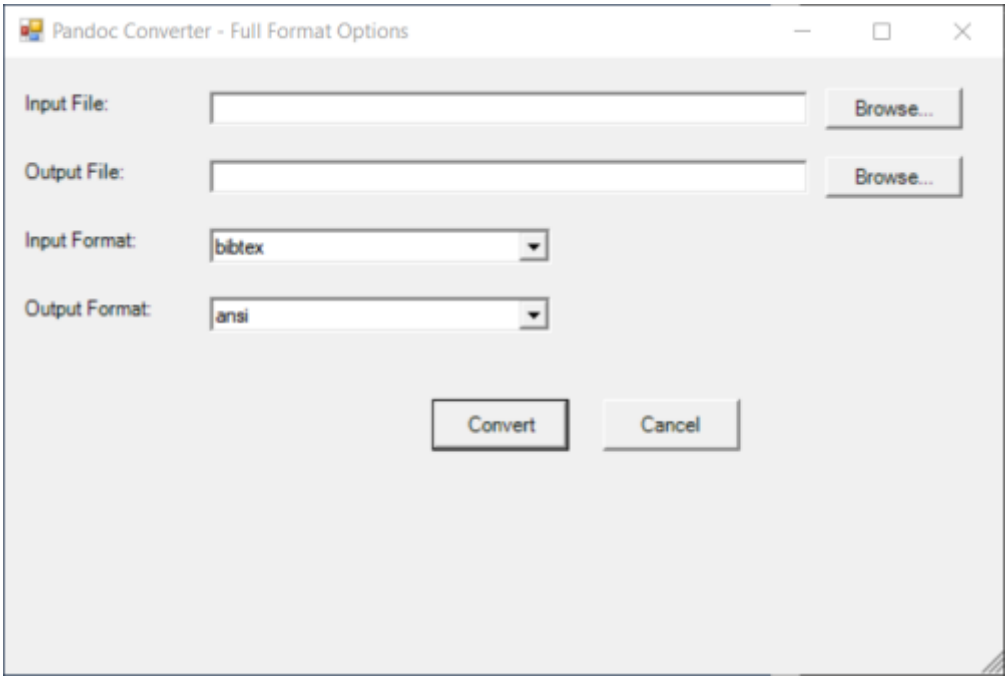

PS: Pandoc Converter GUI

Skrypt jest aplikacją PowerShell, która wykorzystuje Windows Forms w celu stworzenia graficznego interfejsu użytkownika (GUI) umożliwiającego konwersję plików z jednego formatu na inny za pomocą narzędzia Pandoc. W skrócie, skrypt umożliwia użytkownikowi wybór pliku wejściowego, pliku wyjściowego oraz formatów wejściowych i wyjściowych obsługiwanych przez Pandoc.

Mi personalnie ten skrypt przydaje się wieloktronie kiedy tylko coś tworzę. Zwyklę jak coś piszę to jest to w kilku formatach. Jeden to syntax dokuwiki który jest wykorzystywany tutaj na tej stronie, bardzo często też wiele rzeczy które piszę są napisane w LaTeX'u i jeszcze zdarza mi się coś napisać w MarkDownie. Pandoc to narzędzie które pozwala na łatwą konwersję pomiędzy tymi formatami.

Spora część artykułów na tej wiki została przekonwertowana za pomocą tego skryptu z formatów .md i .tex.

Załadowanie wymaganych bibliotek

Skrypt ładuje dwie biblioteki .NET, które umożliwiają stworzenie interfejsu graficznego:

- System.Windows.Forms – do obsługi GUI (okna, przyciski, itp.)
- System.Drawing – do operacji graficznych

Wybór formatów wejściowych i wyjściowych

Skrypt definiuje dwie listy zawierające formaty wejściowe oraz wyjściowe obsługiwane przez Pandoc:

- `$availableInputFormats` – lista formatów plików, które mogą być użyte jako dane wejściowe (np. markdown, docx, bibtex).

- `$availableOutputFormats` – lista formatów plików, które mogą być generowane jako dane wyjściowe (np. pdf, html, docx, dokuwiki).

Użytkownik wybiera formaty z rozwijanych list (ComboBox) w GUI.

Tworzenie formularza

Skrypt tworzy formularz z następującymi elementami:

- Pole tekstowe do wyboru pliku wejściowego (Input File).
- Pole tekstowe do wyboru pliku wyjściowego (Output File).
- Przyciski „Browse…” do przeglądania plików wejściowych i wyjściowych.
- Kombinacje rozwijane (ComboBox) do wyboru formatów wejściowych i wyjściowych.
- Przycisk „Convert” do uruchomienia konwersji.
- Przycisk „Cancel” do anulowania operacji.

Walidacja danych wejściowych

Po kliknięciu przycisku „Convert”, skrypt sprawdza, czy użytkownik wybrał pliki wejściowe i wyjściowe. Jeśli któryś z nich jest pusty, wyświetlany jest komunikat o błędzie.

Obsługa specyficznych przypadków

Jeśli plik wejściowy ma rozszerzenie `.tex` i użytkownik wybierze format „texinfo”, skrypt automatycznie zmienia format wejściowy na „latex”, ponieważ Pandoc wymaga, by pliki TeX były konwertowane jako „latex”, a nie „texinfo”.

Uruchomienie konwersji

Skrypt sprawdza, czy Pandoc jest zainstalowany i dostępny w systemie (sprawdzając polecenie `pandoc`). Jeśli Pandoc nie jest dostępny, użytkownik otrzymuje odpowiedni komunikat o błędzie.

Jeśli wszystkie warunki są spełnione, skrypt uruchamia polecenie Pandoc, przekazując odpowiednie argumenty:

- `-f` – format wejściowy
- `-t` – format wyjściowy
- `-o` – plik wyjściowy

Po zakończeniu konwersji wyświetlany jest komunikat o sukcesie.

Podsumowanie

Skrypt zapewnia łatwy sposób konwertowania plików między wieloma formatami, oferując użytkownikowi prosty interfejs graficzny do wyboru plików i formatów. Obsługuje różnorodne formaty wejściowe i wyjściowe obsługiwane przez Pandoc oraz zapewnia walidację danych i obsługę błędów.

kod skryptu:

pandoc.ps1

pandoc.ps1

```powershell
# Ensure necessary assemblies are loaded
Add-Type -AssemblyName System.Windows.Forms
Add-Type -AssemblyName System.Drawing

# Comprehensive list of Pandoc-supported input formats
$availableInputFormats = @(
    "bibtex",
    "biblatex",
    "bits",
    "commonmark",
    "commonmark_x",
    "creole",
    "csljson",
    "csv",
    "tsv",
    "djot",
    "docbook",
    "docx",
    "dokuwiki",
    "endnotexml",
    "epub",
    "fb2",
    "gfm",
    "haddock",
    "html",
    "ipynb",
    "jats",
    "jira",
    "json",
    "latex",
    "markdown",
    "markdown_mmd",
    "markdown_phpextra",
    "markdown_strict",
    "mediawiki",
    "man",
    "mdoc",
```

```
    "muse",
    "native",
    "odt",
    "opml",
    "org",
    "pod",
    "ris",
    "rtf",
    "rst",
    "t2t",
    "textile",
    "tikiwiki",
    "twiki",
    "typst",
    "vimwiki"
)

# Comprehensive list of Pandoc-supported output formats
$availableOutputFormats = @(
    "ansi",
    "asciidoc",
    "asciidoc_legacy",
    "asciidoctor",
    "beamer",
    "bibtex",
    "biblatex",
    "chunkedhtml",
    "commonmark",
    "commonmark_x",
    "context",
    "csljson",
    "djot",
    "docbook",
    "docbook4",
    "docbook5",
    "docx",
    "dokuwiki",
    "epub",
    "epub3",
    "epub2",
    "fb2",
    "gfm",
    "haddock",
    "html",
    "html5",
    "html4",
    "icml",
    "ipynb",
    "jats_archiving",
    "jats_articleauthoring",
    "jats_publishing",
```

```
    "jats",
    "jira",
    "json",
    "latex",
    "man",
    "markdown",
    "markdown_mmd",
    "markdown_phpextra",
    "markdown_strict",
    "markua",
    "mediawiki",
    "ms",
    "muse",
    "native",
    "odt",
    "opml",
    "opendocument",
    "org",
    "pdf",
    "plain",
    "pptx",
    "rst",
    "rtf",
    "texinfo",
    "textile",
    "slideous",
    "slidy",
    "dzslides",
    "revealjs",
    "s5",
    "tei",
    "typst",
    "xwiki",
    "zimwiki"
)

# Create the main form
$form = New-Object System.Windows.Forms.Form
$form.Text = "Pandoc Converter - Full Format Options"
$form.Size = New-Object System.Drawing.Size(600,400)
$form.StartPosition = "CenterScreen"

# --- Input File Controls ---
$inputLabel = New-Object System.Windows.Forms.Label
$inputLabel.Location = New-Object System.Drawing.Point(10,20)
$inputLabel.Size = New-Object System.Drawing.Size(100,20)
$inputLabel.Text = "Input File:"
$form.Controls.Add($inputLabel)

$inputTextBox = New-Object System.Windows.Forms.TextBox
$inputTextBox.Location = New-Object System.Drawing.Point(120,20)
```

```
$inputTextBox.Size = New-Object System.Drawing.Size(350,20)
$form.Controls.Add($inputTextBox)

$inputBrowseButton = New-Object System.Windows.Forms.Button
$inputBrowseButton.Location = New-Object System.Drawing.Point(480,18)
$inputBrowseButton.Size = New-Object System.Drawing.Size(80,24)
$inputBrowseButton.Text = "Browse..."
$inputBrowseButton.Add_Click({
    $openDialog = New-Object System.Windows.Forms.OpenFileDialog
    $openDialog.Filter = "All Files (*.*)|*.*"
    $openDialog.Title = "Select an Input File"
    if ($openDialog.ShowDialog() -eq
[System.Windows.Forms.DialogResult]::OK) {
        $inputTextBox.Text = $openDialog.FileName
    }
})
$form.Controls.Add($inputBrowseButton)

# --- Output File Controls ---
$outputLabel = New-Object System.Windows.Forms.Label
$outputLabel.Location = New-Object System.Drawing.Point(10,60)
$outputLabel.Size = New-Object System.Drawing.Size(100,20)
$outputLabel.Text = "Output File:"
$form.Controls.Add($outputLabel)

$outputTextBox = New-Object System.Windows.Forms.TextBox
$outputTextBox.Location = New-Object System.Drawing.Point(120,60)
$outputTextBox.Size = New-Object System.Drawing.Size(350,20)
$form.Controls.Add($outputTextBox)

$outputBrowseButton = New-Object System.Windows.Forms.Button
$outputBrowseButton.Location = New-Object System.Drawing.Point(480,58)
$outputBrowseButton.Size = New-Object System.Drawing.Size(80,24)
$outputBrowseButton.Text = "Browse..."
$outputBrowseButton.Add_Click({
    $saveDialog = New-Object System.Windows.Forms.SaveFileDialog
    $saveDialog.Filter = "All Files (*.*)|*.*"
    $saveDialog.Title = "Select Output File Destination"
    if ($saveDialog.ShowDialog() -eq
[System.Windows.Forms.DialogResult]::OK) {
        $outputTextBox.Text = $saveDialog.FileName
    }
})
$form.Controls.Add($outputBrowseButton)

# --- Input Format ComboBox ---
$inputFormatLabel = New-Object System.Windows.Forms.Label
$inputFormatLabel.Location = New-Object System.Drawing.Point(10,100)
$inputFormatLabel.Size = New-Object System.Drawing.Size(100,20)
$inputFormatLabel.Text = "Input Format:"
$form.Controls.Add($inputFormatLabel)
```

```
$inputFormatCombo = New-Object System.Windows.Forms.ComboBox
$inputFormatCombo.Location = New-Object System.Drawing.Point(120,100)
$inputFormatCombo.Size = New-Object System.Drawing.Size(200,20)
$inputFormatCombo.DropDownStyle =
[System.Windows.Forms.ComboBoxStyle]::DropDownList
foreach ($fmt in $availableInputFormats) {
    $inputFormatCombo.Items.Add($fmt) | Out-Null
}
$inputFormatCombo.SelectedIndex = 0
$form.Controls.Add($inputFormatCombo)

# --- Output Format ComboBox ---
$outputFormatLabel = New-Object System.Windows.Forms.Label
$outputFormatLabel.Location = New-Object System.Drawing.Point(10,140)
$outputFormatLabel.Size = New-Object System.Drawing.Size(100,20)
$outputFormatLabel.Text = "Output Format:"
$form.Controls.Add($outputFormatLabel)

$outputFormatCombo = New-Object System.Windows.Forms.ComboBox
$outputFormatCombo.Location = New-Object System.Drawing.Point(120,140)
$outputFormatCombo.Size = New-Object System.Drawing.Size(200,20)
$outputFormatCombo.DropDownStyle =
[System.Windows.Forms.ComboBoxStyle]::DropDownList
foreach ($fmt in $availableOutputFormats) {
    $outputFormatCombo.Items.Add($fmt) | Out-Null
}
$outputFormatCombo.SelectedIndex = 0
$form.Controls.Add($outputFormatCombo)

# --- Convert and Cancel Buttons ---
$convertButton = New-Object System.Windows.Forms.Button
$convertButton.Location = New-Object System.Drawing.Point(250,200)
$convertButton.Size = New-Object System.Drawing.Size(80,30)
$convertButton.Text = "Convert"
$convertButton.DialogResult = [System.Windows.Forms.DialogResult]::OK
$form.Controls.Add($convertButton)

$cancelButton = New-Object System.Windows.Forms.Button
$cancelButton.Location = New-Object System.Drawing.Point(350,200)
$cancelButton.Size = New-Object System.Drawing.Size(80,30)
$cancelButton.Text = "Cancel"
$cancelButton.DialogResult =
[System.Windows.Forms.DialogResult]::Cancel
$form.Controls.Add($cancelButton)

$form.AcceptButton = $convertButton
$form.CancelButton = $cancelButton

# Show the form and wait for user interaction
$result = $form.ShowDialog()
```

```
if ($result -eq [System.Windows.Forms.DialogResult]::OK) {
    $inputFile = $inputTextBox.Text
    $outputFile = $outputTextBox.Text
    $selectedInputFormat = $inputFormatCombo.SelectedItem
    $selectedOutputFormat = $outputFormatCombo.SelectedItem

    # Validate file selections
    if ([string]::IsNullOrWhiteSpace($inputFile)) {
        [System.Windows.Forms.MessageBox]::Show("Please select an input
file.","Error",
            [System.Windows.Forms.MessageBoxButtons]::OK,
            [System.Windows.Forms.MessageBoxIcon]::Error)
        exit
    }
    if ([string]::IsNullOrWhiteSpace($outputFile)) {
        [System.Windows.Forms.MessageBox]::Show("Please select an
output file.","Error",
            [System.Windows.Forms.MessageBoxButtons]::OK,
            [System.Windows.Forms.MessageBoxIcon]::Error)
        exit
    }

    # If input file has .tex extension and user selected 'texinfo',
override to 'latex'
    $ext = [System.IO.Path]::GetExtension($inputFile).ToLower()
    if ($ext -eq ".tex" -and $selectedInputFormat -eq "texinfo") {
        [System.Windows.Forms.MessageBox]::Show("For TeX files, the
input format has been changed from 'texinfo' to 'latex'.",
            "Info", [System.Windows.Forms.MessageBoxButtons]::OK,
            [System.Windows.Forms.MessageBoxIcon]::Information)
        $selectedInputFormat = "latex"
    }

    # Check if pandoc is available
    if (-not (Get-Command pandoc -ErrorAction SilentlyContinue)) {
        [System.Windows.Forms.MessageBox]::Show("Pandoc is not
installed or not in the system path. Please install pandoc and try
again.",
            "Error", [System.Windows.Forms.MessageBoxButtons]::OK,
            [System.Windows.Forms.MessageBoxIcon]::Error)
        exit
    }

    # Run pandoc conversion using selected formats (-f for input, -t
for output)
    pandoc "$inputFile" -f $selectedInputFormat -t
$selectedOutputFormat -o "$outputFile"

    [System.Windows.Forms.MessageBox]::Show("Conversion
complete.`nOutput saved as:" + "`n" + "$outputFile",
```

```
        "Success", [System.Windows.Forms.MessageBoxButtons]::OK,
        [System.Windows.Forms.MessageBoxIcon]::Information)
} else {
    Write-Host "Operation cancelled by user."
}
```