ОГБПОУ «ТОМСКИЙ ТЕХНИКУМ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ»

Утверждаю

Заместитель директора по УМР

\_\_\_\_\_\_\_\_\_\_\_\_/Е.А. Родзик

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_20\_\_\_г.

Методические рекомендации по выполнению   
учебной-практической работы №9

учебной практики   
УП 05. Программирование

Тема: «Реализация калькулятора с помощью UWP»

г. Томск – 2020 г

РАССМОТРЕННО

на заседании ПЦК

«Информационные системы и программирование»

протокол №\_\_\_\_\_\_

от «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_20\_\_\_г.

Председатель ПЦК

\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Фунтиков М.Н.Рекомендации разработаны на основе Федерального государственного образовательного стандарта (далее – ФГОС) по специальностям среднего профессионального образования (далее СПО) 09.02.07 – «Информационные системы и программирование» и в соответствии с примерной основной образовательной программой.

Организация-разработчик:

ОГБОУ СПО «Томский техникум информационных технологий»

Разработчики:

\_\_\_\_\_\_\_\_\_\_\_\_\_/ Сидиков И.Д.

\_\_\_\_\_\_\_\_\_\_\_\_\_/ Владимировна А. В.

**СОДЕРЖАНИЕ**

[1. Общая характеристика учебной-практической работы №9 4](#_Toc22557890)

[2. Краткие теоретические сведения 5](#_Toc22557891)

[3. Задания для выполнения 18](#_Toc22557892)

[4. Индивидуальные задания 19](#_Toc22557893)

[5. Контрольные вопросы 20](#_Toc22557894)

[6. Рекомендованная литература 21](#_Toc22557895)

[ПРИЛОЖЕНИЕ А 22](#_Toc22557896)

[ПРИЛОЖЕНИЕ Б 23](#_Toc22557897)

# ОБЩАЯ ХАРАКТЕРИСТИКА УЧЕБНОЙ-ПРАКТИЧЕСКОЙ РАБОТЫ №8

**Тема:** «Реализация калькулятора с помощью UWP».

**Цель работы:** закрепление навыков по созданию пользовательского интерфейса и разработки ПО на базе UWP на языке программирования C#.

**Проверяемые компетенции:**

ОК 1, ОК 2, ОК 4, ОК 9, ПК11.2.

**Инструкция по выполнению:**

1. Ознакомьтесь с теоретическим материалом, необходимым для выполнения практической работы.
2. Выполните предложенное практическое задание.
3. Оформите отчет по учебной практике (шаблон представлен в приложении)
4. Отправьте отчет на проверку.

**Время выполнения заданий:** 6 часов.

**Критерии оценки:**

|  |  |  |
| --- | --- | --- |
| **№** | **Критерий** | **Баллы** |
| 1 | Отчет соответствует предложенному шаблону | 1 |
| 2 | Задание выполнено правильно | 2 |
| 3 | Представлен код графического интерфейса и программы | 4 |
| 4 | Составлена блок схема к программе | 2 |
| 5 | Программа работает правильно без сбоев | 2 |
| 6 | Студент ответил на контрольные вопросы по лабораторной работе | 2 |
| 7 | Оформление отчета соответствует требованиям (шрифт, поля, отступы, интервалы, оформление рисунков, автоматическое оглавление) | 1 |
| 8 | Своевременность выполнения задания | 1 |
| Итого | | 15 |

**Перевод в пятибалльную систему оценивания:**

«отлично» - 13 - 15 баллов

«хорошо» - 10 -12 баллов

«удовлетворительно» - 7-9

«неудовлетворительно» <7 баллов

# КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

## Что такое Universal Windows Platform

UWP (Universal Windows Platform) представляет собой унифицированную платформу для создания и запуска приложений в Windows 10 и Windows 10 Mobile.

UWP стала результатом фолюции более ранних технологий. Так, с выходом Windows 8 была внедрена новая архитектурная платформа для приложений - Windows Runtime (WinRT), которая позволяла запускать приложения в так называемом режиме Modern (Metro) на десктопах, планшетах. Затем с выходом Windows 8.1 и Windows Phone 8.1 эта технология получила развитие - появились "универсальные приложения", которые можно было запускать сразу Windows 8.1 и WP8.1. И в июле 2015 года официально вышла новая ОС Windows 10. Она использует платформу UWP, которая представляет собой развитие Windows Runtime.

Как подсказывает название платформы, она является универсальной - универсальной для всех устройств экосистемы Windows 10. А это обычные дестопы, планшеты, мобильные устройства, устройства IoT (интернет вещей), Xbox, устройства Surface Hub. И приложение UWP может одинаково работать на всех этих платформах, если на них установлена Windows 10.

Программирование под UWP несет ряд преимуществ:

* Широта распространения. На текущий момент (апрель 2017) Windows 10 установлена уже более чем на 400 миллионах устройств. На десктопах Windows 10 уже опередила Windows 8/8.1.
* Поддержка широкого круга устройств. Десктопы, планшеты, смартфоны, большие планшеты Surface Hub, различные IoT-устройства, в перспективе устройства виртуальной реальности HoloLens - круг устрйоств, на которых может работать Windows 10 действительно широк.
* Поддержка разных языков и технологий программирования. UWP-приложения можно создавать с помощью таких языков, как Visual C++, C#, Visual Basic, JavaScript. В качестве технологии для создания графического интерфейса Visual C++, C# и Visual Basic используют XAML, JavaScript применяет HTML. Кроме того, С++ может вместо XAML использовать DirectX. То есть достаточно распространенные и и знакомые многим технологии.
* Магазин приложений и удобство распространения. Windows Store представляет собой прекрасное место для распространения UWP-приложений, как платных, так и бесплатных. Сами возможности платформы и магазина Windows Store позволяют использовать разные способы монетизации. Например, можно интегрировать в приложения блоки для показа рекламы через различные SDK. Можно распространять за определенную плату, причем оплату можно гибко настраивать. При необходимости можно встроить предоставление ознакомительной версии, после использования которой пользователь может решить, покупать приложение или нет. И также можно монетизировать по модели freemium, при которой приложение условно бесплатное, а отдельные услуги внутри приложения предоставляются за определенную плату. Причем все эти возможности монетизации обесечиваются встроенными инструментами SDK.
* Богатые возможности платформы. UWP многое наследует от Windows Runtime из Windows 8.1 и в то же время предоставляет много новых функцональностей, как, более богатые возможности по интеграции с облаком, использование Cortana, системы уведомлений в Win10 и многое другое.

## Начало работы. Первое приложение

## Создание проекта

Сначала создадим новый проект приложения для универсальной платформы Windows. Для этого типа проекта уже имеются все нужные файлы шаблонов, что избавляет вас от лишней работы.

1. Откройте Visual Studio и в начальном окне выберите. **Создать проект**.
2. На экране **Создание проекта** введите в поле поиска Универсальная платформа Windows, выберите шаблон C# **Пустое приложение (универсальная платформа Windows)** и щелкните **Далее**.

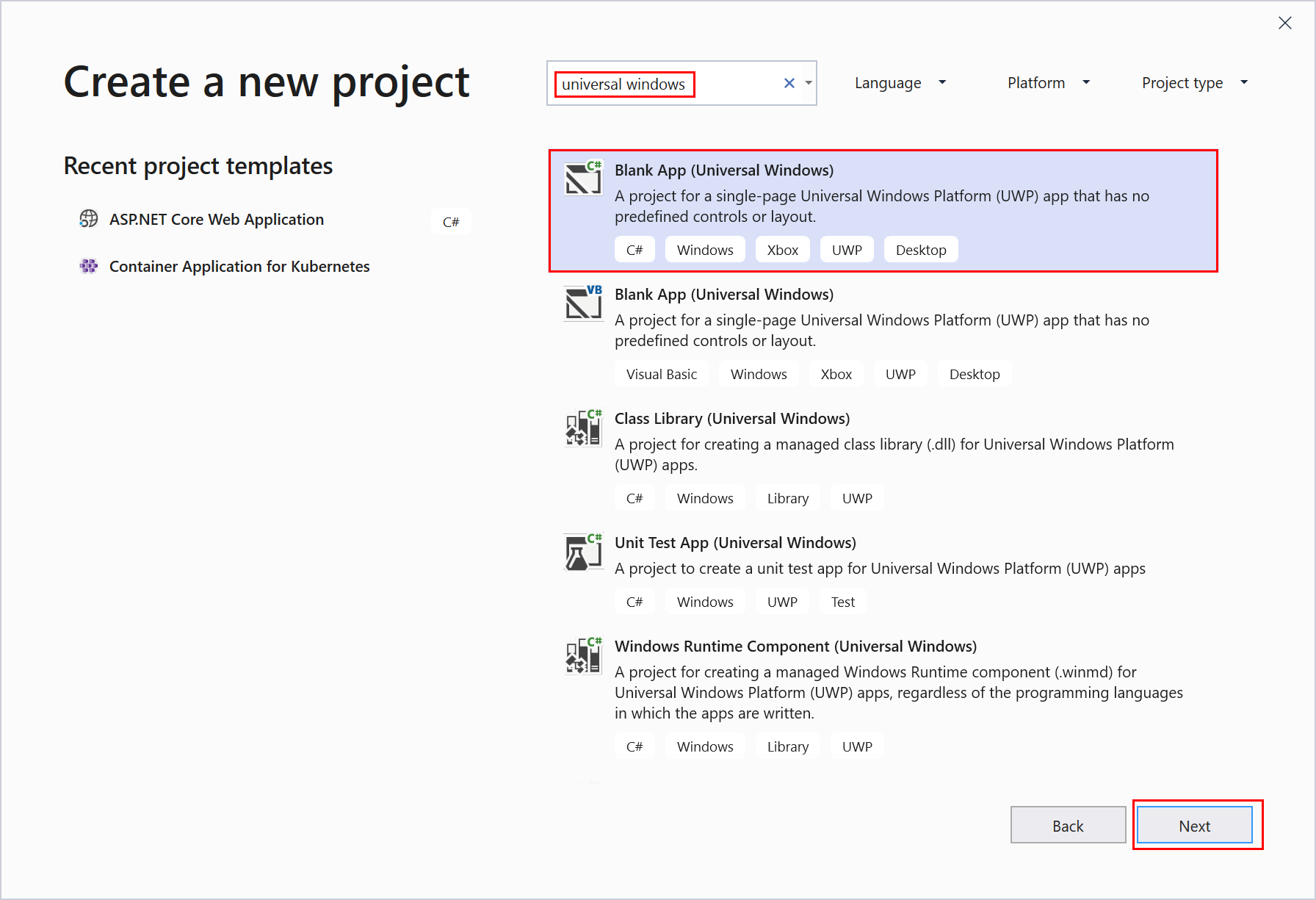


Рисунок 1. Создание проекта UWP

**Примечание**

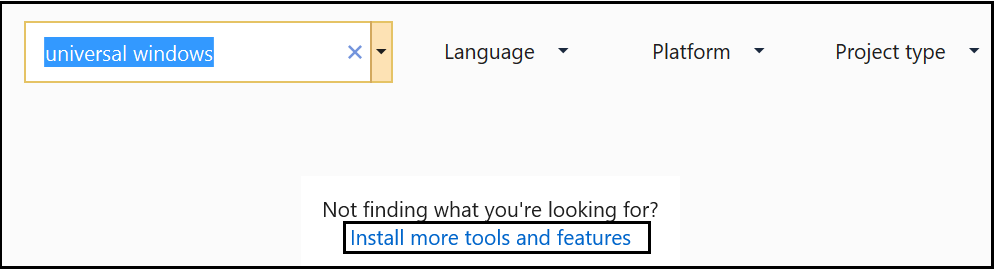
Если вы не видите здесь шаблон **Пустое приложение (универсальное приложение Windows)** , щелкните ссылку **Установка других средств и компонентов**.  
  


Рисунок 2. Окно поиска компонентов UWP

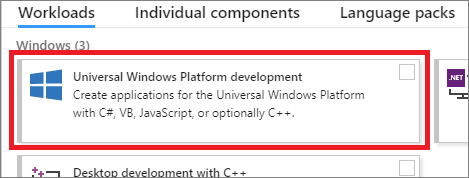
Запускается Visual Studio Installer. Выберите рабочую нагрузку **Разработка приложений для универсальной платформы Windows** и выберите **Изменить**.  
  


Рисунок 3. Установка компонентов UWP

1. Присвойте проекту имя HelloWorld и нажмите кнопку **Создать**.

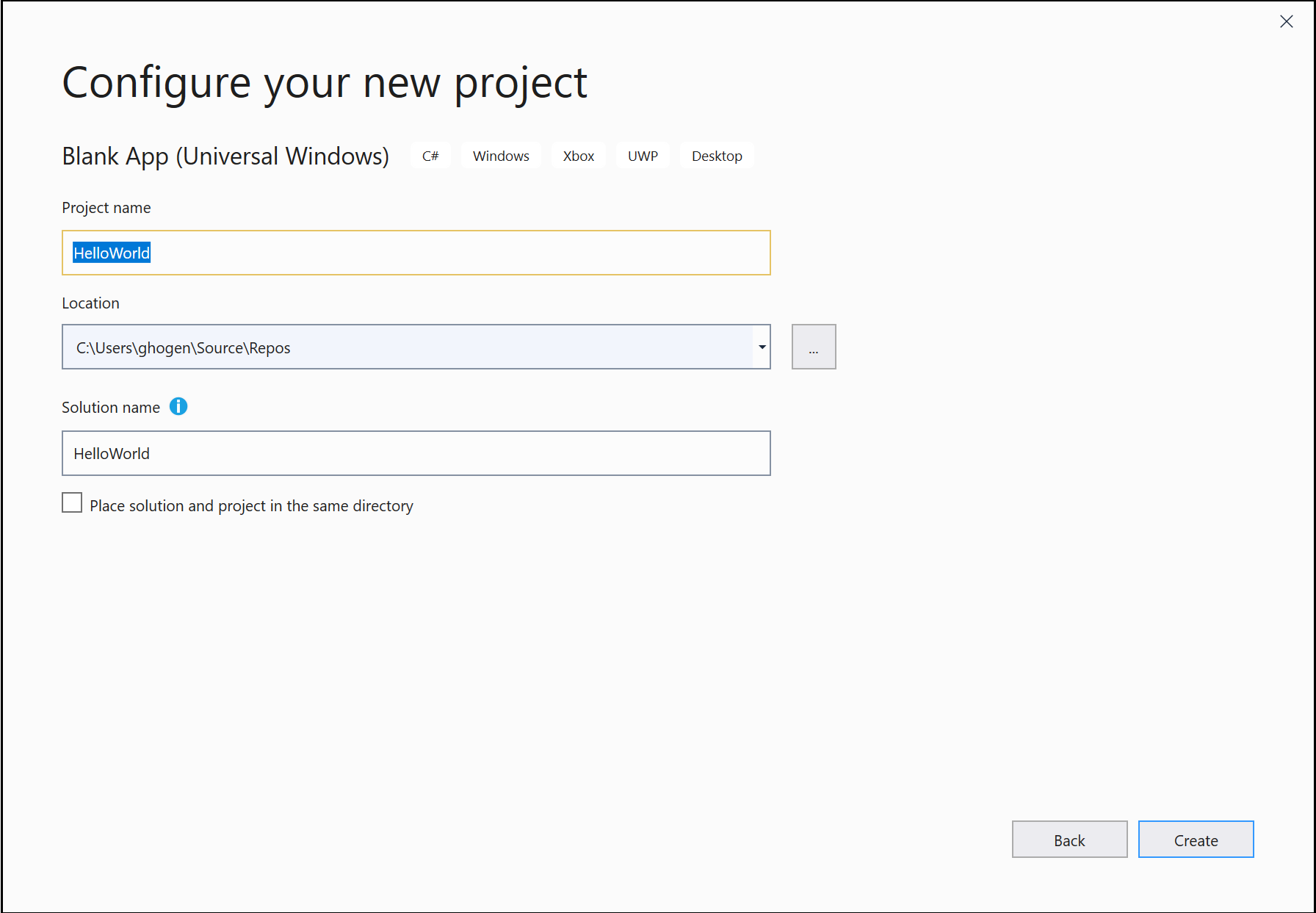


Рисунок 4. Настройка при создание проекта

1. Подтвердите значения по умолчанию для параметров **Целевая версия** и **Минимальная версия** в диалоговом окне **Новый проект приложения для универсальной платформы Windows**.

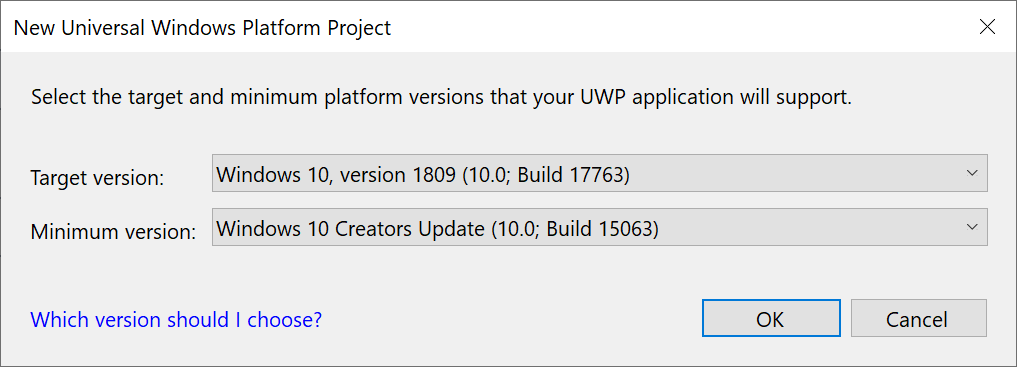


Рисунок 5. Выбор целевой версии OS

**Примечание**

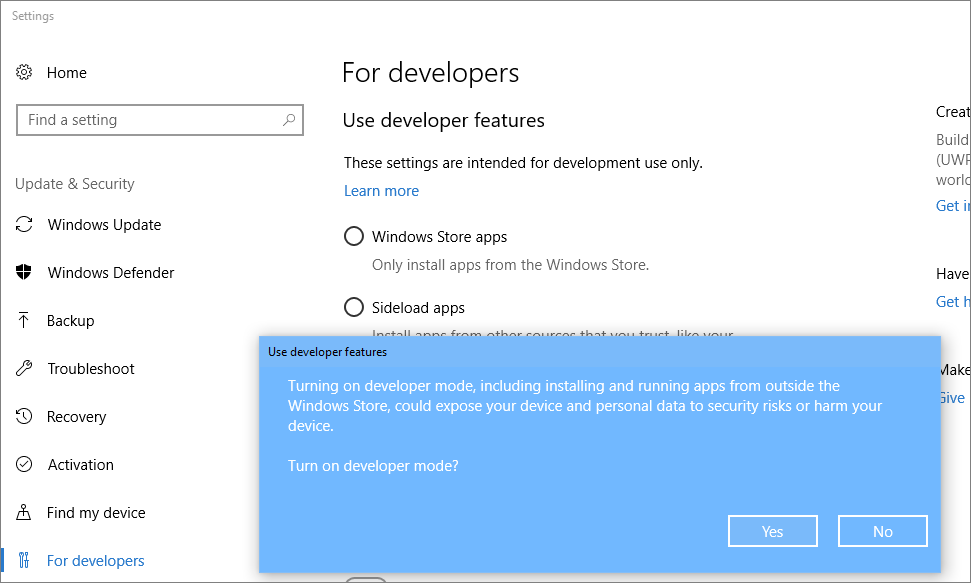
При первом использовании Visual Studio для создания приложения UWP появится диалоговое окно **Параметры**. Выберите **Режим разработчика**, затем нажмите **Да**.  
  


Рисунок 6. Активация режима разработчика в Windows

Visual Studio установит дополнительный пакет режима разработчика. После завершения установки пакета закройте диалоговое окно **Параметры**.

## Создание приложения

Теперь можно приступить к разработке. Мы добавим элемент управления ''Кнопка'', назначим кнопке действие и затем запустим приложение "Hello World", чтобы посмотреть, как оно выглядит.

### Добавление кнопки на холст конструктора

1. В **Обозревателе решений** дважды щелкните на файле MainPage.xaml, чтобы открыть разделенное представление.

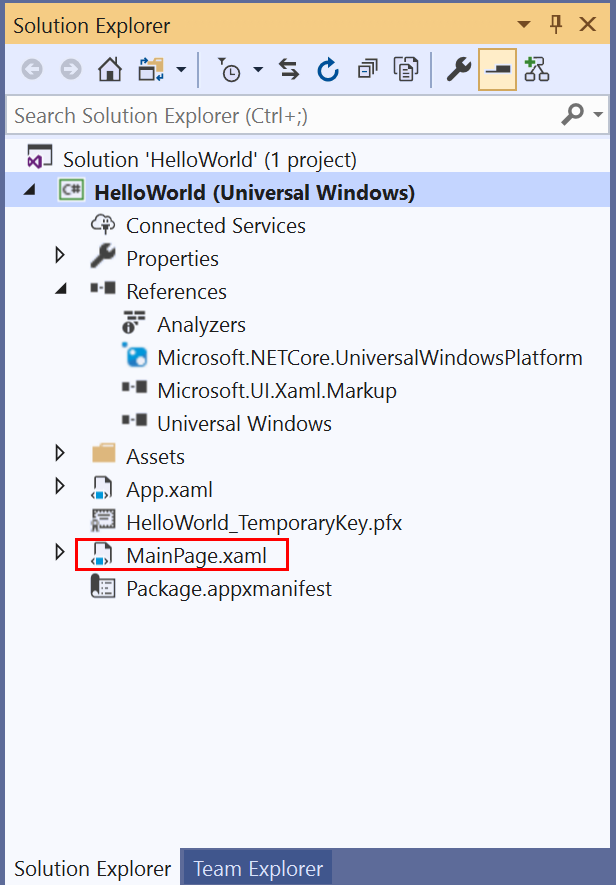


Рисунок 7. Обозреватель решений

Откроется две панели: **Конструктор XAML**, содержащий холст конструктора, и **редактор XAML**, в котором можно добавлять и изменять код.

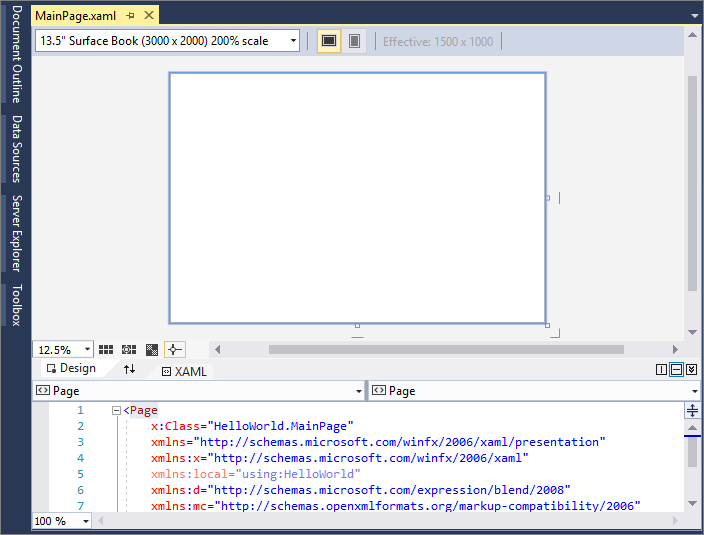


Рисунок 8. Рабочее пространство

1. Щелкните **Панель элементов**, чтобы открыть всплывающее окно "Панель элементов".

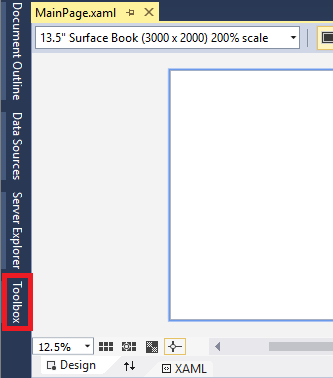


Рисунок 9. Выбор панели элементов

(Если параметр **Панель элементов** отсутствует, его можно открыть в строке меню. Для этого выберите **Вид** > **Панель инструментов**. Либо нажмите клавиши **CTRL**+**ALT**+**X**.)

1. Щелкните значок **Закрепить**, чтобы закрепить окно "Панель элементов".

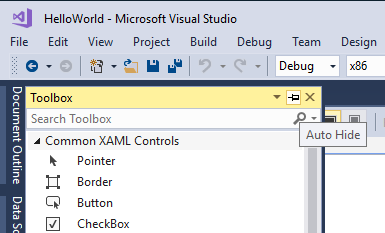


Рисунок 10. Фиксация панели

1. Щелкните элемент управления **Кнопка** и перетащите его на холст конструктора.

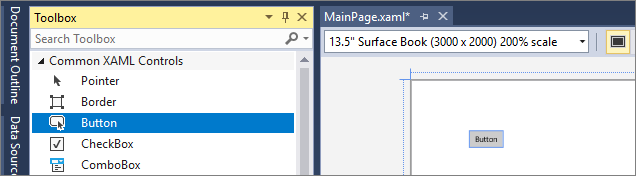


Рисунок 11. Выбор графического компонента

Если посмотреть на код в **редакторе XAML**, мы увидим, что кнопка была также добавлена туда:

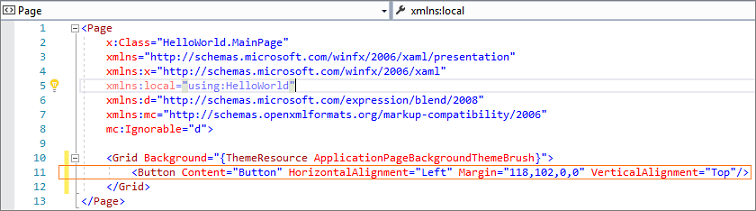


Рисунок 12. Редактор XAML

### Добавление метки к кнопке

1. В **редакторе XAML** измените значение параметра Content кнопки с "Button" на "Hello World!".

Измените значение параметра Content кнопки на "Hello World"

Рисунок 13. Код XAML кнопки

1. Обратите внимание, что кнопка в **Конструкторе XAML** также изменится.

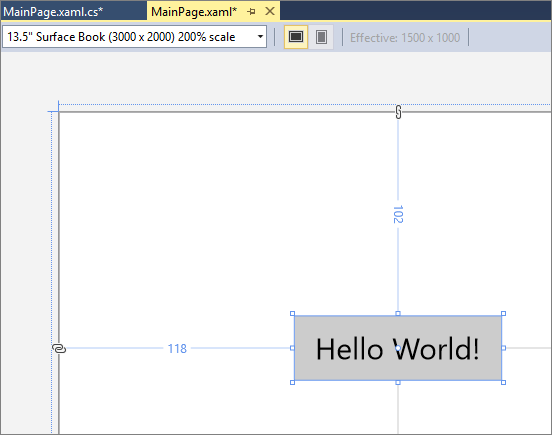


Рисунок 14. Размещение компонентов на интерфейсе

### Добавление обработчика событий

Название "обработчик событий" звучит как что-то сложное, но это всего лишь еще одно имя в коде, которое вызывается, когда происходит какое-то событие. В нашем случае он добавляет действие к кнопке "Hello World!". .

1. Дважды щелкните элемент управления "Кнопка" на холсте конструктора.
2. Измените код обработчика событий в файле MainPage.xaml.cs, страница с выделенным кодом.

Здесь начинается самое интересное. Обработчик события по умолчанию выглядит так:

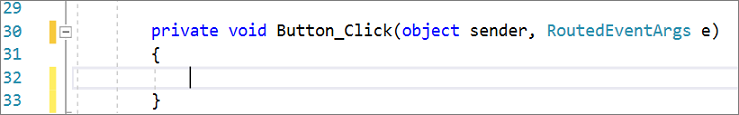


Рисунок 15. Код C# кнопки

Давайте изменим его, чтобы он выглядел следующим образом:

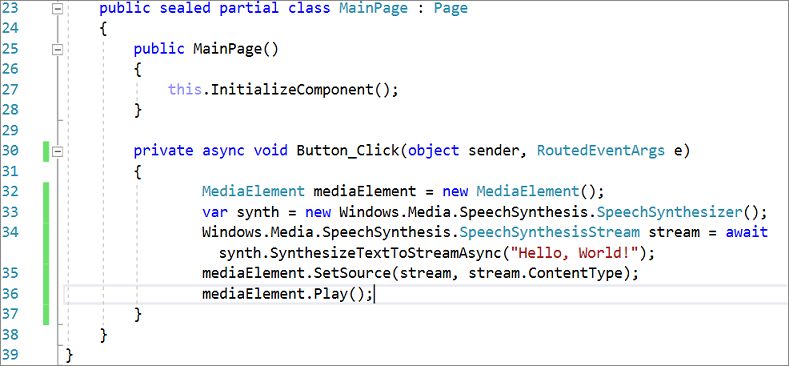


Рисунок 16. Обработка события при взаимодействия с кнопкой

Ниже приведен код для копирования и вставки:

private async void Button\_Click(object sender, RoutedEventArgs e)

{

MediaElement mediaElement = new MediaElement();

var synth = new Windows.Media.SpeechSynthesis.SpeechSynthesizer();

Windows.Media.SpeechSynthesis.SpeechSynthesisStream stream = await synth.SynthesizeTextToStreamAsync("Hello, World!");

mediaElement.SetSource(stream, stream.ContentType);

mediaElement.Play();

}

Что же мы только что сделали?

Этот код использует некоторые функции API Windows для создания объекта синтеза речи и передает ему некоторый текст для озвучивания. (Дополнительные сведения об использовании SpeechSynthesis см. в статье [System.Speech.Synthesis](https://docs.microsoft.com/ru-ru/dotnet/api/system.speech.synthesis).)

## Запуск приложения

Пришло время для построения, развертывания и запуска UWP-приложения "Hello World", чтобы посмотреть, как оно выглядит и как оно звучит. Ниже описывается порядок действий.

1. Используйте кнопку воспроизведения (с меткой **локальный компьютер**), чтобы запустить приложение на локальном компьютере.

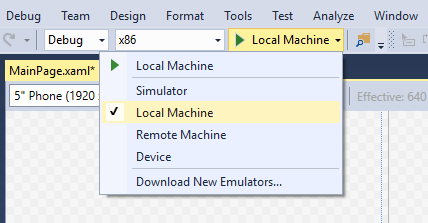


Рисунок 17. Выбор запукска приложения через компилятор

(Вы также можете выбрать команду **Отладка** > **Начать отладку** в строке меню или нажать клавишу F5 для запуска приложения.)

1. Посмотрите на свое приложение, которое появится вскоре после исчезновения всплывающего окна. Приложение должно выглядеть следующим образом:

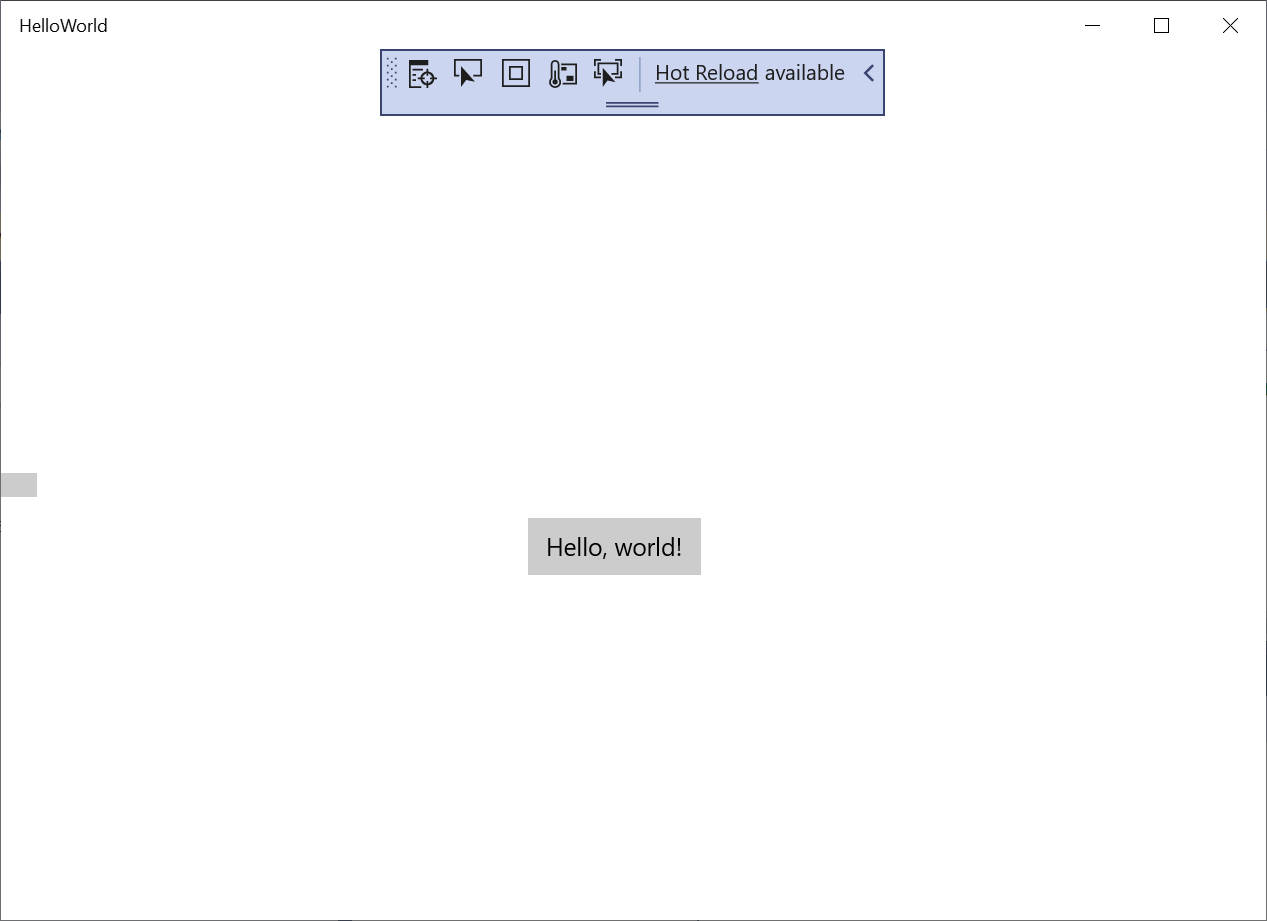


Рисунок 18. Интерфейс программы

1. Нажмите кнопку **Hello World**.

Ваше устройство с Windows 10 буквально произнесет "Hello, World!".

1. Для закрытия приложения нажмите кнопку **Остановить отладку** на панели инструментов. (Либо выберите пункт меню **Отладка** > **Остановить отладку** или нажмите клавиши SHIFT+F5.)

# ЗАДАНИЯ ДЛЯ ВЫПОЛНЕНИЯ

1. Создайте пользовательский интерфейс по заданию
2. Спроектируйте блок схему к программе
3. Отобразить в отчете проделанные шаги
4. Ответить на контрольные вопросы
5. Написать вывод о проделанной работе

# ИНДИВИДУАЛЬНЫЕ ЗАДАНИЯ

|  |
| --- |
| Вариант 1 – Реализуйте программу калькулятор с интерфейсом на UWP. Калькулятор должен считать в 10 системе счислений. Должны быть математические операции (сложение, вычитание, умножение, деление) |
| Вариант 2 – Реализуйте программу калькулятор с интерфейсом на UWP. Калькулятор должен считать в 2 системе счислений. Должны быть математические операции (сложение, вычитание, умножение, деление) |
| Вариант 3 – Реализуйте программу калькулятор с интерфейсом на UWP. Калькулятор должен считать в 16 системе счислений. Должны быть математические операции (сложение, вычитание, умножение, деление) |
| Вариант 4 - Реализуйте программу калькулятор с интерфейсом на UWP. Калькулятор должен считать в 8 системе счислений. Должны быть математические операции (сложение, вычитание, умножение, деление) |
| Вариант 5 – Реализуйте программу калькулятор с интерфейсом на UWP. Калькулятор должен считать в 10 системе счислений. Должны быть математические операции (сложение, вычитание, умножение, деление) |
| Вариант 6 - Реализуйте программу калькулятор с интерфейсом на UWP. Калькулятор должен считать в 2 системе счислений. Должны быть математические операции (сложение, вычитание, умножение, деление) |
| Вариант 7 - Реализуйте программу калькулятор с интерфейсом на UWP. Калькулятор должен считать в 16 системе счислений. Должны быть математические операции (сложение, вычитание, умножение, деление) |
| Вариант 8 - Реализуйте программу калькулятор с интерфейсом на UWP. Калькулятор должен считать в 8 системе счислений. Должны быть математические операции (сложение, вычитание, умножение, деление) |
| Вариант 9 - Реализуйте программу калькулятор с интерфейсом на UWP. Калькулятор должен считать в 4 системе счислений. Должны быть математические операции (сложение, вычитание, умножение, деление) |
| Вариант 10 - Реализуйте программу калькулятор с интерфейсом на UWP. Калькулятор должен считать в 4 системе счислений. Должны быть математические операции (сложение, вычитание, умножение, деление) |

# КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Как реализовывали задание?
2. Плюсы и минусы UWP.
3. Под какую сборку операционной системы вы проектировали приложение и почему?

# РЕКОМЕНДОВАННАЯ ЛИТЕРАТУРА

1. Официальная документация Microsoft C# — Текст: электронный // Microsoft [сайт]. — URL: https://docs.microsoft.com/ (дата обращения: 12.03.2020).
2. Сообщество IT-специалистов — Текст: электронный // Habr [сайт]. — URL: https://habr.com/ (дата обращения: 12.03.2020).
3. Сайт о программирование / — Текст: электронный // Metanit [сайт]. — https://metanit.com/ (дата обращения: 12.03.2020).
4. С# Tutotial — Текст: электронный // Tutotial [сайт]. — URL: https://csharp.net-tutorials.com/ (дата обращения: 13.03.2020).

# ПРИЛОЖЕНИЕ А

**Требования к отчету**

Общие требования:

1. Шрифт – Times New Roman, 14 пт.
2. Интервалы: междустрочный – 1,5 строки, интервал до и после абзаца – 0 пт.
3. Отступ первой строки – 1,25
4. Рисунки и подписи к ним выравниваются по центру.

Требования к структуре отчета:

1. Титульный лист
2. Содержание
3. Основная часть:
   1. Задачи
   2. Код
   3. Блок-схемы
   4. Результат работы
4. Выводы по работе
5. Ответы на контрольные вопросы

# ПРИЛОЖЕНИЕ Б

ОГБПОУ «ТОМСКИЙ ТЕХНИКУМ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ»

Отчет по учебной-практической работе №9

учебной практике   
УП 05. Программирование

Тема: «Реализация калькулятора с помощью UWP»

Выполнил:

студент \_\_\_\_\_\_\_ группы

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Проверил:  
преподаватель

Сидиков И.Д.

г. Томск – 2020 г