Urzadzenia peryferyjne
Obsługa skanera płaskiego

Autorka: Daria Jeżowska, 252731
Prowadzacy: dr inż. Jarosław Mierzwa

1 Zadania do wykonania

- Sprawdź czy skaner działa poprawnie
- Napisać program wykonujacy skanowanie przy pomocy skanera płaskiego (WIA). Możliwości programu obejmuja:
  - skanowanie z wykorzystaniem UI
  - skanowanie bez wykorzystania UI
  - wyświetlanie uzyskanego obrazu
  - zmiana rozdzielczości skanera
  - zmiana trybu skanowania (1-bitowy, skala szarości, RGB)
- Rozszerzyć działanie programu:
  - obsługa różnych trybów przesyłania danych
  - zapis skanowanych obrazów do plików graficznych

2 Wstep

Skaner jest urzadzeniem peryferyjnym służacym do przetwarzania obrazu rzeczywistego do formy cyfrowej, która jest zrozumiała dla komputera. Skaner, w odróżnieniu od aparatu, nie rejestruje całego obrazu na raz, lecz odczytuje kolejne linie obrazu i je rejestruje. Podczas skanowania pod szyba przesuwa sie zespół lampa-lustro. Lampa oświetla obraz, a za pomoca systemu luster i soczewek trafia do elementów światłoczułych w skanerze. Korzystaja one z technologii CCD (Charge Coupled Device) - składajacej sie z elementów światłoczułych, które sa rejestrowane i nastepnie możliwe do odczytania.

3 Przebieg programu

Program został napisany w jezyku C# za pomoca biblioteki służacej do komunikacji ze skanerem - Windows Image Acquisition. Najpierw został stworzony obiekt *DeviceManager*, który odpowiada za znajdowanie urzadzeń, a także pozyskiwanie obrazu oraz *DeviceInfo*, który odpowiada za połaczenie sie z wybranym urzadzeniem. Ważnym elementem jest także *ImageFile*, który służy do zapisywania obrazu. W *public Form1()* ustawiane sa parametry track barów służacych do ustawiania rozdzielczości, jasności i kontrastu.Do comboxa odpowiadajacego za wybór skanowania (RGB, 1-bitowe, odcienie szarości) przypisywane sa możliwe tryby. W petli foreach w kolejnym comboboxie wypisywane sa dostepne urzadzenia.

```
public partial class Form1 : Form
    {
        DeviceManager deviceManager = new DeviceManager();
        DeviceInfo deviceInfo;
        ImageFile imageFile = new ImageFile();

        public Form1()
        {
            InitializeComponent();
            ResolutionTrackBar.Minimum = 0;
            ResolutionTrackBar.Maximum = 600;
            ResolutionTrackBar.TickFrequency = 150;
            brightnessTrackBar.Minimum = -100;
            contrastTrackBar.Minimum = -100;
            brightnessTrackBar.Maximum = 100;
            contrastTrackBar.Maximum = 100;
            ResolutionTrackBar.Value = 150;
            resolution.Text = ResolutionTrackBar.Value.ToString();
            contrast.Text = contrastTrackBar.Value.ToString();
            brightness.Text = brightnessTrackBar.Value.ToString();

            colorBox.Items.Add("RGB");
            colorBox.Items.Add("Grey␣scale");
            colorBox.Items.Add("1-bit");

            foreach (DeviceInfo info in deviceManager.DeviceInfos)
            {
                comboBoxDevice.Items.Add(info.Properties["Name"].get_Value());
            }
            colorBox.SelectedIndex = 0;
            comboBoxDevice.SelectedIndex = 0;
        }
```

scanButton_Click odpowiada za działanie programu po kliknieciu przycisku *Scan* - rozpoczyna skanowanie. Jako deviceInfo ustawiane jest wybrane w comboboxie urzadzenie, nastepnie tworzone jest z nim połaczenie i ustawianie wartości wybranych w comboboxie odpowiadajacym za kolorystke skany oraz rozdzielczość, które sa nastepnie przekazywane do funkcji *settings*, która ustawia je za pomoca funkcji *SetWIAProperty*. Nastepnie skanowany jest obraz i zapamietywany jako bitmapa.

```
private void scanButton_Click(object sender, EventArgs e)
    {
        deviceInfo = deviceManager.DeviceInfos[comboBoxDevice.SelectedIndex+1];
        var connectedDevice = deviceInfo.Connect();
        var scannerItem = connectedDevice.Items[1];
        var res = ResolutionTrackBar.Value;
        int color = 0;
        if (colorBox.SelectedIndex== 1)
        {
            color = 2;
        }
        if (colorBox.SelectedIndex == 2)
        {
            color = 4;
        }
        settings(scannerItem, res, 0, 0, 1250 * (res / 150), 1700 * (res / 150),
                 brightnessTrackBar.Value, contrastTrackBar.Value, color);
        imageFile = (ImageFile)scannerItem.Transfer();

        byte[] imageBytes = (byte[])imageFile.FileData.get_BinaryData();
        MemoryStream savedTmp = new MemoryStream(imageBytes);
        Image bitmap = Image.FromStream(savedTmp);
        scanPictureBox.Image = bitmap;
    }
```

saveButton_Click odpowiada za zapisanie skanu w wybranym przez użytkownika formacie i kliknieciu przycisku "Scan". Po stworzeniu obiektu saveFileDialog i ustawieiu formatów w których użytkownik bedzie mógł zapisać wywołuje sie metoda *ShowDialog()*, która wyświetla okno zpaisywania, a nastepnie zapisywany jest skan.

```
private void saveButton_Click(object sender, EventArgs e)
{
    SaveFileDialog saveFileDialog = new SaveFileDialog();
    saveFileDialog.Filter = " .BMP| *.BMP | *.JPG | *.JPG | *.GIF | *.GIF |
                            *.PNG | *.PNG | *.TIFF | *.TIFF ";
    saveFileDialog.ShowDialog();
    imageFile.SaveFile(saveFileDialog.FileName);
}
```

W kolejnej cześci kodu ustawiane sa wyświetlane obok trackbarów wartości dla etykiet *resolution, contrast* oraz *brightness*.

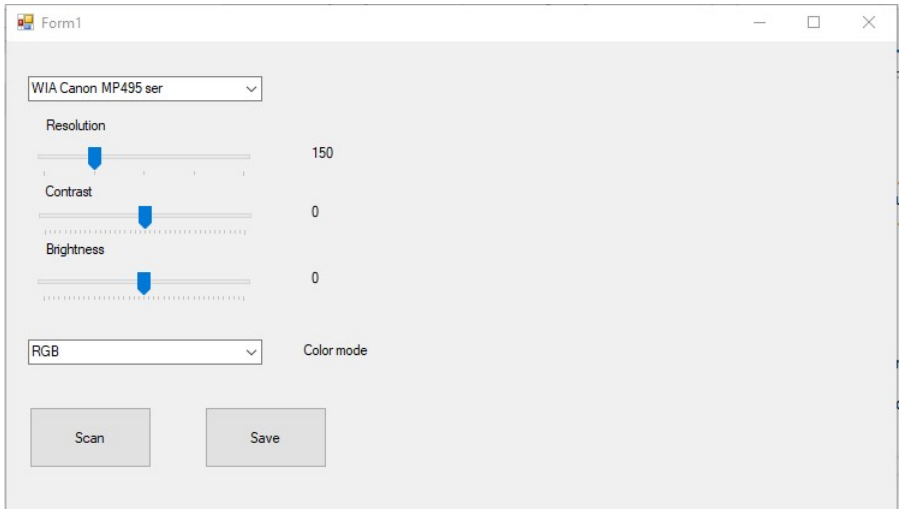
```
private void ResolutionTrackBar_Scroll(object sender, EventArgs e)
{
    resolution.Text = ResolutionTrackBar.Value.ToString();
}
private void trackBar1_Scroll(object sender, EventArgs e)
{
    contrast.Text = contrastTrackBar.Value.ToString();
}
private void trackBar2_Scroll(object sender, EventArgs e)
{
    brightness.Text = brightnessTrackBar.Value.ToString();
}
```

Funkcja *settings* odpowiada za ustawienie odpowiednich wartości odpowiednim zmiennym oraz za wywołanie funkcji, która ustawia parametry skanowania - w naszym przypadku wybieralnymi przez użytkownika sa rozdzielczość, kontrast i jasność. *SetWIAProperty* ustawia już właściwe wartości jako parametry skanowania.

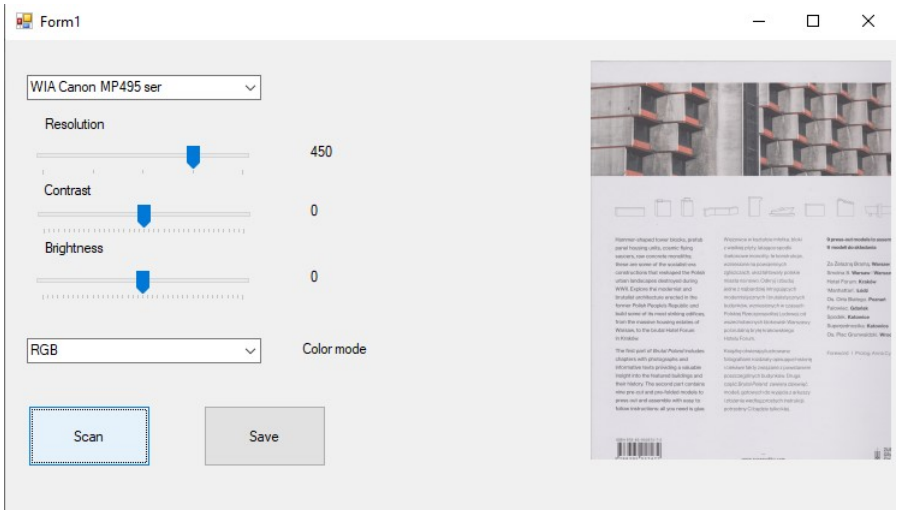
```
private static void settings(IItem scannnerItem, int scanResolutionDPI, int scanStartLeftPixel,
                             int scanStartTopPixel, int scanWidthPixels, int scanHeightPixels,
                             int brightnessPercents, int contrastPercents, int colorMode)
{
    const string WIA_SCAN_COLOR_MODE = "6146";
    const string WIA_HORIZONTAL_SCAN_RESOLUTION_DPI = "6147";
    const string WIA_VERTICAL_SCAN_RESOLUTION_DPI = "6148";
    const string WIA_HORIZONTAL_SCAN_START_PIXEL = "6149";
    const string WIA_VERTICAL_SCAN_START_PIXEL = "6150";
    const string WIA_HORIZONTAL_SCAN_SIZE_PIXELS = "6151";
    const string WIA_VERTICAL_SCAN_SIZE_PIXELS = "6152";
    const string WIA_SCAN_BRIGHTNESS_PERCENTS = "6154";
    const string WIA_SCAN_CONTRAST_PERCENTS = "6155";
    SetWIAProperty(scannnerItem.Properties, WIA_HORIZONTAL_SCAN_RESOLUTION_DPI, scanResolutionDPI);
    SetWIAProperty(scannnerItem.Properties, WIA_VERTICAL_SCAN_RESOLUTION_DPI, scanResolutionDPI);
    SetWIAProperty(scannnerItem.Properties, WIA_HORIZONTAL_SCAN_START_PIXEL, scanStartLeftPixel);
    SetWIAProperty(scannnerItem.Properties, WIA_VERTICAL_SCAN_START_PIXEL, scanStartTopPixel);
    SetWIAProperty(scannnerItem.Properties, WIA_HORIZONTAL_SCAN_SIZE_PIXELS, scanWidthPixels);
    SetWIAProperty(scannnerItem.Properties, WIA_VERTICAL_SCAN_SIZE_PIXELS, scanHeightPixels);
    SetWIAProperty(scannnerItem.Properties, WIA_SCAN_BRIGHTNESS_PERCENTS, brightnessPercents);
    SetWIAProperty(scannnerItem.Properties, WIA_SCAN_CONTRAST_PERCENTS, contrastPercents);
    SetWIAProperty(scannnerItem.Properties, WIA_SCAN_COLOR_MODE, colorMode);
}

private static void SetWIAProperty(IProperties properties, object propName, object propValue)
{
    Property prop = properties.get_Item(ref propName);
    prop.set_Value(ref propValue);
}
```

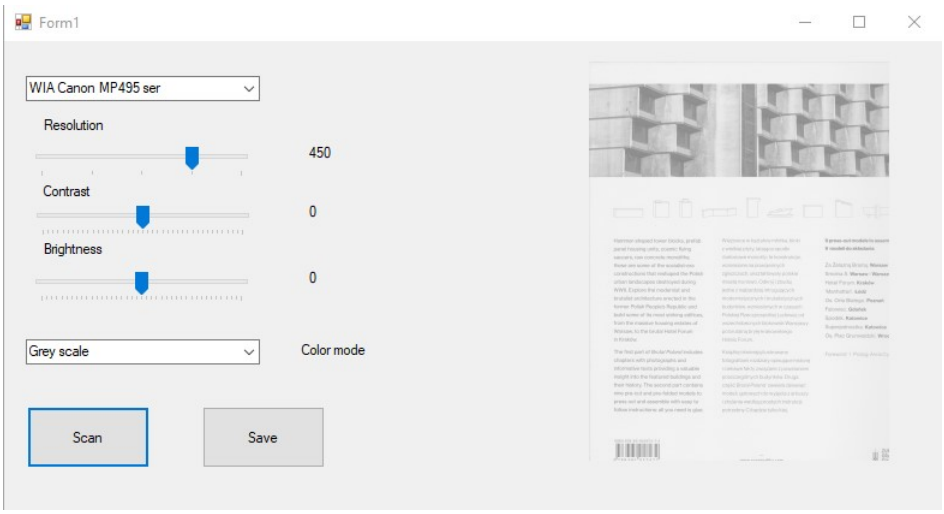
4 Wyglad aplikacji



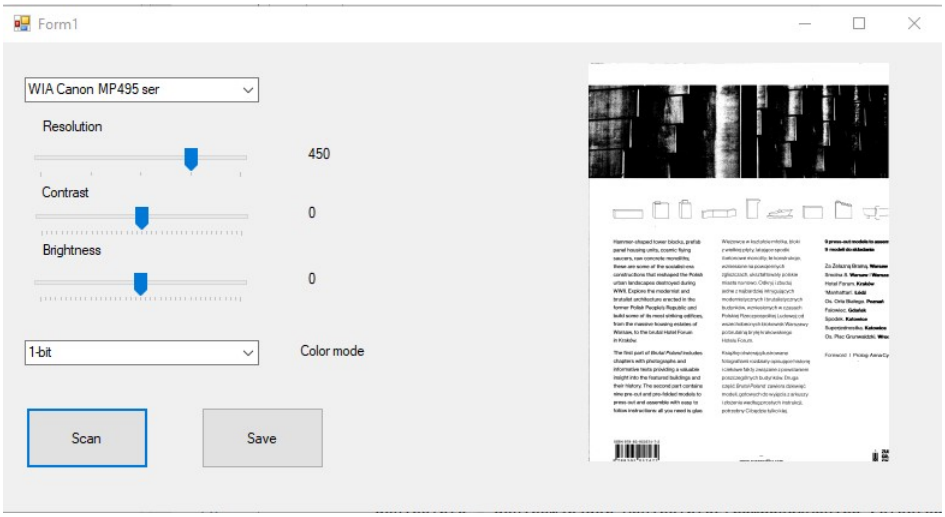
Wyglad aplikacji



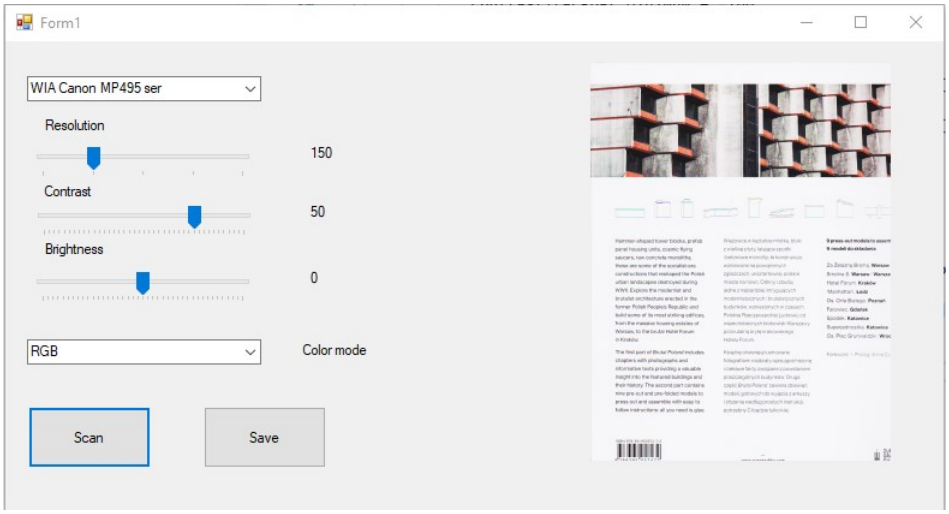
Skanowanie obrazu w RBG



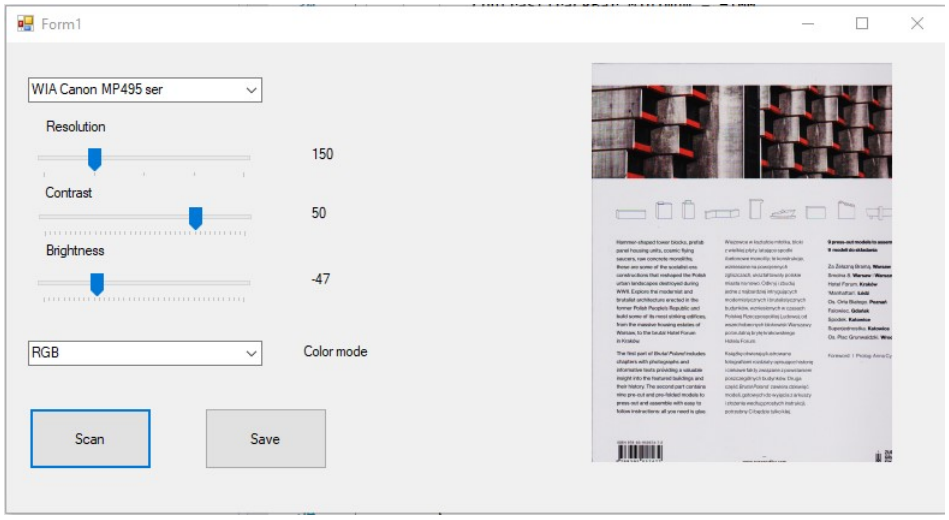
Skanowanie obrazu w odcieniach szarości



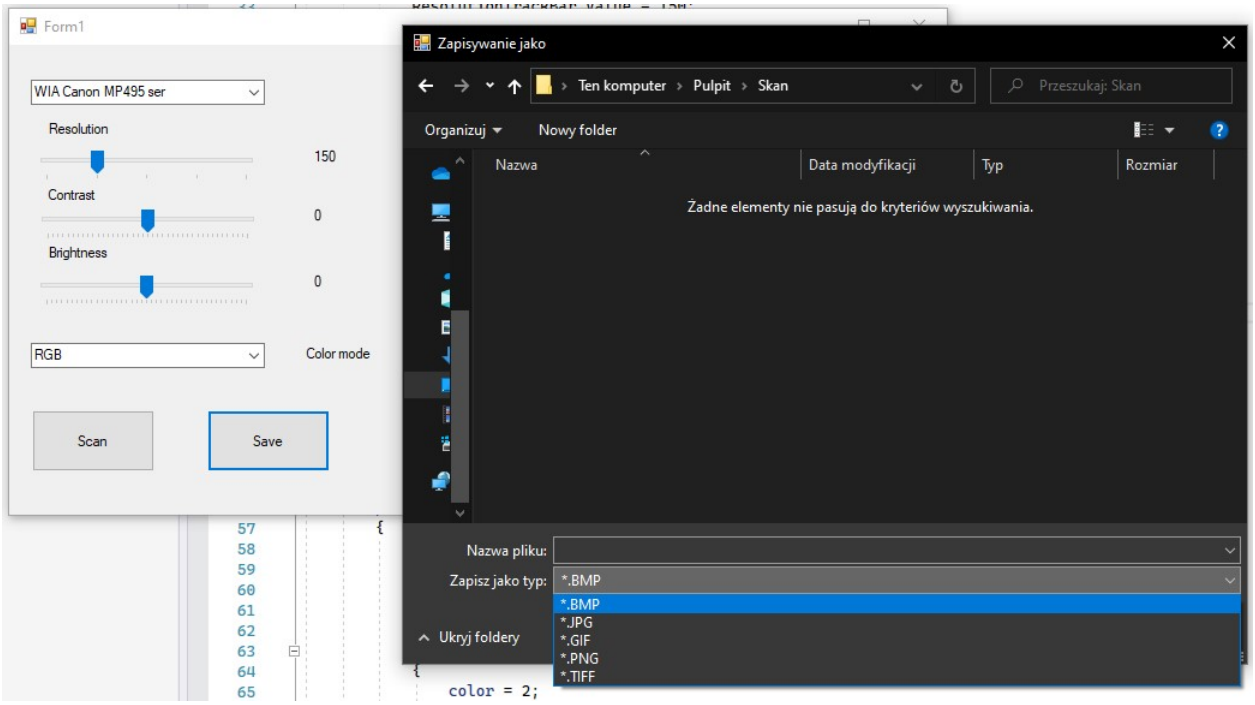
Skanowanie obrazu w 1-bit



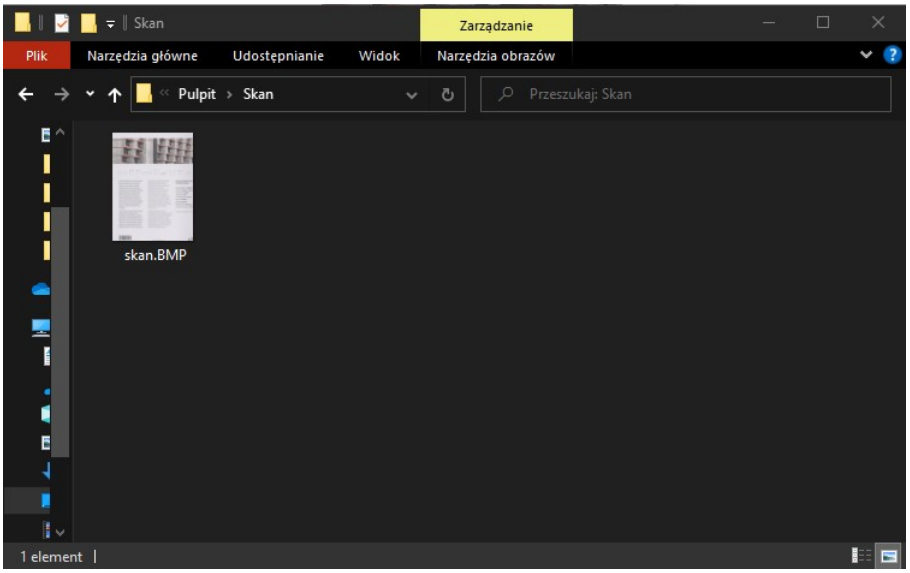
Zwiekszenie kontrastu



Zmniejszenie jasności



Zapis do pliku



Zapisany plik jest widoczny w folderze

5 Wnioski

Zadanie to wymagało wcześniejszego przygotowania i zapoznania sie z biblioteka WIA i jej możliwościami. Musieliśmy wiedzieć, jakie funkcje i obiekty beda przydatnce do przetwarzania obrazu, do łaczenia sie z urzadzeniem i do samego skanowania. Najwieksza napotkana trudnościa było znalezienie odpowiednich informacji o danych funkcjach i obiektach biblioteki WIA ze wzgledu na jej wiek i nie zawsze intuicyjna dokumentacje.