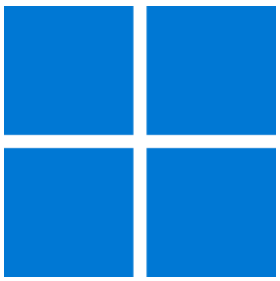
tutorial

Windows App SDK



Radial Layout

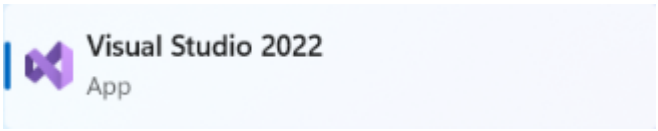
Radial Layout

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

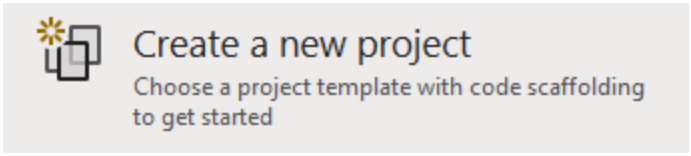
Step 1

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

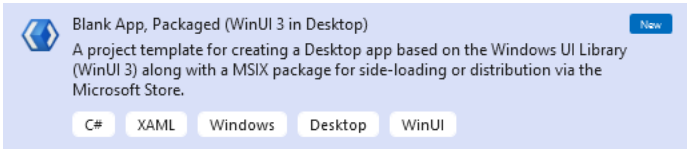
In **Windows 11** choose **Start** and then find or search for **Visual Studio 2022** and then select it.



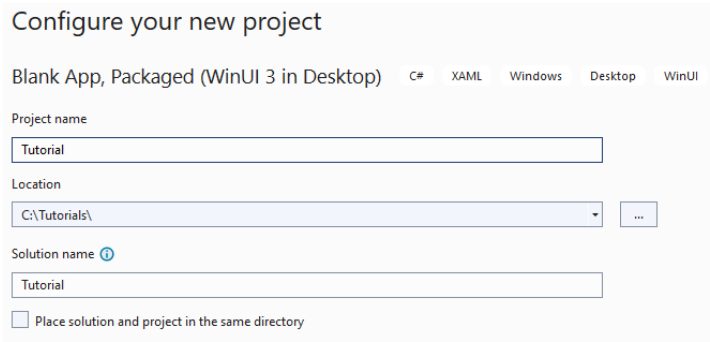
Once **Visual Studio 2022** has started select **Create a new project**.



Then choose the **Blank App, Packages (WinUI in Desktop)** and then select **Next**.

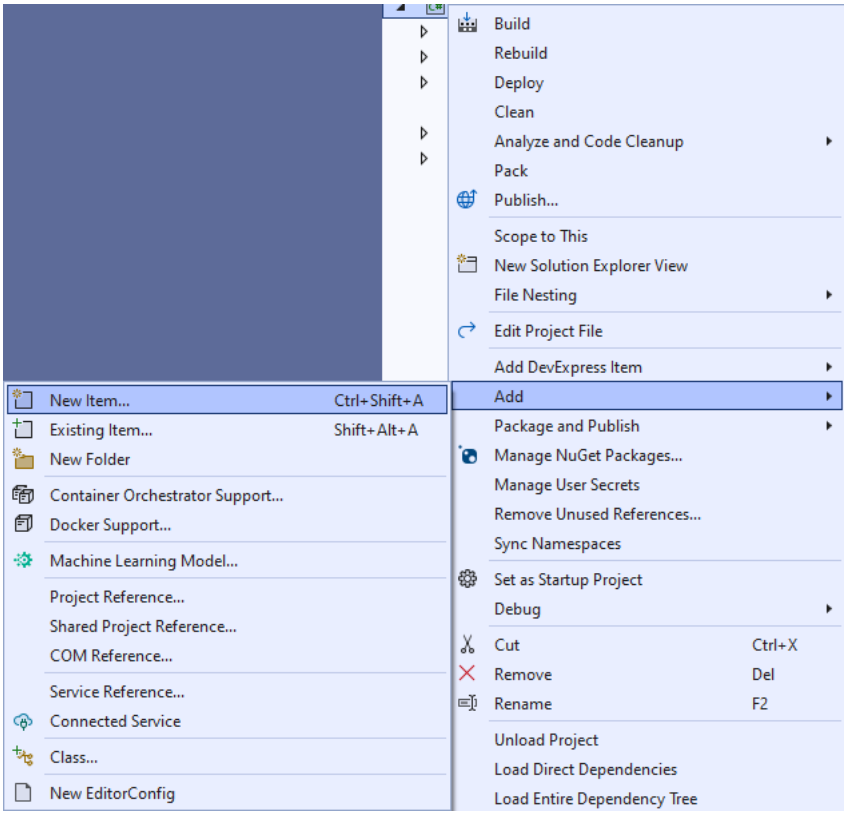


After that in **Configure your new project** type in the **Project name** as *RadialLayout*, then select a Location and then select **Create** to start a new **Solution**.



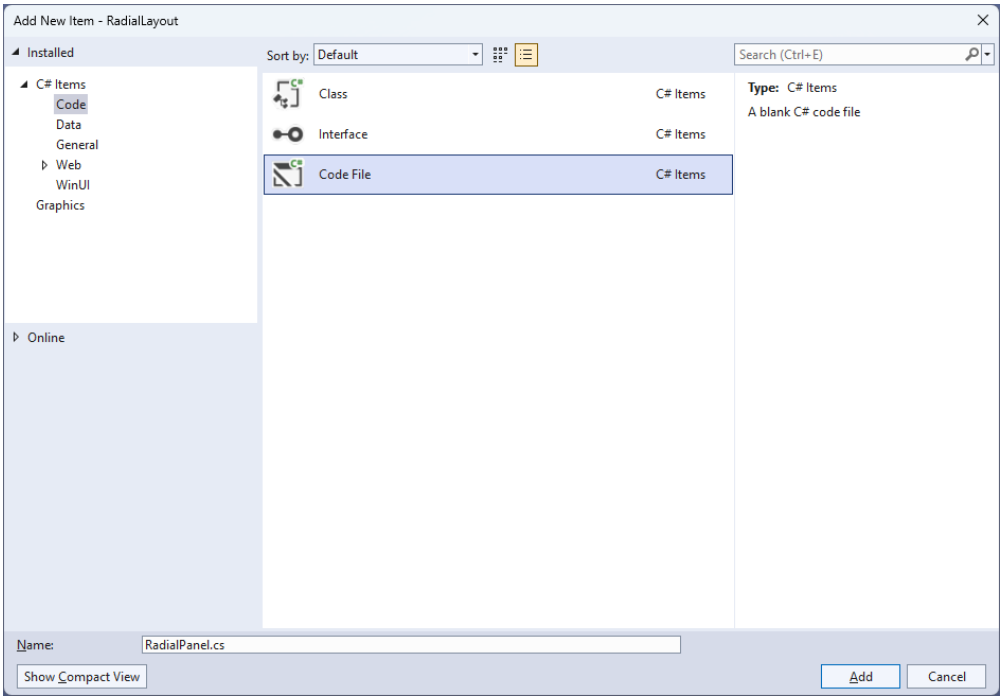
Step 2

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...**



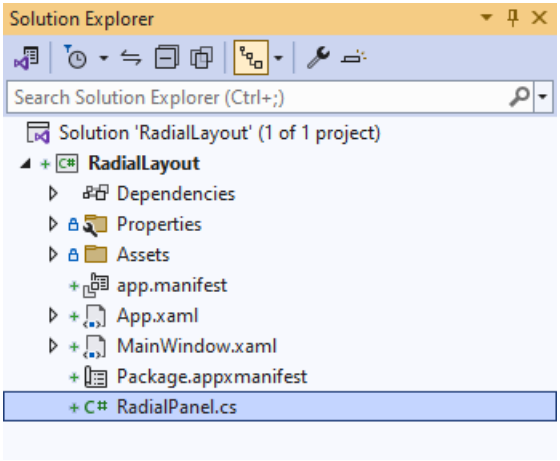
Step 3

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 *RadialPanel.cs* and then **Click** on **Add**.



Step 4

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



Step 5

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

```
using Microsoft.UI.Xaml;
using Microsoft.UI.Xaml.Controls;
using Microsoft.UI.Xaml.Media;
using System;
using Windows.Foundation;

namespace RadialLayout;

public class RadialPanel : Panel
{
    // Dependency Properties & Properties

    // Measure Override Method

    // Arrange Override Method

}
```

There are **`using`** statements for the **User Control**, a **`namespace`** for **`RadialLayout`** along with a **`class`** of **`RadialPanel`** that will represent the **User Control** and **Inherits** the **`class`** of **`Panel`**.

Step 6

Then in the **`namespace`** of **`RadialLayout`** in the **`class`** of **`RadialPanel`** after the **Comment** of **`// Dependency Properties & Properties`** type the following **Dependency Properties** and **Properties**:

```
public static readonly DependencyProperty ItemHeightProperty =
DependencyProperty.Register(nameof(ItemHeight),
typeof(double), typeof(RadialPanel),
new PropertyMetadata(double.NaN));

public static readonly DependencyProperty ItemWidthProperty =
DependencyProperty.Register(nameof(ItemWidth),
typeof(double), typeof(RadialPanel),
new PropertyMetadata(double.NaN));

public static readonly DependencyProperty IsOrientedProperty =
DependencyProperty.Register(nameof(IsOriented),
typeof(bool), typeof(RadialPanel),
new PropertyMetadata(false));

public double ItemHeight
{
    get { return (double)GetValue(ItemHeightProperty); }
    set { SetValue(ItemHeightProperty, value); }
}

public double ItemWidth
{
    get { return (double)GetValue(ItemWidthProperty); }
    set { SetValue(ItemWidthProperty, value); }
}

public bool IsOriented
{
    get { return (bool)GetValue(IsOrientedProperty); }
    set { SetValue(IsOrientedProperty, value); }
}
```

Dependency Properties or **Properties** for the **User Control** can be customised for the **Radial Panel**.

Step 7

While still in the **`namespace`** of **`RadialLayout`** in the **`class`** of **`RadialPanel`** after the **Comment** of **`// Measure Override Method`** type the following **Method**:

```
protected override Size MeasureOverride(Size availableSize)
{
    double itemWidth = ItemWidth;
    double itemHeight = ItemHeight;
    bool hasFixedWidth = !double.IsNaN(itemWidth);
    bool hasFixedHeight = !double.IsNaN(itemHeight);
    var itemSize = new Size(
        hasFixedWidth ? itemWidth : availableSize.Width,
        hasFixedHeight ? itemHeight : availableSize.Height);
    foreach (var element in Children)
    {
        element.Measure(itemSize);
    }
    return itemSize;
}
```

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

Step 8

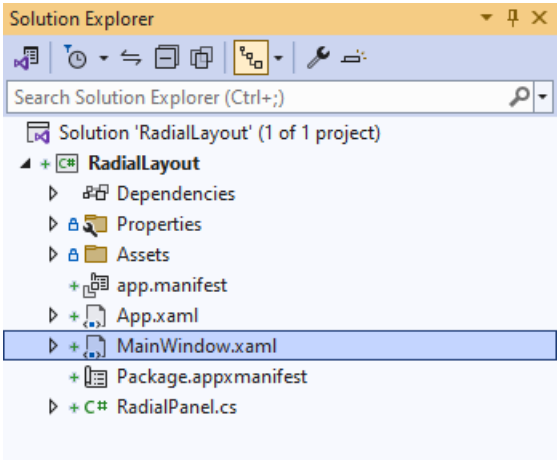
While still in the **namespace** of **RadialLayout** in the **class** of **RadialPanel** after the **Comment** of **// Arrange Override Method** type the following **Method**:

```
protected override Size ArrangeOverride(Size finalSize)
{
    double itemWidth = ItemWidth;
    double itemHeight = ItemHeight;
    bool hasFixedWidth = !double.IsNaN(itemWidth);
    bool hasFixedHeight = !double.IsNaN(itemHeight);
    double radiusX = finalSize.Width * 0.5;
    double radiusY = finalSize.Height * 0.5;
    int count = Children.Count;
    double deltaAngle = 2 * Math.PI / count;
    var centre = new Point(finalSize.Width / 2,
        finalSize.Height / 2);
    for (int i = 0; i < count; i++)
    {
        var element = Children[i];
        var elementSize = new Size(
        hasFixedWidth ? itemWidth : element.DesiredSize.Width,
        hasFixedHeight ? itemHeight : element.DesiredSize.Height);
        double angle = i * deltaAngle;
        double x = centre.X + radiusX * Math.Cos(angle)
            - elementSize.Width / 2;
        double y = centre.Y + radiusY * Math.Sin(angle)
            - elementSize.Height / 2;
        if (IsOriented)
            element.RenderTransform = null;
        else
        {
            element.RenderTransformOrigin = new Point(0.5, 0.5);
            element.RenderTransform = new RotateTransform()
            {
                Angle = angle * 180 / Math.PI
            };
        }
        element.Arrange(new Rect(x, y,
            elementSize.Width, elementSize.Height));
    }
    return finalSize;
}
```

The **Method** of **ArrangeOverride** will position the **Children** of the **Panel** and position them at different degrees of rotation around the centre of the **User Control** and **IsOriented** will either rotate them to face downwards or towards the centre of 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

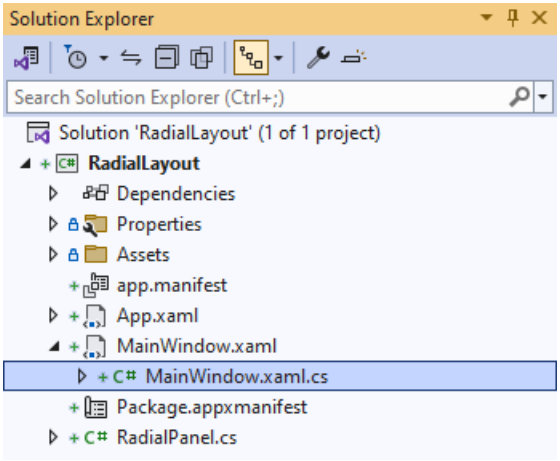
While still in the **XAML** for **MainWindow.xaml** above **`</Window>`**, type in the following **XAML**:

```
<local:RadialPanel Height="500" Width="500" IsOriented="True">
    <Rectangle Width="100" Height="100" Fill="Red"/>
    <Rectangle Width="100" Height="100" Fill="Orange"/>
    <Rectangle Width="100" Height="100" Fill="Yellow"/>
    <Rectangle Width="100" Height="100" Fill="Green"/>
    <Rectangle Width="100" Height="100" Fill="Cyan"/>
    <Rectangle Width="100" Height="100" Fill="Blue"/>
    <Rectangle Width="100" Height="100" Fill="Magenta"/>
    <Rectangle Width="100" Height="100" Fill="Purple"/>
</local:RadialPanel>
```

This **XAML** contains the **User Control** of **`RadialPanel`** with **`IsOriented`** set to **`True`** with 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:

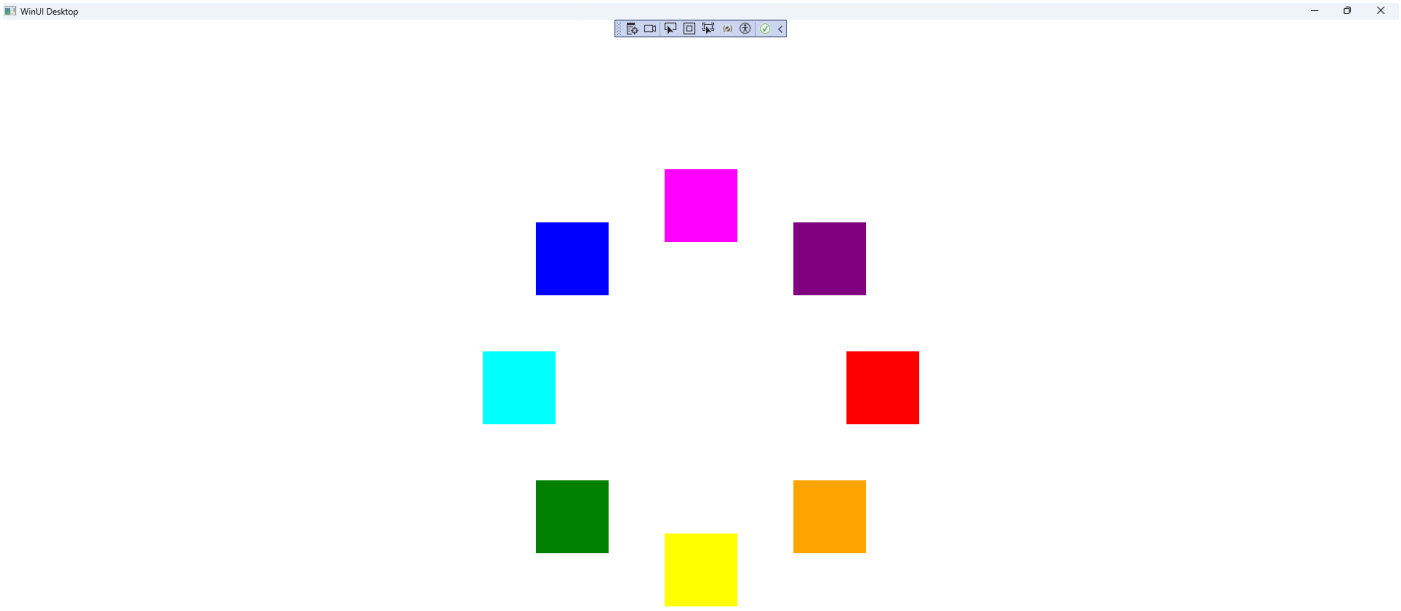
```
private void myButton_Click(object sender, RoutedEventArgs e)
{
    myButton.Content = "Clicked";
}
```

Step 14

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

Step 15

Once running you will see the **Radial 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!