GemBox Support Center

Working with document file stream

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

GemBox.Document provides few overload Load methods and overload Save methods. These methods enable us to work with a physical file (when providing a file's path) or with an in-memory file (when providing a file's stream).

The following is a small demonstration sample of these methods:

C# code

```
DocumentModel document;

// Create input file's stream.
using (Stream input = File.OpenRead("Input.docx"))
    // Load input file from stream.
    document = DocumentModel.Load(input, LoadOptions.DocxDefault);

// Edit document by adding new content.
document.Sections[0].Blocks.Insert(0, new Paragraph(document, "Hello World!"));

// Create output file's stream.
using (FileStream output = File.Create("Output.docx"))
    // Save output file to stream.
    document.Save(output, SaveOptions.DocxDefault);
```

VB.NET code

```
Dim document As DocumentModel

' Create input file's stream.
Using input As Stream = File.OpenRead("Input.docx")
    ' Load input file from stream.
    document = DocumentModel.Load(input, LoadOptions.DocxDefault)
End Using

' Edit document by adding new content.
document.Sections(0).Blocks.Insert(0, New Paragraph(document, "Hello World!"))

' Create output file's stream.
Using output As FileStream = File.Create("Output.docx")
    ' Save output file to stream.
    document.Save(output, SaveOptions.DocxDefault)
End Using
```

Also, among the GemBox.Document's overload Save methods there are some that take an object type as the first parameter. These methods are used for direct streaming of an output file to the web application's client. This object parameter can be of the System.Web.HttpResponse or System.Web.HttpResponseBase type; note that this parameter is by design of an object type in order to avoid GemBox.Document's dependency with System.Web assembly.

Here is an example of streaming a file to the client's browser in ASP.NET Web Forms application:

C# code

```
public partial class _Default : Page
{
    protected void Page_Load(object sender, EventArgs e)
    { }

    protected void Button1_Click(object sender, EventArgs e)
    {
        DocumentModel document = new DocumentModel();

        document.Sections.Add(
            new Section(document,
                new Paragraph(document, "Hello World!")));

        document.Save(this.Response, "Output.docx");
    }
}
```

VB.NET code

```
Public Class _Default
    Inherits Page

    Protected Sub Page_Load(ByVal sender As Object, ByVal e As EventArgs) Handles Me.Loa
d
    End Sub

    Protected Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        Dim document = New DocumentModel()

        document.Sections.Add( _
            New Section(document, _
                New Paragraph(document, "Hello World!")))

        document.Save(Me.Response, "Output.docx")
    End Sub
End Class
```

And here is an example in ASP.NET MVC application:

C# code

```
public class HomeController : Controller
{
    public ActionResult Index()
    {
        return View();
    }

    public void Download()
    {
        DocumentModel document = new DocumentModel();

        document.Sections.Add(
            new Section(document,
                new Paragraph(document, "Hello World!")));
```

```
        document.Save(this.Response, "Output.docx");
    }
}
```

VB.NET code

```
Public Class HomeController
    Inherits Controller

    Function Index() As ActionResult
        Return View()
    End Function

    Sub Download()
        Dim document = New DocumentModel()

        document.Sections.Add( _
            New Section(document, _
                New Paragraph(document, "Hello World!")))

        document.Save(Me.Response, "Output.docx")
    End Sub
End Class
```