MiniPaint

Software Documentation

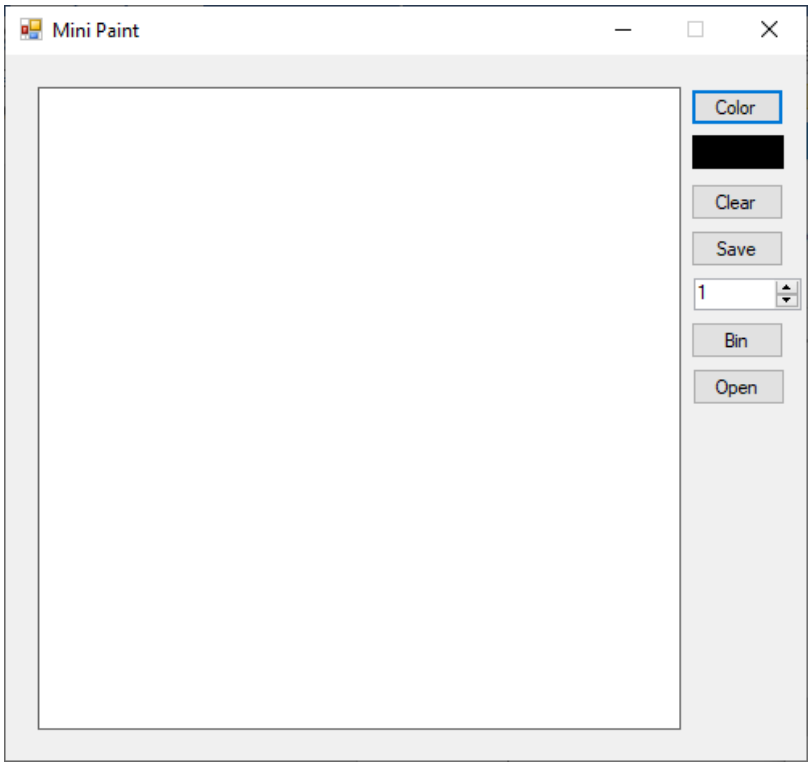
Author: matiwa

Table of contents

Introduction

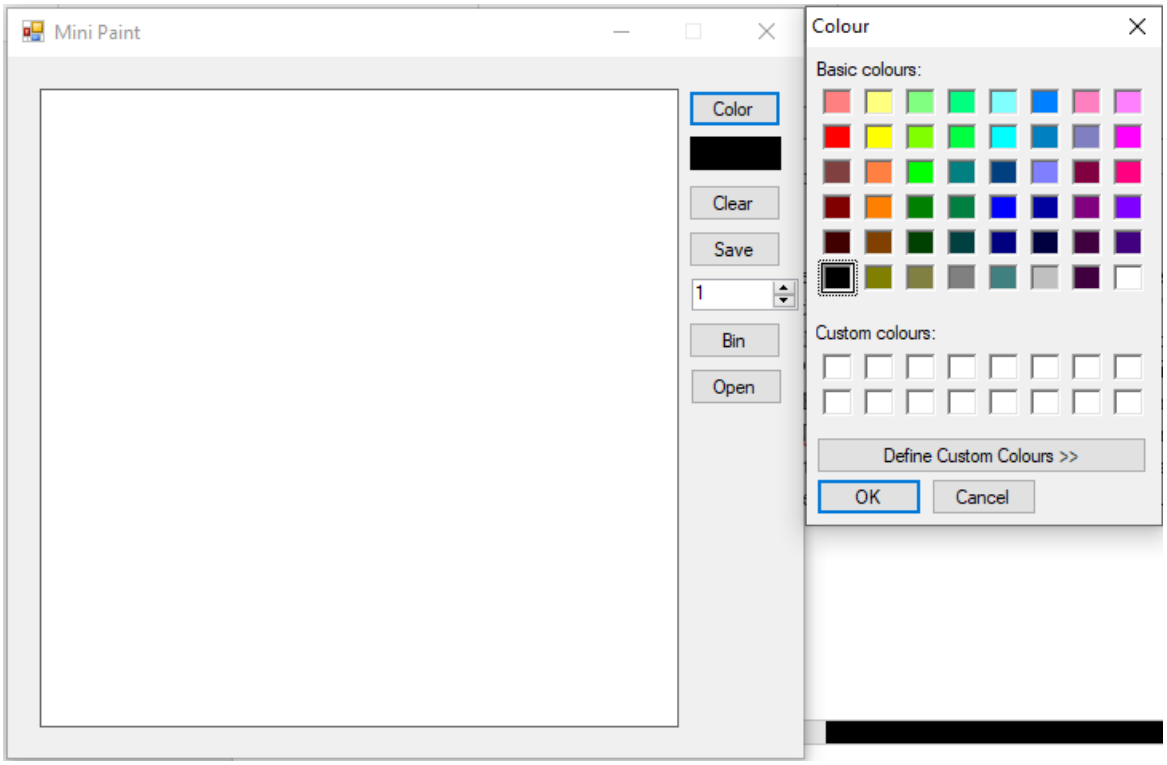
This software documentation includes: description of the application's operation, what is needed for use, algorithms used, interface description and source code description. This application is used to simple sketch.

Describing of the application’s operation

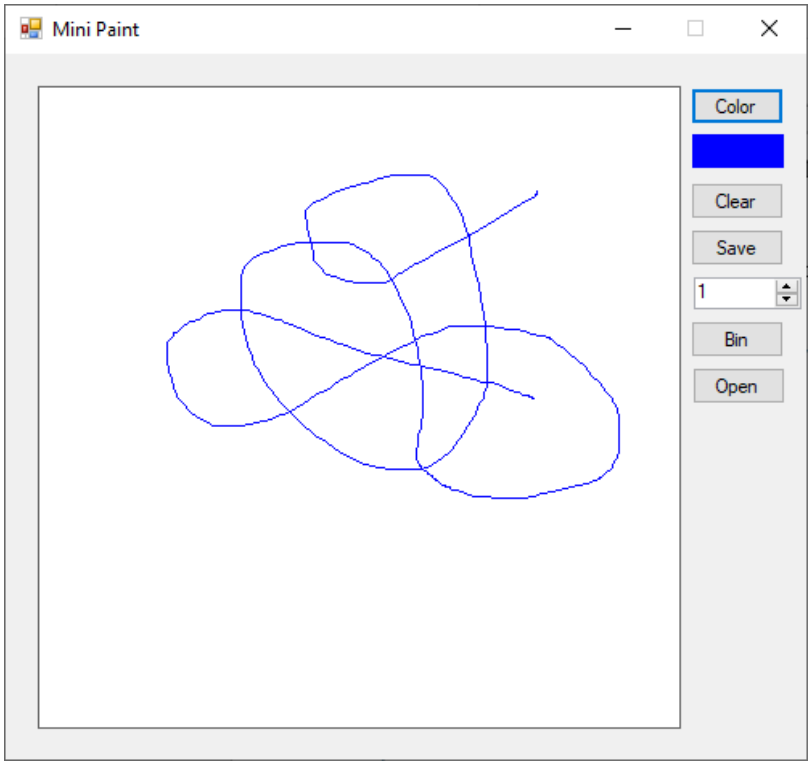


Drawing 1: The beginning of the application's operation [own study]

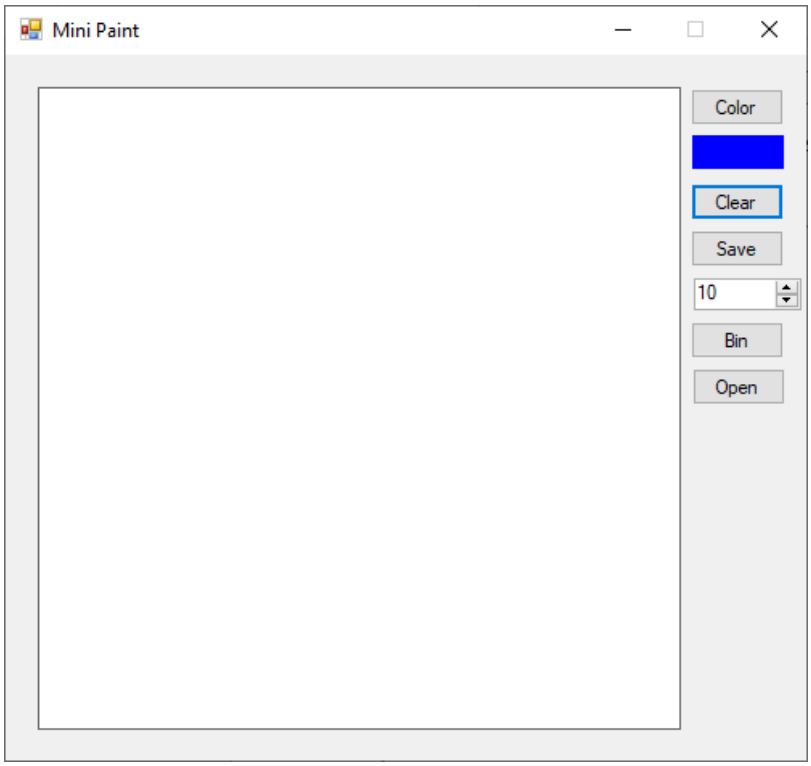
User needs "Color" button to select the pencil color. It is not mandatory to use as the default color is black. The "Clear" button clears the sheet. The purpose of the "Save" button is to bring up a dialog box in which it selects the file saving location, file extension and name. Relevant file extensions include BMP, JPG, and PNG. He can also enter a different type of the file itself, but must put a period and characteristic characters after the name. In case you are interested in resizing the pencil, the NumericUpDown Button control, ranging from 0 to 100, will help you. The "Bin" button is a typical eraser, where pressing it changes the color of the pencil from the selected color to white, and when pressed again, it returns to the previous color. If you want to open an image in an application, that's no problem. Just click the "Open" button, select the file location and the file itself.



Drawing 2: Pencil color selection window [own study]

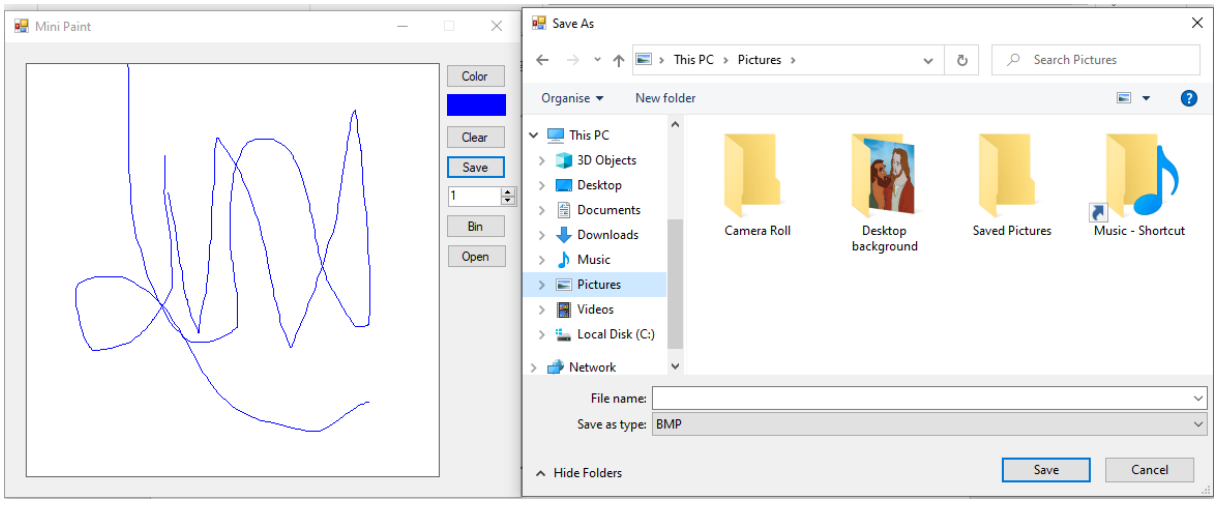


Drawing 3: Selected blue color [own study]

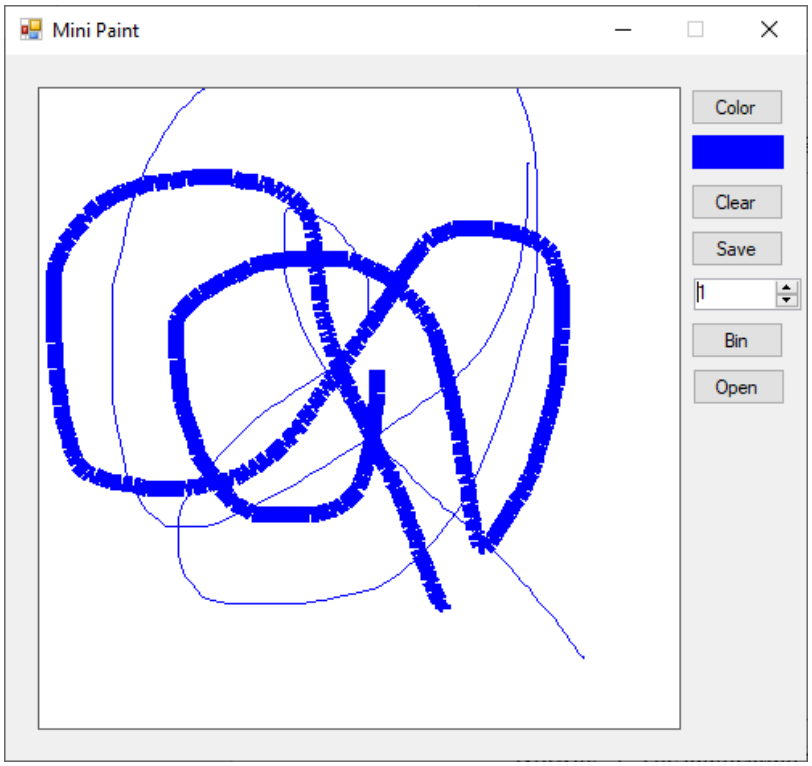


Drawing 4: A cleared window [own study]

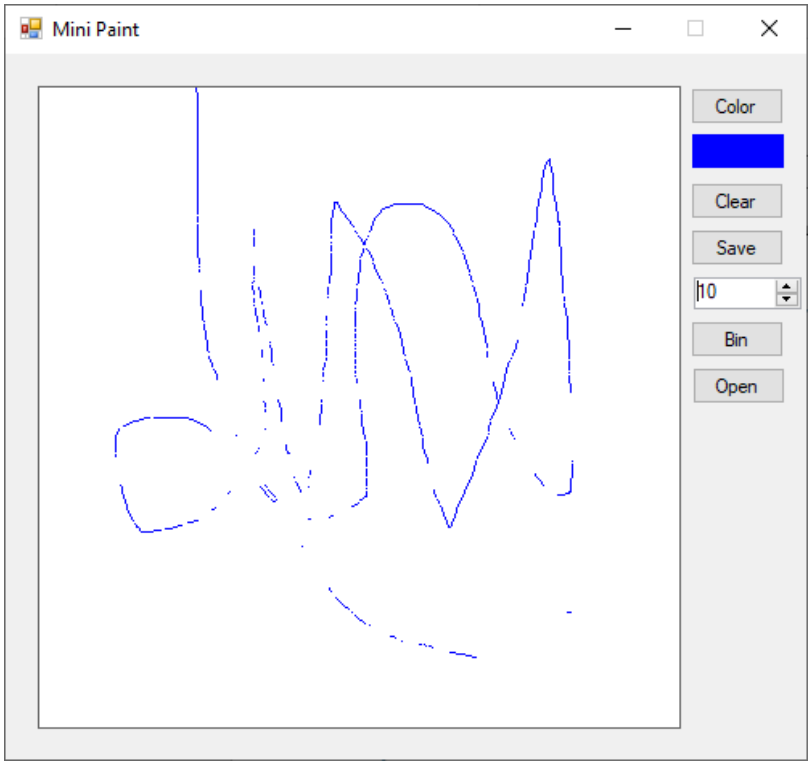
The color remains the same after cleaning the page!



Drawing 5: Save Image Dialog Box (Pictures folder by default) [own study]

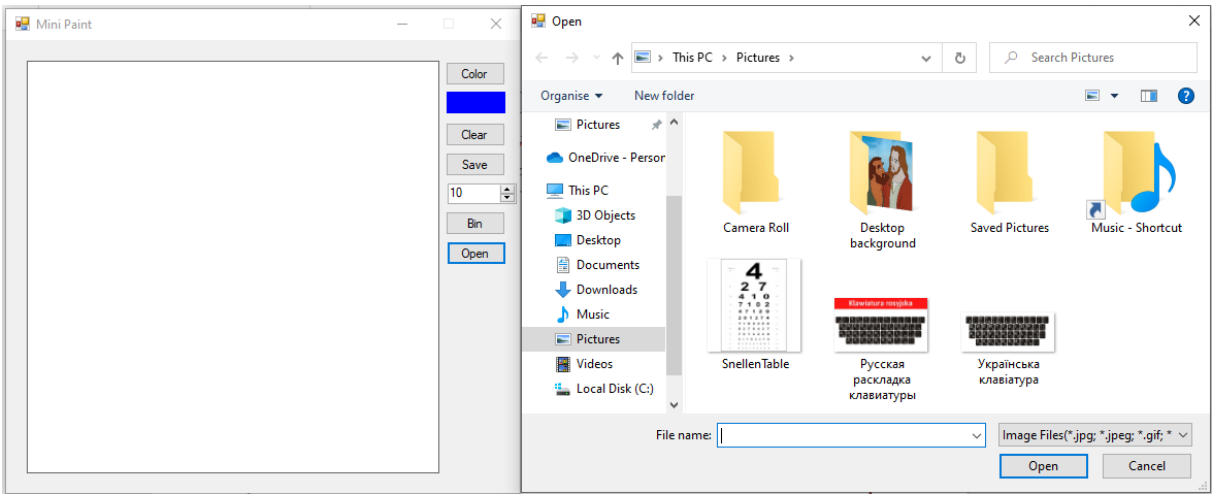


Drawing 6: Change pencil size [own study]

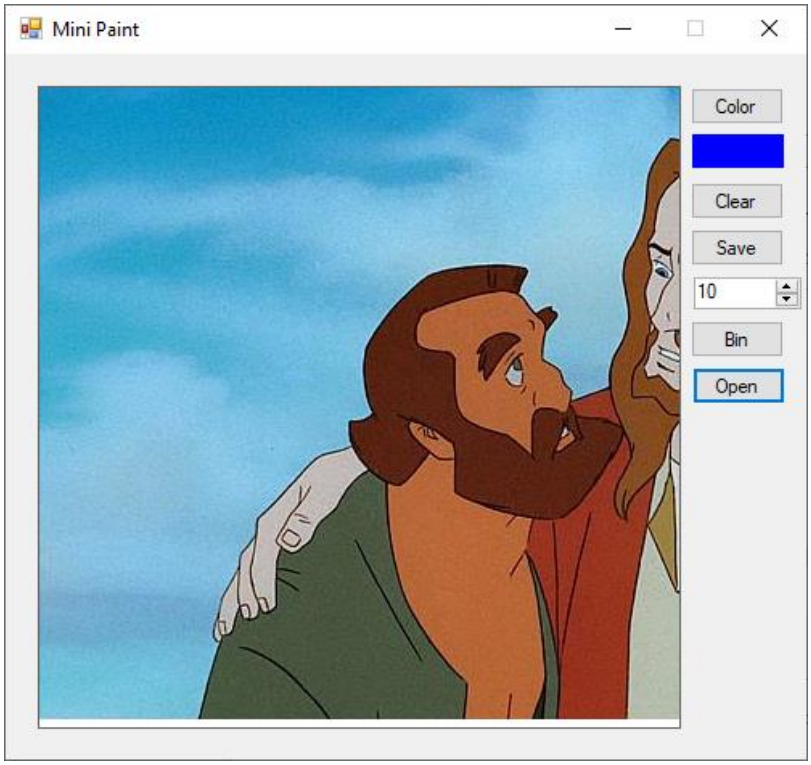


Drawing 7: Gumming work [own study]

Rubberizing the work has the ability to change the size of the eraser!



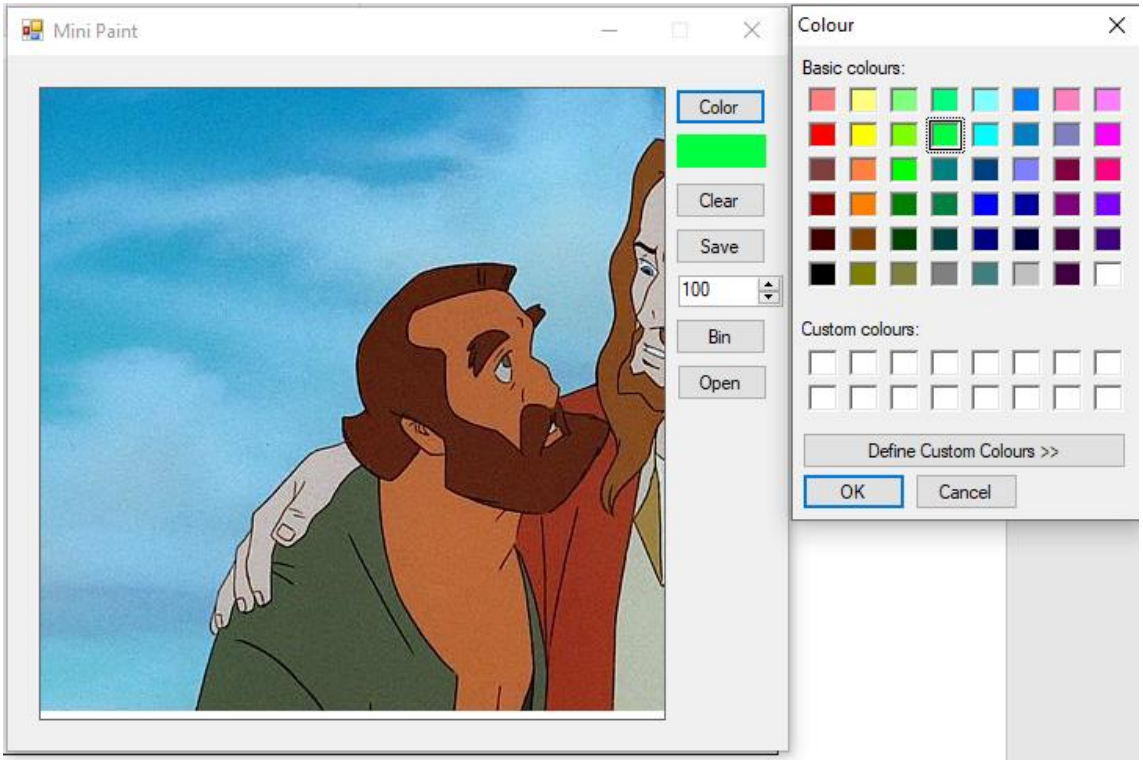
Drawing 8: Opening image dialog [own study]



Drawing 9: The opened image [own study]

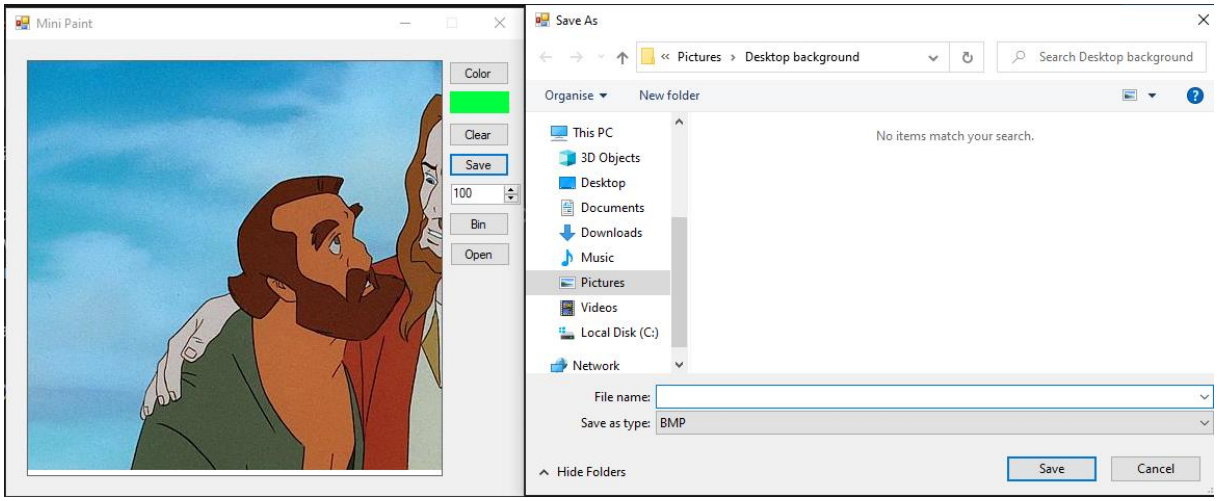
This is the correct use of the program. There may be coding errors that have not been detected by the developer.

Certainly, the drawback of the program is that when the image is opened, the window cleaning function is blocked, the use of an eraser, the effect of changing the pencil size and color is not visible. The other options are kept. Another problem is adapting the card to the size of the image to be opened, in case the user wants to run the image. Please note that the program is not an image editor! You cannot sketch after opening the image!

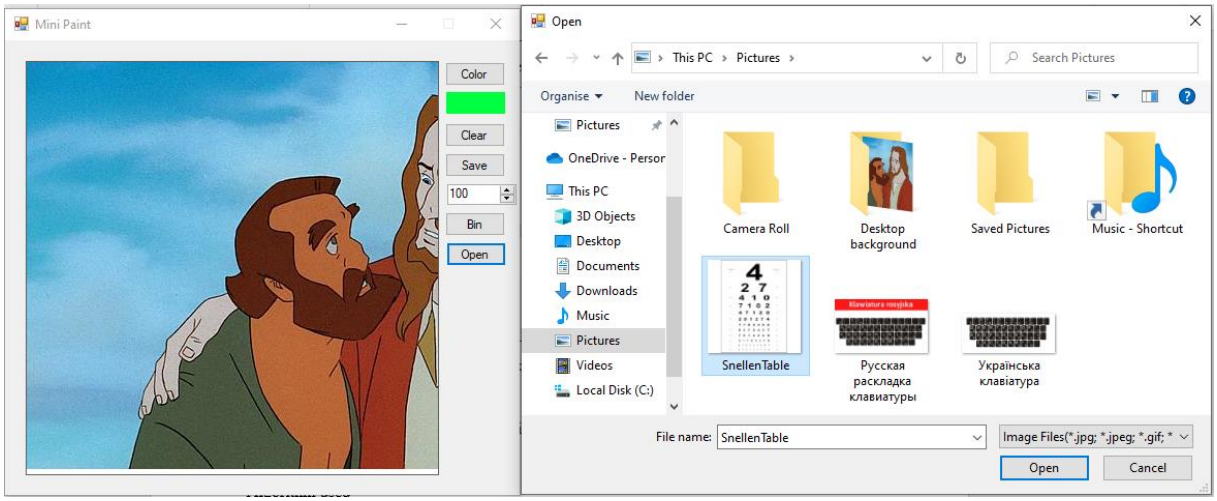


Drawing 10: No chance to preview color change [own study]

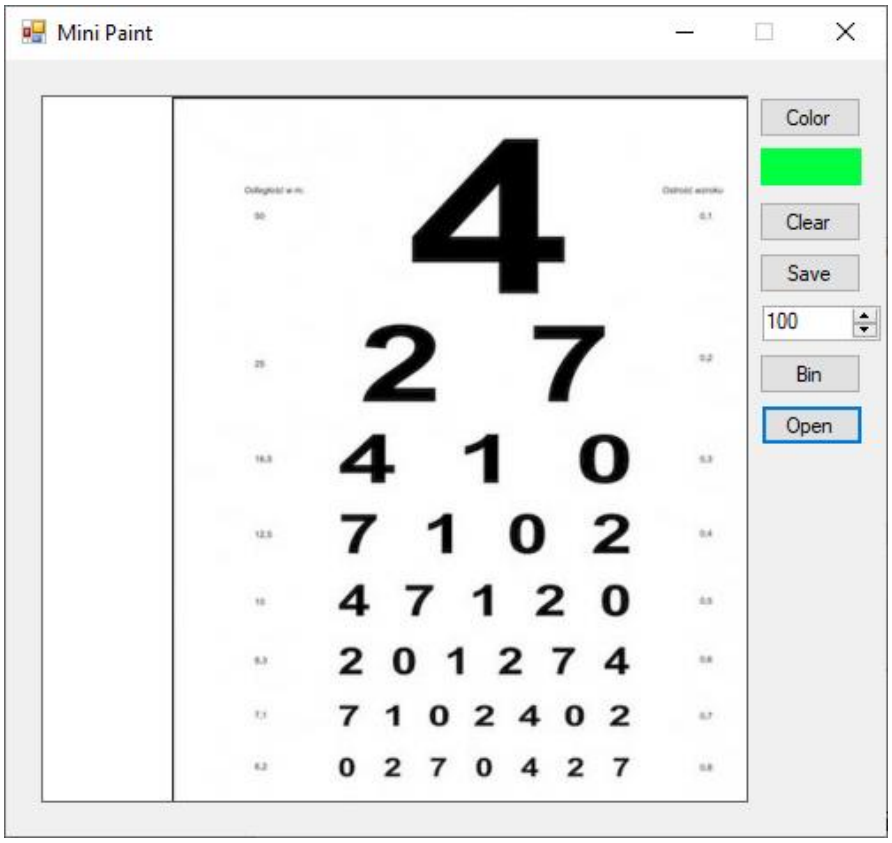
Unable to check the color change, but it is possible to open the dialog and display the new one in the preview.



Drawing 11: A chance to save the image [own study]



Drawing 12: A chance to open another image [own study]



Drawing 13: Opened the another picture [own study]

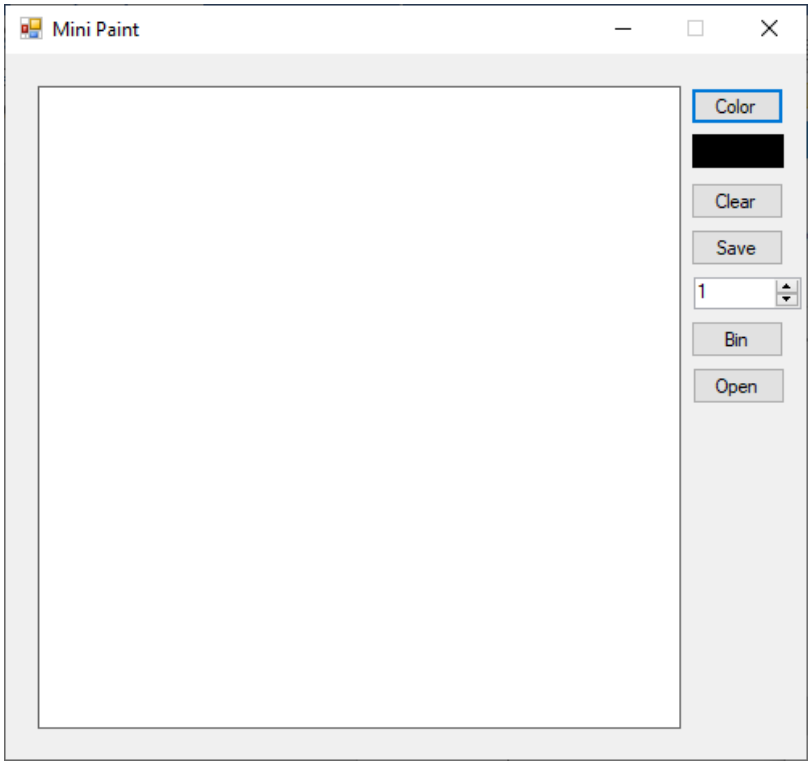
What is needed for use?

The application does not require installation. It only needs the Windows operating system.

Algorithm used

The basic form of the algorithm can be deduced from the previous section. All in all, the app is a typical sketchbook and image viewer. It is not an existing image editor! If the user wants to sketch, he has to restart the program.

Interface description



Drawing 14: Graphical interface [own study]

The interface is typical for a Windows Forms Application. There are essential components: buttons, panel (selected color view), panel (work page) and numericupdown.

Source code description

The project was made in the C# programming language, in the Visual Studio Community 2017 programming environment. All work was done on the Windows 10 operating system. The application's source code looks like this.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace MiniPaint
{
    public partial class frmMiniPaint : Form
    {
        private Graphics g;
        private Point p = Point.Empty;
        private Pen pioro;
        private bool gumka = false;
```

```
public frmMiniPaint()
{
    InitializeComponent();
    imgObrazek.Image = new Bitmap(400, 400);
    g = Graphics.FromImage(imgObrazek.Image);
    nudSize.Value = 1;
    pioro = new Pen(Color.Black, int.Parse(nudSize.Value.ToString()));
}

private void cmdKolor_Click(object sender, EventArgs e)
{
    ColorDialog dialog = new ColorDialog();
    dialog.Color = lblKolor.BackColor;
    dialog.ShowDialog();
    lblKolor.BackColor = dialog.Color;
    pioro.Color = dialog.Color;
}

private void cmdCzysc_Click(object sender, EventArgs e)
{
    g.Clear(Color.White);
    imgObrazek.Refresh();
}

private void cmdZapisz_Click(object sender, EventArgs e)
{
    SaveFileDialog dialog = new SaveFileDialog();
    dialog.Filter = "BMP|*.bmp| JPG|*.jpg| PNG|*.png|All files|*.*";
    dialog.ShowDialog();
    if (dialog.FileName != "") imgObrazek.Image.Save(dialog.FileName);
}

private void imgObrazek_MouseDown(object sender, MouseEventArgs e)
{
    if (e.Button == MouseButtons.Left) p = e.Location;
}

private void imgObrazek_MouseMove(object sender, MouseEventArgs e)
{
    if (e.Button == MouseButtons.Left)
    {
        g.DrawLine(pioro, p, e.Location);
        p = e.Location;
        imgObrazek.Refresh();
    }
}

private void nudSize_ValueChanged(object sender, EventArgs e)
{
    try { pioro.Width = Convert.ToInt32(nudSize.Value.ToString()); }
    catch (Exception) { /*Do nothing*/}
}

private void cmdBin_Click(object sender, EventArgs e)
{
    if (gumka == false)
    {
        gumka = true;
        pioro.Color = Color.White;
    }
    else
    {
```

```
                gumka = false;
                pioro.Color = lblKolor.BackColor;
            }
        }

        private void cmdOtworz_Click(object sender, EventArgs e)
        {
            try
            {
                OpenFileDialog open = new OpenFileDialog();
                open.Filter = "Image Files(*.jpg; *.jpeg; *.gif; *.bmp)|*.jpg; *.jpeg;
*.gif; *.bmp; *.*";
                if (open.ShowDialog() == DialogResult.OK)
                {
                    Bitmap bmp = new Bitmap(open.FileName);
                    imgObrazek.Image = bmp;
                }
            }
            catch (Exception)
            {
                throw new ApplicationException("Failed loading image");
            }
        }
    }
}
```

Listing 1: Source code [own study]

List of drawings

List of listings