

Create a Maths quiz game in Visual studio with C#

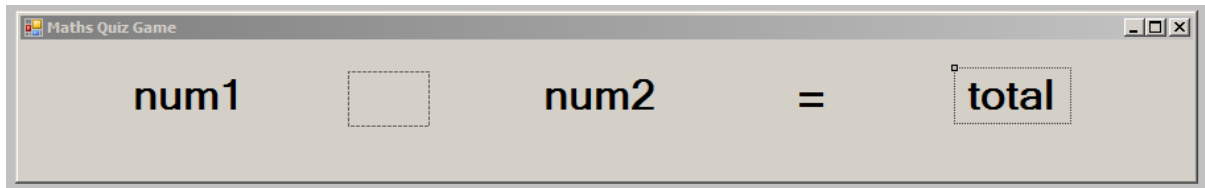
Guess the symbol

This you will need

resource images for the symbols

4 labels

1 picture box called **symbol**



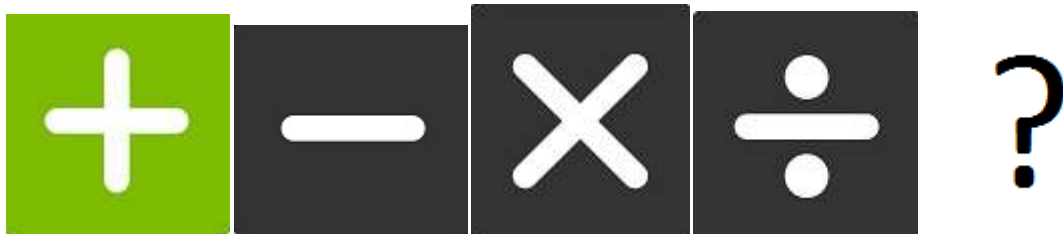
num1 will show the first number

num2 will show the second number

= sign is a label

total will show the total of sum.

empty rectangle in the middle of num1 and num2 is a picture box.



These are the four symbols which we will use in the picture box to represent the sum instructions.

Lets add a instructional label on the screen for the player

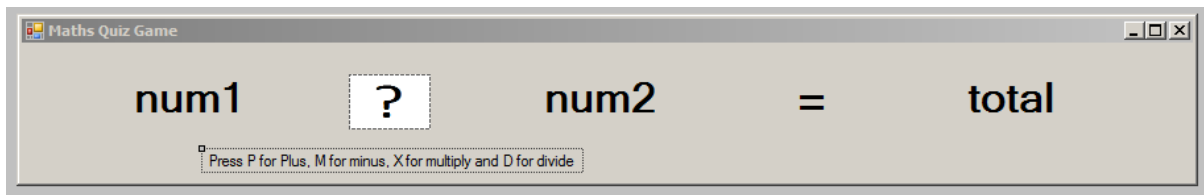
Instruction is

press p for plus

press m for minus

press x for multi

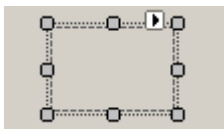
press d for divide



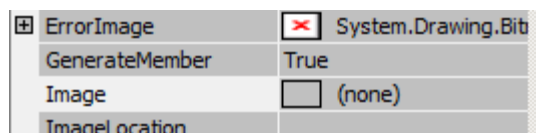
this is the game screen with the instructions for the user.

you can off course use your own initiation and style it any way you want to.

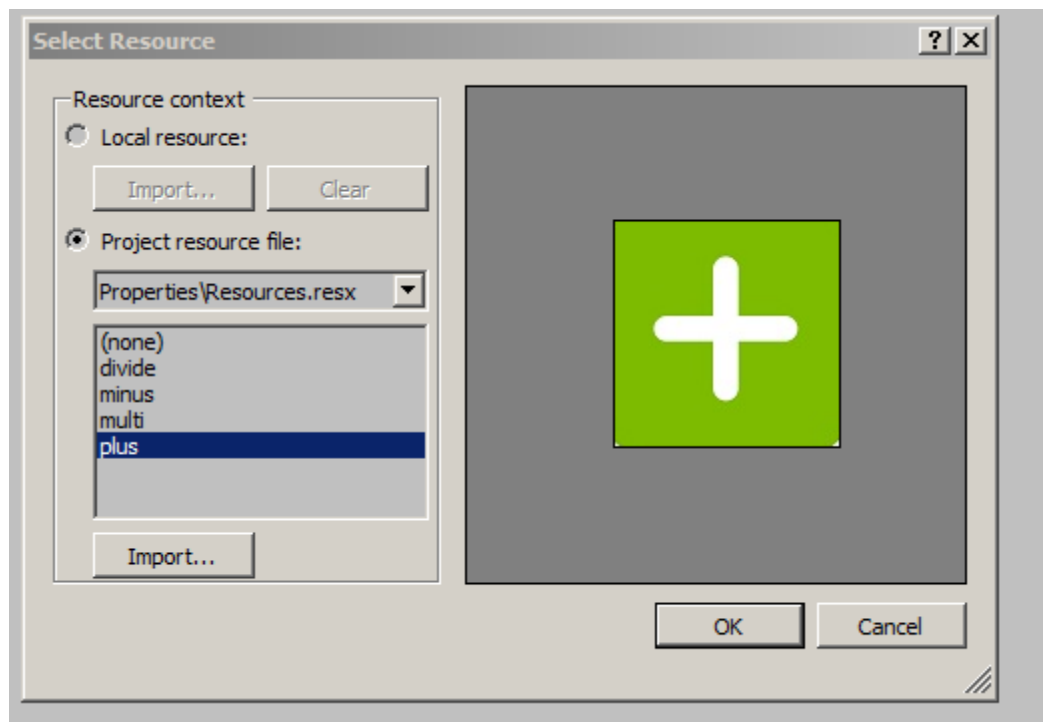
click on the picture box then click on the small triangle on the corner



or you can click on the image properties in the properties panel.

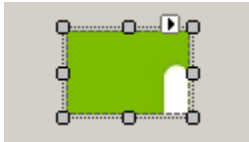


add the four images to the project resource file

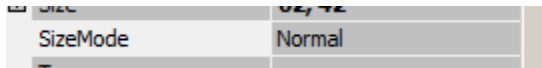


clearly the picture box is too small for the image. We need to change the size mode in the properties panel.

This is what it looks like now

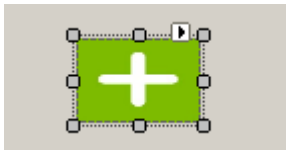


This is the size model further down in the properties menu



Change it to Stretch Image

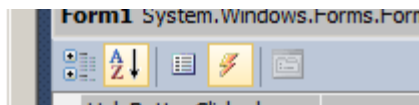
It will fit the image according to the size of the picture box. Problem solved.



Now lets add some events to this form.

Click on the form and click events button in the properties panel.

Its the lightning bolt icon



Find the KEYPRESS event and type KeyisDown and press enter.

This will create and link a key press event to the form.

Now we can change the image according to the key we press while the application is running.

Enter the following code in the keyisdown function

```
if (e.KeyChar == (char)Keys.P || e.KeyChar == 112)
{
    symbol.Image = Properties.Resources.plus;
}
else if (e.KeyChar == (char)Keys.M || e.KeyChar == 109)
{
    symbol.Image = Properties.Resources.minus;
}
else if (e.KeyChar == (char)Keys.X || e.KeyChar == 120)
{
    symbol.Image = Properties.Resources.multi;
}
else if (e.KeyChar == (char)Keys.D || e.KeyChar == 100)
{
    symbol.Image = Properties.Resources.divide;
}
```

We are using ASCII code for the characters and we are using the upper case characters.

C# compiler treats the upper case and lower case characters are two separate types. Therefore we are giving it the value twice.

Here is the table which shows the values we are looking for:

P	112
M	109
X	120
D	100

instead of creating 8 if statements we have entered double conditions in each to save time and space.

let me explain what we are doing in them

inside the if statement `e.KeyChar == (char)Keys.P` this line is looking if the key character matches with capital P then we have a this symbol twice `||` which is the key right next to your Z button on the keyboard. press the shift key and press it twice it will appear on the screen.

Inside a if statement this is used to point out OR.

Basically we are saying to C# if the user presses capital P or lower case P we need to do this.

neat right. after the or we are using the ASCII value of lower case p to trigger the same event.

full like is this: `e.KeyChar == (char)Keys.P || e.KeyChar == 112`

Now lets see if you can read that line of code?

if key character is equals to capital P or key character is equals to 112 (lower case P ASCII code) then do something.

Run the application now and press those keys and watch the picture box change with it.

Thats enough fun for now. Lets get to work.

Enter these 4 variables for the application before the public Form1() line.

These are whats called global variables. Global variables can be used in any function. Local variables can only be used in one function in which its decaled.

```
int numA;  
int numB;  
int totalA;  
int randMath;  
int check;
```

We will now need a playGame function

```
public void playGame()  
{  
    Random ranNumber = new Random();
```

more tutorials on www.mooict.com

```

        numA = ranNumber.Next(1, 40);
        numB = ranNumber.Next(1, 50);
        randMath = ranNumber.Next(1, 5);
        symbol1.Image = Properties.Resources.qq;

```

```

        num1.Text = Convert.ToString(numA);
        num2.Text = Convert.ToString(numB);
        total.Text = Convert.ToString(totalA);
    }

```

Random ranNumber = new Random(); in this line we are calling the random class from C# to help us generate random numbers to do calculations with.

numA = ranNumber.Next(1, 40); in this line we are assigning a random number between 1 and 40 to numA variable.

numB = ranNumber.Next(1, 50); in this line we are assigning a random number between 1 and 50 to numB variable.

randMath = ranNumber.Next(1, 5); this one we are generating a random number between 1 and 4 means 4 numbers in between. This one will be used to check which maths formula are we using.

symbol.Image = Properties.Resources.qq; this line is resetting the picture box to the question mark image we have imported earlier. This is to entice the player to guess it.

it will clear up soon enough for you.

```
num1.Text = Convert.ToString(numA);
```

```
num2.Text = Convert.ToString(numB);
```

```
total.Text = Convert.ToString(totalA);
```

These 3 lines above are simply converting the numbers to text and displaying on the screen.

Now we need add 4 functions to do Maths.

after the play game function add the following

```

public void addNum()
{
    totalA = numA + numB;
}
public void subtractNum()
{
    totalA = numA - numB;
}
public void multiplyNum()
{
    totalA = numA * numB;
}
public void divideNum()
{

```

more tutorials on www.mooict.com

```
        totalA = numA / numB;  
    }
```

You know what these do now don't you. title hint hint.

Notice the **divideNum** is using an integer which means it will often return 0 for many calculations. can convert it to double or float to accept decimal numbers to show accurate calculations. I will leave that your best judgements.

Now back to our play game function.

We want to create a function which will randomly do a maths formula and the user will have to guess which is it.

We will use a switch statement to achieve this.

What is a switch you ask.

A switch statement is kind of like the if statement. If a certain condition is true it will trigger the actions we tell it to do.

this is a example

switch (**age**)

```
    case 11:  
        at school;  
        break;  
    case 18:  
        at college;  
        break;  
    case 25:  
        working;  
        break;  
    etc etc
```

according to the age which is entered this conditional statement will return a value back to us for example at 11 at school, at 18 at college and at 25 working.

Lets see how we can apply it to our maths game.

```
public void playGame()  
{  
    Random ranNumber = new Random();  
    numA = ranNumber.Next(1, 40);  
    numB = ranNumber.Next(1, 50);  
}
```

more tutorials on **www.mooict.com**

```

randMath = ranNumber.Next(1, 5);
symbol.Image = Properties.Resources.qq;

```

```

switch (randMath)
{
    case 1:
        addNum();
        break;
    case 2:
        subtractNum();
        break;
    case 3:
        multiplyNum();
        break;
    case 4:
        divideNum();
        break;
}

num1.Text = Convert.ToString(numA);
num2.Text = Convert.ToString(numB);
total.Text = Convert.ToString(totalA);
}

```

Above is the whole playGame function.

Notice anything different. It has a switch (randMath) condition. So remember when I said randMath will help us with the maths formula.

randMath = 1 it will do add

randMath = 2 it will do subtract

randMath = 3 it will do multiply

randMath = 4 it will do divide

makes sense. Also it will help us to check if the user input is accurate.

Lets create a checkGame function to verify that

```

public void checkGame()
{
    if (check == randMath)
    {
        switch (randMath)
        {
            case 1:
                symbol.Image = Properties.Resources.plus;
                MessageBox.Show("Correct");
                playGame();
                break;
            case 2:
                symbol.Image = Properties.Resources.minus;
                MessageBox.Show("Correct");
                playGame();
                break;
            case 3:
                symbol.Image = Properties.Resources.multi;

```

more tutorials on www.mooict.com

```

        MessageBox.Show("Correct");
        playGame();
        break;
    case 4:
        symbol.Image = Properties.Resources.divide;
        MessageBox.Show("Correct");
        playGame();
        break;
    }
}
else
{
    MessageBox.Show("Wrong Answer");
}
}

```

Here is the whole check game function.

As you have notice I moved the symbol image change to this one inside another switch condition which is inside a if condition. Inception much but we can go deeper.

We are check if the check variable is equals to randMath. Check variable will change according to the player input

for example

P = 1

M = 2

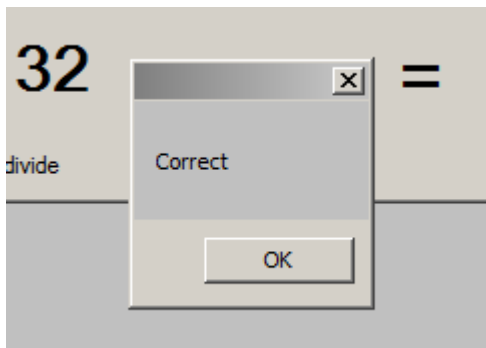
X = 3

D = 4

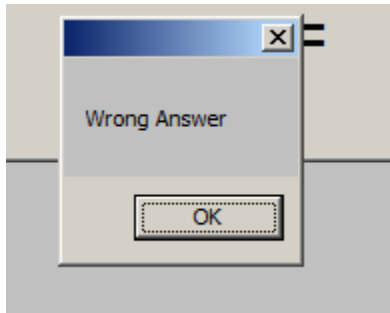
neat right. We will enter that in the keyisdown function in a minute.

Lets go inside this function

switch will only run if the condition is true first after that it will check which maths formula did it run before according to that it will show the image and then it will show a message box stating correct.



if it encounters the player pressed the wrong button it will show wrong answer message box



making sense. Okay

Time for the final part of the puzzle.

```
private void KeyisDown(object sender, KeyPressEventArgs e)
{
    if (e.KeyChar == (char)Keys.P || e.KeyChar == 112)
    {
        check = 1;
        checkGame();
    }
    else if (e.KeyChar == (char)Keys.M || e.KeyChar == 109)
    {
        check = 2;
        checkGame();
    }
    else if (e.KeyChar == (char)Keys.X || e.KeyChar == 120)
    {
        check = 3;
        checkGame();
    }
    else if (e.KeyChar == (char)Keys.D || e.KeyChar == 100)
    {
        check = 4;
        checkGame();
    }
}
```

Here is the keyisdown function.

As you can see each key is changing the check variable accordingly and it is setting it up for the checkgame to do its magic. Well there is no magic only programming. Now run the program if you will and see if that works.

If it does not work then please double check the code here and follow anything that you might of missed.

Errors = learning opportunity

Make it count.