www.bytescout.com

How to convert html email with attachments to pdf with pdf sdk in C# and ByteScout PDF Suite

Learn to code in C# to convert html email with attachments to pdf with pdf sdk with this step-by-step tutorial

The sample source code below will teach you how to convert html email with attachments to pdf with pdf sdk in C#. What is ByteScout PDF Suite? It is the set that includes 6 SDK products to work with PDF from generating rich PDF reports to extracting data from PDF documents and converting them to HTML. This bundle includes PDF (Generator) SDK, PDF Renderer SDK, PDF Extractor SDK, PDF to HTML SDK, PDF Viewer SDK and PDF Generator SDK for Javascript. It can help you to convert html email with attachments to pdf with pdf sdk in your C# application.

These C# code samples for C# guide developers to speed up coding of the application when using ByteScout PDF Suite. Just copy and paste the code into your C# application's code and follow the instructions. Check C# sample code samples to see if they respond to your needs and requirements for the project.

ByteScout provides the free trial version of ByteScout PDF Suite along with the documentation and source code samples.

FOR MORE INFORMATION AND FREE TRIAL:

Download Free Trial SDK (on-premise version)

Read more about ByteScout PDF Suite

Explore API Documentation

Get Free Training for ByteScout PDF Suite

Get Free API key for Web API

visit www.ByteScout.com

Source Code Files:

EmailToPDF_HTMLEmailWithAttachments.NETCore.sln

```
Microsoft Visual Studio Solution File, Format Version 12.00
# Visual Studio 15
VisualStudioVersion = 15.0.27703.2026
MinimumVisualStudioVersion = 10.0.40219.1
Project("{9A19103F-16F7-4668-BE54-9A1E7A4F7556}") = "EmailToPDF_HTMLEmailWithAttachment
EndProject
Global
	GlobalSection(SolutionConfigurationPlatforms) = preSolution
		Debug|Any CPU = Debug|Any CPU
		Release|Any CPU = Release|Any CPU
	EndGlobalSection
	GlobalSection(ProjectConfigurationPlatforms) = postSolution
		{E89F0EAC-8901-4D97-9B3B-B27B85E74C38}.Debug|Any CPU.ActiveCfg = Debug
		{E89F0EAC-8901-4D97-9B3B-B27B85E74C38}.Debug|Any CPU.Build.0 = Debug|A
		{E89F0EAC-8901-4D97-9B3B-B27B85E74C38}.Release|Any CPU.ActiveCfg = Rel
		{E89F0EAC-8901-4D97-9B3B-B27B85E74C38}.Release|Any CPU.Build.0 = Releas
	EndGlobalSection
	GlobalSection(SolutionProperties) = preSolution
		HideSolutionNode = FALSE
	EndGlobalSection
	GlobalSection(ExtensibilityGlobals) = postSolution
		SolutionGuid = {93F50891-FF6A-483A-86F6-F96860F99AF3}
	EndGlobalSection
EndGlobal
```

EmailToPDF_HTMLEmailWithAttachments.sln

```
Microsoft Visual Studio Solution File, Format Version 12.00
# Visual Studio 15
VisualStudioVersion = 15.0.27703.2026
MinimumVisualStudioVersion = 10.0.40219.1
Project("{FAE04EC0-301F-11D3-BF4B-00C04F79EFBC}") = "EmailToPDF_HTMLEmailWithAttachment
EndProject
Global
	GlobalSection(SolutionConfigurationPlatforms) = preSolution
		Debug|Any CPU = Debug|Any CPU
		Release|Any CPU = Release|Any CPU
	EndGlobalSection
	GlobalSection(ProjectConfigurationPlatforms) = postSolution
		{8B33CB1C-B6A2-4750-9D1C-EB963DC8A17D}.Debug|Any CPU.ActiveCfg = Debug
		{8B33CB1C-B6A2-4750-9D1C-EB963DC8A17D}.Debug|Any CPU.Build.0 = Debug|A
		{8B33CB1C-B6A2-4750-9D1C-EB963DC8A17D}.Release|Any CPU.ActiveCfg = Rel
		{8B33CB1C-B6A2-4750-9D1C-EB963DC8A17D}.Release|Any CPU.Build.0 = Releas
	EndGlobalSection
```

```
        GlobalSection(SolutionProperties) = preSolution
                HideSolutionNode = FALSE
        EndGlobalSection
        GlobalSection(ExtensibilityGlobals) = postSolution
                SolutionGuid = {70C9B22B-3F6D-493A-897D-F24861117CAA}
        EndGlobalSection
EndGlobal
```

Program.cs

```
using System;
using System.Diagnostics;
using System.Drawing.Printing;
using System.IO;
using System.Text;
using Bytescout.PDF;
using Bytescout.PDF.Converters;

using Font = Bytescout.PDF.Font;
using SolidBrush = Bytescout.PDF.SolidBrush;

namespace EmailToPDF_HTMLEmailWithAttachments
{
    class Program
    {
        static void Main(string[] args)
        {
            try
            {
                // Parse MessageContents using MsgReader Library
                // MsgReader library can be obtained from: https://github.com/Sicos197
                using (var msg = new MsgReader.Outlook.Storage.Message("HtmlSampleEmai
                {
                    // Get Sender information
                    var from = msg.GetEmailSender(false, false);

                    // Message sent datetime
                    var sentOn = msg.SentOn;

                    // Recipient To information
                    var recipientsTo = msg.GetEmailRecipients(MsgReader.Outlook.Recipie

                    // Recipient CC information
                    var recipientsCc = msg.GetEmailRecipients(MsgReader.Outlook.Recipie

                    // Message subject
                    var subject = msg.Subject;

                    // Get Message Body
                    var msgBody = msg.BodyHtml;
```

```
                // Prepare PDF docuemnt
                using (Document outputDocument = new Document())
                {
                    // Add registration keys
                    outputDocument.RegistrationName = "demo";
                    outputDocument.RegistrationKey = "demo";

                    // Add page
                    Page page = new Page(PaperFormat.A4);
                    outputDocument.Pages.Add(page);

                    // Add sample content
                    Font font = new Font(StandardFonts.Times, 12);
                    Brush brush = new SolidBrush();

                    // Add Email contents
                    int topMargin = 20;
                    page.Canvas.DrawString({code}quot;File Name: {msg.FileName}",
                    page.Canvas.DrawString({code}quot;From: {from}", font, brush,
                    page.Canvas.DrawString({code}quot;Sent On: {(sentOn.HasValue ?
                    page.Canvas.DrawString({code}quot;To: {recipientsTo}", font, br

                    if (!string.IsNullOrEmpty(recipientsCc))
                    {
                        page.Canvas.DrawString({code}quot;CC: {recipientsCc}", font
                    }

                    page.Canvas.DrawString({code}quot;Subject: {subject}", font, br
                    page.Canvas.DrawString("Message body and attachments in next pa

                    // Convert Html body to PDF in order to retain all formatting.
                    using (HtmlToPdfConverter converter = new HtmlToPdfConverter()
                    {
                        converter.PageSize = PaperKind.A4;
                        converter.Orientation = Bytescout.PDF.Converters.PaperOrien

                        // Convert input HTML to stream
                        byte[] byteArrayBody = Encoding.UTF8.GetBytes(msgBody);
                        MemoryStream inputStream = new MemoryStream(byteArrayBody)

                        // Create output stream to store generated PDF file
                        MemoryStream outputStream = new MemoryStream();

                        // Convert HTML to PDF
                        converter.ConvertHtmlToPdf(inputStream, outputStream);

                        // Create new document from generated output stream
                        Document docContent = new Document(outputStream);

                        // Append all pages to main PDF
                        foreach (Page item in docContent.Pages)
                        {
                            outputDocument.Pages.Add(item);
                        }

                        // Get attachments from message (if any, and append to docu
                        if (msg.Attachments.Count > 0)
                        {
                            foreach (MsgReader.Outlook.Storage.Attachment itmAttach
```

```
                                {
                                    // Get Memory Stream
                                    MemoryStream attachmentMemoryStream = new MemorySt

                                    // Append Attachment
                                    Document docAttachment = new Document(attachmentMe

                                    // Append all pages to main PDF
                                    foreach (Page item in docAttachment.Pages)
                                    {
                                        outputDocument.Pages.Add(item);
                                    }
                                }
                            }

                            // Save output file
                            outputDocument.Save("result.pdf");
                        }

                        // Open result document in default associated application (for
                        ProcessStartInfo processStartInfo = new ProcessStartInfo("resu
                        processStartInfo.UseShellExecute = true;
                        Process.Start(processStartInfo);
                    }
                }
            }
            catch (Exception ex)
            {
                Console.WriteLine(ex.Message);
                Console.WriteLine("Press enter key to exit...");
                Console.ReadLine();
            }
        }

    }
}
```

Program_Core.cs

```
using System;
using System.Diagnostics;
using System.Drawing.Printing;
using System.IO;
using System.Text;
using Bytescout.PDF;
using Bytescout.PDF.Converters;

using Font = Bytescout.PDF.Font;
using SolidBrush = Bytescout.PDF.SolidBrush;

namespace EmailToPDF_HTMLEmailWithAttachments
```

```
{
    class Program
    {
        static void Main(string[] args)
        {
            try
            {
                // Registering encoding provider for .net core solution
                System.Text.EncodingProvider encodingProvider = System.Text.CodePagesEr
                Encoding.RegisterProvider(encodingProvider);

                // Parse MessageContents using MsgReader Library
                // MsgReader library can be obtained from: https://github.com/Sicos197
                using (var msg = new MsgReader.Outlook.Storage.Message("HtmlSampleEmai
                {
                    // Get Sender information
                    var from = msg.GetEmailSender(false, false);

                    // Message sent datetime
                    var sentOn = msg.SentOn;

                    // Recipient To information
                    var recipientsTo = msg.GetEmailRecipients(MsgReader.Outlook.Recipie

                    // Recipient CC information
                    var recipientsCc = msg.GetEmailRecipients(MsgReader.Outlook.Recipie

                    // Message subject
                    var subject = msg.Subject;

                    // Get Message Body
                    var msgBody = msg.BodyHtml;

                    // Prepare PDF docuemnt
                    using (Document outputDocument = new Document())
                    {
                        // Add registration keys
                        outputDocument.RegistrationName = "demo";
                        outputDocument.RegistrationKey = "demo";

                        // Add page
                        Page page = new Page(PaperFormat.A4);
                        outputDocument.Pages.Add(page);

                        // Add sample content
                        Font font = new Font(StandardFonts.Times, 12);
                        Brush brush = new SolidBrush();

                        // Add Email contents
                        int topMargin = 20;
                        page.Canvas.DrawString({code}quot;File Name: {msg.FileName}", 
                        page.Canvas.DrawString({code}quot;From: {from}", font, brush, 
                        page.Canvas.DrawString({code}quot;Sent On: {(sentOn.HasValue ?
                        page.Canvas.DrawString({code}quot;To: {recipientsTo}", font, br

                        if (!string.IsNullOrEmpty(recipientsCc))
                        {
                            page.Canvas.DrawString({code}quot;CC: {recipientsCc}", font
                        }
```

```
                    page.Canvas.DrawString({code}quot;Subject: {subject}", font, br
                    page.Canvas.DrawString("Message body and attachments in next pa

                    // Convert Html body to PDF in order to retain all formatting.
                    using (HtmlToPdfConverter converter = new HtmlToPdfConverter())
                    {
                        converter.PageSize = PaperKind.A4;
                        converter.Orientation = Bytescout.PDF.Converters.PaperOrien

                        // Convert input HTML to stream
                        byte[] byteArrayBody = Encoding.UTF8.GetBytes(msgBody);
                        MemoryStream inputStream = new MemoryStream(byteArrayBody)

                        // Create output stream to store generated PDF file
                        MemoryStream outputStream = new MemoryStream();

                        // Convert HTML to PDF
                        converter.ConvertHtmlToPdf(inputStream, outputStream);

                        // Create new document from generated output stream
                        Document docContent = new Document(outputStream);

                        // Append all pages to main PDF
                        foreach (Page item in docContent.Pages)
                        {
                            outputDocument.Pages.Add(item);
                        }

                        // Get attachments from message (if any, and append to docu
                        if (msg.Attachments.Count > 0)
                        {
                            foreach (MsgReader.Outlook.Storage.Attachment itmAttach
                            {
                                // Get Memory Stream
                                MemoryStream attachmentMemoryStream = new MemorySt

                                // Append Attachment
                                Document docAttachment = new Document(attachmentMem

                                // Append all pages to main PDF
                                foreach (Page item in docAttachment.Pages)
                                {
                                    outputDocument.Pages.Add(item);
                                }
                            }
                        }

                        // Save output file
                        outputDocument.Save("result.pdf");
                    }

                    // Open result document in default associated application (for
                    ProcessStartInfo processStartInfo = new ProcessStartInfo("resu
                    processStartInfo.UseShellExecute = true;
                    Process.Start(processStartInfo);
                }
            }
        }
        catch (Exception ex)
        {
```

```
                Console.WriteLine(ex.Message);
                Console.WriteLine("Press enter key to exit...");
                Console.ReadLine();
            }
        }

    }
}
```

packages.config

```
<?xml version="1.0" encoding="utf-8"?>
<packages>
  <package id="MsgReader" version="3.4.0" targetFramework="net45" />
  <package id="OpenMcdf" version="2.2.1.3" targetFramework="net45" />
</packages>
```

VIDEO

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