How to use image preprocessing filters in PowerShell using ByteScout Text Recognition SDK

Write code in PowerShell to use image preprocessing filters with this step-by-step tutorial

The sample shows steps and algorithm of how to use image preprocessing filters and how to make it work in your PowerShell application. ByteScout Text Recognition SDK: the text recognition SDK to help with extraction of text using OCR from scanned images and documents. Supports English and non-Latin languages, can take PDF as input. It can use image preprocessing filters in PowerShell.

The SDK samples like this one below explain how to quickly make your application do use image preprocessing filters in PowerShell with the help of ByteScout Text Recognition SDK. Just copy and paste the code into your PowerShell application's code and follow the instruction. This basic programming language sample code for PowerShell will do the whole work for you to use image preprocessing filters.

Free trial version of ByteScout Text Recognition SDK is available for download from our website. Get it to try other source code samples for PowerShell.

PowerShell - ImagePreprocessingFilters.ps1

```
# Add reference to ByteScout.TextRecognition.dll assembly
Add-Type -Path "c:\Program Files\ByteScout Text Recognition
SDK\net40\ByteScout.TextRecognition.dll"

$InputDocument = "..\..\skewed.png"
$OutputDocument = ".\result.txt"

# Create and activate TextRecognizer instance
$textRecognizer = New-Object ByteScout.TextRecognition.TextRecognizer
$textRecognizer.RegistrationName = "demo"
$textRecognizer.RegistrationKey = "demo"

try {
    # Load document (image or PDF)
    $textRecognizer.LoadDocument($InputDocument)

    # Set the location of OCR language data files
    $textRecognizer.OCRLanguageDataFolder = "c:\Program Files\ByteScout Text
Recognition SDK\ocrdata_best\"

    # Set OCR language.
    # "eng" for english, "deu" for German, "fra" for French, "spa" for Spanish, etc.
- according to files in "ocrdata" folder
    # Find more language files at https://github.com/bytescout/ocrdata
    $textRecognizer.OCRLanguage = "eng"
    # Add deskew filter that automatically rotates the image to make the text
horizontal.
    # Note, it analyzes the left edge of scanned text. Any dark artifacts may prevent
    # the correct angle detection.
    $textRecognizer.ImagePreprocessingFilters.AddDeskew()

    # Other filters that may be useful to improve recognition
    # (note, the filters are applied in the order they were added):

    # Improve image contrast.
    #$textRecognizer.ImagePreprocessingFilters.AddContrast()

    # Apply gamma correction.
    #$textRecognizer.ImagePreprocessingFilters.AddGammaCorrection()

    # Apply median filter. Helps to remove noise.
    #$textRecognizer.ImagePreprocessingFilters.AddMedian()

    # Apply dilate filter. Helps to cure symbols erosion.
    #$textRecognizer.ImagePreprocessingFilters.AddDilate()

    # Lines removers. Removing borders of some tables may improve the recognition.
    #$textRecognizer.ImagePreprocessingFilters.AddHorizontalLinesRemover()
    #$textRecognizer.ImagePreprocessingFilters.AddVerticalLinesRemover()


    # Recognize text from all pages and save it to file
    $textRecognizer.SaveText($OutputDocument)

    # Open the result file in default associated application (for demo purposes)
    & $OutputDocument
}
catch {
    # Display exception
    Write-Host $_.Exception.Message
}

$textRecognizer.Dispose()
```

PowerShell - run.bat

```
@echo off

powershell -NoProfile -ExecutionPolicy Bypass -Command "&
.\ImagePreprocessingFilters.ps1"
echo Script finished with errorlevel=%errorlevel%

pause
```

FOR MORE INFORMATION AND FREE TRIAL:

Download Free Trial SDK (on-premise version)

Read more about ByteScout Text Recognition SDK

Explore documentation

Visit www.ByteScout.com

or

Get Your Free API Key for www.PDF.co Web API