

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

Windows App SDK

Lucky Dice





# Lucky Dice

**Lucky Dice** shows how you can create a simple random number game and display this using a control

from **NuGet** using the **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 *LuckyDice*, 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 **Manage NuGet Packages…**

Graphical user interface, application

Description automatically generated

## Step 3

Then in the **NuGet Package Manager** from the **Browse** tab search for **Comentsys.Toolkit.WindowsAppSdk** and then select **Comentsys.Toolkit.WindowsAppSdk by Comentsys** as indicated and select **Install**

Graphical user interface, text, application, email

Description automatically generated

This will add the package for **Comentsys.Toolkit.WindowsAppSdk** to your **Project**. If you get the **Preview Changes** screen saying **Visual Studio is about to make changes to this solution. Click OK to proceed with the changes listed below.** You can read the message and then select **OK** to **Install** the package, then you can close the **tab** for **Nuget: LuckyDice** by selecting the **x** next to it.

## Step 4

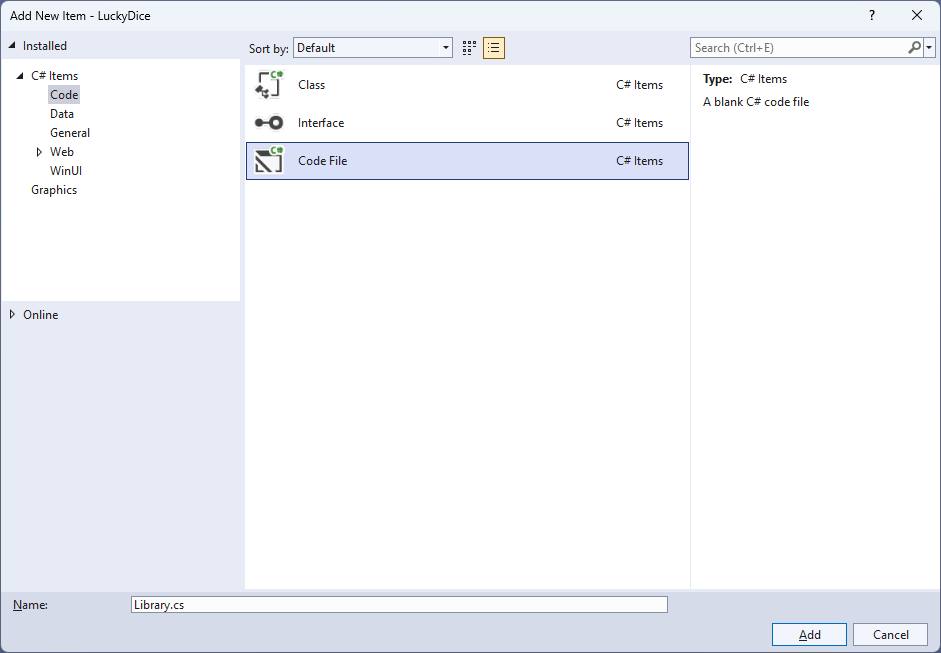
Then in **Visual Studio** within **Solution** **Explorer** for the **Solution**, right click on the **Project** shown below the **Solution** and then select **Add** then **New Item…**

Table

Description automatically generated with low confidence

## Step 5

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



## Step 6

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

using Comentsys.Toolkit.WindowsAppSdk;

using Microsoft.UI;

using Microsoft.UI.Xaml;

using Microsoft.UI.Xaml.Controls;

using Microsoft.UI.Xaml.Input;

using Microsoft.UI.Xaml.Media;

using System;

using Windows.UI;

public class Library

{

private readonly Random \_random = new((int)DateTime.UtcNow.Ticks);

public Dice Get(Color foreground, Color background)

{

Dice dice = new()

{

Margin = new Thickness(25),

CornerRadius = new CornerRadius(10),

Foreground = new SolidColorBrush(foreground),

Background = new SolidColorBrush(background)

};

dice.Tapped += (object sender, TappedRoutedEventArgs e) =>

((Dice)sender).Value = \_random.Next(1, 7);

return dice;

}

public void New(StackPanel panel)

{

panel.Children.Clear();

panel.Children.Add(Get(Colors.Red, Colors.WhiteSmoke));

panel.Children.Add(Get(Colors.Blue, Colors.WhiteSmoke));

}

}

The **Class** that has been defined in *Library.cs* has a **using** for the packageof **Comentsys.Toolkit.WindowsAppSdk** amongst others needed. Then there is **Random** which will be used to select randomised numbers from. There is a **Method** of **Get** which is used to create a **Dice** which is a controlin **Comentsys.Toolkit.WindowsAppSdk** and it has two **Parameters** for the **foreground** which will control the colour of the **Pips** and **background** for the **Dice**. An **Event** handler is attached to the **Tapped** event for the **Dice** using **Arrow Syntax** with **=>** which is a useful shorthand. This will set the **Property** for the **Value** of the **Dice** from *1* to below *7*. This is then used in the **Method** of **New** which takes a **StackPanel** as a **Parameter** which it then **Clears** and adds two **Dice** with some colours provided for the **foreground** and **background**.

## Step 7

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

## Step 8

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 9

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

<Grid>

<Viewbox>

<StackPanel Margin="50" Name="Display" Orientation="Horizontal"

HorizontalAlignment="Center" VerticalAlignment="Center" Loaded="New"/>

</Viewbox>

<CommandBar VerticalAlignment="Bottom">

<AppBarButton Icon="Page2" Label="New" Click="New"/>

</CommandBar>

</Grid>

This **XAML** contains a **Grid** with a **Viewbox** which will **Scale** a **StackPanel**. It has a **Loaded** event handler for **New** which is also shared by the **AppBarButton**.

## Step 10

|  |  |
| --- | --- |
| 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 11

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 12

Once **myButton\_Click(...)** has been removed, type in the following **Code** below the end of the **Constructor** of **public MainWindow() { ... }**:

private readonly Library \_library = new();

private void New(object sender, RoutedEventArgs e) =>

\_library.New(Display);

Here an **Instance** of **Library** is created then below this is the **Method** of **New** that will be used with **Event Handler** from the **XAML**, this **Method** uses **Arrow** **Syntax** with the **=>** for an expression bodywhich is useful when a **Method** only has one line.

## Step 13

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

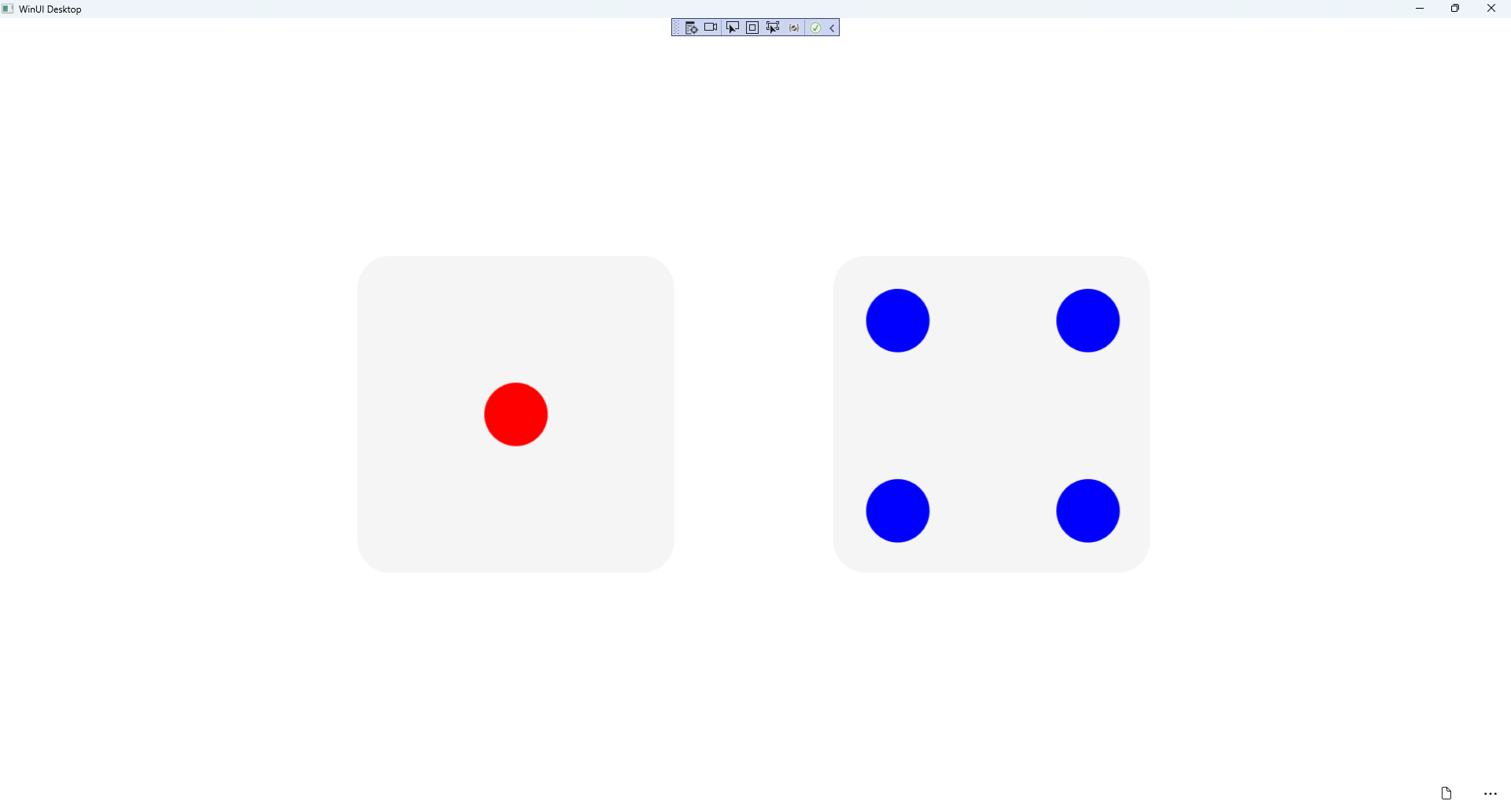
## Step 14

Once running you should see the **Dice** elements.

****

## Step 15

You can **Click** on either **Dice** to *“roll”* them and see what outcome you get or select *New* to reset



## 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)! |  |