

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

Windows App SDK

Fruit Game





# Fruit Game

**Fruit Game** shows how you can create a simple slots-like game to match three symbols together using

emoji and a toolkit 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 *FruitGame*, 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.

## Step 4

Then while still in the **NuGet Package Manager** from the **Browse** tab search for **Comentsys.Assets.FluentEmoji** and then select **Comentsys.Assets.FluentEmoji by Comentsys** as indicated and select **Install**

Graphical user interface, text, application, email

Description automatically generated

This will add the package for **Comentsys.Assets.FluentEmoji** 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: FruitGame** by selecting the **x** next to it.

## Step 5

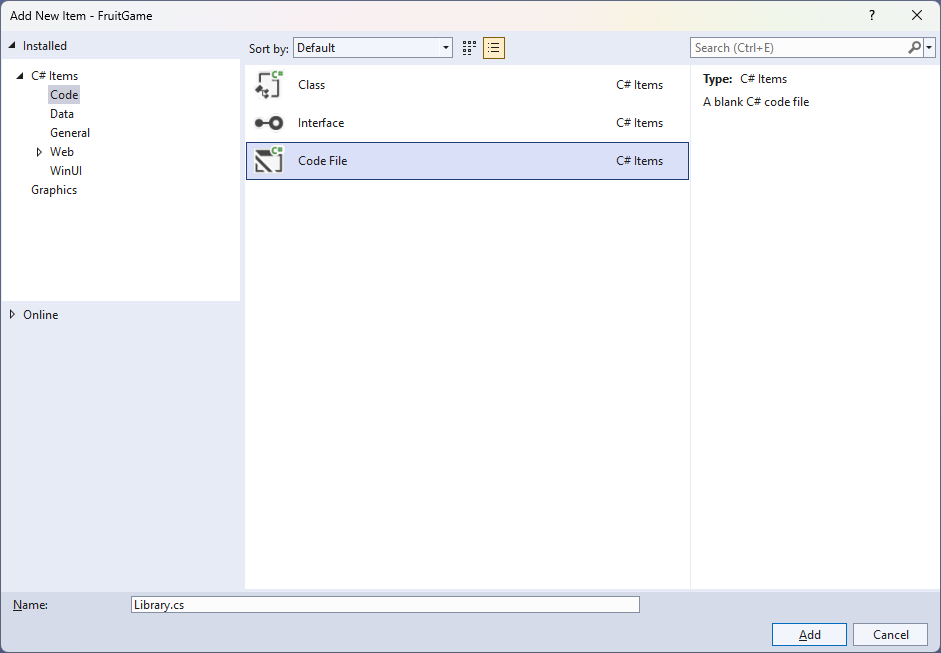
Then in **Visual Studio** within **Solution** **Explorer** for the **Solution**, right click on the **Project** shown below the **Solution** and then select **Add** then **New Item…**

Table

Description automatically generated with low confidence

## Step 6

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 7

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

using Comentsys.Assets.FluentEmoji;

using Comentsys.Toolkit.WindowsAppSdk;

using Microsoft.UI.Xaml;

using Microsoft.UI.Xaml.Controls;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

public class Library

{

private const string title = "Fruit Game";

private const int delay\_duration = 250;

private const int size = 3;

private readonly Dictionary<int, FluentEmojiType> \_options = new()

{

{ 0, FluentEmojiType.SlotMachine },

{ 1, FluentEmojiType.GreenApple },

{ 2, FluentEmojiType.Grapes },

{ 3, FluentEmojiType.Lemon },

{ 4, FluentEmojiType.Cherries },

{ 5, FluentEmojiType.Banana },

{ 6, FluentEmojiType.Melon },

{ 7, FluentEmojiType.Tangerine },

{ 8, FluentEmojiType.Bell }

};

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

private int \_spins;

private Dialog \_dialog;

private StackPanel \_panel = new();

// Choose, Option & Set

// Play

// Add, Layout & New

}

**Class** defined so far *Library.cs* has **using** for packageof **Comentsys.Toolkit.WindowsAppSdk** and others. It also has **Constants** to represent things needed in the game and there are **Variables** to keep track of values used in the game and elements for the look-and-feel of the game.

## Step 8

Still in the **Class** for *Library.cs* after the **Comment** of **// Choose, Option & Set** type the following **Methods**:

private List<int> Choose(int minimum, int maximum, int total)

{

var choose = new List<int>();

var values = Enumerable.Range(minimum, maximum).ToList();

for (int index = 0; index < total; index++)

{

var value = \_random.Next(0, values.Count);

choose.Add(values[value]);

}

return choose;

}

private Viewbox Option(int index, int option) => new()

{

Child = new Asset

{

Name = $"{index}",

AssetResource = FlatFluentEmoji.Get(\_options[option])

}

};

private void Set(int index, int option) =>

(\_panel.FindName($"{index}") as Asset)

.AssetResource = FlatFluentEmoji.Get(\_options[option]);

**Choose** is used to select a set of random numbers as you can have the same value for multiple slots these are not unique. **Option** is used to get the assets needed for the emoji that will represent the values for the slots and **Set** is used to update the asset being displayed in the slots.

## Step 9

While still in the **Class** for *Library.cs* after the **Comment** of **// Play** type in the following **Method**:

private async void Play()

{

var values = Choose(1, \_options.Count - 1, size);

for(int index = 0; index < size; index++)

{

for(int option = 1; option <= values[index]; option++)

{

Set(index, option);

await Task.Delay(delay\_duration);

}

}

\_spins++;

if (values.All(a => a.Equals(values.First())))

{

var content = new StackPanel()

{

Orientation = Orientation.Vertical

};

content.Children.Add(new TextBlock()

{

HorizontalTextAlignment = TextAlignment.Center,

Text = $"Spin {\_spins} matched"

});

var fruit = new StackPanel()

{

Height = 100,

Orientation = Orientation.Horizontal

};

foreach(int value in values)

{

fruit.Children.Add(new Asset

{

AssetResource = FlatFluentEmoji.Get(\_options[value])

});

}

content.Children.Add(fruit);

\_dialog.Show(content);

\_spins = 0;

}

}

**Play** is used with **Choose** to get the random slots which will then be displayed with **Set** if they all match a **Dialog** will be displayed showing which slot matched.

## Step 10

While still in the **Class** for *Library.cs* after the **Comment** of **// Add, Layout & New** type the following **Methods**:

private void Add(StackPanel panel, int index)

{

Button button = new()

{

Width = 150,

Height = 150,

Margin = new Thickness(5),

Content = Option(index, 0)

};

button.Click += (object sender, RoutedEventArgs e) =>

Play();

panel.Children.Add(button);

}

private void Layout(StackPanel panel)

{

panel.Children.Clear();

for (int index = 0; index < size; index++)

{

Add(panel, index);

}

}

public void New(StackPanel panel)

{

\_spins = 0;

\_dialog = new Dialog(panel.XamlRoot, title);

\_panel = panel;

Layout(\_panel);

}

**Add** will create the **Buttons** to be **Clicked** by setting the event handler for **Click** to the **Method** for **Play**, **Layout** will create the layout for the game and **New** will start a new game.

## Step 11

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

## Step 12

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 13

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 14

|  |  |
| --- | --- |
| 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 15

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 16

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);

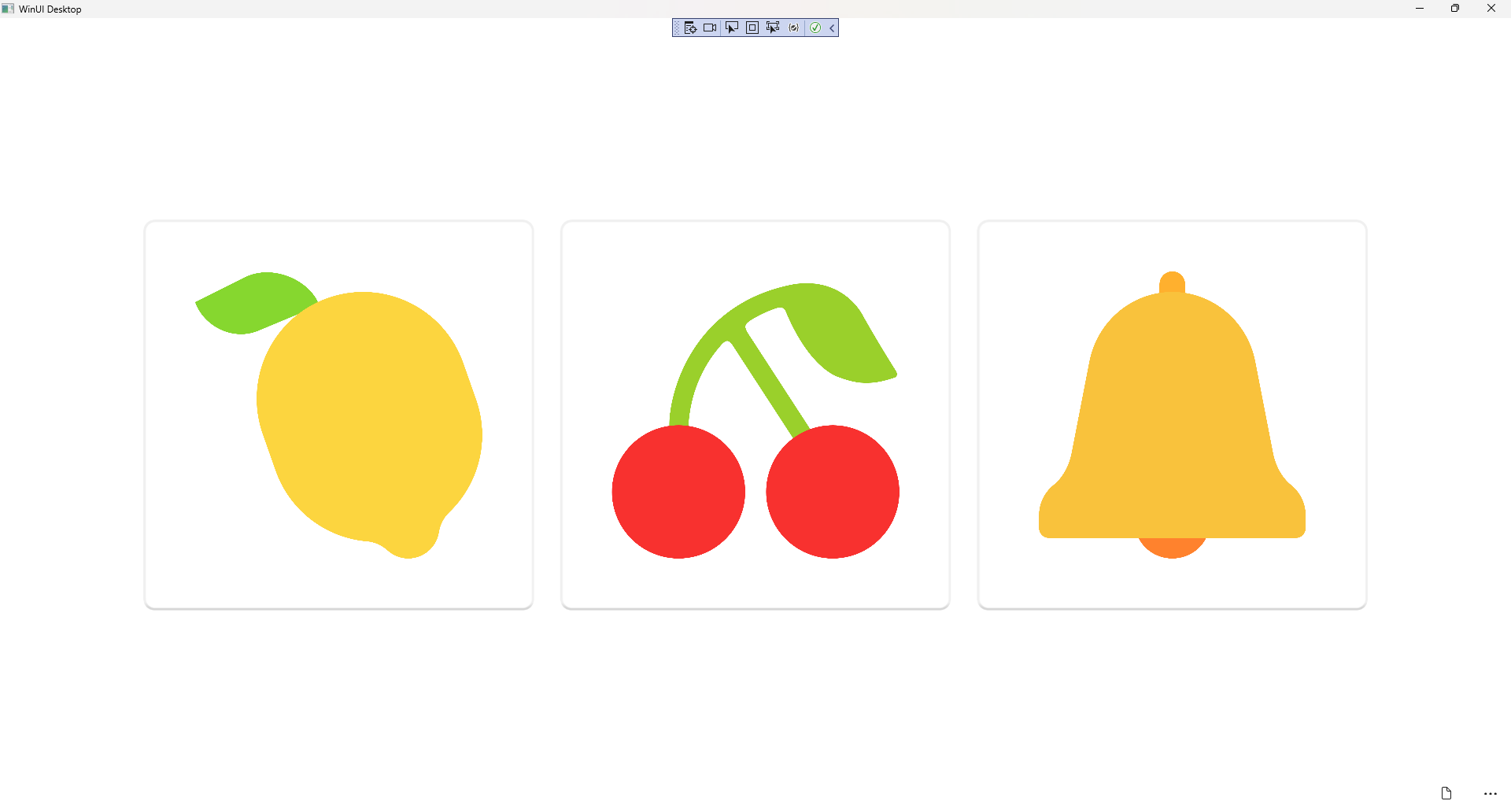
Here an **Instance** of the **Class** of **Library** is created then below this is the **Method** of **New** that will be used with **Event Handler** from the **XAML**, this **Method** uses Arrow Syntax with the **=>** for an Expression Bodywhich is useful when a **Method** only has one line.

## Step 17

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

## Step 18

Once running you can then select any **Button** to spin the slots and if three match you win, try to see how many times you can, or you can select *New*to start a new game.

****

## Step 19

|  |  |
| --- | --- |
| 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)! |  |