

Kurs:  
Urządzenia Peryferyjne - Laboratorium

---

# SKANER PŁASKI. WPROWADZANIE INFORMACJI DO KOMPUTERA

---

*Autor:*

ALEKSANDRA CHRUSTEK, 263900

JONASZ TROCHA, 263951

Prowadzący:  
dr inż. Dariusz Caban

13.12.2023

# 1 Wstęp

Celem ćwiczenia było zapoznanie się z zasadami działania skanera płaskiego oraz napisanie oprogramowania do zapisywania obrazu zeskanowanego.

## 2 Informacje teoretyczne

### 2.1 Budowa skanera (CCD, CIS, LIDE)

#### 2.1.1 CCD (Charge-Coupled Device)

CCD to rodzaj matrycy światłoczułej złożonej z pikseli, gdzie każdy piksel rejestruje ilość światła. Światło przechodzi przez soczewki i trafia na matrycę CCD, gdzie generowane są sygnały elektryczne proporcjonalne do ilości padającego światła.

#### 2.1.2 CIS (Contact Image Sensor)

CIS to układ złożony z diod świetlnych i detektorów zamontowanych bezpośrednio na powierzchni skanera. Światło odbite od dokumentu trafia bezpośrednio na detektory, generując sygnały elektryczne.

#### 2.1.3 LIDE (LED Indirect Exposure)

LIDE wykorzystuje diody LED do oświetlania dokumentu. Światło LED jest kierowane na powierzchnię dokumentu, a odbite światło jest rejestrowane przez detektory. obsługą dźwięku.

### 2.2 Kolory cyfrowe

#### 2.2.1 Filtry

Stosowane do eliminowania niepożądanych efektów, np. efektu czerwonych oczu.

#### 2.2.2 Balans Bieli

Dostosowuje kolorystykę obrazu, usuwając dominację barw.

#### 2.2.3 Rozdzielczość skanowania

Określa ilość detali, jakie skaner może zarejestrować. Wyrażana w dpi (dots per inch).

#### 2.2.4 De-mozaikowanie

Proces eliminowania efektu mozaiki, który występuje w obrazach cyfrowych zapisanych w formacie Bayera.

### **2.2.5 Zoom**

Zmiana skali obrazu, niekoniecznie związana z fizycznym zbliżaniem optycznym.

## **2.3 Biblioteki wspierające programowanie skanerów**

### **2.3.1 TWAIN (Technology Without An Interesting Name)**

Standard umożliwiający komunikację między oprogramowaniem a urządzeniami do akwizycji obrazu.

### **2.3.2 WIA (Windows Image Acquisition)**

Zapewnia interfejs dla komunikacji między urządzeniem a systemem operacyjnym Windows.

### **2.3.3 SANE (Scanner Access Now Easy)**

Otwarte oprogramowanie zapewniające dostęp do skanerów na platformach Unix i Linux.

### **2.3.4 ISIS (Image and Scanner Interface Specification)**

Standard w przemyśle skanowania, umożliwiający komunikację z różnymi urządzeniami do akwizycji obrazu.

## **2.4 Formaty zapisu informacji graficznej**

### **2.4.1 JPG (Joint Photographic Experts Group)**

Format kompresji stratnej, efektywny dla fotografii.

### **2.4.2 PNG (Portable Network Graphics)**

Format bezstratnej kompresji, wspierający przezroczystość.

### **2.4.3 TIFF (Tagged Image File Format)**

Format umożliwiający przechowywanie obrazów z zachowaniem dużej jakości i informacji o metadanych.

### **2.4.4 BMP (Bitmap)**

Format przechowujący obrazy w postaci mapy bitowej, bez kompresji.

### **2.4.5 RLE (Run-Length Encoding)**

Metoda kompresji danych, stosowana m.in. w formacie BMP.

### 3 Realizacja zadania

Aplikacja jest zaprojektowana do obsługi skanera płaskiego. Należało napisać aplikację umożliwiającą uzyskanie obrazu na ekranie monitora po zeskanowaniu obiektu, oraz zapisanie obrazu do zbioru z możliwością wybrania jednego z kilku podanych formatów, realizację takich opcji jak zmiana parametrów skanowania (rozdzielczość, tryb skanowania), obracanie obrazu o 90, 180, 270 stopni w obie strony. W tych celach użyto biblioteki WIA (Windows Image Acquisition).

Poniżej przedstawione zostały poszczególne metody do obsługi skanera płaskiego:

1. Metoda `Form1_Load(object sender, EventArgs e)`: Inicjalizuje formularz po jego załadowaniu. Wywołuje metodę `ListScanners()` do wylistowania dostępnych skanerów. Ustawia domyślną ścieżkę wyjściową na folder tymczasowy. Ustawia domyślny format pliku na JPEG.

```
private void Form1_Load(object sender, EventArgs e)
{
    ListScanners();

    // Set start output folder TMP
    textBox1.Text = Path.GetTempPath();
    // Set JPEG as default
    comboBox1.SelectedIndex = 1;
}
```

2. Metoda `ListScanners()`: Czyści zawartość kontrolki `listBox1`. Tworzy instancję `DeviceManager`. Przechodzi przez listę urządzeń i dodaje do listy skanerów (tylko urządzenia skanujące).

```
private void ListScanners()
{
    // Clear the ListBox.
    listBox1.Items.Clear();

    // Create a DeviceManager instance
    var deviceManager = new DeviceManager();

    // Loop through the list of devices and add the name to the listbox
    for (int i = 1; i <= deviceManager.DeviceInfos.Count; i++)
```

```

{
    // Add the device only if it's a scanner
    if (deviceManager.DeviceInfos[i].Type !=
        WiaDeviceType.ScannerDeviceType)
    {
        continue;
    }

    // Add the Scanner device to the listbox
    (the entire DeviceInfos object)
    // Important: we store an object of type scanner
    (which ToString method returns the name of the scanner)
    listBox1.Items.Add(
        new Scanner(deviceManager.DeviceInfos[i])
    );
}
}

```

3. Metoda `button1_Click(object sender, EventArgs e)`: Rozpoczyna asynchroniczne zadanie skanowania przez wywołanie `Task.Factory.StartNew(StartScanning)`. Po zakończeniu skanowania, wywołuje metodę `TriggerScan()`.

```

private void button1_Click(object sender, EventArgs e)
{
    Task.Factory.StartNew(StartScanning).ContinueWith(result => TriggerScan());
}

```

4. Metoda `TriggerScan()`: Wyświetla komunikat o pomyślnym zeskanowaniu obrazu na konsoli.

```

private void TriggerScan()
{
    Console.WriteLine("Image succesfully scanned");
}

```

5. Metoda `StartScanning()`: Wybiera zaznaczony skaner z listy. W zależności od wybranych opcji (format, rozmiar i tryb koloru), ustawia parametry skanowania. Wybiera odpowiedni format pliku na podstawie wyboru użytkownika. Skanuje obraz przy użyciu metody `ScanImage` obiektu skanera. Zapisuje zeskanowany obraz w określonym formacie i ścieżce. Wyświetla zeskanowany obraz w kontrolce `pictureBox1`.

```

public void StartScanning()
{
    Scanner device = null;

    this.Invoke(new MethodInvoker(delegate ()
    {
        device = listBox1.SelectedItem as Scanner;
    }));

    if (radioButton1.Checked)
    {
        System.Console.WriteLine("A4 checked");
        device.width_pixel = 2500;
        device.height_pixel = 3400;
    }
    else if (radioButton2.Checked)
    {
        System.Console.WriteLine("A5 checked");
        device.width_pixel = 1600;
        device.height_pixel = 2350;
    }

    if (radioButton3.Checked)
    {
        System.Console.WriteLine("Kolor");
        device.color_mode = 1;
    }
    else if (radioButton4.Checked)
    {
        System.Console.WriteLine("Czerń");
        device.color_mode = 2;
    }

    if (device == null)
    {
        MessageBox.Show("You need to select first
            an scanner device from the list",
            "Warning",
            MessageBoxButtons.OK, MessageBoxIcon.Warning);
    }
}

```

```

        return;
    }else if(String.IsNullOrEmpty(textBox2.Text))
    {
        MessageBox.Show("Provide a filename",
                        "Warning",
                        MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }

    ImageFile image = new ImageFile();
    string imageExtension = "";

    this.Invoke(new MethodInvoker(delegate ()
    {
        switch (comboBox1.SelectedIndex)
        {
            case 0:
                image = device.ScanImage(WIA.FormatID.wiaFormatPNG);
                imageExtension = ".png";
                break;
            case 1:
                image = device.ScanImage(WIA.FormatID.wiaFormatJPEG);
                imageExtension = ".jpeg";
                break;
            case 2:
                image = device.ScanImage(WIA.FormatID.wiaFormatBMP);
                imageExtension = ".bmp";
                break;
            case 3:
                image = device.ScanImage(WIA.FormatID.wiaFormatGIF);
                imageExtension = ".gif";
                break;
            case 4:
                image = device.ScanImage(WIA.FormatID.wiaFormatTIFF);
                imageExtension = ".tiff";
                break;
        }
    }));

```

```

// Save the image
var path = Path.Combine(textBox1.Text, textBox2.Text + imageExtension);

if (File.Exists(path))
{
    File.Delete(path);
}

image.SaveFile(path);
pictureBox1.Image = new Bitmap(path);
}

```

6. Metoda `button2_Click(object sender, EventArgs e)`: Wywołuje okno dialogowe wyboru folderu. Ustawia ścieżkę folderu wybranego przez użytkownika w kontrolce `textBox1`.

```

private void button2_Click(object sender, EventArgs e)
{
    FolderBrowserDialog folderDlg = new FolderBrowserDialog();
    folderDlg.ShowNewFolderButton = true;
    DialogResult result = folderDlg.ShowDialog();

    if (result == DialogResult.OK)
    {
        textBox1.Text = folderDlg.SelectedPath;
    }
}

```

7. Obsługa zdarzeń `radioButton1_CheckedChanged`, `radioButton2_CheckedChanged`, `radioButton3_CheckedChanged`, `radioButton4_CheckedChanged`: Obsługuje zmiany stanu zaznaczenia radiobuttonów.

## 4 Wnioski

Zadanie miało na celu wprowadzenie w obszar programowania urządzeń peryferyjnych, konkretnie skanera płaskiego. Realizacja tego zadania okazała się wymagająca, ponieważ wiązała się z koniecznością zaznajomienia się z zasadami zmieniania rozdzielczości, kolorystyki oraz obracania obrazu oraz zrozumieniem, jak obsługa skanera płaskiego wygląda z perspektywy zaimplementowanej aplikacji. W konsekwencji, musieliśmy znaleźć sposoby na efektywne skanowanie oczekiwanego formatu obrazu z oczekiwanymi parametrami i przełożenie tych sposobów na konkretne działania na poziomie aplikacji. W



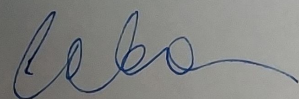
trakcie laboratorium udało się zaimplementować wszystkie wymagane przez instrukcję metody, z wyjątkiem obracania obrazu. Aplikacja umożliwia skanowanie dokumentów z różnymi parametrami. Wykorzystuje Windows Image Acquisition (WIA) do komunikacji z urządzeniem do skanowania. Obsługuje różne formaty obrazów, a także pozwala na wybór rozdzielczości i trybu kolorów. Umożliwia użytkownikowi wybór skanera z listy dostępnych urządzeń. Skanowane obrazy są zapisywane w wybranym formacie i lokalizacji. W rezultacie, zadanie dostarczyło nam praktycznej wiedzy na temat programowania urządzeń peryferyjnych, a zdobyte umiejętności mogą być użyteczne w projektach obejmujących obsługę różnorodnych urządzeń wejściowych.

Aleksandra Chmielek,  
Jonasz Trocha

13.12.2023

### Lab 5 - Skaner plaski

1. Założyliśmy od napisanie opłiki wykrywającej urządzenie do skanowania i łączącej się z wybranym urządzeniem.
2. Następnie pobraliśmy wartości ustawień skanowania ze strony z dokumentacją i patrzyliśmy jak modyfikowanie wartości i parametrów wpływa na skanowanie.
3. Zajęliśmy się dodaniem ~~opgi~~ ~~zmian~~ ~~kontrastu~~ i ~~jasności~~ oraz ~~opgi~~ wyboru między kolorowym skanowaniem, a czarno-białym. Umieściliśmy okienko, w którym można wybrać wymienione opcje, skanować w poszukiwaniu urządzeń, wybrać urządzenie, skanować, zapisać do pliku oraz okienko wyświetlające zeskanowany obraz.
4. Dodaliśmy wybór formatu: A4 lub A5.

  
13.12.2023