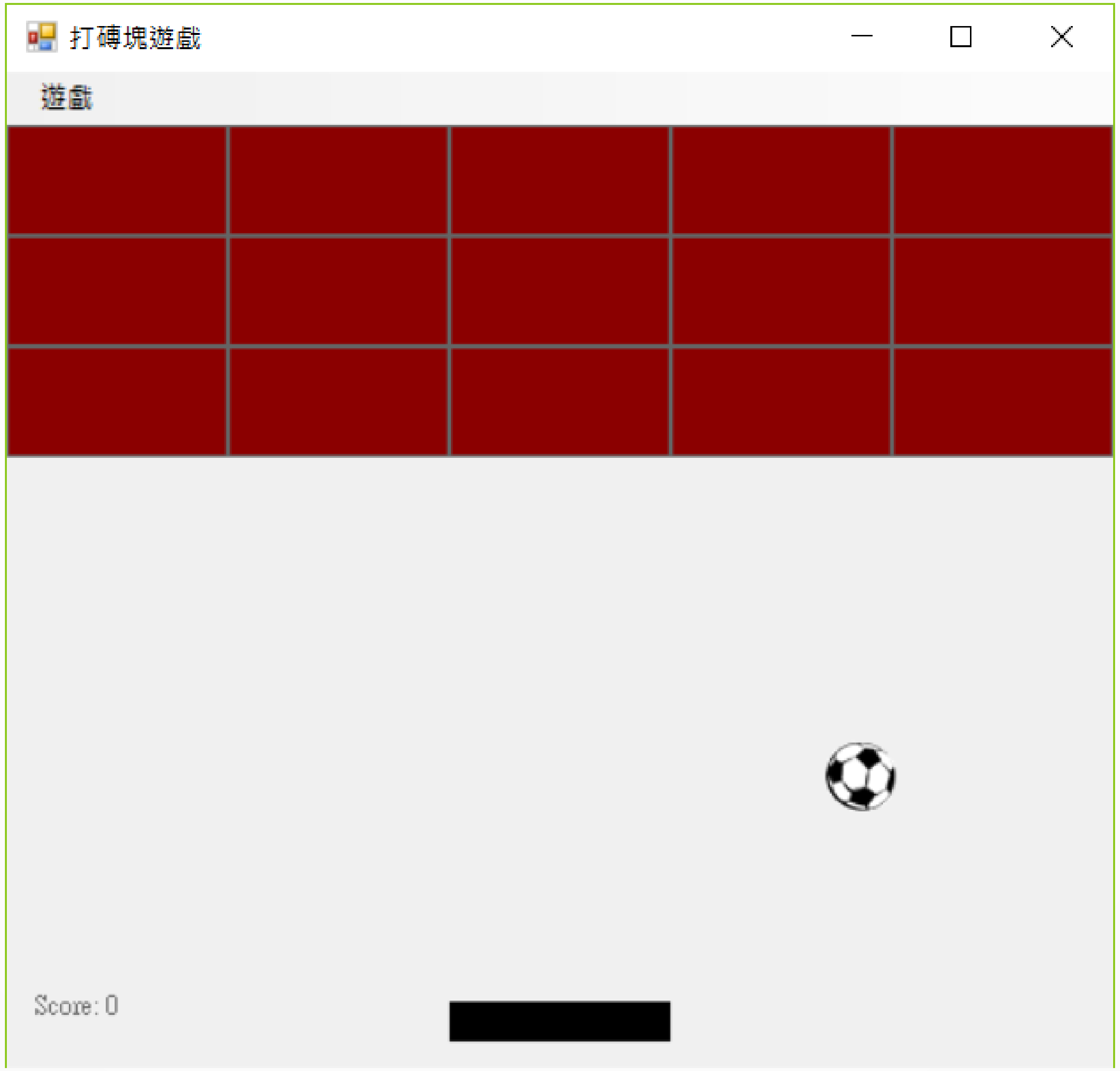
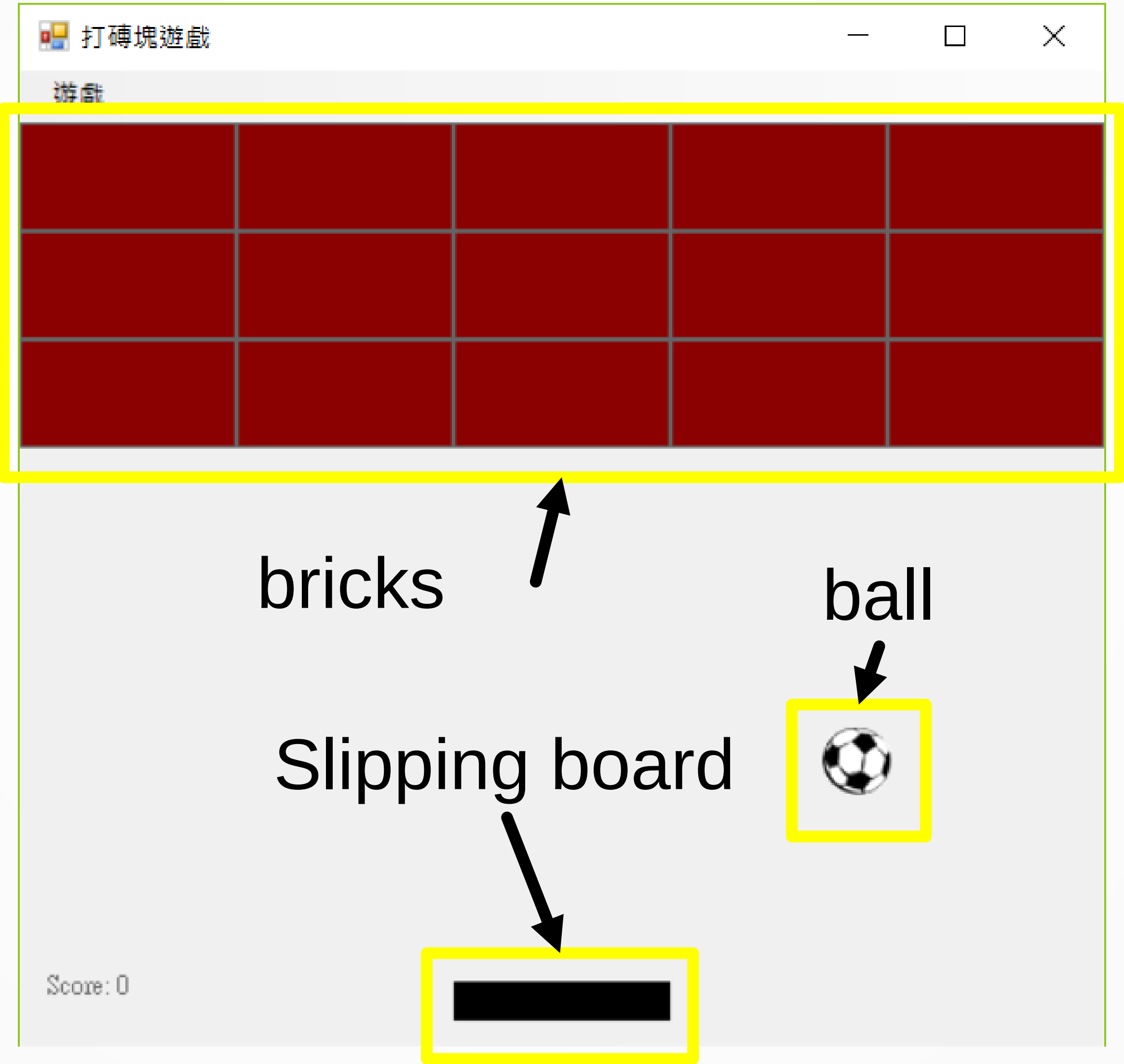
Brick Buster

Make an easy C# window game

Brick Buster game



Brick Buster



Score: 0

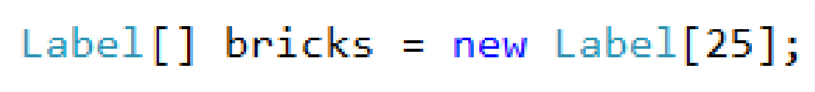
bricks

ball

Slipping board

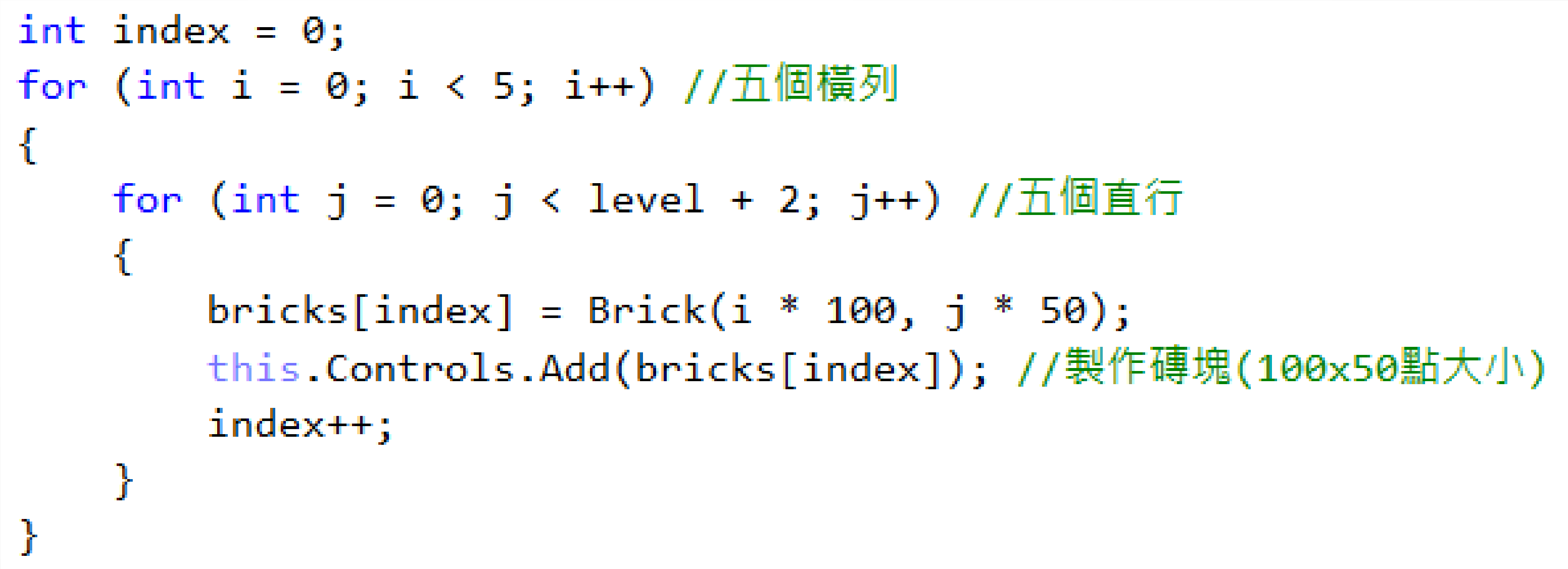
Bricks

- Use Label



```
Label[] bricks = new Label[25];
```

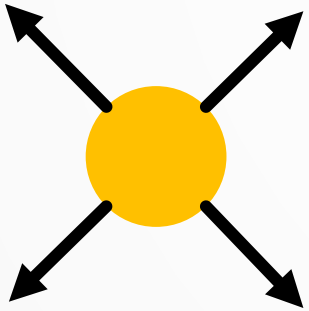
- Use the program to do dynamical generation



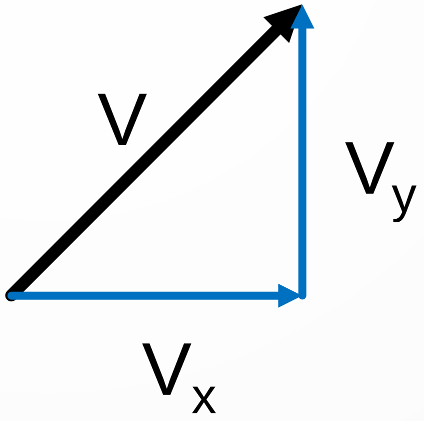
```
int index = 0;
for (int i = 0; i < 5; i++) //五個橫列
{
    for (int j = 0; j < level + 2; j++) //五個直行
    {
        bricks[index] = Brick(i * 100, j * 50);
        this.Controls.Add(bricks[index]); //製作磚塊(100x50點大小)
        index++;
    }
}
```

Ball

- Use PictureBox
- Move with the same speed on screen
- Bounce between items



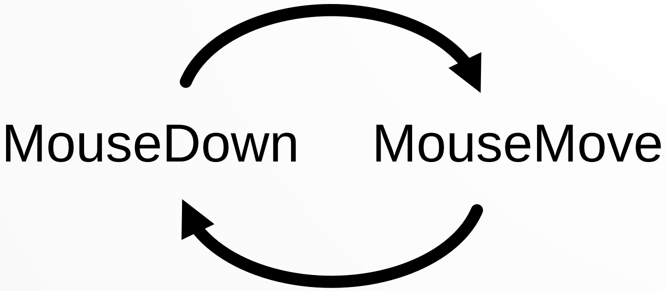
Four ways

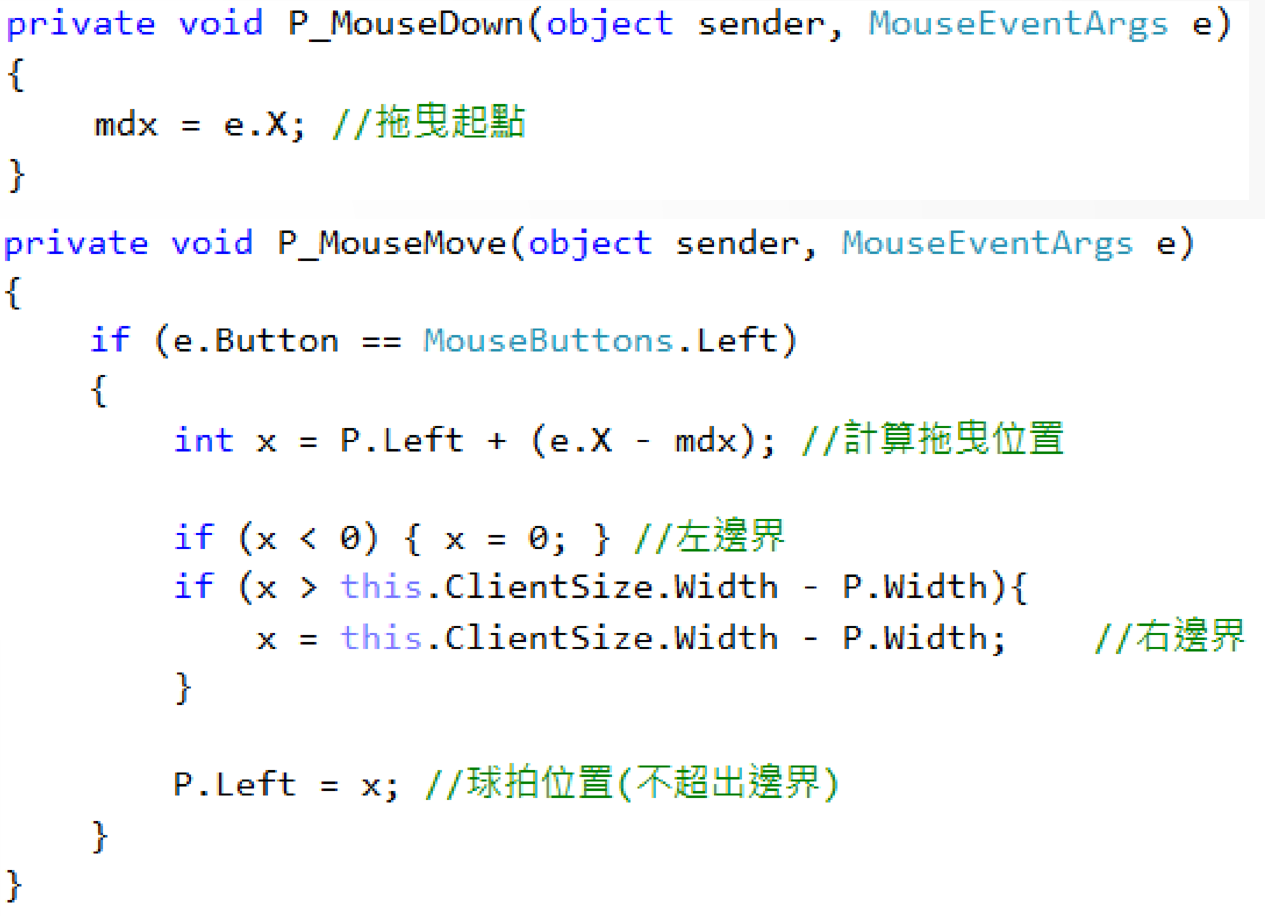


$V = V_x + V_y$

Board

- Use PictureBox
- Use the mouse to control
- Can move right and left





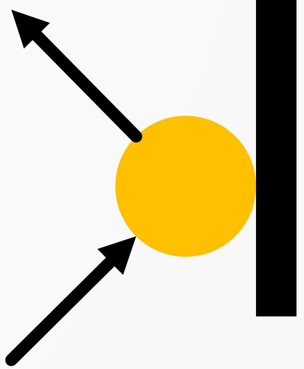
```
private void P_MouseDown(object sender, MouseEventArgs e)
{
    mdx = e.X; //拖曳起點
}
private void P_MouseMove(object sender, MouseEventArgs e)
{
    if (e.Button == MouseButtons.Left)
    {
        int x = P.Left + (e.X - mdx); //計算拖曳位置

        if (x < 0) { x = 0; } //左邊界
        if (x > this.ClientSize.Width - P.Width){
            x = this.ClientSize.Width - P.Width;    //右邊界
        }

        P.Left = x; //球拍位置(不超出邊界)
    }
}
```

Checking the bounces

- Between the ball and the wall
- Between the ball and the board
- Between the ball and the bricks

Bounces between the ball and the wall

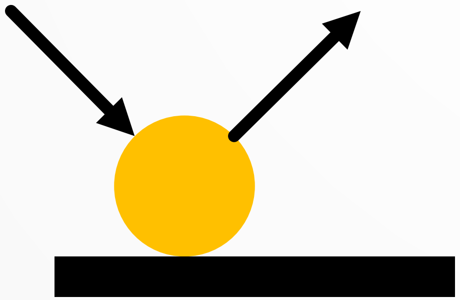
- Check whether the ball hits the boarder; if so, change the moving direction:
  - ball.Left < 0 ➔bounce to the right ➔ Vx[ball] = abs(Vx[ball])
  - ball.Right > this.CilentSize.Width ➔ bounce to the left ➔
    Vx[ball] = -abs(Vx[ball])
  - ball.Top < MenuStrip.Height ➔bounce down➔ Vy[ball] = abs(Vy[ball])
- Judge whether the ball exceed the bottom boarder
  - ball.Top > this.ClientSize.Height ➔ Game Over

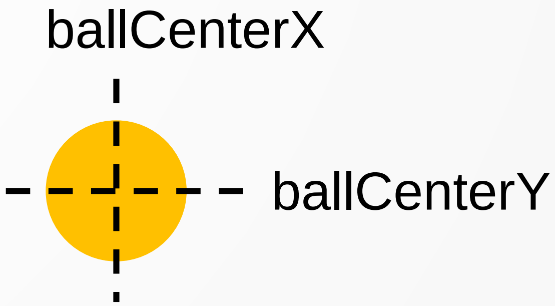


Bounces between the ball and the board

- Check whether the ball is lower than the board, and check whether it is in the range of the board. if so, make the ball bounce up.
  - ball.Bottom > board.Top && ballCenterY <= board.Top
  - ballCenterX >= board.Left && ballCenterX <= board.Right

➔bounce up➔ Vy[ball] = -abs(Vy[ball])



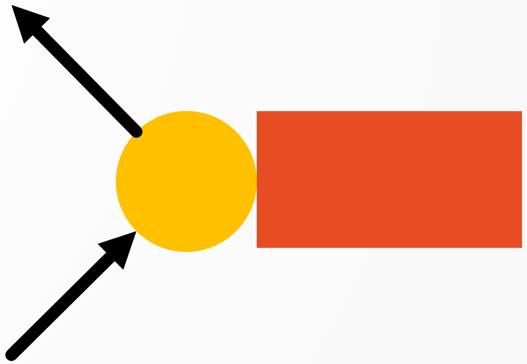


Bounces between the ball and the bricks

- Similar to the bounces with the wall. Check whether the boarder of the ball overlaps with the bricks.
  - ball.Right >= brick.Left && (ball.Right - brick.Left) <= Abs(Vx[v])

    ➔the moment when the left of the bricks collapse with the ball

    ➔ the ball bounces left
  - Works the same for the other three directions
- After the bounces, clear the bricks
  - brick.Dispose()



Simulate the movement of the objects

- Use timer. With each tick occurred:
  - Move the ball (change location)
  - Check whether the ball hit the wall
  - Check whether the ball hit the board
  - Check whether the ball hit the bricks
- Repeat without suspension to form an animation

Done!

Any Question?