**Thông tin đồ án game Dò Mìn (Minesweeper)**

**Luật chơi**

Minisweeper (của Windows) có phần giao diện chính là một bảng các ô vuông xếp liền nhau tạo thành một hình chữ nhật có chiều rộng và dài tối thiểu là 9 ô (đơn vị là ô vuông) và số mìn tối thiểu là 10.

Trong bảng này sẽ có các ô được đặt mìn ngẫu nhiên và nhiệm vụ của người chơi là mở tất cả các ô không có mìn bằng cách click chuột trái vào các ô đó, khi chỉ còn các ô có mìn còn lại thì kết thúc màn chơi.

**Các trạng thái của một ô**

Tùy theo quá trình khởi tạo và thao tác của người dùng mà các ô trong bảng có thể có một hoặc vài trạng thái trong các trạng thái sau:

* Có mìn: được đặt ngẫu nhiên lúc khởi tạo
* Đã mở: Khi người dùng nhấn chuột trái vào ô
* Được cắm cờ: Khi người dùng nhấn phải vào ô
* Được đánh dấu: Khi nhấn phải vào ô đã được “cắm cờ”
* Bình thường: không có tất cả các trạng thái trên

**Các trường hợp khi mở một ô**

Khi mở một ô **X** nào đó, có 3 trường hợp có thể xảy ra:

1. X có mìn: hiện tất cả mìn trong bảng ra và ‘game over’.
2. X không có mìn nhưng 8 ô xung quanh có mìn: hiện số mìn xung quanh vào X.
3. X không có mìn và xung quanh cũng không có mìn: mở lần lượt các ô xung quanh X cho đến khi gặp các trường hợp 1 và 2.

**Cắm cờ và đánh dấu**

Minisweeper cho phép bạn đánh dấu các ô nghi ngờ có mìn bằng cách “cắm cờ” và “đánh dấu”. Khi bạn “cắm cờ”, tức là bạn xác định rằng ô đó có mìn và ô đó được hiển thị là một lá cờ. Bạn không thể mở ô đó bằng chuột trái được. Bạn chỉ có thể hủy bỏ trạng thái “cắm cờ” bằng cách click chuột phải, tùy theo thiết lập mà ô đó sẽ chuyển sang trại thái “đánh dấu” hoặc “bình thường”.

Khi bạn “đánh dấu”, tức là bạn đoán rằng ô đó có thể có mìn nhưng không chắc chắn.

# ****Xây dựng chương trình****

Lớp **Cell** dựa trên các trạng thái của một ô. Lớp này chỉ bao gồm các field sau:

public class Cell

{

    public bool IsMine = false;

    public bool IsOpened = false;

    public bool IsFlag = false;

    public bool IsMarked = false;

    public int  MinesAround = 0;

}

Lớp **MinesBoard** sẽ chứa một mảng hai chiều các đối tượng kiểu **Cell**, ngoài ra còn chứa dữ liệu cần thiết khác như số dòng, số cột, số mìn, số lá cờ được cắm, số ô đã được mở và hai trạng thái thắng, thua.

Việc khởi tạo MinesBoard sẽ lặp vào random đánh dấu trạng thái **IsMine** của các Cell là true:

private void InitBoard()

{

    // […]

    while (count < \_MinesCount)

    {

        int index = rnd.Next(\_CellsCount);

        int r = index / \_Cols;

        int c = index % \_Cols;

        if (!\_cells[r, c].IsMine)

        {

            \_cells[r, c].IsMine = true;

            count++;

        }

    }

}

## **Phần Presentation**

Hiển thị và xử lý thao tác của người dùng.

protected override void OnPaint(PaintEventArgs e)

{

    e.Graphics.FillRectangle(Brushes.LightGray, 0, 0, this.Width, this.Height);

    for (int i = 0; i < \_board.\_Rows; i++)

    {

        int y = CELL\_SIZE \* i;

        for (int j = 0; j < \_board.\_Cols; j++)

        {

            int x = CELL\_SIZE \* j;

            if (\_board[i, j].IsOpened)

            {

                if (\_board[i, j].IsMine)

                {

                    e.Graphics.FillRectangle(Brushes.Red, x, y, CELL\_SIZE, CELL\_SIZE);

                    e.Graphics.DrawImage(\_imgBomb, x, y);

                }

                else if (\_board[i, j].MinesAround > 0)

                {

                    string s = \_board[i, j].MinesAround.ToString();

                    SizeF size = e.Graphics.MeasureString(s, this.Font);

                    e.Graphics.DrawString(s,

                        this.Font, new SolidBrush(\_foreColors[\_board[i, j].MinesAround - 1]),

                            x + (CELL\_SIZE - size.Width) / 2, y + (CELL\_SIZE - size.Height) / 2);

                }

            }

            else

                e.Graphics.DrawImage(\_imgCell, x, y);

            if (\_board.\_IsLost)

            {

                if (\_board[i, j].IsMine)

                    e.Graphics.DrawImage(\_imgBomb, x, y);

            }

            if (\_board[i, j].IsFlag)

            {

                e.Graphics.DrawImage(\_imgFlag, x, y);

            }

            else if (\_board[i, j].IsMarked)

            {

                string s = "?";

                SizeF size = e.Graphics.MeasureString(s, this.Font);

                e.Graphics.DrawString(s,

                    this.Font, Brushes.Black,

                        x + (CELL\_SIZE - size.Width) / 2, y + (CELL\_SIZE - size.Height) / 2);

            }

            // vertical

            if(i==0)

            e.Graphics.DrawLine(Pens.Gray, x, 0, x, this.Height);

        }

        // hoz

        e.Graphics.DrawLine(Pens.Gray, 0, y, this.Width, y);

    }

    base.OnPaint(e);

}

**Xử lý thao tác người dùng**

protected override void OnMouseDown(MouseEventArgs e)

{

    if (!\_board.\_IsLost && !\_board.\_IsFinish)

    {

        int c = e.X / CELL\_SIZE;

        int r = e.Y / CELL\_SIZE;

        if (\_board[r, c].IsOpened)

            return;

        if (!\_board[r, c].IsFlag && e.Button == MouseButtons.Left)

        {

            \_board[r, c].IsMarked = false;

            if (\_board.OpenCell(r, c))

            {

                \_board.\_IsLost = true;

                pictureBox1.Left = e.X - pictureBox1.Width / 2;

                pictureBox1.Top = e.Y - pictureBox1.Height / 2;

                pictureBox1.Visible = true;

                timer1.Enabled = true;

                // Dùng cho event

                OnMinesExplode();

            }

            else

            {

                // Win

                if (RemainCellsCount == MinesCount)

                {

                    \_board.\_IsFinish = true;

                    // Cắm cờ tất cả ô còn lại

                    for (int i = 0; i < Rows; i++)

                    {

                        for (int j = 0; j < Cols; j++)

                        {

                            if (!\_board[i, j].IsOpened)

                                \_board[i, j].IsFlag = true;

                        }

                    }

                }

            }

            Invalidate();

        }

        else if (e.Button == MouseButtons.Right)

        {

            if (\_board[r, c].IsMarked)

            {

                \_board[r, c].IsMarked = false;

            }

            else

            {

                \_board[r, c].IsFlag = !\_board[r, c].IsFlag;

                \_board[r, c].IsMarked = !\_board[r, c].IsFlag;

                if (\_board[r, c].IsFlag)

                    \_board.\_FlagsCount++;

                else

                    \_board.\_FlagsCount--;

            }

            Invalidate();

        }

        // Dùng cho event

        OnCellClick();

    }

    base.OnMouseDown(e);

}

**Thêm các Event**

public event EventHandler CellClick;

public event EventHandler MinesExplode;

#region CustomEvent

private void OnCellClick()

{

    if (CellClick != null)

        CellClick(this, null);

}

protected void OnMinesExplode()

{

    if (MinesExplode != null)

        MinesExplode(this, null);

}

#endregion