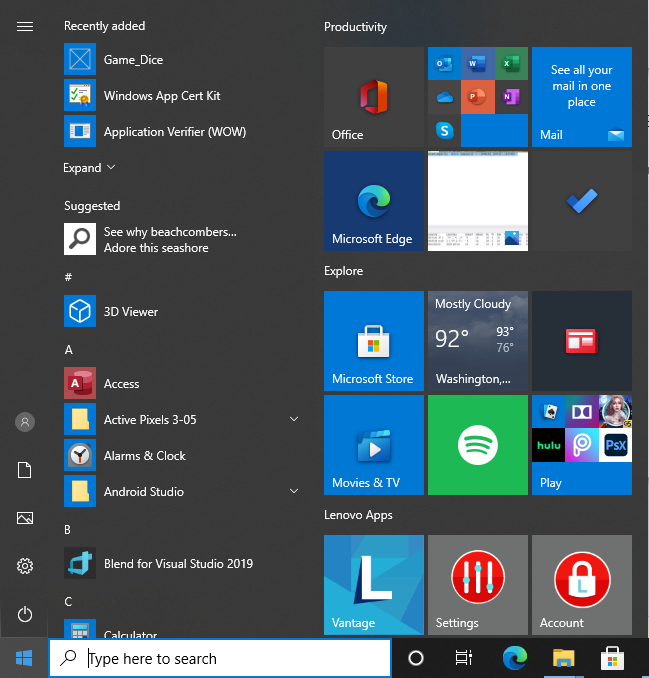
**About IDE for Development Software(Mobile and Web Applications)**

1. Open **Microsoft Visual Studio 2019** in windows version 2010



**Descriptions** –

When you are staring development so you may already be familiar with the IDE like Microsoft Development Environment from previous work/study or your own research.

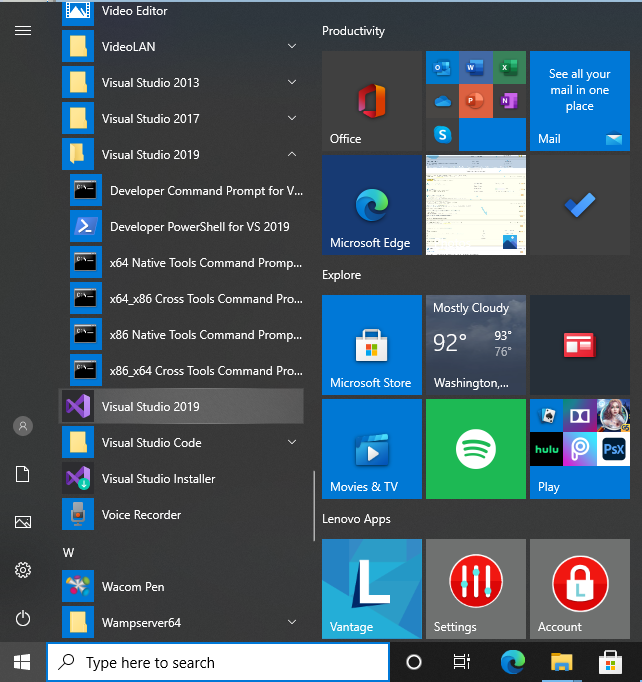
1. *I Created This for windows for my session and the name of project is dice game.*

***The Project name I choose Dice Game between two players.***

In this session we will look at the basic operation of the platform, a C# Windows Platform application project template and finally some of the extended features of the Integrated Development Environment (VS 2019).

**How to Open the Development Environment…….**

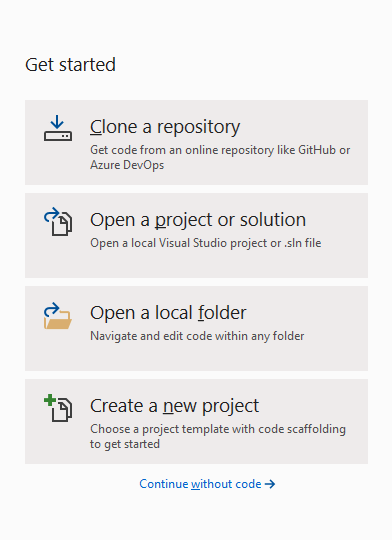
To open the IDE - Microsoft Development environment, follow the below steps.

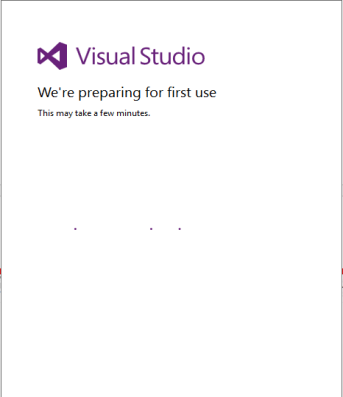
1. Click the on the **Windows**  button in the lower left corner of your screen
2. Or press the  **Windows Key** on your keyboard.
3. When the programs list appears.
4. Scroll down the programs list until you type the **V or you can search with 2019** in search box and then select (by clicking on it) the **Visual Studio 2019** IDE.
5. We can Open VS 2019 in bottom tool bar as below screen.

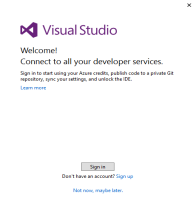


1. The Visual Studio load screen will appear, and the application will load.





1. After Installation you need to register or login with Microsoft account to run VS(IDE) or asked to connect all your developer services, for now just select the **Not now, maybe later** option by clicking on it.

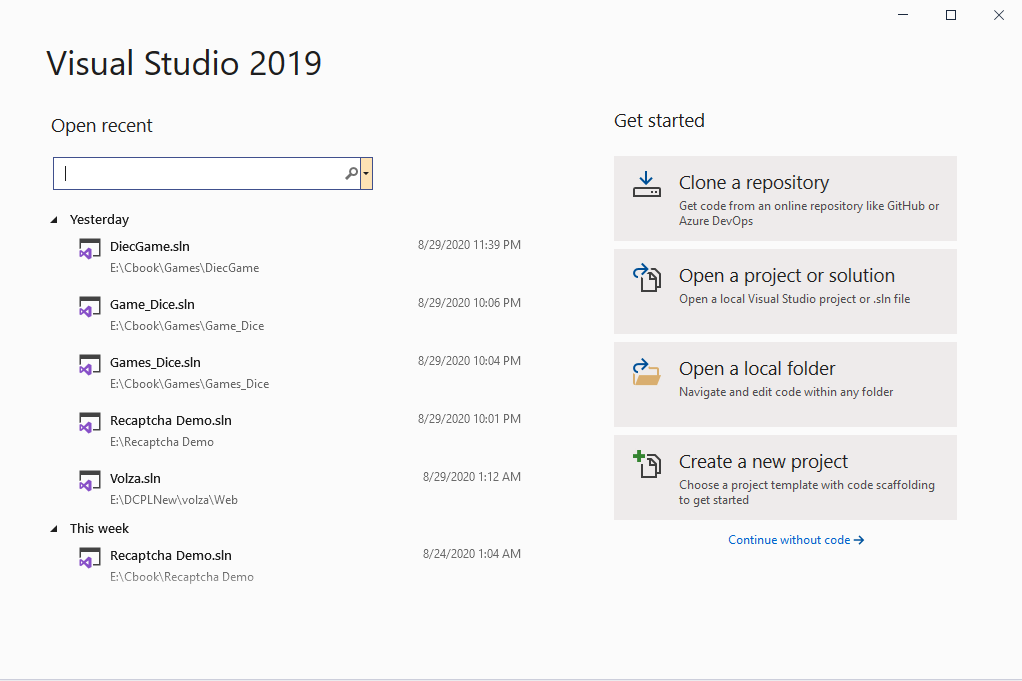


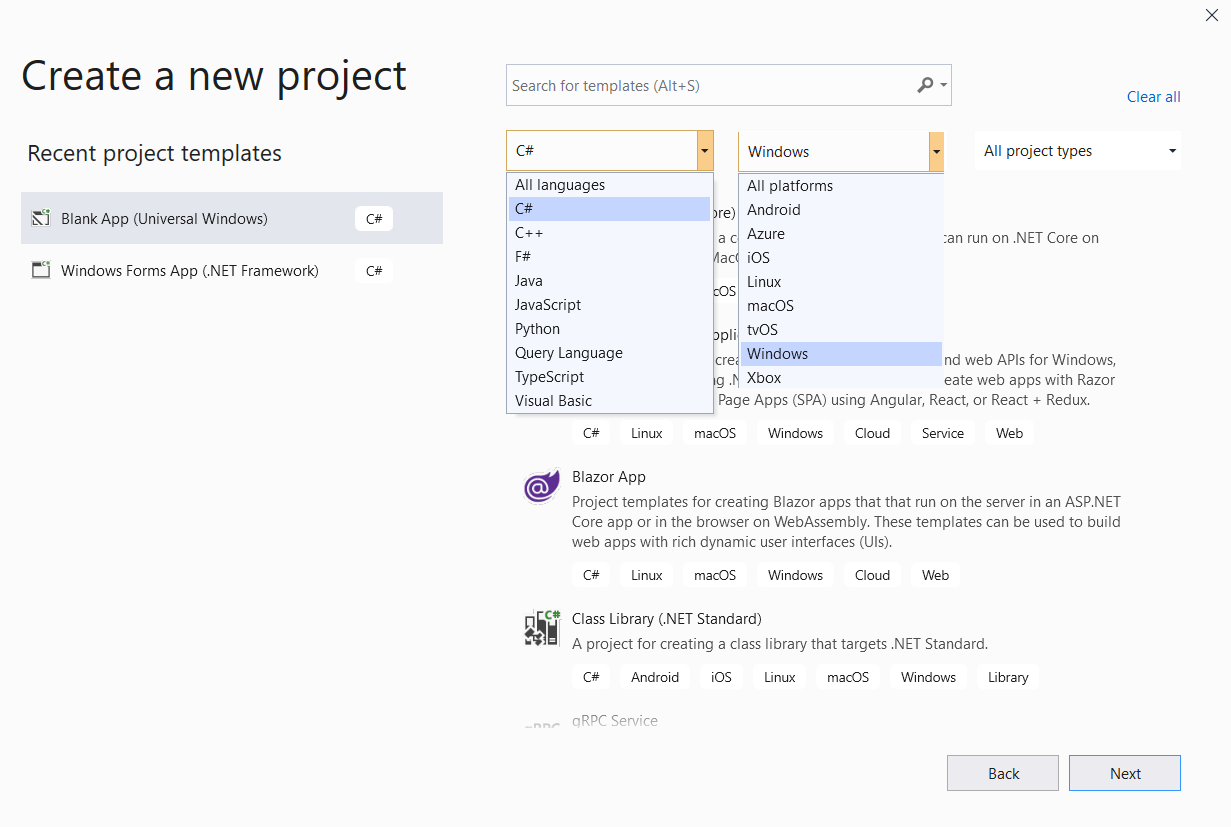
1. **If you are Opening** **the Development Environment for the First Time…… Then IDE will be clean with no existing open projects Like below screen….**

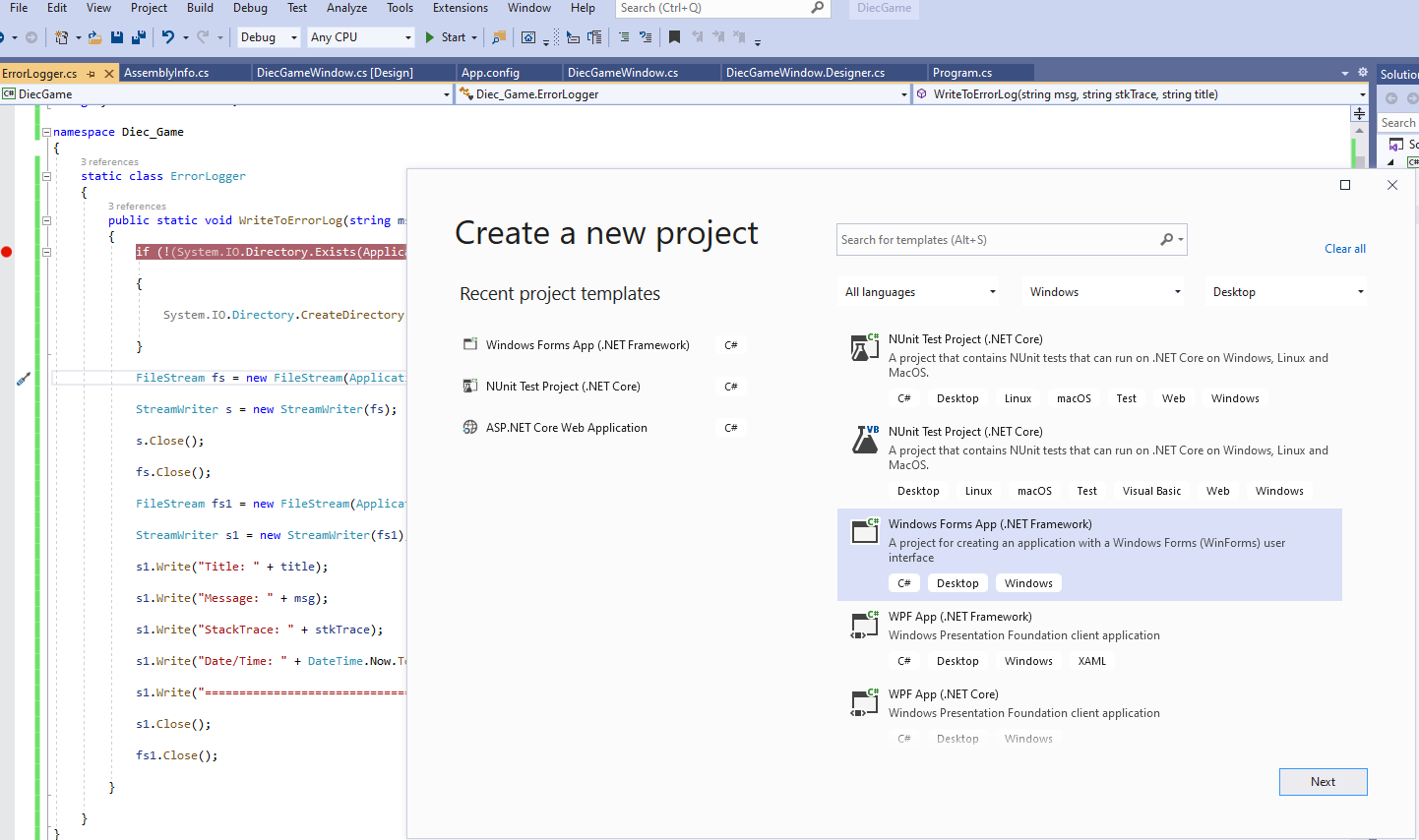
****

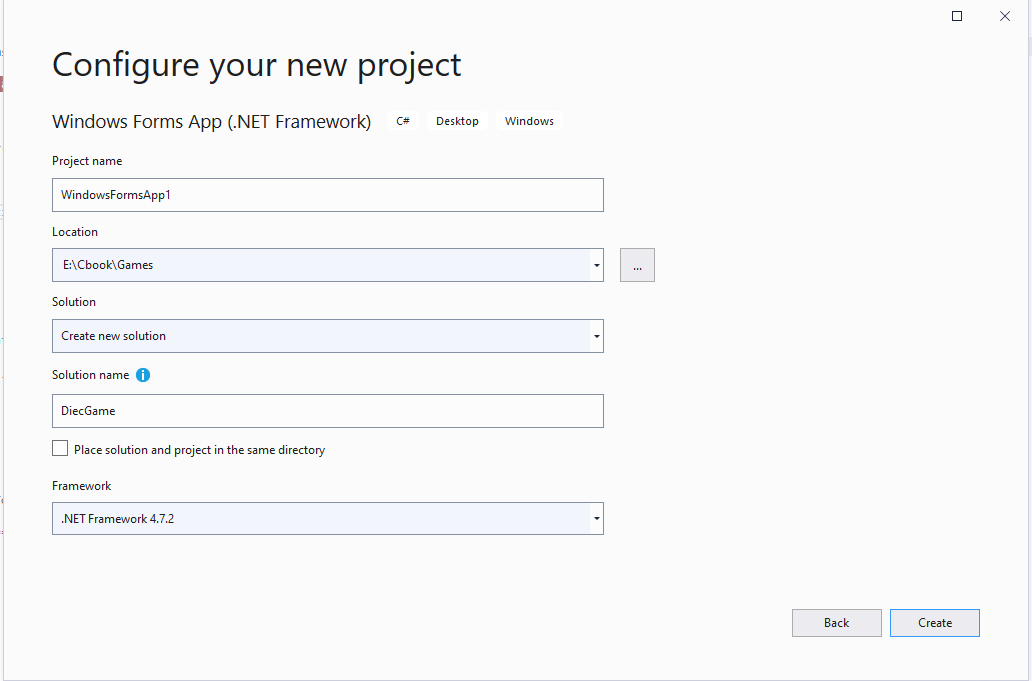
* **Select Visual C# from the Development settings: list.**
* **Select the Blue option from the Choose your color theme options.**
* **Click the Start Visual Studio button to open the development Environment.**
* **It will take a while to configure the environment.**
* **As the prompt will tell you.**

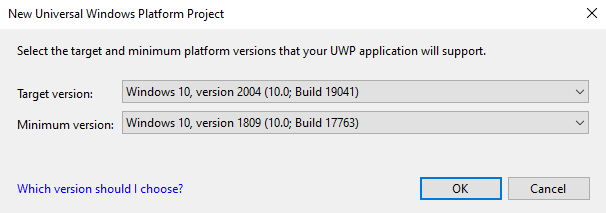
1. **If you are opened** **the Development Environment before it …… Then IDE will show previous projects with existing/recent open projects Like below screen….**



1. At some point the **Create a New Project** screen will load up, allowing you to select a project type to work with.
2. If this is the first time you have created a project, select **C*#*** and **Windows** from the search list options. This refines the list of project builds for you.



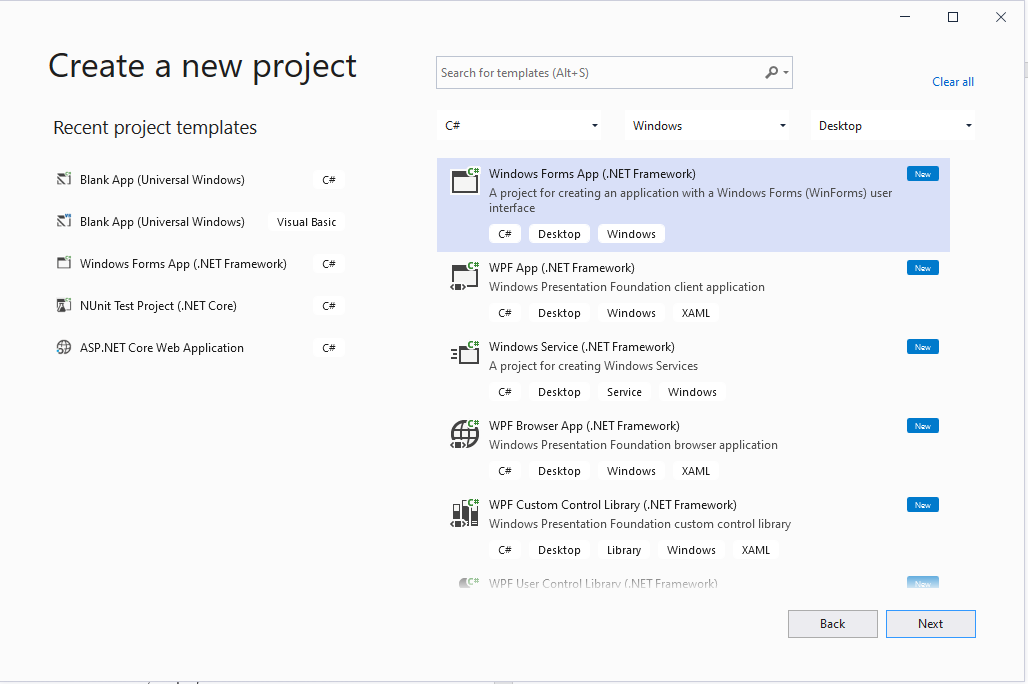
1. We will be building a Windows application so click the Window Form App (.Net Framework) option and then click the Next button.
2. When the Configure your new project window appears, enter a name for your project in the Project name text box (I used Dice Game), type a location for your project file in the Location text box or use the directory finder … button then tick the Place solution and project in the same directory check box.
3. When you are decided with your name and location of your project click the **Create** button.
4. Your projects should be built on the latest (or close too) Windows 10 version - the system in class should be set up for you.
5. So click **OK** when the build version window appears.
6. In the labs we will use the **Windows 10.**



Notes:

* If you are using your own laptop or when the labs update - see the notes in Appendix A for Home Use under the Checking the build and setting the developer mode section.
* If the settings are different in the Labs (J-block) - point this out to your lecturer.

1. At this point the development environment will load.

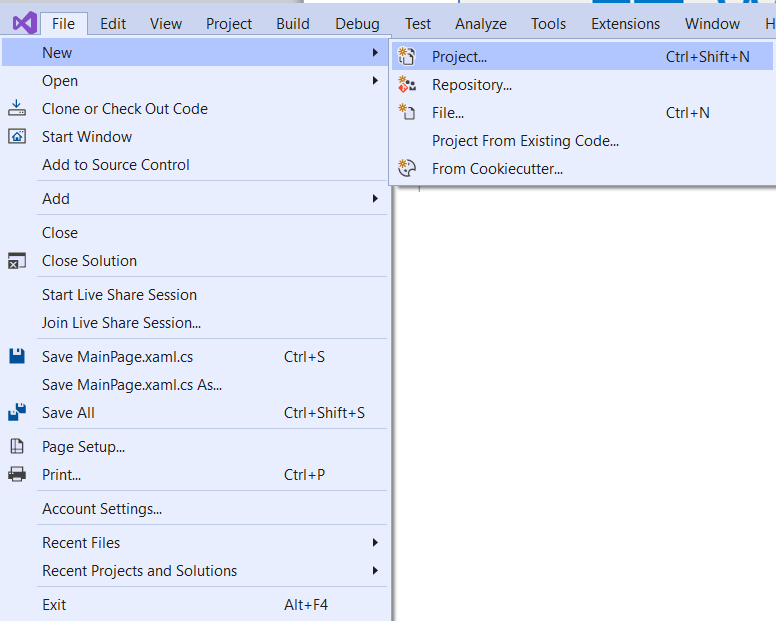


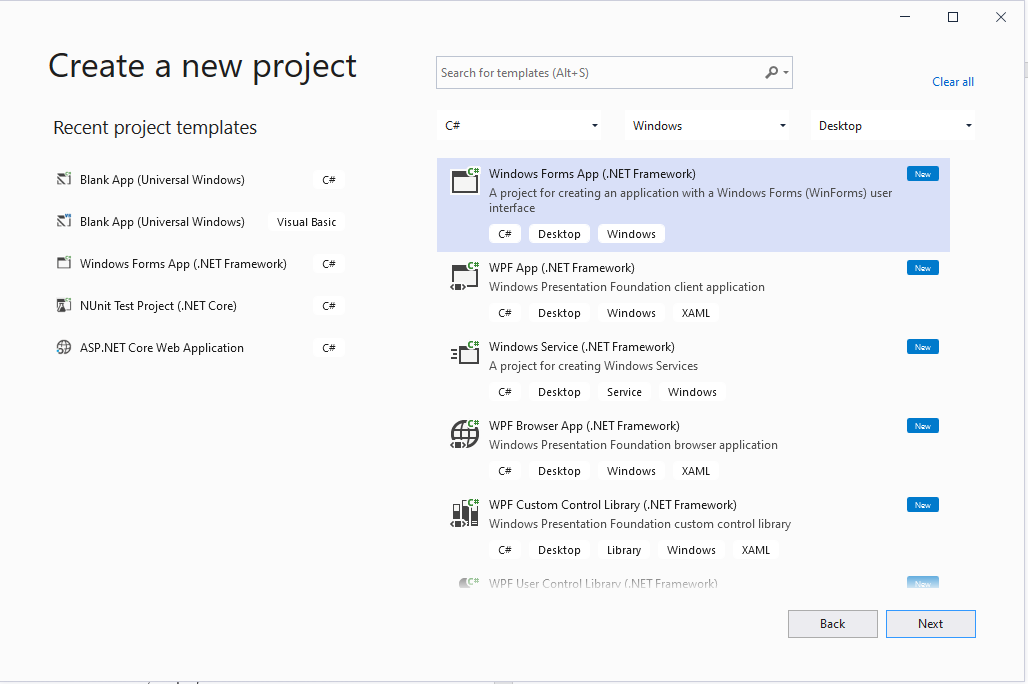
**The Development Environment**

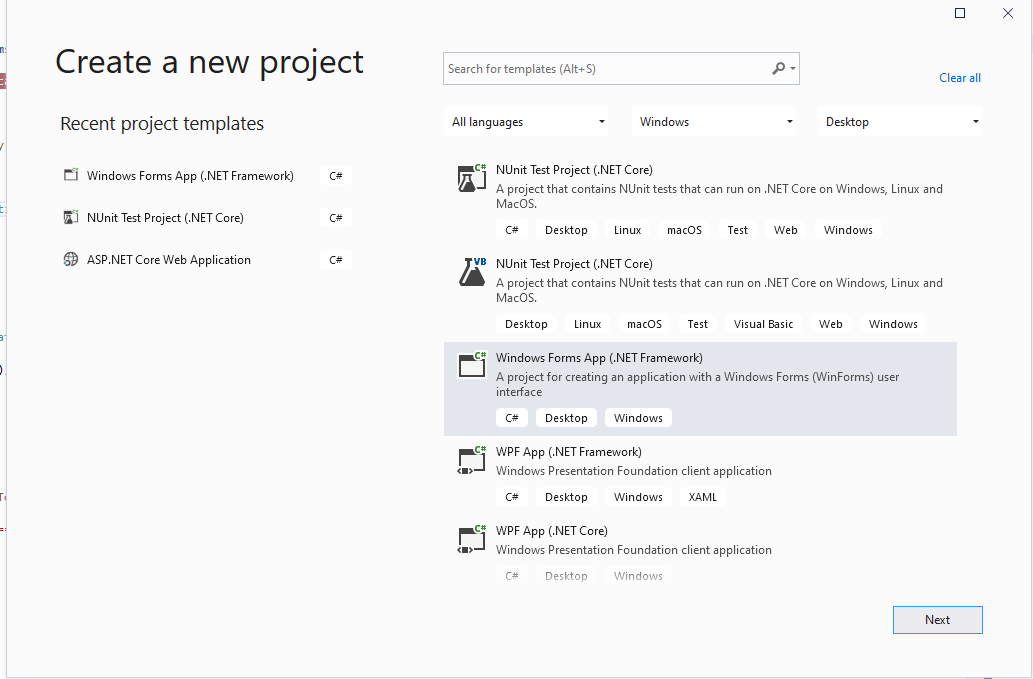
1. When developing applications in the Microsoft environment there is a large set of project templates that can be used to set up project types quickly. The development done during this part of the course will focus on the Windows Platform (Window Form App) project type (this will allow us to develop interfaces and also provide a basic application to add our code).
2. We currently have the environment loaded, but let’s look at an alternative way of opening a project. This project type can be accessed as a ‘Project Type’ from the ’New Projects’ menu in the Integrated Development Environment and offers a pre-made entry point for developing C# applications using Windows Platform interfaces.

**Creating a Window Form Project.**

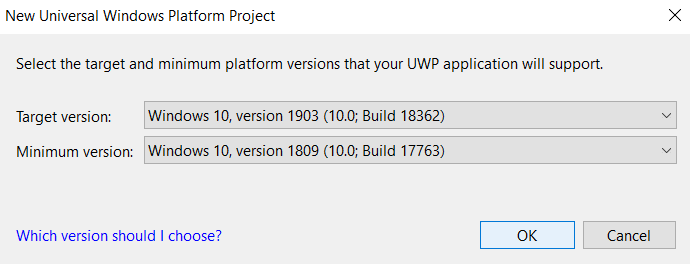
1. To create a project, open visual studio and then select the **File > New > Project…** option.



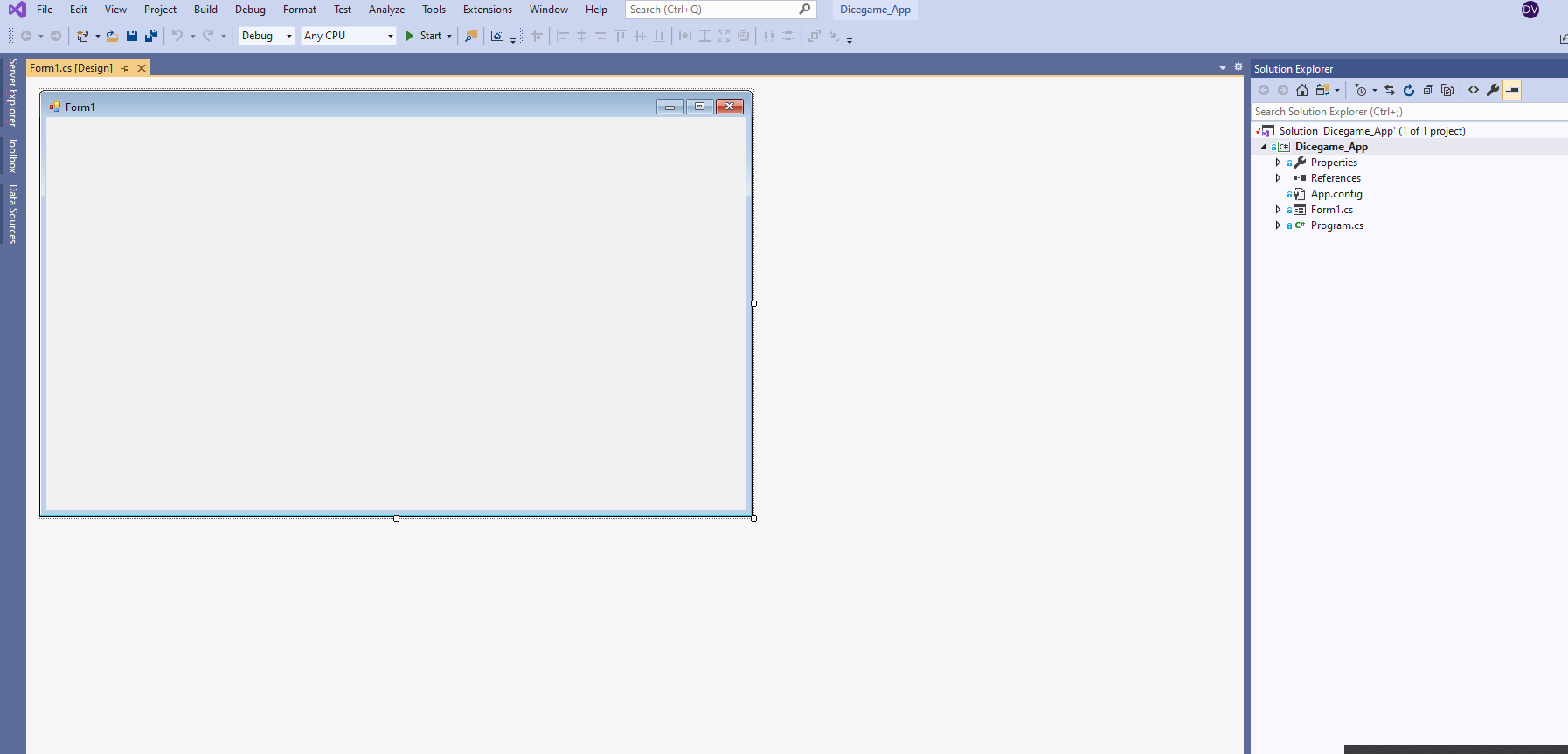
1. To opens and Create a new project window. As we are building a Windows application click the Window Form App option on top as in screen shot and then click the Next button. 
2. When the **Configure your new project** window appears, enter a name for your project in the **Project name** text box (I used Dice Game), type a location for your project file in the **Location** text box or use the directory finder **… button** thentick the **Place solution and project in the same directory** check box.
3. When you have decided with your name and location of your project click the **Create** button.



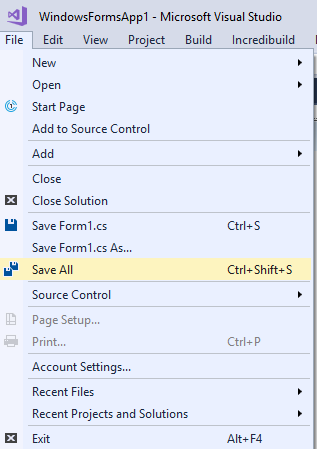
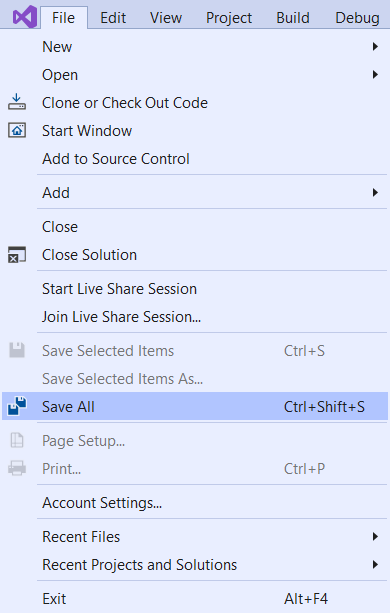
1. Your projects should be built on the latest (or close too) Windows 10 version - the system in class should be set up for you.
2. So, click **OK** when the build version window appears.
3. During configuration the Microsoft Visual Studio load up screen may be displayed.



1. At this point the development environment will load



1. **Saving a Project**

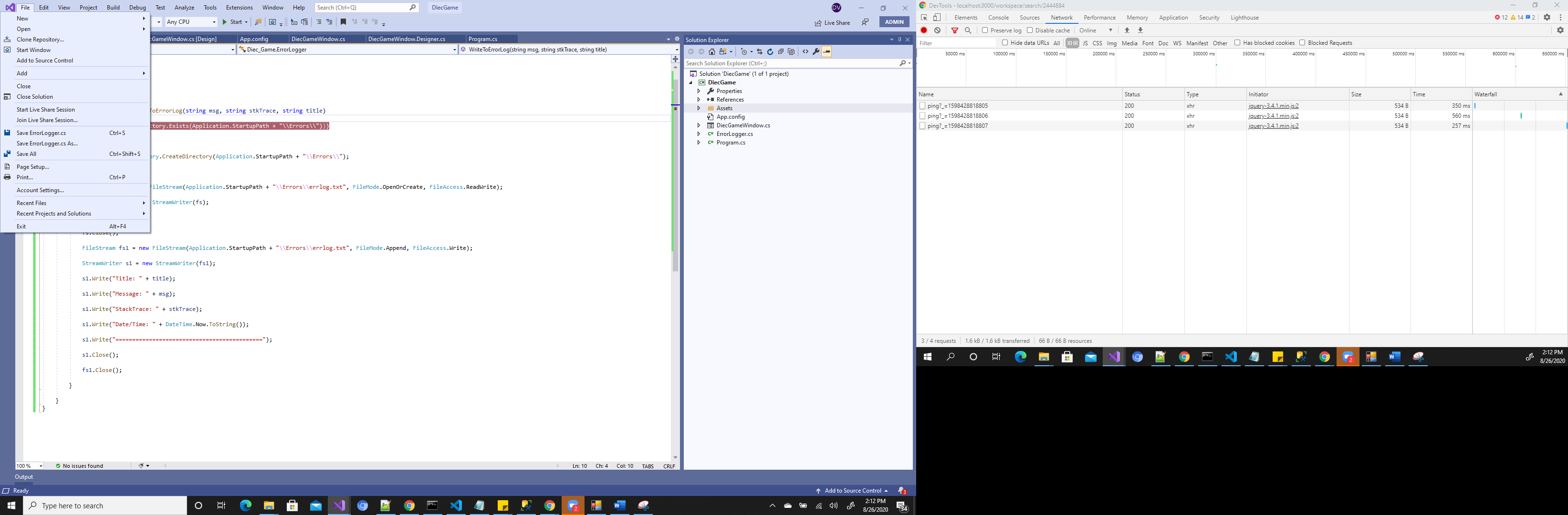
* ****The **File > Save All** option can be used to save the project as you work on it to the folder set up when the project was created.
* The project is saved automatically (to the folder set up when the project was created) every time the project is ***built*** or ***run***.
* When You doing code make sure all work should be save by ctrl+S.

1. **Closing a Project**

* A project can be closed using the **File > Exit** option, on the keyboard by using Alt+F4 or by clicking the Project Workspace **X** button.
* The project is *not saved* automatically when you exit so be careful to save your work before you exit.

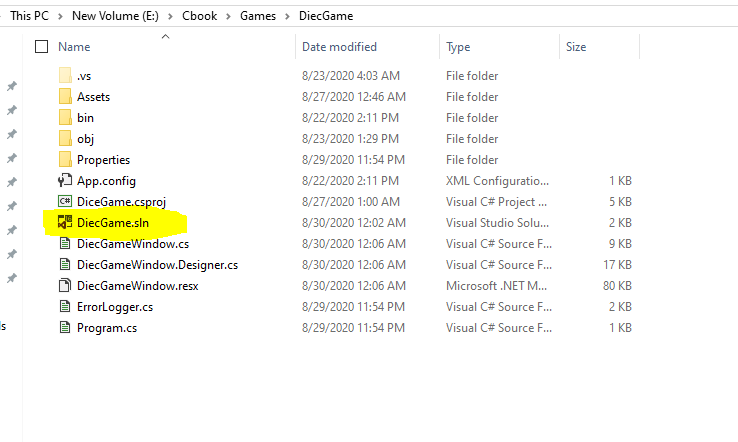


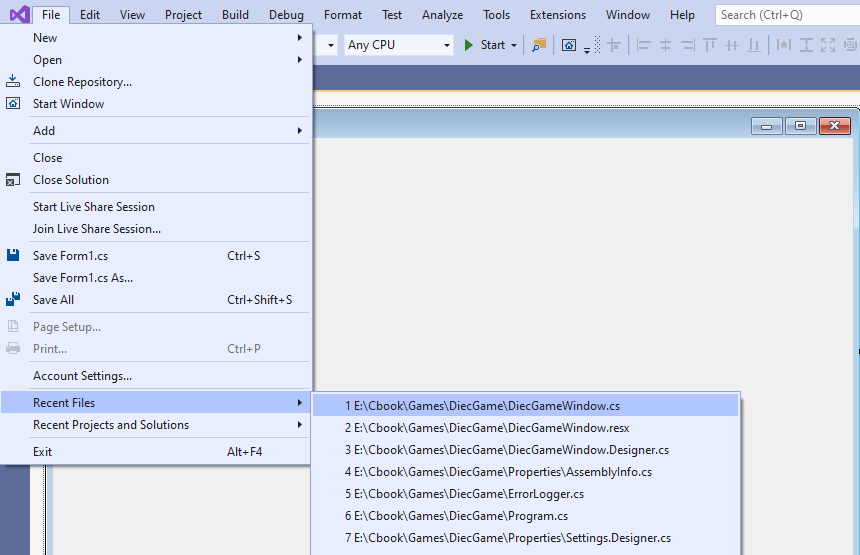
1. If you have made changes to your code the save reminder prompt should appear, make sure you select the **Save** button option to save your work. And Debug the app or check response

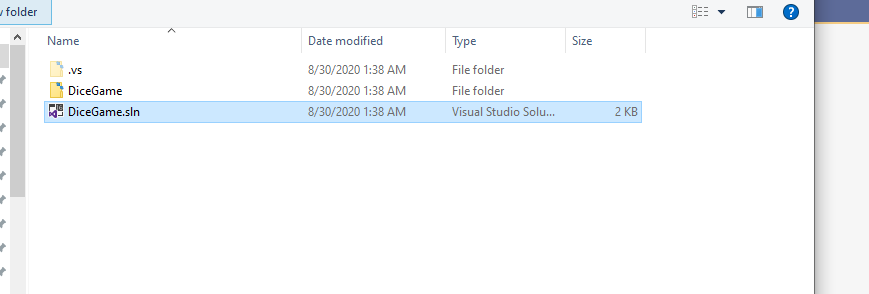


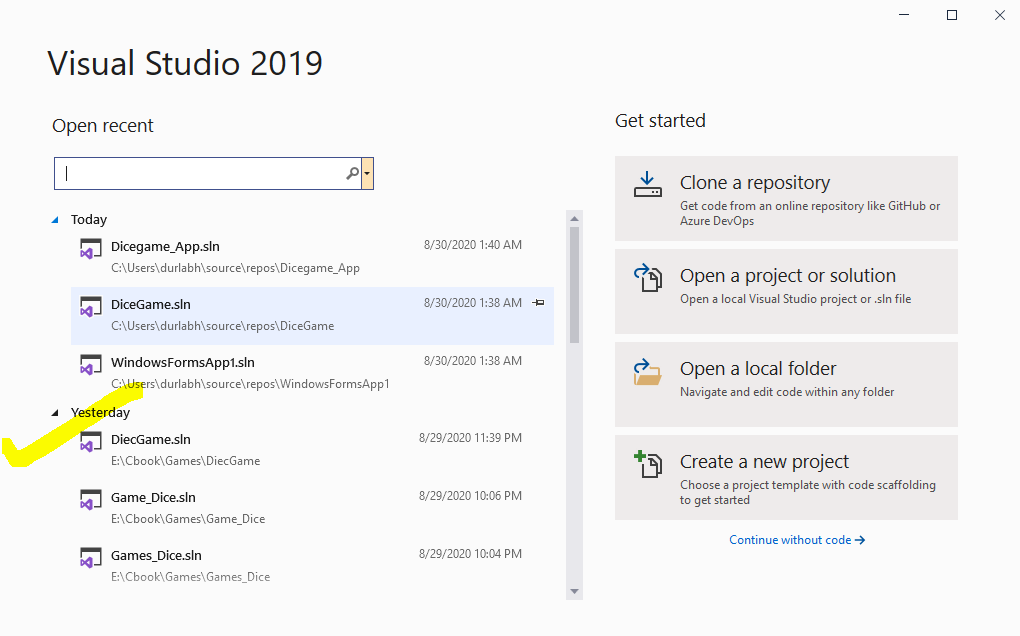
1. Opening an Existing Project –

* We have options to open project…
* From the start screen, by selecting a project from the Open resent project list.
* From the start screen, by selecting the **Open a project or solution** option.
* Then navigating to the correct project folder, selecting the **.sln** file and clicking on the **Open** button.
* An existing project can be opened from the main page of the IDE.









1. **What’s in a Project Workspace?**

Before we start to investigate the IDE, we need to make sure that we have the basic windows present.

Select the **View tab** and **click** on the following options:

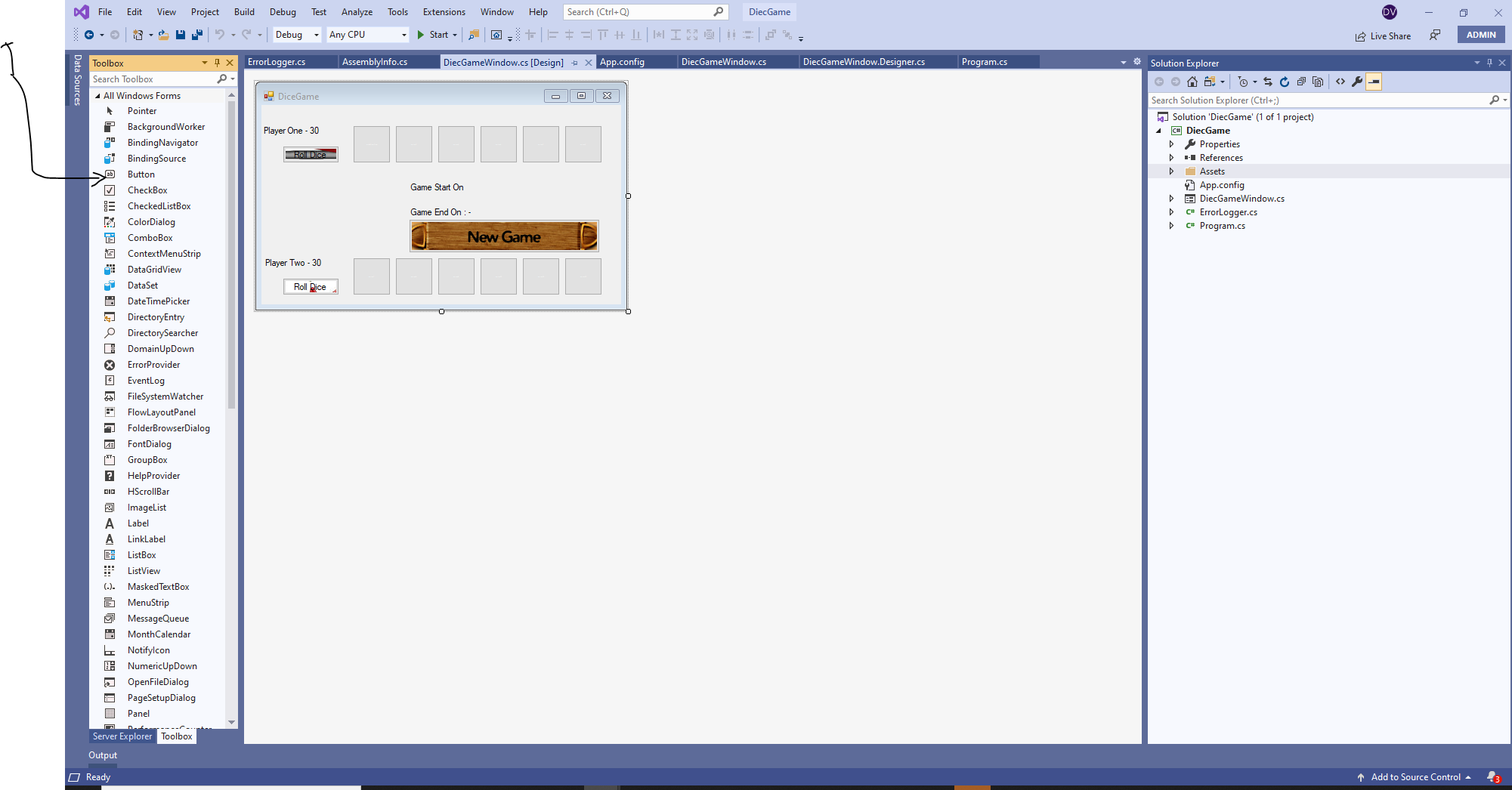
* Error List
* Output
* Toolbox
* Properties Window

**Double Click** on the following files in the Solution Explorer window:

* DiceGameWindow.cs
* DiceGameWindow.cs[design]

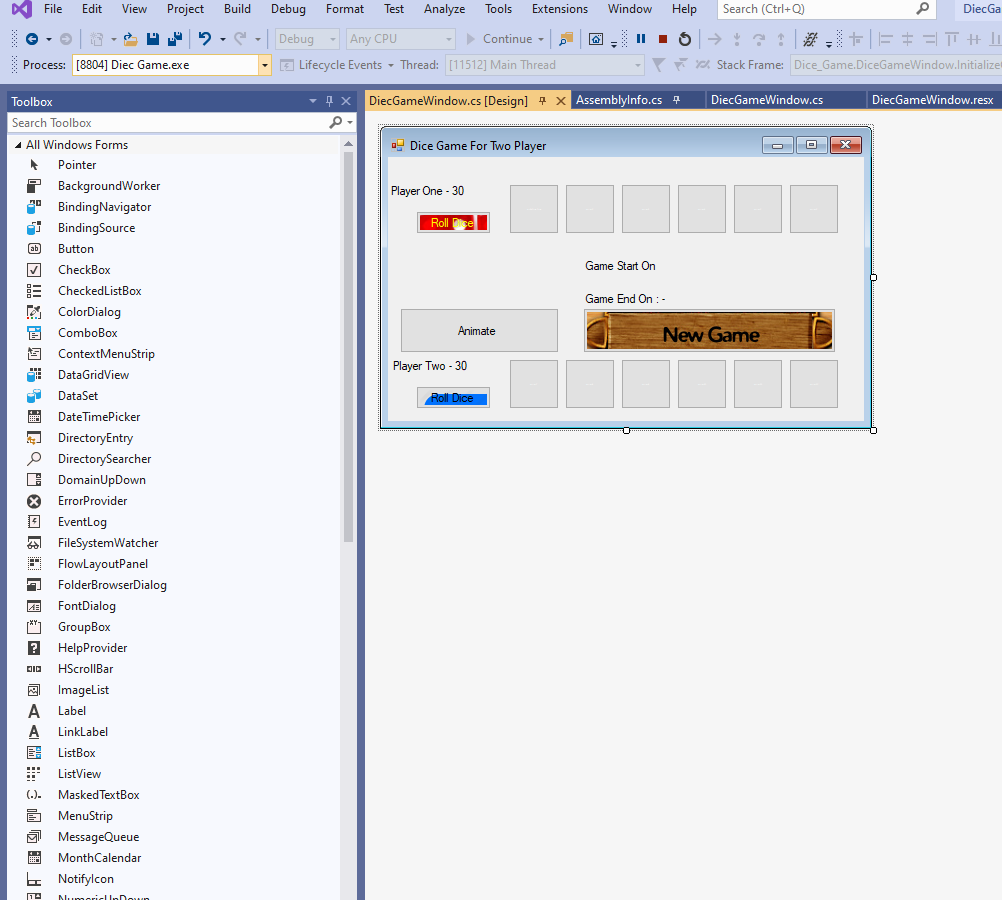


1. You should see the following configuration.



1. **Moving Between Windows**

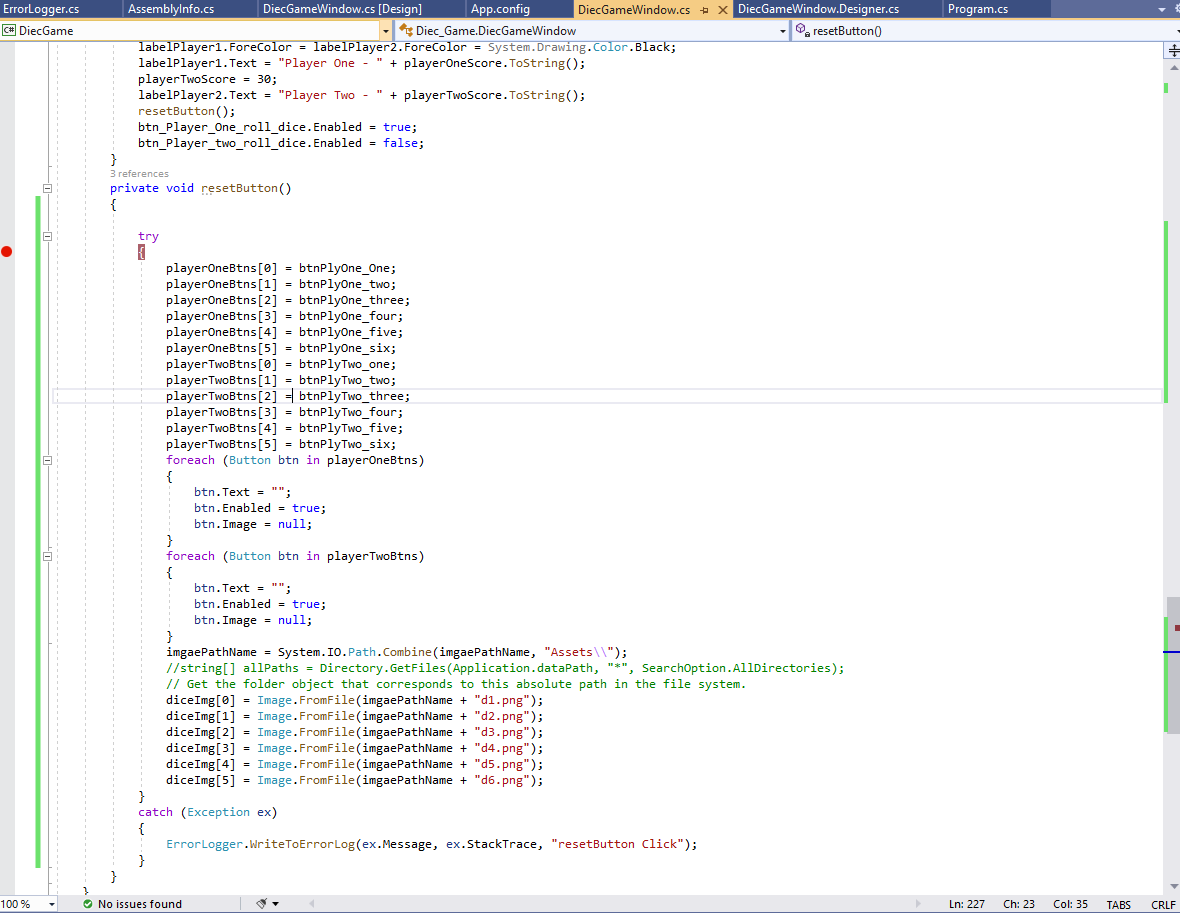
* To move between the **Design/CS** Window and **Code** Window in the IDE (Integrated Development Environment) is simple.
* We currently have the Design Window visible



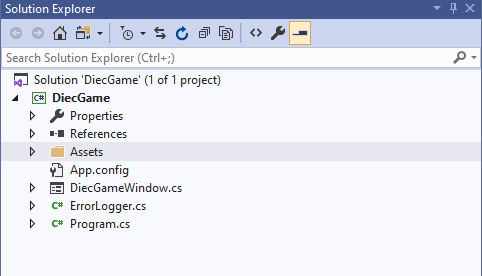
1. To move between the **Design** and **Code** windows simply click on **View Code**  icon in the Solution Explorer



1. The Code Window will be displayed…



1. To go back to the Design/cs Window simply double click on the DiceGameWinodw.cs file in the solution explorer.

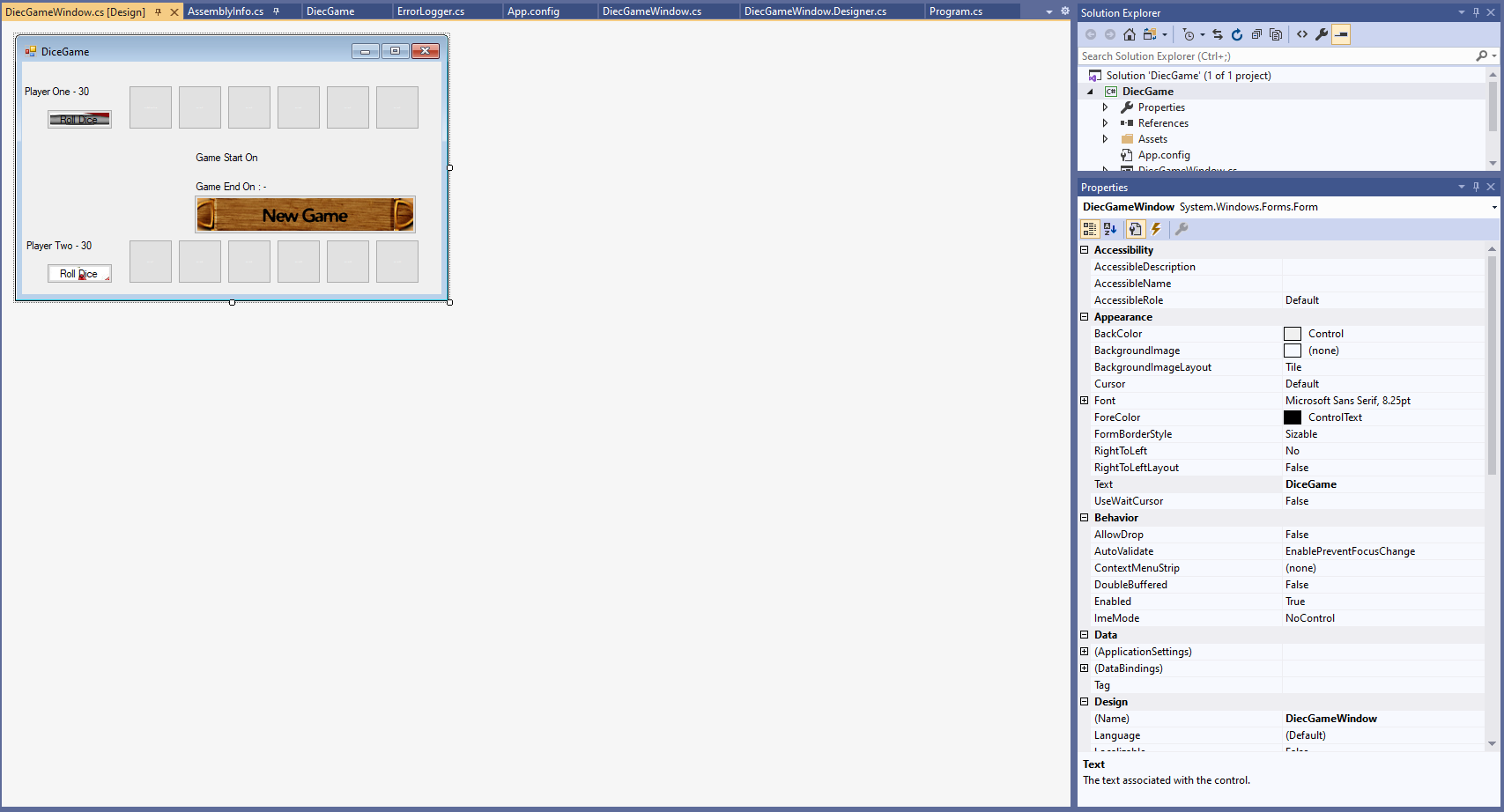


1. The Design Window will be displayed…….



1. Once the windows have been displayed in the Project Workspace, they will be a selection of window tabs that you can click on to navigate between windows.

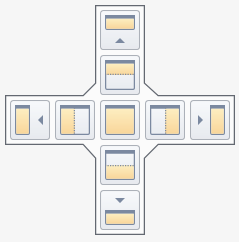
* Design/cs window with navigation tab
* Code window with navigation tab.



1. **Working with Visual Studio Windows**

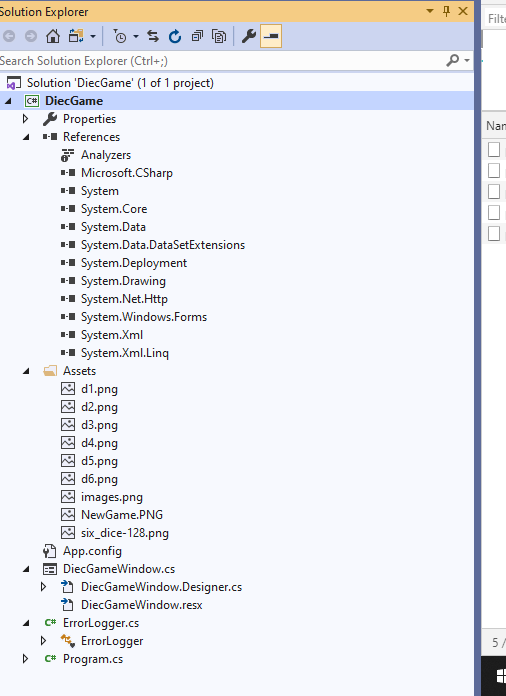


1. Windows can be moved around and docked in the IDE by dragging them by the top of the window panel and placing them on the required setting in the docking icon that appears.

* ****The docking icons have several options allowing you to configure the environment as you like
* ****Try this out ….

1. **Using the Solution Explorer**

* The Solution Explorer is used to manage a projects files and resources.
* When we select a Windows Platform Application for our project type the IDE builds and configures the project for us.
* There are seven components that make up a project and we will look at them as we progress through this course.
* Our primarily concern will be the Page Designer / cs editor (DiceGameWindow.xaml) and the associated C# source file (DiceGameWindow.cs) as these are the main sections of the project template, we will be working in.
* The Assets folder is used to store any resources used in the project image files etc. and any dependency libraries are listed under the References folder.



1. Files are displayed under the Project in a tree structure with a drop-down feature used to expand  or collapse  file groups.
2. File groups can be opened and closed by clicking on the triangles next to them. Individual files can be loaded into the IDE and viewed by double clicking on them.
3. Several files are required for a project build, the main files of interest to us are the Page Designer / cs (dicegameWindow), and associated C# source file (dicegameWindow.cs).

Some technical information

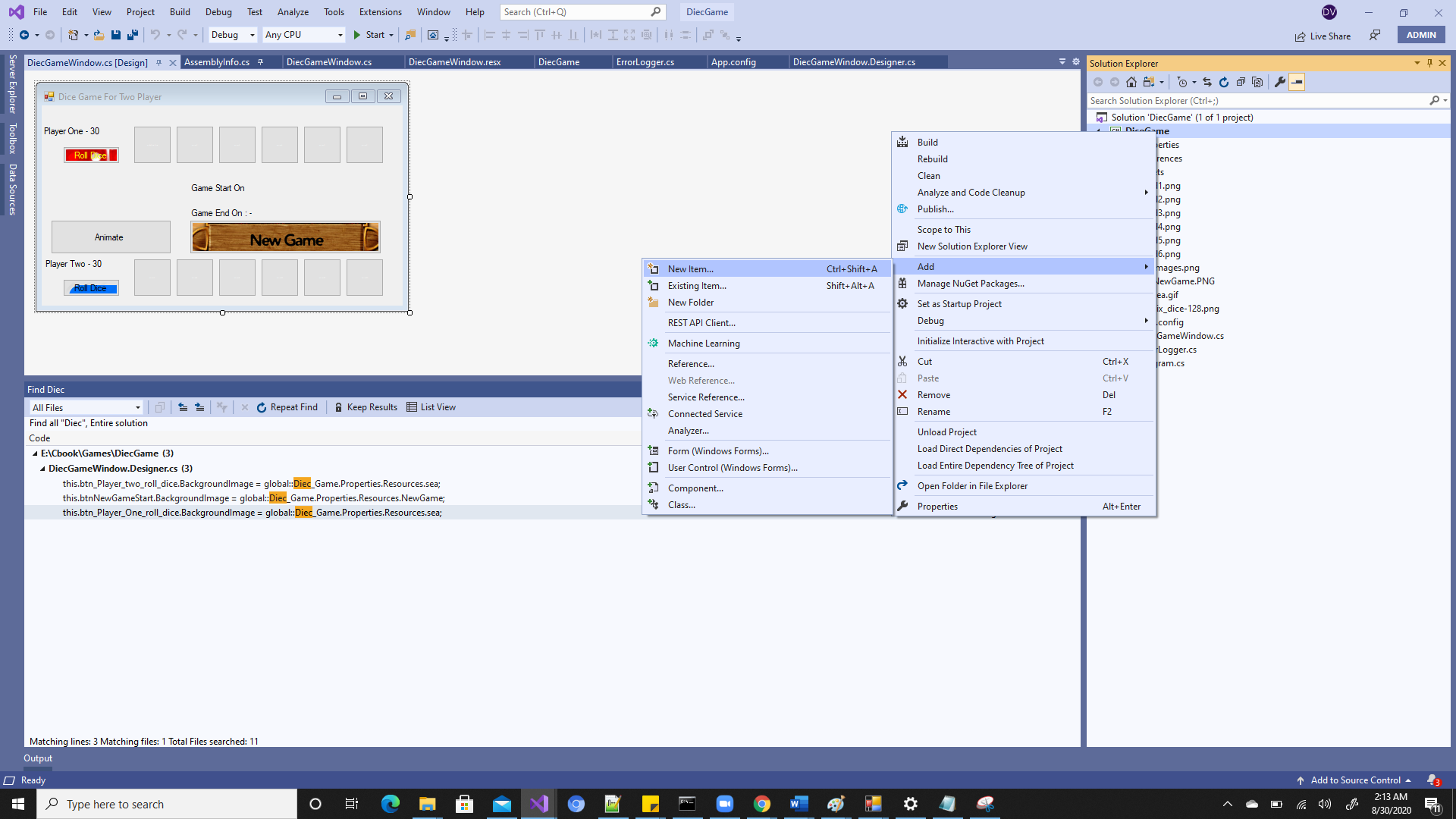
***Reference link when I was working as starting* - Microsoft** –

[<https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/inside-a-program/hello-world-your-first-program?tabs=windows>].

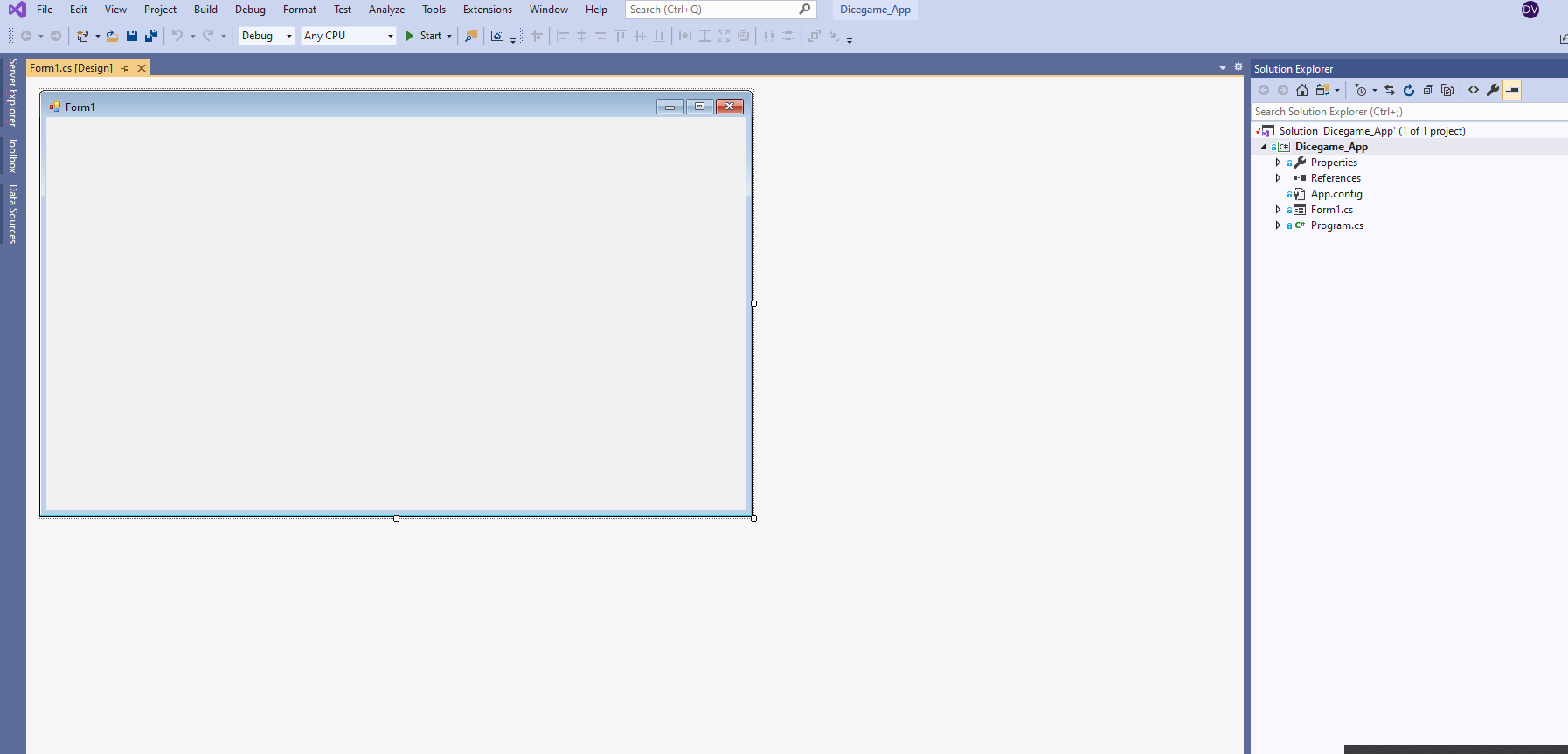
**And Search Window App in C# then You will see more content.**

1. Now We will create Window Form App.

* **Adding Pages to a Project**
* New Pages can be added to a project by **right clicking** on the project name in the Solution Explorer and then selecting the **Add > New Items** option.

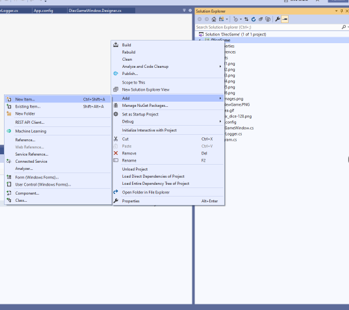


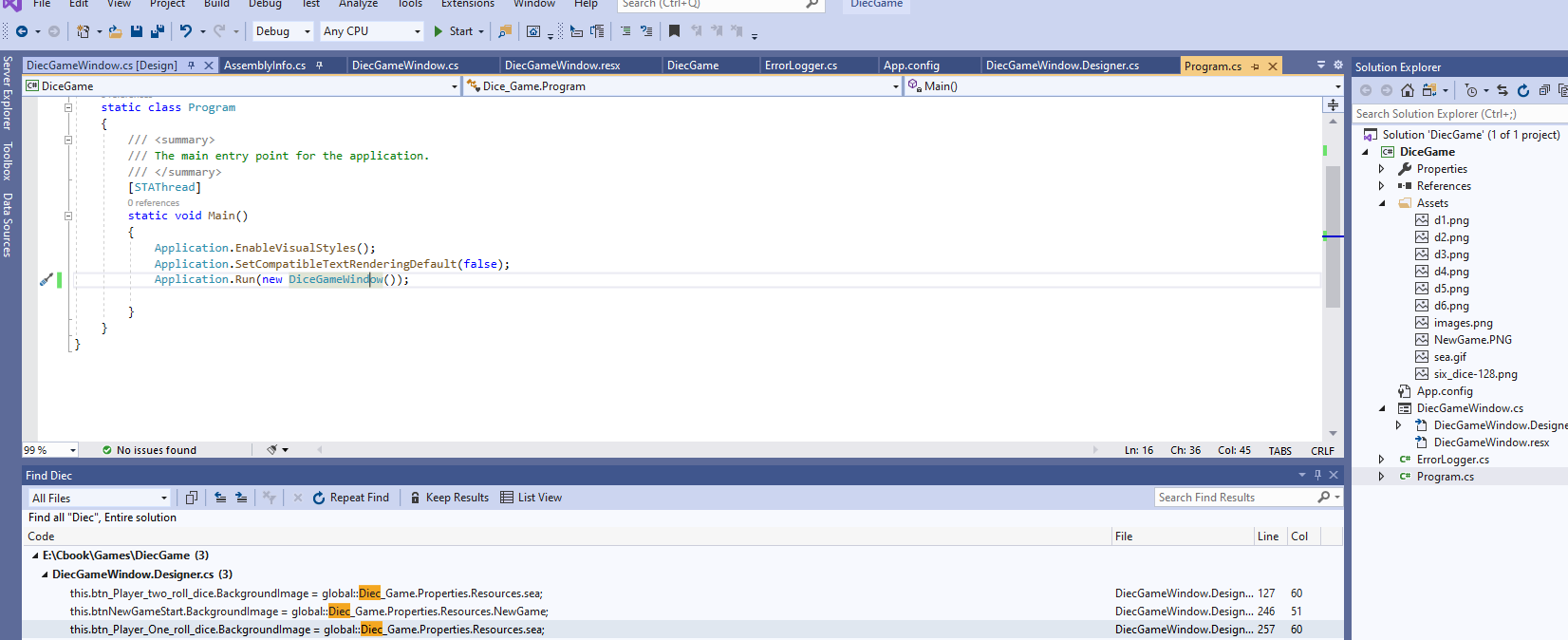
1. Select the **New From** item from the list, in the **Name:** box call the page **form2** and **click** the **Add** button



1. The new form will then be added to the project and displayed. Look in the Solution Explorer, it is now listed there. We can call a Page anything, but we tend to call them something that relates to their functionalityso we can identify them easily in the Solution Explorer window.
2. **Adding a Class to a Project**

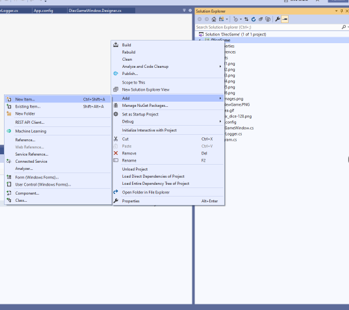
* We can add a Class file to our project. This type of file is used for writing C# code in the format of a Class.
* Please Note: We will look at classes and the coding of classes in some detail in a couple of weeks. In this section we are concerned with being able to add a class type file to our project using the Solution Explorer.
* New classes can be added to a project by **right clicking** on the project name in the Solution Explorer and then selecting the **Add > New Items** option.



1. Select the **Class** Visual C# item, in the **Name:** box call the class **MyClasss** and then clickthe **Add** button give name as you want.
2. **Adding a Resource to a Project**

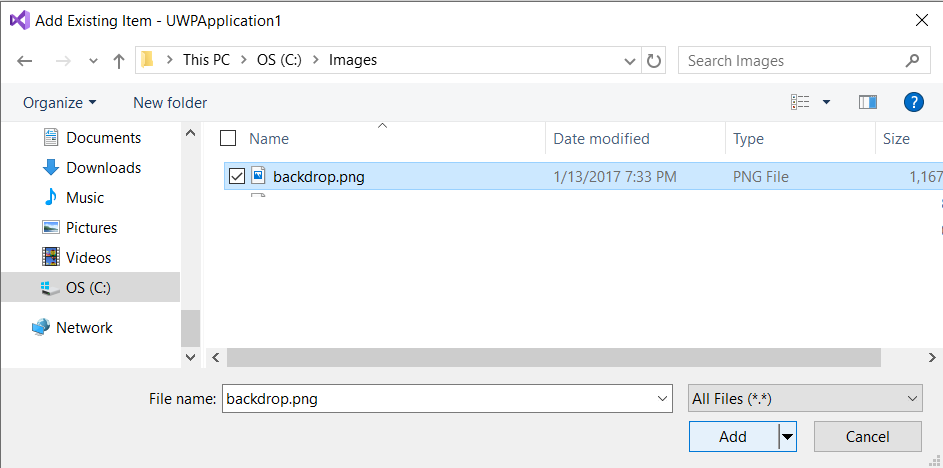
We can add a resource file to our project, these tend to be images and graphic files.

**Please Note: We will look at resources later in the course in some detail. In this section we are concerned with being able to add a resource to our project using the Solution Explorer.**



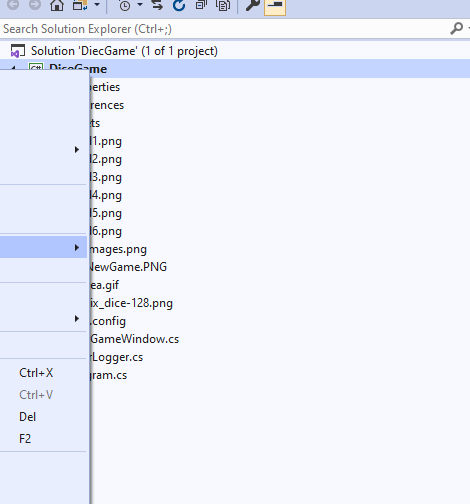
Resources can be added to a project by **right clicking** on the project name in the Solution Explorer and then selecting the **Add > Existing Item…** option.

This will activate the explorer **Add Existing Item** window.

****

Navigate to your image folder and the image you want to load, then click the **Add** button.

**Note**: Resources required for any of the course work in this course will supplied with the examples in a Resources folder. In this case navigate to the Resources folder and select the backdrop.png file.

****

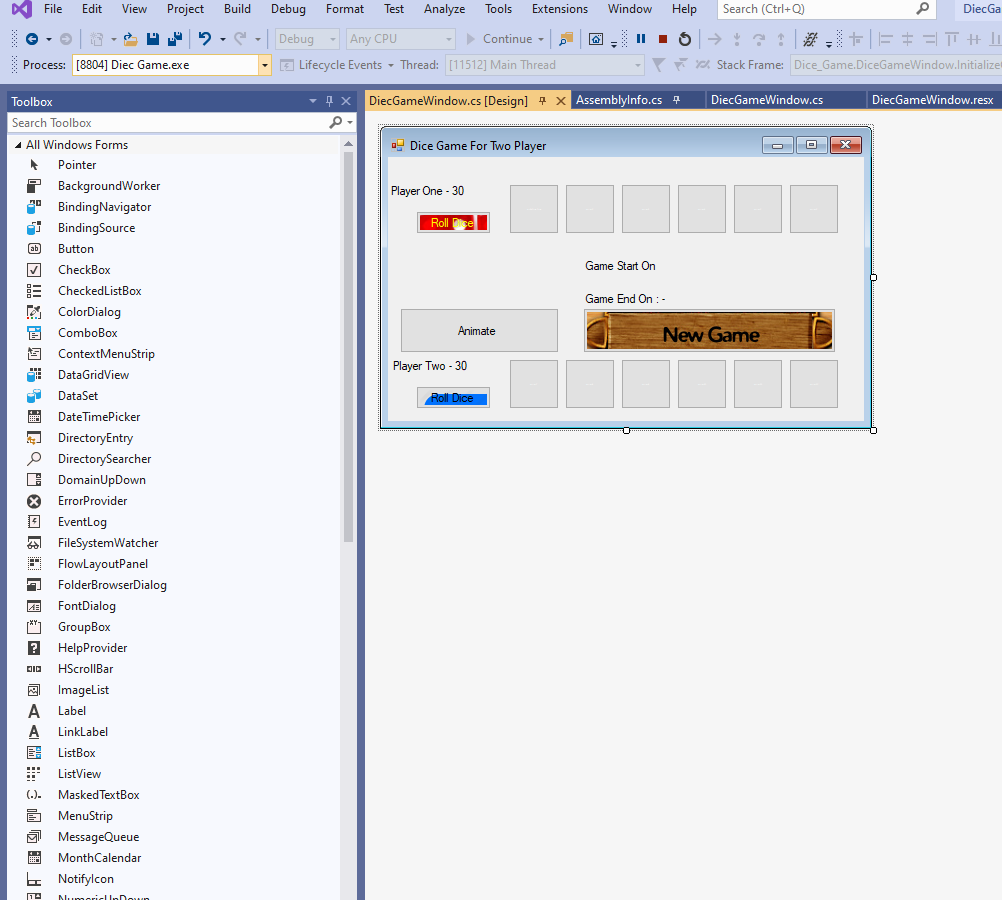
The new resource in this example **backdrop.png** will now be added to the project.

Look in the Solution Explorer under Assets for the new backbrop.png file.

1. **Using the Designer window form**

We can design Pages with the aid of the Design Window, Toolbar, Properties Window and CS editor. Let’s go back to the Design Window by double clicking on the **DiceGameWindow.Designer.cs** file in the solution explorer or selecting the **DiceGameWindow.Designer.cs tab**.

Now we will make our page the size of a standard screen by selecting the **13.3 Desktop** option from the dropdown list (top left of the designer window).



1. **The Designer Window**

In this example we will use the Designer Window and Toolbox to design the interface.

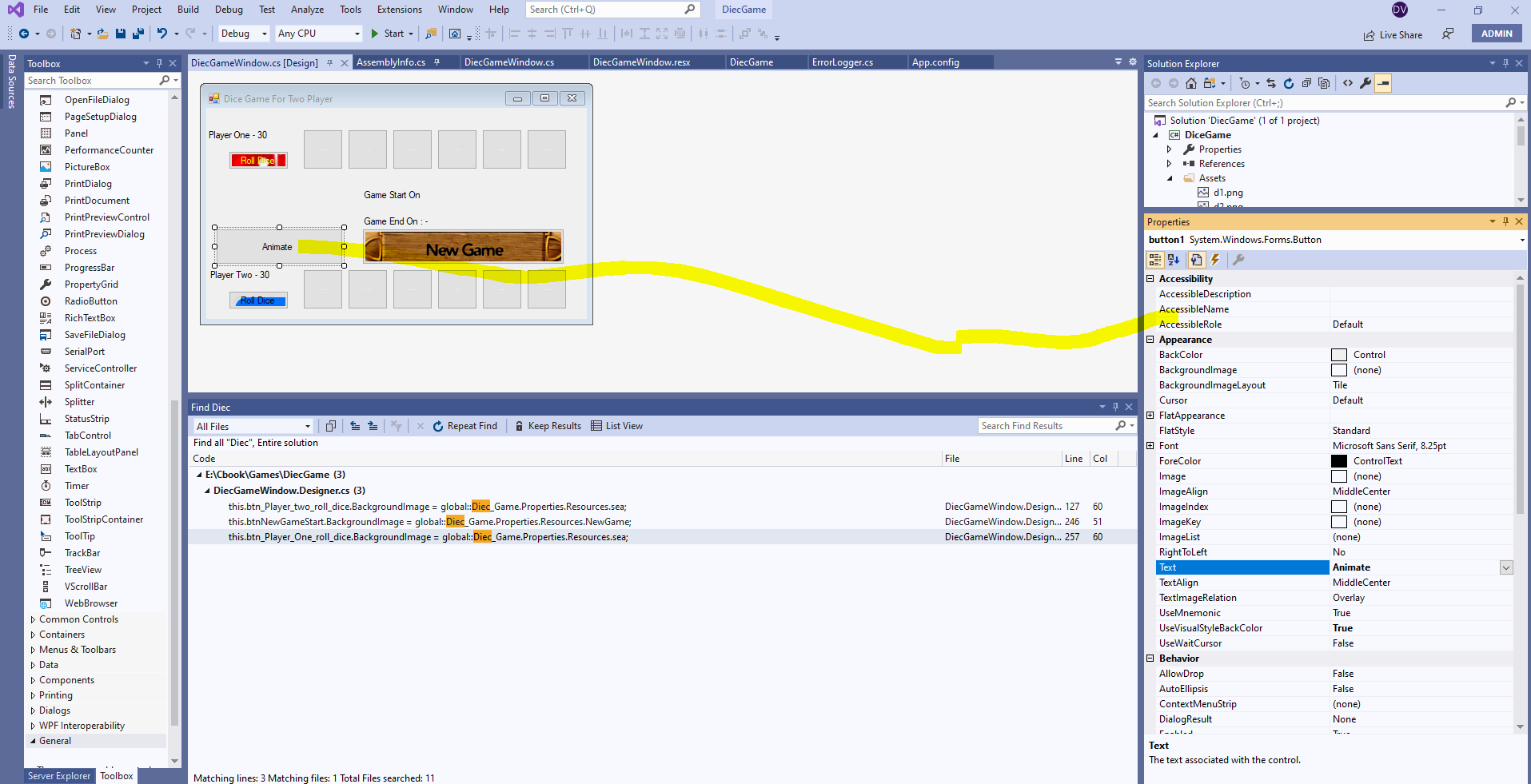
Objects can be dragged from the Toolbar to the Page, there they can be moved and edited.



As We have done with UI as above Screen if you want more component you can add from toobox.

1. **The Properties Window**

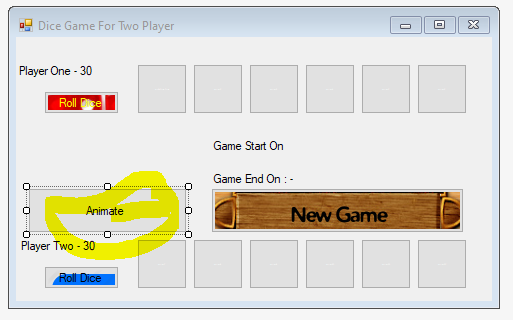
The Properties window can then be used to change the configuration of the Page and objects added to the Page from the Toolbox.Clicking on a page or an object on the page will activate the Properties window associated with the selected object.

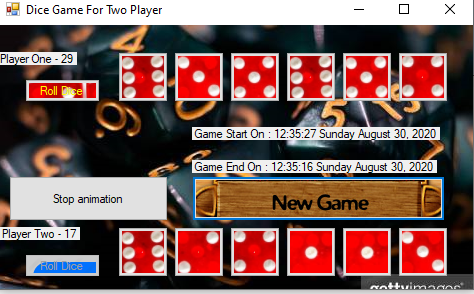


1. Click on the page and the associated Properties window will be displayed.
2. The properties window can then be used to change the configuration of the page
3. Click on a button and the associated Properties window will be displayed.
4. **If the Properties window is not visible. Do you remember how we set up the environment windows (select the View tab and click on the Properties Window option).**
5. **Just for Learning– Let’s add a background  to the form Window**

When we load the window then it will auto magically loading..

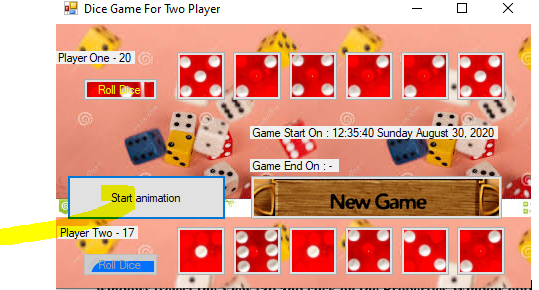
I have added a button to stop and start animation…







**Start Button**



**Stop Button**



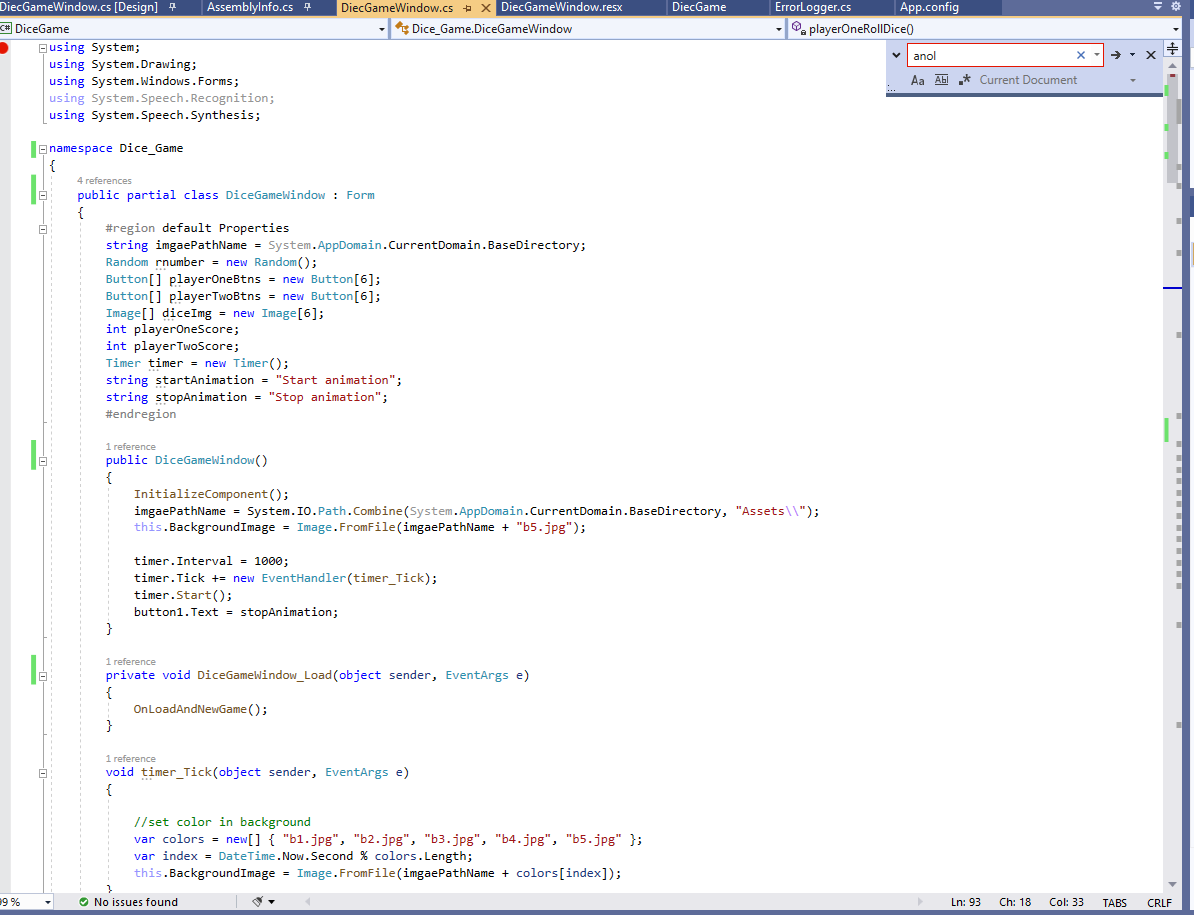
1. **Exercise 1: Just for Exp.**

Add a new file to the Assets folder. The file is in the Resources folder all dice’s Images

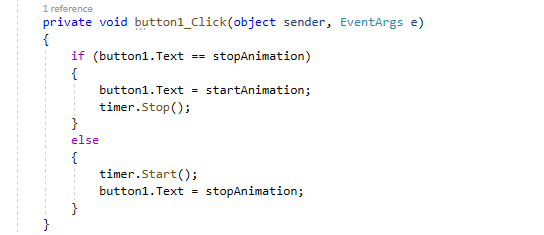
* Added Six button for dice
* Added one New Game Button
* Added two Roll Dice Button for both players.
* Added Start on and End on Date and Time on label.
* Added two labels to show score and winner in green color
* Added functionality Text to Speech for allowance winner.



1. This page contains the C# code for our page

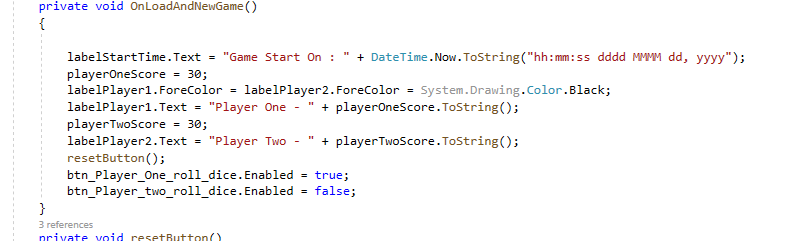


1. The ‘**using’** keyword allows you to link classes into your code (these classes contain functionality supplied by the environment).
2. Note how the **{ }** are used to define blocks of code.
3. In this piece of code we have a ‘**namespace’** (project) called **Dice\_Game** which has a Page in it called ‘DiceGameWindow**’** with some start-up code (constructor) for the page.
4. And **Initialize Component**
5. The event generated is called **Button\_Click**, this is where we can put any code that we want to run (something to happen) when the button is clicked.

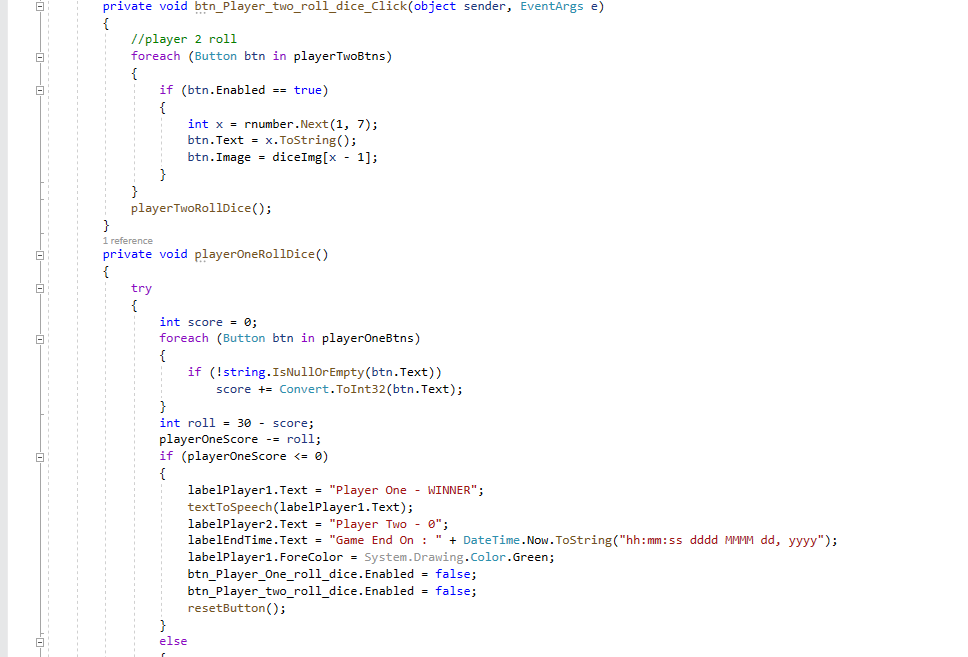


The click event fires when the button is clicked, any code we put inside the event will also be run.

1. **On New Game Button Click**



1. **On Roll Dice Click**

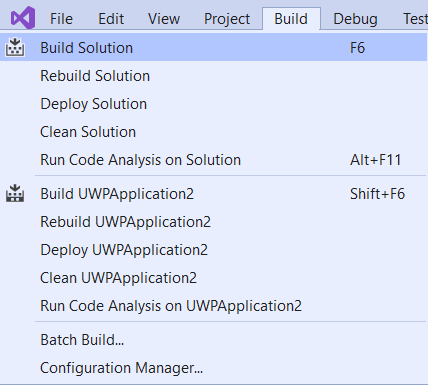


1. **Project Execution**

Now we have some code in our project let’s try and run it.

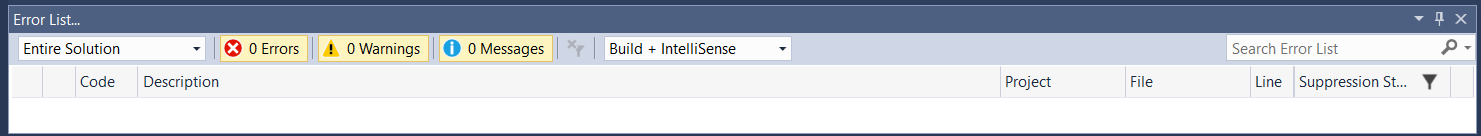
The process is to:

* Build the code, this allows the compiler to check for any errors.
* Run the code, the form will run on our desktop.
* Stop the application from running, so that we can continue coding.

**Building the Application**

To build an application select the **Build > Build** **Solution** option. If you need to completely rebuild (overwrite the build in memory) then use the Rebuild Solution option.

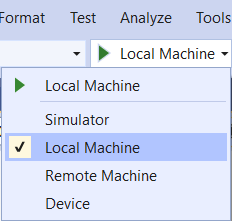
This will build the solution and display any errors (in the error list) if the code is wrong.



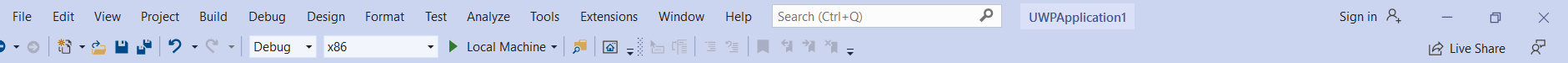
1. **Build not done Successful**

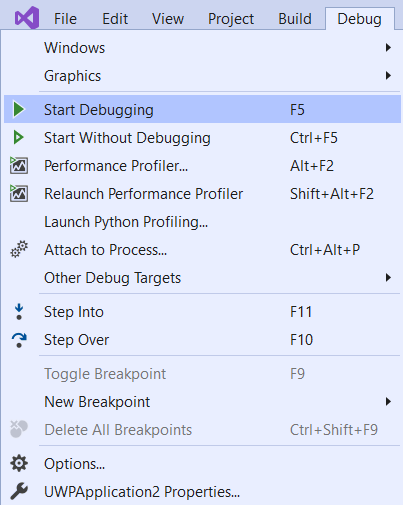
**Note: If the Error List is not visible then select the View > Error List option to display the window.**

**Running the Application**

There are several options that allow us to run the application.

First check the platform is set to local machine, by selecting the **Local Machine** option on the dropdown list.

Click the **Start**  arrow on the toolbar.



Or selecting the **Debug > Start** **Debugging** option.

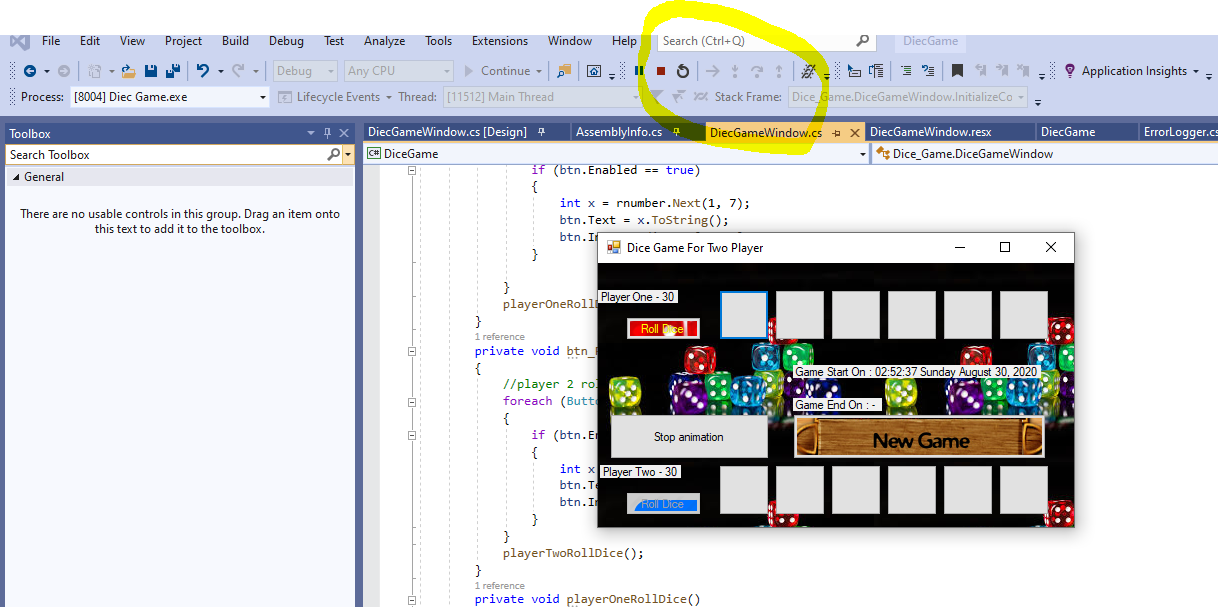
Or press **F5 key** on your keyboard.

1. **Stopping the Application Running**

To stop the application running any of the following methods can be used.

Close the application itself (by whatever mechanism you have designed into the application), for this application, click the close button (**X**) on Form1.

Click the Stop Debugging square  .



Select **Debug > Stop Debugging** option

Press **Shift + F5** keys on your keyboard.

1. **Exercise: Introduction to Visual Studio 2019**

Work through the each of the following sections of the course material ***again*** and produce a Window Windows Platform Project, call the project **DiceGame**.

* Opening the Development Environment
* The Development Environment
  + Creating a Window form Project
  + Saving a Project
  + Closing a Project
  + Open an Existing Project
  + What’s in a Project Workspace?
    - Moving Between Windows
    - Working with Windows Studio Windows
    - Using the Solution Explorer
      * Adding a window form to a Project
      * Adding a Class to a Project
      * Add a Resource to a Project
    - Using the Page Designer
      * The Design Window
      * The Properties Window
      * Adding an Image to a Page
    - Using the Code Editor
      * Generating Code
* Project Execution
  + Building the Application
  + Running the Application
  + Stopping the Application Running
* Extend the notes
* Using your current code implement of a way to get back from extra pages to the DiceGameWindow.
* Research - We can also use something called a image/text to speech.

Example Project: DiceGame

1. **IDE Coding Features**

In this section we will look at several useful features of the Integrated Development Environment.

Note: Use the **IDECodingFeatures** project to examine the IDE coding features.

**Comments**

Comments are added to code to explain what is happing and make your code readable. Adding a comment makes the compiler ignore the text associated with it. Comments appear as green text in the code editor.

To comment a single line use the **//.**

|  |
| --- |
| // A single line comment |

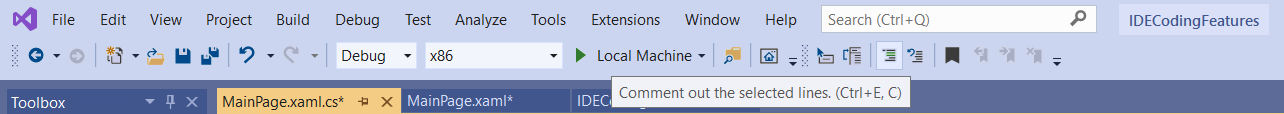
To comment a block use the /\* \*/ it has two formats.

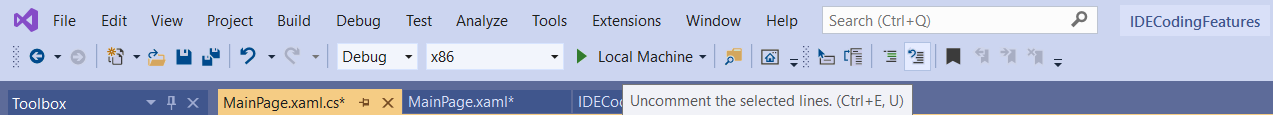
|  |
| --- |
| /\*  \* A block of comments  \* A block of comments  \*/  /\*  A block of comments  A block of comments  \*/ |

We also have a special format that auto generates the header for a method, typing **///** above the method auto generates the header block for you. We then fill in the relevant information ourselves.

|  |
| --- |
| /// <summary>  /// </summary>  /// <param name="inputA"></param>  /// <param name="inputB"></param>  /// <returns></returns>  /// <summary>  /// A test method  /// </summary>  /// <param name="inputA">Integer used in calculation</param>  /// <param name="inputB">Integer used in calculation</param>  /// <returns>Result of the calculation</returns>  public int test(int inputA, int inputB)  {  return inputA + inputB;  } |

Comments can be added or removed using the comments options in the tool bar.



****

1. **Exercise: IDE Coding Features**

Use the IDE Coding Features section as a reference to work on this exercise.

Use and extend the **FeaturesExercise** project for this exercise, we will be working in the **dicegamewindow.cs** file.

Work through the each of the following tasks updating the **FeaturesExercise** project **dicegamewindow.cs** file where required:

* ***Line Numbers*** – Switch the line numbers on and off on the **dicegamewindow**.cs code. Now decide which you prefer and set that option.
* ***Indentation*** - Indent all the code in the **dicegamewindow**.cs file.
* ***Bookmarks*** –Set a Bookmark on every line with an open curly bracket **{** and closed curly bracket **}**.
* ***Comments and Help*** – Wherever you see comments in the code in this format **// What does this mean (int)** access the Help files (highlight and F1 key) and update the comment with what it means.

e.g. explain what int means.

// What does this mean (int)

int a;

* ***#region / #endregion Directives*** – Setup the following regions in the **dicegamewindow**.cs file in the Refractor code method:
* First Change
* Second Change
* Third Change
* ***Refactor Code*** – Refactor all the following variables:
* Change ABC to aNumber
* Change XYZ to aDecimalNumber
* Change addsomething to addSomeNumbers
* **IntelliSense** – Experiment with the IntelliSenc feature.

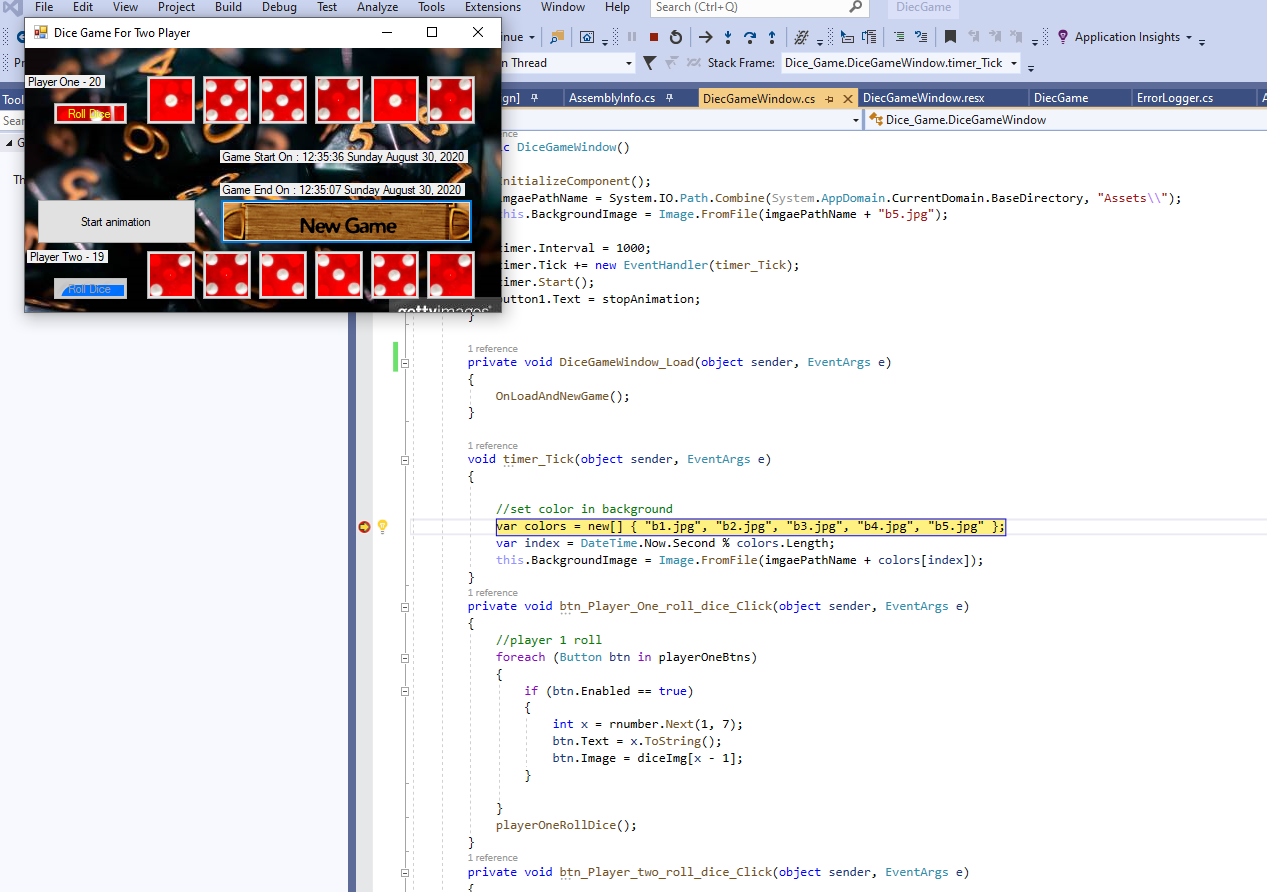
1. **How to Debug an Application**

There are times when our code is not working correctly or is just full of errors.

That is when we need to debug our code.

The IDE has several features that report on the status of code and allow us to observe its operation.

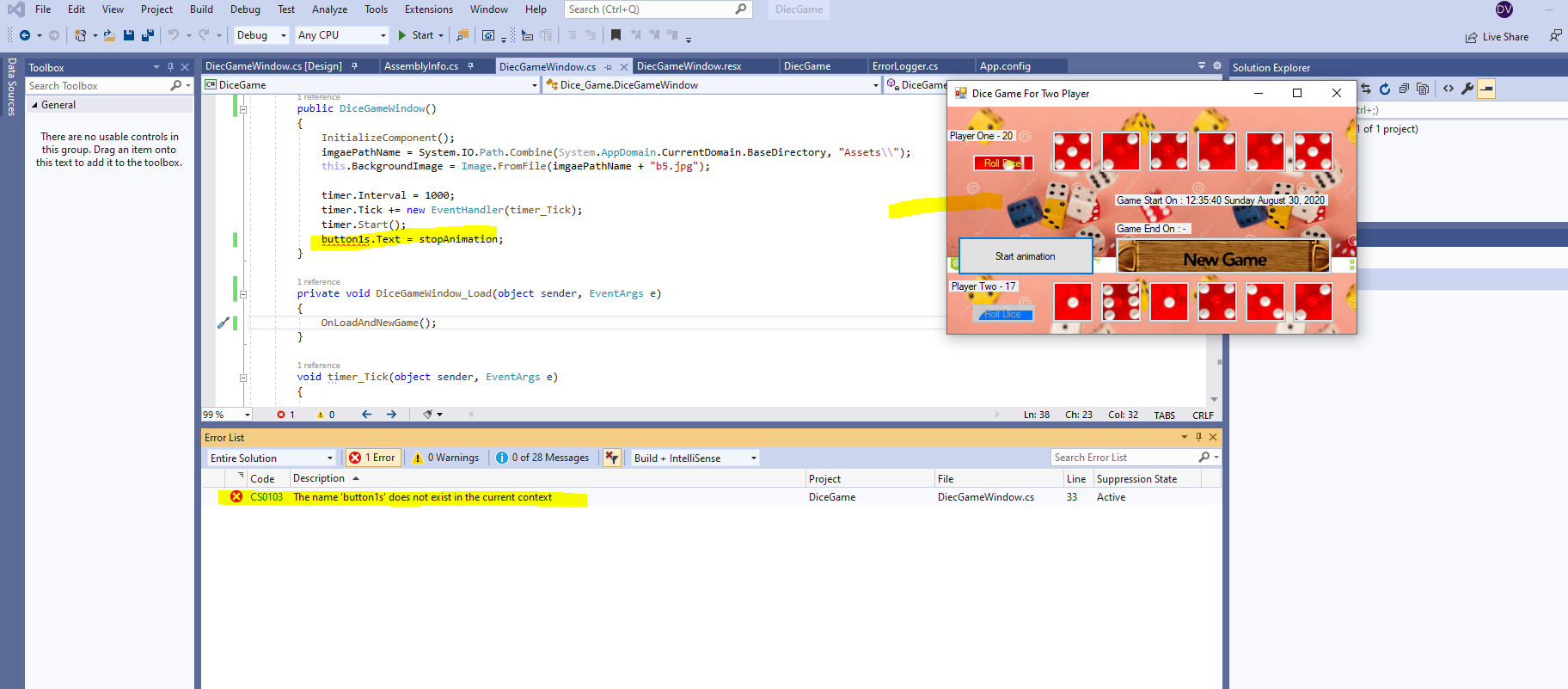
Note: Use the **HowToDebug** project to examine the debugging coding features.



1. **Error List Window**

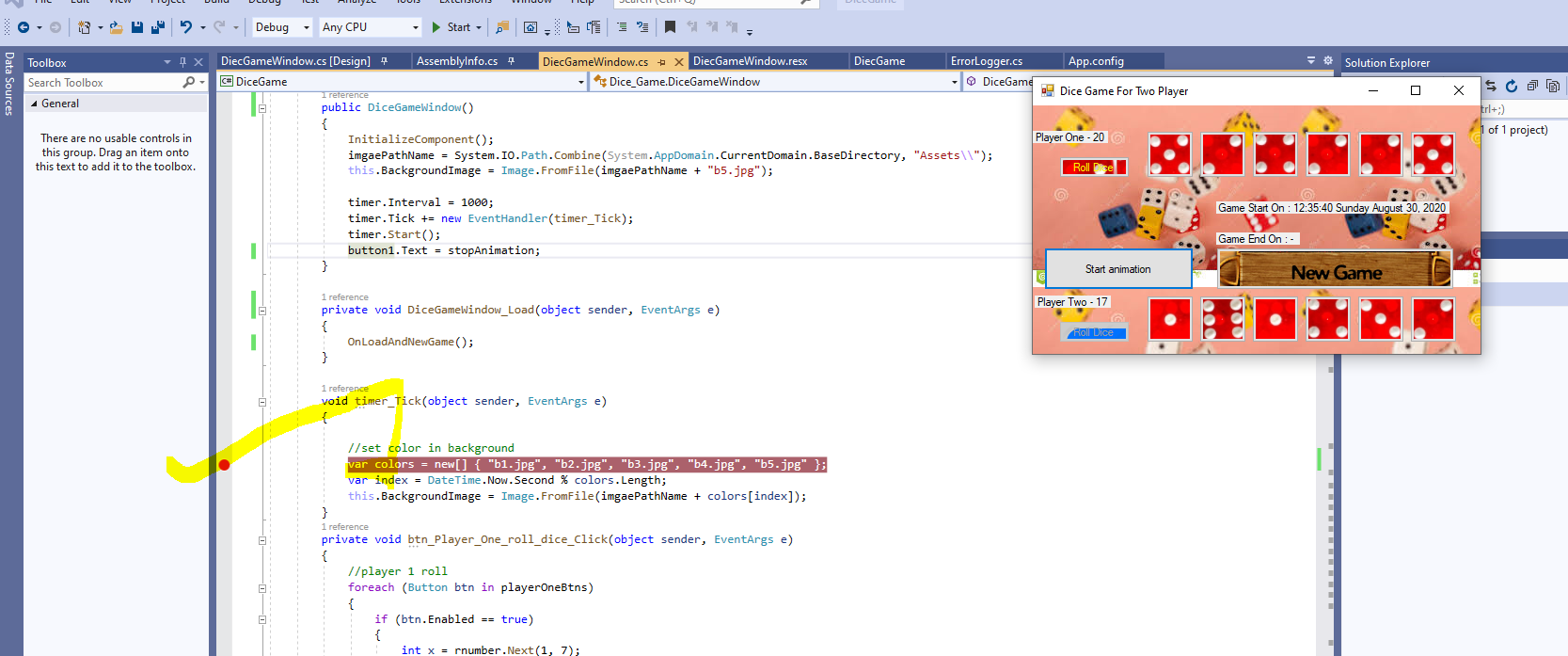
The first indication we have an error is when we code the application or build the application.

Any errors (syntactical) will appear with a red line underneath them and an error message will appear in the Error List that explains the error/s.

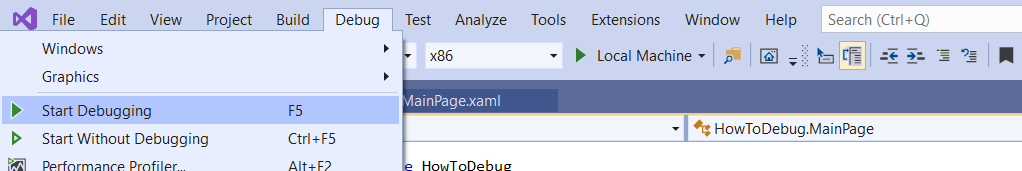


1. **Set a Breakpoint**

A breakpoint can be set by ***clicking in the left margin*** on the line of code you need the break point.



Note: You can only set a break point on a line with code on.

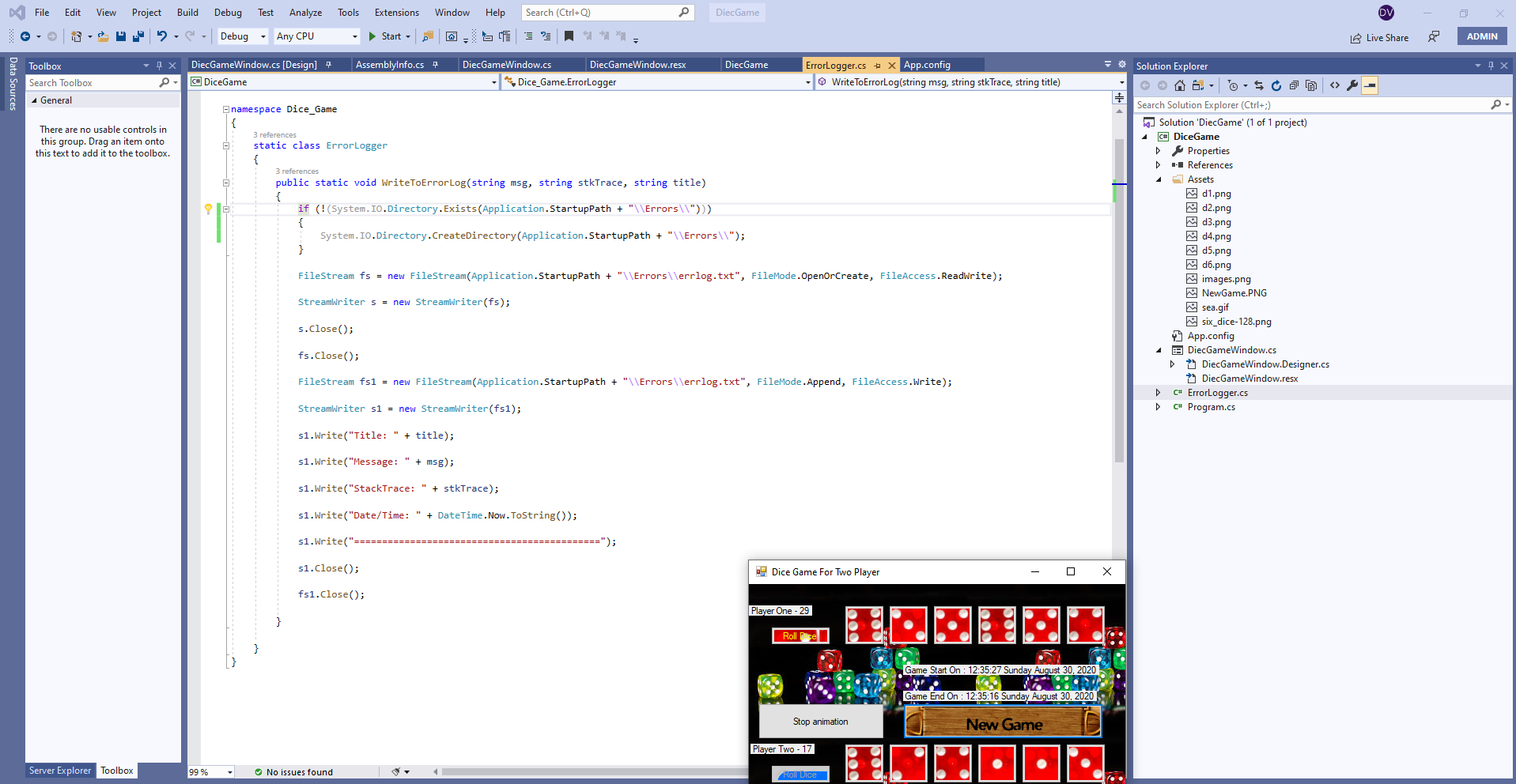
1. **Start the Debugger**

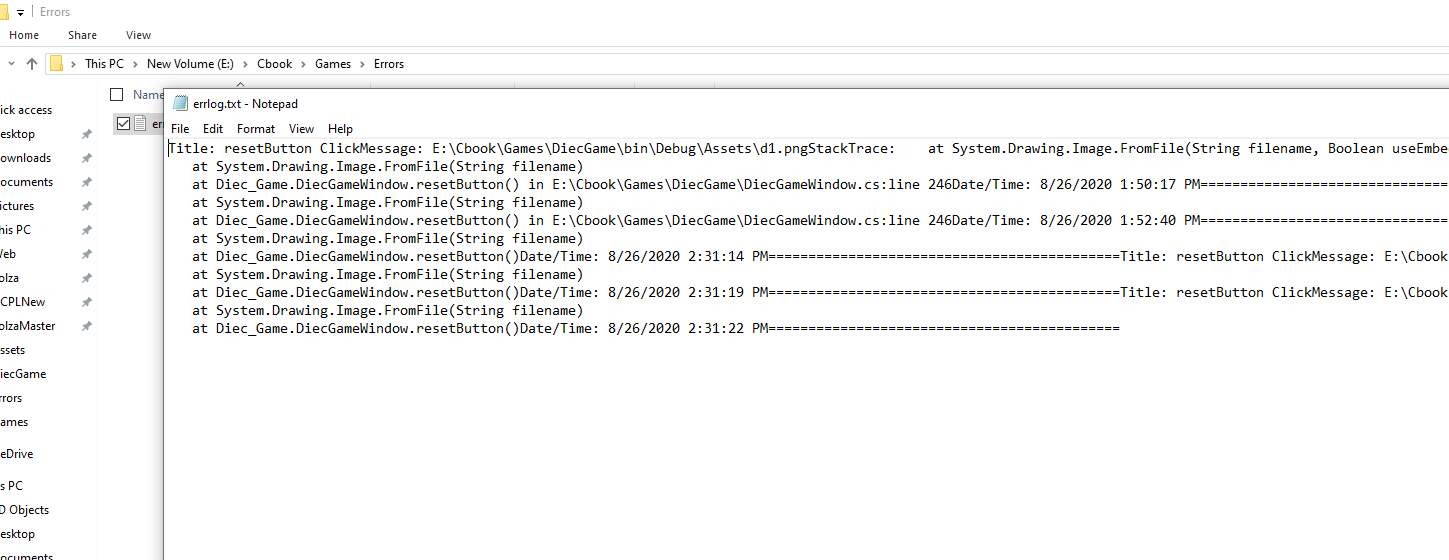
The debugger can be started by selecting the **Debug > Start Debugging** option.

When the flow through the code reaches the breakpoint (in this example when the button is pressed on your application that is being run) the debugger stops the application.

1. **Added Error handing in Logs File**

**We have Added a call called – ErrorLogger This will add all logs in file it will generate at run time in error folder.**





**Thanks,**