

Документ подписан простой электронной подписью  
Информация о владельце:  
ФИО: Таныгин Максим Олегович  
Должность: и.о. декана факультета фундаментальной и прикладной информатики  
Дата подписания: 21.09.2023 12:55:38  
Уникальный программный ключ:  
65ab2aa0d384efe8480e6a4c688eddbc475e411a

**МИНОБРНАУКИ РОССИИ**  
Федеральное государственное бюджетное образовательное  
учреждение высшего образования  
«Юго-Западный государственный университет»  
(ЮЗГУ)

*Кафедра программной инженерии*

УТВЕРЖДАЮ



Проректор по учебной работе  
О.Г. Локтионова  
\_\_\_\_\_ 2017 г.

**Создание приложения WPF  
в среде Visual Studio Community 2017**

Методические указания к лабораторным работам  
по курсу «Языки объектно-ориентированного  
программирования»

УДК 681.3

Составитель Е.И.Аникина

Рецензент

Кандидат технических наук, доцент кафедры программной инженерии *Н.Н. Бочанова*

**Создание приложения WPF в среде Visual Studio Community 2017:** методические указания к лабораторным работам по курсу «Языки объектно-ориентированного программирования»/Юго-Зап. гос. ун-т; сост. Е.И.Аникина. Курск, 2017. 20 с.

Содержит теоретические сведения и задания для выполнения лабораторных работ по изучению технологии создания и отладки приложений WPF в интегрированной среде разработки Visual Studio Community 2017.

Предназначено для студентов направлений подготовки, входящих в группы «Компьютерные и информационные науки» и «Информатика и вычислительная техника».

Текст печатается в авторской редакции.

Подписано в печать . Формат 60x84 1/16.  
Усл. печ. л. . Уч.-изд. л. . Тираж 100 экз. Заказ .  
Бесплатно.

Юго-Западный государственный университет  
305040, Курск, ул.50 лет Октября, 94.

## Настройка интегрированной среды разработки (IDE)

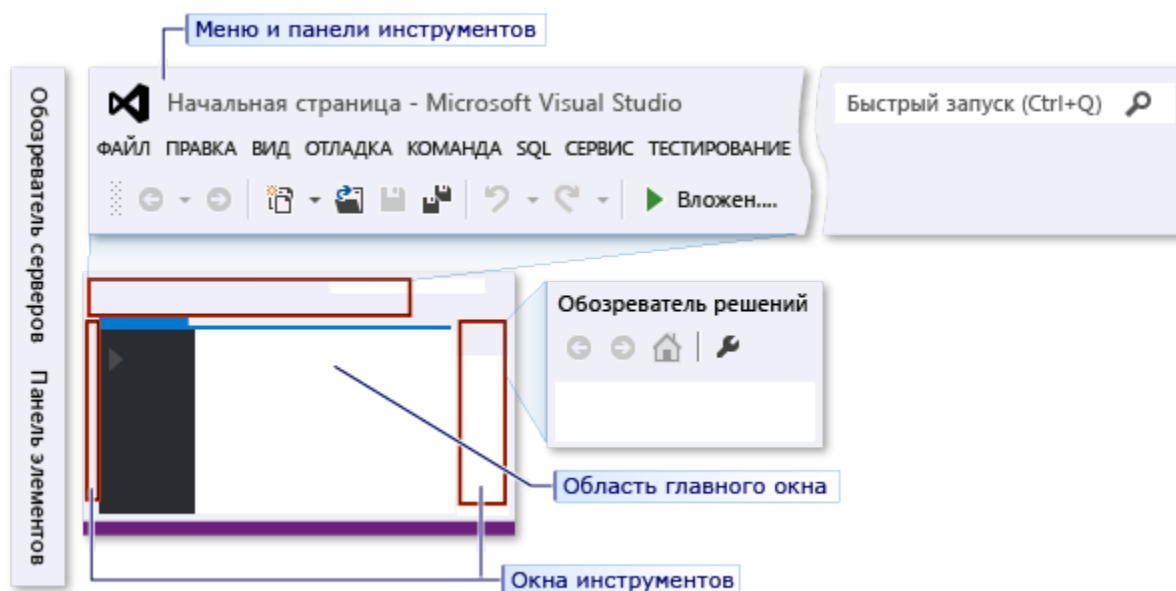
При первом запуске Visual Studio будет предложено войти в это приложение с использованием учетной записи Майкрософт: [Вход в Visual Studio](#). Необязательно входить сразу, можно сделать это позже.

После запуска Visual Studio далее необходимо выбрать сочетание предопределенных параметров интегрированной среды разработки. Каждое сочетание параметров предназначено для упрощения разработки приложений.

Это пошаговое руководство предполагает, что действуют **Обычные параметры разработки**, что соответствует минимальному объему настройки интегрированной среды разработки. Если вы уже выбрали C# или Visual Basic (подходит любой вариант), нет необходимости изменять свои настройки. Если вы хотите изменить настройки, можно воспользоваться **Мастером импорта и экспорта параметров**. См. раздел [Настройка параметров разработки в Visual Studio](#).

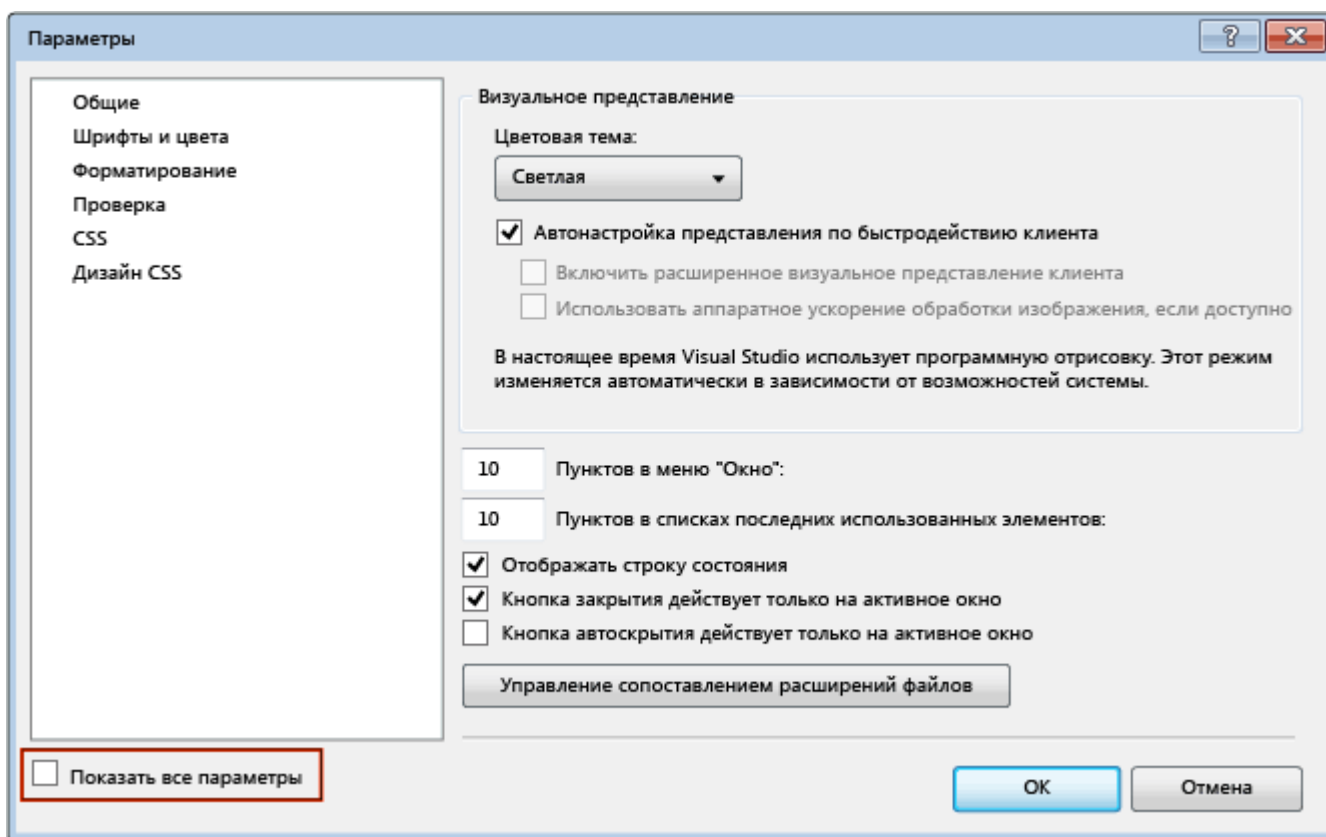
После открытия Visual Studio вы увидите окна инструментов, меню и панели инструментов, а также основную область окна. Окна инструментов прикреплены по левой и правой сторонам окна приложения, а в верхней его части располагаются панель быстрого запуска, строка меню, а также стандартная панель инструментов. В центре окна приложения располагается **Начальная страница**. При загрузке решения или проекта редакторы и конструкторы отображаются в области **Начальной страницы**. При создании приложения вы будете проводить большую часть времени в этой центральной области.

Рисунок 2. Интегрированная среда разработки Visual Studio



С помощью диалогового окна **Параметры** можно дополнительно настроить Visual Studio, например изменить в редакторе начертание и размер шрифта текста или изменить цветовую тему интегрированной среды разработки. В зависимости от применённого сочетания параметров, некоторые элементы в этом диалоговом окне могут не отображаться. Можно настроить, чтобы все возможные параметры отображались, выбрав флажок **"Показать все параметры"**.

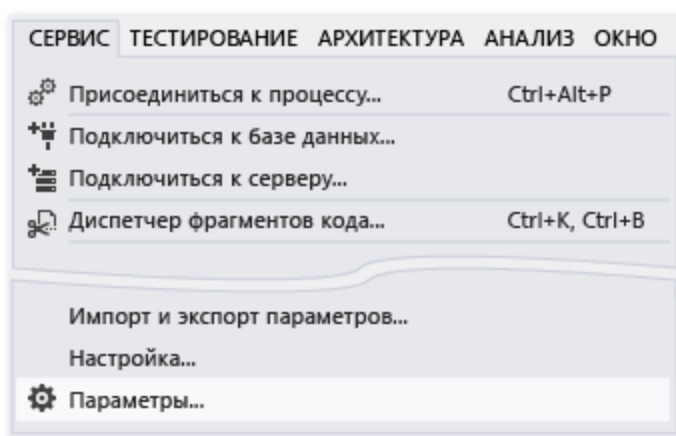
Рисунок 3. Диалоговое окно "Параметры"



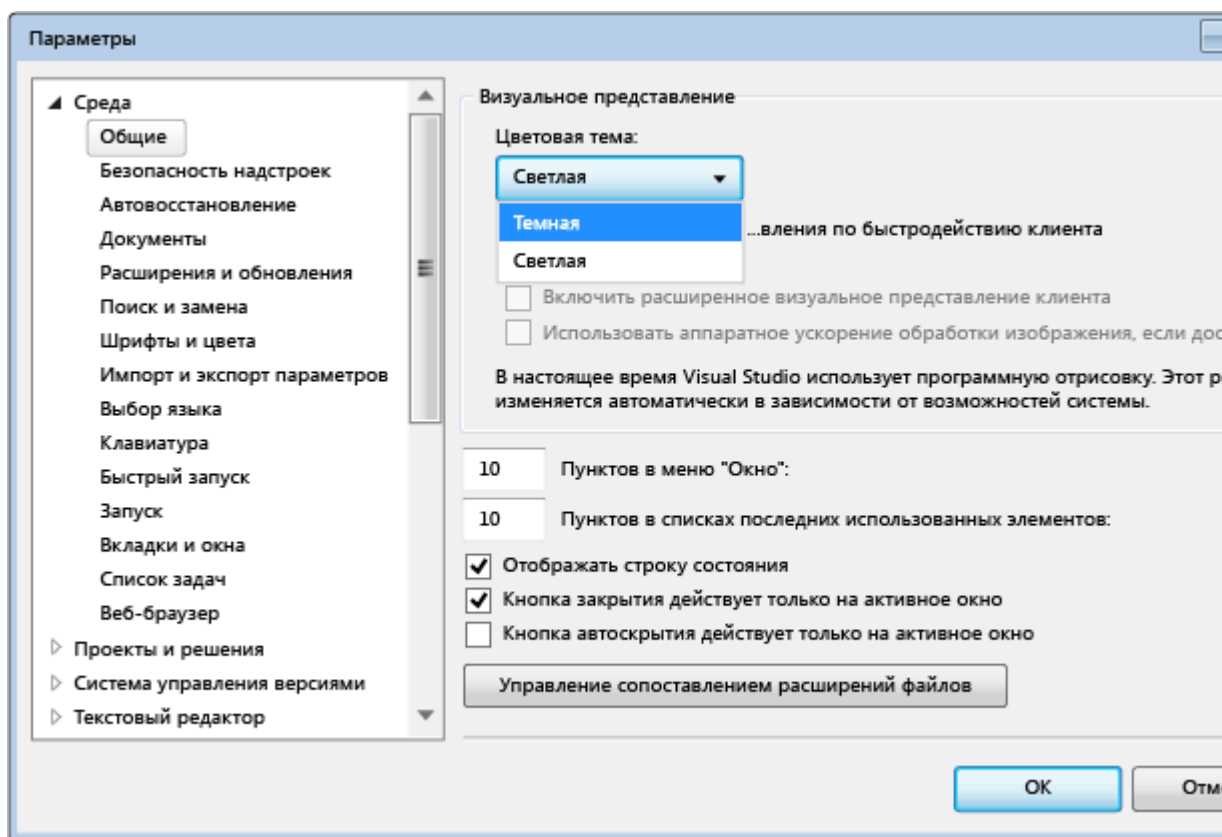
В этом примере вы поменяете цветовую тему интегрированной среды разработки со светлой на тёмную. При желании можно сразу перейти к созданию проекта.

Изменение цветовой темы интегрированной среды разработки

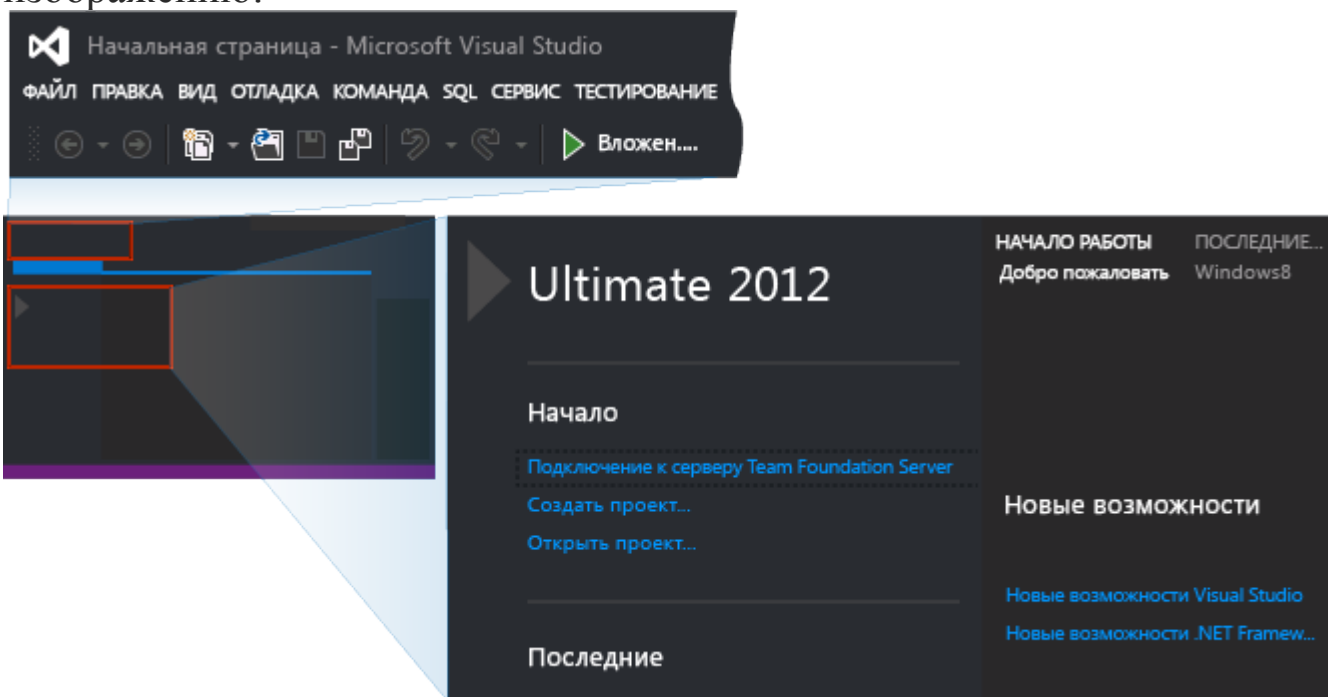
1. Откройте диалоговое окно **Параметры**, выбрав в меню **Сервис** сверху пункт **Параметры...**



2. Измените **Цветовую** тему на **Тёмную**, а затем щелкните **ОК**.



Цвета в Visual Studio будут соответствовать следующему изображению:



На изображениях в остальной части этого пошагового руководства используется светлая тема. Дополнительные сведения о настройке интегрированной среды разработки см. в разделе [Настройка параметров разработки в Visual Studio](#).

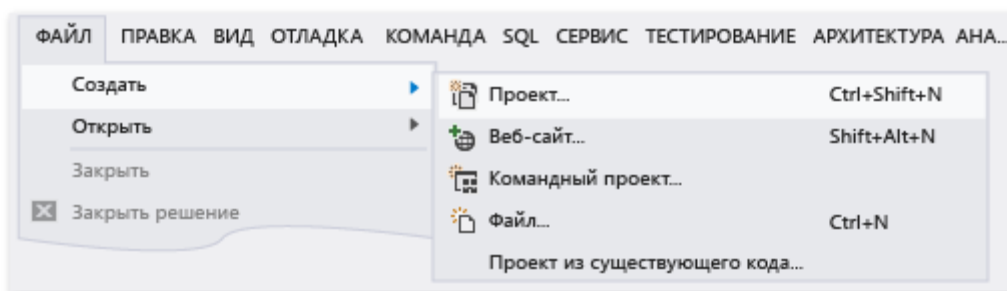
## Создание простого приложения WPF

### Создание проекта

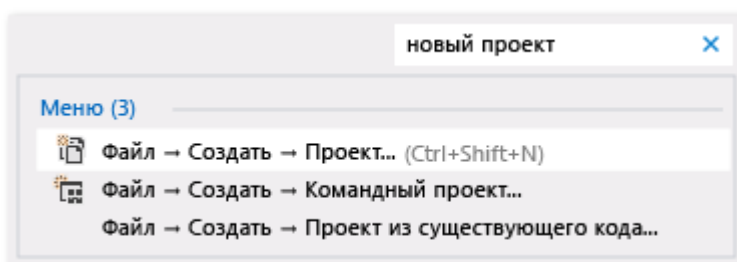
При создании приложения в Visual Studio сначала создаётся проект и решение. Этот пример демонстрирует создание решения Windows Presentation Foundation.

### Создание проекта WPF

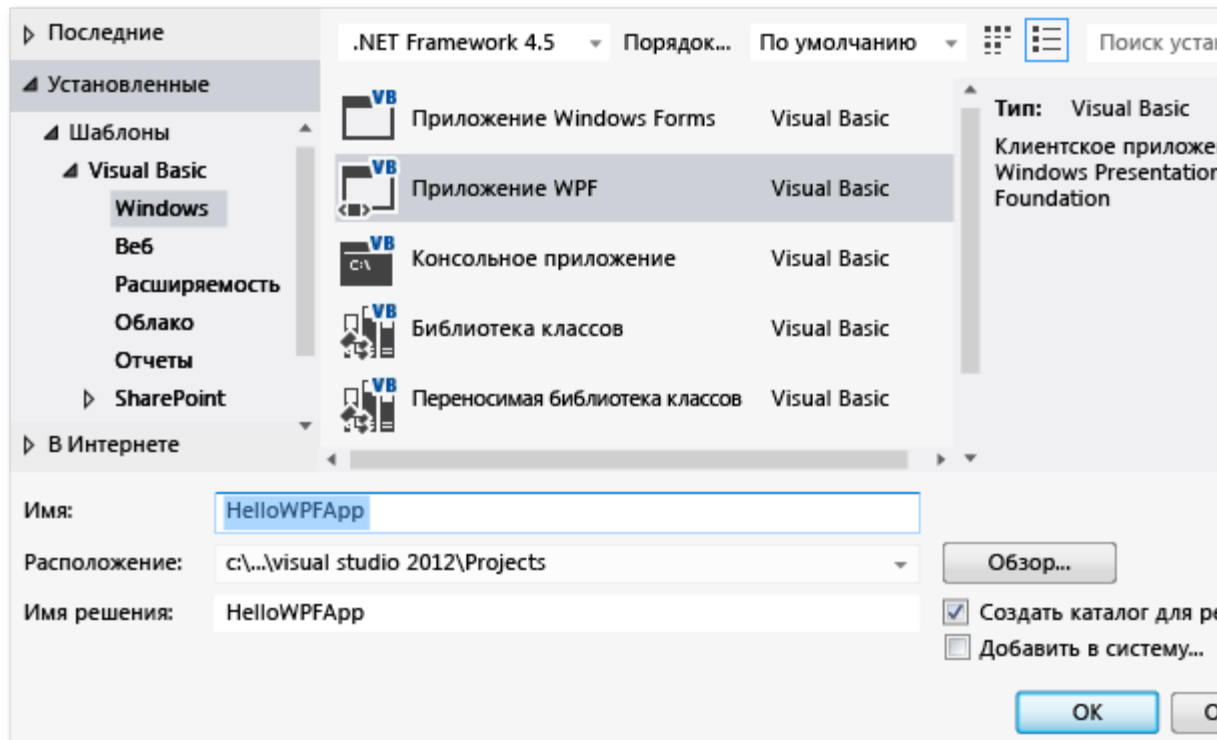
1. Создавать новый проект. В меню последовательно выберите пункты **Файл, Создать, Проект...**



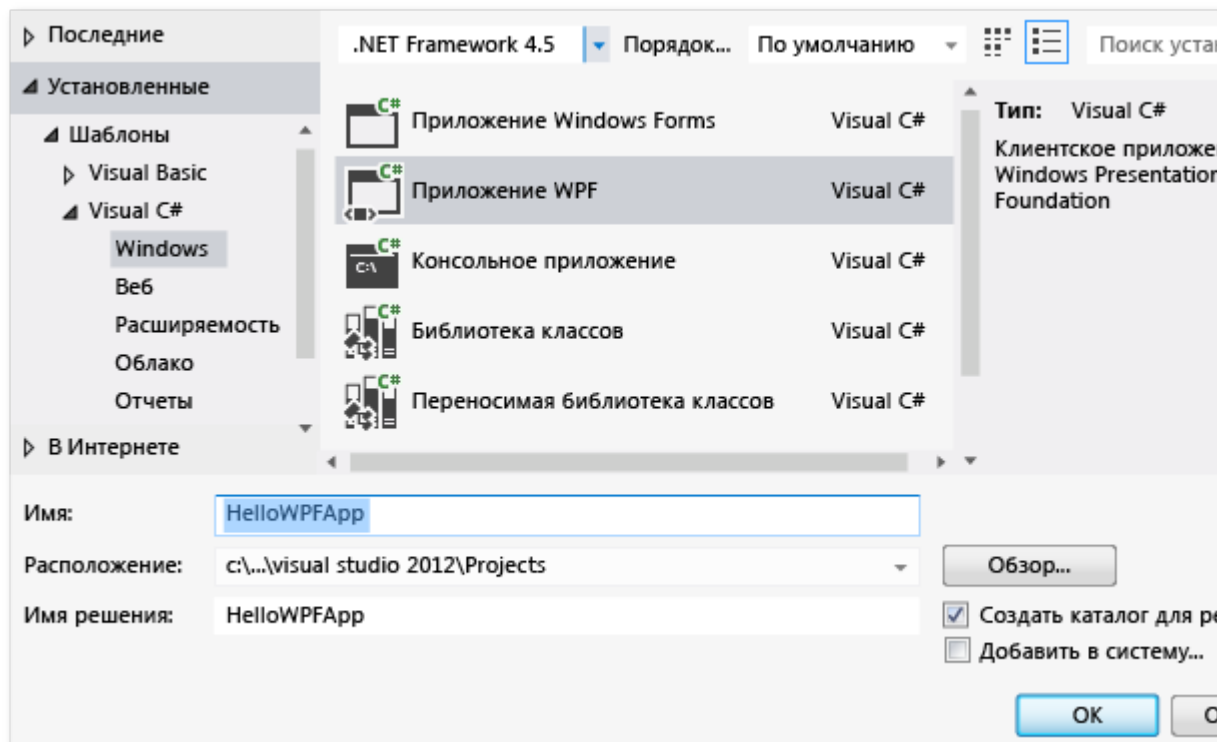
Кроме того, можно ввести "New Project" в окне **Быстрый запуск**, чтобы выполнить то же самое действие.



2. Выберите шаблон приложения WPF Visual Basic или Visual C#, воспользовавшись, например, путем по меню в левой области Установленные, Шаблоны, Visual C#, Windows, а затем выбрав «Приложение WPF» в средней области окна. В нижней части диалогового окна нового проекта назовите проект HelloWPFApp.



Or

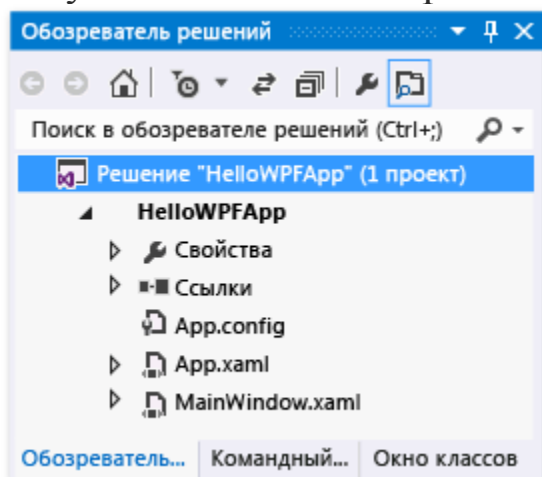


Visual Studio создает решение и проект HelloWPFApp, и **Обозреватель решений** показывает различные файлы. Конструктор WPF отображает представление конструирования и представление XAML файла



MainWindow.xaml в одном разделенном представлении. Сдвигая разделитель, можно делать любое из представлений больше или меньше. Можно выбрать для просмотра только визуальное представление или только представление XAML. (Дополнительные сведения см. в разделе [WPF Designer for Windows Forms Developers](#).) Следующие элементы отображаются в **Обозревателе решений**:

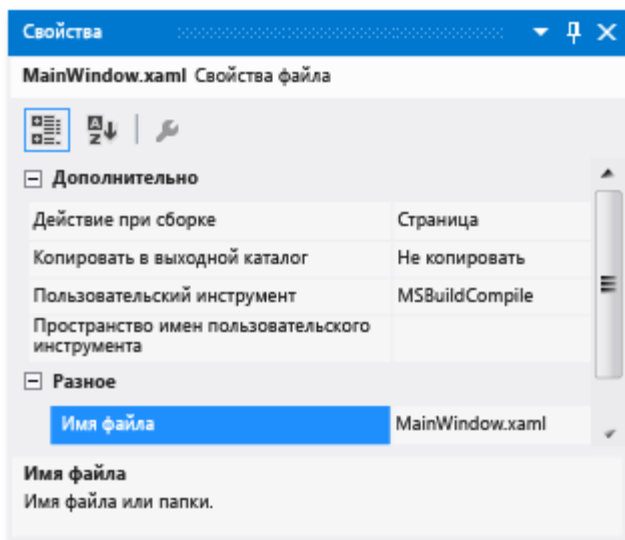
Рисунок 5. Элементы проекта



После создания проекта его можно настраивать. С помощью окна **Свойства** (в меню **Вид**) можно отображать и изменять параметры элементов проекта, элементов управления и других элементов в приложении. С помощью свойств проекта и страниц свойств можно отображать и изменять параметры проектов и решений

Изменение имени MainWindow.xaml

1. В следующей процедуре вы дадите MainWindow более конкретное имя. В **Обозревателе решений** выберите файл MainWindow.xaml. Должно отображаться окно **Свойства**, но если его нет, выберите в меню **Вид** пункт **Окно свойств**. Измените значение свойства **Имя файла** на **Greetings.xaml**.



**Обозреватель решений** показывает, что файл теперь называется `Greetings.xaml`, и если развернуть узел `MainWindow.xaml` (переместив курсор на узел и нажав клавишу стрелки вправо), то видно, что файл `MainWindow.xaml.vb` (или `MainWindow.xaml.cs`) теперь называется `Greetings.xaml.vb` (или `Greetings.xaml.cs`). Этот файл с текстом программы вложен в узел файла `.xaml`, что означает их тесную связь.

### Внимание

Это изменение вызовет ошибку; сведения по ее отладке и исправлению будут предоставлены позднее.

2. В **Обозревателе решений** откройте файл `Greetings.xaml` в представлении конструктора (путем нажатия клавиши ВВОД при выбранном узле) и выберите заголовок окна с помощью мыши.
3. В окне "**Свойства**" измените значение свойства "**Заголовок**" на `Greetings`.

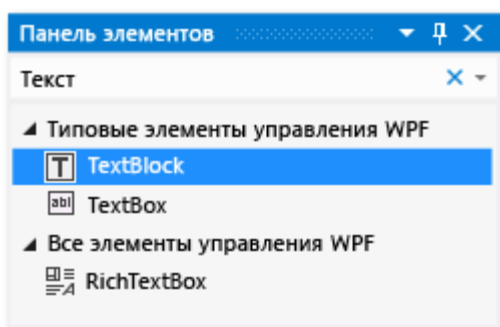
Заголовок окна для `MainWindow.xaml` теперь содержит текст `Greetings`.

Конструирование пользовательского интерфейса (ИП)

В приложение будет добавлено три типа элементов управления: элемент управления `TextBlock`, два элемента управления `RadioButton`, и элемент управления `Button`.

Добавление элемента управления `TextBlock`

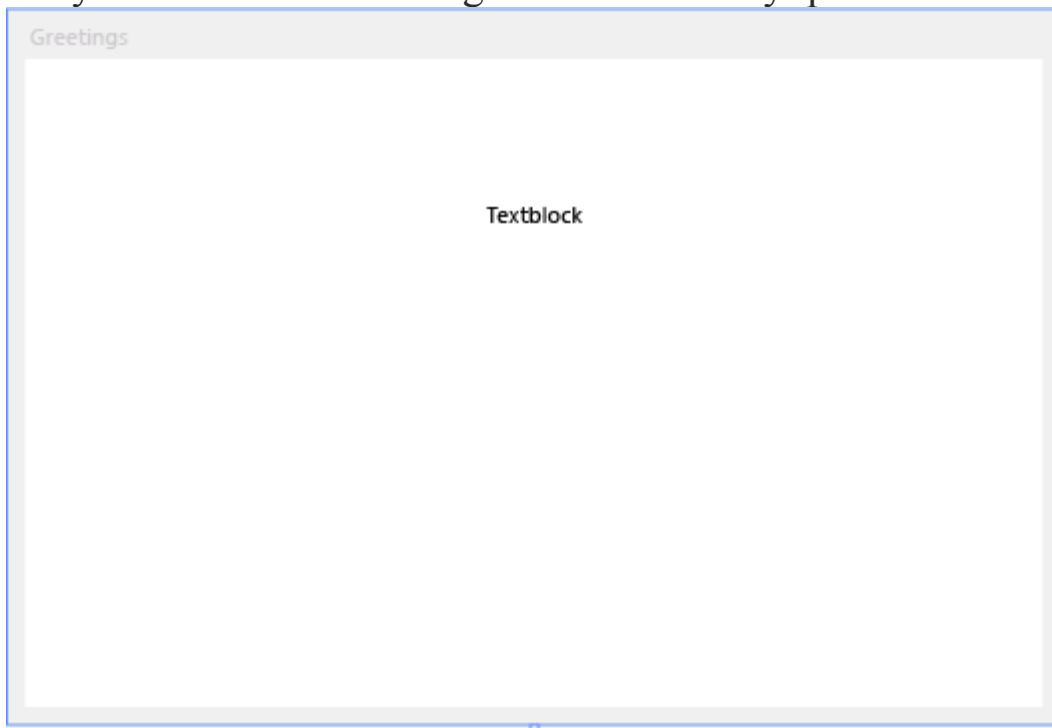
1. Откройте окно **Панель элементов**, выбрав в меню **Вид** пункт **Панель элементов**.
2. В "**Панели элементов**" найдите элемент управления `TextBlock`.



3. Добавьте элемент управления `TextBlock` на поверхность разработки, выбрав элемент управления `TextBlock` и перетащив его на поверхность. Отцентрируйте этот элемент в верхней части окна.

Ваше окно должно быть похоже на изображённое на следующем рисунке:

Рисунок 7. Окно "Greetings" с элементом управления `TextBlock`



Разметка XAML должна выглядеть приблизительно так, как показано ниже:

```
<TextBlock HorizontalAlignment="Center" TextWrapping="Wrap"
VerticalAlignment="Center" RenderTransformOrigin="4.08,2.312"
Margin="237,57,221,238"><Run
Text="TextBlock"/><InlineUIContainer><TextBlock
TextWrapping="Wrap" Text="TextBlock"/>
```

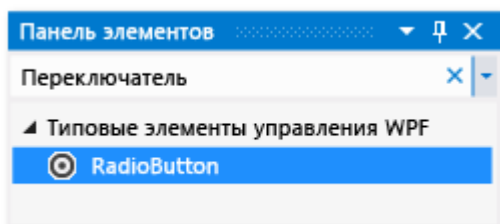
Настройка текста в текстовом блоке

1. В представлении XAML найдите разметку для TextBlock и измените атрибут Text: **Text="Select a message option and then choose the Display button."**
2. Если в элементе управления TextBlock не отображается весь текст в представлении конструирования, увеличьте элемент TextBlock (используя маркеры на краях).
3. Сохраните изменения, нажав клавиши Ctrl-s или с помощью меню **Файл**.

Далее вы добавите в форму два элемента управления [RadioButton](#).

Добавление переключателей

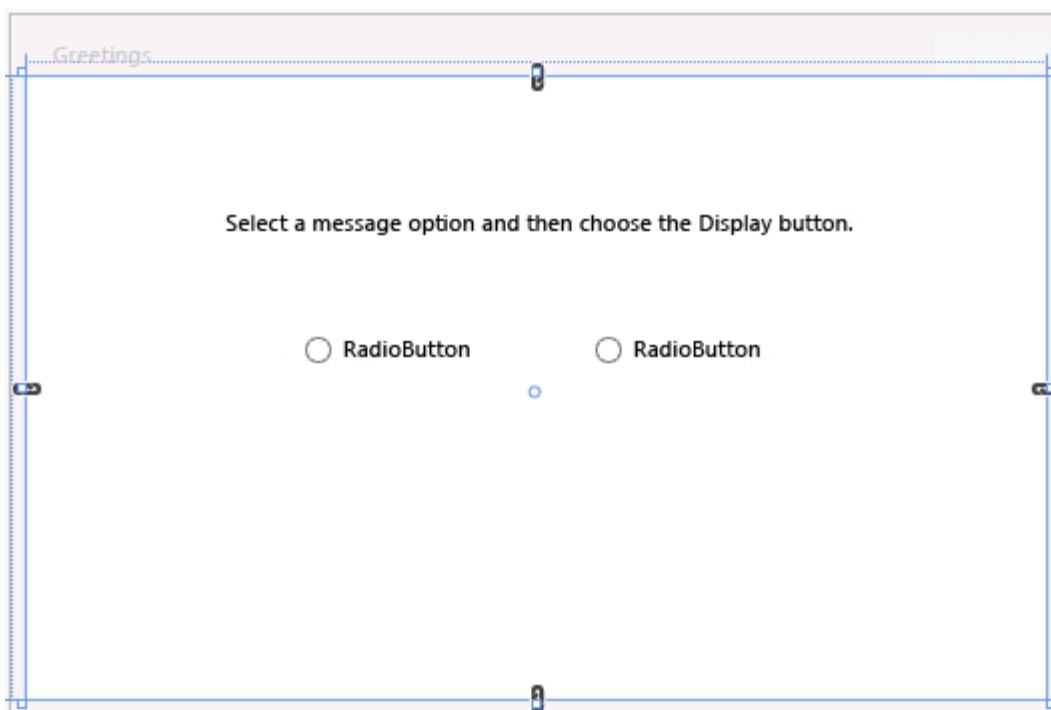
1. В "**Панели элементов**" найдите элемент управления RadioButton.



2. Добавьте два элемента управления RadioButton на поверхность разработки, выбрав элемент RadioButton и дважды перетащив его на поверхность, и переместите кнопки (выбрав их и используя клавиши со стрелками), чтобы кнопки отображались рядом и под элементом управления TextBlock.

Окно должно выглядеть следующим образом:

Рисунок 8. Переключатели (RadioButton) в окне "Greetings".



3. В окне **Свойства** для левого элемента управления `RadioButton` измените свойство **Имя** (свойство в верхней части окна **Свойства**), задав ему значение `RadioButton1`. Убедитесь, что выбран элемент управления `RadioButton` и в форме нет фоновой сетки; поле «Тип» в окне свойств под полем «Имя» должно иметь значение «`RadioButton`».
4. В окне **Свойства** для правого элемента управления `RadioButton` измените свойство **Имя** на `RadioButton2` и сохраните изменения, нажав клавиши `Ctrl-s` или с помощью меню **Файл**. Перед изменением и сохранением убедитесь, что выбран элемент управления `RadioButton`.

Теперь можно добавить отображаемый текст для каждого элемента управления `RadioButton`. Следующая процедура обновляет свойство "**Content**" (содержимое) для элемента управления `RadioButton`.

Добавление отображаемого текста для каждого переключателя

1. Откройте контекстное меню элемента управления `RadioButton1` (выбрав его и нажав правую кнопку мыши), выберите команду **Изменить текст**, а затем введите `Hello`.
2. Откройте контекстное меню элемента управления `RadioButton2` (выбрав его и нажав правую кнопку

мышью), выберите команду **Изменить текст**, а затем введите **Goodbye**.

Последний элемент пользовательского интерфейса, который необходимо добавить, — это элемент управления Button (Кнопка).

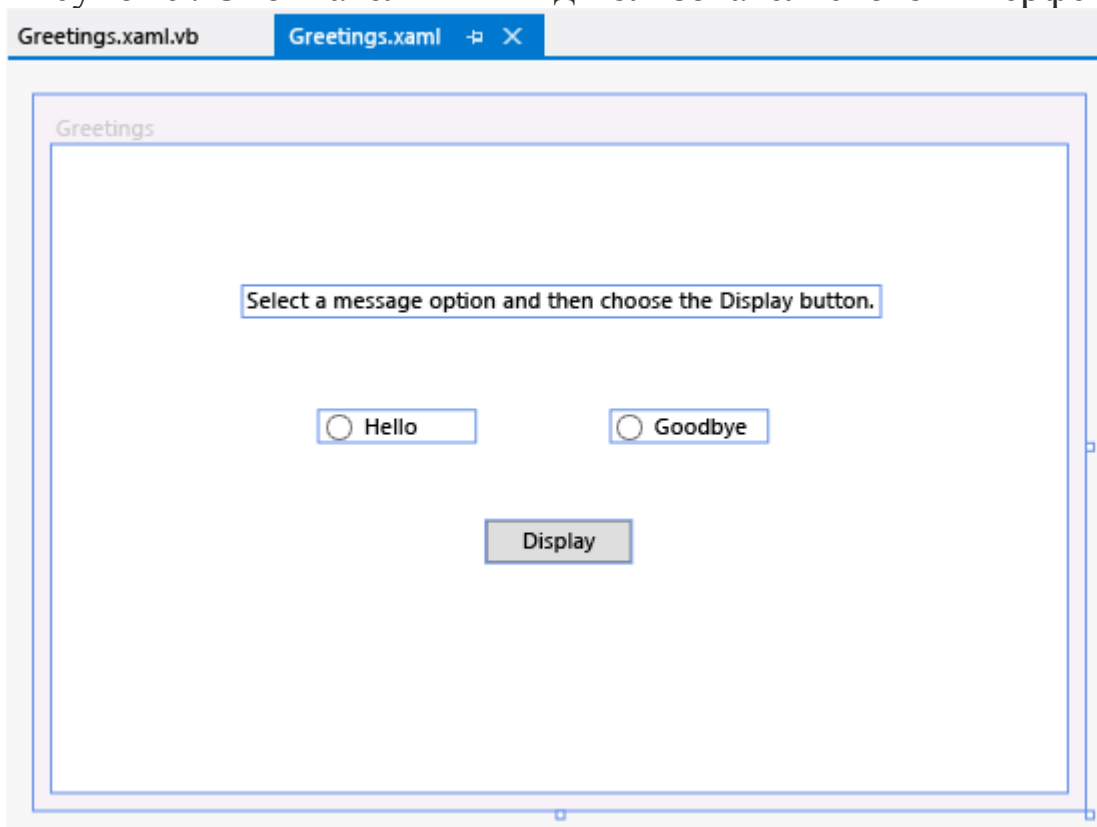
### Добавление элемента управления Button

1. В Панели элементов найдите элемент управления Button и добавьте его на поверхность разработки под элементами управления RadioButton, выбрав кнопку и перетащив ее на форму в представлении конструирования.
2. В представлении XAML измените значение свойства **Содержимое** элемента управления «Кнопка» с **Content="Button"** на **Content="Display"** и сохраните изменения (используя Ctrl-s или меню **Файл**).

Разметка должна быть похожа на следующий пример: `<Button Content="Display" HorizontalAlignment="Left" VerticalAlignment="Top" Width="75" Margin="215,204,0,0"/>`

Ваше окно должно быть похоже на следующий рисунок.

Рисунок 9. Окончательный вид пользовательского интерфейса



## Добавление кода к кнопке Display

После запуска приложения окно сообщения появится только тогда, когда пользователь сначала выберет переключатель, а затем нажмет кнопку **Display**. Если выбран переключатель "Hello" будет отображено одно окно сообщения, а если выбран переключатель "Goodbye" — другое. Для создания этой функциональности вы добавите код для события Button\_Click в файл Greetings.xaml.vb (или Greetings.xaml.cs).

## Добавление кода для отображения окон сообщений

1. В области конструктора дважды щёлкните по кнопке **Display**.

Откроется файл Greetings.xaml.vb (или Greetings.xaml.cs) с курсором, помещённым в обработчик события Button\_Click. Также можно добавить обработчик событий щелчка следующим образом (если во вставленном коде какие-либо имена подчеркнуты красной волнистой линией, то, возможно, не были выбраны и переименованы элементы управления RadioButton на поверхности разработки):

Для Visual Basic обработчик событий должен выглядеть следующим образом:

VB

```
Private Sub Button_Click_1(sender As Object, e As RoutedEventArgs)
```

```
End Sub
```

Для Visual C# обработчик события должен выглядеть следующим образом:

C#

```
private void Button_Click_1(object sender, RoutedEventArgs e)
{
}
}
```

2. Для Visual Basic введите следующий код:

VB

```
If RadioButton1.IsChecked = True Then
    MessageBox.Show("Hello.")
Else RadioButton2.IsChecked = True
    MessageBox.Show("Goodbye.")
End If
```

Для Visual C# введите следующий код:

```
if (RadioButton1.IsChecked == true)
{
    MessageBox.Show("Hello.");
}
else
{
    RadioButton2.IsChecked = true;
    MessageBox.Show("Goodbye.");
}
```

3. Сохраните приложение.

### Отладка и тестирование приложения

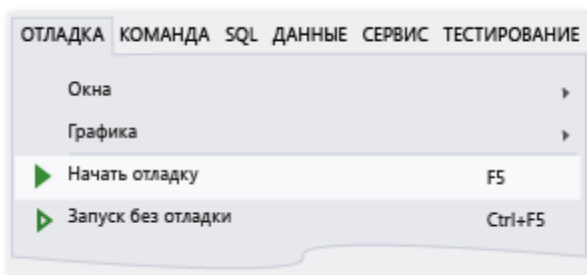
Далее вы проведёте отладку приложения, выполнив поиск ошибок и убедившись, что оба окна сообщения отображаются правильно. Приведенные ниже инструкции описывают, как выполнить сборку и запустить отладчик (дополнительные сведения см. в разделах [Построение приложения WPF](#) и [Отладка WPF](#)).

### Поиск и исправление ошибок

На этом шаге вам предстоит найти ошибку, которую мы намеренно допустили ранее, изменив имя файла XAML главного окна.

### Начало отладки и поиск ошибки

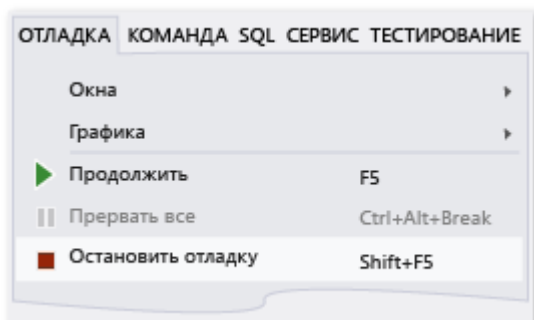
1. Запустите отладчик, выбрав **Отладка**, затем **Начать отладку**.





Появится диалоговое окно, показывающее, что произошло исключение `IOException` "Не удалось обнаружить ресурс 'mainwindow.xaml'".

2. Нажмите кнопку **ОК**, а затем остановите отладчик.



Файл `Mainwindow.xaml` был переименован в `Greetings.xaml` в начале этого пошагового руководства, но код по-прежнему ссылается на файл `Mainwindow.xaml` как на начальный универсальный код ресурса (URI) для приложения, поэтому проект не может быть запущен.

Задание `Greetings.xaml` в качестве начального универсального кода ресурса (URI)

1. В **Обозревателе решений** откройте файл `App.xaml` (в проекте C#) или файл `Application.xaml` (в проекте Visual Basic) в представлении XAML (его невозможно открыть в представлении конструирования), выбрав файл и нажав клавишу ВВОД или дважды щелкнув его.
2. Измените `StartupUri="Mainwindow.xaml"` на `StartupUri="Greetings.xaml"` и сохраните изменения, нажав `Ctrl-s`.

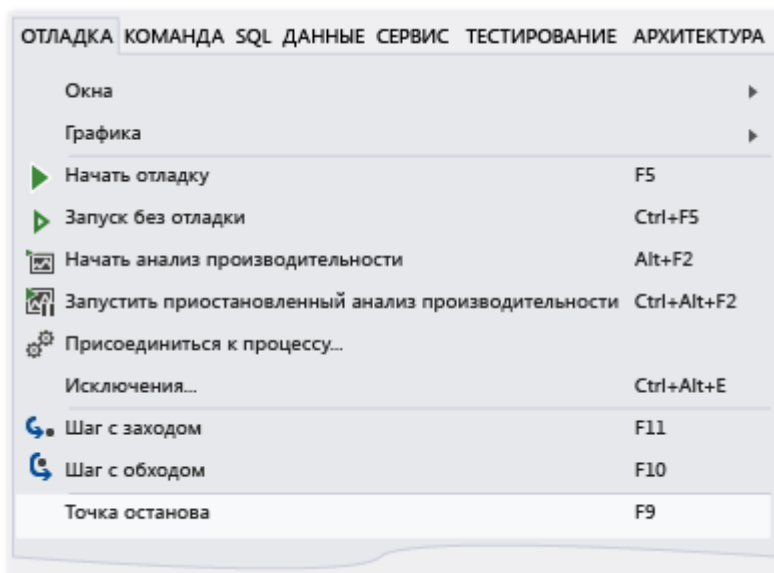
Запустите отладчик снова (клавишей F5). Должно появиться окно `Greetings` приложения.

Отладка с точками останова

Путём добавления точек останова, можно тестировать код во время отладки. Для добавления точки останова можно выбрать в меню **Отладка** пункт **Точка останова** или щелкнуть в левой области редактора рядом со строкой кода, на которой требуется приостановить выполнение.

Добавление точек останова

1. Откройте файл Greetings.xaml.vb или Greetings.xaml.cs и выделите строку: `MessageBox.Show("Hello.")` следующей строкой.
2. Добавьте точку останова, выбрав меню Отладка, затем —Точка останова.



Красный кружок появится рядом с строкой кода с краю левого поля окна редактора.

3. Выделите строку: `MessageBox.Show("Goodbye.")` следующей строкой.
4. Нажмите клавишу F9, чтобы добавить точку останова, затем нажмите клавишу F5, чтобы начать отладку.
5. В окне "Greetings" (Приветствие) выберите переключатель "Hello" (Привет), а затем нажмите кнопку "Display" (Показать).  
Строка `MessageBox.Show("Hello.")` выделяется желтым цветом. В нижней части интегрированной среды разработки окна «Видимые», «Локальные» и «Контрольные значения» закреплены вместе на левой стороне, а окна «Стек вызовов», «Точки останова», «Команда», «Интерпретация» и окно вывода закреплены вместе на правой стороне.
6. В строке меню выберите "Отладка" – "Шаг с выходом".

Приложение возобновит выполнение, и появится окно сообщения со словом «Hello» (Привет).

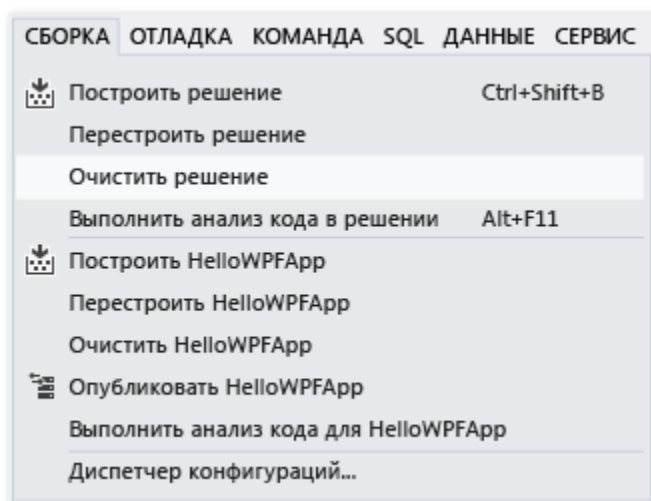
7. Нажмите кнопку **OK** в окне сообщения, чтобы закрыть его.
8. В окне **"Greetings"** (Приветствие) выберите переключатель **"Goodbye"** (До свидания), а затем нажмите кнопку **"Display"** (Показать).  
Строка **MessageBox.Show("Goodbye.")** выделяется желтым цветом.
9. Нажмите клавишу F5, чтобы продолжить отладку. Когда появится окно сообщения, нажмите в нем кнопку **OK**, чтобы закрыть его.
10. Нажмите комбинацию клавиш SHIFT + F5 (нажмите сначала клавишу SHIFT и, удерживая ее нажатой, нажмите клавишу F5), чтобы остановить отладку.
11. В строке меню выберите **"Отладка" – "Выключить все точки останова"**.

### Сборка окончательной версии приложения

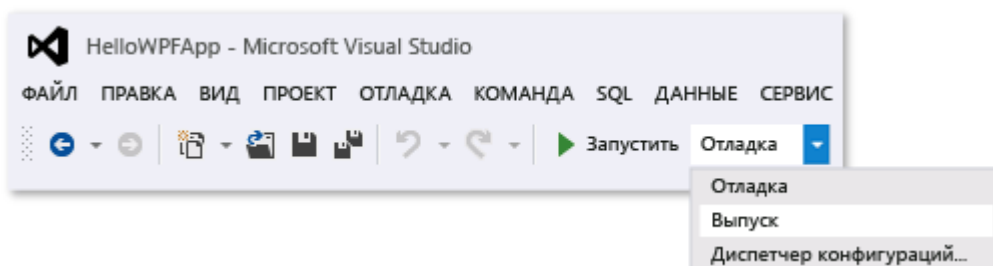
Теперь, когда вы убедились, что всё работает, можно подготовить рабочее (release) построение приложения.

### Очистка файлов решения и сборка окончательной версии

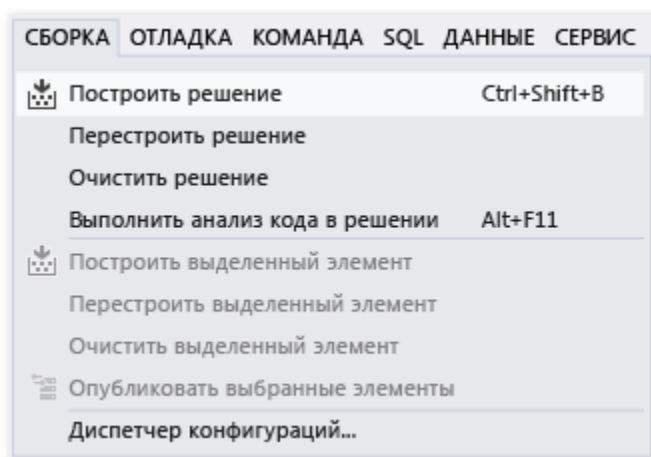
1. Выберите в главном меню **Сборка** и **Очистить решение** для удаления промежуточных файлов и выходных файлов, которые были созданы в ходе предыдущих сборок. Это не является обязательным, но очищает результаты отладочной сборки.



- Измените конфигурацию сборки для HelloWPFApp с **Отладка** на **Выпуск** с помощью раскрывающегося списка на панели инструментов (сейчас это «Отладка»).



- Постройте решение, выбрав **Сборка**, затем **Собрать решение** или нажмите клавишу F6.



Поздравляем с завершением выполнения этого пошагового руководства! Построенный EXE-файл находится в каталоге решения и проекта (... \HelloWPFApp\HelloWPFApp\bin\Release\). Чтобы изучить больше примеров, см. раздел [Примеры Visual Studio](#).