# GemBox Support Center



## Save all worksheets in a single HTML

Mario - GemBox - 2017-02-09 - 0 Comments - in GemBox.Spreadsheet

When using the GemBox.Spreadsheet to generate an HTML file from our spreadsheet file, we can define an HtmlType which we require.
A default HtmlType.Html will export all the spreadsheet content into a several files (like the main page, worksheet frames, pictures, etc.).

If we want to export the content of the entire spreadsheet into a single file (together with images), we can use a single MIME HTML (MHTML) format file, for example:

### C# code

```
ExcelFile ef = ExcelFile.Load("Sample.xlsx");
HtmlSaveOptions options = new HtmlSaveOptions() { SelectionType = SelectionType.Entire
File, HtmlType = HtmlType.Mhtml };
ef.Save("Sample.mhtml", options);
```

### VB.NET code

```
Dim ef As ExcelFile = ExcelFile.Load("Sample.xlsx")
Dim options As HtmlSaveOptions = New HtmlSaveOptions() With {.SelectionType = Selectio
nType.EntireFile, .HtmlType = HtmlType.Mhtml}
ef.Save("Sample.mhtml", options)
```

But if we want to export all the worksheets data into a single HTML file, we can use an HtmType.HtmlTable in order to create a file according to our requirements.

For example, we can loop through every sheet in the file, set it as an active sheet and then save it by using the HtmlType.HtmlTable and SelectionType.ActiveSheet.
Now that we have every sheet as an HTML table, we can create an HTML file and arrange HTML tables in some custom manner.

Here is an extension method which will do a simple merge of all sheets into a single file:

### C# code

```
public static class GemBoxSpreadsheetHelper
{
    public static void SaveAsSingleHtml(this ExcelFile ef, string filePath)
    {
        StringBuilder singleHtmlFileContent = new StringBuilder();
        string htmlTopContent = "<!DOCTYPE html><html><body>";
        string htmlBottomContent = "</body></html>";

        int worksheetCount = ef.Worksheets.Count;
        HtmlSaveOptions saveOption = new HtmlSaveOptions() { SelectionType = SelectionTy
pe.ActiveSheet, HtmlType = HtmlType.HtmlTable };

        singleHtmlFileContent.Append(htmlTopContent);
        for (int i = 0; i < worksheetCount; i++)
        {
```

```
            ef.Worksheets.ActiveWorksheet = ef.Worksheets[i];
            using (var stream = new MemoryStream())
            {
                ef.Save(stream, saveOption);
                singleHtmlFileContent.Append(saveOption.Encoding.GetString(stream.ToArray()));
            }
        }
        singleHtmlFileContent.Append(htmlBottomContent);

        File.WriteAllText(filePath, singleHtmlFileContent.ToString(), saveOption.Encoding);
    }
}
```

### VB.NET code

```
Module GemBoxSpreadsheetHelper
    <System.Runtime.CompilerServices.Extension>
    Public Sub SaveAsSingleHtml(ef As ExcelFile, filePath As String)
        Dim singleHtmlFileContent As New StringBuilder()
        Dim htmlTopContent As String = "<!DOCTYPE html><html><body>"
        Dim htmlBottomContent As String = "</body></html>"

        Dim worksheetCount As Integer = ef.Worksheets.Count
        Dim saveOption As HtmlSaveOptions = New HtmlSaveOptions() With {.SelectionType
= SelectionType.ActiveSheet, .HtmlType = HtmlType.HtmlTable}

        singleHtmlFileContent.Append(htmlTopContent)
        For i As Integer = 0 To worksheetCount - 1
            ef.Worksheets.ActiveWorksheet = ef.Worksheets(i)
            Using stream = New MemoryStream()
                ef.Save(stream, saveOption)
                singleHtmlFileContent.Append(saveOption.Encoding.GetString(stream.ToArray()))
            End Using
        Next
        singleHtmlFileContent.Append(htmlBottomContent)

        File.WriteAllText(filePath, singleHtmlFileContent.ToString(), saveOption.Encoding)
    End Sub
End Module
```

And here is how we can use it:

### C# code

```
ExcelFile ef = ExcelFile.Load("Sample.xlsx");
ef.SaveAsSingleHtml("Sample.html");
```

### VB.NET code

```
Dim ef As ExcelFile = ExcelFile.Load("Sample.xlsx")
ef.SaveAsSingleHtml("Sample.html")
```

Also, if we need to stream this single HTML file to the client's browser, we can adjust our extension method as follows:

### C# code

```
public static class GemBoxSpreadsheetHelper
{
    public static void SaveAsSingleHtml(this ExcelFile ef, HttpResponse response)
    {
```

```
        // ...

        //File.WriteAllText(filePath, singleHtmlFileContent.ToString(), saveOption.Encoding);
        response.Write(singleHtmlFileContent.ToString());
        response.Flush();
        response.Close();
        response.End();
    }
}
```

### VB.NET code

```
Module GemBoxSpreadsheetHelper
    <System.Runtime.CompilerServices.Extension>
    Public Sub SaveAsSingleHtml(ef As ExcelFile, response As HttpResponse)
        ' ...

        'File.WriteAllText(filePath, singleHtmlFileContent.ToString(), saveOption.Encoding)
        response.Write(singleHtmlFileContent.ToString())
        response.Flush()
        response.Close()
        response.End()
    End Sub
End Module
```

And use it like this:

### C# code

```
ExcelFile ef = ExcelFile.Load(Path.Combine(Server.MapPath(""), "Sample.xlsx"));
ef.SaveAsSingleHtml(this.Response);
```

### VB.NET code

```
Dim ef As ExcelFile = ExcelFile.Load(Path.Combine(Server.MapPath(""), "Sample.xlsx"))
ef.SaveAsSingleHtml(Me.Response)
```

Attachments

- GemBoxSpreadsheetHelper.cs [2.21 KB]
- GemBoxSpreadsheetHelper.vb [2.28 KB]