.NET MAUI Project, Item Templates and Code Snippets

This VS extension is loaded with Project, Item Templates and Code Snippets for working with .NET MAUI in Visual Studio 2022.

This has project template for MAUI, MAUI Blazor and MAUI Class Library project and it has been named as:

* .NET MAUI App (Preview 8)
* .NET MAUI App (C#) (Preview 8)
* .NET MAUI Blazor App (Preview 8)
* .NET MAUI Class Library (Preview 8)

And reg. Item templates:

ContentPage, in both XAML and C#, and has been named as:

* Content Page (.NET MAUI)
* Content Page (C#) (.NET MAUI)

ContentView, in both XAML and C#, and has been named as:

* Content View (.NET MAUI)
* Content View (C#) (.NET MAUI)

A page for defining app visual hierarchy with Shell and a page for managing resources with Resource Dictionary, both in XAML.

Has XAML Code Snippets for:

In the XAML page, type the short name and hit the Tab key twice to insert the snippet.

Snippets mentioned in bold-face also works as a **SurroundWith** snippet too (from **Xaml** section).

In the Output Format column, text highlighted in different colors infer the following:

* Yellow color are placeholders where user can modify the values
* Green color are derived values, can’t be modified. For example, containing class name
* Torquoise color are reflected values, where the placeholder value is filled-in

| **Snippet** | **Short Name** | **Output Format** |
| --- | --- | --- |
| **Grid** | **grid** | <Grid ColumnDefinitions="" RowDefinitions="">  </Grid> |
| **Flex Layout** | **flex** | <FlexLayout>  </FlexLayout> |
| **Stack Layout** | **stack** | <StackLayout>  </StackLayout> |
| **Horizontal Stack Layout** | **hstack** | <HorizontalStackLayout>  </HorizontalStackLayout> |
| **Vertical Stack Layout** | **vstack** | <VerticalStackLayout>  </VerticalStackLayout> |
| **Style** | **style** | <Style TargetType="Page">  <Style> |
| Color | color | <Color x:Key="Success">Green</Color> |
| Resources | res | <ContentPage.Resources>  <ResourceDictionary>  </ResourceDictionary>  </ContentPage.Resources> |
| Gestures | gesture | <Label.GestureRecognizers>  </Label.GestureRecognizers> |
| Tap Gesture Recognizer | tap | <TapGestureRecognizer /> |
| Drag Gesture Recognizer | drag | <DragGestureRecognizer /> |
| Drop Gesture Recognizer | drop | <DropGestureRecognizer /> |
| Pan Gesture Recognizer | pan | <PanGestureRecognizer /> |
| Pinch Gesture Recognizer | pinch | <PinchGestureRecognizer /> |
| Swipe Gesture Recognizer | swipe | <SwipeGestureRecognizer /> |

Has C# Code Snippets for:

In the C# code file, type the short name and hit the Tab key twice to insert the snippet.

Snippets mentioned in bold-face also works as a **SurroundWith** snippet too (from **CSharp** section).

In the Output Format column, text highlighted in different colors infer the following:

* Yellow color are placeholders where user can modify the values
* Green color are derived values, can’t be modified. For example, containing class name
* Torquoise color are reflected values, where the placeholder value is filled-in

| **Snippet** | **Short Name** | **Output Format** |
| --- | --- | --- |
| **Async Event Handler** | **aeh** | private async void MyMethod(object sender, EventArgs e)  {  } |
| Attached Property | propap | *Here assuming MyClass is the containing type.*  public static readonly BindableProperty NameProperty = BindableProperty.CreateAttached(nameof(NameProperty), typeof(string), typeof(MyClass), default(string));  public static string GetName(BindableObject bindable) => (string)bindable.GetValue(NameProperty);  public static void SetName(BindableObject bindable, string value) => bindable.SetValue(NameProperty, value); |
| Bindable Property | propbp | *Here assuming MyClass is the containing type.*  public static readonly BindableProperty NameProperty = BindableProperty.Create(nameof(Name), typeof(string), typeof(MyClass), default(string));  public string Name  {  get => (string)GetValue(NameProperty);  set => SetValue(NameProperty, value);  } |
| Comet Property (MVU) | propcomet | public string Name  {  get => GetProperty<string>();  set => SetProperty(value);  } |
| Cross Platform | xplat | #if ANDROID  #elif IOS  #elif MACCATALYST  #elif WINDOWS  #endif |
| **Event Handler** | **eh** | private void MyMethod(object sender, EventArgs e)  {  } |
| Method | method | private method MyMethod()  {  } |
| **Record**  **(C# 9.0 or higher)** | **record** | record MyRecord  {  } |
| **Record Struct**  **(C# 10.0 or higher)** | **recstruct** | Record struct MyRecStruct  {  } |
| ViewModel Property | propvm | private string name;  public string Name  {  get => name;  set => SetProperty(ref name, value);  } |

Note: Currently, these project templates target .NET MAUI Preview 8 (.NET 6 RC1) and to support further changes in newer .NET 6 releases, an update to this VS extension will be made available accordingly.