

[](https://www.tutorialr.com/tutorials/)

Windows App SDK

Docking Layout





# Docking Layout

**Docking Layout** shows how to create a **Docking** **Panel** using **Windows App SDK**

## Step 1

Follow **Setup and Start** on how to get **Setup** and **Install** what you need for **Visual Studio 2022** and **Windows App SDK**.

|  |  |
| --- | --- |
| In **Windows 11** choose **Start** and then find or search for **Visual Studio 2022** and then select it. | Text  Description automatically generated |
| Once **Visual Studio 2022** has started select **Create a new project**. | **Graphical user interface, text  Description automatically generated** |
| Then choose the **Blank App, Packages (WinUI in Desktop)** and then select **Next**. | **Graphical user interface, text  Description automatically generated** |
| After that in **Configure your new project** type in the **Project name** as *DockingLayout*, then select a Location and then select **Create** to start a new **Solution**. | **Graphical user interface, text, application, email  Description automatically generated** |

## Step 2

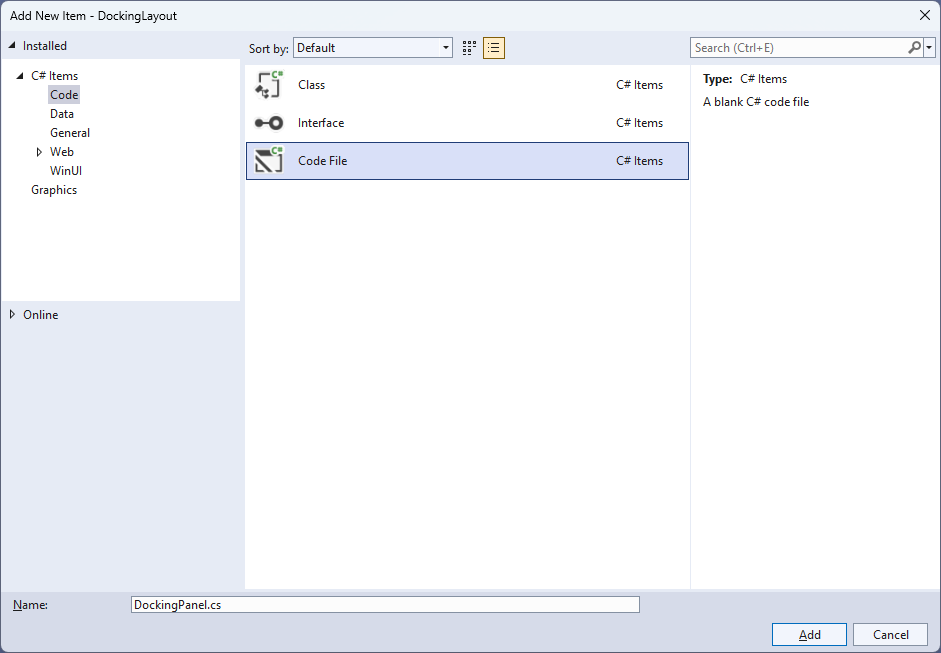
Then in **Visual Studio** within **Solution** **Explorer** for the **Solution**, right click on the **Project** shown below the **Solution** and then select **Add** then **New Item…**

Table

Description automatically generated with low confidence

## Step 3

Then in **Add New Item** from the **C# Items** list, select **Code** and then select **Code File** from the list next to this, then type in the name of *DockingPanel.cs* and then **Click** on **Add**.



## Step 4

|  |  |
| --- | --- |
| Then from **Solution** **Explorer** for the **Solution** double-click on **DockingPanel.cs** to see the **Code** for the **User Control**. |  |

## Step 5

You will now be in the **View** for the **Code** of *DockingPanel.cs*, within this type in the following **Code**:

using Microsoft.UI.Xaml;

using Microsoft.UI.Xaml.Controls;

using System;

using Windows.Foundation;

namespace DockingLayout;

public class DockingPanel : Panel

{

public enum Dock

{

Left,

Top,

Right,

Bottom

}

// Dependency Properties, Properties, Get Dock & Set Dock Methods

// Measure Override Method

// Arrange Override Method

}

There are **using** statements for the **User Control**, a **namespace** for **DockingLayout** along with a **class** of **DockingPanel** that will represent the **User Control** and **Inherits** the **class** of **Panel** which has an **enum** of **Dock** for the different **Docking** options supported by the **User Control.**

## Step 6

Then in the **namespace** of **DockingLayout** in the **class** of **DockingPanel** after the **Comment** of **// Dependency Properties, Properties, Get Dock & Set Dock Methods** type the following **Dependency Properties**, **Properties** and **Methods**:

public static readonly DependencyProperty LastChildFillProperty =

DependencyProperty.Register(nameof(LastChildFill), typeof(bool),

typeof(DockingPanel), new PropertyMetadata(false));

public static readonly DependencyProperty DockProperty =

DependencyProperty.RegisterAttached(nameof(Dock), typeof(Dock),

typeof(DockingPanel), new PropertyMetadata(Dock.Left));

public bool LastChildFill

{

get { return (bool)GetValue(LastChildFillProperty); }

set { SetValue(LastChildFillProperty, value); }

}

public static Dock GetDock(UIElement element)

{

ArgumentNullException.ThrowIfNull(element);

return (Dock)element.GetValue(DockProperty);

}

public static void SetDock(UIElement element, Dock dock)

{

ArgumentNullException.ThrowIfNull(element);

element.SetValue(DockProperty, dock);

}

**Dependency Properties** or **Properties** for the **User Control** can be customised for the **Docking Panel** along with some convention-based **Methods** of **GetDock** and **SetDock** used with the **Property** of **Dock**.

## Step 7

While still in the **namespace** of **DockingLayout** in the **class** of **DockingPanel** after the **Comment** of **// Measure Override Method** type the following **Method**:

protected override Size MeasureOverride(Size availableSize)

{

double width = 0.0;

double height = 0.0;

double maxWidth = 0.0;

double maxHeight = 0.0;

foreach (var element in Children)

{

var remainingSize = new Size(

Math.Max(0.0, availableSize.Width - width),

Math.Max(0.0, availableSize.Height - height));

element.Measure(remainingSize);

var desiredSize = element.DesiredSize;

switch (GetDock(element))

{

case Dock.Left:

case Dock.Right:

maxHeight = Math.Max(maxHeight, height + desiredSize.Height);

width += desiredSize.Width;

break;

case Dock.Top:

case Dock.Bottom:

maxWidth = Math.Max(maxWidth, width + desiredSize.Width);

height += desiredSize.Height;

break;

}

}

maxWidth = Math.Max(maxWidth, width);

maxHeight = Math.Max(maxHeight, height);

return new Size(maxWidth, maxHeight);

}

The **Method** of **MeasureOverride** will **Measure** the **Size** required to layout the **Children** of the **Panel**.

## Step 8

While still in the **namespace** of **DockingLayout** in the **class** of **DockingPanel** after the **Comment** of **// Arrange Override Method** type the following **Method**:

protected override Size ArrangeOverride(Size finalSize)

{

double left = 0.0;

double top = 0.0;

double right = 0.0;

double bottom = 0.0;

var children = Children;

var count = children.Count - (LastChildFill ? 1 : 0);

var index = 0;

foreach (var element in children)

{

var rect = new Rect(left, top,

Math.Max(0.0, finalSize.Width - left - right),

Math.Max(0.0, finalSize.Height - top - bottom));

if (index < count)

{

var desiredSize = element.DesiredSize;

switch (GetDock(element))

{

case Dock.Left:

left += desiredSize.Width;

rect.Width = desiredSize.Width;

break;

case Dock.Top:

top += desiredSize.Height;

rect.Height = desiredSize.Height;

break;

case Dock.Right:

right += desiredSize.Width;

rect.X = Math.Max(0.0, finalSize.Width - right);

rect.Width = desiredSize.Width;

break;

case Dock.Bottom:

bottom += desiredSize.Height;

rect.Y = Math.Max(0.0, finalSize.Height - bottom);

rect.Height = desiredSize.Height;

break;

}

}

element.Arrange(rect);

index++;

}

return finalSize;

}

The **Method** of **ArrangeOverride** will position the **Children** of the **Panel** using the **Property** of **Dock** getting the correct **Size** of them for the **User Control**.

## Step 9

|  |  |
| --- | --- |
| Within **Solution** **Explorer** for the **Solution** double-click on **MainWindow.xaml** to see the **XAML** for the **Main Window**. |  |

## Step 10

In the **XAML** for **MainWindow.xaml** there be some **XAML** for a **StackPanel**, this should be **Removed** by removing the following:

<StackPanel Orientation="Horizontal"

HorizontalAlignment="Center" VerticalAlignment="Center">

<Button x:Name="myButton" Click="myButton\_Click">Click Me</Button>

</StackPanel>

## Step 11

While still in the **XAML** for **MainWindow.xaml** above **</Window>**, type in the following **XAML**:

<local:DockingPanel LastChildFill="True"

HorizontalAlignment="Center" VerticalAlignment="Center">

<Rectangle Width="100" Height="100" Fill="Red"

Margin="10" local:DockingPanel.Dock="Top"/>

<Rectangle Width="100" Height="100" Fill="Orange"

Margin="10" local:DockingPanel.Dock="Top"/>

<Rectangle Width="100" Height="100" Fill="Yellow"

Margin="10" local:DockingPanel.Dock="Bottom"/>

<Rectangle Width="100" Height="100" Fill="Green"

Margin="10" local:DockingPanel.Dock="Bottom"/>

<Rectangle Width="100" Height="100" Fill="Cyan"

Margin="10" local:DockingPanel.Dock="Left"/>

<Rectangle Width="100" Height="100" Fill="Blue"

Margin="10" local:DockingPanel.Dock="Left"/>

<Rectangle Width="100" Height="100" Fill="Magenta"

Margin="10" local:DockingPanel.Dock="Right"/>

<Rectangle Width="100" Height="100" Fill="Purple"

Margin="10" local:DockingPanel.Dock="Right"/>

</local:DockingPanel>

This **XAML** contains the **User Control** of **DockingPanel** with **LastChildFill** set to **True** and the **Children** containing **Controls** for a **Rectangle** in various colours.

## Step 12

|  |  |
| --- | --- |
| Then, within **Solution** **Explorer** for the **Solution** select the arrow next to **MainWindow.xaml** then double-click on **MainWindow.xaml.cs** to see the **Code** for the **Main Window**. |  |

## Step 13

In the **Code** for **MainWindow.xaml.cs** there be a **Method** of **myButton\_Click(...)** this should be **Removed** by removing the following:

private void myButton\_Click(object sender, RoutedEventArgs e)

{

myButton.Content = "Clicked";

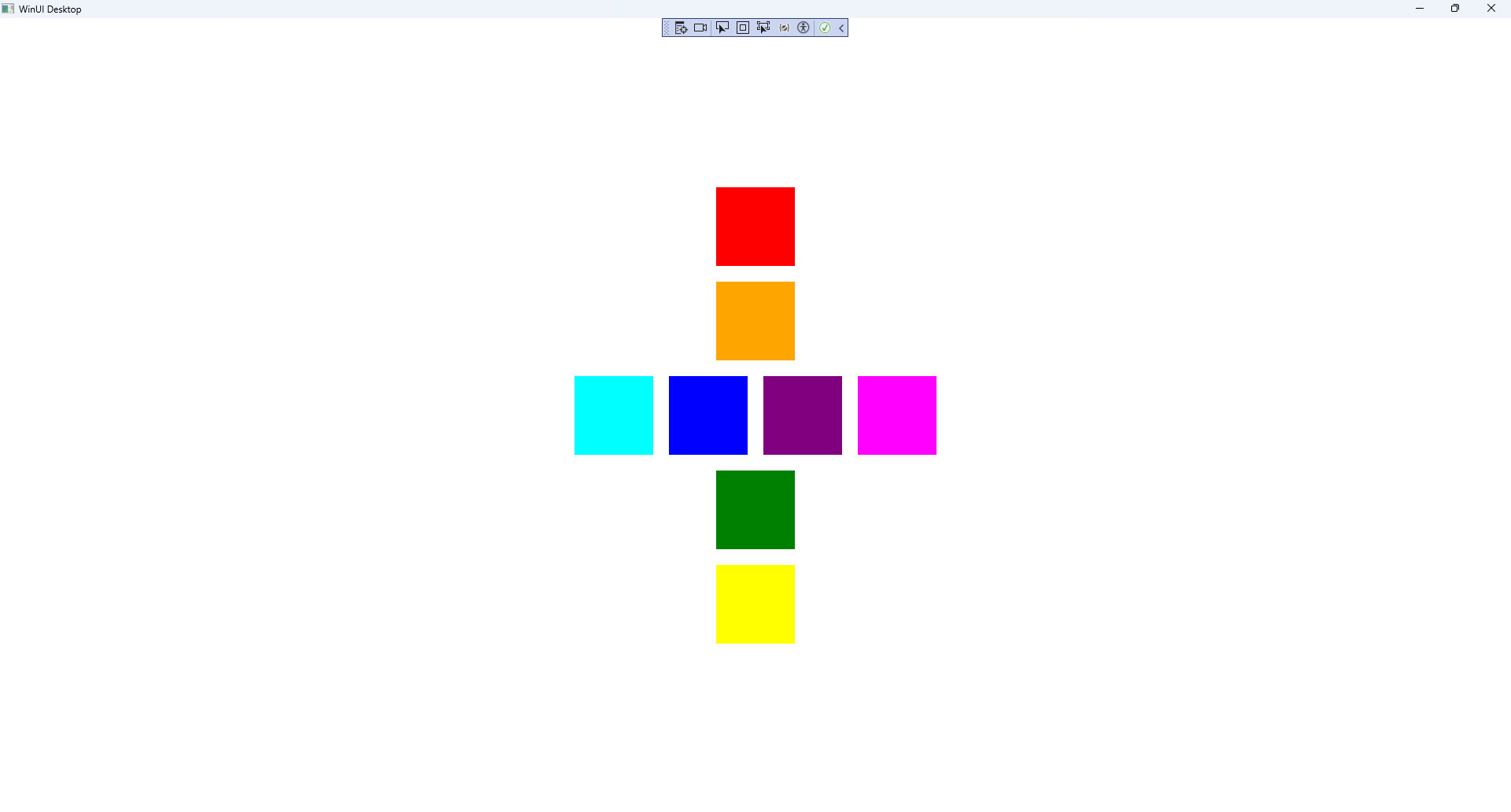
}

## Step 14

|  |  |
| --- | --- |
| That completes the **Windows App SDK** application. In **Visual Studio 2022** from the **Toolbar** select **DockingLayout (Package)** to **Start** the application. |  |

## Step 15

Once running you will see the **Docking Panel** displayed.

****

## Step 16

|  |  |
| --- | --- |
| To **Exit** the **Windows App SDK** application, select the **Close** button from the top right of the application as that concludes this **Tutorial** for **Windows App SDK** from [tutorialr.com](https://tutorialr.com)! |  |