2021 tavaszi félév.

Projekt dokumentáció

Arduino based remote controller for platformer game

Miliás Bálint NK: G2SSZ1 Email: balint.milias@gmail.com
Martin Dávid NK: X9BPHI Email: martindavidsuli@gmail.com

Tartalomjegyzék

I. Bevezetés:

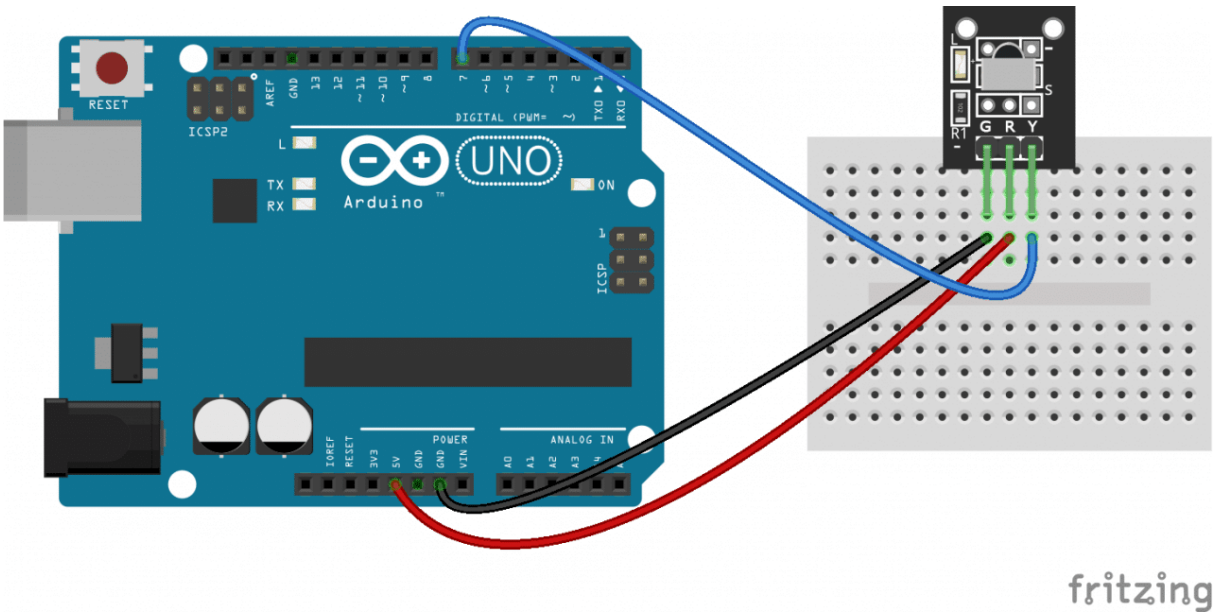
A feladat 2 részre bontható le.

1: Egy kontroller megalkotása egy Arduino UNO mikrokontroller segítségével. A kontroller alapja egy hétköznapi távirányító, amely az infravörös jelek leadásával képes az Arduinon és egy Visual Stúdió Console Application-ön keresztül a billentyűzetünk és az egerünk gomblenyomásait szimulálni.

2: Egy platformer játék készítése, amit a fent említett kontrollerrel tudunk irányítani. A játék Windows Form Application alapon van létrehozva Microsoft Visual Stúdióban.

II. Kontroller

A távirányító általában infravörös fény kibocsátásával ad utasításokat ezek vételére felkészített szerkezetnek, mint amilyenek például a televíziók vagy más elektronikus eszközök. Ahhoz, hogy az arduino tudja fogadni egy távirányító jelét egy infrared receiver-re van szükség melyet az 1. ábrán látható módon kell összeilleszteni. [1]



1. ábra Infrared Receiver és Arduino összeillesztése

Amint az arduino képes a jel fogadására az IDE fejlesztőkörnyezetben kell dekódolnunk (2. ábra). A távirányítónak minden gombja külön infrared jelet küld, amit ha hexadecimális módon megjelenítünk, akkor már el tudjuk választani kód szerint a gombjainkat. A jelet, a soros port maximum 115200 bit/másodperc fogadja. Led-es visszajelzéssel látszik a mikrokontrolleren, ha beérkezik valamilyen jel. Végül egy ciklus kiírja a jeleket hexadecimális értékkel.[2]

2. ábra Jelek dekódolása

A dekódolt jelet egy Visual stúdión fejlesztett console application veszi át, amely képes szimulálni egy kontrollert (3. ábra). A billentyű

lenyomását és felemelését általunk határozott gyorsasággal lehet konfigurálni milliszekundumos pontossággal.

```
0 references
class Program
{
    0 references
    static void Main(string[] args)
    {
        InputSimulator sim = new InputSimulator(); //interfész a billentyűzet vagy egér bevitel szimulálására
        SerialPort myport = new SerialPort();
        myport.BaudRate = 115200;  // a soros port maximum 115200 bit/másodperc átvitelére képes.
        myport.PortName = "COM3"; // com3 as port használata
        myport.Open(); // port megnyitása




        do
        {


            string a = myport.ReadExisting();


                if (a == "62\r\n") //távirányító ha 62-es jelet küld
                {


                    sim.Keyboard.KeyDown(VirtualKeyCode.RIGHT); // Jobb nyíil billentyű lenyomása
                Thread.Sleep(100);                              // Késleltetés
                sim.Keyboard.KeyUp(VirtualKeyCode.RIGHT);       //Billentyű felemelése

                }


            if (a == "65\r\n")
```

3. ábra Billentyűzet lenyomásának szimulálása

A távirányítón található gombmennyiségnek köszönhetően sok lehetőségünk van különböző billentyű parancsokat beállítani.

III. Játék

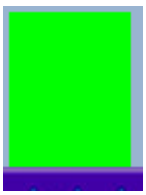
A játék menete: A játékos egy olyan pályán van, ahol van egy zárt ajtó, be kell gyűjtenie a kulcsot a szint másik végéből, miközben összegyűjti a pályán lévő érméket, úgy hogy nem esik le a platformokról. Ha összegyűjtötted a kulcsot, akkor ki tudod nyitni az ajtót, és befejezheted a szintet.[3]

Célok:

1. Egy platformer játék létrehozása a Visual Stúdió c# programozási nyelvvel
2. Több picturebox használata és vezérlésük egy timer segítségével
3. Gravitációs és ugróerő létrehozás
4. A háttér, a tárgyak és a platformok görgetése, ahogy a játékos balra vagy jobbra mozog.

Formatervezés:

Egy Windows Form Application méretét 2000 és 480 pixelre módosítottuk és hátterét kék színűre állítottuk. Ezután már hozzáadhatunk platformokat, érméket, ajtót és kulcsot. A properties résznél adjuk hozzá a képeket majd el is nevezzük és méretet is adunk nekik.



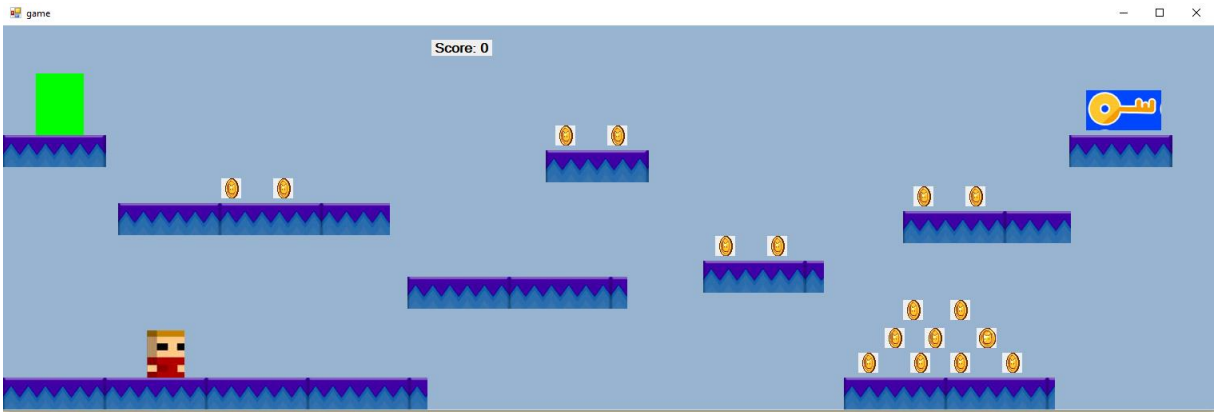
Name: door
Tag: door
Image: door
Size: 50;60

Name: player
Tag: none
Image: player
Size Mode: Auto Size
Location: 86, 398

Name: coin
Tag: coin
Image: coin
Size: 35, 30

Name: key
Tag: key
Image: key
Size Mode: Auto Size

A játék 4. ábrán látható.



4. ábra A játék kinézete az elemek elhelyezése után

Kódolás:

Változók: (5. ábra)

```
bool goLeft, goRight, jumping, hasKey; //változók  a játékos irányítására illetve hogy van e nála kulcs

int jumpSpeed = 10; // ugrás sebesség beállítása
int force = 16; // az ugrás ereje
int score = 0; //alapértelmezett pontszám

int playerSpeed = 10; //a játékos sebességét 10-re állítja
int backgroundSpeed = 8; //beállítja a háttér sebességét
```

5. ábra Változók

Billentyű lenyomva esemény: (6. ábra)

Ez az esemény akkor lép működésbe, amikor a játékos megnyom egy billentyűt a billentyűzeten.

```
1 reference
private void KeyIsDown(object sender, KeyEventArgs e)
{
    //ha a játékos megnyomta a bal oldali billentyűt és a játékos a panelen belül van
    //akkor a goleft boolean-t true-ra állítjuk.
    if (e.KeyCode == Keys.Left)
    {
        goLeft = true;
    }
    //akkor a goRight boolean-t true-ra állítjuk.
    if (e.KeyCode == Keys.Right)
    {
        goRight = true;
    }
    // ha a játékos megnyomta a space-t és a jumping false
    if (e.KeyCode == Keys.Space && jumping == false)
    {
        //akkor a jumpingot true-ra állitja
        jumping = true;
    }
```

6. ábra Billentyű lenyomva esemény

Billentyű felenged esemény: (7. ábra)

```
1 reference
private void KeyIsUp(object sender, KeyEventArgs e)
{
    // ha a játékos felengedte a bal oldali billentyűt akkor a goleft boolean-t false-ra állítjuk
    if (e.KeyCode == Keys.Left)
    {
        goLeft = false;
    }
    // ha a játékos felengedte a jobb oldali gombot akkor a goright booleant-t false ra állítjuk

    if (e.KeyCode == Keys.Right)
    {
        goRight = false;
    }
    // ha a gombokat felengedtük, ellenőrizzük hogy a jumping igaz-e
    //ha igen, akkor vissza kell állítanunk false-ra, hogy a játékos újra tudjon ugrani.
    if (jumping == true)
    {
        jumping = false;
    }
}
```

7. ábra Billentyű felengedése esemény

MainGameTimerEvent: (8-9-10. ábra)

Ez az esemény irányítja az egész játékot, a játékos mozgásától kezdve a környezeten át a tárgyak eltávolításáig, valamint a tárgyak form-ból való eltávolításáig, amikor azok összeütköznek egymással.

```
1 reference
private void MainTimeEvent(object sender, EventArgs e)
{
    // eredmeny
    txtScore.Text = "Score: " + score;
    txtScore.Left = 536;

    // az ugrássebesség  összekapcsolása a játékossal és helyével
    player.Top += jumpSpeed;

    // ha a goleft true és a playerleft nagyobb mint 60 pixel
    // csak ezután mozgassuk a játékost a balra
    if (goLeft == true && player.Left > 60)
    {
        player.Left -= playerSpeed;
    }
    //ha a goright true
    //és a playerleft + (szélessége + 60 ) kisebb mint a form szélessége
    // ezután a játékost  jobbra mozgatjuk hozzáadva a bal oldalához
    if (goRight == true && player.Left + (player.Width + 60) < this.ClientSize.Width)
    {
        player.Left += playerSpeed;
    }

    // ha a goleft igaz és a background bal oldala kisebb mint 0
    // akkor movegamelent function
    if (goLeft == true && background.Left < 0)
    {
        background.Left += backgroundSpeed;
        MoveGameElements("forward");
    }

    if (goRight == true && background.Left > -1372)
    {
        background.Left -= backgroundSpeed;
        MoveGameElements("back");
    }
```

8. ábra

```
//ha az ugrás igaz
// akkor az ugrás sebességét -12-re változtatjuk
// csökkents a erőt 1-re
if (jumping == true)
{
    jumpSpeed = -12;
    force -= 1;
}
else
{
    // másképpen az ugrás sebességét 12-re változtatjuk
    jumpSpeed = 12;
}

// ha az ugrás igaz és az erő kisebb, mint 0
// akkor az ugrást hamisra változtatjuk
if (jumping == true && force < 0)
{
    jumping = false;
}

// a ciklus ami a from összes vezérlőelemének ellenőrzését végzi
foreach (Control x in this.Controls)
{
    // x egy picturebox aminek platform tag-je van
    if (x is PictureBox && (string)x.Tag == "platform")
    {
        // ekkor ellenőrizzük, hogy a játékos ütközik-e a platformmal.
        // és az ugrás értéke hamis
        if (player.Bounds.IntersectsWith(x.Bounds) && jumping == false)
        {
            force = 8;
            player.Top = x.Top - player.Height; //  a játékost a picturebox tetejére helyezzük
            jumpSpeed = 0;
        }

        x.BringToFront();

    }
    // ha a picture box tagje coin
    if (x is PictureBox && (string)x.Tag == "coin")
    {
        //ha a játékos ütközik az érme picturebox-ával

        if (player.Bounds.IntersectsWith(x.Bounds) && x.Visible == true)
        {
            x.Visible = false; //eltüntetjük
            score += 1;// a score-t növeljük 1-el
        }

    }
}
```

9. ábra

```
//ha az ajtóhoz ér és a haskey igaz akkor játék megáll és üzenetdoboz
// jatek ujrainditas function
if (player.Bounds.IntersectsWith(door.Bounds) && hasKey == true)
{

    GameTimer.Stop();
    MessageBox.Show("Nyertél " + Environment.NewLine + "nyomj enter hogy ujrakezd");
    RestartGame();
}

// ha a játékos a form magassága alá megy, akkor vége a játéknak
if (player.Top + player.Height > this.ClientSize.Height)
{
    GameTimer.Stop();
    MessageBox.Show("Meghaltál!" + Environment.NewLine + "nyomj enter hogy ujrakezd");
    RestartGame();
}
```

10. ábra

A játék bezárása: (11 ábra)

```
// játék bezárása
1 reference
private void CloseGame(object sender, FormClosedEventArgs e)
{
    Application.Exit();

}
```

11. ábra Játék bezárása

A játék újrakezdése: (12-ábra)

```
//játék ujrakezdes
2 references
private void RestartGame()
{
    Form1 newWindow = new Form1();
    newWindow.Show();
    this.Hide();
}
2 references
```

12. ábra Játék újrakezdése

Játékelemek mozgása: (13.ábra)

```
private void MoveGameElements(string direction)
{

    foreach (Control x in this.Controls)
    { //ha az x picturebox és az x tag egyenlő platform vagy coin vagy key vagy door
        if (x is PictureBox && (string)x.Tag == "platform" || x is PictureBox && (string)x.Tag == "coin" || x is PictureBox && (string)x.Tag == "key" || x is PictureBox && (string)x.Tag == "door")
        {
            // ha az irany az back
            // akkor a háttér balra megy
            if (direction == "back")
            {
                x.Left -= backgroundSpeed;
            }
            // ha előre akkor jobbra
            if (direction == "forward")
            {
                x.Left += backgroundSpeed;
            }


        }
    }


}
```

13. ábra Játék elemek mozgása

IV References

[1] Zeebaree, S. R., & Yasin, H. M. (2014). Arduino based remote controlling for home: power saving, security and protection. International Journal of Scientific & Engineering Research, 5(8), 266-272.

[2] instructables „Arduino-Controlled Platformer Game With Joystick and IR Receiver" (2021) [Online] Available: https://www.instructables.com/Arduino-Controlled-Platformer-Game-With-Joystick-a/

[3] mooict „C# Tutorial – Create a simple platform game in visual studio" (2021) [Online] Available: https://www.mooict.com/c-tutorial-create-a-simple-platform-game-in-visual-studio/