GemBox Support Center

How to use GemBox.Spreadsheet in Classic ASP

Mario - GemBox - 2017-02-10 - 0 Comments - in GemBox.Spreadsheet

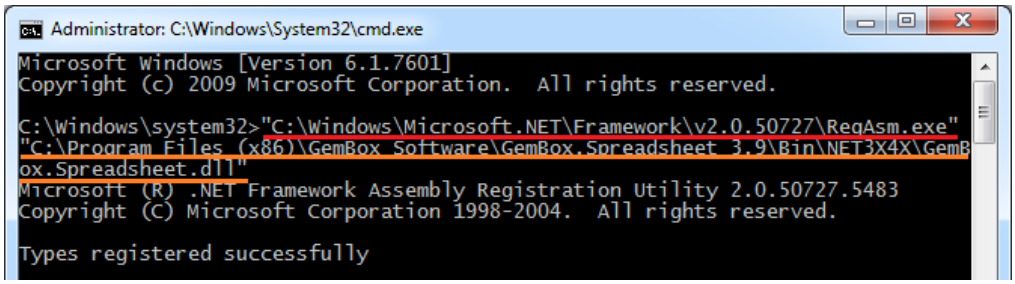
GemBox.Spreadsheet is a utility library for .NET applications (Windows Forms, WPF, ASP.NET, etc.), but it can also be used from within a Classic ASP web page. To do this, we need to register GemBox.Spreadsheet to COM with the following steps:

1. Download and install the latest version of GemBox.Spreadsheet from this page.
2. Open the Command Prompt (CMD) as Administrator (see here how to do this).
3. Register GemBox.Spreadsheet to COM by running the following command in CMD:

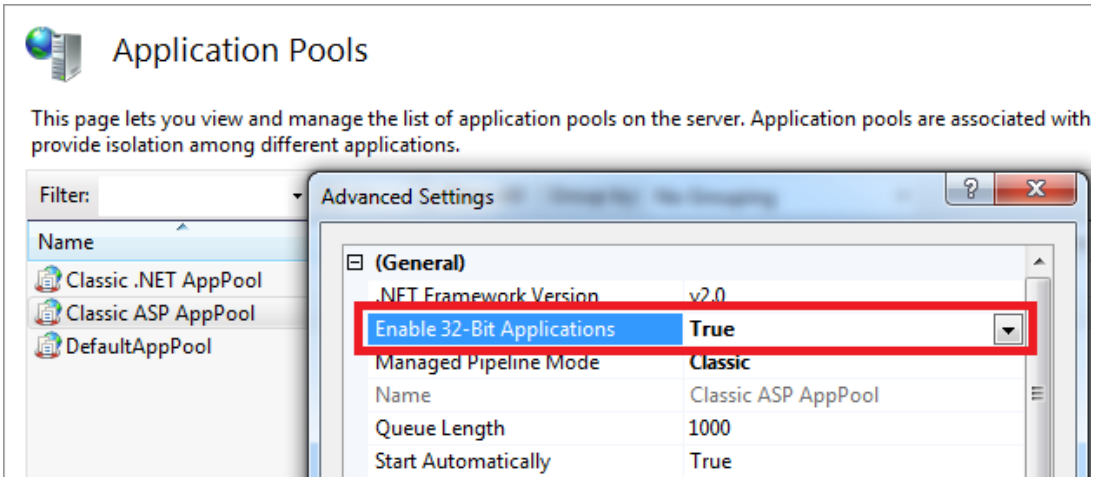
   "*<path to RegAsm.exe>*" "*<path to GemBox.Spreadsheet.dll>*"

   E.g:

   "*C:\Windows\Microsoft.NET\Framework\v2.0.50727\RegAsm.exe*" "*C:\Program Files (x86)\GemBox Software\GemBox.Spreadsheet 3.9\Bin\NET3X4X\GemBox.Spreadsheet.dll*"



Now we can start using GemBox.Spreadsheet through VBScript code.
NOTE: To create a COM object(s) from a GemBox.Spreadsheet assembly on x64 OS, you'll need to ensure that the Application Pool that your website is using has the "Enable 32-Bit Applications" option enabled:



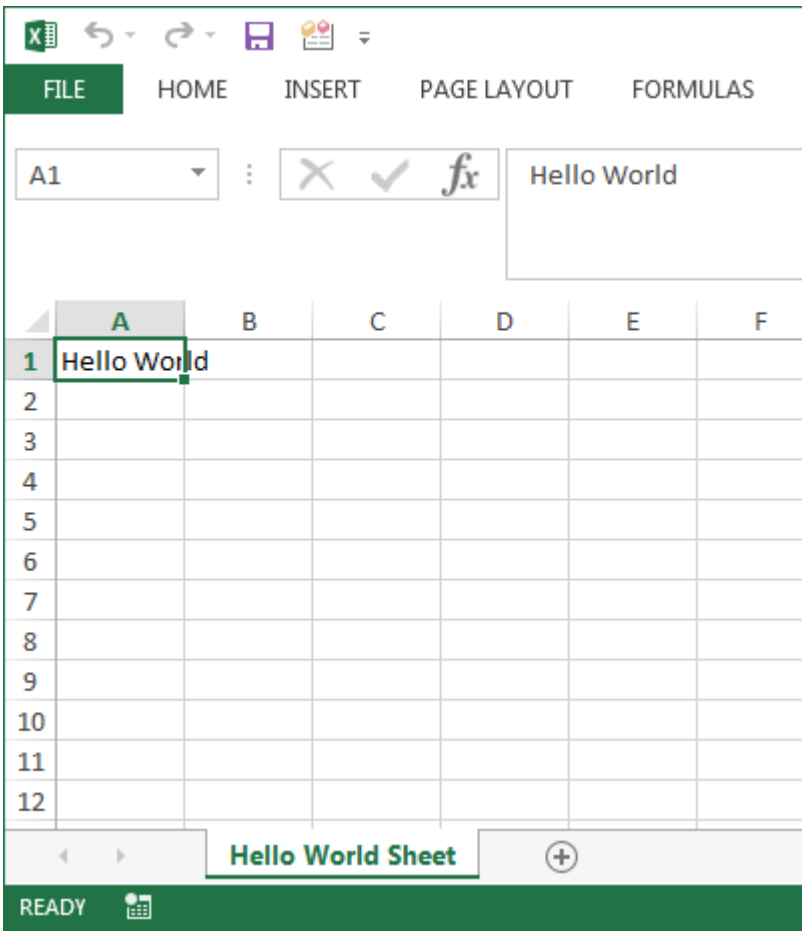
Sample 1: Create an Excel file in VBScript

The following sample demonstrates how we can create a new Excel file:

```
<%
  ' Create ComHelper object.
  Set comHelper = CreateObject("GemBox.Spreadsheet.ComHelper")
  ' Set license.
  ' NOTE: If you're using a Professional version you'll need to put
your serial key below.
  comHelper.ComSetLicense("FREE-LIMITED-KEY")

  ' Create new excel file.
  Set workbook = CreateObject("GemBox.Spreadsheet.ExcelFile")
  ' Create new sheet.
  Set worksheet = workbook.Worksheets.Add("Hello World Sheet")
  ' Get A1 cell.
  Set cell = worksheet.Cells.Item(0)
  ' Set A1 cell's value.
  cell.Value = "Hello World"
  ' Save "HelloWorld.xlsx" file.
  workbook.Save(Server.MapPath(".") & "\HelloWorld.xlsx")
%>
```

The resulting HelloWorld.xlsx file:



Sample 2: Read and Write Excel file in VBScript

The following sample demonstrates how we can load an existing Excel file, update it with some new data, and save it as a PDF file:

```
<%
  ' Create ComHelper object.
  Set comHelper = CreateObject("GemBox.Spreadsheet.ComHelper")
  ' Set license.
  ' NOTE: If you're using a Professional version you'll need to put
your serial key below.
  comHelper.ComSetLicense("FREE-LIMITED-KEY")

  ' Load excel file.
  Set workbook = comHelper.Load(Server.MapPath(".") & "\Input.xlsx")
  ' Get first sheet.
  Set worksheet = workbook.Worksheets.Item(0)

  ' Create new data.
  Set newData = Server.CreateObject("Scripting.Dictionary")
  newData.Add "Bob Garvey", 1000
```

```
  newData.Add "Ben Stilwell", 2000
  newData.Add "Peter Pan", 3000
  ' Get cells from "A6" to "B8".
  Set cellsRange = worksheet.Cells.GetSubrange("A6", "B8")
  ' Update cells with new data.
  Dim cellIndex
  cellIndex = 0
  For Each name In newData
    cellsRange.Item(cellIndex).Value = name
    cellIndex = cellIndex + 1
    cellsRange.Item(cellIndex).Value = newData(name)
    cellIndex = cellIndex + 1
  Next
  ' Save "Output.pdf" file.
  workbook.Save(Server.MapPath(".") & "\Output.pdf")
%>
```

Used input Input.xlsx file:

The resulting Output.pdf file:

Name	Salary
John Doe	$3,472
Fred Nurk	$3,061
Hans Meier	$3,006
Ivan Horvat	$4,848
Bob Garvey	$1,000
Ben Stilwell	$2,000
Peter Pan	$3,000