Visual Basic Tutorial - make a binary calculation game

Welcome to this binary calculation tutorial. In this exciting tutorial we will create a fully functional binary calculation game using Visual Studio 2010 in Visual Basic programming language. To start off we need to learn about binary and then take it from there.

Our program will be able to do the following

1. Use combo drop down menu to select from 0 to 1
2. Use loops to find which combo box is active
3. How to add values to a string using a loop
4. Ask random question to player and make them match the number in binary
5. Check if the answer is right then present a new question
6. Check if the answer is wrong then add 1 to the tried option
7. Show the actual binary number on screen
8. Show the denary/decimal number on the screen
9. Show message box
10. Reset the combo boxes

We are creating a 8 bit binary game. Take a look at the table below

128	64	32	16	8	4	2	1

As you can see in the table binary calculations start with 1 then 2 then 4 and so on. Each number is double of the last number. Binary has only two numbers 0 and 1.

If we want to calculate 1 in 8 bit binary it looks something like this

128	64	32	16	8	4	2	1
0	0	0	0	0	0	0	**1**

00000001 = 1 (1)

Now calculate 10

128	64	32	16	8	4	2	1
0	0	0	0	**1**	0	**1**	0

0001010 = 10 (8+2)

Now calculate 160

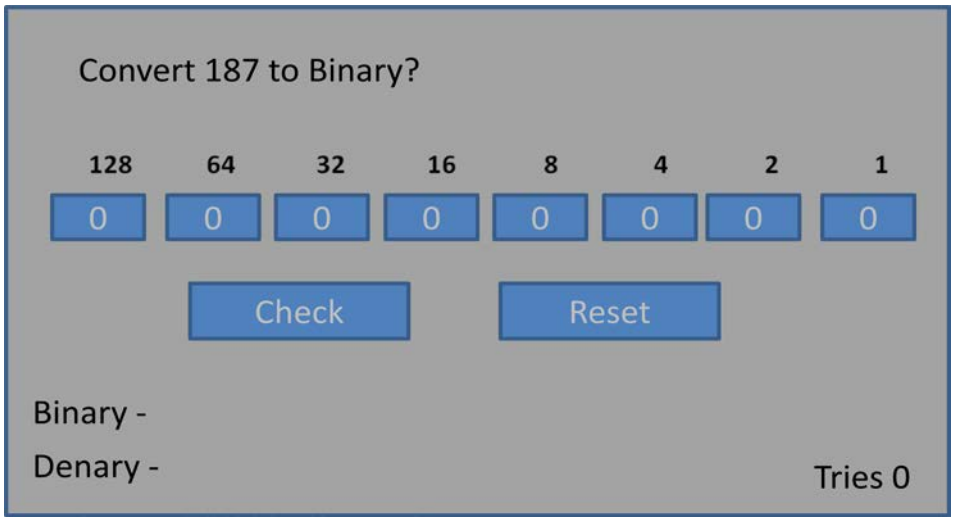
128	64	32	16	8	4	2	1
1	0	**1**	0	0	0	0	0

10100000 = 160 (128 + 32)

Can you calculate 14, 17, 80 and 99?

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

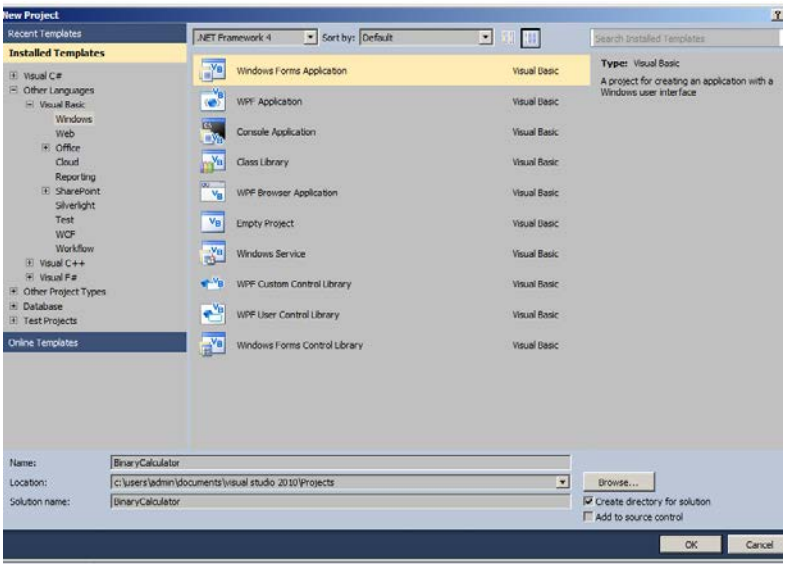
1 means that sector is active and 0 is inactive. Computers only understand numbers so we found a way to manipulate numbers to help the processor break it down in binary and then convert it back to give us the result. Everything we know in computing is binary, every file and every action we take.



Here is a layout design for the binary application. We don't want our user to get discouraged answering the questions so we won't be keeping scores however for each questions users can see how many times they have tried it. Once they move on the next question tires will be reset to 0.

All those blue boxes that you see are combo or drop down boxes. In there we will have 2 values 1 and 0.

First Lets start Visual Studio and Choose a new Windows Form Project



Name the project Binary Calculator and click on OK.

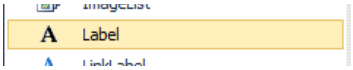
more tutorials on **www.mooict.com**

Here is our empty Form. In this form we will build our binary calculator game. It's going to be awesome. Let's start by changing few things in the properties.

ShowInTaskbar	True
Size	**620, 310**
SizeGripStyle	Auto
StartPosition	WindowsDefaultLocation
Tag	
Text	**Binary Calculator**
TopMost	False
TransparencyKey	

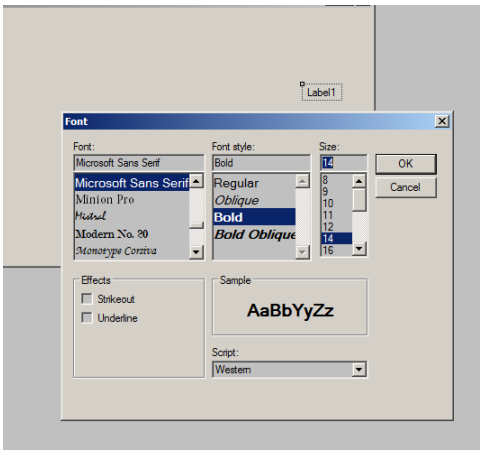
Change the Text to Binary Calculator

Change the size 620, 310 (620 Width and 310 Height)

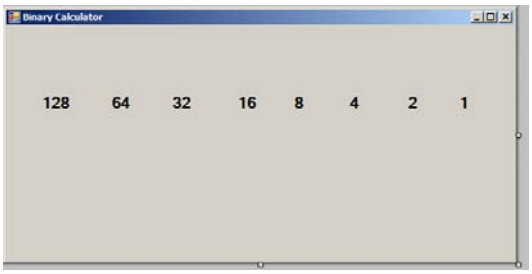
Here is the empty canvas. Now let's add some Labels to it to begin with. 

We will start adding the labels for the numbers first. So we will go from Right to Left.

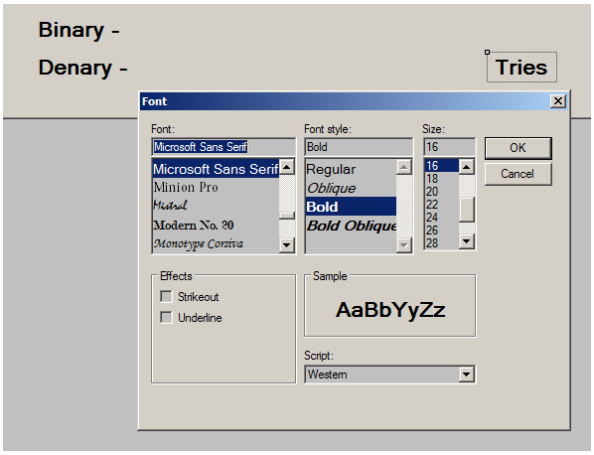
Each label will have the same properties in this line we want them to have 14 font size and Bold effect.



You can do this for each of them. Organise them in a spacious way so it doesn't look cluttered.



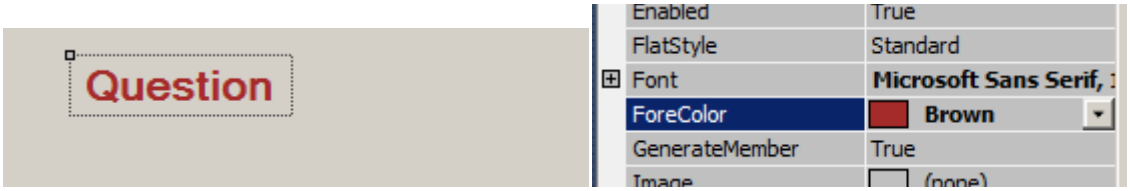
Lets add 3 more labels in the bottom.



All the of the labels are font 16 and Bold.

The binary label will show the binary result of the equation, denary will calculate the sum of the binary and show them in numbers and lastly the tries label will show many attempts the player took.

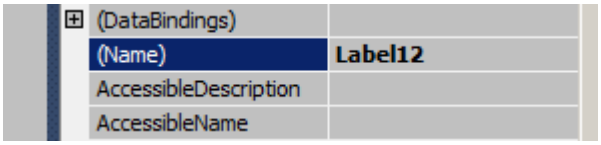
Once last label left to do which is the title label and this one will be asking the question for the players to answer.



The question title label is the same as the binary and denary once except we changed the fore colour to Brown.

Now to make rest of the development easier for us lets change some properties for the last 4 labels we have added to the project.

In my case the question label is the which is Label12

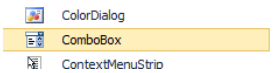


We want to change the name so it's easier for us to call this specific component in the code without remembering which number they are. Makes sense?

Good

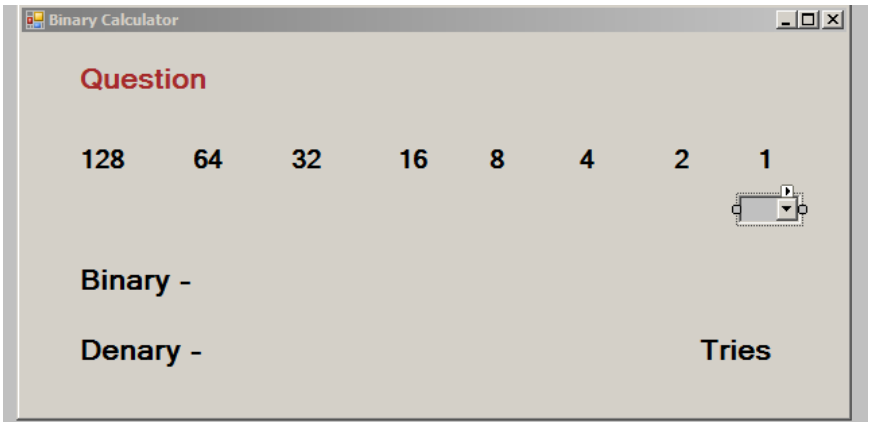
For the question label change the name to question (all lower case), binary label change to binary (all lower case), denary change to denary, tries change tries. (make sure all of them are in lower case)

Now its time to add the combo boxes



Before you go in gun blazing let's consider some serious design issues which might crop up later on. We will be adding 8 different combo boxes and they have to placed in order. Now later on this will become

very clear on why we did this but for now in order follow along this tutorial you need to place the combo boxes from right to left. Make sure the box 1 is under 1 and box 2 is under 2 and so on.

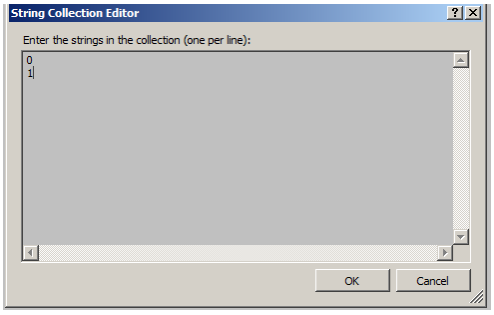


Add your first combo box under 1 then 2 then 4 and so on. We need the combo boxes to be in order so when we run the loop it will organise their properties in appropriate ways for us. For now lets add some items inside this combo box.

Click on the combo box and look in the properties window - >

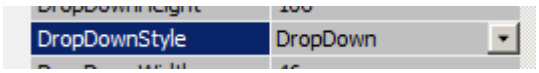
Find the items option 

Click on the 3 dots



Add 0 enter 1. We are adding the values which can be used to calculate the binary. Click on once done.

Look in the properties window again for DROP DOWN STYLE option

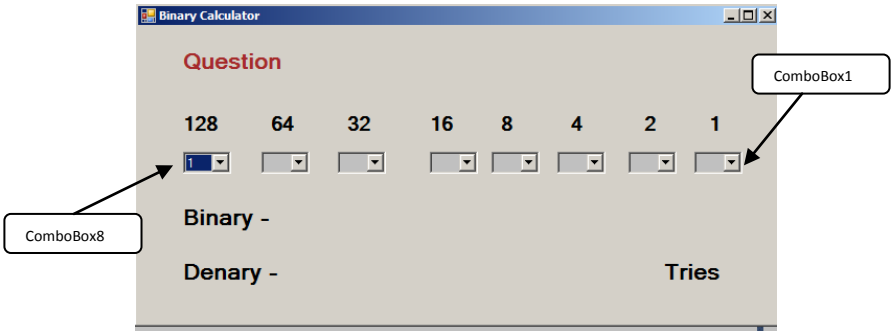


Change this to drop down list

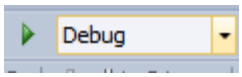
Yep

Now add the rest of the combo boxes with the same settings.

Check back when you are done.



This is what it looks like now. So lets run this application to get a feel for it.

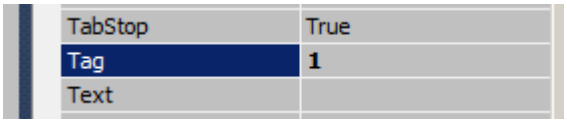
Pres F5 or Click on the play button on the top 

Now comes the important bits.

Make sure that you added the combo boxes from right to left meaning ComboBox1 in the far right all way to ComboBox8 on the far left.

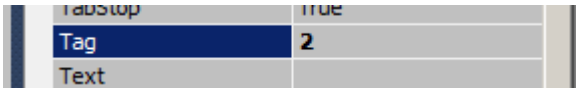
So inside each of the combo boxes there is a property option called TAG we will be using this to carry the value of each binary section.

Click on the combo box under 1



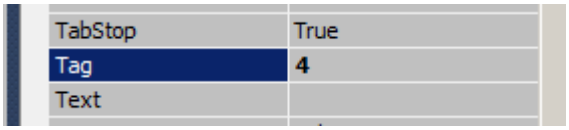
We added 1 inside this tag option.

Now click on the combo box under 2



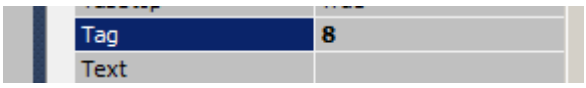
We added 2 inside this tag option.

Now click on combo box under 4



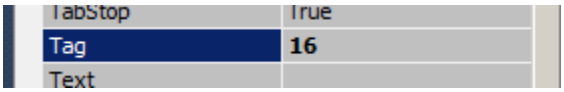
We added 4 inside this tag option

Now click box under 8



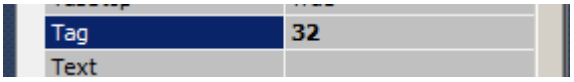
We added 8 inside this tag option

Now click under 16



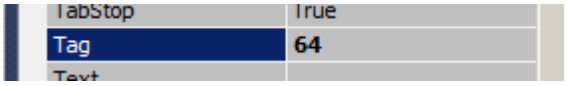
We added 16 inside this tag option.

Now click on the combo box under 32



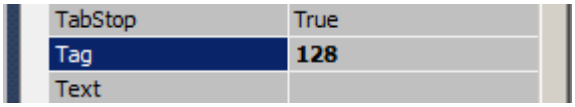
We added 32 inside this tag option.

Now click on the combo box under 64



We added 64 inside this tag option.

Now click on the combo box under 128



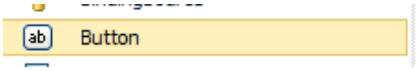
We added 128 inside this tag option.

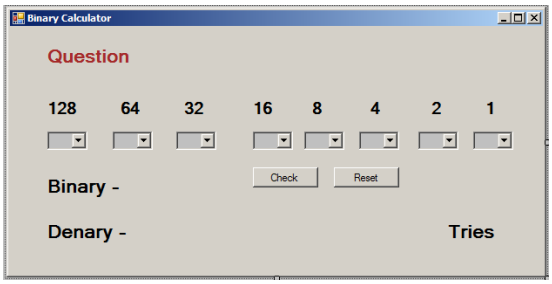
Okay now all done.

Next section we need two buttons on the stage.

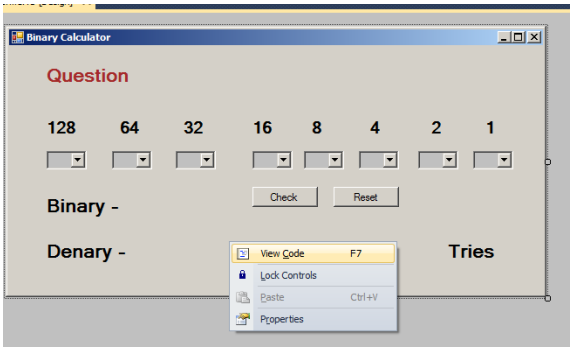
1 will be called Check

2 will be called Reset



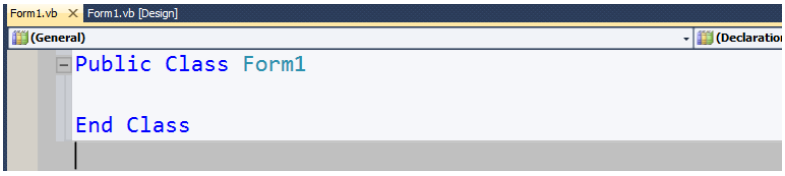


Here we are. All done with the graphical user interface now it's time to go in the background and do some programming.



To start lets right click on the form and Click on View Code

This will take us to the programming window



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

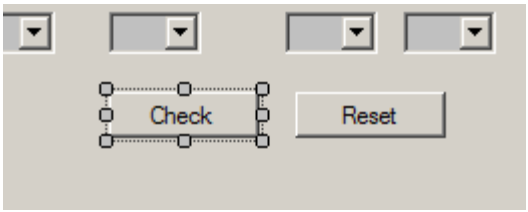
Everything we will code will go between the Class and before the end class. Make sure nothing you are doing is outside the end class line.

Lets add our first few variables

```
Public Class Form1

    Dim total As Integer 'save the total for each binary calculated
    Dim randomNum As Integer 'generates a random number
    Dim attempt As Integer 'will be keeping log of the tries in the game

End Class
```

We are declaring 3 integers in the above. Total integer will save the total of our binary into denary, random Num will generate a random number for us and attempt will keep track of the number of tries. We have commented the code on the right because its a good practice to do so and we will encourage you to develop that habit. It will save a lot of time and when working on a paid project will save money too.

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




```
Public Class Form1

    Dim total As Integer 'save the total for each binary calculated
    Dim randomNum As Integer 'generates a random number
    Dim attempt As Integer 'will be keeping log of the tries in the game

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

    End Sub
End Class
```

Visual Studio will add the button1_click function. All the of the code for this button will go there before the end sub like. Make sure no code is entered after the End Sub line because it will throw an error.

```
        'reset the binary and denary labels
        binary.Text = "Binary -"
        denary.Text = "Denary -"
```

Add these two lines first. Remember when we changed the names of the labels to easily remember which we are working on, this is why. Now calling them binary and denary makes more sense and we don't need to use label 10 or 11.

Now under that last line add the following:

```
        'this loop will go through the combo boxes and check which is active
        'it will check which is active by search for the 1 in binary that means active.
        For Each ctl As Control In Me.Controls
            If TypeOf ctl Is ComboBox Then
                If ctl.Text = "1" Then
                    total += Convert.ToInt16(ctl.Tag)
                End If
                binary.Text += ctl.Text 'show the binary numbers on that binary label
            End If
        Next
        'end of the loop
```

Here we are using a For loop. This for loop is specially designed to find the combo boxes available in the form. We start the loop by `For Each ctl As Control In Me.Controls` We are first giving the loop a variable called ctl its just a name the type of this variable is Control meaning system components such as buttons, labels, text boxes or even combo boxes.

Then we running a IF statement `If TypeOf ctl Is ComboBox` Then which will check if that specific ctl is a type of combo boxes in the form, if true it will do the following.

Now we are going to run another If statement to check if the combo box is active meaning it has the value one present `If ctl.Text = "1"` Then if so then do the following.

Remember those tags we added before here is why `total += Convert.ToInt16(ctl.Tag)`
as you can see we are adding the total by converting the tags to integers and then adding them up.

Then we end that If statement by stating End If

lastly **binary.Text += ctl.Text** means that we want to add all of the 0s and 1s to that label.

End the IF statement by saying End If

End the loop by saying Next.

Under the end loop line add this line

```
denary.Text += total.ToString() 'show the total in numbers from binary
```

This will show the total in normal numbers on the denary label.

Add the following code under that denary text one

```
        'if statement below will check whether the player has accurately solved the binary
        'if so it will show the correct message box
        'else it will add 1 to the attempt variable
        If total = randomNum Then
            MessageBox.Show("Correct")
            randomNum = Rnd() * 255
            question.Text = "Convert " + randomNum.ToString() + " To Binary"
            attempt = 0
            tries.Text = attempt.ToString()
            resetCombo()
        Else
            attempt += 1
            tries.Text = attempt.ToString()
        End If
```

Here we are running a IF statement to check if the maths of this game and whether the player has made a correct calculation.

If the total is equals to random number then we will show a message box to say its correct, generate another random number between 0 and 255, show that random number on the screen, reset the attempts to 0, show the tires on the screen, we will run the reset combo function(we haven't created this yet so don't run the program it will not run yet.)

If the user calculation is not correct then it will add 1 to the attempt and then show it on the tries label.

```
        Button1.Enabled = False
        Button2.Enabled = True
```

Lastly for the button1_click function lets add the following code in there. The above 2 lines will simply disable the check button and enable the rest button. This way the player cannot cheat. Nice right.

Now lets make that reset combo function.

```
    Private Sub resetCombo()

        'all the codes will go here

    End Sub
```

Here is the empty reset Combo. This function has one primary duty which is to reset all of the combo boxes to 0. In binary 0 means inactive therefore by resetting them all our players can active which they see fit.

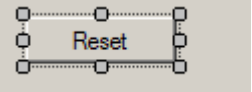
Lets add the codes in there now.

```
        'for loop to check the combo boxes in form
        'once they are found we reset all their values to 0
        For Each ctl As Control In Me.Controls
            If TypeOf ctl Is ComboBox Then
                ctl.Text = "0"
            End If
        Next

        'reset all of the labels in the game too
        total = 0
        denary.Text = "Denary- "
        binary.Text = "Binary- "
```

Here you can see that we are using the similar for loop as we did before. Instead of adding anything we are simply checking for the combo boxes which exist on the form then we are resetting their values to 0.

After that we are also resetting the labels and variables to 0. Notice we did not include the attempt variable this is because we want to keep track of the user errors until they get it right.

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



Visual Studio will add the following code for us

```
Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click

End Sub
```

Now we will add our own code inside this. Since we have created a separate reset combo function we can just simply call on it and it will do the job for us. This is called reusing code without rewriting them.

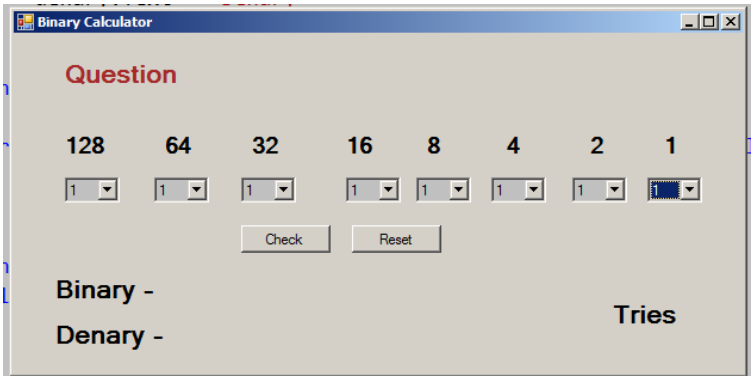
```
        resetCombo() 'this is how we call the reset function
        Button1.Enabled = True 'enable the check button
        Button2.Enabled = False 'disable the reset button
```

Here we are calling the reset function first, this will set all of the combo boxes to 0. And then enable the check button and disable the reset button.

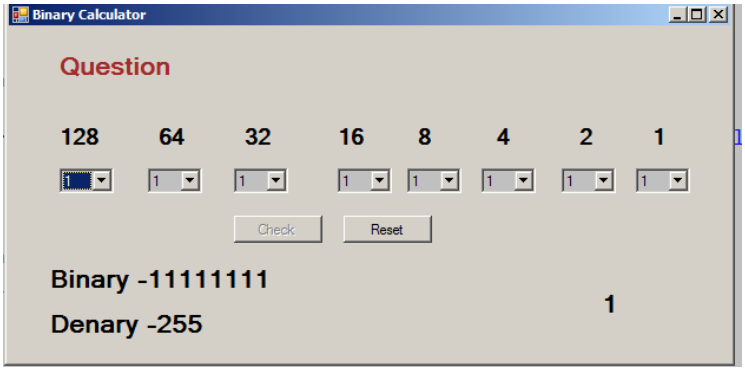
Lets run the program now to check a few things

Click on Run or Press F5 as before

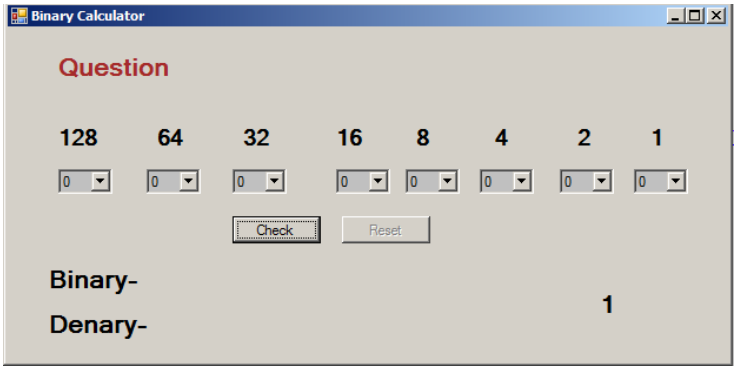
I've set all of the combo boxes to 1 meaning they are active. Adding them together should be 255 total.



Binary and Denary both are correct in this case. Now click on the set button to see if that works



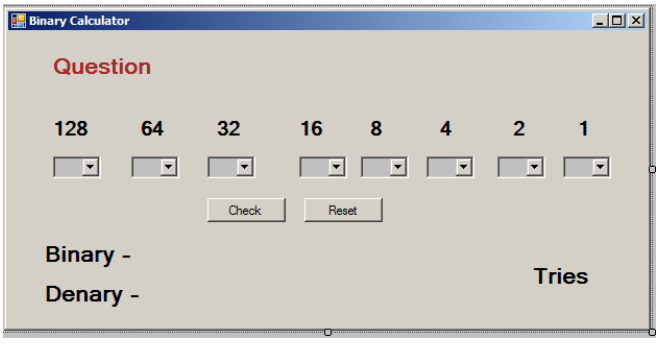
Yes it did. Well done you :)



Now we have a little more work to do and then we are done.

As you probably seen for yourself everything is working in the app except for the question, the user doesn't know what to answer thus far. So we need to fix that.

Lets go back to the design view of the app and double click on the form. Yes i mean the FORM not any component on it.



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

Notice I don't have any components selected only the form. So once I double click on this visual studio will add the event codes same as it did for both buttons.

```
Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load

End Sub
```

Lets add our own code for this Form1_Load function. This function will run only once when the Form is loaded to screen for the first time. We want to ask the question right away.

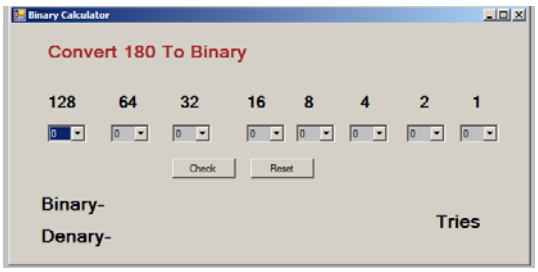
```
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load

        resetCombo() 'run the reset function to set all the combo boxes ro 0
        randomNum = Rnd() * 255 'generate a random number between 0 and 255
        question.Text = "Convert " + randomNum.ToString() + " To Binary" 'show the question
on screen.

    End Sub
```

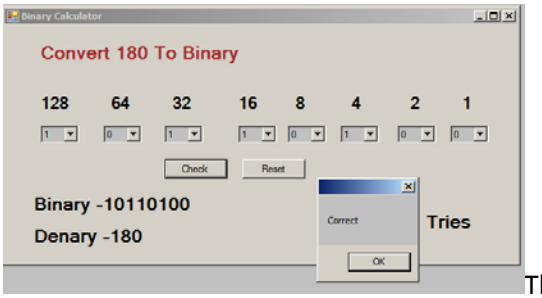
Here we are running the reset combo function again so its all selected to 0.

We are generating a random number between 0 and 255, its the range of our binary calculation

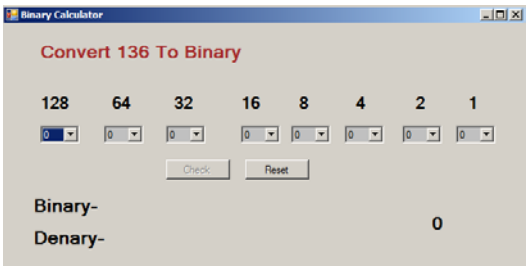
lastly we will ask the question to the user on the question label by inserting the random number variable on the label and showing it on the screen.

Lets run the program now.

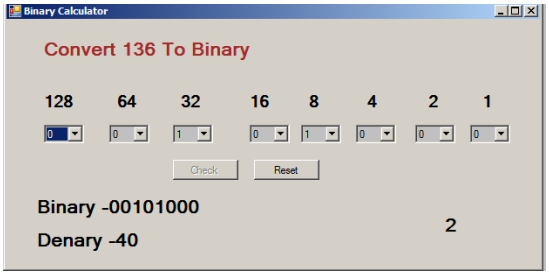




That worked. Check out that Binary as well.



And now the question has changed too. WOW amazing



Now the attempt is working too.

For the full code check out Page 2 on www.mooict.com tutorial page.

Well done for following thus far. Hope you had Fun.

Moo On ..