How to convert HTML email with attachments to PDF in C# with ByteScout PDF SDK

This tutorial will show how to convert HTML email with attachments to PDF in C#

Learn how to convert HTML email with attachments to PDF in C# with this source code sample. ByteScout PDF SDK is the pdf library that can create, update and modify PDF files. Supports text with fonts and style selections, layers, form fields, drawing lines and objects, automatic tables, images. Can be used to create and fill pdf forms and you can use it to convert HTML email with attachments to PDF with C#.

Fast application programming interfaces of ByteScout PDF SDK for C# plus the instruction and the code below will help you quickly learn how to convert HTML email with attachments to PDF. Follow the instructions from the scratch to work and copy the C# code. Test C# sample code examples whether they respond your needs and requirements for the project.

ByteScout free trial version is available for download from our website. It includes all these programming tutorials along with source code samples.

C# - 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/Sicos1977/MSGReader
                using (var msg = new
MsgReader.Outlook.Storage.Message("HtmlSampleEmailWithAttachment.msg"))
                {
```

```
                    // 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.RecipientType.To, false, false);

                    // Recipient CC information
                    var recipientsCc =
msg.GetEmailRecipients(MsgReader.Outlook.RecipientType.Cc, false, false);

                    // 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($"File Name: {msg.FileName}", font,
brush, 20, (topMargin += 20));
                        page.Canvas.DrawString($"From: {from}", font, brush, 20,
(topMargin += 20));
                        page.Canvas.DrawString($"Sent On: {(sentOn.HasValue ?
sentOn.Value.ToString("MM/dd/yyyy HH:mm") : "")}", font, brush, 20, (topMargin +=
20));
                        page.Canvas.DrawString($"To: {recipientsTo}", font, brush,
20, (topMargin += 20));

                        if (!string.IsNullOrEmpty(recipientsCc))
                        {
                            page.Canvas.DrawString($"CC: {recipientsCc}", font,
brush, 20, (topMargin += 20));
                        }

                        page.Canvas.DrawString($"Subject: {subject}", font, brush,
20, (topMargin += 20));
                        page.Canvas.DrawString("Message body and attachments in next
page.", font, brush, 20, (topMargin += 20));

                        // 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.PaperOrientation.Portrait;

                            // 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
document)
                            if (msg.Attachments.Count > 0)
                            {
                                foreach (MsgReader.Outlook.Storage.Attachment
itmAttachment in msg.Attachments)
                                {
                                    // Get Memory Stream
                                    MemoryStream attachmentMemoryStream = new
MemoryStream(itmAttachment.Data);

                                    // Append Attachment
                                    Document docAttachment = new
Document(attachmentMemoryStream);

                                    // 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 demo purpose)
                        ProcessStartInfo processStartInfo = new
ProcessStartInfo("result.pdf");
                        processStartInfo.UseShellExecute = true;
                        Process.Start(processStartInfo);
```

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

    }
}
```

C# - 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.CodePagesEncodingProvider.Instance;
                Encoding.RegisterProvider(encodingProvider);

                // Parse MessageContents using MsgReader Library
                // MsgReader library can be obtained from:
https://github.com/Sicos1977/MSGReader
                using (var msg = new
MsgReader.Outlook.Storage.Message("HtmlSampleEmailWithAttachment.msg"))
                {
                    // 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.RecipientType.To, false, false);

                    // Recipient CC information
                    var recipientsCc =
msg.GetEmailRecipients(MsgReader.Outlook.RecipientType.Cc, false, false);

                    // 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($"File Name: {msg.FileName}", font,
brush, 20, (topMargin += 20));
                        page.Canvas.DrawString($"From: {from}", font, brush, 20,
(topMargin += 20));
                        page.Canvas.DrawString($"Sent On: {(sentOn.HasValue ?
sentOn.Value.ToString("MM/dd/yyyy HH:mm") : "")}", font, brush, 20, (topMargin +=
20));
                        page.Canvas.DrawString($"To: {recipientsTo}", font, brush,
20, (topMargin += 20));

                        if (!string.IsNullOrEmpty(recipientsCc))
                        {
                            page.Canvas.DrawString($"CC: {recipientsCc}", font,
brush, 20, (topMargin += 20));
                        }

                        page.Canvas.DrawString($"Subject: {subject}", font, brush,
20, (topMargin += 20));
                        page.Canvas.DrawString("Message body and attachments in next
page.", font, brush, 20, (topMargin += 20));

                        // 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.PaperOrientation.Portrait;

                            // 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
document)
                            if (msg.Attachments.Count > 0)
                            {
                                foreach (MsgReader.Outlook.Storage.Attachment
itmAttachment in msg.Attachments)
                                {
                                    // Get Memory Stream
                                    MemoryStream attachmentMemoryStream = new
MemoryStream(itmAttachment.Data);

                                    // Append Attachment
                                    Document docAttachment = new
Document(attachmentMemoryStream);

                                    // 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 demo purpose)
                        ProcessStartInfo processStartInfo = new
ProcessStartInfo("result.pdf");
                        processStartInfo.UseShellExecute = true;
                        Process.Start(processStartInfo);
                    }
                }
            }
            catch (Exception ex)
            {
```

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

    }
}
```

C# - packages.config

```
```

FOR MORE INFORMATION AND FREE TRIAL:

[Download Free Trial SDK (on-premise version)]

[Read more about ByteScout PDF SDK]

[Explore documentation]

[Visit www.ByteScout.com]

or

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