using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

using System.Xml.Serialization;

namespace oliskoMidii

{

// Tietue mihin tulee pelaajien tietoja

public struct Tilastot

{

public string pelaaja;

public int voitto;

public int häviö;

public int tasapelia;

}

public partial class Form1 : Form

{

// Muuttujia

private bool \_allowclick = true;

private PictureBox \_ekaArvaus;

private readonly Random \_random = new Random();

private readonly Timer \_clickTimer = new Timer();

int ticks = 30;

readonly Timer timer = new Timer { Interval = 1000 };

bool vuoro = true;

public Tilastot[] tilet = new Tilastot[2];

private string ukkelit = "VähäTilastoja.xml";

List<Tilastot> tilastoja = new List<Tilastot>();

public int voittaja = 0;

public int voittaja1 = 0;

public int häviäjä = 0;

public int häviäjä1 = 0;

public int tasapeli;

// Tiedoston teko,tallennus ja lukeminen = Seriaize ja Deserialize

public void SerializeXML(List<Tilastot> input)

{

System.Xml.Serialization.XmlSerializer serializer = new XmlSerializer(input.GetType());

var sw = new StreamWriter(ukkelit);

serializer.Serialize(sw, input);

sw.Close();

}

public List<Tilastot> DeserializeXML()

{

if (File.Exists(ukkelit))

{

StreamReader stream = new StreamReader(ukkelit);

XmlSerializer ser = new XmlSerializer(typeof(List<Tilastot>));

object obj = ser.Deserialize(stream);

stream.Close();

return (List<Tilastot>)obj;

}

else

return null;

}

public Form1()

{

InitializeComponent();

}

private PictureBox[] pictureBoxes

{

get { return Controls.OfType<PictureBox>().ToArray(); }

}

// 24 korttia kuvat

private static IEnumerable<Image> Images

{

get

{

return new Image[]

{

Properties.Resources.gorilla,

Properties.Resources.hevonen,

Properties.Resources.Janis,

Properties.Resources.kana,

Properties.Resources.kilpikonna,

Properties.Resources.kirahvi,

Properties.Resources.kissa,

Properties.Resources.koirax,

Properties.Resources.kotka,

Properties.Resources.käärme,

Properties.Resources.leijona,

Properties.Resources.pingviini,

};

}

}

// 16 korttia kuvat

private static IEnumerable<Image> imakes

{

get

{

return new Image[]

{

Properties.Resources.gorilla,

Properties.Resources.hevonen,

Properties.Resources.Janis,

Properties.Resources.kana,

Properties.Resources.kilpikonna,

Properties.Resources.kirahvi,

Properties.Resources.kissa,

Properties.Resources.koirax,

};

}

}

// Tulostaa tulokset tiedostoon ja aloittaa uuden pelin kun vanha päättyy.

private void Uudet()

{

Tulostus();

foreach (var pic in pictureBoxes)

{

pic.Tag = null;

pic.Visible = true;

}

Piilota();

Randomit();

ticks = 30;

timer.Start();

Pisteet1.Text = "0";

Pisteet2.Text = "0";

}

// Vaihtaa pictureboxeihin kysymysmerkkikuvan

private void Piilota()

{

foreach(var pic in pictureBoxes)

{

pic.Image = Properties.Resources.kysmäri;

vuoro = !vuoro;

}

}

// Random järjestys

private PictureBox GetFreeSlot()

{

int num;

do

{

num = \_random.Next(0, pictureBoxes.Count());

}

while (pictureBoxes[num].Tag != null);

return pictureBoxes[num];

}

// Asettaaa kuvat paikoilleen ja katsoo että kaikkia tulee 2kpl

private void Randomit()

{

if (cBox1.SelectedIndex.Equals(0))

{

foreach (var image in Images)

{

GetFreeSlot().Tag = image;

GetFreeSlot().Tag = image;

}

}

else

{

foreach(var image in imakes)

{

GetFreeSlot().Tag = image;

GetFreeSlot().Tag = image;

}

}

}

// Kääntää kortteja ja vertailee niitä, lisää pisteitä funktion avulla, vaihtaa vuoroa ja tarkistaa onko kaikki pictureboxit jo käännetty eli poissa näkyvistä

private void ClickImage(object sender, EventArgs e)

{

if (!\_allowclick) return;

var pic = (PictureBox)sender;

if(\_ekaArvaus == null)

{

\_ekaArvaus = pic;

pic.Image = (Image)pic.Tag;

return;

}

pic.Image = (Image)pic.Tag;

if(pic.Image == \_ekaArvaus.Image && pic != \_ekaArvaus)

{

pic.Visible = \_ekaArvaus.Visible = false;

{

\_ekaArvaus = pic;

}

Piilota();

}

else

{

\_allowclick = false;

\_clickTimer.Start();

}

if(pic.Image != \_ekaArvaus.Image && pic != \_ekaArvaus)

{

vuoro = !vuoro;

}

else

{

KummanPisteet();

}

if(checkB1.Checked == false)

{

if (vuoro == true)

{

LBL4.Text = "Vuoro : " + txtB1.Text;

}

else

{

LBL4.Text = "Vuoro : " + txtB2.Text;

}

}

else

{

LBL4.Visible = false;

}

\_ekaArvaus = null;

if (pictureBoxes.Any(p => p.Visible)) return;

Uudet();

}

private void \_clickTimer\_Tick(object sender, EventArgs e)

{

Piilota();

\_allowclick = true;

\_clickTimer.Stop();

}

// Vertaa kummalle pisteet tulevat oikean parin löytämisestä

private void KummanPisteet()

{

if (checkB1.Checked == false)

{

if (vuoro == true)

{

Pisteet1.Text = Convert.ToString(Convert.ToInt32(Pisteet1.Text) + 1);

}

else

{

Pisteet2.Text = Convert.ToString(Convert.ToInt32(Pisteet2.Text) + 1);

}

}

else

{

if (vuoro == true)

{

Pisteet1.Text = Convert.ToString(Convert.ToInt32(Pisteet1.Text) + 1);

}

else

{

Pisteet1.Text = Convert.ToString(Convert.ToInt32(Pisteet1.Text) + 1);

}

}

}

// Vertailee kuka voitti, hävisi tai tuliko tasapeli

private void Tulokset()

{

int x = Convert.ToInt32(Pisteet1.Text);

int y = Convert.ToInt32(Pisteet2.Text);

if (x > y)

{

voittaja = x - x + 1;

häviäjä1 = y - y + 1;

MessageBox.Show(txtB1.Text + " voitti!");

}

else if (x < y)

{

voittaja1 = y - y + 1;

häviäjä = x - x + 1;

MessageBox.Show(txtB2.Text + " voitti!");

}

else if(x == y)

{

tasapeli = +1;

MessageBox.Show("Peli päättyi tasapeliin!");

}

}

// Vie tulokset tiedostoon

public void Tulostus()

{

if (checkB1.Checked == false)

{

Tulokset();

Tilastot e1;

e1.pelaaja = txtB1.Text;

e1.voitto = voittaja;

e1.häviö = häviäjä;

e1.tasapelia = tasapeli;

Tilastot e2;

e2.pelaaja = txtB2.Text;

e2.voitto = voittaja1;

e2.häviö = häviäjä1;

e2.tasapelia = tasapeli;

tilet[0] = e1;

tilet[1] = e2;

int ind = -1;

for(int i = 0; i < tilastoja.Count; i++)

{

if(tilastoja[i].pelaaja == e1.pelaaja)

{

ind = i;

break;

}

}

if(ind != -1)

{

e1.tasapelia += tilastoja[ind].tasapelia;

e1.voitto += tilastoja[ind].voitto;

e1.häviö += tilastoja[ind].häviö;

tilastoja.RemoveAt(ind);

}

for (int i = 0; i < tilastoja.Count; i++)

{

if (tilastoja[i].pelaaja == e2.pelaaja)

{

ind = i;

break;

}

}

if (ind != -1)

{

e2.tasapelia += tilastoja[ind].tasapelia;

e2.voitto += tilastoja[ind].voitto;

e2.häviö += tilastoja[ind].häviö;

tilastoja.RemoveAt(ind);

}

tilastoja.Add(e1);

tilastoja.Add(e2);

}

else

{

Tulokset();

Tilastot e1;

e1.pelaaja = txtB1.Text + " (Pelattu yksinpelinä)";

e1.voitto = voittaja;

e1.häviö = häviäjä;

e1.tasapelia = tasapeli;

tilet[0] = e1;

int ind = -1;

for (int i = 0; i < tilastoja.Count; i++)

{

if (tilastoja[i].pelaaja == e1.pelaaja)

{

ind = i;

break;

}

}

if (ind != -1)

{

e1.tasapelia += tilastoja[ind].tasapelia;

e1.voitto += tilastoja[ind].voitto;

e1.häviö += tilastoja[ind].häviö;

tilastoja.RemoveAt(ind);

}

tilastoja.Add(e1);

}

SerializeXML(tilastoja);

}

private void Form1\_Load(object sender, EventArgs e)

{

if (File.Exists(ukkelit))

{

tilastoja = DeserializeXML();

}

}

/\* Käskee antamaan pelaajan/pelaajien nimet ennenkun peli voi alkaa

Aloittaa pelin niillä asetuksilla mitä käyttäjä on valinnut\*/

private void button1\_Click(object sender, EventArgs e)

{

checkB1.Enabled = false;

if (checkB1.Checked == true)

{

if (txtB1.Text == string.Empty)

{

MessageBox.Show("Anna pelinimesi");

return;

}

}

else

{

if (txtB1.Text == string.Empty || txtB2.Text == string.Empty)

{

MessageBox.Show("Anna pelaajien nimet");

return;

}

}

if (this.cBox1.SelectedIndex.Equals(0))

{

Randomit();

Piilota();

\_clickTimer.Interval = 1000;

\_clickTimer.Tick += \_clickTimer\_Tick;

LBL5.Text = txtB1.Text + " löytämät parit : ";

LBL6.Text = txtB2.Text + " löytämät parit : ";

if (checkB1.Checked == false)

{

LBL4.Text = "Vuoro : " + txtB1.Text;

}

}

else

{

this.Controls.Remove(pictureBox17);

this.Controls.Remove(pictureBox18);

this.Controls.Remove(pictureBox19);

this.Controls.Remove(pictureBox20);

this.Controls.Remove(pictureBox21);

this.Controls.Remove(pictureBox22);

this.Controls.Remove(pictureBox23);

this.Controls.Remove(pictureBox24);

Randomit();

Piilota();

\_clickTimer.Interval = 1000;

\_clickTimer.Tick += \_clickTimer\_Tick;

LBL5.Text = txtB1.Text + " löytämät parit : ";

LBL6.Text = txtB2.Text + " löytämät parit : ";

if (checkB1.Checked == false)

{

LBL4.Text = "Vuoro : " + txtB1.Text;

}

}

}

// Käynnistää pelin uudelleen

private void BTN2\_Click(object sender, EventArgs e)

{

Application.Restart();

}

//Tekee pelaaja2:n tyhjistä tiedoista näkymättömiä jos käyttäjä valitsee pelata yksin

private void checkB1\_CheckedChanged(object sender, EventArgs e)

{

if(checkB1.Checked == true)

{

LBL6.Visible = false;

txtB2.Visible = false;

LBL3.Visible = false;

Pisteet2.Visible = false;

LBL4.Visible = false;

}

else

{

LBL6.Visible = true;

txtB2.Visible = true;

LBL3.Visible = true;

Pisteet2.Visible = true;

LBL4.Visible = true;

}

}

// Pakotetaan käyttäjä kirjoittamaan pelaajien nimet

private void txtB1\_Validated(object sender, EventArgs e)

{

TextBox tb = (TextBox)sender;

ep1.SetError(tb, "");

}

private void txtB1\_Validating(object sender, CancelEventArgs e)

{

TextBox tb = (TextBox)sender;

if (tb.Text.Length <= 0)

{

ep1.SetError(tb, "Kenttä pakollinen");

e.Cancel = true;

}

}

private void Form1\_FormClosing(object sender, FormClosingEventArgs e)

{

}

// Avataan tilastot sisältävä tiedosto

private void tilastotToolStripMenuItem\_Click(object sender, EventArgs e)

{

System.Diagnostics.Process.Start("VähäTilastoja.xml");

}

// Avataan ohje msgbox

private void ohjeetToolStripMenuItem\_Click(object sender, EventArgs e)

{

MessageBox.Show("Muistipelissä on tarkoitus löytää kaksi samanlaista kuvaa muiden kuvien joukosta.Vuorossa oleva pelaaja kääntää kaksi korttia klikkaamalla niitä.Jos kuvat ovat samat, pelaaja on löytänyt parin ja saa jatkaa.Jos paria ei löydy, siirtyy vuoro seuraavalle.Voittaja on se, joka löytää eniten kuvapareja.",

"Muistipelin ohjeet");

}

// Suljetaan peli

private void poistuPelistäToolStripMenuItem\_Click(object sender, EventArgs e)

{

Environment.Exit(0);

}

}

}