GemBox Support Center

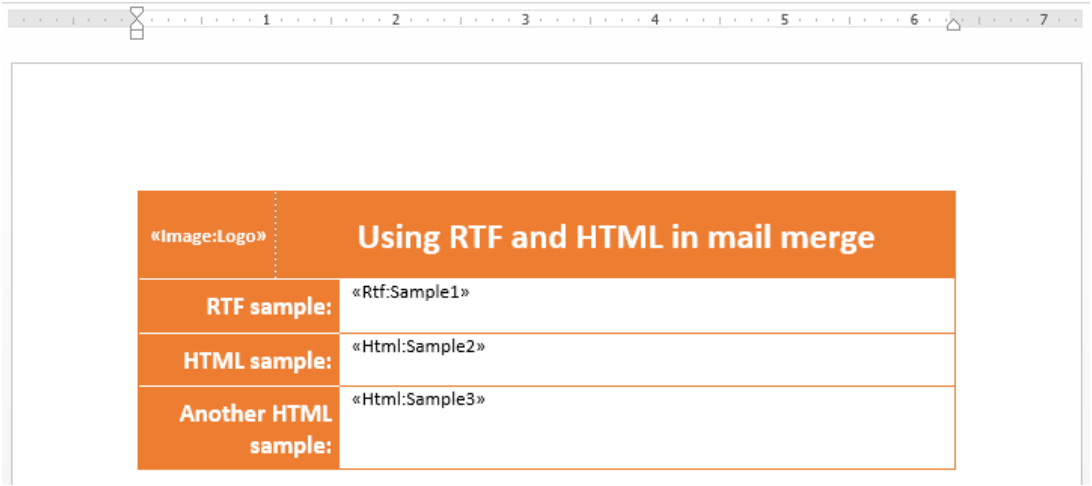
How to insert HTML and RTF content during the mail merge process

Mario - GemBox - 2018-07-18 - 3 Comments - in GemBox.Document

The following sample will demonstrate how you can insert HTML and RTF formatted text in a mail merge process.

Additionally, this sample will demonstrate how to use a FieldMappings collection together with FieldMerging event to define custom MergeFields. This will be done by specifying an arbitrary prefix (such as "Html:", "Rtf:", "Image:", etc.) for the MergeField's name and then implementing merging customizations for the appropriate MergeFields.

The following is a used sample input file ("RTF & HTML in mail merge - Input.docx"):



We'll map merge fields that have any of our custom prefixes ("Rtf:", "Html:", "Image:") to data source names that are equal to field names without prefixes. Also, we'll customize the mail merge operation by using FieldMerging event to implement our custom prefixes:

- Fields with an "Image:" prefix will insert a Picture element from the provided byte array.
- Fields with an "Rtf:" prefix will insert an RTF formatted text from the provided string.
- Fields with an "Html:" prefix will insert a HTML formatted text from the provided string.

C# code

```
DocumentModel document = DocumentModel.Load("RTF & HTML in mail merge - Input.docx");

// Map merge fields, e.g. "Rtf:Sample1" field's name to "Sample1" data source name.
foreach (string fieldName in document.MailMerge.GetMergeFieldNames())
{
```

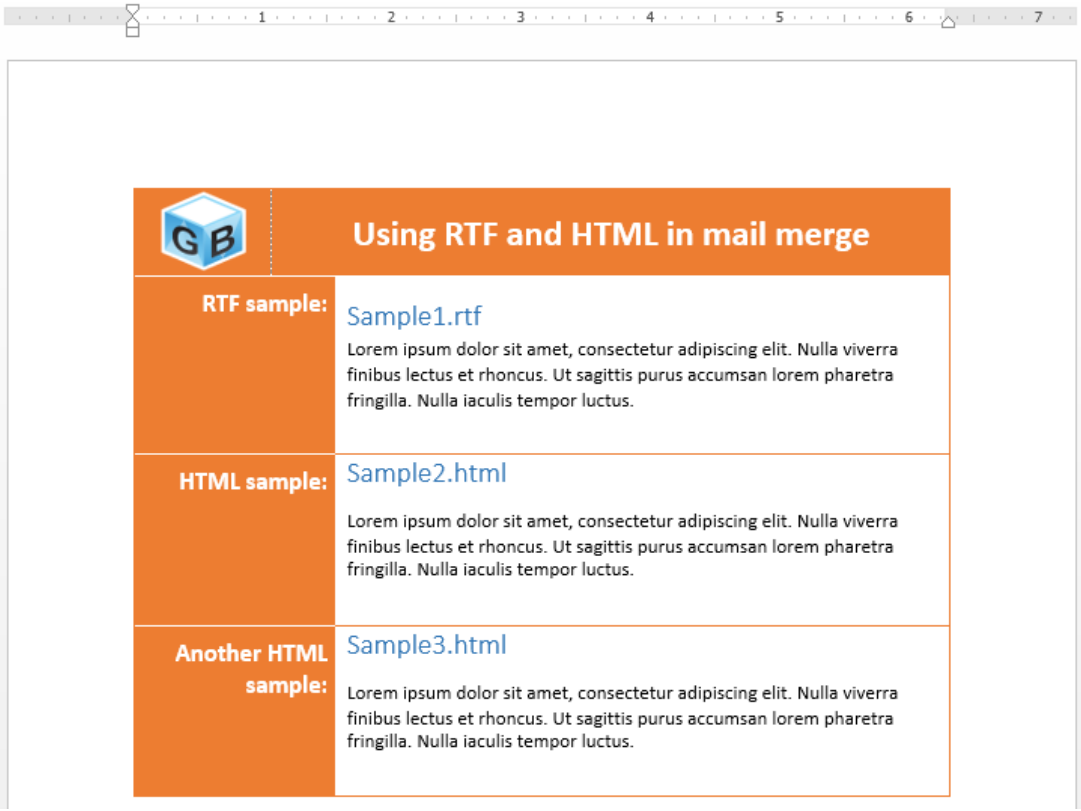
```
    int index = fieldName.IndexOf(':');
    if (index > 0)
        document.MailMerge.FieldMappings.Add(fieldName, fieldName.Substring(index + 1));
}

// Customize mail merge operation with a support for our custom prefixes.
document.MailMerge.FieldMerging += (sender, e) =>
{
    if (!e.IsValueFound)
        return;

    if (e.FieldName.StartsWith("Image:"))
        e.Inline = new Picture(e.Document, new MemoryStream((byte[])e.Value));

    else if (e.FieldName.StartsWith("Rtf:"))
    {
        e.Field.Content.End.LoadText((string)e.Value, LoadOptions.RtfDefault);
        e.Inline = null;
    }

    else if (e.FieldName.StartsWith("Html:"))
    {
        e.Field.Content.End.LoadText((string)e.Value, LoadOptions.HtmlDefault);
        e.Inline = null;
    }
};

// Execute mail merge operation, for demonstrations purposes
// we'll use a simple anonymous type as a data source.
document.MailMerge.Execute(
    new
    {
        Logo = File.ReadAllBytes("Logo.png"),
        Sample1 = File.ReadAllText("Sample1.rtf"),
        Sample2 = File.ReadAllText("Sample2.html"),
        Sample3 = File.ReadAllText("Sample3.html")
    });

document.Save("RTF & HTML in mail merge - Output.docx");
```

VB.NET code

```
Dim document As DocumentModel = DocumentModel.Load("RTF & HTML in mail merge -
 Input.docx")

' Map merge fields, e.g. "Rtf:Sample1" field's name to "Sample1" data source name.
For Each fieldName As String In document.MailMerge.GetMergeFieldNames()
    Dim index As Integer = fieldName.IndexOf(":"c)
    If index > 0 Then
        document.MailMerge.FieldMappings.Add(fieldName, fieldName.Substring(index + 1))
    End If
Next

' Customize mail merge operation with a support for our custom prefixes.
AddHandler document.MailMerge.FieldMerging, _
    Sub(sender, e)
        If Not e.IsValueFound Then
            Return
        End If
```

```
        If e.FieldName.StartsWith("Image:") Then
            e.Inline = New Picture(e.Document, New MemoryStream(DirectCast(e.Value, Byte())
))

        ElseIf e.FieldName.StartsWith("Rtf:") Then
            e.Field.Content.End.LoadText(DirectCast(e.Value, String), LoadOptions.RtfDefault)
            e.Inline = Nothing

        ElseIf e.FieldName.StartsWith("Html:") Then
            e.Field.Content.End.LoadText(DirectCast(e.Value, String), LoadOptions.HtmlDefault)
            e.Inline = Nothing
        End If
    End Sub

' Execute mail merge operation, for demonstrations purposes
' we'll use a simple anonymous type as a data source.
document.MailMerge.Execute(
    New With { _
        .Logo = File.ReadAllBytes("Logo.png"), _
        .Sample1 = File.ReadAllText("Sample1.rtf"), _
        .Sample2 = File.ReadAllText("Sample2.html"), _
        .Sample3 = File.ReadAllText("Sample3.html") _
    })

document.Save("RTF & HTML in mail merge - Output.docx")
```

The following is the resulting "RTF & HTML in mail merge - Output.docx" file:



Comments (3)

Jeroen Penninx
Mon, 2nd Jul 2018
9:32am

This works well, except the font face get's lost. So bold, italic from the html appear in the word file, but even though my template en

the specific merge fields define Calibri as the font, the merged text will be in Times new roman.

Is there any way to for the entire document to use the same font face?

We could do this, but then all bold, italic formatting etc gets lost:

```
var fields = document.GetChildElements(true,
ElementType.Run).ToList();
foreach (Run run in fields)
{
run.CharacterFormat = new CharacterFormat
{
FontName = "Calibri"
};
}
```

Mario - GemBox
Mon, 2nd Jul 2018 10:33am

Hi,

Note, there are few LoadText overload methods: https://www.gemboxsoftware.com/document/help/html/Overload_GemBox_Document_ContentRange_LoadText.htm

Currently, only when using "LoadText(String)" the loaded content will have an inherited formatting (for example, in your case it would inherit the MergeField's formatting).

But when using any other LoadText method, in which you're explicitly specifying the formatting that should be used, then a new content will be generated from the specified input text and the provided format.

In other words, when using "LoadText(String, HtmlLoadOptions)" the new content is being generated based on only the provided input HTML text, it does not matter what MergeField's formatting is.

Nevertheless, you could try using something like the following in order to obtain the desired inheritance.

- REMOVE THE FOLLOWING:

```
else if (e.FieldName.StartsWith("Html:"))
{
e.Field.Content.End.LoadText((string)e.Value,
LoadOptions.HtmlDefault);
e.Inline = null;
}
```

- ADD THE FOLLOWING:

```
else if (e.FieldName.StartsWith("Html:"))
{
CharacterFormat format = e.Field.CharacterFormat;

string html = (string)e.Value;
string htmlStyle = string.Format(
"&lt;style&gt;* {{ font-family:{0}; font-size:{1}pt; font-
weight:{2}; font-style:{3}; }}&lt;/style&gt;",
format.FontName,
format.Size,
format.Bold ? "bold" : "normal",
format.Italic ? "italic" : "normal");

e.Field.Content.End.LoadText(htmlStyle + html,
LoadOptions.HtmlDefault);
e.Inline = null;
}
```

I hope this helps.

Regards,

Mario
GemBox Ltd.

Jeroen Penninx
Mon, 2nd Jul 2018
1:36pm

Thank you for your prompt reaction. Unfortunately the [style] directive rendered literally in my document. It did inspire me to a bit different solution though. See below. Basically I added a div around my content with the desired style info. Thank you again.

```
else if (e.FieldName.StartsWith("Html:"))
{
var format = e.Field.CharacterFormat;

var html = (string)e.Value;
var htmlStyle = string.Format(
"<div st yle='font-family:{0}; font-size:{1}pt; font-weight:{2};
font-style:{3};'>{4}</div>",
format.FontName,
format.Size,
format.Bold ? "bold" : "normal",
format.Italic ? "italic" : "normal",
html);

e.Field.Content.End.LoadText(htmlStyle,
LoadOptions.HtmlDefault);
e.Inline = null;
}
```