How to convert PDF to JPEG from URL asynchronously for PDF to image API in PowerShell with PDF.co Web API

How to convert PDF to JPEG from URL asynchronously in PowerShell with easy ByteScout code samples to make PDF to image API. Step-by-step tutorial

Source code documentation samples provide quick and easy way to add a required functionality into your application. PDF.co Web API helps with PDF to image API in PowerShell. PDF.co Web API is the flexible Web API that includes full set of functions from e-signature requests to data extraction, OCR, images recognition, pdf splitting and pdf splitting. Can also generate barcodes and read barcodes from images, scans and pdf.

You will save a lot of time on writing and testing code as you may just take the code below and use it in your application. Sample code in PowerShell is all you need. Copy-paste it to your the code editor, then add a reference to PDF.co Web API and you are ready to try it! You can use these PowerShell sample examples in one or many applications.

Our website provides free trial version of PDF.co Web API that includes source code samples to help with your PowerShell project.

PowerShell - ConvertPdfToJpegFromUrlAsynchronously.ps1

```
# Cloud API asynchronous "PDF To JPEG" job example.
# Allows to avoid timeout errors when processing huge or scanned PDF documents.

# The authentication key (API Key).
# Get your own by registering at https://app.pdf.co/documentation/api
$API_KEY = "**********************************"

# Direct URL of source PDF file.
$SourceFileUrl = "https://bytescout-com.s3.amazonaws.com/files/demo-files/cloud-
api/pdf-to-image/sample.pdf"
# Comma-separated list of page indices (or ranges) to process. Leave empty for all
pages. Example: '0,2-5,7-'.
$Pages = ""
# PDF document password. Leave empty for unprotected documents.
$Password = ""
# (!) Make asynchronous job
$Async = $true


# Prepare URL for `PDF To JPEG` API call
$query = "https://api.pdf.co/v1/pdf/convert/to/jpg?password={0}&pages={1}&url=
{2}&async={3}" -f `
    $Password, $Pages, $SourceFileUrl, $Async
```

```
$query = [System.Uri]::EscapeUriString($query)

try {
    # Execute request
    $jsonResponse = Invoke-RestMethod -Method Get -Headers @{ "x-api-key" = $API_KEY
} -Uri $query

    if ($jsonResponse.error -eq $false) {
        # Asynchronous job ID
        $jobId = $jsonResponse.jobId
        # URL of generated JSON file available after the job completion; it will
contain URLs of result PDF files.
        $resultJsonFileUrl = $jsonResponse.url

        # Check the job status in a loop.
        do {
            $statusCheckUrl = "https://api.pdf.co/v1/job/check?jobid=" + $jobId
            $jsonStatus = Invoke-RestMethod -Method Get -Headers @{ "x-api-key" =
$API_KEY } -Uri $statusCheckUrl

            # Display timestamp and status (for demo purposes)
            Write-Host "$(Get-date): $($jsonStatus.status)"

            if ($jsonStatus.status -eq "success") {
                # Download JSON file with URLs of result PDF files
                $jsonJpegUrls = Invoke-RestMethod -Method Get -Uri $resultJsonFileUrl

                # Download generated JPEG files
                $part = 1;
                foreach ($url in $jsonJpegUrls) {
                    $localFileName = ".\page$($part).jpg"

                    Invoke-WebRequest -Headers @{ "x-api-key" = $API_KEY } -OutFile
$localFileName -Uri $url

                    Write-Host "Downloaded `"$($localFileName)`""
                    $part++
                }
                break
            }
            elseif ($jsonStatus.status -eq "working") {
                # Pause for a few seconds
                Start-Sleep -Seconds 3
            }
            else {
                Write-Host $jsonStatus.status
                break
            }
        }
        while ($true)
    }
    else {
        # Display service reported error
        Write-Host $jsonResponse.message
    }
}
catch {
    # Display request error
    Write-Host $_.Exception
}
```

```
```

PowerShell - run.bat

```
@echo off

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

pause
```
