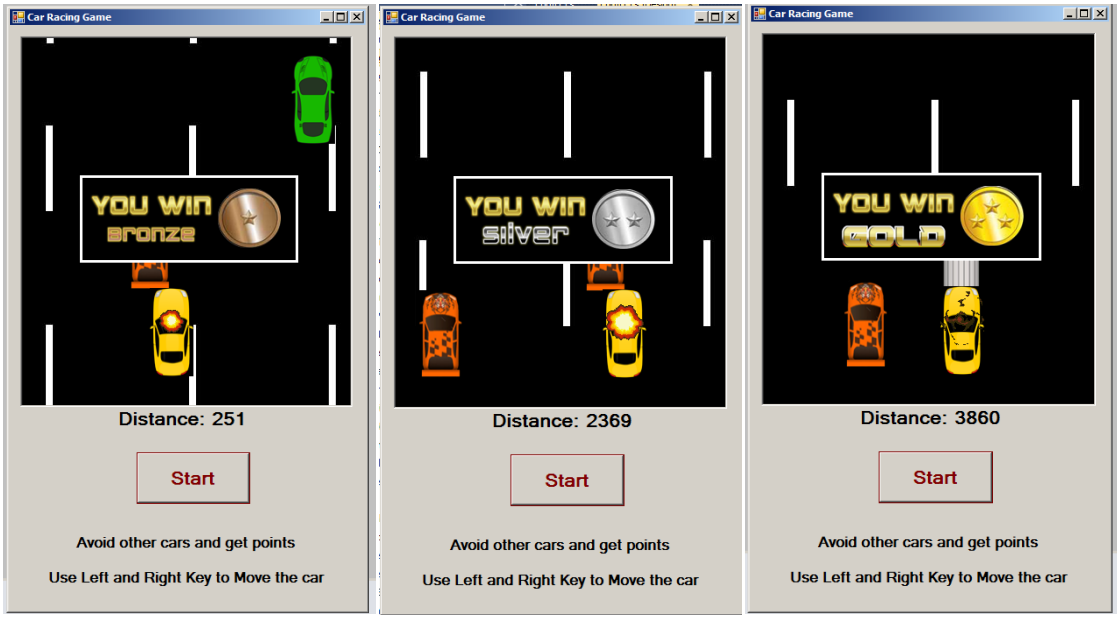
C# Top Down Car Racing Game Tutorial

In this tutorial we are going to create a full top down car racing game. The game principles are as the score goes up so does speed of the game, the is over once you hit another car. In this tutorial we are using Visual Studio and C# (sharp) programming language. We have pre designed the game assets and you can use your own but you need make sure they all have the same height and width.

In this tutorial you will learn -

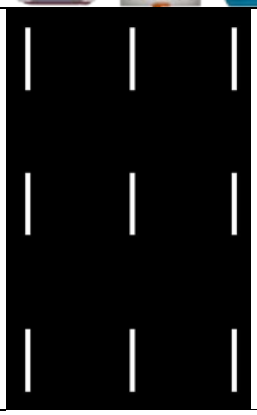
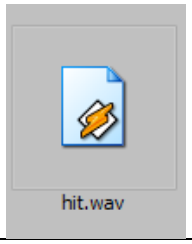
1. How to create a arcade game in C#
2. How to identify the collision between multiple objects / picture boxes
3. How to integrate key down and key up events into the windows form
4. How to create a parallax scrolling background on windows form
5. How to create a game level using visual studio component Panel
6. How to dynamically change the images in the picture boxes
7. How create and run functions
8. How to use Booleans
9. How to create a scoring system and reward the players appropriately
10. How play WAV music file using only C#(Sharp) codes

Remember this project was created and tested over a period of few days, I encountered multiple errors. All the code are checked, tested and made sure its workable. So I hope you achieve the similar result, if not please refer back to the tutorial.



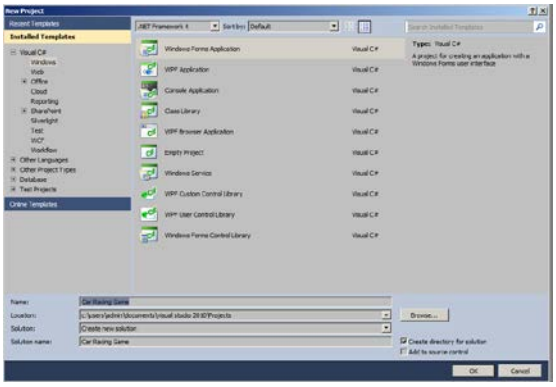
This is the final game. We you progress you will gain points and judging from the points you can either win Bronze, Silver or Gold trophy.

```
// this green text in the codes are the comment, they don't affect the programming in any way and they are designed to be ignored by the compilers. We use them to better understand and leave notes in the code to help the developer. So we have explain each line of code in the green text.
```

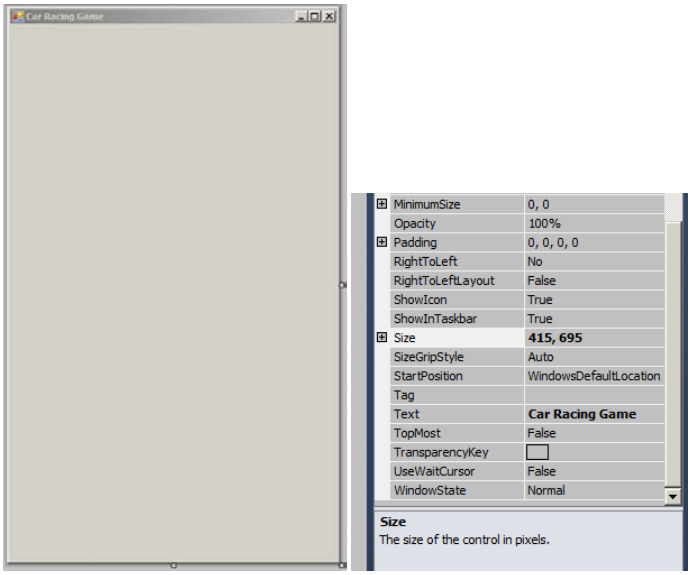
Assets	Description
	AI Cars These are cards which will be seen on the screen. They are all PNG file format with full transparency.
	Road Marking This image is a JPEG image which shows the road marking and it will animated from top to sliding to bottom to give the illusion of the car moving forward.
	Trophies. These are trophies you can win in the game. The main objective of the game is to drive as far as possible without crashing into another car. This image will be shown when the game ends depending of what the player has achieved.
	This is the driver. We chose this simple yellow car to resemble our humble driver.
	Explosion This is the animated explosion GIF which we will use to show the end of GAME.
Car Crash AUDIO WAV File 	We have a sound of crash crashing which will occur when the car collides with another. Why WAV and not MP3 file. Its easier to play a WAV file directly using WAV file. To play Mp3 file you will need some additional DLL file to be added to the project.

Visual Studio Components	Description
Button	To restart the button
Timer	To animate, speed up traffic and act as the main game event.
Label 1	Distance (that's it this is a static text on the screen)
Label 2	Track the distance travelled in the game. It will start from 00.
Label 3	Instructions for the game (This is a static text on the screen to instruct the players on how to play the game. This will be a Multi Line Label)
Panel	This will be the game screen Height 424 Width 380 Background Colour Black.
Picture Box 1 - roadTrack1	This will act the as the road track image as seen above. This picture box will scroll down to give the illusion of the moving road.
Picture Box 2 - roadTrack2	This will be the second picture box with the exact height and width as picture box to give the illusion of parallax scrolling for the road track.
Picture Box 3 - player	This will be the player picture box. We will be controlling this object with the left and right arrow key. The object will only move horizontally.
Picture Box 4 - AI1	This is the AI car.
Picture Box 5 - AI2	This is the second AI car. *How come we have 8 AI car images and only 2 AI car picture boxes. Well it's not a good practice to use so many picture boxes we want to make this game light weight and fast. So when the AI car leaves the screen it will get relocated to top of the screen, while that is happening the cars swap the inside images with other cars to make it look like there are more cars on the road.*
Picture Box 6 - explosion	This is the explosion animation, which will invisible for most of the until the crash happens, then we can animate it on top of the player car picture box.
Picture Box 7 - Trophy	This picture box will show end of the where we will see what the player has achieved. We have set the scores for the game where better the players skills to avoid cars on the road the better they score. Bronze - 1000 or less score Silver- 2000 or more score Gold- 3500 or more score

Start a new project in visual studio name the project Choose Visual C# Windows Form and name it **Car Racing Game**



From the default settings of the form change the following settings first Change the text to **Car Racing Game**

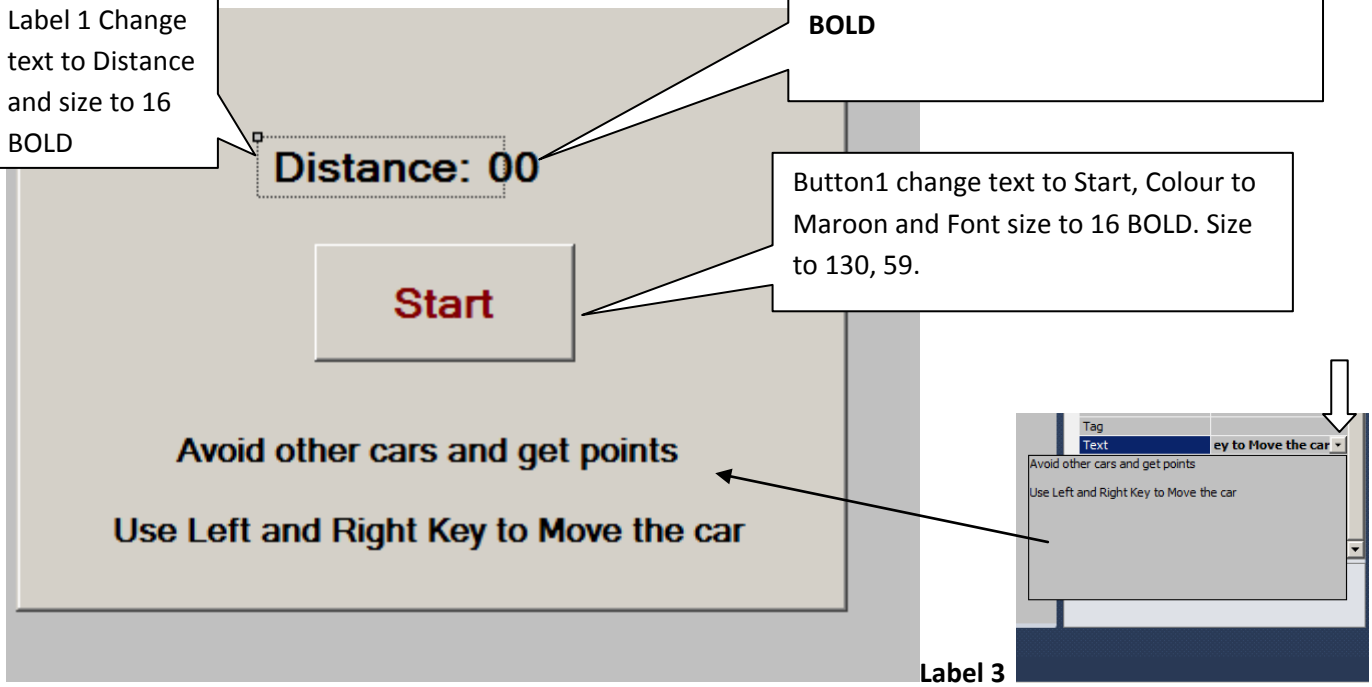


In Form properties change the size to width 415 and width 695

Label 1 Change text to Distance and size to 16 BOLD

Label 2 Change text to 00 and change the **name** to **distance. Change text size to 16 BOLD**

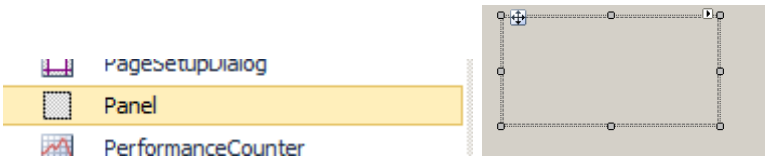
Button1 change text to Start, Colour to Maroon and Font size to 16 BOLD. Size to 130, 59.



Label 3

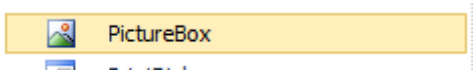
Click on the drop down button next to the text option and you can add multiline to the label.

Now we add the panel. Click on the Panel component in the tool box and drag and it to the form

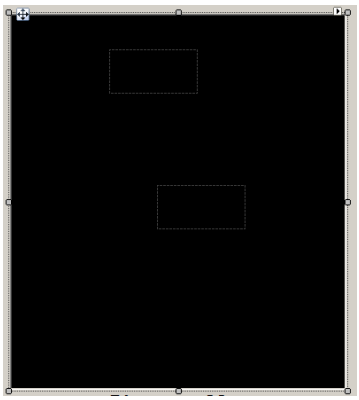


While the panel is selected look at the **properties** window and change the following settings

BackColor - Black BorderStyle - Fixed3D Location - 12, 12 Size - 380, 424	



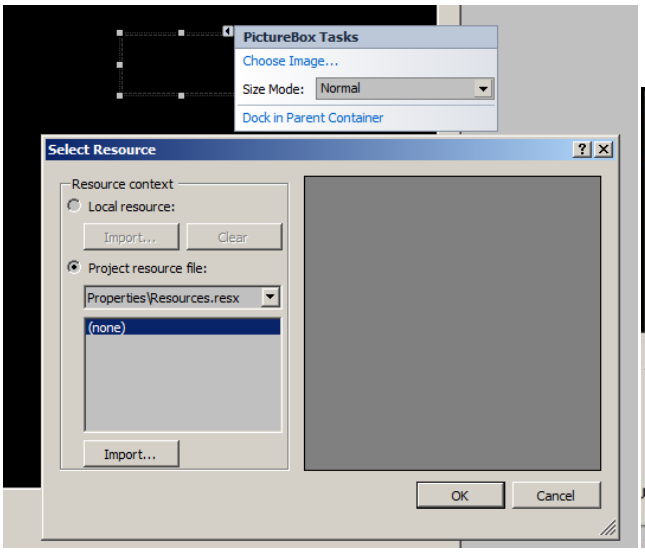
Now add two picture boxes to the form.

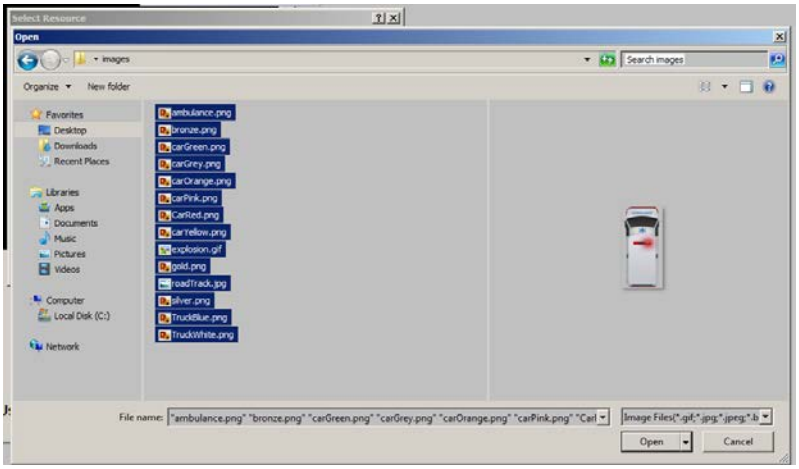


We need to add all of our resources to the project.

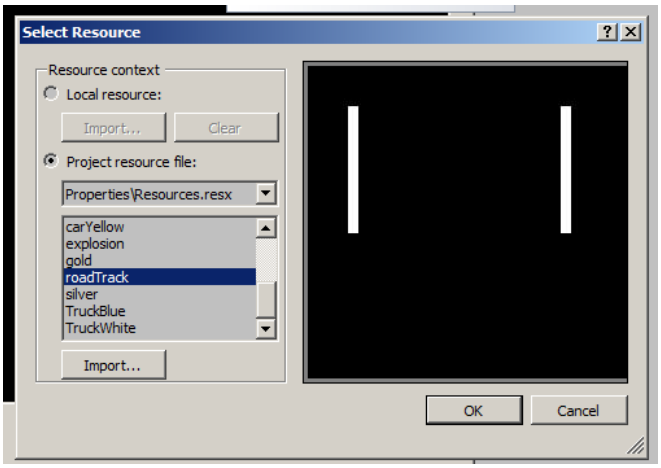
Click on the picture box 1 and Click on the little triangle on corner right of the picture box.

Click on Choose image select and click on Import



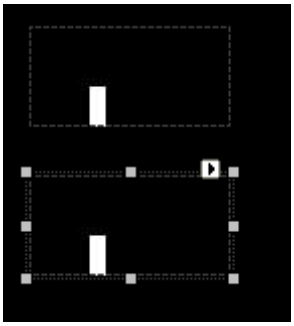


From the list of images select the roadTrack image and click OK



Once you have imported all of the images they will be copied to the resources folder in the project document. Now you only have to click on choose image and then select the right image from the list.

Now click on the picture box and do the same.

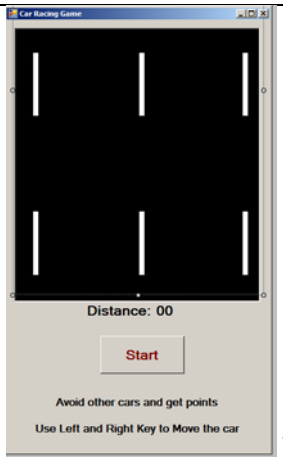


They don't look like what we want them to at the moment so let's change some info in the properties and then they can have the desired effects.

Click on the noted picture boxes and change their properties as mentioned below -

PictureBox1	PictureBox2

Name - roadTrack1 backColor - black Location = -2, -638 Size = 385, 632 SizeMode = StretchImage	Name - roadTrack2 backColor -black Location = -3, -222 Size = 385, 632 SizeMode = StretchImage



This is the view thus far.

Notice how we can't see the overlapping game road on the screen that's because of the panel. The panel kind of only shows that's visible inside it and not what is outside. It also groups the contents inside its height and width.

Now let's add 5 more picture boxes.

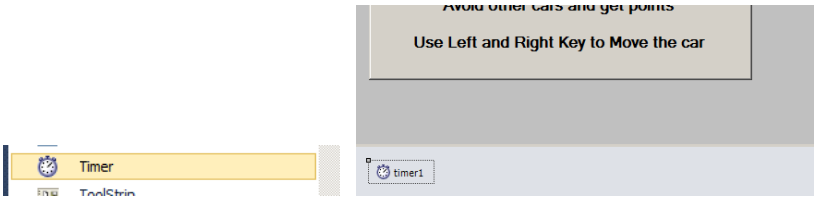
1 - AI Car 2 - AI Car - 3- Player Car 4- Explosion 5 -Trophies

Lets change the images as below

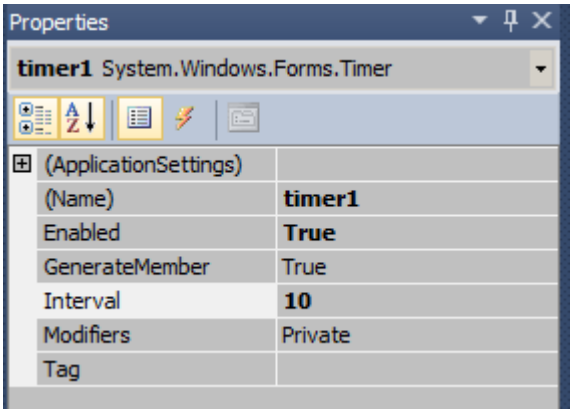
pictureBox1	Name = AI1 backColor = Transparent Image = carGrey (Click on the ... and choose the image) Location = 66, 19 Size = 50, 101 SizeMode = StetchImage	
pictureBox2	Name = AI2 backColor = Transparent Image = carGreen (Click on the ... and choose the image) Location = 294, 85 Size = 50, 101 SizeMode = StetchImage	
pictureBox3	Name = explosion backColor = Transparent Image = explosion (Click on the ... and choose the image) Location = 153, 234 Size = 64, 64 SizeMode = StetchImage	
pictureBox4	Name = player backColor = Transparent Image = carYellow (Click on the ... and choose the image) Location = 161, 286	

	Size = 50, 101 SizeMode = StetchImage	
pictureBox5	Name = trophy backColor = Transparent Image = bronze (Click on the ... and choose the image) Location = 66, 157 Size = 250,100 SizeMode = StetchImage	

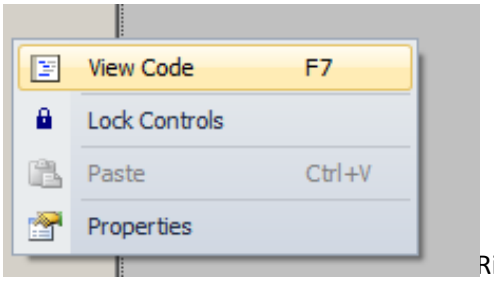
Get the timer object from the tool box and drag it to the form.



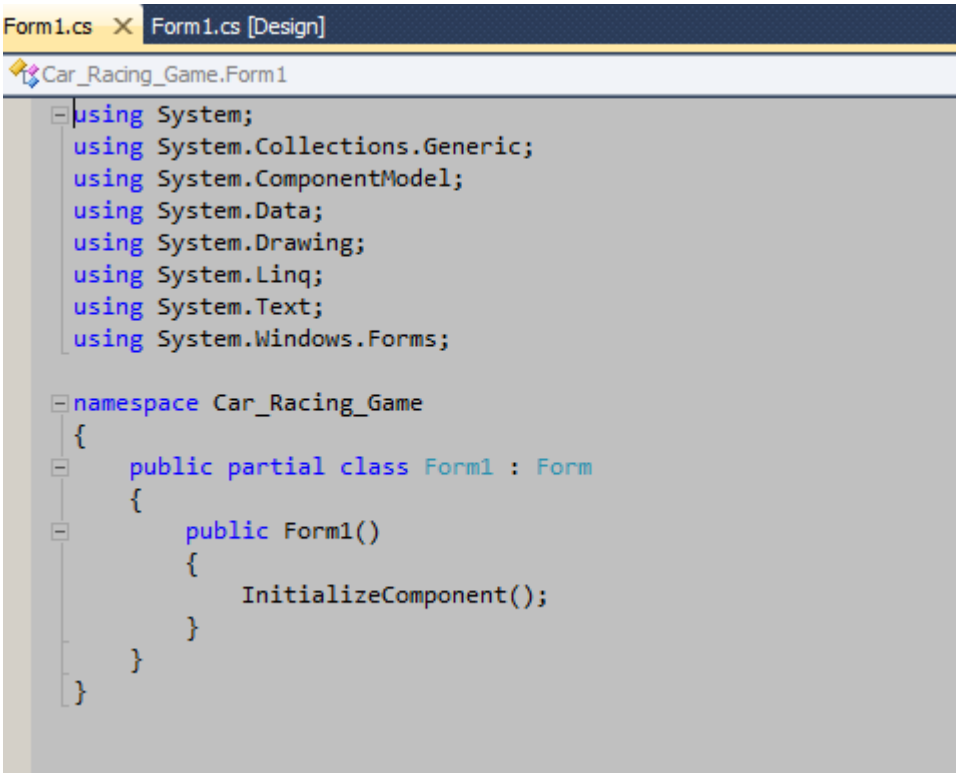
With the timer object added lets change some properties of the timer object.



Now we are done with all of the game graphics. Lets get started on the coding. Yeaayyy



Right Click on the form and Click on View Code or press F7.



This is the default code of the project we are going to add our code in here. Make sure you don't delete any of the brackets unnecessarily.

Enter the highlighted code below

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace Car_Racing_Game
{
    public partial class Form1 : Form
    {
        //global variables
        int carSpeed = 5;
        int roadSpeed = 5;
        bool carLeft;
        bool carRight;
        int trafficSpeed = 5;
        int Score = 0;
        Random rnd = new Random();

        public Form1()
        {
            InitializeComponent();
        }
    }
}
```

See how we made space before the public form function and entered our global variables.

int carSpeed is the speed which our car will move. Int roadSpeed is which our roads will move down wards, bool carLeft and bool carRight are the booleans we will need to move the car left and right based on the key board events, int trafficSpeed is how fast the traffic will move down wards, int score will keep track of how far we travelled and we are also invoking a Random class which will generate the random numbers for us inside the rnd variable.

Remember all of the above are global variables and they can be accessed from anywhere in the form. Unless we declare a variable inside a function then it becomes a local variable which can only be used inside that said function, otherwise we can access them from any function as required.

Lets create a reset function which we can use when the game starts and we can also use it to reset when some one ends the game want to play again.

```
        private void Reset()
        {

        }
```

This is how we declare the reset function.

Now enter the following code inside it.

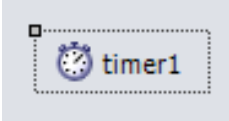
```
        private void Reset()
        {
            trophy.Visible = false; // hide the trophy image

            button1.Enabled = false; // disable the button when game is running

            explosion.Visible = false; // hide the explosion image

            trafficSpeed = 5; // set the traffic back to default

            roadSpeed = 5; // set the road speed back to default

            Score = 0; // reset score to 0

            player.Left = 161; // reset player left
            player.Top = 286; // reset player top

            carLeft = false; // reset the moving left to false
            carRight = false; // reset the moving right to false


            // move the AI to default position this will be off the screen
            AI1.Left = 66;
            AI1.Top = -120;

            AI2.Left = 294;
            AI2.Top = -185;

            //reset the road to their default position
            roadTrack2.Left = -3;
            roadTrack2.Top = -222;
            roadTrack1.Left = -2;
            roadTrack1.Top = -638;

            //start the timer
            timer1.Start();

        }
```

Now lets call this function, we call the function by simply referring back to its name Reset(). If you see this any where in the code it means by simply referring to this we are invoking all its logic to the memory.

```
        public Form1()
        {
            InitializeComponent();
            Reset();
        }
```

Right inside the form1 function under the initialize component function we are calling the reset function. So when the game starts we get right into the action.

Go back to the design view and double click on the timer1 object we added earlier.



<-- Double click this

```
        private void timer1_Tick(object sender, EventArgs e)
        {

        }
```

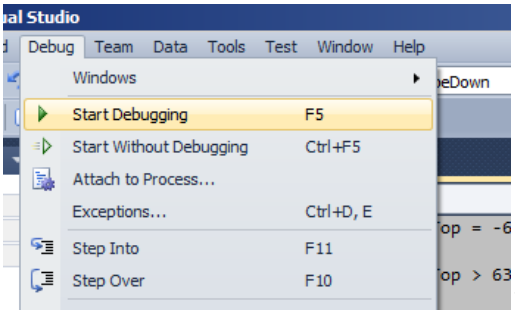
The code above is what visual studio will add to our project, this is the event that will trigger every 10 milliseconds which we set the timer to.

Inside the timer we will add all of our animation and the game rules.

```
        private void timer1_Tick(object sender, EventArgs e)
        {
            Score++; // increase the score as we move

            distance.Text = "" + Score; // show the score on the distance label

            roadTrack1.Top += roadSpeed; // move the track 1 down with the +=
            roadTrack2.Top += roadSpeed; // move the track 2 down with the +=

            // if the track has gone past -630 then we set it back to default
            // this means it will give us a seamless animation
            if (roadTrack1.Top > 630)
            {
                roadTrack1.Top = -630;
            }
            if (roadTrack2.Top > 630)
            {
                roadTrack2.Top = -630;
            }
            // end of track animation.

        }
```

The code above is only some part of Timer 1 function. What we have done here is just to give you an insight into how the game played, now if you want to run the game you can.

Click on the debug menu bar and click on start debugging



This is will run the game.

Now you will see the tracks are moving and distance number is increasing. That was the code so far and it's doing exactly that. Also notice that the button is disabled, AI cars, explosion image and the trophy images are out of the scene.

Here are rest of the timer function. **Remember don't run the code yet because we haven't declared all of the functions mentioned in the game yet. We will prompt you again when to run it.** If run this with the code below it will give you errors because many of the functions mentioned below we haven't declared yet.

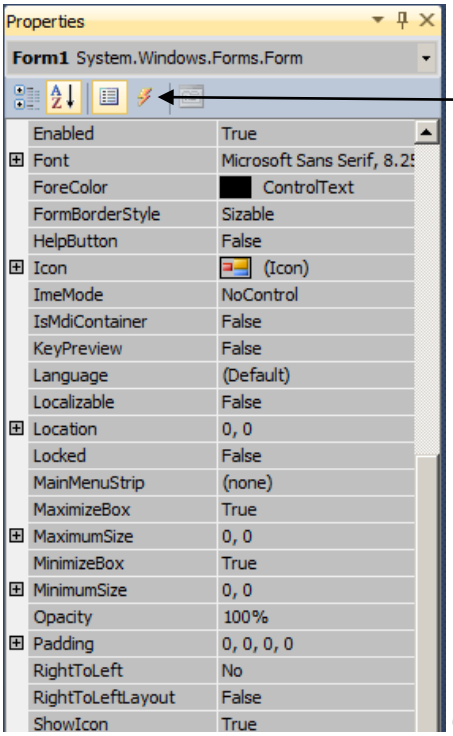
```
        private void timer1_Tick(object sender, EventArgs e)
        {
            Score++; // increase the score as we move

            distance.Text = "" + Score; // show the score on the distance label

            roadTrack1.Top += roadSpeed; // move the track 1 down with the +=
            roadTrack2.Top += roadSpeed; // move the track 2 down with the +=

            // if the track has gone past -630 then we set it back to default
            // this means it will give us a seamless animation
            if (roadTrack1.Top > 630)
            {
                roadTrack1.Top = -630;
            }
            if (roadTrack2.Top > 630)
            {
                roadTrack2.Top = -630;
            }
            // end of track animation.

            if (carLeft) { player.Left -= carSpeed; } // move the car left if the car left is true
            if (carRight) { player.Left += carSpeed; } // move the car right if the car right is true

            // end of car moving

            //bounce the car off the boundaries of the panel
            if (player.Left < 1)
            {
```

```
                carLeft = false; // stop the car from going off screen
            }
            else if (player.Left + player.Width > 380)
            {
                carRight = false;
            }
            // end of the boundaries checks


            //move the AI cars down
            AI1.Top += trafficSpeed;
            AI2.Top += trafficSpeed;

            //respawn the AIs and change the their images
            if (AI1.Top > panel1.Height)
            {
                changeAI1(); // change the AI car images once they left the scene
                AI1.Left = rnd.Next(2, 160); // random numbers where they appear on the left
                AI1.Top = rnd.Next(100, 200) * -1; // random numbers where they appear on top
            }

            if (AI2.Top > panel1.Height)
            {
                changeAI2(); // change the AI car images once they left the scene
                AI2.Left = rnd.Next(185, 327); // random numbers where they appear on the left
                AI2.Top = rnd.Next(100, 200) * -1; // random numbers where they appear on top
            }
            // end of respawning the AIs and image changing

            // hit test the player and AI
            //below if statement is checking multiple conditions
            // if player hits AI1 OR player hits AI2
            if (player.Bounds.IntersectsWith(AI1.Bounds) || player.Bounds.IntersectsWith(AI2.Bounds))
            {
                gameOver(); // this will run when the player hits an AI object
            }
            // end of hit testing the player.

            // speed up the traffic
            // below we are checking for multiple conditions
            // if score is above 100 AND below 500
            if (Score > 100 && Score < 500)
            {
                trafficSpeed = 6;
                roadSpeed = 7;
            }
            // if score is above 500 AND below 1000
            else if (Score > 500 && Score < 1000)
            {
                trafficSpeed = 7;
                roadSpeed = 8;
            }
            // if score is above 1200
            else if (Score > 1200)
            {
                trafficSpeed = 9;
                roadSpeed = 10;
            }
            // end of the traffic speeding up

        }
```

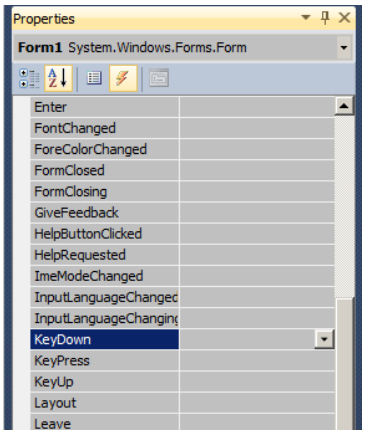
The code is explained inside with the comments.

Now let's go back to the design view once more to add two events to the FORM.

Click on the form and look at the properties window.



Click on that lightning bolt icon



We are interested in the two events **KeyDown** and **KeyUp**

In the key down event type moveCar and press enter. go back and type stopCar in the key up event.

Type the following in the move car function.

```
        private void moveCar(object sender, KeyEventArgs e)
        {
            //if the player pressed the left key AND the player is inside the panel
            // then we set the car left boolean to true
            if (e.KeyCode == Keys.Left && player.Left > 0)
            {
                carLeft = true;
            }
// if player pressed the right key and the player left plus player width is less then the panel1 width
            // then we set the player right to true
            if (e.KeyCode == Keys.Right && player.Left + player.Width < panel1.Width)
            {
                carRight = true;
            }
        }
```

Type the following in the stop car function

```
        private void stopCar(object sender, KeyEventArgs e)
        {
            // if the LEFT key is up we set the car left to false
            if (e.KeyCode == Keys.Left)
            {
                carLeft = false;
            }
            // if the RIGHT key is up we set the car right to false
            if (e.KeyCode == Keys.Right)
            {
                carRight = false;
            }
        }
```

Now let's create the changeAI1 function.

```
        private void changeAI1()
        {
            int num = rnd.Next(1, 8); // we set up a local variable to generate a number between 1 and 8
            // by using a switch statement we can show any image based on that number
            // switch statement will check which number has been generated and will displayer the images as requested
            switch (num)
            {
                 // if the number generated is 1 we show the green car
                case 1:
                    AI1.Image = Properties.Resources.carGreen;
                    break;
                // if the number generated is 2 we show the grey car
                case 2:
                    AI1.Image = Properties.Resources.carGrey;
                    break;
                // if the number generated is 3 we show the orange car
                case 3:
                    AI1.Image = Properties.Resources.carOrange;
                    break;
                // if the number generated is 4 we show the pink car
                case 4:
                    AI1.Image = Properties.Resources.carPink;
                    break;
                // if the number generated is 5 we show the red car
                case 5:
                    AI1.Image = Properties.Resources.CarRed;
                    break;
                // if the number generated is 6 we show the blue truck
                case 6:
                    AI1.Image = Properties.Resources.TruckBlue;
                    break;
                // if the number generated is 7 we show the white truck
                case 7:
                    AI1.Image = Properties.Resources.TruckWhite;
                    break;
                // if the number generated is 8 we show the ambulance
                // this is why its important to name your images so they make logical sense
                case 8:
                    AI1.Image = Properties.Resources.ambulance;
                    break;
                default:
                    break;
            }
        }
```

Since all of the images are an object we can change their properties in the code. In this case we are using a switch statement to change the images based on the random number generated. The logic behind this is each time the car leaves bottom of the play area this function will run generate a random number between 1 and 8 and then it will change the car based on the number.

`AI1.Image = Properties.Resources.carGreen;`

The line above is where we are accessing the images we imported into the resources before. This is why it's important to name your images because if we called the images 3dg7I.jpg we would never guess what it is plus visual studio doesn't show the extension of the file either from the resources.

the changeAI2 function is exactly the same.

only with AI2 this time.

```
        private void changeAI2()
        {
            int num = rnd.Next(1, 8); // we set up a local variable to generate a number between 1 and 8
            // by using a switch statement we can show any image based on that number
            // switch statement will check which number has been generated and will displayer the images as requested
            switch (num)
            {
                // if the number generated is 1 we show the green car
                case 1:
                    AI2.Image = Properties.Resources.carGreen;
                    break;
                // if the number generated is 2 we show the grey car
                case 2:
                    AI2.Image = Properties.Resources.carGrey;
                    break;
                // if the number generated is 3 we show the orange car
                case 3:
                    AI2.Image = Properties.Resources.carOrange;
                    break;
                // if the number generated is 4 we show the pink car
                case 4:
                    AI2.Image = Properties.Resources.carPink;
                    break;
                // if the number generated is 5 we show the red car
                case 5:
                    AI2.Image = Properties.Resources.CarRed;
                    break;
                // if the number generated is 6 we show the blue truck
                case 6:
                    AI2.Image = Properties.Resources.TruckBlue;
                    break;
                // if the number generated is 7 we show the white truck
                case 7:
                    AI2.Image = Properties.Resources.TruckWhite;
                    break;
                // if the number generated is 8 we show the ambulance
                case 8:
                    AI2.Image = Properties.Resources.ambulance;
                    break;
                default:
                    break;
            }
        }
```

Now let's create the game over function. This function is set to run when the player has hit the other AI's on the screen.

```
         private void gameOver()
         {
             trophy.Visible = true; // change the trophy to visible

             timer1.Stop(); // stop the timer

             button1.Enabled = true; // enable the button so we can use it now

             //showing the explosion image on top of the car image
             explosion.Visible = true; // make the image visible
             player.Controls.Add(explosion); // add the explosion image on top of the player image
             explosion.Location = new Point(-8, 5); // we are moving the image so its suitably positioned
             explosion.BackColor = Color.Transparent; // change the background to transparent
             explosion.BringToFront();// bring to front of the player image

             // final score trophy

             // if the player score less than a 1000 we give them a bronze
             if (Score < 1000)
             {
                 trophy.Image = Properties.Resources.bronze;

             }
             // if player scored more than 2000 then give them a silver
             if (Score > 2000)
             {
```

```
                trophy.Image = Properties.Resources.silver;
            }

            // if player scored more than 3500 then give them a gold trophy
            if (Score > 3500)
            {
                trophy.Image = Properties.Resources.gold;
            }
        }
```

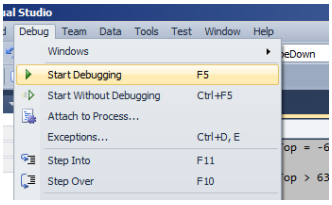
We are making all of the images we made invisible back to visible and enabling the button so we can interact with it.

Now lets go back to the design view and double click on the button.


```
        private void button1_Click(object sender, EventArgs e)
        {
            Reset();
        }
```

inset the reset function calling inside the button 1 function.

Each time we end the game we can the restart the game. Now let's run the game.



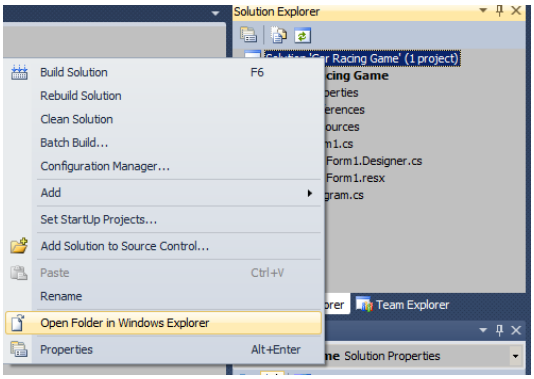
Start Debugging the game.

Ha! it worked.

The GIF is animating on top of the car.
Timer stopped
The car isn't going off the borders left or right
The trophy is showing
AI cars are changing and coming back
Traffic speeds up as the scores goes up

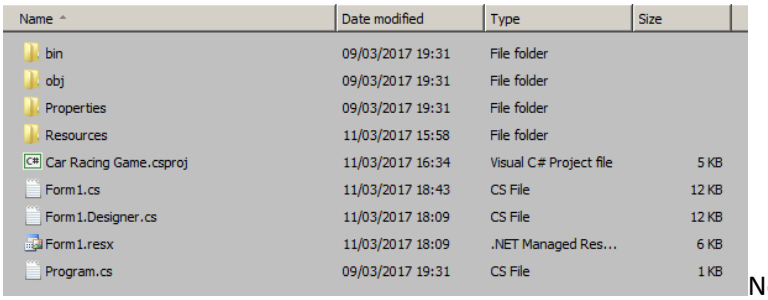
One last thing, SOUND.

We have a WAV file called hit we need to move it in the debug folder.



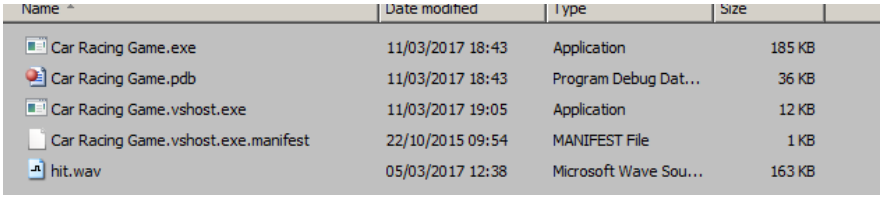
Right Click on the solutions explorer and click on open folder in WIndows Explorer.

All the resources are going to be inside the Car Racing Game folder.



Now go inside the bin/debug folder

Here paste the hit.wav file



Now lets go back to visual studio we need to create a new function

This one will be called play sound

```
        private void playSound()
        {
            System.Media.SoundPlayer player = new System.Media.SoundPlayer(@"hit.wav");
            player.Play();
        }
```

In this function we are creating a new instance of the class sound player and we are directing it to the hit.wav file we pasted before. Now if you notice we didn't give the file any folder directory, now when the game runs it will look for the hit.wav file where that EXE file is that is why we pasted it in the bin/debug folder.

Now we need to run this function in the game over function.

So inside the game over function mention playSound();

```
        // if player scored more than 3500 then give them a gold trophy
        if (Score > 3500)
        {
            trophy.Image = Properties.Resources.gold;
        }

        playSound(); // play the hit sound
    }
```

Now play the game and test it out.

Be proud of the game you created.

Well done.