C# Breakout Game Development Tutorial

This is a tutorial for the beginner programmers. There are many different way to create this classic game, in this tutorial we have aimed it for the beginner programmer to get a result for their efforts.

Open Visual Studio, Create a new project in C# Windows Form application and name it breakoutgame.

Click on the form and change the following in the properties window

Change the Text to Break Out Game and change the size 924 and 706. 924 is the width of the form and 706 is the height of the form.

Find the BackColor option and change it to black for the form.

We will add a picture box to the form

More tutorials on www.mooict.com
Change the size to **125** and **23** also change the TAG section to block

![Image of tag block]

Change the **BackColor** to white so we can see it.

Now we are going to copy this picture box 4 times to side

![Image of copy picture box]

Now drag across and select all of them and copy and paste to the bottom of it 6 times till you get the same outlook as below.

![Image of final outlook]

More tutorials on [www.mooict.com](http://www.mooict.com)
Now we have total of 35 picture boxes tagged as block in the form. It’s important to remember this number.

All of these picture boxes have the same height, width and tag name which will be used in the code to identify them.

Now add a button to the form

| Change the following in the properties for the button  |
|---------------------------------|----------------|
| (Name) = player                 | BackColor = White |
| Enabled = false                 | Location = 33, 657 |
| Size = 175, 23                  | Text = Player     |

Now let’s add another picture box to form. This picture box will used as the ball.

Once you added the picture box to the form change the properties to the following

(Name) = ball

BackColor = Yellow

Size = 23, 26

Lastly add a label to screen this will keep the score for us.

| Change the following in the properties. Place this label on top left corner of the screen. |
|---------------------------------|----------------|

More tutorials on [www.mooict.com](http://www.mooict.com)
Lastly we need to add our main component the **TIMER**

![Timer component](image)

Drag the timer to form

![Timer properties](image)

Change the following:

**Enabled** = true

**Interval** = 20

Now let's get started with the CODE yeaayy

First right click on the form and click on View Code

More tutorials on [www.mooict.com](http://www.mooict.com)
```csharp
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 breakoutGame
{
    public partial class Form1 : Form
    {
        bool goRight;
        bool goLeft;
        int speed = 10;
        int ballx = 5;
        int bally = 5;
        int score = 0;
        private Random rnd = new Random();

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

This is the basic code which is the template for every Windows application.
Look carefully at how the code is laid out. We made some space on top of the form1() line and entered these variables. All of these variables are declared outside any function so they are classed as global variables.

**bool goRight** is a Boolean which is set to false by default. We will set it to true when the player press the right button.

**bool goLeft** is a Boolean which is set to false as before, this will also be set to True when player presses the left button. Both of these Booleans will turn back to false when player releases the said buttons left or right.

**int ballx** is a integer with the value of 5.

**int bally** is a integer with the value of 5. Both of these integers are used to move the ball in the X or Y axis in the windows form.

**int score** is the integer which will hold out total score for the game. This integer will add 1 to itself when we hit a block and take it off the screen.

```csharp
private Random rnd = new Random();
```

This line above is creating a new instance of the CLASS, we have created a *rnd* variable which will be used to create a random number. We will be using this RND variable to select a random colour for the blocks. See below

```csharp
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 breakoutGame
{
    public partial class Form1 : Form
    {
        bool goRight;
        bool goLeft;
        int speed = 10;

        int ballx = 5;
        int bally = 5;

        int score = 0;

        public Form1()
        {
            InitializeComponent();

            foreach (Control x in this.Controls)
            {
                if (x is PictureBox && x.Tag == "block")
                {
                    Color randomColor = Color.FromArgb(rnd.Next(256), rnd.Next(256), rnd.Next(256));
                    x.BackColor = randomColor;
                }
            }
        }
    }
}
```
Inside the Form1 function we are going to run a for each loop which will check the components used in this application. We are able to do various things with for each loop in this case we are looking for picture boxes with the tag of BLOCK.

Start from the top

```csharp
foreach (Control x in this.Controls)
```

this is the first declaration of the for each loop. We are giving it a variable called X with the type of control. This will help us identify any component of windows form.

We always start with the open curly bracket after the condition of the loop {

```csharp
if (x is PictureBox && x.Tag == "block")
```

Above we are starting the IF statement to find the picture box for instance we are saying if x is a type of picture AND x has the tag of block then do the following. Once again we start the IF statement with the open curly bracket {

```csharp
Color randomColor = Color.FromArgb(rnd.Next(256), rnd.Next(256), rnd.Next(256));
```

we are creating a local variable called randomColor and then we are using the colour properties to randomly choose from a range of RGB (red, green and blue) elements with our RND variable.

```csharp
x.BackColor = randomColor;
```

This line assigning a single colour to each x -> picture box found on the form.

Let's not forget to end the IF statement curly brackets }

and the for loop curly brackets }

Run the game now.

Looking good.

Let go back to for the form and add the key down and key up events

After you have clicked on the Form look at the properties window. Click on the little lightning bolt icon and you will see the following options

More tutorials on [www.mooict.com](http://www.mooict.com)
Here you add various different events to the form to create your up application.

The two events we want are key down and key up.

Let's find the key down first.

Type `keydown` (no spaces) and press enter.

you will go back to the code view. Come back to the form view and now find the key up.

type `keysup` and press enter.

Let's find the key is down function and enter the code inside of it.

```csharp
private void keyisdown(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)
    {
        goLeft = true;
    }
}
```

More tutorials on [www.mooict.com](http://www.mooict.com)
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 < 920)
    {
        goRight = true;
    }
}

if (e.KeyCode == Keys.Left && player.Left > 0)
{
    goLeft = true;
}

The code reads that if key code is equals to LEFT AND player button left is greater than 0 then we change the go left to true. This statement stops our player from leaving the form. We always want the player to stay within the playground.

if (e.KeyCode == Keys.Right && player.Left + player.Width < 920)
{
    goRight = true;
}

This if statement is for the right key. The same as the left key but a little different. IF the key code is equals to RIGHT AND player left plus the width which gives us the full width to the right of the button is less than 920 which is the width of our form then we change the go right to true. This way the player cannot leave the form to the right either.

Now find the key is up function and enter the code below

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

In the function above once the player releases the left or right key we change the go left and right back to false. So the player button cannot move when the button isn’t pressed.

Go back to the form design view and double click on the timer

This will automatically create a link for the timer. This is the most important part.

Let's see how it works with simple 2 lines of code to start off.

More tutorials on www.mooict.com
```csharp
private void timer1_Tick(object sender, EventArgs e)
{
    ball.Left += ballx;
    ball.Top += bally;
}
```

Ball.Left += ballx; This line of code will move the ball to the left each time the timer ran. Ball.Top += bally will move the ball downwards each time the timer runs.

If you run the game now, the yellow square will disappear in the bottom of the screen.

This will be a long function all of the codes are explained on the right side of the table.

<table>
<thead>
<tr>
<th>Code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>private void timer1_Tick(object sender, EventArgs e)</td>
<td>This is the top part of the function. This gives the name of the function</td>
</tr>
<tr>
<td></td>
<td>and the object which is bound to it. In this case its linked to the timer</td>
</tr>
<tr>
<td></td>
<td>on the form.</td>
</tr>
<tr>
<td>ball.Left += ballx;</td>
<td>We discussed this earlier it will move the ball left.</td>
</tr>
<tr>
<td>ball.Top += bally;</td>
<td>We discussed this line was discussed earlier too.</td>
</tr>
<tr>
<td>label1.Text = &quot;Score: &quot; + score;</td>
<td>This line is calling the label1 we added to the form and we will dynamically</td>
</tr>
<tr>
<td></td>
<td>change the text as the score increased throughout the same. We have</td>
</tr>
<tr>
<td></td>
<td>inserted the score variable which will increase each time a block is</td>
</tr>
<tr>
<td></td>
<td>taken out.</td>
</tr>
<tr>
<td>if (goLeft) { player.Left -= speed; } // move left</td>
<td>This line is checking if the go left variable is true if so then we allow</td>
</tr>
<tr>
<td></td>
<td>the user to move left by stating -= minus equals to the speed variable.</td>
</tr>
<tr>
<td></td>
<td>Doing minus equals moves it to the left.</td>
</tr>
<tr>
<td>if (goRight) { player.Left += speed; } // move right</td>
<td>This line is checking if the go right variable is equals to true, if so it</td>
</tr>
<tr>
<td></td>
<td>will do a plus equals to speed meaning it will add the speed towards the</td>
</tr>
<tr>
<td></td>
<td>right and move the button to the right.</td>
</tr>
<tr>
<td>if (player.Left &lt; 1)</td>
<td>Although we have mentioned that we don't want the player to leave the</td>
</tr>
<tr>
<td>{</td>
<td>scene, sometimes it will still allow the player to somewhat overlap the</td>
</tr>
<tr>
<td>goLeft = false; // stop the car from going off</td>
<td>form and leave the scene, however we can put some code in the timer to</td>
</tr>
<tr>
<td></td>
<td>stop them from moving out. This line is checking if the players left is</td>
</tr>
<tr>
<td></td>
<td>less than 1 then we change the go left to false.</td>
</tr>
<tr>
<td>else if (player.Left + player.Width &gt; 920)</td>
<td>Same again for this one, we are checking if the player left plus the</td>
</tr>
<tr>
<td>{</td>
<td>player width is greater than 920 meaning the right side of the form then</td>
</tr>
<tr>
<td>goRight = false;</td>
<td>we change it back to false. Thus the player paddle doesn't move out of the</td>
</tr>
<tr>
<td>}</td>
<td>scene.</td>
</tr>
<tr>
<td>if (ball.Left + ball.Width &gt; ClientSize.Width</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td>it can be one OR the other. If ball left plus the ball width is greater</td>
</tr>
<tr>
<td>// this will bounce the object from the left or right side</td>
<td>than the form (Client Size) width then OR (</td>
</tr>
<tr>
<td>}</td>
<td>meaning its on the edge of left side of the</td>
</tr>
</tbody>
</table>

More tutorials on [www.mooict.com](http://www.mooict.com)
Screen then we change the ballx from positive to negative. This will reverse the effect and bounce the ball off the sides.

if (ball.Top < 0 || ball.Bounds.IntersectsWith(player.Bounds))
{
    bally = -bally; // this will bounce the object from top and bottom border
}

if (ball.Top + ball.Height > ClientSize.Height)
{
    gameOver();
}

foreach (Control x in this.Controls)
{
    if (x is PictureBox && x.Tag == "block")
    {
        if (ball.Bounds.IntersectsWith(x.Bounds))
        {
            this.Controls.Remove(x);
            bally = -bally;
            score++;
        }
    }
}

if (score > 34)
{
    gameOver();
    MessageBox.Show("You Win");
}

We need to add the last function the game.

private void gameOver()
{
    timer1.Stop();
}

The game over function will only stop the main timer so the whole game stops.
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 breakoutGame
{
    public partial class Form1 : Form
    {
        bool goRight;
        bool goLeft;
        int speed = 10;
        int ballx = 5;
        int bally = 5;
        int score = 0;
        private Random rnd = new Random();

        public Form1()
        {
            InitializeComponent();
            foreach (Control x in this.Controls)
            {
                if (x is PictureBox && x.Tag == "block")
                {
                    Color randomColor = Color.FromArgb(rnd.Next(256), rnd.Next(256), rnd.Next(256));
                    x.BackColor = randomColor;
                }
            }
        }

        private void keyisdown(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)
            {
                goLeft = 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 < 920)
            {
                goRight = true;
            }
        }

        private void keyisup(object sender, KeyEventArgs e)
        {
            // if the LEFT key is up we set the car left to false
            if (e.KeyCode == Keys.Left)
            {
                goLeft = false;
            }
            // if the RIGHT key is up we set the car right to false
            if (e.KeyCode == Keys.Right)
            {
                goRight = false;
            }
        }
    }
}
private void timer1_Tick(object sender, EventArgs e)
{
    ball.Left += ballx;
    ball.Top += bally;

    label1.Text = "Score: " + score;
    if (goLeft) { player.Left -= speed; } // move left
    if (goRight) { player.Left += speed; } // move right

    if (player.Left < 1)
    {
        goLeft = false; // stop the car from going off screen
    }
    else if (player.Left + player.Width > 920)
    {
        goRight = false;
    }
    if (ball.Left + ball.Width > ClientSize.Width || ball.Left < 0)
    {
        ballx = -ballx; // this will bounce the object from the left or right border
    }
    if (ball.Top < 0 || ball.Bounds.Intersects(player.Bounds))
    {
        bally = -bally; // this will bounce the object from top and bottom border
    }
    if (ball.Top + ball.Height > ClientSize.Height)
    {
        gameOver();
    }
    foreach (Control x in this.Controls)
    {
        if (x is PictureBox && x.Tag == "block")
        {
            if (ball.Bounds.IntersectsWith(x.Bounds))
            {
                this.Controls.Remove(x);
                bally = -bally;
                score++;
            }
        }
    }
}

if (score > 34)
{
    gameOver();
    MessageBox.Show("You Win");
}

private void gameOver()
{
    timer1.Stop();
}