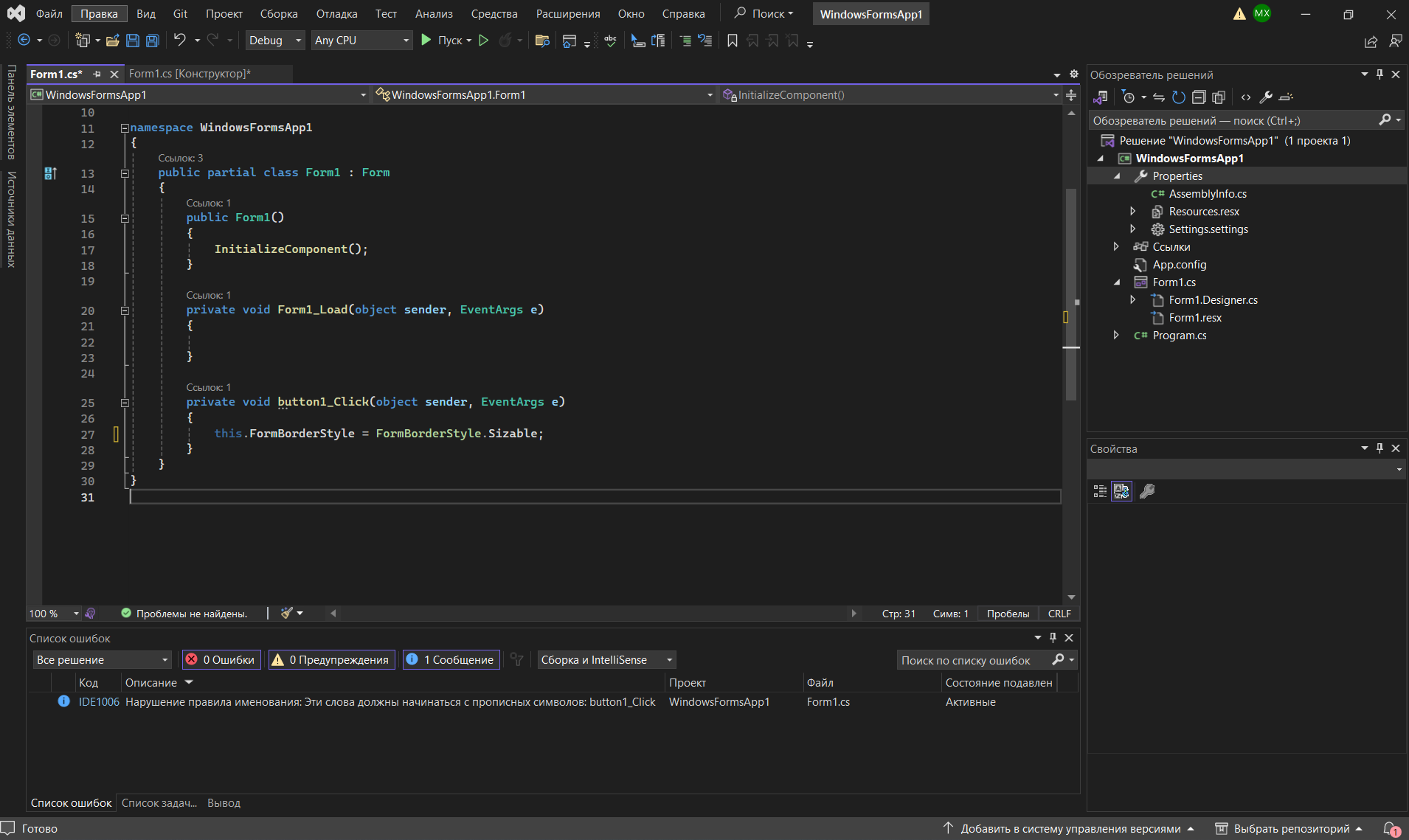


Поочередно выберите каждую кнопку, и в окне **Properties** задайте свойству кнопок **Text** значения **Border Style, Resize** и **Opacity**.

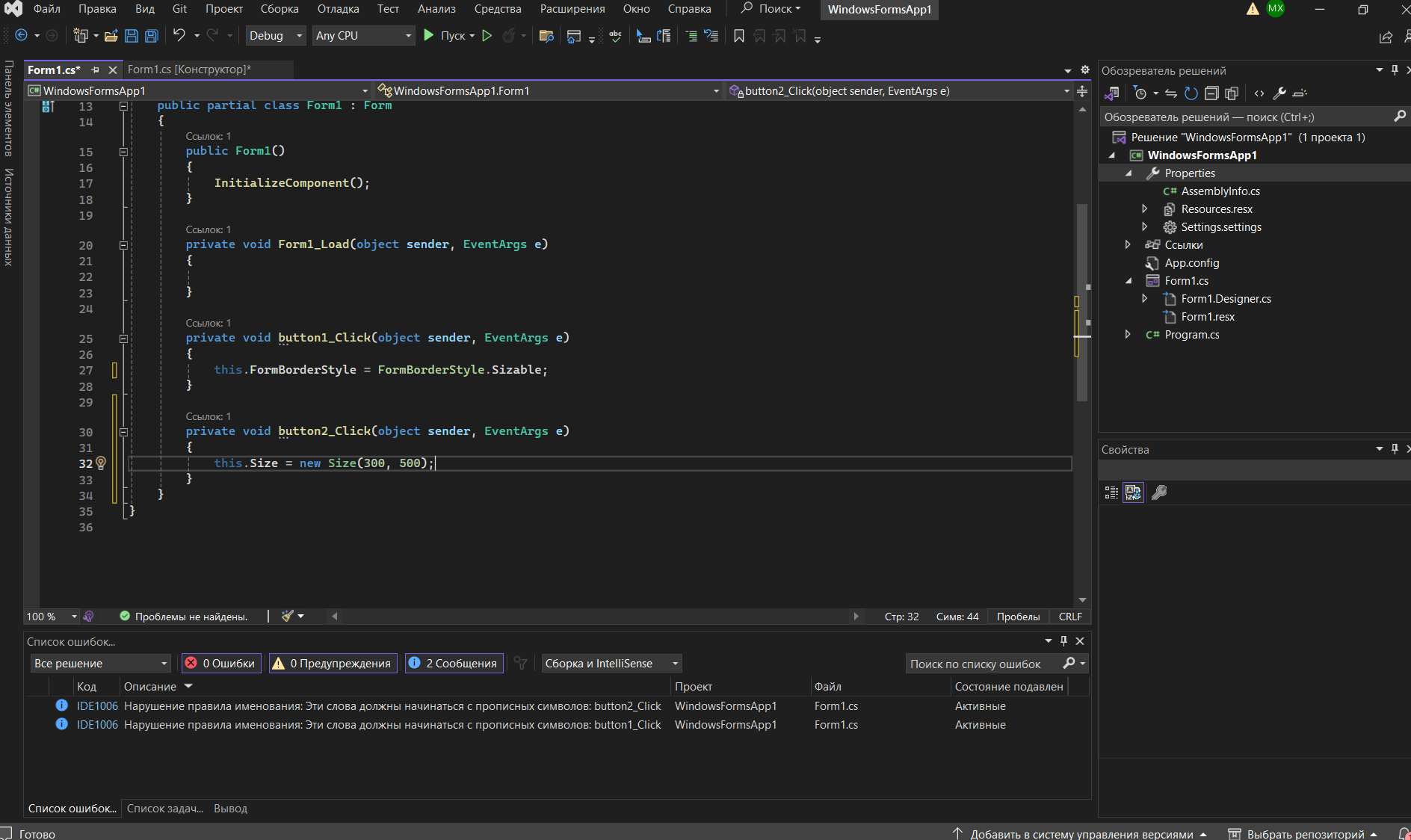


Для кнопки **Border Style** задайте свойство **Anchor – Top, Left**.

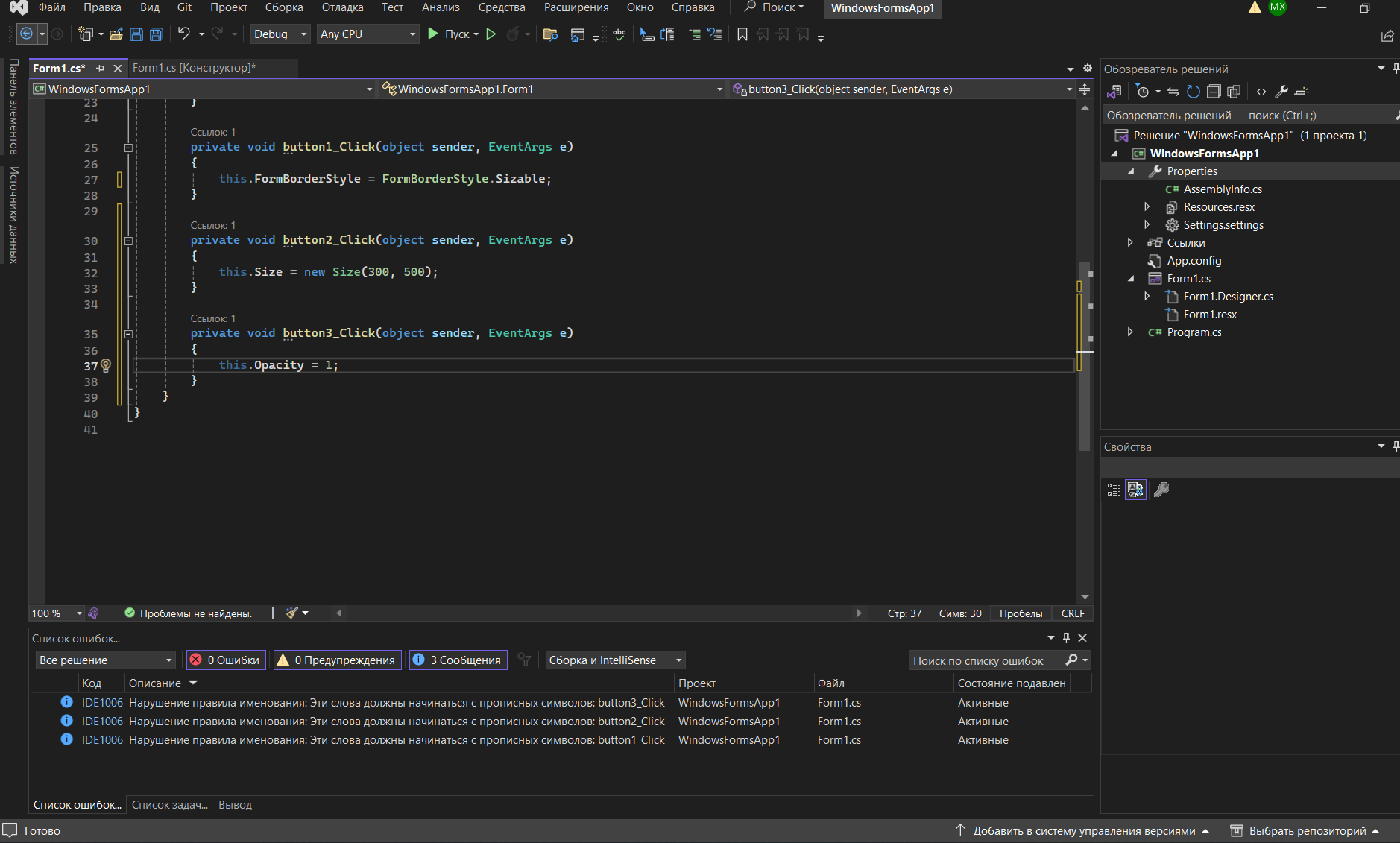
1. **Реализация обработчиков событий**



В конструкторе дважды щелкните кнопку **Border Style**,чтобы открыть окно с кодом обработчика события **Button1 Click**. Добавьте в этот метод следующую строку кода: this.FormBorderStyle = FormBorderStyle.Sizable;

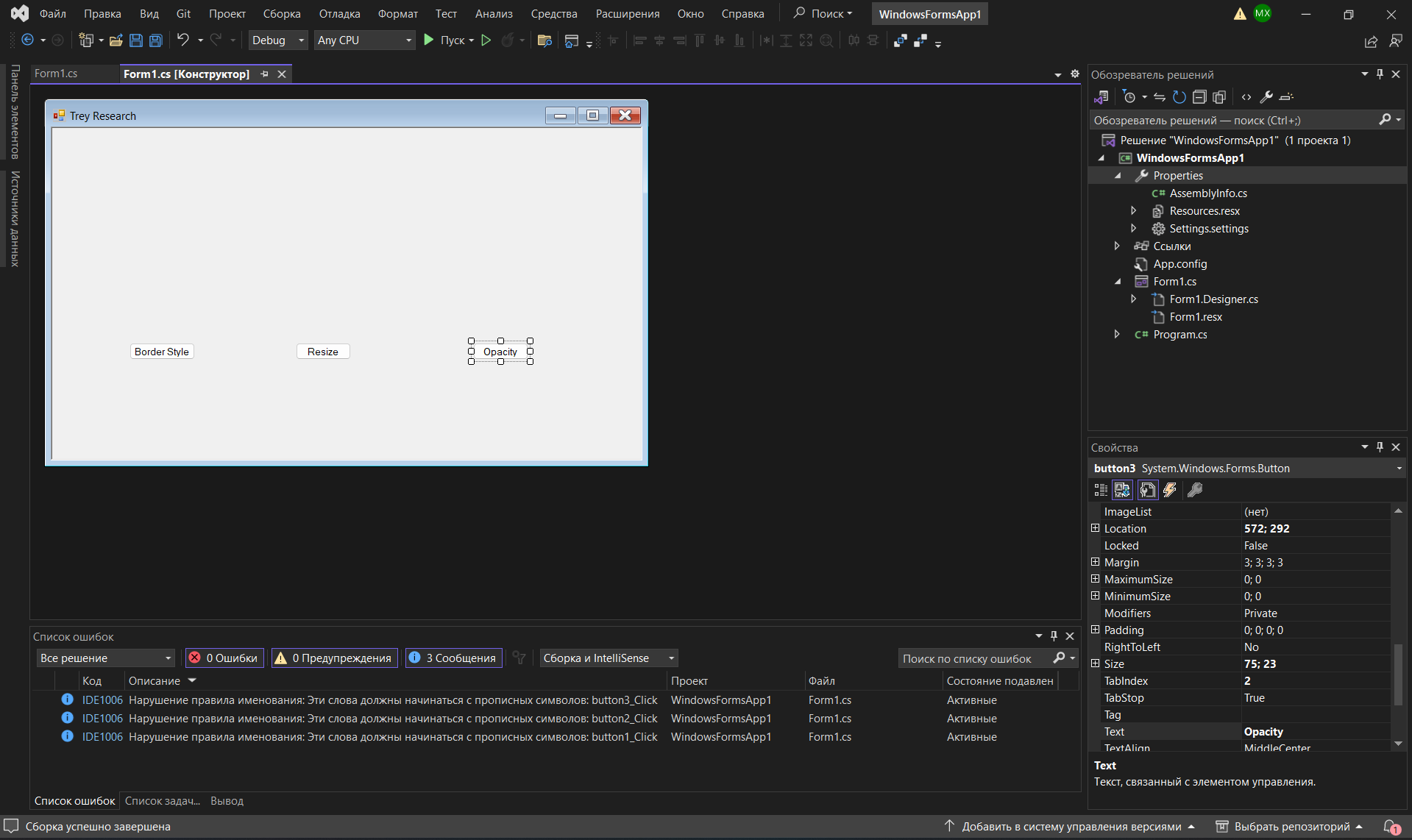


Возвратитесь в окно конструктора, дважды щелкните кнопку **Resize** и добавьте следующую строку: this.Size = new Size (300, 500);

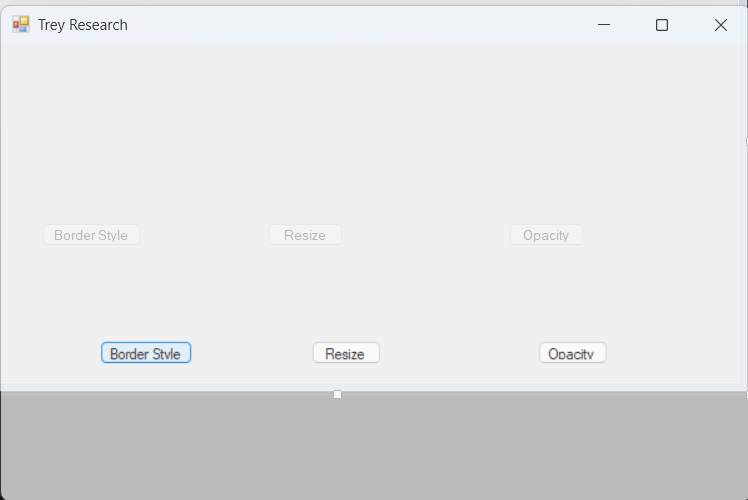


Возвратитесь в окно конструктора, дважды щелкните кнопку **Opacity** и добавьте следующую строку: this.Opacity = 1;

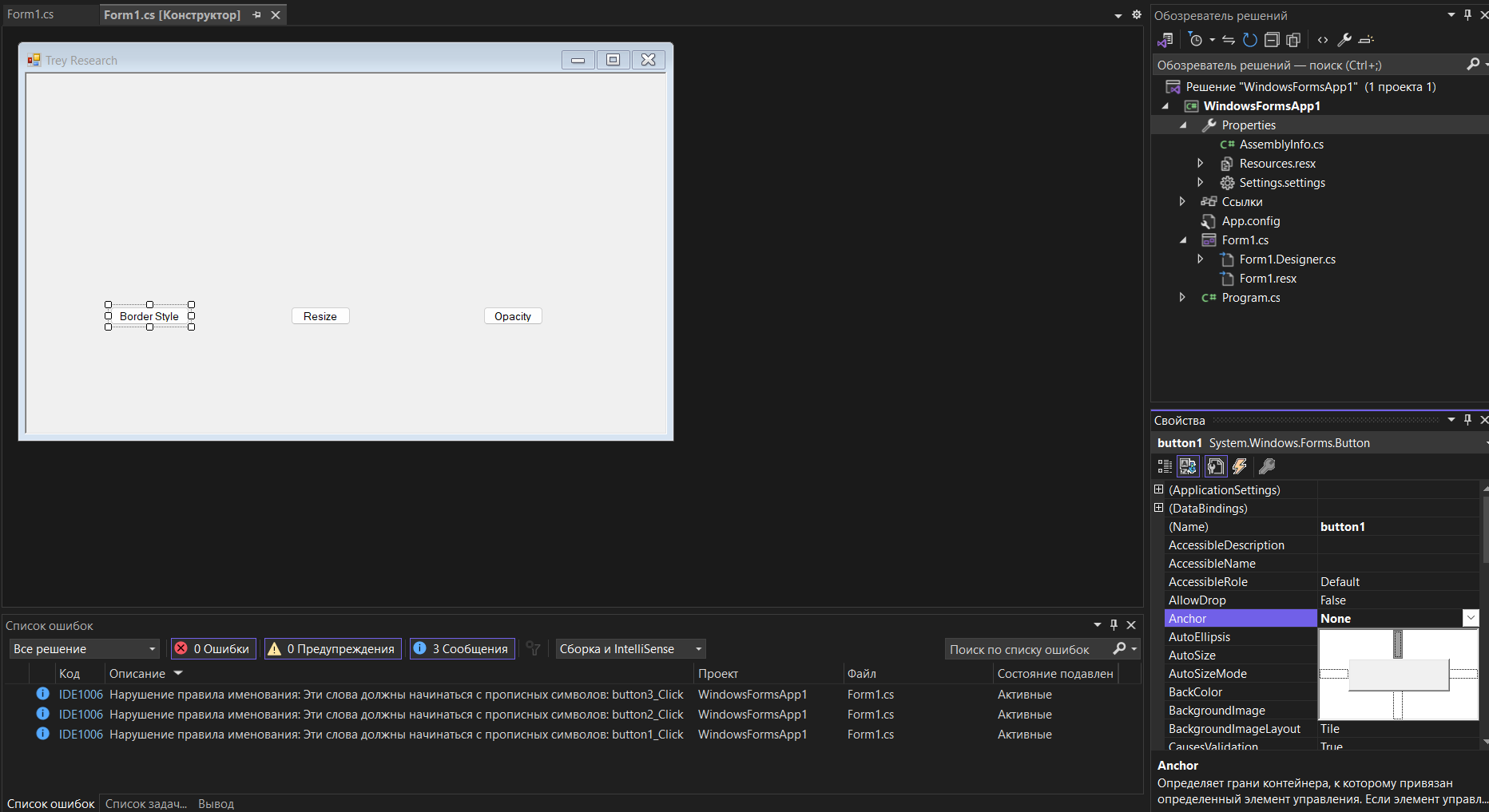
1. **Запуск готового решения**

****

Для построения решения выберите меню **Build (Построение (у меня Сборка))**, далее команду **Build Solution (Построить решение (у меня Собрать решение))**. При наличии ошибок исправьте их и снова постройте решение. В дальнейшем при необходимости выбора последовательности действий очередность команд будет описываться, например, так: **Build | Build Solution**.



Нажмите **Ctrl + F5** или выберите **Debug (Отладка) | Start Without Debugging (Запуск без отладки)**, чтобы запустить приложение. Щелкайте каждую кнопку и наблюдайте, как изменяется вид формы.

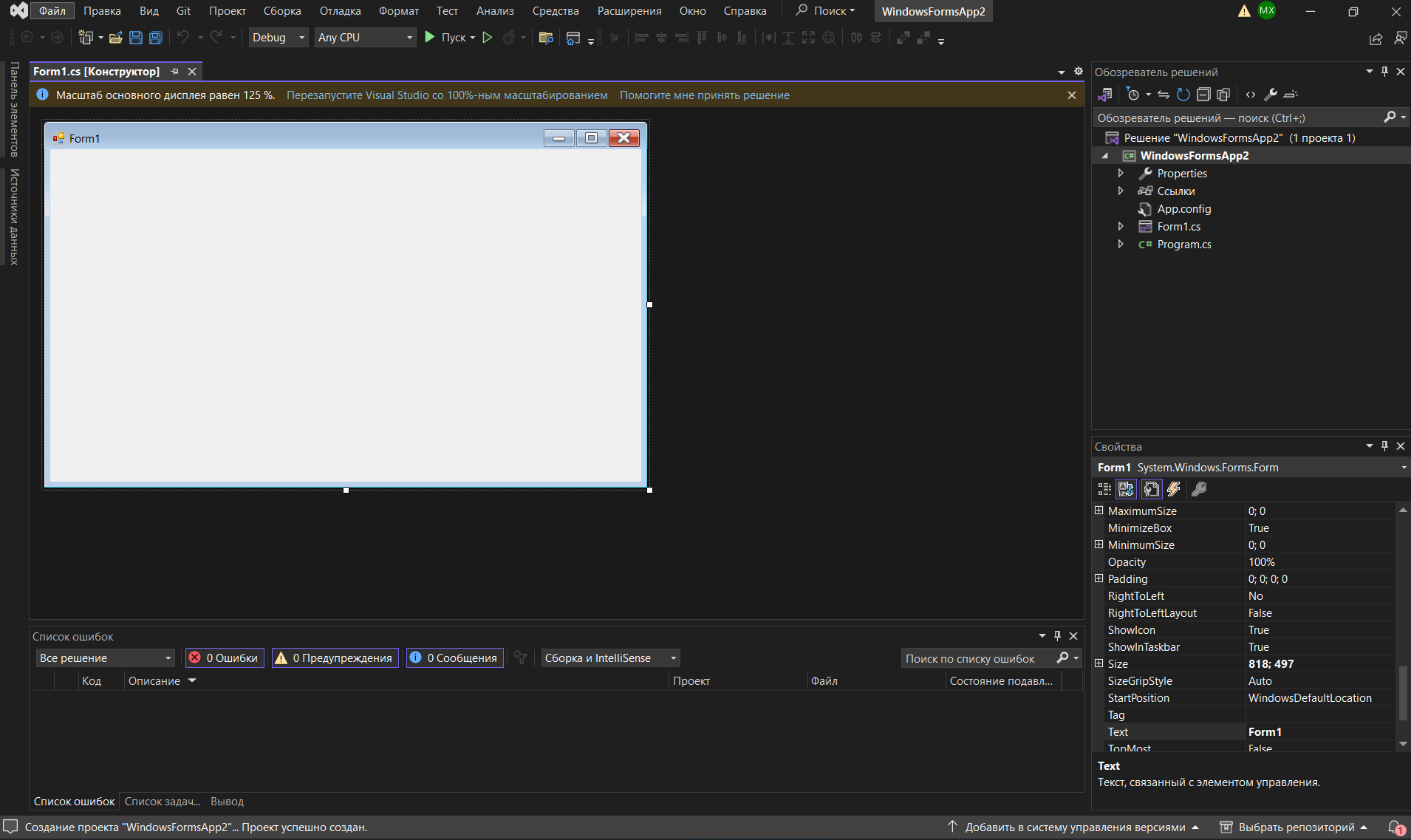


Измените поочередно расположение левой и верхней границ формы и сравните поведение кнопок внутри формы. Обратите внимание, что расстояние до этих границ от кнопки **Border Style** остается постоянным. Почему?

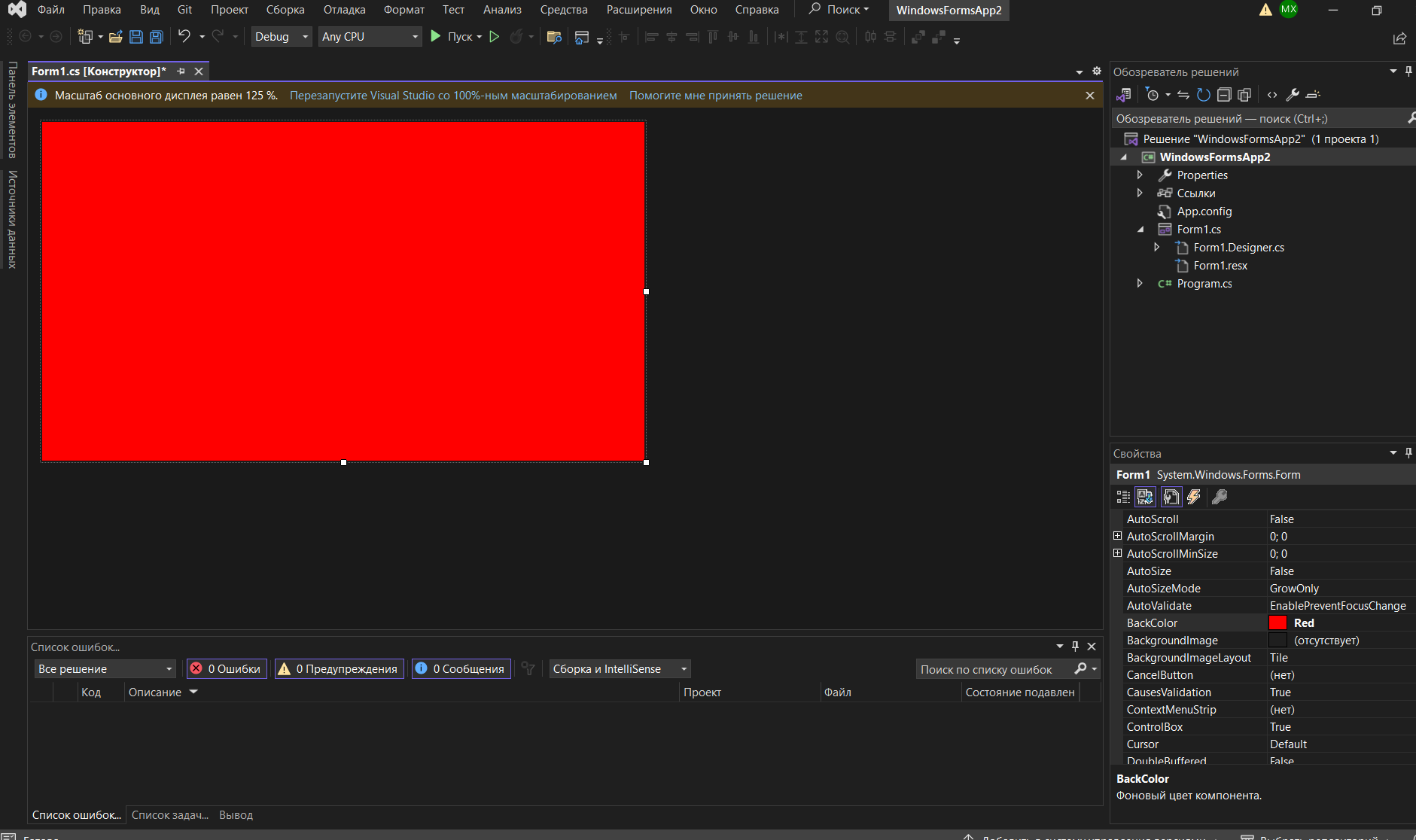
*Потому что при изменении левой и верхней границ формы мы меняем положение формы на экране, а внутренние элементы (кнопки) остаются на своих фиксированных координатах внутри формы.*

**Упражнение 2. Создание непрямоугольной формы Windows**

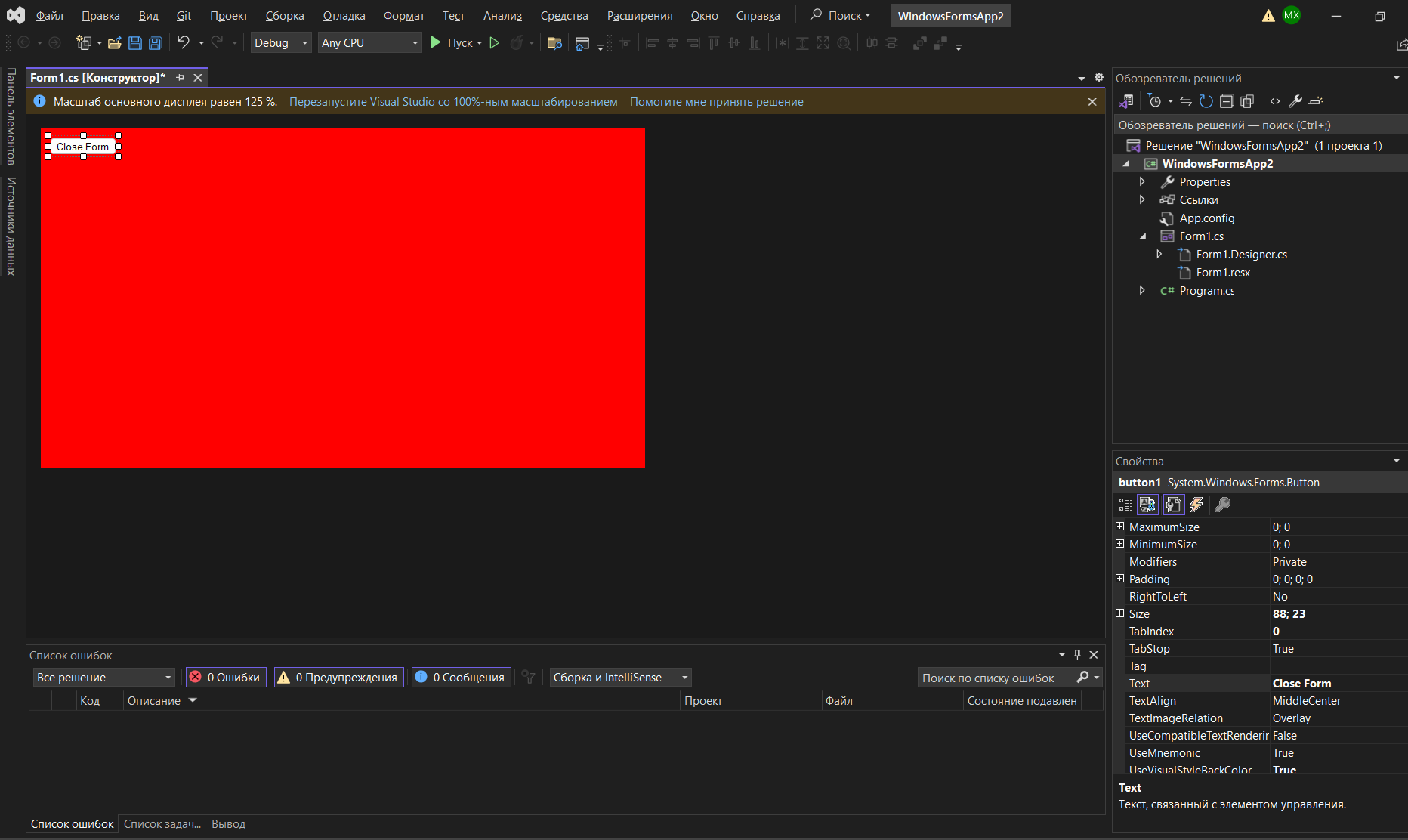
1. **Создаем треугольную форму Windows**



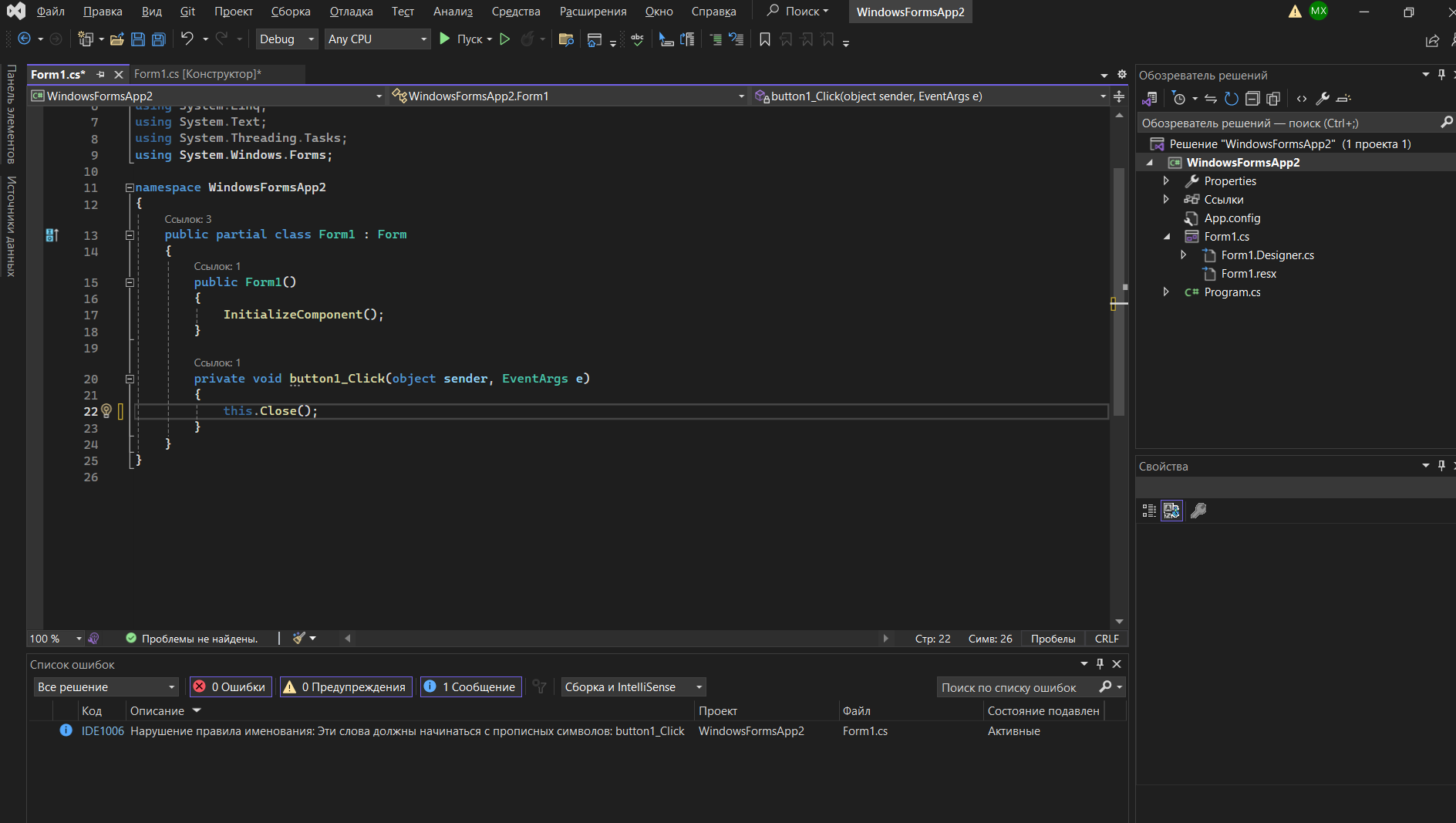
Откройте Visual Studio и создайте новый проект Windows Forms. Проект откроется с формой по умолчанию с именем **Form1** в конструкторе.



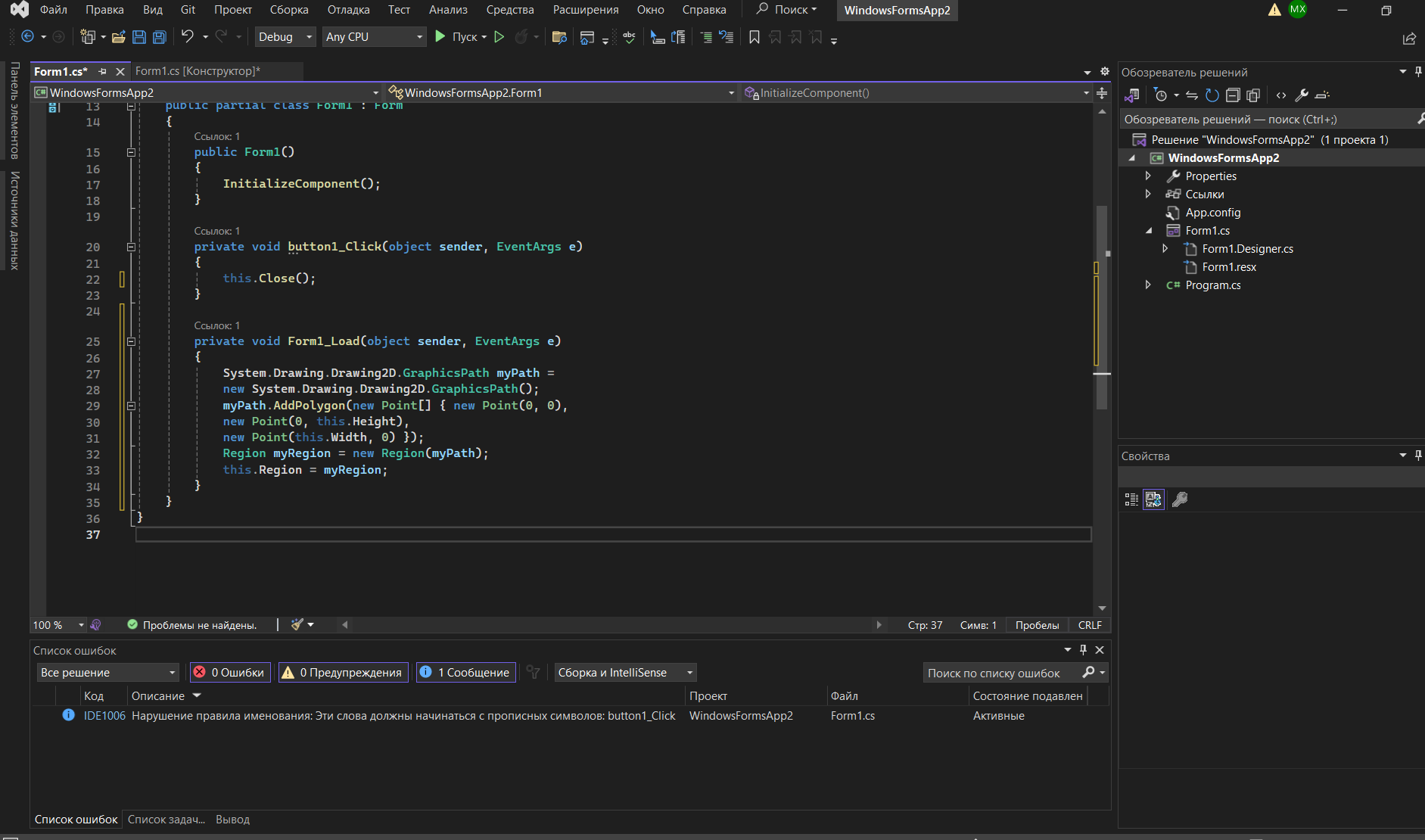
В окне **Properties** задайте свойству **FormBorderStyle** значение **None**, а свойству **BackColor** значение **Red**. В этом случае форму легче будет увидеть при тестировании приложения.



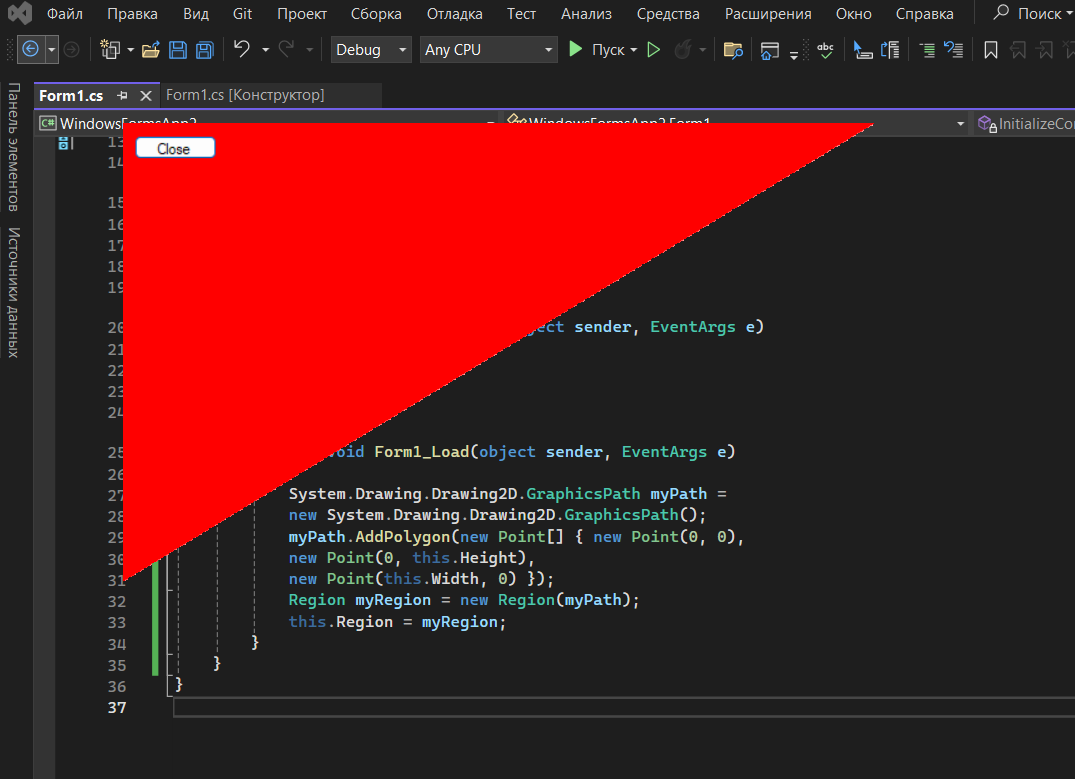
Перетащите кнопку из **Toolbox** (Панель элементов) в левый верхний угол формы. Задайте свойству **Text** кнопки значение **Close Form**.



Дважды щелкните кнопку **Close Form** и добавьте в обработчик события **Button1 Click** следующий код: this.Close();



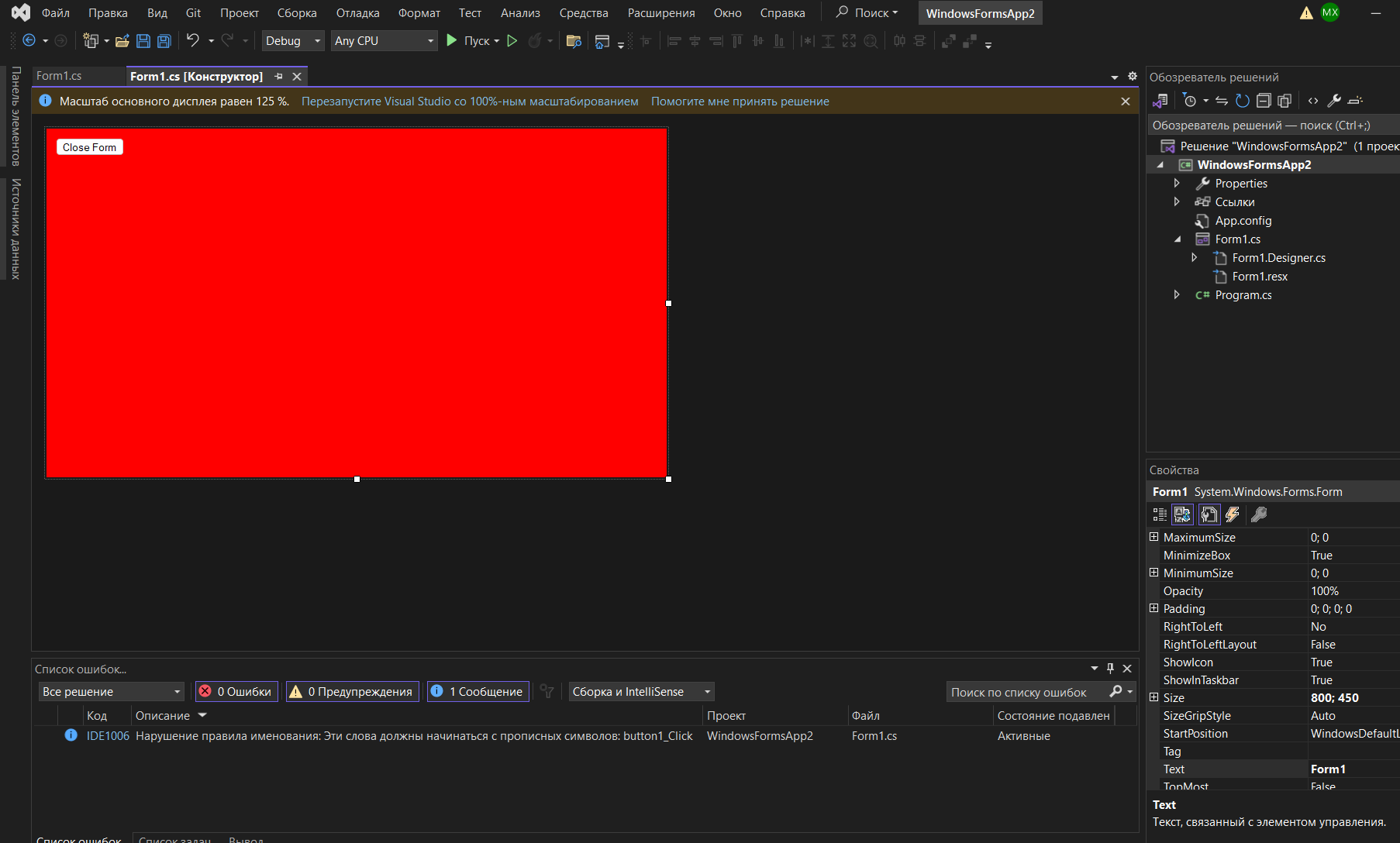
В конструкторе дважды щелкните форму, чтобы открыть обработчик события **Form1 Load**. Добавьте в этот метод следующий код (он задает области формы треугольную форму указанием многоугольника с тремя углами): System.Drawing.Drawing2D.GraphicsPath myPath = new System.Drawing.Drawing2D.GraphicsPath(); myPath.AddPolygon(new Point[] { new Point(0, 0), new Point(0, this.Height), new Point(this.Width, 0) }); Region myRegion = new Region(myPath); this.Region = myRegion;

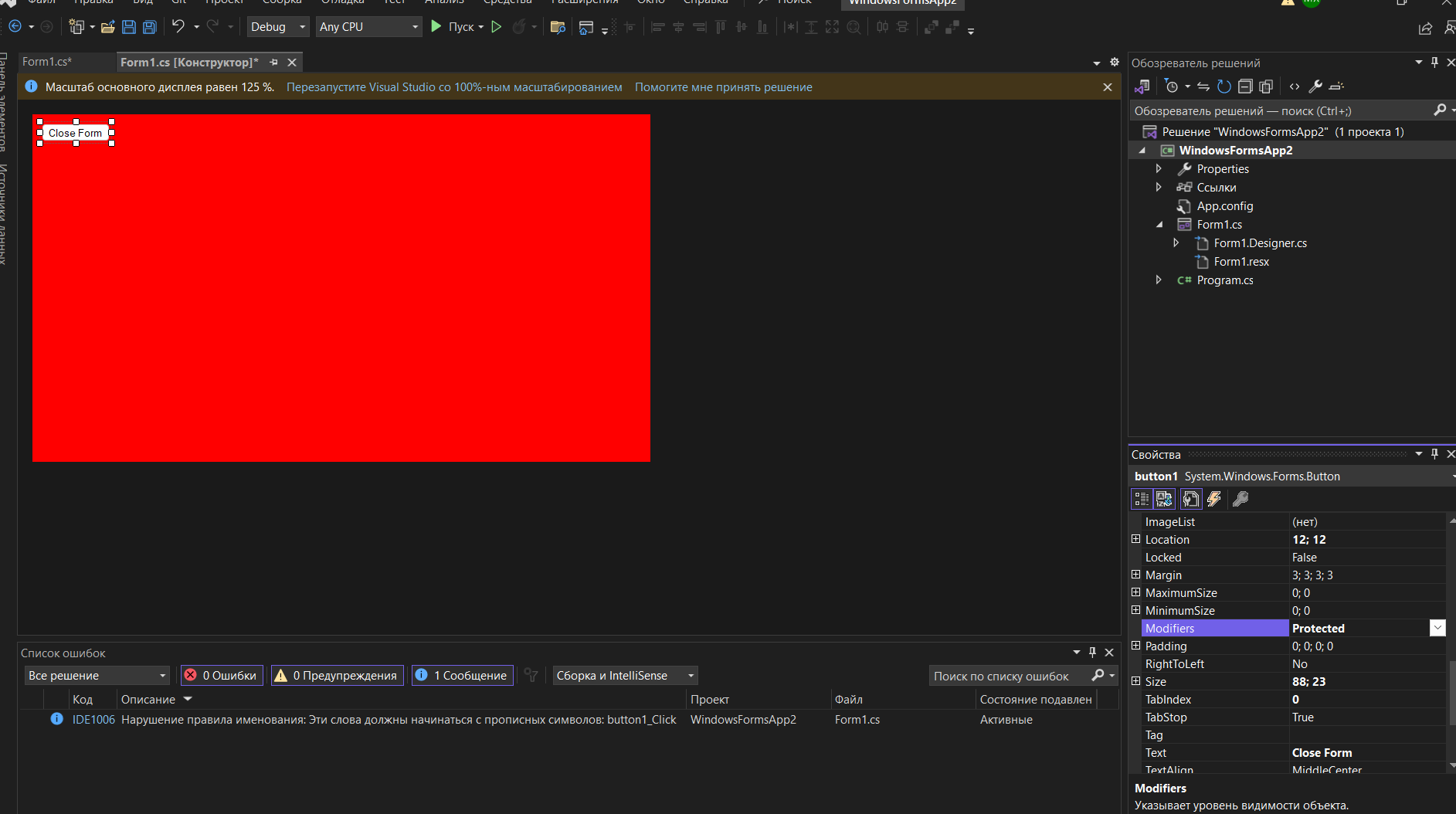
Постройте и запустите приложение. Появится треугольная форма.

**Упражнение 3. Создание наследуемой формы**

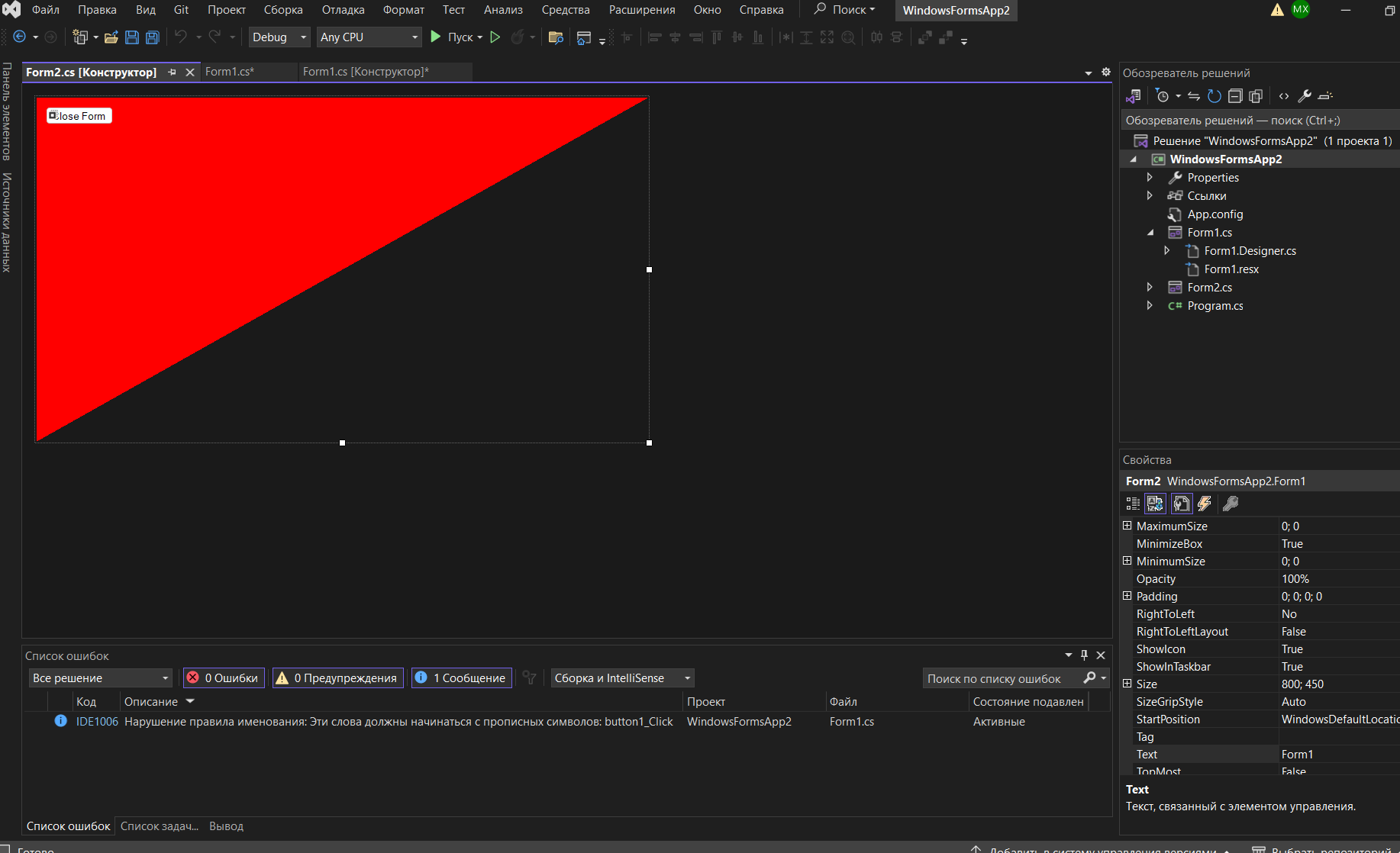
1. **Создаем наследуемую форму**

****

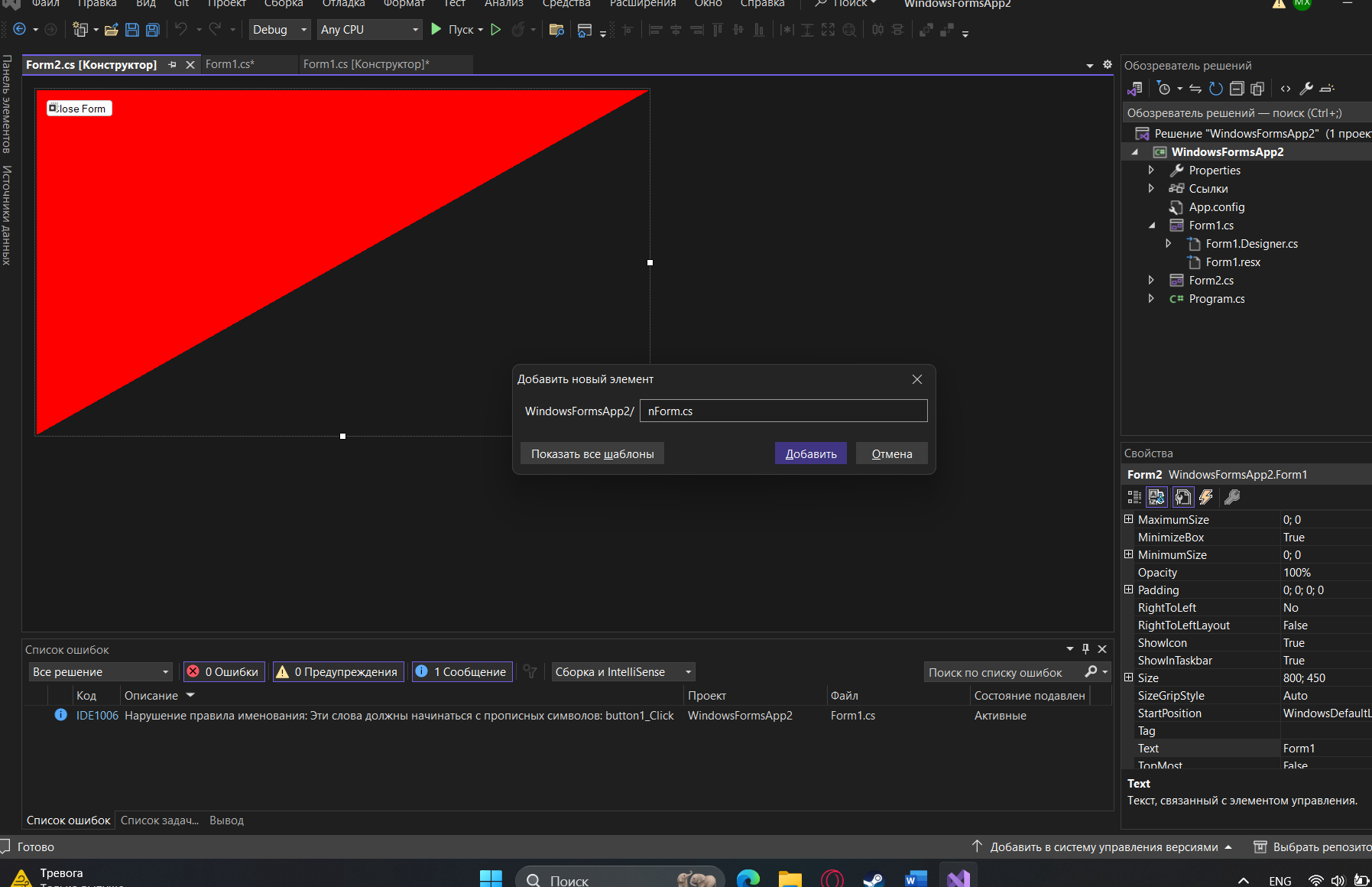
Откройте проект из предыдущего упражнения. Базовой формой для создания производной будет треугольная форма.



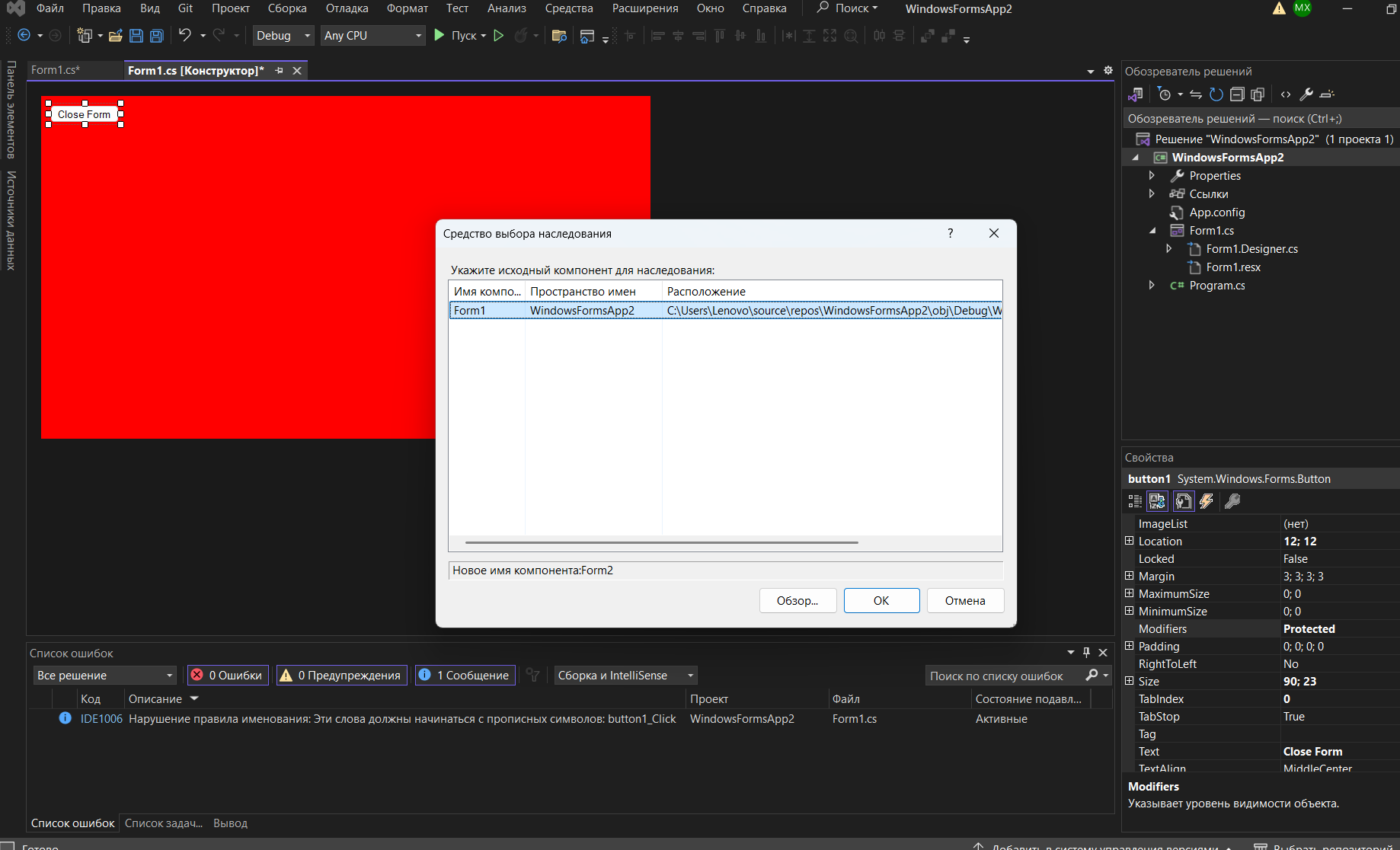
Для кнопки **Close Form** задайте свойство **Modifiers** как **protected**.



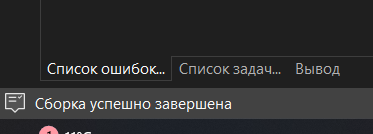
Добавьте производную форму: меню **Project (Проект) | Add Windows Form…(Добавить форму Windows)**, в окне **Categories (Категории)** укажите **Windows Form**, в окне **Templates (Шаблоны)** выберите **Inherited Form (Наследуемая форма)**.



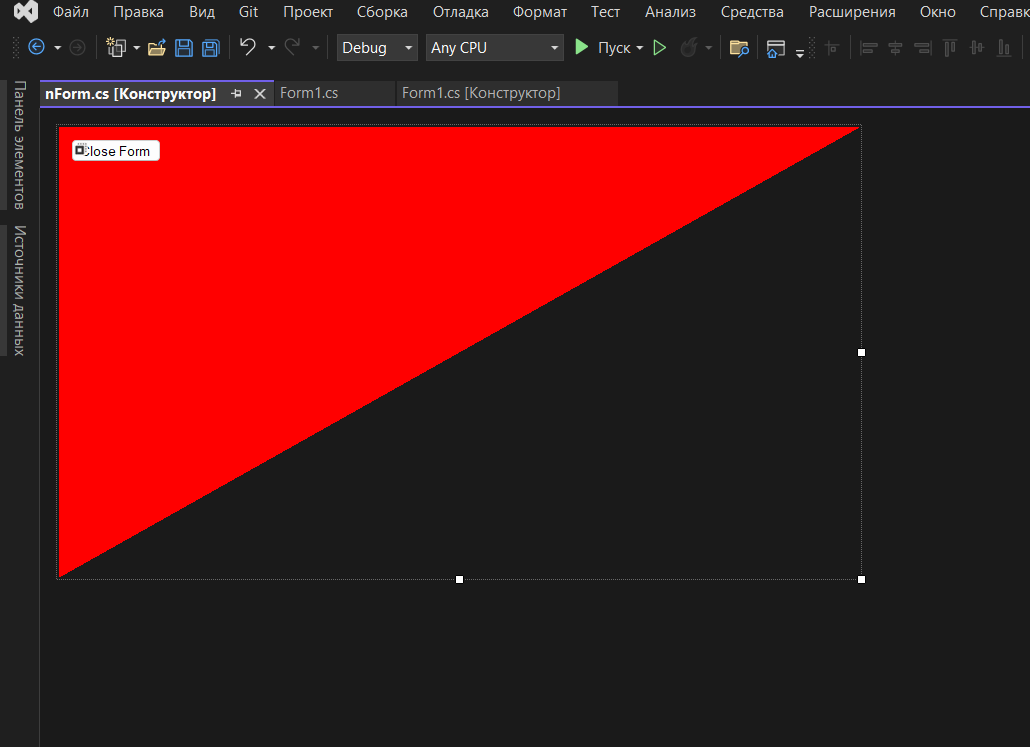
В окне **Add New Item** в поле **Name** укажите название формы: nForm.cs и нажмите **Add** для добавления формы.



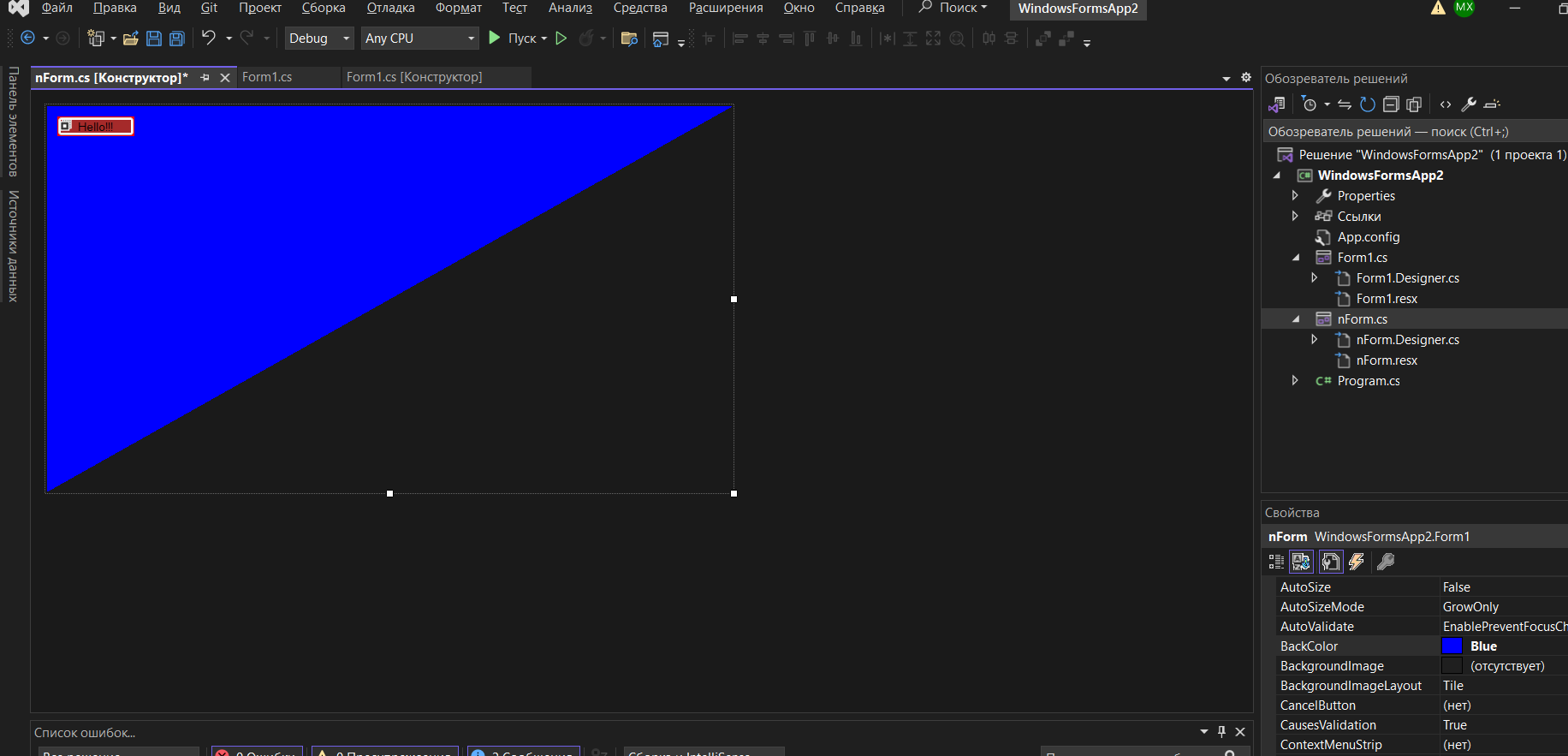
В появившимся окне **Inheritance Picker**, в котором отображаются все формы текущего проекта, выберите базовую форму **Form1** и нажмите ОК.



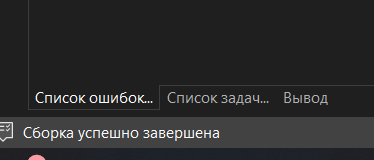
Постройте проект



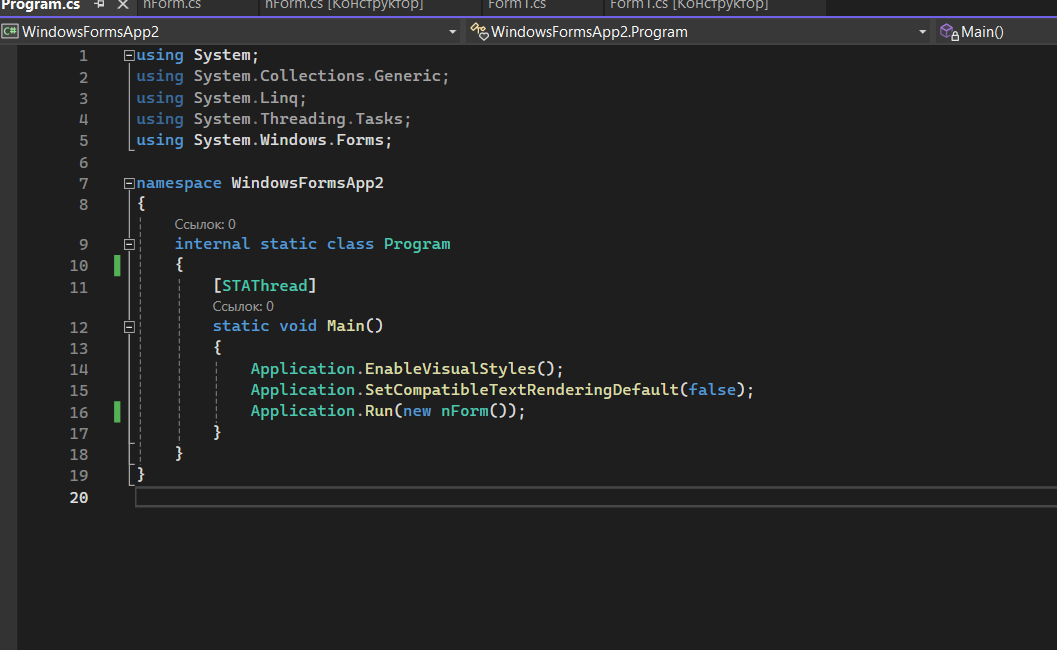
Откройте форму **nForm** в режиме конструктора. Проверьте, что она имеет треугольную форму и свойства базовой формы и элемента управления наследованы.



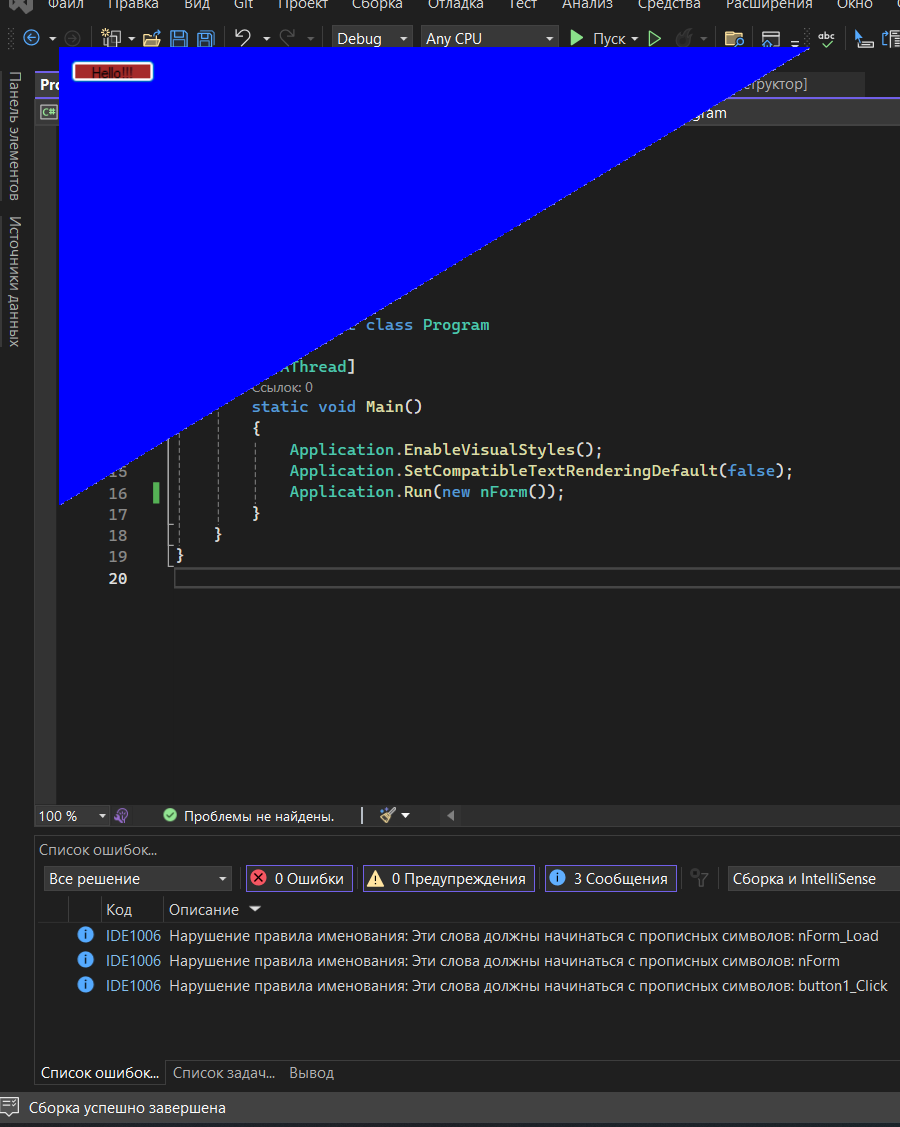
Настройте свойства производной формы: для кнопки: свойство **Text –** Hello!!!; свойство **BackColor** – Blue



Постройте проект



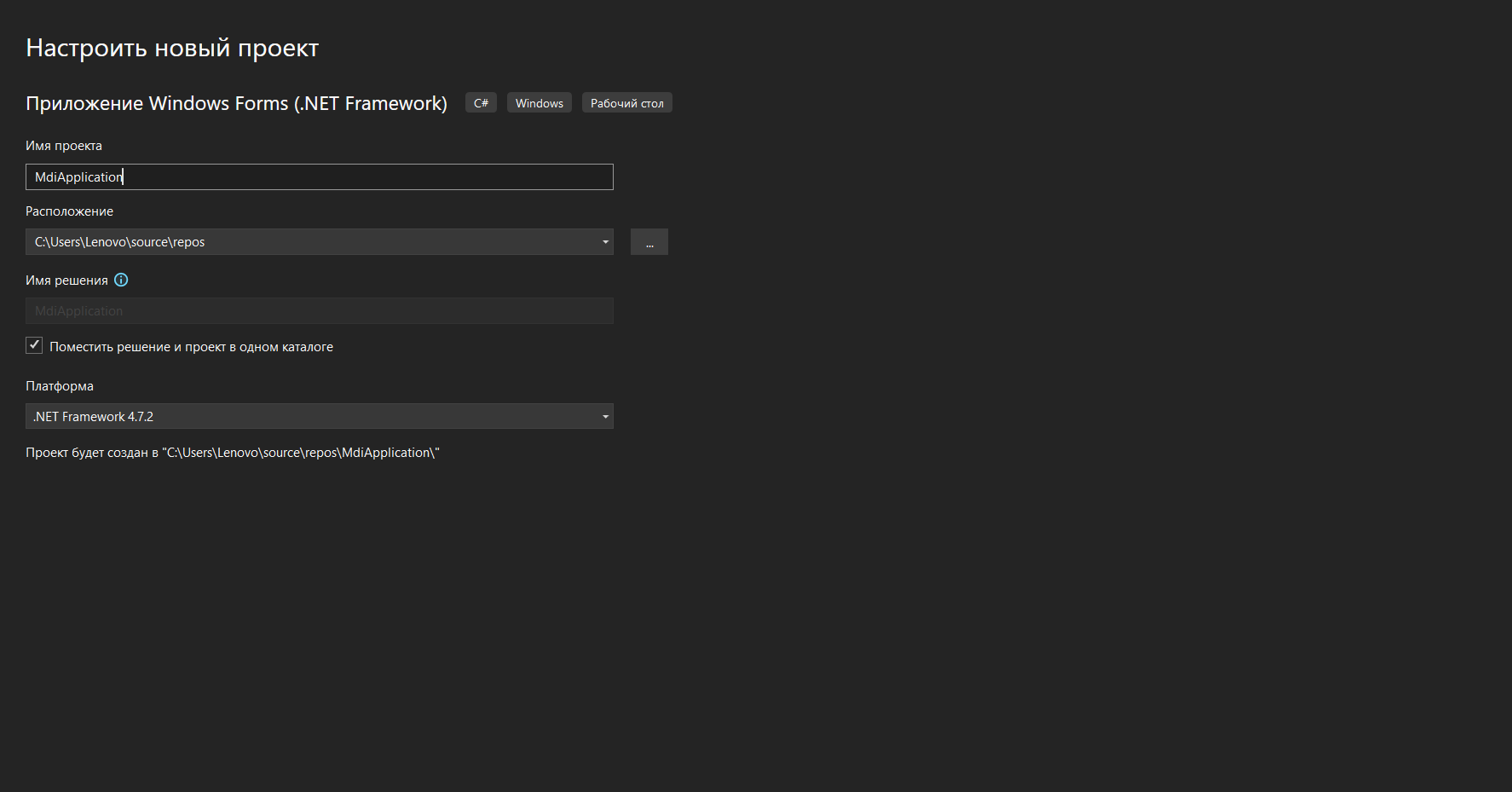
Задайте производную форму в качестве стартовой, указав в функции **Main** следующий код: Application.Run(new nForm());



Постройте и запустите приложение. Должна открыться производная форма со своими свойствами. Проверьте, наследуется ли закрытие формы кнопкой.

**Упражнение 4. Создание MDI-приложения**

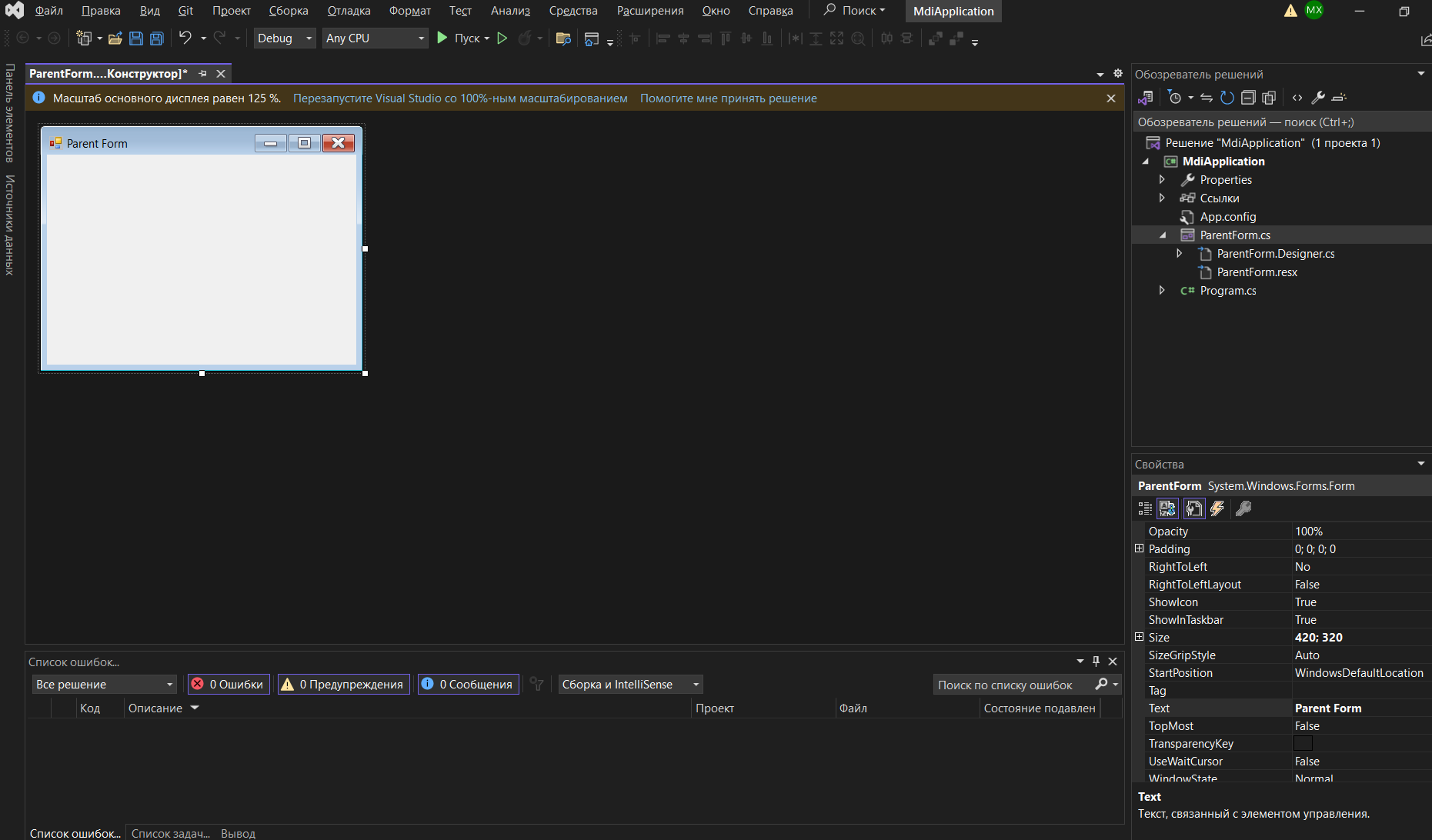
1. **Создание нового проекта с базовой формой**



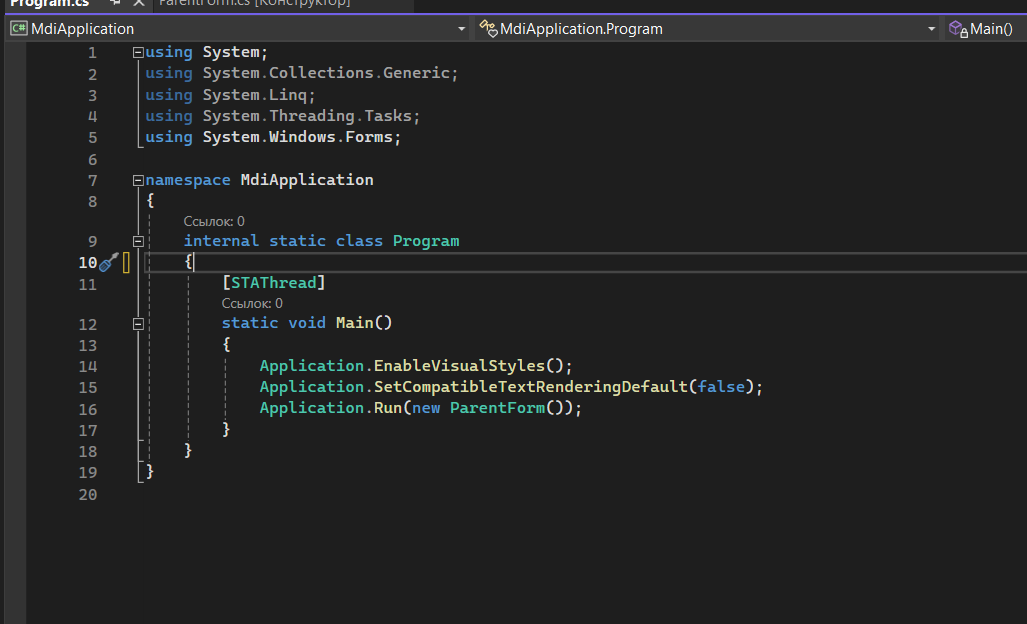
Создайте новый проект **Windows Forms**, укажите имя **MdiApplication**.



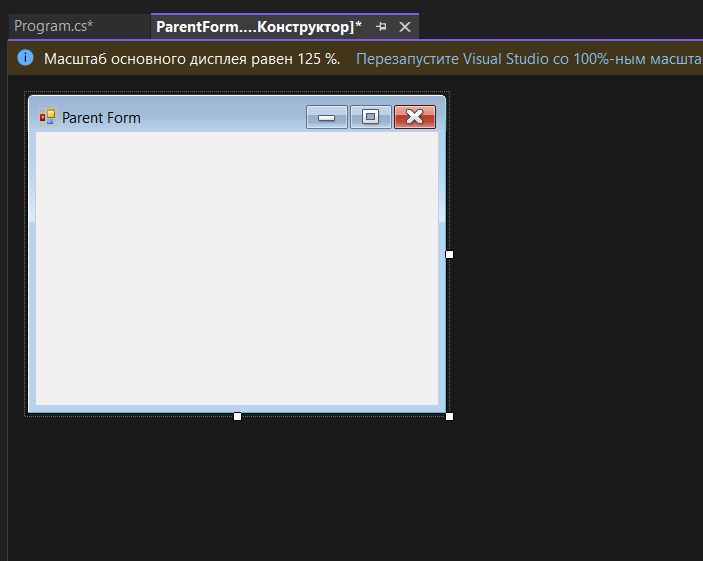
Переименуйте файл **Form1.cs** на **ParentForm.cs**.



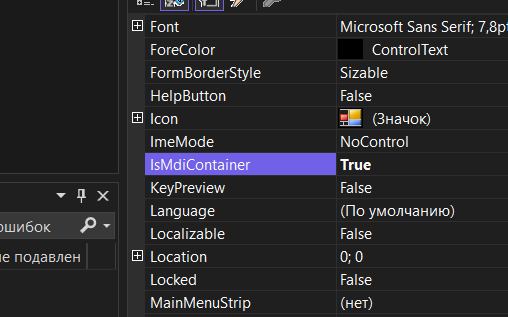
Для формы задайте следующие свойства: **Size –** 420;320; **Text –** Parent Form.



Проверьте, что произошли изменения в функции **Main** так, чтобы форма **ParentForm** стала стартовой.

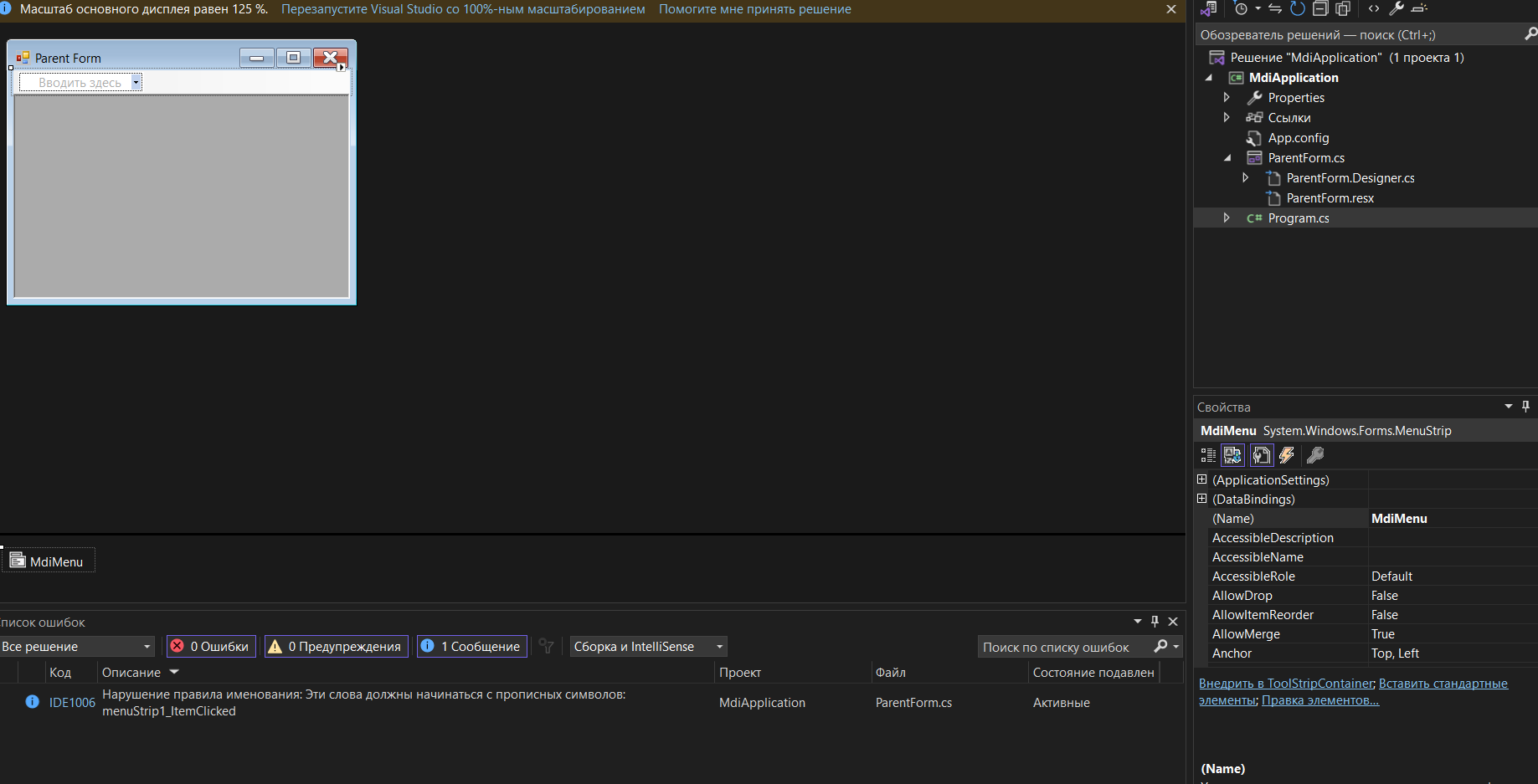


Откройте файл **ParentForm.cs** В режиме конструктора.

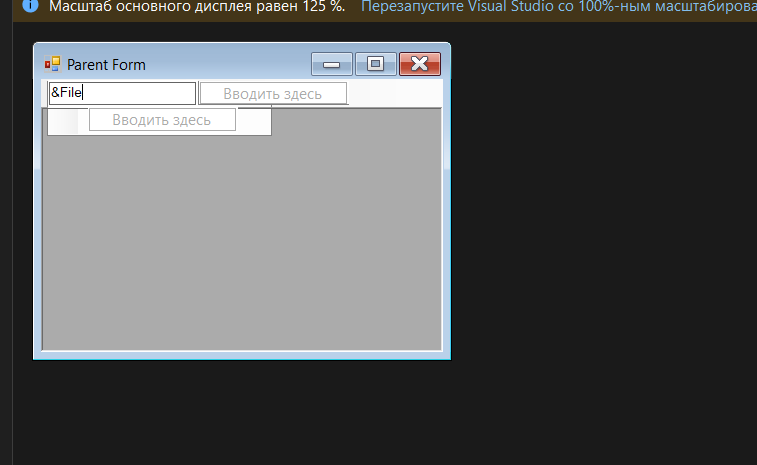


Для свойства формы **IsMdiContainer** задайте значение **True**. Таким способом эта форма будет определена как родительская форма MDI.

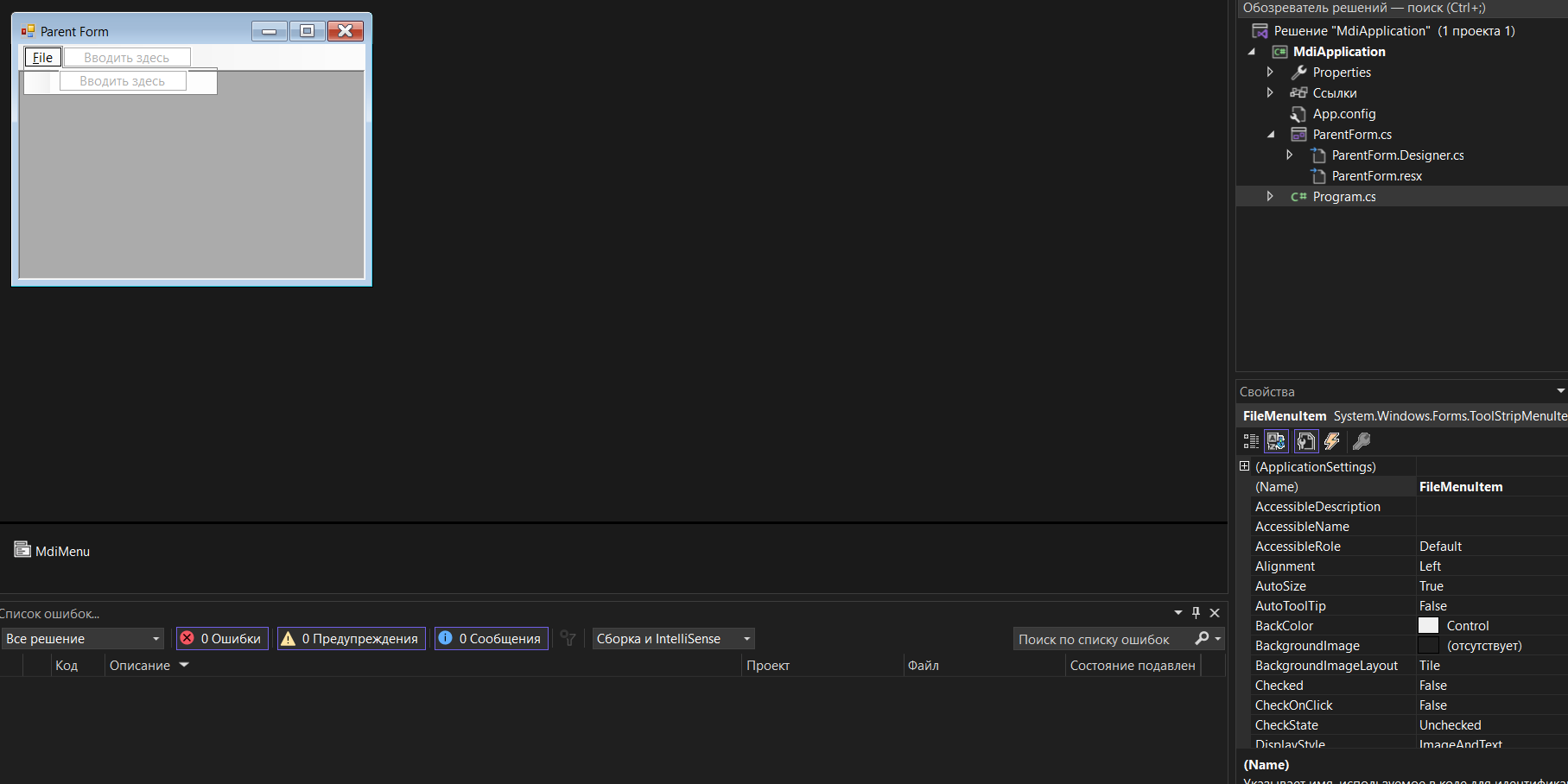
1. **Создание меню для работы с формами**



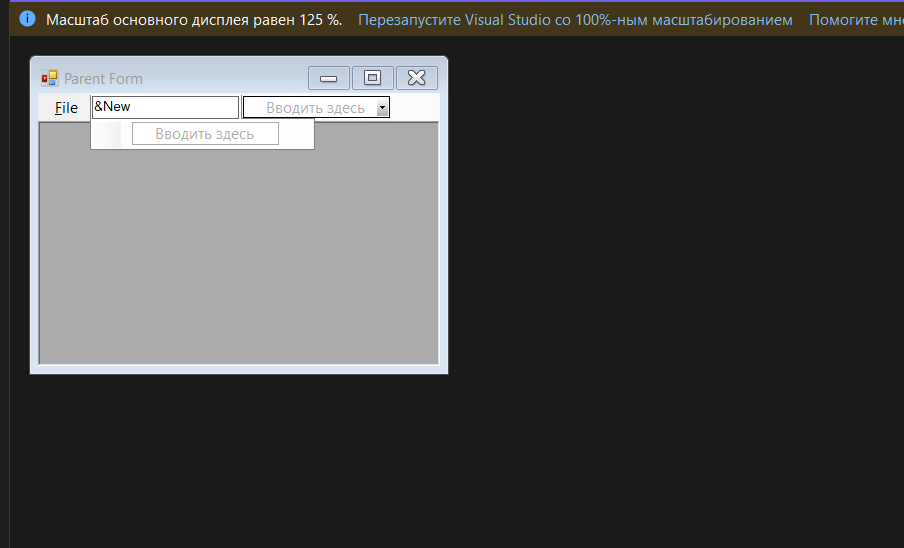
Откройте ПИ **Toolbox**, добавьте на форму ЭУ **MenuStrip** и задайте для его свойства **Name** значение **MdiMenu**.



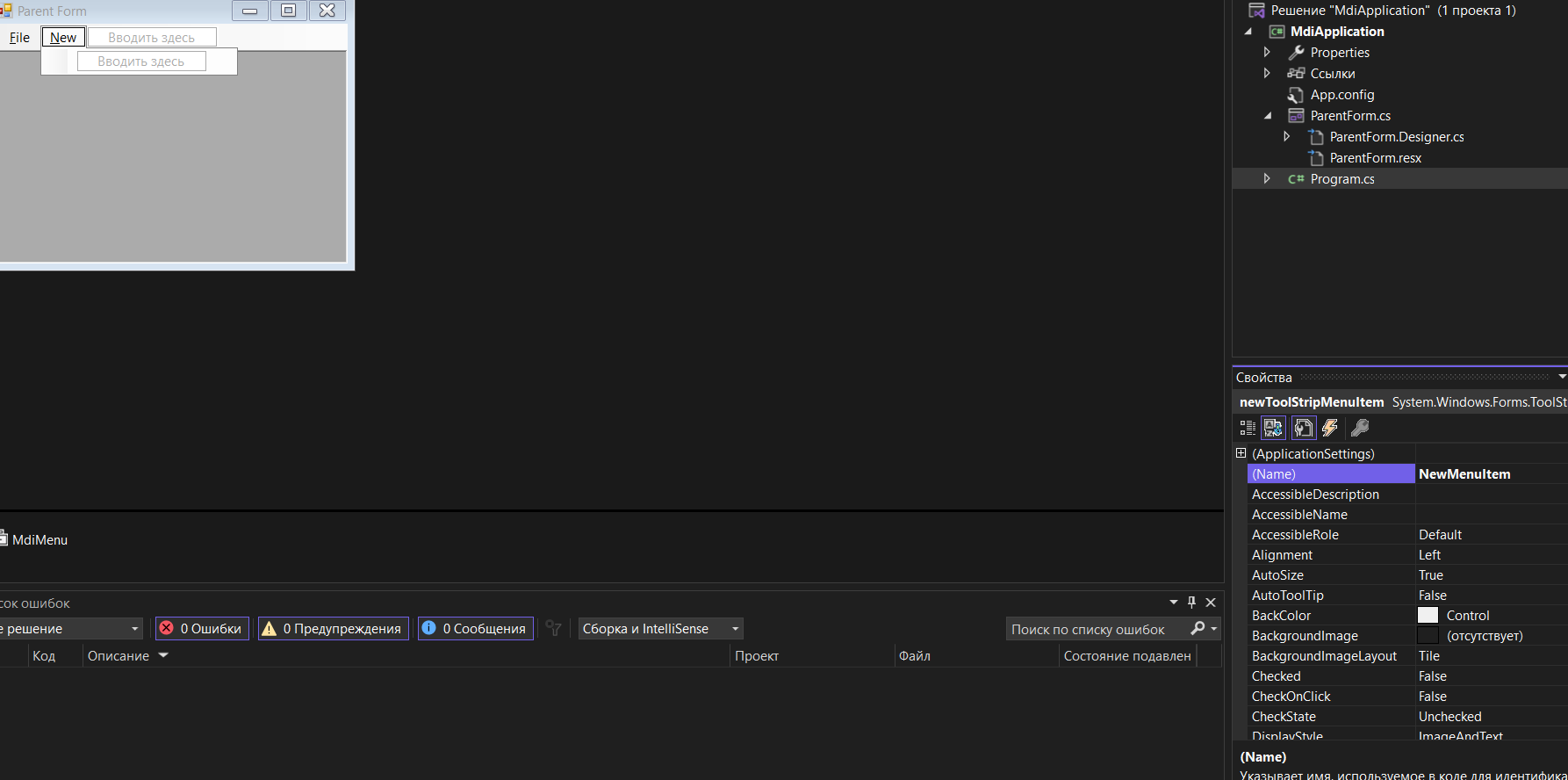
Выделите меню в верхней части формы и задайте имя первого пункта меню **&File**.



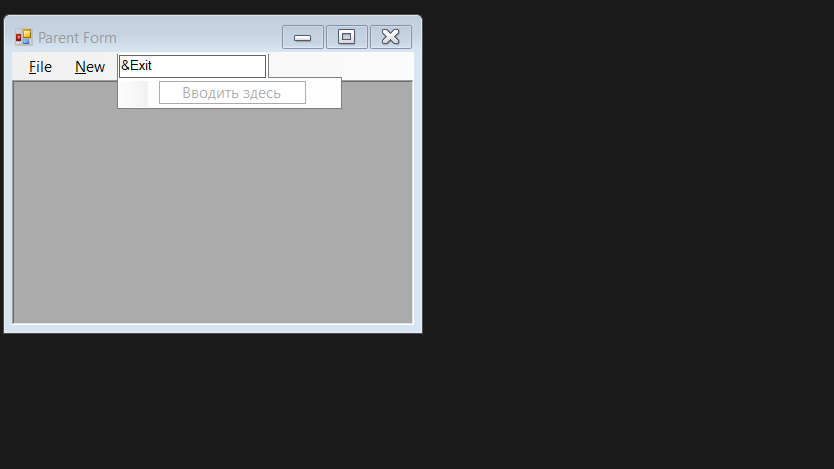
Для свойства **Name** пункта меню **File** задайте значение **FileMenuItem**.



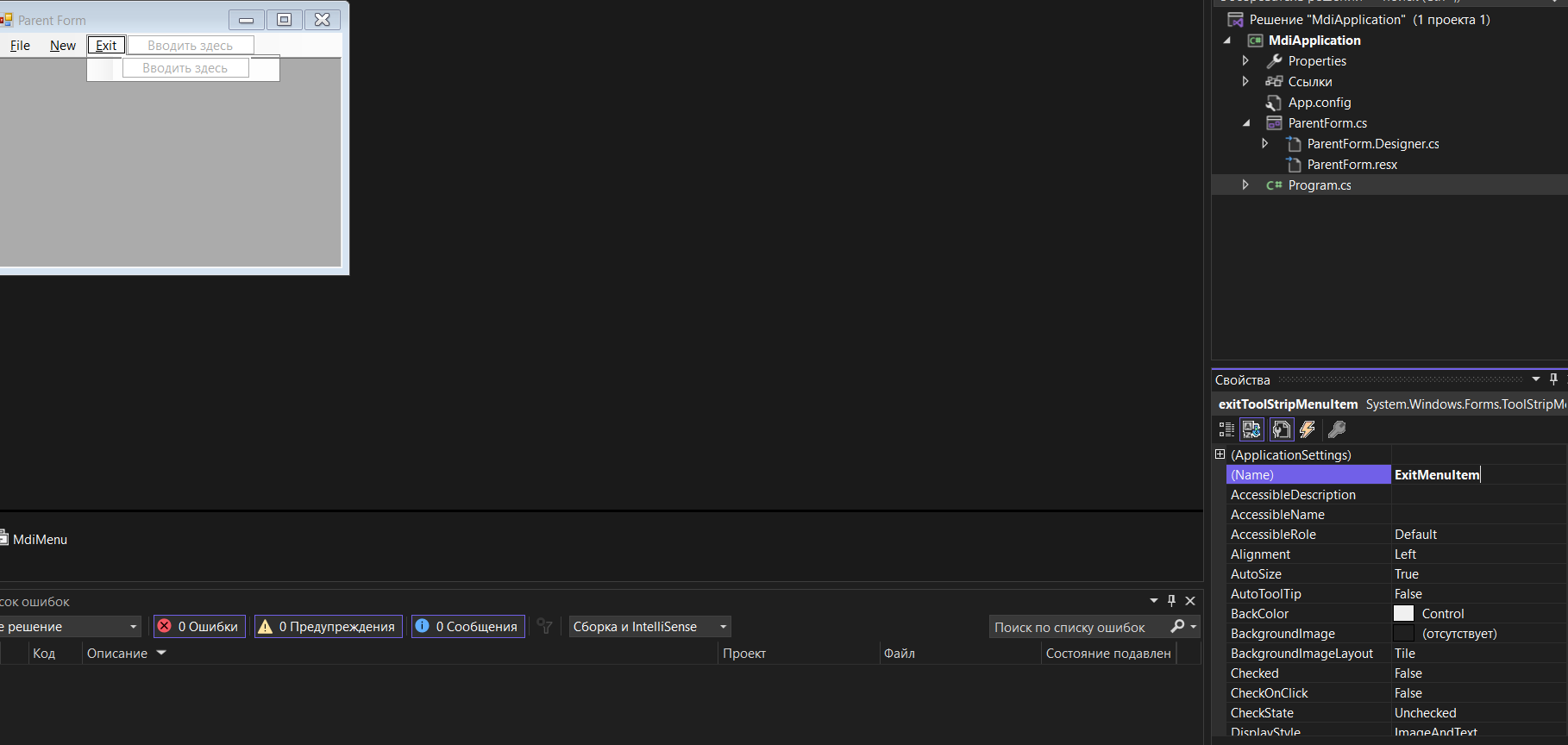
Выделите элемент, появившийся под элементом **File**, и задайте его как **&New**.



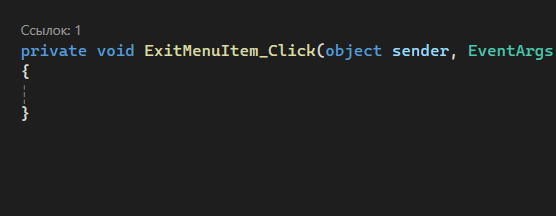
Для свойства **Name** пункта меню **New** задайте значение **NewMenuItem**.



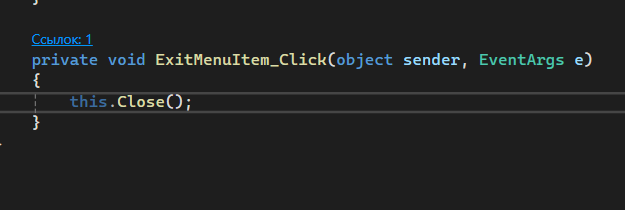
Выделите элемент, появившийся под элементом **New**, и задайте его как **&Exit**.



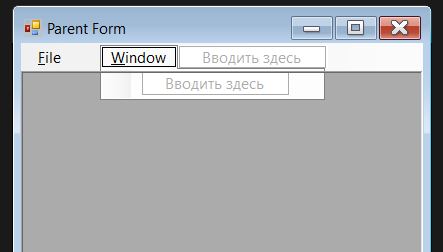
Для свойства **Name** пункта меню **Exit** задайте значение **ExitMenuItem**.



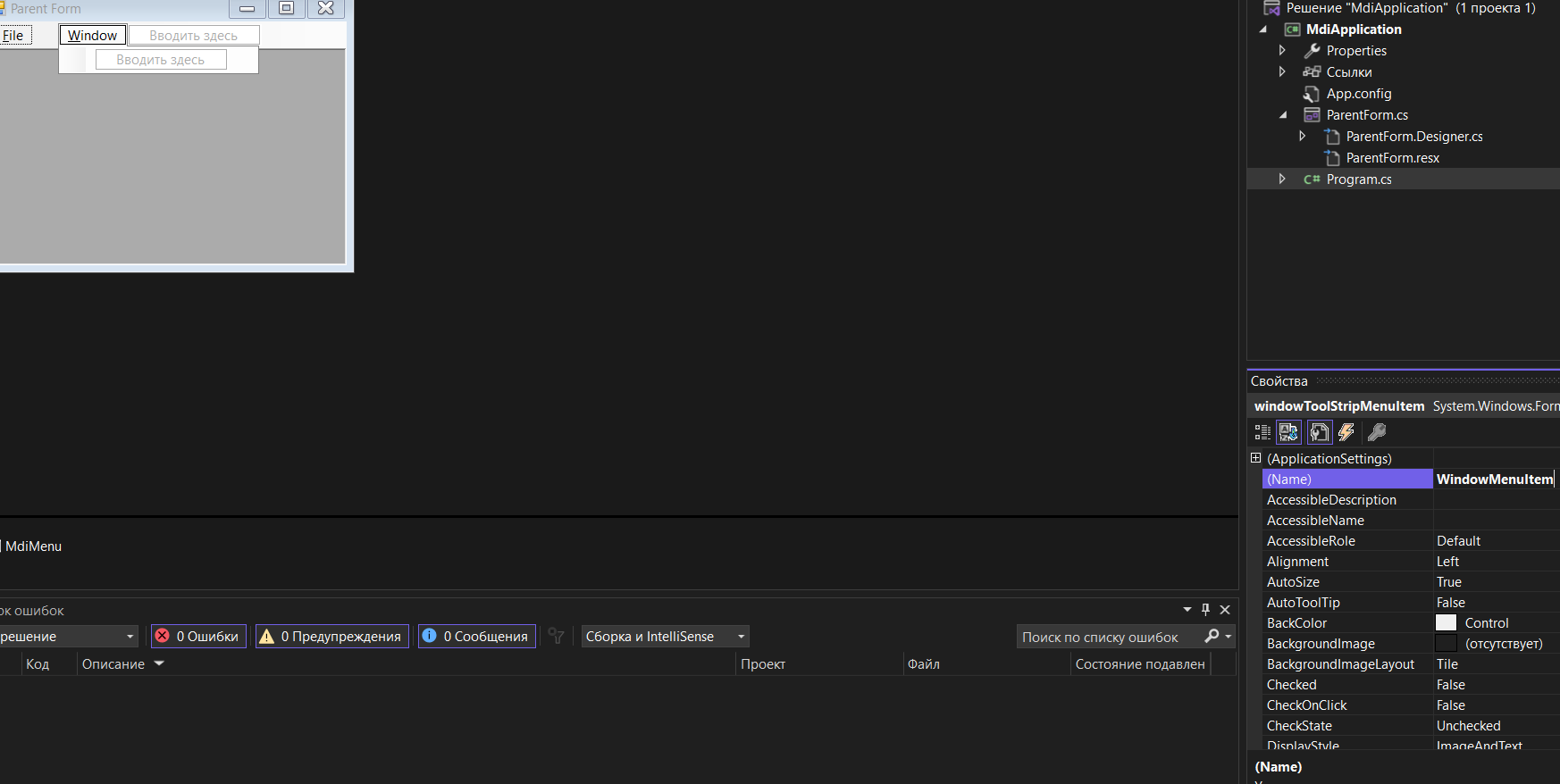
Дважды кликните левой кнопкой мыши по пункту меню **Exit** для создания обработчика события **Click**.



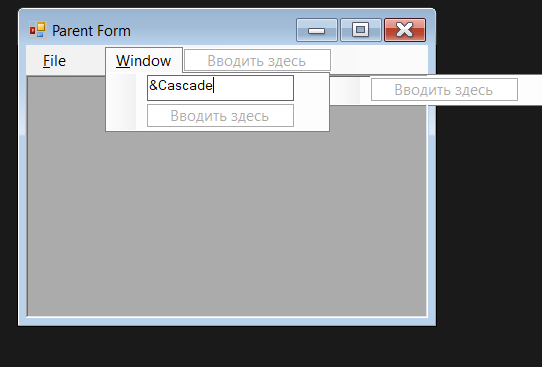
В обработчик события **Click** для пункта меню **Exit** добавьте следующий код: this.Close();



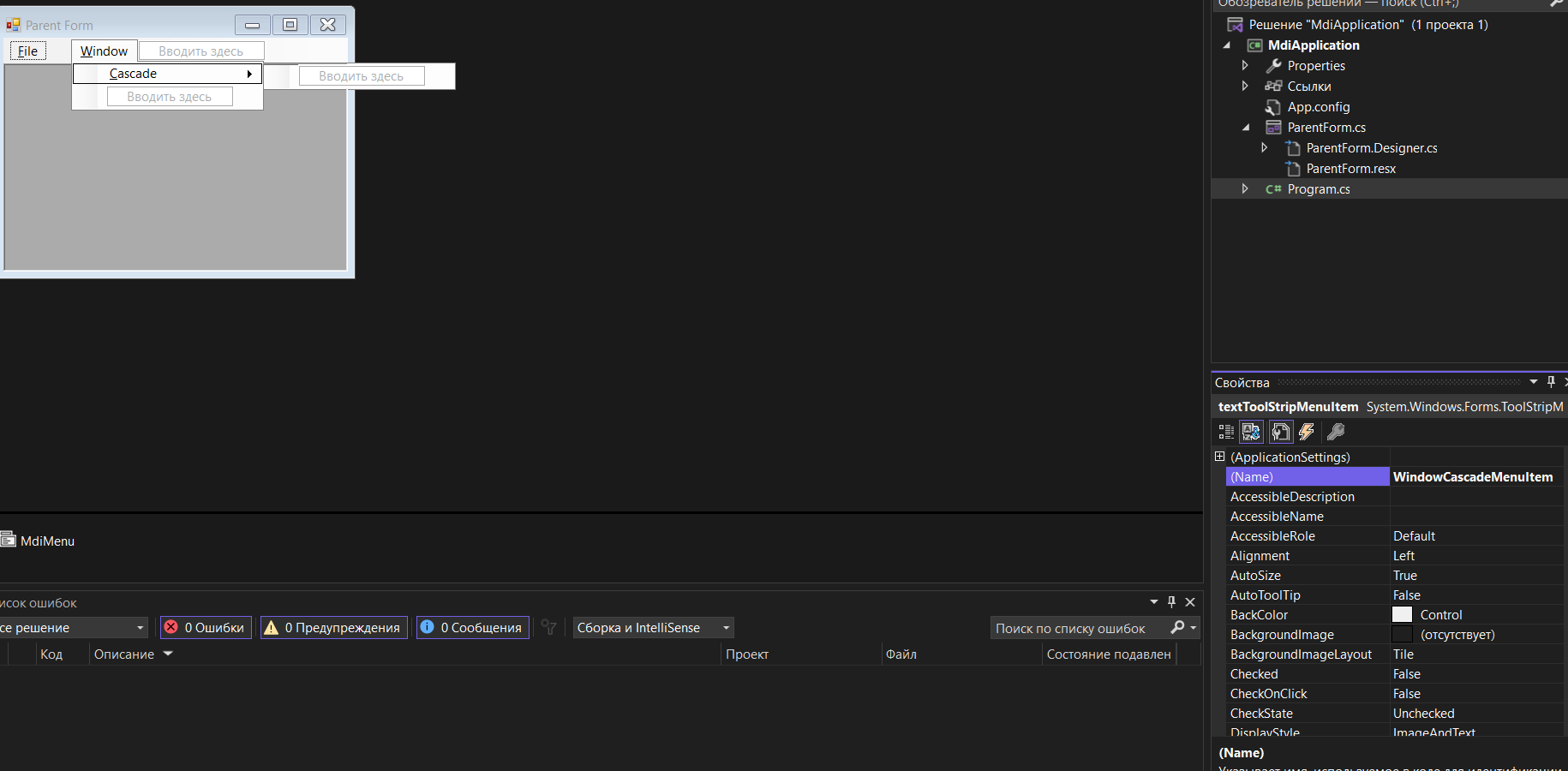
Выделите второй пункт меню справа от **File** и задайте его значением **&Windows**.



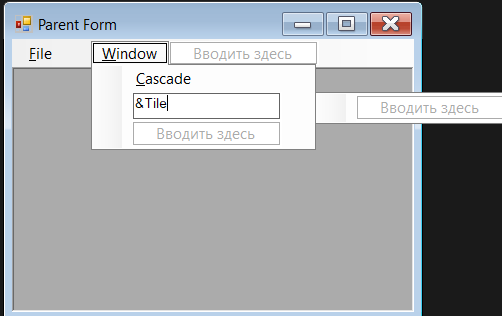
Для свойства **Name** пункта меню **Window** задайте значение **WindowMenuItem**.



Выделите элемент, появившийся под элементом **Window**, и задайте его как **&Cascade**.



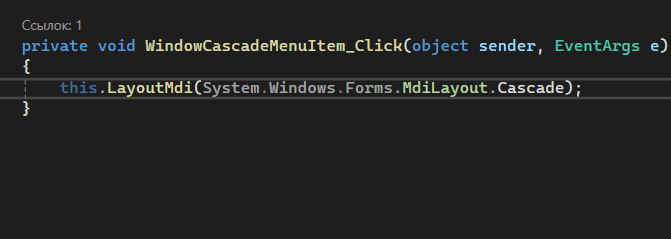
Для свойства **Name** пункта меню **Cascade** задайте значение **WindowCascadeMenuItem**.



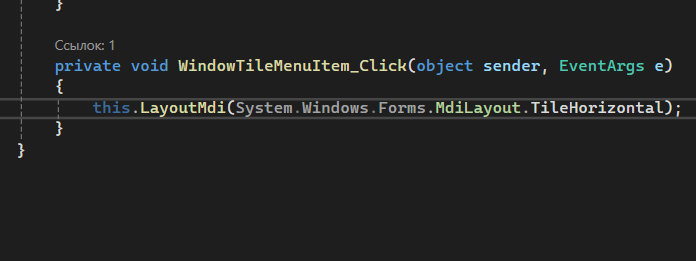
Выделите элемент, появившийся под элементом **Cascade**, и задайте его как **&Tile**.



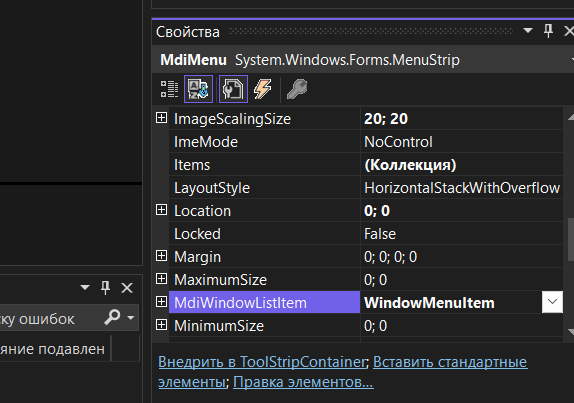
Для свойства **Name** пункта меню **Tile** задайте значение **WindowTileMenuItem**.



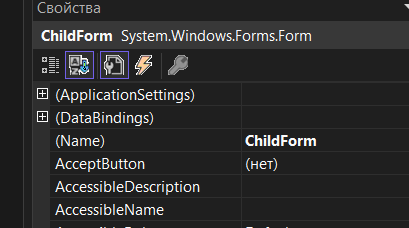
Дважды кликните левой кнопкой мыши по пункту меню **Cascade** для создания обработчика события **Click**.



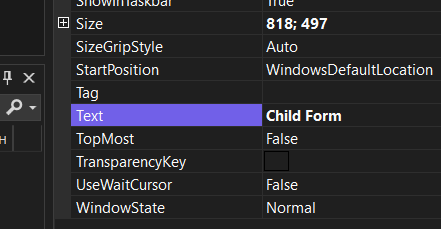
В обработчик события **Click** для пункта меня **Tile** добавьте следующий код: this.LayoutMdi(System.Windows.Forms.MdiLayout.TileHorizontal);



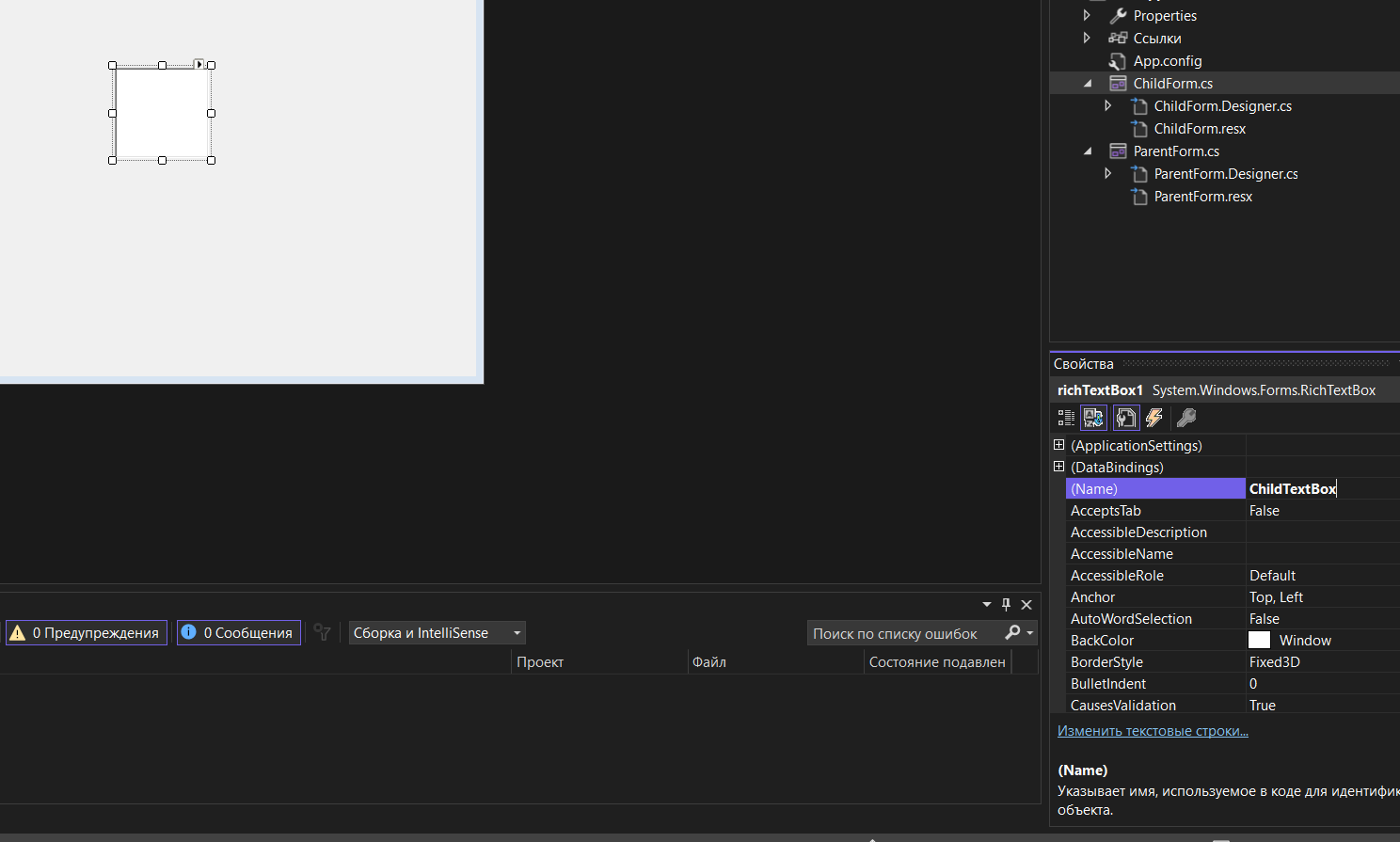
В конструкторе выберите компонент **Mdimenu**. Укажите в свойстве **MdiWindowListItem** имя пункта, созданного для этого – **WindowMenuItem**.



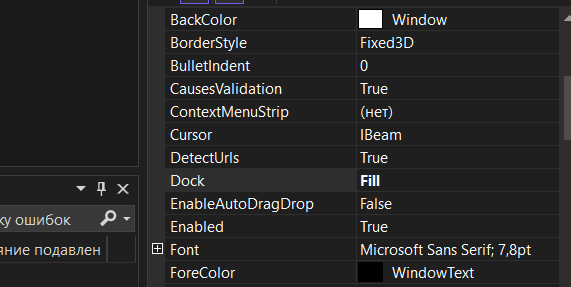
Задайте имя формы **ChildForm.cs**.



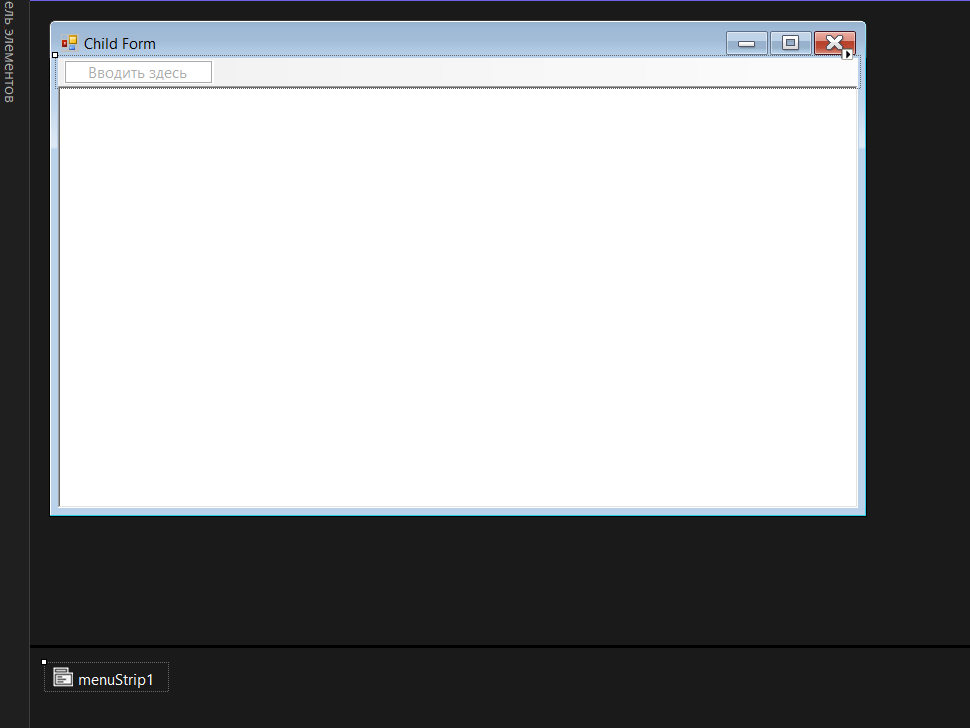
Для свойства **Text** формы задайте значение **Child Form**.



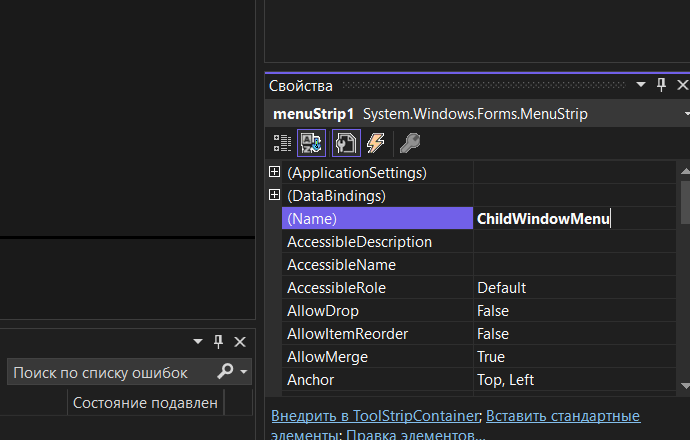
На ПИ **Toolbox** дважды кликните левой кнопкой мыши по ЭУ **RichTextBox** и задайте для его свойства **Name** значение **ChildTextBox**.



Для свойства **Dock** ЭУ **RichTextBox** задайте значение **Fill**.



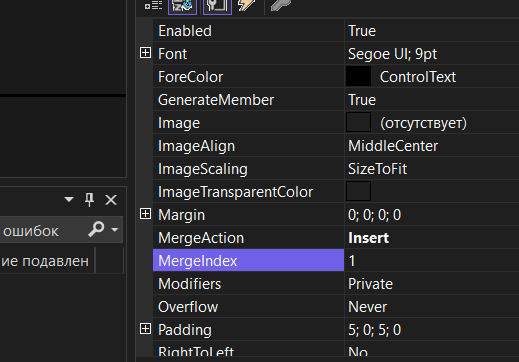
На ПИ **Toolbox** дважды кликните левой кнопкой мыши по ЭУ **MenuStrip**.



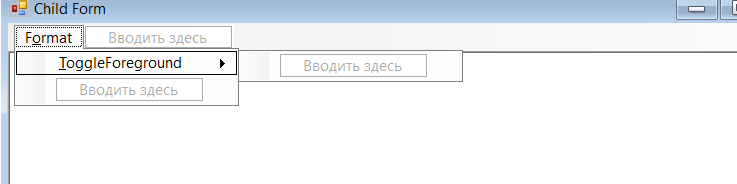
Для свойства **Name** ЭУ **MenuStrip** задайте значение **ChildWindowMenu**.



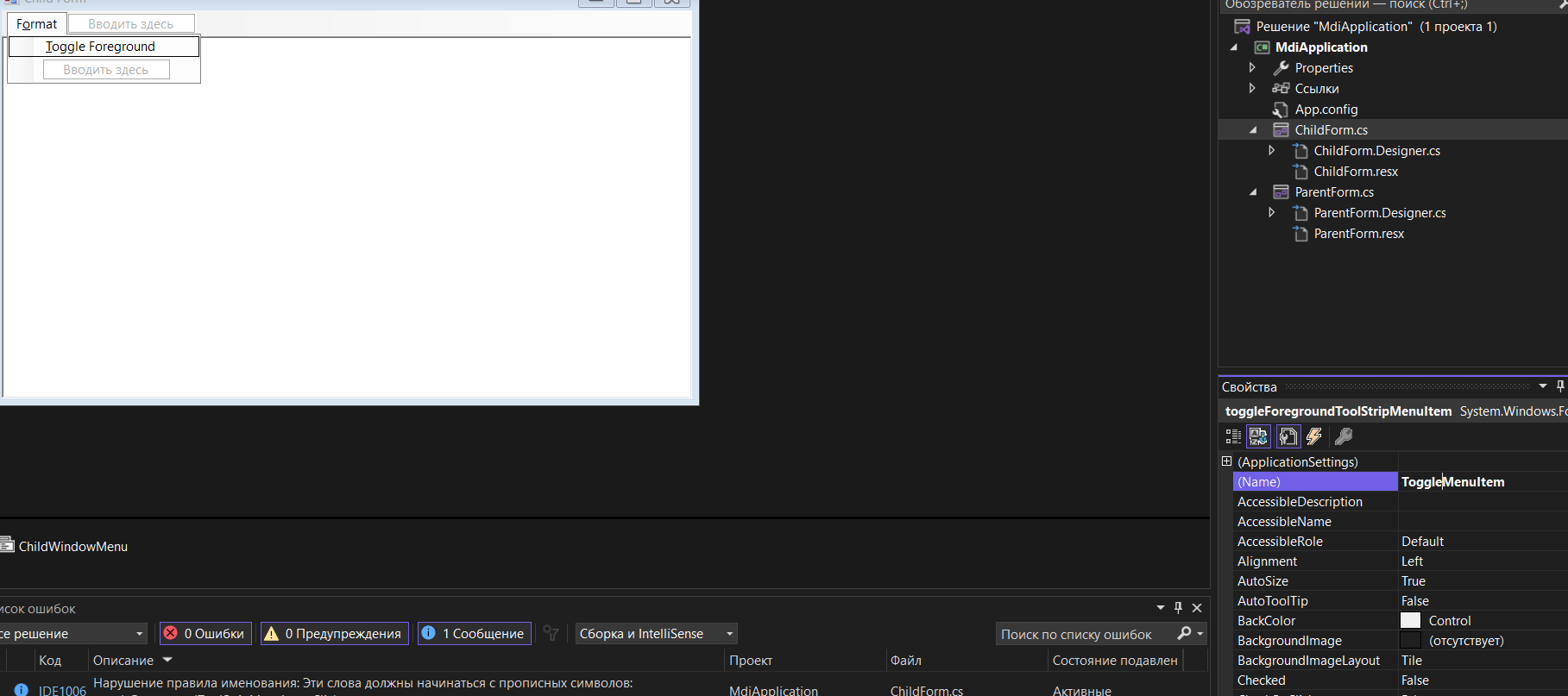
Выделите меню в верхней части формы и наберите текст **F&ormat**.



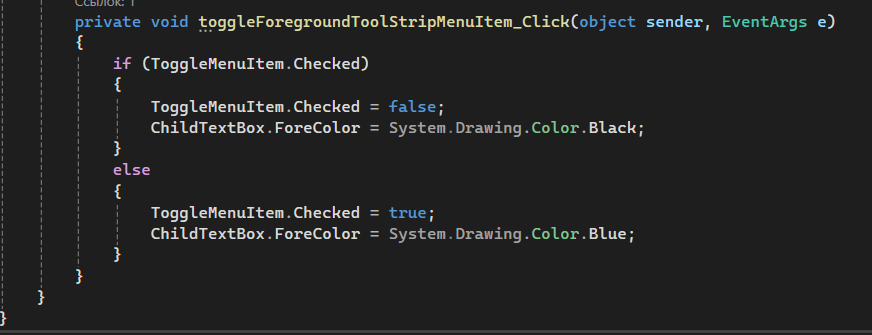
Для свойства **Name** пункта меню **Format** задайте значение **FormatMenuItem**, для свойства **MergeAction** установите значение **Insert**, а свойству **MergeIndex** – **1**. В это случае меню **Format** будет располагаться после **File** при объединении базового и дочерних меню.



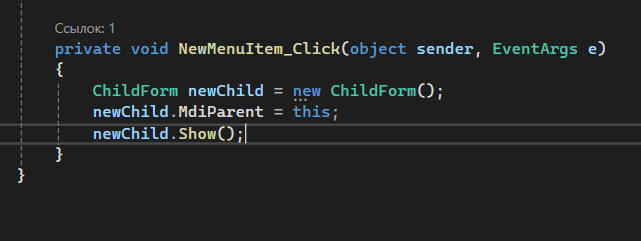
Выделите элемент, появившийся под элементом **Format**, и наберите текст **&Toggle Foregound**.



Для свойства **Name** пункта меню **Toggle Foreground** задайте значение **ToggleMenuItem**.



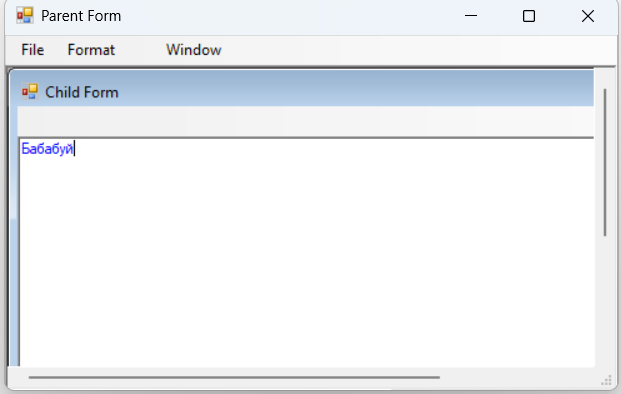
Дважды кликните левой кнопкой мыши по пункту меню **Toggle Foreground** и добавьте следующий код в обработчик событий **Click**.



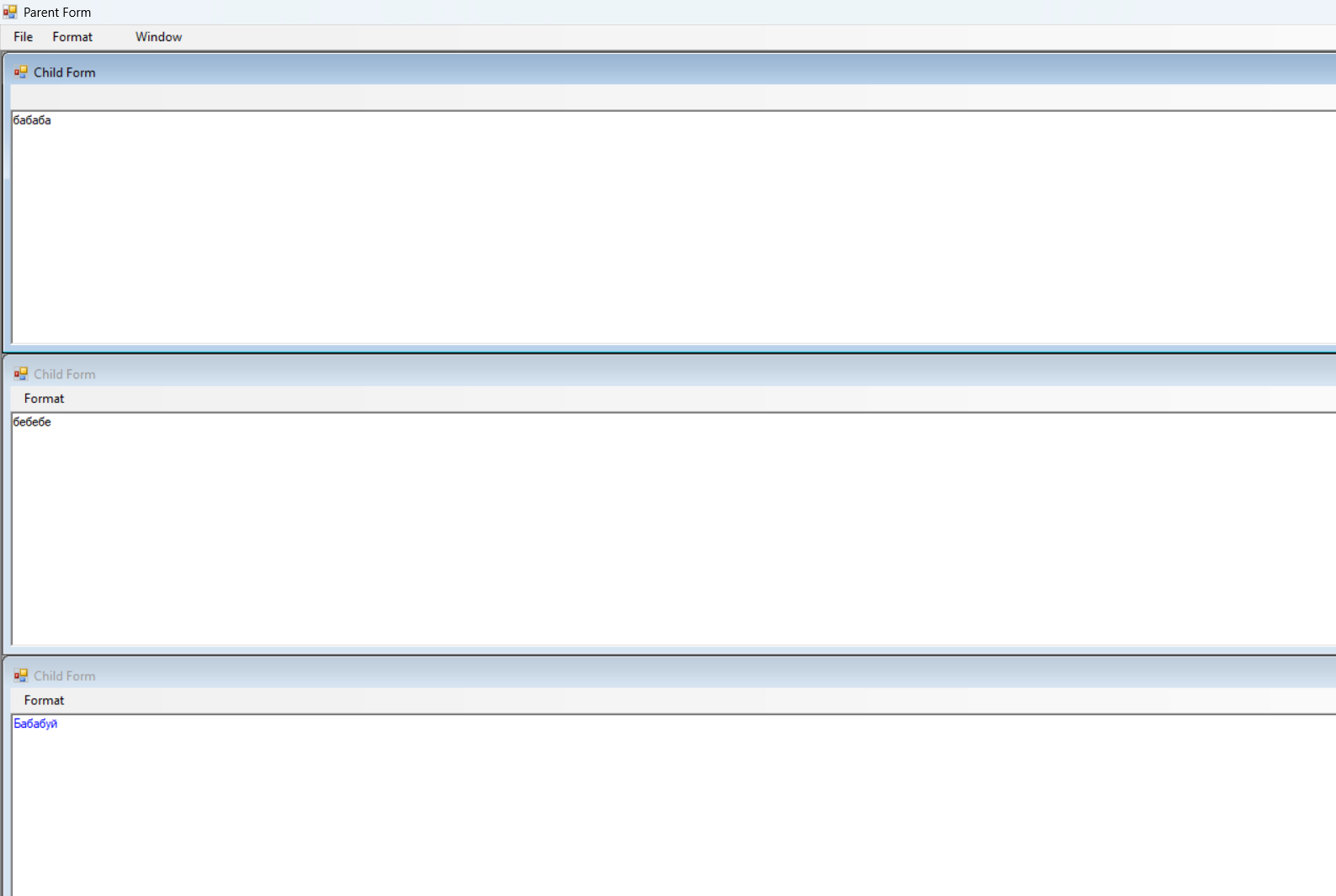
Дважды кликните левой кнопкой мыши по кнопке **New** в меню **File** для создания обработчика события **Click** и добавьте следующий код для обработчика события **Click** пункта меню **New**.



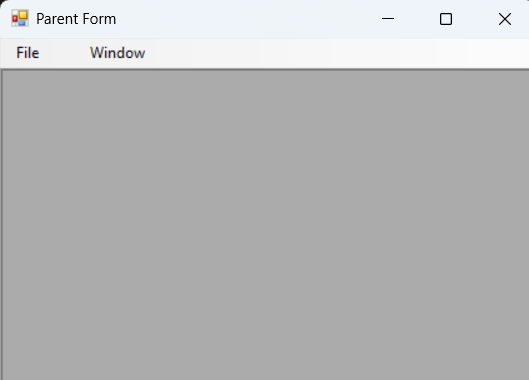
Когда появится родительская форма, выберите пункт меню **File | New**.



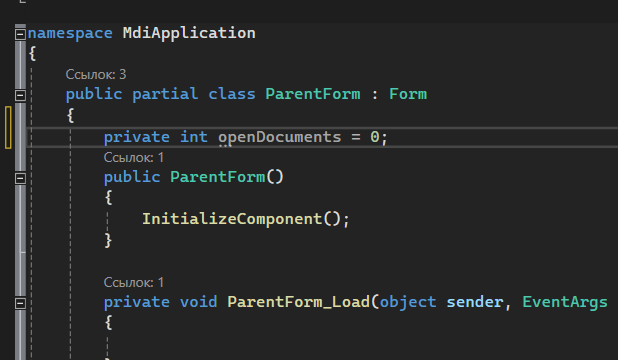
Наберите какой-нибудь текст в дочернем окне и воспользуйтесь пунктом меню **Format** для изменения цвета шрифта текста.



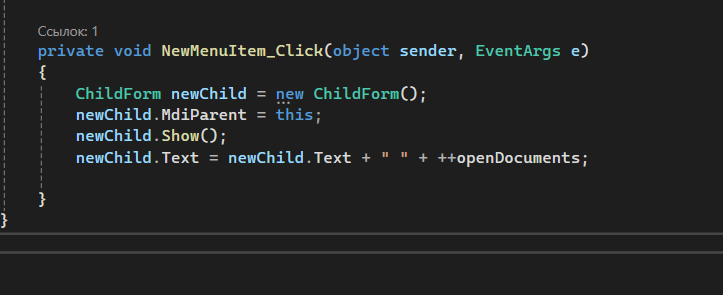
Выберите пункт меню **Window | Tile**. Обратите внимание на то, что дочерние окна выстраиваются в упорядоченном порядке.



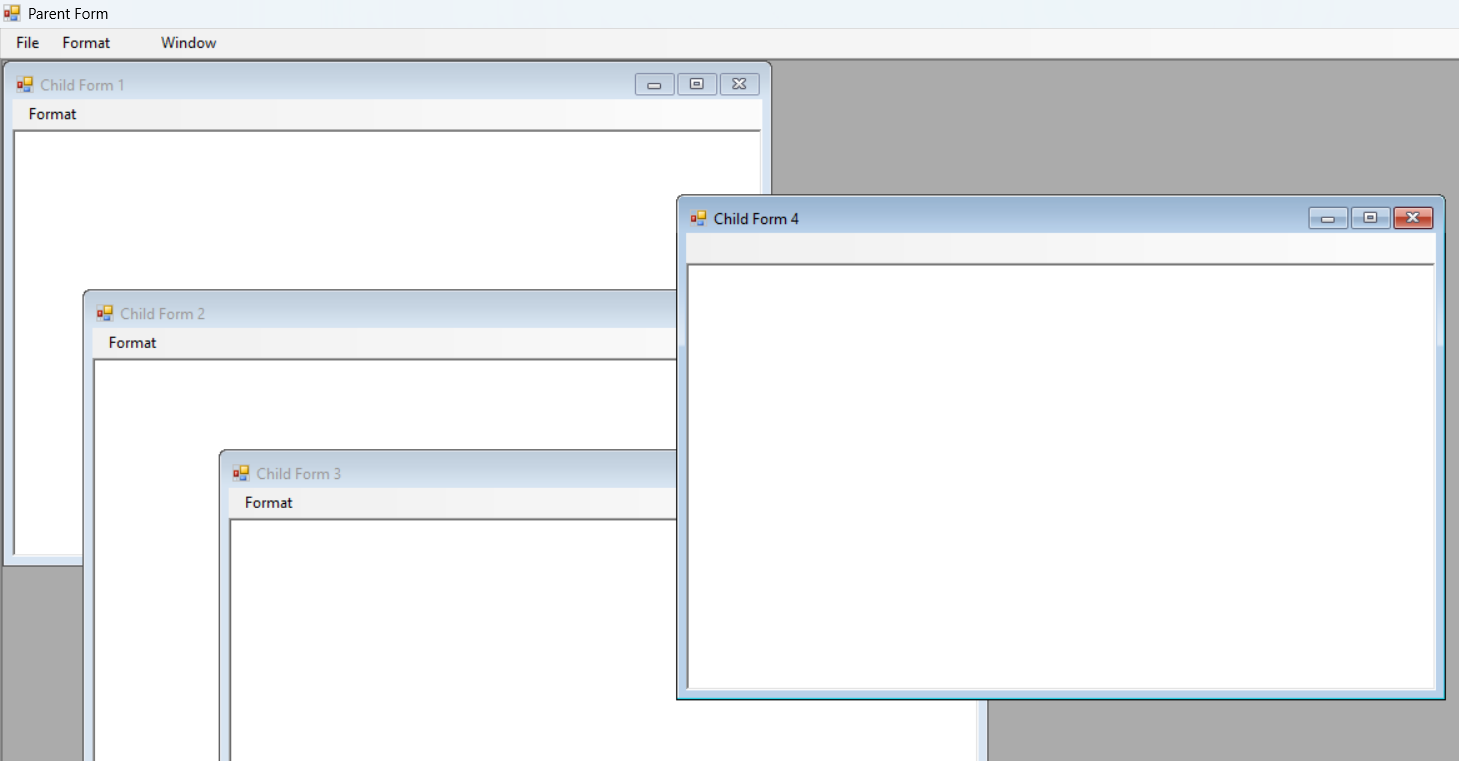
Обратите внимание на то, что когда закроется последнее дочернее окно, из меню родительской формы исчезнет пункт **Format**.



Откройте код родительской формы и в классе **ParentForm** объявите переменную **openDocuments**: private int openDocuments = 0;



К свойству **Text** дочерней формы добавьте счетчик числа открываемых документов (в коде обработчика события **Click** для пункта меню **New**): newChild.Text = newChild.Text+" "+ ++openDocuments;



Запустите приложение. Теперь заготовки новых документов содержат порядковый номер.