Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота № 2

з дисципліни «Технології та засоби розробки комп'ютерної графіки та мультимедіа»

Тема: «__Переміщення зображення по довільній траекторії. Переміщення зображення за допомогою клавіатури. Переміщення зображення за допомогою маніпулятора миші__»

Виконав:

студент групи ІО-25

Льоскін Іван Вадимович

Перевірила:

ст. вик. кафедри ІСТ

Хмелюк Марина Сергіївна

Дата здачі __20.12.2024__

Захищено з балом _____

Київ 2024

Завдання:

- намалювати або підключити об'єкт (зображення)
- змінити координати зображення і вивести його на нове місце і так далі
- відновити екран в місці, де було зображення
- реалізувати безперервну траєкторію руху зображення
- розташувати на екрані об'єкти-перешкоди, і забезпечити відбиванняя від них зображження, яке рухається
- додати зображення та забезпечити управління переміщенням і перемиканням між ними за допомогою клавіатури
- додати зображення та забезпечити управління переміщенням і перемиканням між зображеннями за допомогою маніпулятора миші

Хід роботи:

Аналогічно попереднім лабораторним роботам створюємо новий проект. Також створюємо нову іконку, встановлюємо її, додаємо задній фон формі, змінюємо ім’я форми та її текст. Також додаємо з Toolbox компонент PictoreBox, загружаємо у папку Resources довільне зображення, та у властивостях PictoreBox, вказуємо полю SizeMode значення Zoom, щоб було видно всю картинку і вона не була обрізаною. Також додаємо ще один PictoreBox ти пристуваємо до роботи. Для цього перейдемо в файл UserInteractionForm.cs:

```
using System;
using System.Drawing;
using System.Windows.Forms;

namespace WindowsFormsApp4
{
    public partial class UserInteractionForm : Form
    {
        public UserInteractionForm()
        {
            InitializeComponent();
            this.ClientSize = new Size(800, 600);
            // create obstacle
            obstacle1 = new Panel
            {
                BackColor = Color.Red,
                Size = new Size(100, 100),
                Location = new Point(300, 180)
            };
            // add obstacle on the form
            this.Controls.Add(obstacle1);
            initialImagePosition = new Point(10, 10);
```

```
            this.KeyDown += new KeyEventHandler(Form1_KeyDown);

            activePictureBox = pictureBox1;
            // subscribe on events
            pictureBox1.MouseDown += new
MouseEventHandler(PictureBox_MouseDown);
            pictureBox1.MouseMove += new
MouseEventHandler(PictureBox_MouseMove);
            pictureBox1.MouseUp += new MouseEventHandler(PictureBox_MouseUp);

            pictureBox2.MouseDown += new
MouseEventHandler(PictureBox_MouseDown);
            pictureBox2.MouseMove += new
MouseEventHandler(PictureBox_MouseMove);
            pictureBox2.MouseUp += new MouseEventHandler(PictureBox_MouseUp);

        }

        private Panel obstacle1;
        private Point initialImagePosition;
        Timer moveTimer = new Timer();
        int xSpeed = 20;
        int ySpeed = 20;
        private PictureBox activePictureBox; // current PictureBox
        private bool isDragging = false; // flag for dragging
        private bool isMooving = true;
        private Point mouseOffset; // cursor shift related of PictureBox

        private void Form1_KeyDown(object sender, KeyEventArgs e)
        {
            if (e.KeyCode == Keys.Up)
            {
                if (activePictureBox.Location.Y > 20)
                {
                    activePictureBox.Location = new
Point(activePictureBox.Location.X, activePictureBox.Location.Y - 20);
                }
            }
            else if (e.KeyCode == Keys.Down)
            {
                if (activePictureBox.Bottom < ClientSize.Height - 20)
                {
                    activePictureBox.Location = new
Point(activePictureBox.Location.X, activePictureBox.Location.Y + 20);
                }
            }
            else if (e.KeyCode == Keys.Left)
            {
                if (activePictureBox.Left > 20)
                {
                    activePictureBox.Location = new
Point(activePictureBox.Location.X - 20, activePictureBox.Location.Y);
                }
```

```
            }
            else if (e.KeyCode == Keys.Right)
            {
                if (activePictureBox.Right < ClientSize.Width - 20)
                {
                    activePictureBox.Location = new
Point(activePictureBox.Location.X + 20, activePictureBox.Location.Y);
                }
            }
            else if (e.KeyCode == Keys.Tab)
            {
                activePictureBox = activePictureBox == pictureBox1 ?
pictureBox2 : pictureBox1;
            }
            else if (e.KeyCode == Keys.R) {
                moveTimer.Stop();
                activePictureBox.Location = initialImagePosition;
                xSpeed = Math.Abs(xSpeed);
                ySpeed = Math.Abs(ySpeed);
            }
            else if (e.KeyCode == Keys.Space)
            {
                Console.WriteLine(isMooving);
                if (isMooving)
                {
                    moveTimer.Stop();
                    isMooving = false;
                }
                else
                {
                    moveTimer.Start();
                    isMooving = true;
                }
            }
            Invalidate(); // render
        }

        private bool IsColliding(Control pictureBox, Panel obstacle)
        {
            Rectangle pictureBoxRect = pictureBox.Bounds;
            Rectangle obstacleRect = obstacle.Bounds;

            return pictureBoxRect.IntersectsWith(obstacleRect);
        }

        private void PictureBox_MouseDown(object sender, MouseEventArgs e)
        {
            // set active PictureBox
            activePictureBox = sender as PictureBox;

            if (activePictureBox != null && e.Button == MouseButtons.Left)
            {
                // run the procces of drugging
```

```
                isDragging = true;
                isMooving = false;
                moveTimer.Stop();

                // calculste cursor shift related to PictureBox
                mouseOffset = new Point(e.X, e.Y);
            }
        }

        private void PictureBox_MouseMove(object sender, MouseEventArgs e)
        {
            if (isDragging && activePictureBox != null)
            {
                // calculate new position of PictureBox
                Point newLocation = activePictureBox.Location;
                newLocation.X += e.X - mouseOffset.X;
                newLocation.Y += e.Y - mouseOffset.Y;
                activePictureBox.Location = newLocation;
            }
        }

        private void PictureBox_MouseUp(object sender, MouseEventArgs e)
        {
            // end dragging
            if (e.Button == MouseButtons.Left)
            {
                isDragging = false;
            }
        }

        private void Infinity_Moove(object sender, EventArgs e)
        {
            isMooving = true;
            moveTimer.Tick += (s, eventArgs) =>
            {

                if (IsColliding(activePictureBox, obstacle1))
                {
                    if (activePictureBox.Location.X < obstacle1.Location.X)
                    {
                        xSpeed = -Math.Abs(xSpeed);  // change vector of
movement if met right side of obstacle
                    }
                    else
                    {
                        xSpeed = Math.Abs(xSpeed);   // change vector of
movement if met left side of obstacle
                    }

                    if (activePictureBox.Location.Y < obstacle1.Location.Y)
                    {
                        ySpeed = -Math.Abs(ySpeed);  // change vector of
movement if met top side of obstacle
```

```
                    }
                    else
                    {
                        ySpeed = Math.Abs(ySpeed);    // change vector of
movement if met bottom side of obstacle
                    }
                }

                // change vector of movement if met one of the sides
                if (activePictureBox.Location.X >= ClientSize.Width -
activePictureBox.Width || activePictureBox.Location.X <= 0)
                {
                    xSpeed = -xSpeed;
                }
                if (activePictureBox.Location.Y >= ClientSize.Height -
activePictureBox.Height || activePictureBox.Location.Y <= 0)
                {
                    ySpeed = -ySpeed;
                }

                // update position
                activePictureBox.Location = new
Point(activePictureBox.Location.X + xSpeed, activePictureBox.Location.Y +
ySpeed);
            };
            moveTimer.Start();
        }

        private void pictureBox1_Click(object sender, EventArgs e)
        {

        }
        private void Form1_Load(object sender, EventArgs e)
        {
            Infinity_Moove(sender, EventArgs.Empty);
        }
    }
}
```

Пояснення коду для форми UserInteractionForm:

Сточатку ми ініціалізуємо нашу форму, визначаємо її розмір, пвсля чого створюємо перешкоду ***obstance1*** *та визначаємо її характеристика та додаємо її до форми. створюємо значення, що буде відповідати за початкову позицію, а також для події* ***KeyDown*** *підписуємося на метод* ***Form1_KeyDown****. Аналогічно для обох* ***PictureBox*** *підписуємося на методи що відповідатимуть за реалізацію функціоналу* ***drag and drop***

```
public UserInteractionForm()
{
    InitializeComponent();
    this.ClientSize = new Size(800, 600);
    // create obstacle
    obstacle1 = new Panel
    {
        BackColor = Color.Red,
        Size = new Size(100, 100),
        Location = new Point(300, 180)
    };
    // add obstacle on the form
    this.Controls.Add(obstacle1);
    initialImagePosition = new Point(10, 10);
    this.KeyDown += new KeyEventHandler(Form1_KeyDown);

    activePictureBox = pictureBox1;
    // subscribe on events
    pictureBox1.MouseDown += new MouseEventHandler(PictureBox_MouseDown);
    pictureBox1.MouseMove += new MouseEventHandler(PictureBox_MouseMove);
    pictureBox1.MouseUp += new MouseEventHandler(PictureBox_MouseUp);

    pictureBox2.MouseDown += new MouseEventHandler(PictureBox_MouseDown);
    pictureBox2.MouseMove += new MouseEventHandler(PictureBox_MouseMove);
    pictureBox2.MouseUp += new MouseEventHandler(PictureBox_MouseUp);

}
```

Визначаєсо наші змінні серед них також швидкість переміщення по осях прапорці, що показують чи перетягується та чи рухається ***PictureBox****.*

Також змінна, що відповідає за те наскільки зміститься курсор відносно ***PictureBox***

```
private Panel obstacle1;
private Point initialImagePosition;
Timer moveTimer = new Timer();
int xSpeed = 20;
int ySpeed = 20;
private PictureBox activePictureBox; // current PictureBox
private bool isDragging = false; // flag for dragging
private bool isMooving = true; // flag for infinity move
private Point mouseOffset; // cursor shift related of PictureBox
```

Здебільшого так виглядає логіка обробки події при взаємодії з клавіатурою:

- *Стрілка вправо - рух вправо*
- *Стрілка вліво - рух вліво*
- *Стрілка уверх - рух уверх*
- *Стрілка вниз - рух вниз*
- *R - зупинити рух то поставити зображення у початкове положення*
- *Пробіл - зупинити / запустити рух зображення*
- *Таб - перемикання між зображеннями*

```
else if (e.KeyCode == Keys.Right)
{
    if (activePictureBox.Right < ClientSize.Width - 20)
    {
        activePictureBox.Location = new Point(activePictureBox.Location.X + 20, activePictureBox.Location.Y);
    }
}
else if (e.KeyCode == Keys.Tab)
{
    activePictureBox = activePictureBox == pictureBox1 ? pictureBox2 : pictureBox1;
}
else if (e.KeyCode == Keys.R) {
    moveTimer.Stop();
    activePictureBox.Location = initialImagePosition;
    xSpeed = Math.Abs(xSpeed);
    ySpeed = Math.Abs(ySpeed);
}
else if (e.KeyCode == Keys.Space)
{
    Console.WriteLine(isMooving);
    if (isMooving)
    {
        moveTimer.Stop();
        isMooving = false;
    }
    else
    {
        moveTimer.Start();
        isMooving = true;
    }
}
Invalidate(); // render
```

Метод ***IsColliding*** *для перевірки, чи перетинаються два прямокутники на екрані*

```
1 reference
private bool IsColliding(Control pictureBox, Panel obstacle)
{
    Rectangle pictureBoxRect = pictureBox.Bounds;
    Rectangle obstacleRect = obstacle.Bounds;

    return pictureBoxRect.IntersectsWith(obstacleRect);
}
```

*При натисканні на ліву кнопку миші, зупиняємо таймек, що відповідає за ітераційне виконання певної логіки, сетаємо значення **isDtagging** в **true**, а **isMoving** в **false**. Запам'ятовуємо позиці. в якій було натиснуто на зображення.*

```
private void PictureBox_MouseDown(object sender, MouseEventArgs e)
{
    // set active PictureBox
    activePictureBox = sender as PictureBox;

    if (activePictureBox != null && e.Button == MouseButtons.Left)
    {
        // run the procces of drugging
        isDragging = true;
        isMooving = false;
        moveTimer.Stop();

        // calculste cursor shift related to PictureBox
        mouseOffset = new Point(e.X, e.Y);
    }
}
```

Поки переміщаємо зображення - обраховуємо нову позицію зобрадення

```
private void PictureBox_MouseMove(object sender, MouseEventArgs e)
{
    if (isDragging && activePictureBox != null)
    {
        // calculate new position of PictureBox
        Point newLocation = activePictureBox.Location;
        newLocation.X += e.X - mouseOffset.X;
        newLocation.Y += e.Y - mouseOffset.Y;
        activePictureBox.Location = newLocation;
    }
}
```

*Коли відпускаємо кнопку миші, тоді значення **isDragging** змінюємо на **false***

```
2 references
private void PictureBox_MouseUp(object sender, MouseEventArgs e)
{
    // end dragging
    if (e.Button == MouseButtons.Left)
    {
        isDragging = false;
    }
}
```

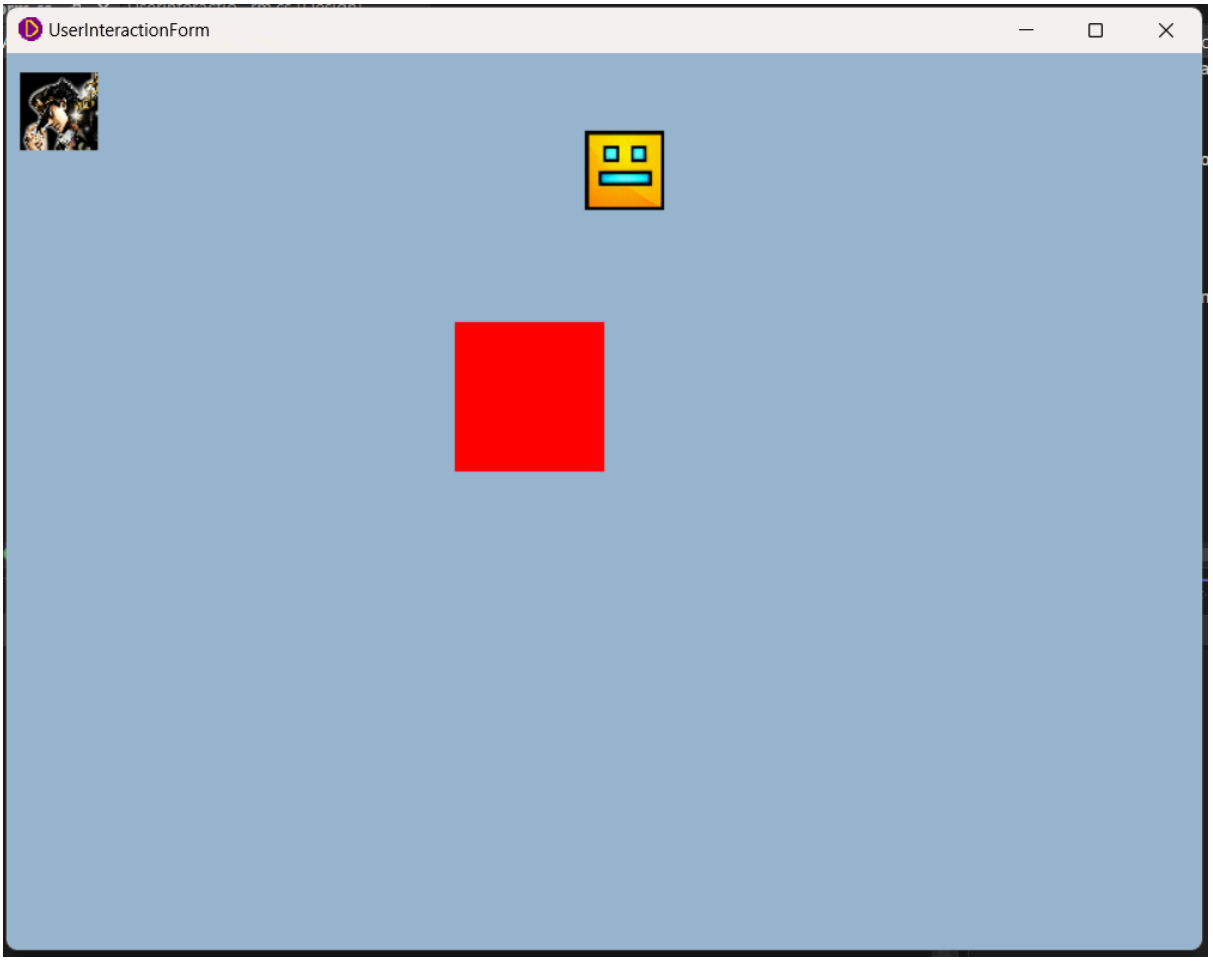
*У функції **Infinity_Moove** запускаємо таймер, та перевіряємо, якщо було зіткнення, тоді перевіряємо з якою із сторін було зіткнення і в залежності від цього змінюємо значення **xSpeed** або **ySpeed** на додатнє якщо зіткнення було з нижньою або правою стороною інакше змінюємо на протилежне значення. Також перевіряємо чи не виходить зображення за межі форми після чого оновлюємо координати зображення.*

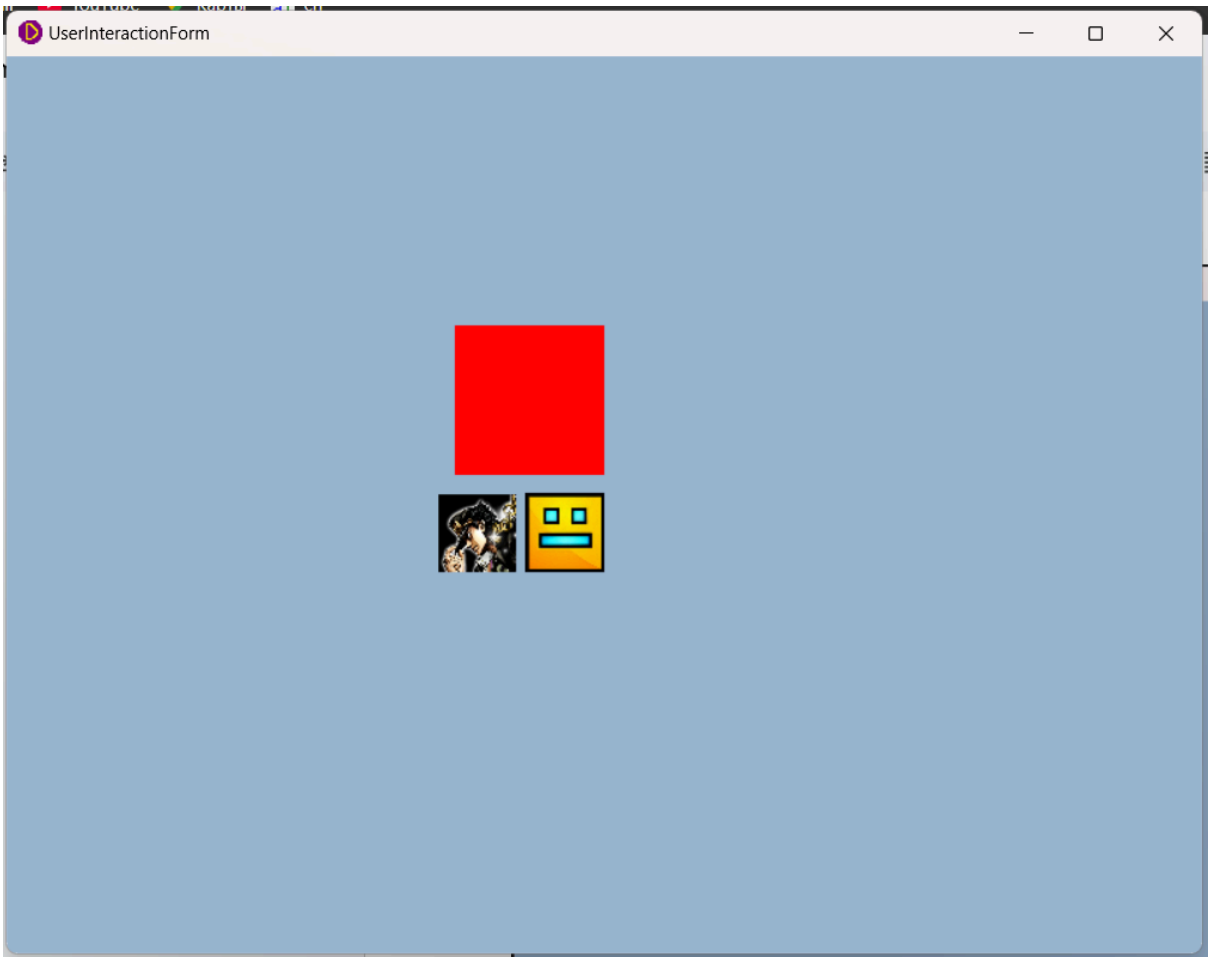
```
isMooving = true;
moveTimer.Tick += (s, eventArgs) =>
{

    if (IsColliding(activePictureBox, obstacle1))
    {
        if (activePictureBox.Location.X < obstacle1.Location.X)
        {
            xSpeed = -Math.Abs(xSpeed);  // change vector of movement if met right side of obstacle
        }
        else
        {
            xSpeed = Math.Abs(xSpeed);   // change vector of movement if met left side of obstacle
        }

        if (activePictureBox.Location.Y < obstacle1.Location.Y)
        {
            ySpeed = -Math.Abs(ySpeed);  // change vector of movement if met top side of obstacle
        }
        else
        {
            ySpeed = Math.Abs(ySpeed);   // change vector of movement if met bottom side of obstacle
        }
    }

    // change vector of movement if met one of the sides
    if (activePictureBox.Location.X >= ClientSize.Width - activePictureBox.Width || activePictureBox.Location.X <= 0)
    {
        xSpeed = -xSpeed;
    }
    if (activePictureBox.Location.Y >= ClientSize.Height - activePictureBox.Height || activePictureBox.Location.Y <= 0)
    {
        ySpeed = -ySpeed;
    }

    // update position
    activePictureBox.Location = new Point(activePictureBox.Location.X + xSpeed, activePictureBox.Location.Y + ySpeed);
};
moveTimer.Start();
```

*При ініціалізації форми, як тільки її було відрендерено запускаємо вічний рух **Infinity_Moove***

```
private void Form1_Load(object sender, EventArgs e)
{
    Infinity_Moove(sender, EventArgs.Empty);
}
```

Результат:





Висновок:
Виконуючи цю лабораторну роботу я зіштовхнувся з проблемою того, як відстежити відбивання від перешкод і це було дуже цікаво. Також ознайомився зі взаємодією графічних компонентів з клавіатурою та методом drag and drop в Windows Form Application