www.bytescout.com

How to add text and images to PDF in PowerShell with ByteScout Cloud API Server

Continuous learning is a crucial part of computer science and this tutorial shows how to add text and images to PDF in PowerShell

Quickly learn how to add text and images to PDF in PowerShell with this sample source code. What is ByteScout Cloud API Server? It is the ready to deploy Web API Server that can be deployed in less than thirty minutes into your own in-house Windows server (no Internet connnection is required to process data!) or into private cloud server. Can store data on in-house local server based storage or in Amazon AWS S3 bucket. Processing data solely on the server using built-in ByteScout powered engine, no cloud services are used to process your data!. It can help you to add text and images to PDF in your PowerShell application.

Want to quickly learn? This fast application programming interfaces of ByteScout Cloud API Server for PowerShell plus the guidelines and the code below will help you quickly learn how to add text and images to PDF. Follow the instructions from scratch to work and copy the PowerShell code. Check PowerShell sample code samples to see if they respond to your needs and requirements for the project.

The trial version of ByteScout Cloud API Server can be downloaded for free from our website. It also includes source code samples for PowerShell and other programming languages.

FOR MORE INFORMATION AND FREE TRIAL:

Download Free Trial SDK (on-premise version)

Read more about ByteScout Cloud API Server

Explore API Documentation

Get Free Training for ByteScout Cloud API Server

Get Free API key for Web API

visit www.ByteScout.com

Source Code Files:

AddTextByFindingTargetCoordinates.ps1

```
# Direct URL of source PDF file.
$SourceFileUrl = "https://bytescout-com.s3.amazonaws.com/files/demo-files/cloud-api/pd

# Search string.
$SearchString = 'Notes'

# Please NOTE: In this sample we're assuming Cloud Api Server is hosted at "https://lo
# If it's not then please replace this with with your hosting url.

# Prepare URL for PDF text search API call.
$queryFindText = "https://localhost/pdf/find?url=$($SourceFileURL)&searchString=$($Sea
$queryFindText = [System.Uri]::EscapeUriString($queryFindText)

try {
    # Execute request
    $jsonResponseFindText = Invoke-RestMethod -Method Get  -Uri $queryFindText

    if ($jsonResponseFindText.error -eq $false) {

        # Display search information
        $itemFindText = $jsonResponseFindText.body[0];

        Write-Host "Found text at coordinates $($itemFindText.left), $($itemFindText.t

        #Comma-separated list of page indices (or ranges) to process. Leave empty for
        $Pages = ""

        # PDF document password. Leave empty for unprotected documents.
        $Password = ""

        # Destination PDF file name
        $DestinationFile = "./result.pdf"

        # Text annotation params
        $Type = "annotation";
        $X = $itemFindText.left;
        $Y = $itemFindText.top + 25;
        $Text = "Some notes will go here... Some notes will go here.... Some notes wil
        $FontName = "Times New Roman";
        $FontSize = 12;
        $Color = "FF0000";

        $resultFileName = [System.IO.Path]::GetFileName($DestinationFile)

        # * Add Text *
        # Prepare request to `PDF Edit` API endpoint
        $query = "https://localhost/pdf/edit/add?name=$($resultFileName)&password=$($P
        $query = [System.Uri]::EscapeUriString($query)

        try {
            # Execute request
            $jsonResponse = Invoke-RestMethod -Method Get  -Uri $query
```

```
            if ($jsonResponse.error -eq $false) {
                # Get URL of generated barcode image file
                $resultFileUrl = $jsonResponse.url

                # Download output file
                Invoke-WebRequest -Uri $resultFileUrl -OutFile $DestinationFile

                Write-Host "Generated PDF saved to '$($DestinationFile)' file."
            }
            else {
                # Display service reported error
                Write-Host $jsonResponse.message
            }
        }
        catch {
            # Display request error
            Write-Host $_.Exception
        }
    }
    else {
        # Display service reported error
        Write-Host $jsonResponse.message
    }
}
catch {
    # Display request error
    Write-Host $_.Exception
}
```

run.bat

```
@echo off

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

pause
```

VIDEO

https://www.youtube.com/watch?v=NEwNs2b9YN8