How to find table in pdf and extract as xml with pdf extractor sdk in VB.NET and ByteScout Data Extraction Suite

Learn to code in VB.NET to find table in pdf and extract as xml with pdf extractor sdk with this step-by-step tutorial

This sample source code below will display you how to find table in pdf and extract as xml with pdf extractor sdk in VB.NET. ByteScout Data Extraction Suite is the bundle that includes three SDK tools for data extraction from PDF, scans, images and from spreadsheets: PDF Extractor SDK, Data Extraction SDK, Barcode Reader SDK. It can find table in pdf and extract as xml with pdf extractor sdk in VB.NET.

The following code snippet for ByteScout Data Extraction Suite works best when you need to quickly find table in pdf and extract as xml with pdf extractor sdk in your VB.NET application. IF you want to implement the functionality, just copy and paste this code for VB.NET below into your code editor with your app, compile and run your application. Enjoy writing a code with ready-to-use sample VB.NET codes.

If you want to try other source code samples then the free trial version of ByteScout Data Extraction Suite is available for download from our website. Just try other source code samples for VB.NET.

FOR MORE INFORMATION AND FREE TRIAL:

Download Free Trial SDK (on-premise version)

Read more about ByteScout Data Extraction Suite

Explore API Documentation

Get Free Training for ByteScout Data Extraction Suite

Get Free API key for Web API

visit www.ByteScout.com

Source Code Files:

Program.vb

```vb
Imports Bytescout.PDFExtractor

Class Program
        Friend Shared Sub Main(args As String())

        ' Create Bytescout.PDFExtractor.XMLExtractor instance
        Dim xmlExtractor As New XMLExtractor()
        xmlExtractor.RegistrationName = "demo"
        xmlExtractor.RegistrationKey = "demo"

        ' Create Bytescout.PDFExtractor.TableDetector instance
        Dim tableDetector As New TableDetector()
        tableDetector.RegistrationName = "demo"
        tableDetector.RegistrationKey = "demo"

        ' We should define what kind of tables we should detect.
        ' So we set min required number of columns to 3 ...
        tableDetector.DetectionMinNumberOfColumns = 3
        ' ... and we set min required number of rows to 3
        tableDetector.DetectionMinNumberOfRows = 3

            ' Load sample PDF document
        xmlExtractor.LoadDocumentFromFile(".\sample3.pdf")
        tableDetector.LoadDocumentFromFile(".\sample3.pdf")

            ' Get page count
        Dim pageCount As Integer = tableDetector.GetPageCount()

            For i As Integer = 0 To pageCount - 1
        Dim t As Integer = 1
        ' Find first table and continue if found
        If (tableDetector.FindTable(i)) Then
            Do
                ' Set extraction area for XML extractor to rectangle received from
                xmlExtractor.SetExtractionArea(tableDetector.FoundTableLocation)
                ' Export the table to XML file
                xmlExtractor.SavePageXMLToFile(i, "page-" + i.ToString() + "-table-
                t = t + 1
            Loop While tableDetector.FindNextTable()
        End If
        Next

        ' Cleanup
            xmlExtractor.Dispose()
            tableDetector.Dispose()

        ' Open first output file in default associated application (for demo purposes)
        System.Diagnostics.Process.Start("page-0-table-1.xml")

        End Sub
End Class
```

VIDEO

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

ON-PREMISE OFFLINE SDK

60 Day Free Trial or Visit ByteScout Data Extraction Suite Home Page
Explore ByteScout Data Extraction Suite Documentation
Explore Samples
Sign Up for ByteScout Data Extraction Suite Online Training

ON-DEMAND REST WEB API

Get Your API Key
Explore Web API Docs
Explore Web API Samples

visit www.ByteScout.com

visit www.PDF.co