How to read barcode from URL asynchronously for barcode reader API in PowerShell using PDF.co Web API

How to read barcode from URL asynchronously for barcode reader API in PowerShell: How To Tutorial

Today you are going to learn how to read barcode from URL asynchronously in PowerShell. PDF.co Web API was made to help with barcode reader API in PowerShell. PDF.co Web API is the Rest API that provides set of data extraction functions, tools for documents manipulation, splitting and merging of pdf files. Includes built-in OCR, images recognition, can generate and read barcodes from images, scans and pdf.

The SDK samples like this one below explain how to quickly make your application do barcode reader API in PowerShell with the help of PDF.co Web API. 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! Enjoy writing a code with ready-to-use sample PowerShell codes to add barcode reader API functions using PDF.co Web API in PowerShell.

ByteScout free trial version is available for FREE download from our website. Programming tutorials along with source code samples are included.

PowerShell - ReadBarcodeFromUrlAsynchronously.ps1

```
# Cloud API asynchronous "Barcode Reader" 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 file to search barcodes in.
$SourceFileURL = "https://bytescout-com.s3.amazonaws.com/files/demo-files/cloud-
api/barcode-reader/sample.pdf"
# Comma-separated list of barcode types to search.
# See valid barcode types in the documentation
https://app.pdf.co/documentation/api/1.0/barcode/read_from_url.html
$BarcodeTypes = "Code128,Code39,Interleaved2of5,EAN13"
# Comma-separated list of page indices (or ranges) to process. Leave empty for all
pages. Example: '0,2-5,7-'.
$Pages = ""
# (!) Make asynchronous job
$Async = $true


# Prepare URL for `Barcode Reader` API call
```

```
$query = "https://api.pdf.co/v1/barcode/read/from/url?types=$($BarcodeTypes)&pages=$($Pages)&url=$($SourceFileURL)&async=$($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 with decoded barcodes that will available after the job completion
        $resultFileUrl = $jsonResponse.url

        # Check the job status in a loop.
        # If you don't want to pause the main thread you can rework the code
        # to use a separate thread for the status checking and completion.
        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 decoded barcodes
                $jsonFoundBarcodes = Invoke-RestMethod -Method Get -Headers @{ "x-api-key" = $API_KEY } -Uri $resultFileUrl

                # Display found barcodes in console
                foreach ($barcode in $jsonFoundBarcodes)
                {
                    Write-Host "Found barcode:"
                    Write-Host "  Type: " + $barcode.TypeName
                    Write-Host "  Value: " + $barcode."Value"
                    Write-Host "  Document Page Index: " + $barcode."Page"
                    Write-Host "  Rectangle: " + $barcode."Rect"
                    Write-Host "  Confidence: " + $barcode."Confidence"
                    Write-Host ""
                }
                break
            }
            elseif ($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 "&
.\ReadBarcodeFromUrlAsynchronously.ps1"
echo Script finished with errorlevel=%errorlevel%

pause
```

FOR MORE INFORMATION AND FREE TRIAL:

Download Free Trial SDK (on-premise version)

Read more about PDF.co Web API

Explore documentation

Visit www.ByteScout.com

or

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