打磚塊

許弘駿

遊戲設計

- 1. 角色設計
  - 球
  - 板子
  - 磚塊
- 進行方式
  - 球彈跳, 撞到磚塊, 磚塊消失
  - 用板子反彈球

角色設計

- 球(ball)
  - 畫出來
  - 移動 (並且判斷邊界)
- 板子(rocket)
  - 畫出來
  - 移動 (判斷邊界)
  - 判斷是否碰撞球
- 磚塊
  - 畫出來 (若還沒被打到)
  - 判斷是否碰撞球, 被碰到, 則設成被打到了.

classball

```
class classball
  {
      public Point position; // 球的位置
      public Point velocity; // 球的速度
      public Size clientSize; // 視窗寬高
      public int Ball_Width = 3; // 球的半徑
      SolidBrush myBrush = new SolidBrush(Color.Blue);

      public classball(Point position, Point velocity, Size clientSize, int Ball_Width, Color color)
      {
          this.position = position; // 球的位置
          this.velocity = velocity; // 球的速度
          this.clientSize = clientSize; // 視窗寬高
          this.Ball_Width = Ball_Width;
          myBrush.Color = color;
      }

      public void Draw(Graphics G)
      {
          G.FillEllipse(myBrush, position.X - Ball_Width, position.Y - Ball_Width, Ball_Width * 2, Ball_Width * 2);
          G.DrawEllipse(Pens.Black, position.X - Ball_Width, position.Y - Ball_Width, Ball_Width * 2, Ball_Width * 2);
      }
```

classball

```
public void Move()
      {
         position.X += velocity.X;
         position.Y += velocity.Y;
         // 右邊界
         if (position.X > clientSize.Width - Ball_Width)
         {
            velocity.X = -velocity.X;
            position.X = clientSize.Width - Ball_Width;
         }
         // 下邊界
         else if (position.Y > clientSize.Height - Ball_Width)
         {
            velocity.Y = -velocity.Y;
            position.Y = clientSize.Height - Ball_Width;
         }
         // 左邊界
         else if (position.X < Ball_Width)
         {
            velocity.X = -velocity.X;
            position.X = Ball_Width;
         }
         // 上邊界
         else if (position.Y < Ball_Width)
         {
            velocity.Y = -velocity.Y;
            position.Y = Ball_Width;
         }
      }
```

Rocket (板子)

```
class Rocket
  {
      public Point position; // 板子的位置
      public Size clientSize; // 視窗寬高
      public Size rocketSize; // 板子的大小
      SolidBrush myBrush = new SolidBrush(Color.Blue);

      public Rocket(Point position, Size rocketSize, Size ClientSize, SolidBrush mBrush)
      {
          this.position = position;
          this.rocketSize = rocketSize;
          this.clientSize = ClientSize;
          this.myBrush = mBrush;
      }
      public void Draw(Graphics G)
      {
          G.FillRectangle(myBrush, position.X, position.Y, rocketSize.Width, rocketSize.Height);
          G.DrawRectangle(Pens.Black, position.X, position.Y, rocketSize.Width, rocketSize.Height);
      }
```

Rocket (板子)

```
public void move(int X)
      {
         position.X = X;
         if (position.X < 0)
         {
            position.X = 0;
         }
         else if (position.X > clientSize.Width - rocketSize.Width)
         {
            position.X = clientSize.Width - rocketSize.Width;
         }
      }
public bool Collides(classball Ball)
      {
         if (position.X + rocketSize.Width > Ball.position.X - Ball.Ball_Width &&
            position.X - rocketSize.Width < Ball.position.X + Ball.Ball_Width &&
            position.Y + rocketSize.Height > Ball.position.Y - Ball.Ball_Width &&
            position.Y - rocketSize.Height < Ball.position.Y + Ball.Ball_Width)
            return true;
         else
            return false;
      }
```

Stone (磚塊)

```
class Stone
 {
     public Point position; // 磚塊的位置
     public Size stoneSize; // 磚塊的大小
     public bool isVisible = true;
     SolidBrush myBrush = new SolidBrush(Color.Blue);

     public Stone(Point position, Size stoneSize, SolidBrush mBrush)
     {
         this.position = position;
         this.stoneSize = stoneSize;
         this.myBrush = mBrush;
     }

     public void Draw(Graphics G)
     {
         if (isVisible)
         {
             G.FillRectangle(myBrush, position.X, position.Y, stoneSize.Width, stoneSize.Height);
             G.DrawRectangle(Pens.Black, position.X, position.Y, stoneSize.Width, stoneSize.Height);
         }
     }
```

Stone (磚塊)

```
public bool Collides(classball Ball)
    {
        if (position.X + stoneSize.Width > Ball.position.X - Ball.Ball_Width &&
            position.X - stoneSize.Width < Ball.position.X + Ball.Ball_Width &&
            position.Y + stoneSize.Height > Ball.position.Y - Ball.Ball_Width &&
            position.Y - stoneSize.Height < Ball.position.Y + Ball.Ball_Width)
            return true;
        else
            return false;
    }
```

Form1

```
classball ball;
       Rocket rocket;
       List<Stone> stone=new List<Stone>();
       public Form1()
       {
           InitializeComponent();
           Point pt = new Point(300, 480);
           Point velocity = new Point(5, -5);
           Size clientSize = this.ClientSize;
           ball = new classball(pt, velocity, clientSize, 5, Color.AliceBlue);
           pt.X=280;
           pt.Y=500;
           Size rsize=new Size(45,8);
           SolidBrush mBrush = new SolidBrush(Color.Cyan);
           rocket = new Rocket(pt, rsize, ClientSize, mBrush);

           SolidBrush sBrush = new SolidBrush(Color.DarkGray);
           Size ssize = new Size(32, 10);
           for (int i = 0; i < 15; i++)
           {
               pt.X=60+i*33;
               pt.Y=100;
               Stone s = new Stone(pt, ssize, sBrush);
               stone.Add(s);
           }
       }
```

Timer1

```
private void timer1_Tick(object sender, EventArgs e)
        {
            ball.Move();
            if (rocket.Collides(ball))
            {
                ball.velocity.Y *= -1;
            }
            else
            {
                for (int i = 0; i < stone.Count; i++)
                {
                    if (stone[i].isVisible && stone[i].Collides(ball))
                    {
                        ball.velocity.Y *= -1;
                        stone[i].isVisible = false;
                        break;
                    }
                }
            }
            this.Invalidate();
        }
```

滑鼠移動

```
private void Form1_MouseMove(object sender, MouseEventArgs e)
        {
            rocket.move(e.X);
            this.Invalidate();
        }
```

畫圖

```
private void Form1_Paint(object sender, PaintEventArgs e)
    {
        rocket.Draw(e.Graphics);
        ball.Draw(e.Graphics);
        for (int i = 0; i < stone.Count; i++)
        {
            if (stone[i].isVisible) stone[i].Draw(e.Graphics);
        }
    }
```

畫面