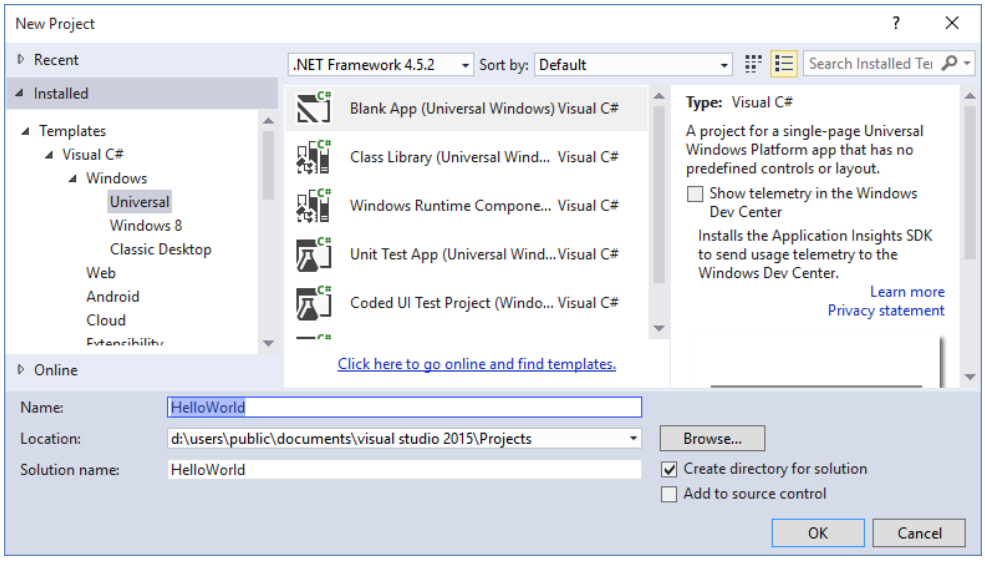
Microsoft Visual Studio Community 2015

Visual Studio Community 2015 ist eine kostenlose IDE mit leistungsfähigen Programmier- und Entwicklungswerkzeugen für Windows, iOS und Android. Sie ist für einzelne Entwickler, Open Source-Entwicklung, wissenschaftliche Forschung, Bildungseinrichtungen und kleine professionelle Teams kostenlos erhältlich und kann hier heruntergeladen werden.

Um sich mit den Möglichkeiten der IDE vertraut zu machen, empfehlen wir die Programmierung einer einfachen Anwendung, die wir in 4 Schritten nachzuvollziehen machen wollen.

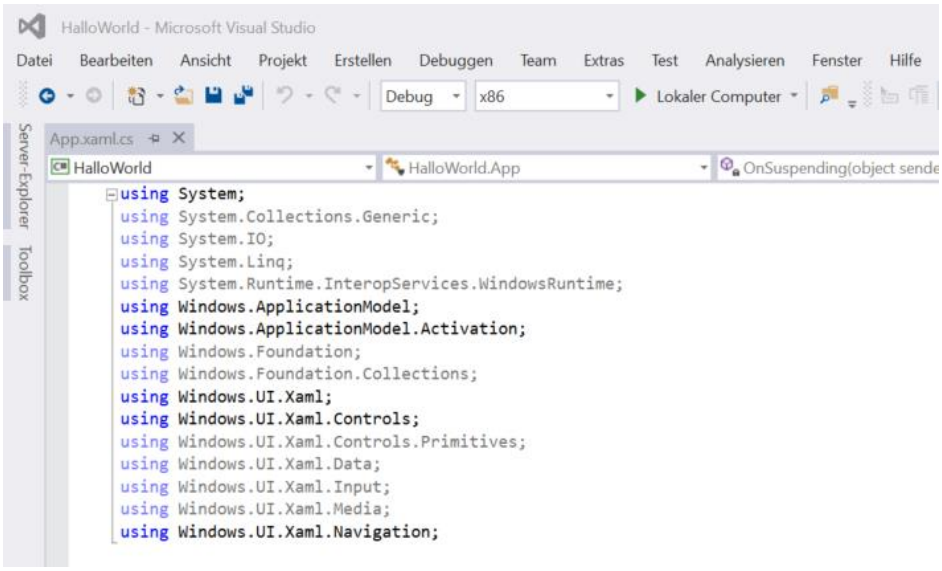
Schritt 1: Erstellen eines neuen Projekts in Visual Studio

1. Starten Sie Visual Studio 2015.Der Visual Studio 2015-Startbildschirm wird angezeigt. (Hinweis: Im weiteren Verlauf wird Visual Studio 2015 kurz als Visual Studio bezeichnet.)
2. Klicken Sie im Menü **Datei** auf **Neu** und dann auf **Projekt**. Das Dialogfeld **Neues Projekt** wird geöffnet. Im linken Bereich des Dialogfelds können Sie den Typ der anzuzeigenden Vorlagen auswählen.
3. Erweitern Sie im linken Bereich die Option **Installiert > Vorlagen > Visual C# > Windows** und wählen Sie anschließend die Vorlagengruppe **Universal**. Im mittleren Bereich des Dialogfelds sehen Sie eine Liste mit Projektvorlagen für universelle Windows-Plattform-Apps (UWP).

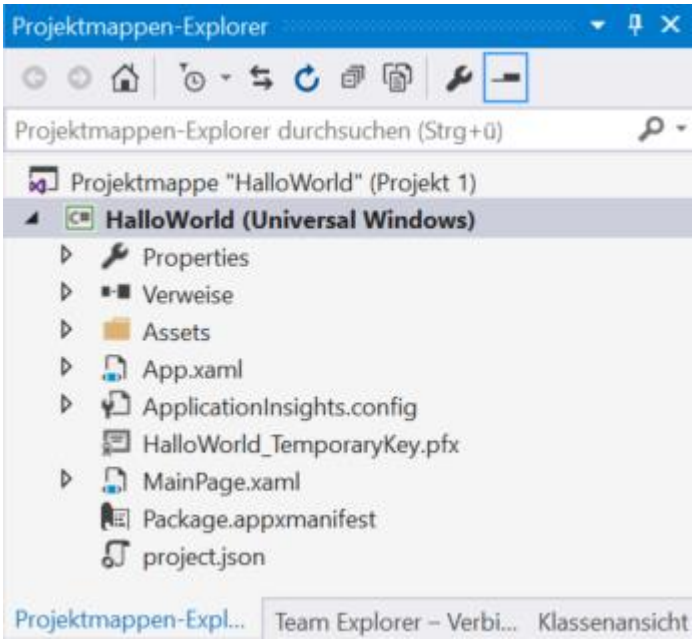


4. Wählen Sie im mittleren Bereich die Projektvorlage **Leere App (universelle Windows-App)** aus. Die Vorlage **Leere App** stellt eine minimale UWP-App bereit, die kompiliert und ausgeführt wird, aber keine UI-Steuerelemente oder -Daten enthält. Die App wird später in diesem Lernprogramm noch mit Steuerelementen versehen.
5. Geben Sie im Textfeld **Name** den Namen "HelloWorld" ein.
6. Klicken Sie auf **OK**, um das Projekt zu erstellen.

Visual Studio erstellt Ihr Projekt und zeigt Ihnen den Inhalt der Datei „App.xaml.cs“ an.



Starten Sie durch Drücken der Tastenkombination <Strg+Alt+L> oder über den Menüpunkt <Ansicht> den **Projektmappen-Explorer**.



Die **Projektmappe** umfasst eine Reihe von Dateien:

- Eine Datei „Package.appxmanifest“ beschreibt die App und die darin enthaltenen Dateien.
- Im Ordner <Assets> werden Bilder vorgehalten, die für das Startmenü, den Windows Store und den Begrüßungsbildschirm gebraucht werden.
- XAML- und Codedateien für die App (App.xaml und App.xaml.cs) sowie die Startseite (MainPage.xaml) und eine entsprechende Codedatei (MainPage.xaml.cs), die beim Start der App ausgeführt werden.

Diese Dateien werden für alle UWP-Apps mit C# benötigt. Sie sind Teil jedes Projekts, das Sie mit Visual Studio erstellen.

Schritt 2: Anpassen der Startseite

Inhalt der Dateien
Doppelklicken Sie zum Anzeigen und Bearbeiten einer Datei im Projekt im **Projektmappen-Explorer** auf die gewünschte Datei. XAML-Dateien werden in einer geteilten Ansicht geöffnet, die sowohl die Entwurfsoberfläche als auch den XAML-Editor enthält.

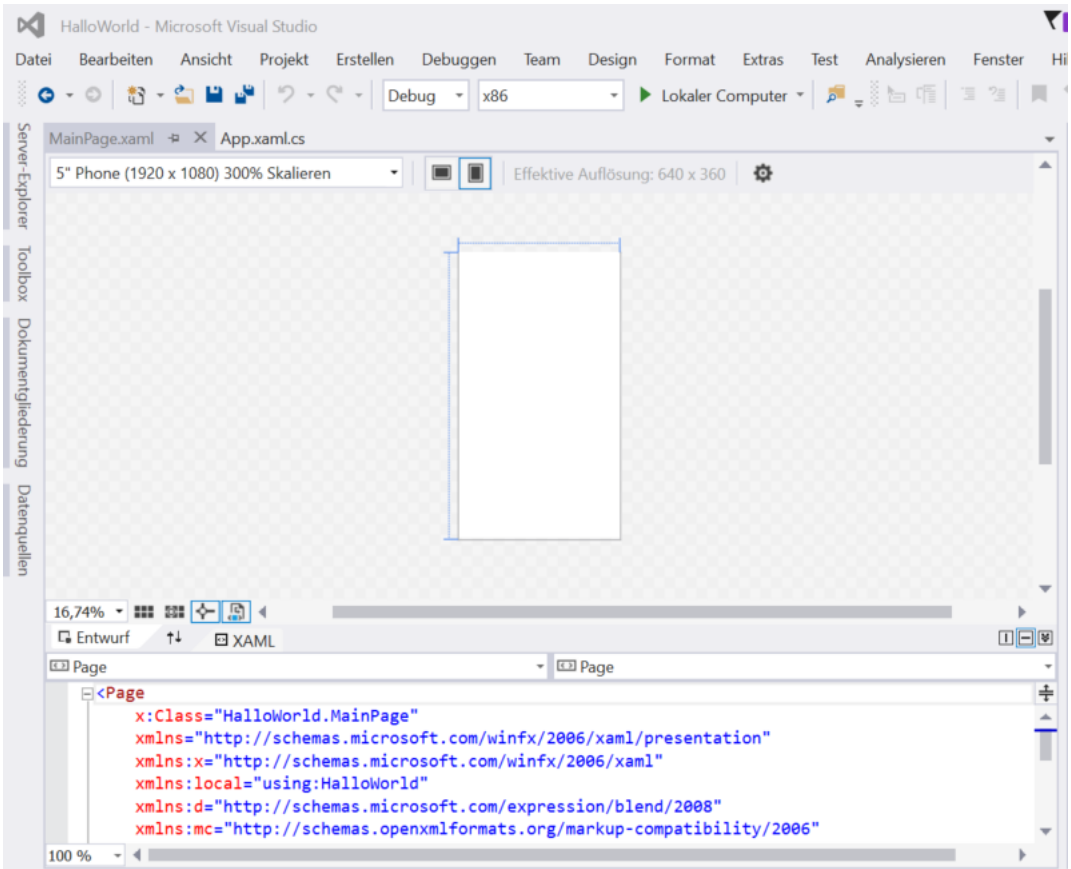
In diesem Lernprogramm verwenden Sie lediglich einige der zuvor aufgeführten Dateien: App.xaml, MainPage.xaml und MainPage.xaml.cs.

„App.xaml“ und „App.xaml.cs“
In „App.xaml“ werden die Ressourcen deklariert, die in der App zur Anwendung kommen. „App.xaml.cs“ ist eine mit „App.xaml“ verknüpfte Datei. Beide Dateien zusammen bilden beide eine Klasse.

Den Einstiegspunkt für Ihre App bildet die Datei „App.xaml.cs“. Sie enthält einen Konstruktor, der die in „App.xaml“ deklarierten Elemente initialisiert und unterbricht.

MainPage.xaml
In der Datei „MainPage.xaml“ definieren Sie die Benutzeroberfläche für Ihre App. Sie können Elemente direkt per XAML-Markup hinzufügen oder die Designtools von Visual Studio verwenden. „MainPage.xaml.cs“ ist die mit“ MainPage.xaml“ verknüpfte Datei. Hier fügen Sie Ihre App-Logik und den „Ereignishandler“ hinzu.
Auch diese beiden Dateien bilden im Projekt eine eigene Klasse.

So passen Sie die Startseite an
Klicken Sie im Projektmappen-Explorer doppelt auf „MainPage.xaml“. Es öffnet sich folgende Ansicht:



In der unteren Hälfte des Fensters findet sich der zur Ansicht passende Code:

```
<Page
    x:Class="HelloWorld.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:local="using:HelloWorld"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d">

    <Grid Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">

    </Grid>
</Page>
```

Fügen Sie den folgenden Code mittels „Copy&Paste“ zwischen <Grid Background="{ThemeResource ApplicationPageBackgroundThemeBrush}"> und „</Grid>“ ein.

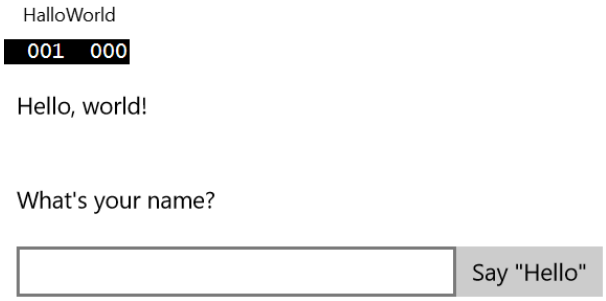
```
<StackPanel x:Name="contentPanel" Margin="8,32,0,0">
    <TextBlock Text="Hello, world!" Margin="0,0,0,40"/>
    <TextBlock Text="What's your name?"/>
    <StackPanel x:Name="inputPanel" Orientation="Horizontal" Margin="0,20,0,20">
        <TextBox x:Name="nameInput" Width="280" HorizontalAlignment="Left"/>
        <Button x:Name="inputButton" Content="Say &amp;quot;Hello&amp;quot;"/>
    </StackPanel>
    <TextBlock x:Name="greetingOutput"/>
</StackPanel>
```

In diesem Code sind Elemente enthalten, die den Namen des Benutzers anfordern und als Nutzer festlegen. Weitere Elemente dienen zum Anzeigen einer Begrüßung. Einige dieser Steuerelemente haben Namen, um später im Code darauf verweisen zu können.

Schritt 3: Starten der App

Damit die App auf Ihrem Computer angezeigt wird, muss man sie „debuggen“. Drücken Sie dazu <F5> auf Ihrer Tastatur.

Nach kurzer Zeit öffnet sich ein Fenster mit folgender Ansicht:



Schließen Sie diese Ansicht, indem Sie auf das Kreuz im Fenster rechts oben klicken

Schritt 4: Erstellen eines Ereignishandlers

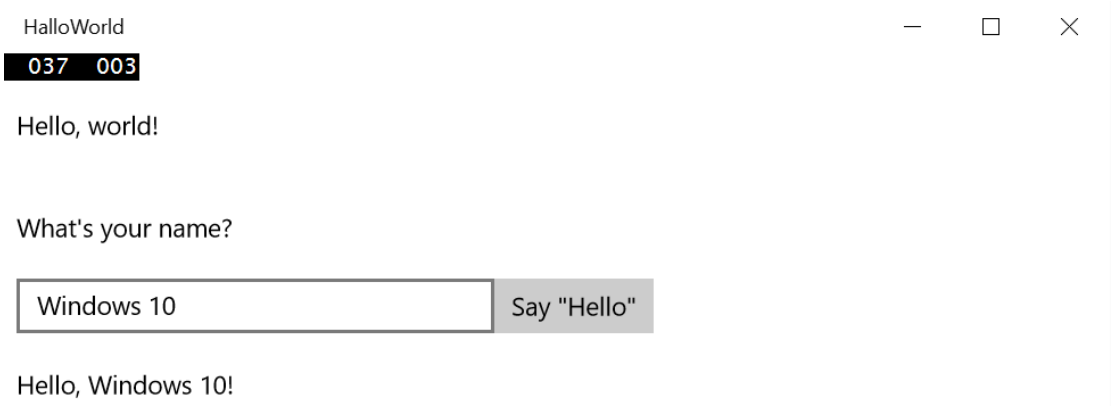
XAML-Elemente können Meldungen senden, wenn bestimmte Ereignisse eintreten. Ein häufig verwendetes Ereignisse in vielen Apps ist das Klicken auf ein **Button**-Element.
Wir definieren nun ein solches **Click**-Ereignis.

So fügen Sie den Ereignishandler hinzu

1. Setzen Sie hinter „<Grid“ in der „MainPage.xaml“ den Cursor im <Button-Element hinter "Say "Hello"".
2. Tippen Sie zunächst ein Leerzeichen und danach einen Buchstaben ein und suchen Sie in dem sich öffnenden Menü nach „Click“.
3. Ein Doppelklick darauf erweitert die gewählte Programmzeile um den Eintrag Click=““. Gleichzeitig wird die Anzeige <Neuer Ereignishandler> eingeblendet. Ein Doppelklick darauf vervollständigt die Befehlszeile. Sie lautet nun: Click=“inputButton_Click“.
4. Klicken Sie anschließend mit der rechten Maustaste auf diesen Eintrag und wählen Sie in dem sich öffnenden Menü „Gehe zu Definition“, die Sie an eine bestimmte Stelle der „MainPage.xaml.cs“ bringt.
5. Hier fügen Sie den Code aus Zeile 3 unten ein (greetingOutput …).

```
private void Button_Click(object sender, RoutedEventArgs e)
{
   greetingOutput.Text = "Hello, " + nameInput.Text + "!";
}
```

6. Debuggen Sie die App durch Drücken der F5.Taste. Wenn Sie nun einen Namen in das Textfeld eingeben und anschließend auf die Schaltfläche <Say „Hello“> klicken, zeigt die App eine personalisierte Begrüßung an.



7. Beenden Sie die App, indem Sie auf das Kreuz im Fenster rechts oben klicken.

Schritt 5: Anpassen der App an verschiedene Fenstergrößen

Zum Anpassen der App an verschiedene Bildschirmgrößen müssen Sie ein **VisualStateManager**-Element hinzufügen, das die Eigenschaften für unterschiedliche Ansichtszustände festlegt.

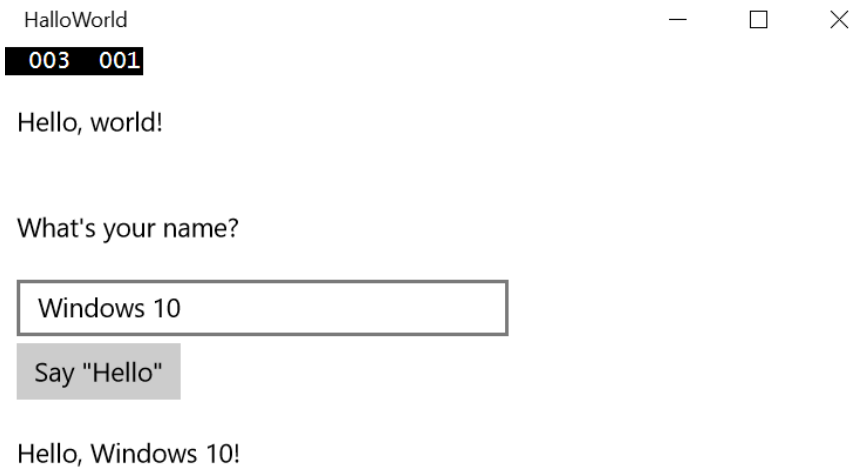
So passen Sie das App-Layout an

1. Fügen Sie in der „MainPage.xaml“ mittels „Copy&Paste“ zwischen der Zeile „<Grid Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">“

und
<StackPanel x:Name="contentPanel" Margin="8,32,0,0">
den folgenden Code-Block hinzu:

```
<VisualStateManager.VisualStateGroups>
  <VisualStateGroup>
    <VisualState x:Name="wideState">
      <VisualState.StateTriggers>
        <AdaptiveTrigger MinWindowWidth="641" />
      </VisualState.StateTriggers>
    </VisualState>
    <VisualState x:Name="narrowState">
      <VisualState.StateTriggers>
        <AdaptiveTrigger MinWindowWidth="0" />
      </VisualState.StateTriggers>
      <VisualState.Setters>
        <Setter Target="inputPanel.Orientation" Value="Vertical"/>
        <Setter Target="inputButton.Margin" Value="0,4,0,0"/>
      </VisualState.Setters>
    </VisualState>
  </VisualStateGroup>
</VisualStateManager.VisualStateGroups>
```

2. Debuggen Sie die App auf Ihrem Computer durch erneutes Drücken der F5-Taste. Sie sehen, dass die APP genauso wie vorher aussieht, es sei denn, Sie verringern die Fensterbreite Bewegen Sie dazu den Mauszeiger über den rechten Rand. Wenn sich der Zeiger zu einem Doppelpfeil wandelt, drücken Sie die linke Maustaste und bewegen die Maus mit gedrückter Taste nach links. Bei 641 Pixeln lässt sich das Fenster nicht weiter verkleinern, aber das Feld <Say „Hello“> wird nach unten verschoben.



Zusammenfassung
Herzlichen Glückwunsch, Sie haben Ihre erste App mit VisualStudio 2015 erstellt!