

PDF Combiner

A simple Windows Forms app to combine multiple PDF files in a single PDF file

Table of Contents

Symbol Reference	1
PdfCombiner Namespace	1
PdfCombiner.Properties Namespace	1
Classes	1
Resources Class	1
Settings Class	3
Classes	4
BaseException Class	4
BaseException Constructor	5
ExceptionExtensions Class	5
ExceptionExtensions Methods	6
FrmMain Class	6
FrmMain.FrmMain Constructor	8
FrmMain Fields	8
FrmMain Methods	10
Program Class	21
Program Methods	21
Files	21
AssemblyInfo.cs	22
Exceptions.cs	22
FrmMain.cs	23
FrmMain.Designer.cs	30
PdfCombiner.csproj	33
PdfCombiner.sln	33
Program.cs	33
Resources.Designer.cs	34
Settings.Designer.cs	35
Index	a

1 Symbol Reference





1.1 PdfCombiner Namespace

This is namespace PdfCombiner.

Namespaces

Name	Description
Properties (see page 1)	This is namespace PdfCombiner.Properties.



Classes

	Name	Description
	BaseException (see page 4)	This function is used to take exceptions which can be thrown in PDF Combiner App
	ExceptionExtensions (see page 5)	This is class PdfCombiner.ExceptionExtensions.
	FrmMain (see page 6)	This is class PdfCombiner.FrmMain.
	Program (see page 21)	This is class PdfCombiner.Program.

1.1.1 PdfCombiner.Properties Namespace

This is namespace PdfCombiner.Properties.



Classes

	Name	Description
	Resources (see page 1)	A strongly-typed resource class, for looking up localized strings, etc.
	Settings (see page 3)	This is class PdfCombiner.Properties.Settings.

1.1.1.1 Classes

The following table lists classes in this documentation.

Classes

	Name	Description
	Resources (see page 1)	A strongly-typed resource class, for looking up localized strings, etc.
	Settings (see page 3)	This is class PdfCombiner.Properties.Settings.

1.1.1.1.1 Resources Class

A strongly-typed resource class, for looking up localized strings, etc.

Class Hierarchy

PdfCombiner.Properties.Resources

C#

```
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Resources.Tools.StronglyTypedResourceBuilder",
"15.0.0.0")]
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.Runtime.CompilerServices.CompilerGeneratedAttribute()]
internal class Resources;
```


File

Resources.Designer.cs ([see page 34](#))



Description

This class was auto-generated by the StronglyTypedResourceBuilder class via a tool like ResGen or Visual Studio. To add or remove a member, edit your .ResX file then rerun ResGen with the /str option, or rebuild your VS project.



Methods

	Name	Description
	Resources (see page 2)	This is Resources, a member of class Resources.

Resources Fields

	Name	Description
	resourceCulture (see page 2)	This is resourceCulture, a member of class Resources.
	resourceMan (see page 3)	This is resourceMan, a member of class Resources.

Resources Properties

	Name	Description
	Culture (see page 3)	Overrides the current thread's CurrentUICulture property for all resource lookups using this strongly typed resource class.
	ResourceManager (see page 3)	Returns the cached ResourceManager instance used by this class.

1.1.1.1.1.1 Resources.Resources Constructor**C#**

```
[global::System.Diagnostics.CodeAnalysis.SuppressMessageAttribute("Microsoft.Performance",
"CA1811:AvoidUncalledPrivateCode")]
internal Resources();
```

Description

This is Resources, a member of class Resources.

Body Source

```
1:
[global::System.Diagnostics.CodeAnalysis.SuppressMessageAttribute("Microsoft.Performance",
"CA1811:AvoidUncalledPrivateCode")]
2: internal Resources() {
3: }
```

1.1.1.1.1.2 Resources Fields**1.1.1.1.1.2.1 Resources.resourceCulture Field****C#**

```
private static global::System.Globalization.CultureInfo resourceCulture;
```

Description

This is resourceCulture, a member of class Resources.

1.1.1.1.1.2.2 Resources.resourceMan Field

C#

```
private static global::System.Resources.ResourceManager resourceMan;
```

Description

This is resourceMan, a member of class Resources.

1.1.1.1.1.3 Resources Properties

1.1.1.1.1.3.1 Resources.Culture Property

Overrides the current thread's CurrentUICulture property for all resource lookups using this strongly typed resource class.

C#

```
[global::System.ComponentModel.EditorBrowsableAttribute(global::System.ComponentModel.EditorBrowsableState.Advanced)]
internal static global::System.Globalization.CultureInfo Culture;
```

1.1.1.1.1.3.2 Resources.ResourceManager Property

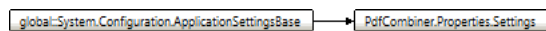
Returns the cached ResourceManager instance used by this class.

C#

```
[global::System.ComponentModel.EditorBrowsableAttribute(global::System.ComponentModel.EditorBrowsableState.Advanced)]
internal static global::System.Resources.ResourceManager ResourceManager;
```

1.1.1.1.2 Settings Class

Class Hierarchy



C#

```
[global::System.Runtime.CompilerServices.CompilerGeneratedAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("Microsoft.VisualStudio.Editors.SettingsDesigner.SettingsSingleFileGenerator",
"11.0.0.0")]
internal sealed class Settings : global::System.Configuration.ApplicationSettingsBase;
```

File

Settings.Designer.cs ([see page 35](#))

Description

This is class PdfCombiner.Properties.Settings.

Settings Fields

	Name	Description
	defaultInstance (see page 4)	This is defaultInstance, a member of class Settings.

Settings Properties

	Name	Description
	Default (see page 4)	This is Default, a member of class Settings.

1.1.1.1.2.1 Settings Fields

1.1.1.1.2.1.1 Settings.defaultInstance Field

C#

```
private static Settings defaultInstance =  
((Settings)(global::System.Configuration.ApplicationSettingsBase.Synchronized(new  
Settings())));
```

Description

This is defaultInstance, a member of class Settings.

1.1.1.1.2.2 Settings Properties

1.1.1.1.2.2.1 Settings.Default Property

C#

```
public static Settings Default;
```





Description

This is Default, a member of class Settings.

1.1.2 Classes

The following table lists classes in this documentation.

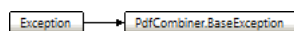
Classes

	Name	Description
	BaseException (see page 4)	This function is used to take exceptions which can be thrown in PDF Combiner App
	ExceptionExtensions (see page 5)	This is class PdfCombiner.ExceptionExtensions.
	FrmMain (see page 6)	This is class PdfCombiner.FrmMain.
	Program (see page 21)	This is class PdfCombiner.Program.

1.1.2.1 BaseException Class

This function is used to take exceptions which can be thrown in PDF Combiner App

Class Hierarchy




C#

```
[Serializable]  
public abstract class BaseException : Exception;
```

File

Exceptions.cs ([see page 22](#))

Methods

	Name	Description
	BaseException (see page 5)	This is BaseException, a member of class BaseException.

1.1.2.1.1 BaseException Constructor

1.1.2.1.1.1 BaseException.BaseException Constructor ()

C#

```
protected BaseException();
```

Description

This is BaseException, a member of class BaseException.

Body Source

```
1: protected BaseException()  
2: {  
3: }
```

1.1.2.1.1.2 BaseException.BaseException Constructor (string)

C#

```
protected BaseException(string message);
```

Description

This is BaseException, a member of class BaseException.

Body Source

```
1: protected BaseException(string message) : base(message)  
2: {  
3: }
```

1.1.2.1.1.3 BaseException.BaseException Constructor (string, Exception)

C#

```
protected BaseException(string message, Exception innerException);
```

Description

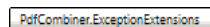
This is BaseException, a member of class BaseException.

Body Source

```
1: protected BaseException(string message, Exception innerException) : base(message,  
innerException)  
2: {  
3: }
```

1.1.2.2 ExceptionExtensions Class

Class Hierarchy

 PdfCombiner.ExceptionExtensions

C#

```
public static class ExceptionExtensions;
```



File

Exceptions.cs ( see page 22)

Description

This is class PdfCombiner.ExceptionExtensions.

ExceptionExtensions Methods

	Name	Description
	GetAllMessages ( see page 6)	This method is used to take all messages with stack trace info in exception

1.1.2.2.1 ExceptionExtensions Methods**1.1.2.2.1.1 ExceptionExtensions.GetAllMessages Method**

This method is used to take all messages with stack trace info in exception

C#

```
public static string GetAllMessages(this Exception exception);
```

Parameters

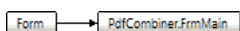
Parameters	Description
this Exception exception	Exception

Returns

Detailed Error Message

Body Source

```
1: public static string GetAllMessages(this Exception exception)
2: {
3:     var messages = exception.Message;
4:     if (exception.InnerException != null)
5:         messages += " " + exception.InnerException.GetAllMessages();
6:     if (!string.IsNullOrEmpty(exception.StackTrace))
7:         messages += " " + exception.StackTrace;
8:     if (!string.IsNullOrEmpty(exception.Source))
9:         messages += " " + exception.Source;
10:    return messages;
11: }
```

1.1.2.3 FrmMain Class**Class Hierarchy****C#**

```
public class FrmMain : Form;
```

File

FrmMain.Designer.cs ( see page 30)















Description

This is class PdfCombiner.FrmMain.














Methods




	Name	Description
	FrmMain ( see page 8)	This is FrmMain, a member of class FrmMain.

FrmMain Fields

	Name	Description
	AppTitle (see page 8)	This is AppTitle, a member of class FrmMain.
	btnAddFile (see page 8)	This is btnAddFile, a member of class FrmMain.
	btnAddFolder (see page 8)	This is btnAddFolder, a member of class FrmMain.
	btnClearList (see page 8)	This is btnClearList, a member of class FrmMain.
	btnCombineTextSharp (see page 9)	This is btnCombineTextSharp, a member of class FrmMain.
	btnCombinePdfSharp (see page 9)	This is btnCombinePdfSharp, a member of class FrmMain.
	components (see page 9)	Required designer variable.
	lbFiles (see page 9)	This is lbFiles, a member of class FrmMain.
	lblDetails (see page 9)	This is lblDetails, a member of class FrmMain.
	menuItemDelete (see page 9)	This is menuItemDelete, a member of class FrmMain.
	menuItemOrderAscending (see page 9)	This is menuItemOrderAscending, a member of class FrmMain.
	menuItemOrderDescending (see page 10)	This is menuItemOrderDescending, a member of class FrmMain.
	menuStrip (see page 10)	This is menuStrip, a member of class FrmMain.
	pbFiles (see page 10)	This is pbFiles, a member of class FrmMain.

FrmMain Methods

	Name	Description
	btnAddFile_Click (see page 10)	It is used to add single or multiple PDF files to combine When you choose file or files The listbox in the form will be filled with the name of the files Which you choose to combine
	btnAddFolder_Click (see page 11)	This function is used to add PDF files in a folder which you choose in Folder Dialog recursively When you use it the added files will be seen in listbox
	btnClearList_Click (see page 11)	This function is used to clear the file list in listbox
	btnCombineTextSharp_Click (see page 12)	This function is used to take list of PDF files in listbox, Combine them with a single file in location which you choose in folder dialog And you can see the progress in progress bar It uses iTextSharp Nuget Package
	btnCombinePdfSharp_Click (see page 13)	This function is used to take list of PDF files in listbox, Combine them with a single file in location which you choose in folder dialog And you can see the progress in progress bar It uses PdfSharp Nuget Package
	DeleteFilesFromListBox (see page 14)	This method is used to Delete selected files from ListBox This is used different ways in this app So we created this as a method
	Dispose (see page 15)	Clean up any resources being used.
	FrmMain_FormClosed (see page 15)	This function is used to terminate application
	FrmMain_Shown (see page 15)	This function is used to assign menu to listbox When form is shown
	InitializeComponent (see page 16)	Required method for Designer support - do not modify the contents of this method with the code editor.
	InitializeFileDialog (see page 18)	This function is used to set options of file dialog like Filter, start path etc.
	lbFiles_KeyDown (see page 19)	This function is used to delete files in listbox which are chosen If you press the Delete button in your keyboard The selected files will be deleted from the list
	menuItemDelete_Click (see page 19)	This is used to delete selected items in listbox When menu item Delete is clicked and opened dialog

	menultemOrderAscending_Click (🔗 see page 19)	This method is used to order elements in listbox ascending
	menultemOrderDescending_Click (🔗 see page 20)	This method is used to order elements in listbox ascending
	SortItems (🔗 see page 20)	This method is used to order items in listbox And return them with the given order as parametre

1.1.2.3.1 FrmMain.FrmMain Constructor

C#

```
public FrmMain();
```

Description

This is FrmMain, a member of class FrmMain.

Body Source

```
1: public FrmMain()
2: {
3:     InitializeComponent();
4: }
```

1.1.2.3.2 FrmMain Fields

1.1.2.3.2.1 FrmMain.AppTitle Field

C#

```
public const string AppTitle = "PDF Combiner";
```

Description

This is AppTitle, a member of class FrmMain.

1.1.2.3.2.2 FrmMain.btnAddFile Field

C#

```
private System.Windows.Forms.Button btnAddFile;
```

Description

This is btnAddFile, a member of class FrmMain.

1.1.2.3.2.3 FrmMain.btnAddFolder Field

C#

```
private System.Windows.Forms.Button btnAddFolder;
```

Description

This is btnAddFolder, a member of class FrmMain.

1.1.2.3.2.4 FrmMain.btnClearList Field

C#

```
private System.Windows.Forms.Button btnClearList;
```

Description

This is btnClearList, a member of class FrmMain.

1.1.2.3.2.5 FrmMain.btnCombineITextSharp Field

C#

```
private System.Windows.Forms.Button btnCombineITextSharp;
```

Description

This is btnCombineITextSharp, a member of class FrmMain.

1.1.2.3.2.6 FrmMain.btnCombinePdfSharp Field

C#

```
private System.Windows.Forms.Button btnCombinePdfSharp;
```

Description

This is btnCombinePdfSharp, a member of class FrmMain.

1.1.2.3.2.7 FrmMain.components Field

Required designer variable.

C#

```
private System.ComponentModel.IContainer components = null;
```

1.1.2.3.2.8 FrmMain.lbFiles Field

C#

```
private System.Windows.Forms.ListBox lbFiles;
```

Description

This is lbFiles, a member of class FrmMain.

1.1.2.3.2.9 FrmMain.lblDetails Field

C#

```
private System.Windows.Forms.Label lblDetails;
```

Description

This is lblDetails, a member of class FrmMain.

1.1.2.3.2.10 FrmMain.menuItemDelete Field

C#

```
private System.Windows.Forms.ToolStripMenuItem menuItemDelete;
```

Description

This is menuItemDelete, a member of class FrmMain.

1.1.2.3.2.11 FrmMain.menuItemOrderAscending Field

C#

```
private System.Windows.Forms.ToolStripMenuItem menuItemOrderAscending;
```

Description

This is menuItemOrderAscending, a member of class FrmMain.

1.1.2.3.2.12 FrmMain.menuItemOrderDescending Field

C#

```
private System.Windows.Forms.ToolStripItem menuItemOrderDescending;
```

Description

This is menuItemOrderDescending, a member of class FrmMain.

1.1.2.3.2.13 FrmMain.menuStrip Field

C#

```
private System.Windows.Forms.ContextMenuStrip menuStrip;
```

Description

This is menuStrip, a member of class FrmMain.

1.1.2.3.2.14 FrmMain.pbFiles Field

C#

```
private System.Windows.Forms.ProgressBar pbFiles;
```

Description

This is pbFiles, a member of class FrmMain.

1.1.2.3.3 FrmMain Methods

1.1.2.3.3.1 FrmMain.btnAddFile_Click Method

It is used to add single or multiple PDF files to combine When you choose file or files The listbox in the form will be filled with the name of the files Which you choose to combine

C#

```
private void btnAddFile_Click(object sender, EventArgs e);
```

Parameters

Parameters	Description
object sender	The sender info (For example Main Form)
EventArgs e	Event Arguments

Body Source

```
1: private void btnAddFile_Click(object sender, EventArgs e)
2: {
3:     using (var dialogAddFile = new OpenFileDialog())
4:     {
5:         InitializeFileDialog(dialogAddFile);
6:         var result = dialogAddFile.ShowDialog();
7:         var addedFileCount = 0;
8:         if (result == DialogResult.OK && dialogAddFile.FileNames != null)
9:         {
10:            foreach (var file in dialogAddFile.FileNames)
11:            {
12:                if (!lbFiles.Items.Contains(file))
13:                {
14:                    lbFiles.Items.Add(file);
15:                    addedFileCount++;
16:                }
17:            }
18:        }
19:    }
20: }
```

```
17:         }
18:         MessageBox.Show(addedFileCount + " file/s successfully added to the list",
AppTitle, MessageBoxButtons.OK, MessageBoxIcon.Information);
19:     }
20: }
21: }
```

1.1.2.3.3.2 FrmMain.btnAddFolder_Click Method

This function is used to add PDF files in a folder which you choose in Folder Dialog recursively When you use it the added files will be seen in listbox

C#

```
private void btnAddFolder_Click(object sender, EventArgs e);
```

Parameters

Parameters	Description
object sender	The sender info (For example Main Form)
EventArgs e	Event Arguments

Body Source

```
1: private void btnAddFolder_Click(object sender, EventArgs e)
2: {
3:     using (var dialogAddFolder = new FolderBrowserDialog())
4:     {
5:         var result = dialogAddFolder.ShowDialog();
6:         if (result == DialogResult.OK &&
!string.IsNullOrEmpty(dialogAddFolder.SelectedPath))
7:         {
8:             var fileNames = Directory.GetFiles(dialogAddFolder.SelectedPath, "*.pdf",
SearchOption.AllDirectories);
9:             var addedFileCount = 0;
10:            foreach (var file in fileNames)
11:            {
12:                if (!lbFiles.Items.Contains(file))
13:                {
14:                    lbFiles.Items.Add(file);
15:                    addedFileCount++;
16:                }
17:            }
18:            MessageBox.Show(addedFileCount + " file/s successfully added to the list",
AppTitle, MessageBoxButtons.OK, MessageBoxIcon.Information);
19:        }
20:    }
21: }
```

1.1.2.3.3.3 FrmMain.btnClearList_Click Method

This function is used to clear the file list in listbox

C#

```
private void btnClearList_Click(object sender, EventArgs e);
```

Parameters

Parameters	Description
object sender	The sender info (For example Main Form)
EventArgs e	Event Arguments

Body Source

```
1: private void btnClearList_Click(object sender, EventArgs e)
2: {
3:     var fileCount = lbFiles.Items.Count;
4:     lbFiles.Items.Clear();
```

```

5:     MessageBox.Show(fileCount + " file/s successfully deleted from the list", AppTitle,
6:     MessageBoxButtons.OK, MessageBoxIcon.Information);
6: }

```

1.1.2.3.4 FrmMain.btnCombineITextSharp_Click Method

This function is used to take list of PDF files in listbox, Combine them with a single file in location which you choose in folder dialog And you can see the progress in progress bar It uses iTextSharp Nuget Package

C#

```
private void btnCombineITextSharp_Click(object sender, EventArgs e);
```

Parameters

Parameters	Description
object sender	The sender info (For example Main Form)
EventArgs e	Event Arguments

Body Source

```

1: private void btnCombineITextSharp_Click(object sender, EventArgs e)
2: {
3:     try
4:     {
5:         if (lbFiles.Items.Count < 2)
6:         {
7:             MessageBox.Show("There must be at least 2 PDF files to combine", AppTitle,
8:             MessageBoxButtons.OK, MessageBoxIcon.Error);
9:         }
10:        else
11:        {
12:            using (var dialogExport = new FolderBrowserDialog())
13:            {
14:                var result = dialogExport.ShowDialog();
15:                if (result == DialogResult.OK &&
16:                !string.IsNullOrEmpty(dialogExport.SelectedPath))
17:                {
18:                    var outputFileName = dialogExport.SelectedPath + "/" +
19:                    Guid.NewGuid().ToString() + ".pdf";
20:                    var fileCount = lbFiles.Items.Count;
21:                    var combinedFiles = 0;
22:
23:                    var outputFile = new Document();
24:                    using (FileStream outputFileStream = new FileStream(outputFileName,
25:                    FileMode.Create))
26:                    {
27:                        var pdfWriter = new iTextSharp.text.pdf.PdfCopy(outputFile,
28:                        outputFileStream);
29:                        if (pdfWriter == null)
30:                        {
31:                            return;
32:                        }
33:                        pdfWriter.SetFullCompression();
34:                        outputFile.Open();
35:                        for (int i = 0; i < lbFiles.Items.Count; i++)
36:                        {
37:                            try
38:                            {
39:                                var pdfReader = new
40:                                iTextSharp.text.pdf.PdfReader(lbFiles.Items[i].ToString());
41:                                pdfReader.ConsolidateNamedDestinations();
42:                                for (int j = 1; j <= pdfReader.NumberOfPages; j++)
43:                                {
44:                                    var page = pdfWriter.GetImportedPage(pdfReader, j);
45:                                    pdfWriter.AddPage(page);
46:                                }
47:                                pdfReader.Close();

```

```

42:                combinedFiles++;
43:            }
44:            catch (Exception)
45:            {
46:                fileCount--;
47:            }
48:
49:            pbFiles.Value = (combinedFiles * 100 / fileCount);
50:            if (pbFiles.Value > pbFiles.Maximum)
51:                pbFiles.Value = pbFiles.Maximum;
52:        }
53:        pdfWriter.Close();
54:        outputFile.Close();
55:    }
56:
57:    MessageBox.Show(fileCount + " PDF files successfully combined in "
+ outputFileName, AppTitle, MessageBoxButtons.OK, MessageBoxIcon.Information);
58:
59:    pbFiles.Value = pbFiles.Minimum;
60:    }
61:    }
62:    }
63:    }
64:    catch (Exception ex)
65:    {
66:        MessageBox.Show("There is an error while combining PDF files in list. Details:
" + ex.GetAllMessages(), AppTitle, MessageBoxButtons.OK, MessageBoxIcon.Error);
67:        pbFiles.Value = pbFiles.Minimum;
68:    }
69: }

```

1.1.2.3.3.5 FrmMain.btnCombinePdfSharp_Click Method

This function is used to take list of PDF files in listbox, Combine them with a single file in location which you choose in folder dialog And you can see the progress in progress bar It uses PdfSharp Nuget Package

C#

```
private void btnCombinePdfSharp_Click(object sender, EventArgs e);
```

Parameters

Parameters	Description
object sender	The sender info (For example Main Form)
EventArgs e	Event Arguments

Body Source

```

1: private void btnCombinePdfSharp_Click(object sender, EventArgs e)
2: {
3:     try
4:     {
5:         if (lbFiles.Items.Count < 2)
6:         {
7:             MessageBox.Show("There must be at least 2 PDF files to combine", AppTitle,
+ MessageBoxButtons.OK, MessageBoxIcon.Error);
8:         }
9:         else
10:        {
11:            using (var dialogExport = new FolderBrowserDialog())
12:            {
13:                var result = dialogExport.ShowDialog();
14:                if (result == DialogResult.OK &&
+ !string.IsNullOrEmpty(dialogExport.SelectedPath))
15:                {
16:                    var outputFileName = dialogExport.SelectedPath + "/" +
+ Guid.NewGuid().ToString() + ".pdf";
17:                    var outputFile = new PdfDocument();

```

```

18:         outputFile.Options.CompressContentStreams = true;
19:         outputFile.Options.EnableCcittCompressionForBilevelImages = true;
20:         outputFile.Options.FlateEncodeMode =
PdfFlateEncodeMode.BestCompression;
21:         var fileCount = lbFiles.Items.Count;
22:         var combinedFiles = 0;
23:
24:         for (int i = 0; i < lbFiles.Items.Count; i++)
25:         {
26:             try
27:             {
28:                 var inputDocument =
PdfReader.Open(lbFiles.Items[i].ToString(), PdfDocumentOpenMode.Import);
29:                 var count = inputDocument.PageCount;
30:                 for (int idx = 0; idx < count; idx++)
31:                 {
32:                     var page = inputDocument.Pages[idx];
33:                     outputFile.AddPage(page);
34:                 }
35:                 inputDocument.Close();
36:                 combinedFiles++;
37:             }
38:             catch (Exception)
39:             {
40:                 fileCount--;
41:             }
42:
43:             pbFiles.Value = (combinedFiles * 100 / fileCount);
44:             if (pbFiles.Value > pbFiles.Maximum)
45:                 pbFiles.Value = pbFiles.Maximum;
46:         }
47:         outputFile.Save(outputFileName);
48:         outputFile.Close();
49:
50:         MessageBox.Show(fileCount + " PDF files successfully combined in "
+ outputFileName, AppTitle, MessageBoxButtons.OK, MessageBoxIcon.Information);
51:
52:         pbFiles.Value = pbFiles.Minimum;
53:     }
54: }
55:
56: }
57: catch (Exception ex)
58: {
59:     MessageBox.Show("There is an error while combining PDF files in list. Details:
" + ex.GetAllMessages(), AppTitle, MessageBoxButtons.OK, MessageBoxIcon.Error);
60:     pbFiles.Value = pbFiles.Minimum;
61: }
62: }

```

1.1.2.3.3.6 FrmMain.DeleteFilesFromListBox Method

This method is used to Delete selected files from ListBox This is used different ways in this app So we created this as a method

C#

```
private void DeleteFilesFromListBox();
```

Body Source

```

1: private void DeleteFilesFromListBox()
2: {
3:     var allItems = new List<string>();
4:     var removedItems = new List<string>();
5:     var fileCount = lbFiles.SelectedItems.Count;
6:     for (int i = 0; i < lbFiles.SelectedItems.Count; i++)
7:     {
8:         removedItems.Add(lbFiles.SelectedItems[i].ToString());

```



```

9:     }
10:    foreach (var item in lbFiles.Items)
11:    {
12:        if (!removedItems.Any(j => j == item.ToString()))
13:            allItems.Add(item.ToString());
14:    }
15:    lbFiles.Items.Clear();
16:    foreach (var item in allItems)
17:    {
18:        lbFiles.Items.Add(item);
19:    }
20:    MessageBox.Show(fileCount + " file/s are deleted from the list.", AppTitle,
    MessageBoxButtons.OK, MessageBoxIcon.Information);
21: }

```

1.1.2.3.7 FrmMain.Dispose Method

Clean up any resources being used.

C#

```
protected override void Dispose(bool disposing);
```

Parameters

Parameters	Description
bool disposing	true if managed resources should be disposed; otherwise, false.

Body Source

```

1: protected override void Dispose(bool disposing)
2: {
3:     if (disposing && (components != null))
4:     {
5:         components.Dispose();
6:     }
7:     base.Dispose(disposing);
8: }

```

1.1.2.3.8 FrmMain.FrmMain_FormClosed Method

This function is used to terminate application

C#

```
private void FrmMain_FormClosed(object sender, FormClosedEventArgs e);
```

Parameters

Parameters	Description
object sender	The sender info (For example Main Form)
FormClosedEventArgs e	Event Arguments

Body Source

```

1: private void FrmMain_FormClosed(object sender, FormClosedEventArgs e)
2: {
3:     MessageBox.Show("Thanks for using this app", AppTitle, MessageBoxButtons.OK,
    MessageBoxIcon.Information);
4:     Application.Exit();
5: }

```

1.1.2.3.9 FrmMain.FrmMain_Shown Method

This function is used to assign menu to listbox When form is shown

C#

```
private void FrmMain_Shown(object sender, EventArgs e);
```

Parameters

Parameters	Description
object sender	The sender info (For example Main Form)
EventArgs e	Event Arguments

Body Source

```

1: private void FrmMain_Shown(object sender, EventArgs e)
2: {
3:     lblFiles.ContextMenuStrip = menuStrip;
4: }

```

1.1.2.3.3.10 FrmMain.InitializeComponent Method

Required method for Designer support - do not modify the contents of this method with the code editor.

C#

```
private void InitializeComponent();
```

Body Source

```

1: private void InitializeComponent()
2: {
3:     this.components = new System.ComponentModel.Container();
4:     System.ComponentModel.ComponentResourceManager resources = new
System.ComponentModel.ComponentResourceManager(typeof(FrmMain));
5:     this.btnCombinePdfSharp = new System.Windows.Forms.Button();
6:     this.btnAddFolder = new System.Windows.Forms.Button();
7:     this.btnAddFile = new System.Windows.Forms.Button();
8:     this.btnClearList = new System.Windows.Forms.Button();
9:     this.lblFiles = new System.Windows.Forms.ListBox();
10:    this.lblDetails = new System.Windows.Forms.Label();
11:    this.pbFiles = new System.Windows.Forms.ProgressBar();
12:    this.btnCombineITextSharp = new System.Windows.Forms.Button();
13:    this.menuStrip = new System.Windows.Forms.ContextMenuStrip(this.components);
14:    this.menuItemDelete = new System.Windows.Forms.ToolStripMenuItem();
15:    this.menuItemOrderAscending = new System.Windows.Forms.ToolStripMenuItem();
16:    this.menuItemOrderDescending = new System.Windows.Forms.ToolStripMenuItem();
17:    this.menuStrip.SuspendLayout();
18:    this.SuspendLayout();
19:    //
20:    // btnCombinePdfSharp
21:    //
22:    this.btnCombinePdfSharp.Location = new System.Drawing.Point(487, 12);
23:    this.btnCombinePdfSharp.Name = "btnCombinePdfSharp";
24:    this.btnCombinePdfSharp.Size = new System.Drawing.Size(169, 23);
25:    this.btnCombinePdfSharp.TabIndex = 0;
26:    this.btnCombinePdfSharp.Text = "Combine Files (PdfSharp)";
27:    this.btnCombinePdfSharp.UseVisualStyleBackColor = true;
28:    this.btnCombinePdfSharp.Click += new
System.EventHandler(this.btnCombinePdfSharp_Click);
29:    //
30:    // btnAddFolder
31:    //
32:    this.btnAddFolder.Location = new System.Drawing.Point(12, 38);
33:    this.btnAddFolder.Name = "btnAddFolder";
34:    this.btnAddFolder.Size = new System.Drawing.Size(156, 23);
35:    this.btnAddFolder.TabIndex = 2;
36:    this.btnAddFolder.Text = "Add Folder";
37:    this.btnAddFolder.UseVisualStyleBackColor = true;
38:    this.btnAddFolder.Click += new System.EventHandler(this.btnAddFolder_Click);
39:    //
40:    // btnAddFile
41:    //
42:    this.btnAddFile.Location = new System.Drawing.Point(12, 12);
43:    this.btnAddFile.Name = "btnAddFile";

```

```
44:         this.btnAddFile.Size = new System.Drawing.Size(156, 23);
45:         this.btnAddFile.TabIndex = 3;
46:         this.btnAddFile.Text = "Add Files";
47:         this.btnAddFile.UseVisualStyleBackColor = true;
48:         this.btnAddFile.Click += new System.EventHandler(this.btnAddFile_Click);
49:         //
50:         // btnClearList
51:         //
52:         this.btnClearList.Location = new System.Drawing.Point(174, 12);
53:         this.btnClearList.Name = "btnClearList";
54:         this.btnClearList.Size = new System.Drawing.Size(90, 49);
55:         this.btnClearList.TabIndex = 4;
56:         this.btnClearList.Text = "Clear File List";
57:         this.btnClearList.UseVisualStyleBackColor = true;
58:         this.btnClearList.Click += new System.EventHandler(this.btnClearList_Click);
59:         //
60:         // lbFiles
61:         //
62:         this.lbFiles.FormattingEnabled = true;
63:         this.lbFiles.Location = new System.Drawing.Point(12, 88);
64:         this.lbFiles.Name = "lbFiles";
65:         this.lbFiles.SelectionMode = System.Windows.Forms.SelectionMode.MultiExtended;
66:         this.lbFiles.Size = new System.Drawing.Size(644, 186);
67:         this.lbFiles.TabIndex = 5;
68:         this.lbFiles.KeyDown += new
System.Windows.Forms.KeyEventHandler(this.lbFiles_KeyDown);
69:         //
70:         // lblDetails
71:         //
72:         this.lblDetails.AutoSize = true;
73:         this.lblDetails.Location = new System.Drawing.Point(12, 65);
74:         this.lblDetails.Name = "lblDetails";
75:         this.lblDetails.Size = new System.Drawing.Size(0, 13);
76:         this.lblDetails.TabIndex = 6;
77:         //
78:         // pbFiles
79:         //
80:         this.pbFiles.Location = new System.Drawing.Point(270, 12);
81:         this.pbFiles.Name = "pbFiles";
82:         this.pbFiles.Size = new System.Drawing.Size(211, 49);
83:         this.pbFiles.TabIndex = 7;
84:         //
85:         // btnCombineITextSharp
86:         //
87:         this.btnCombineITextSharp.Location = new System.Drawing.Point(487, 38);
88:         this.btnCombineITextSharp.Name = "btnCombineITextSharp";
89:         this.btnCombineITextSharp.Size = new System.Drawing.Size(169, 23);
90:         this.btnCombineITextSharp.TabIndex = 9;
91:         this.btnCombineITextSharp.Text = "Combine Files (iTextSharp)";
92:         this.btnCombineITextSharp.UseVisualStyleBackColor = true;
93:         this.btnCombineITextSharp.Click += new
System.EventHandler(this.btnCombineITextSharp_Click);
94:         //
95:         // menuStrip
96:         //
97:         this.menuStrip.Items.AddRange(new System.Windows.Forms.ToolStripItem[] {
98:             this.menuItemDelete,
99:             this.menuItemOrderAscending,
100:             this.menuItemOrderDescending});
101:         this.menuStrip.Name = "menuStrip";
102:         this.menuStrip.Size = new System.Drawing.Size(178, 70);
103:         //
104:         // menuItemDelete
105:         //
106:         this.menuItemDelete.Image =
((System.Drawing.Image)(resources.GetObject("menuItemDelete.Image")));
107:         this.menuItemDelete.Name = "menuItemDelete";
108:         this.menuItemDelete.Size = new System.Drawing.Size(177, 22);
109:         this.menuItemDelete.Text = "Delete";
```

```

110:         this.menuItemDelete.Click += new System.EventHandler(this.menuItemDelete_Click);
111:         //
112:         // menuItemOrderAscending
113:         //
114:         this.menuItemOrderAscending.Image =
((System.Drawing.Image)(resources.GetObject("menuItemOrderAscending.Image")));
115:         this.menuItemOrderAscending.Name = "menuItemOrderAscending";
116:         this.menuItemOrderAscending.Size = new System.Drawing.Size(177, 22);
117:         this.menuItemOrderAscending.Text = "Order (Ascending)";
118:         this.menuItemOrderAscending.Click += new
System.EventHandler(this.menuItemOrderAscending_Click);
119:         //
120:         // menuItemOrderDescending
121:         //
122:         this.menuItemOrderDescending.Image =
((System.Drawing.Image)(resources.GetObject("menuItemOrderDescending.Image")));
123:         this.menuItemOrderDescending.Name = "menuItemOrderDescending";
124:         this.menuItemOrderDescending.Size = new System.Drawing.Size(177, 22);
125:         this.menuItemOrderDescending.Text = "Order (Descending)";
126:         this.menuItemOrderDescending.Click += new
System.EventHandler(this.menuItemOrderDescending_Click);
127:         //
128:         // FrmMain
129:         //
130:         this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
131:         this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
132:         this.ClientSize = new System.Drawing.Size(668, 287);
133:         this.Controls.Add(this.btnCombineITextSharp);
134:         this.Controls.Add(this.pbFiles);
135:         this.Controls.Add(this.lblDetails);
136:         this.Controls.Add(this.lbFiles);
137:         this.Controls.Add(this.btnClearList);
138:         this.Controls.Add(this.btnAddFile);
139:         this.Controls.Add(this.btnAddFolder);
140:         this.Controls.Add(this.btnCombinePdfSharp);
141:         this.Name = "FrmMain";
142:         this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
143:         this.Text = "PDF Combiner";
144:         this.FormClosed += new
System.Windows.Forms.FormClosedEventHandler(this.FrmMain_FormClosed);
145:         this.Shown += new System.EventHandler(this.FrmMain_Shown);
146:         this.menuStrip.ResumeLayout(false);
147:         this.ResumeLayout(false);
148:         this.PerformLayout();
149:
150: }

```

1.1.2.3.3.11 FrmMain.InitializeFileDialog Method

This function is used to set options of file dialog like Filter, start path etc.

C#

```
private static void InitializeFileDialog(OpenFileDialog dialogAddFile);
```

Parameters

Parameters	Description
OpenFileDialog dialogAddFile	File Dialog

Body Source

```

1: private static void InitializeFileDialog(OpenFileDialog dialogAddFile)
2: {
3:     dialogAddFile.Multiselect = true;
4:     dialogAddFile.Filter = "pdf files (*.pdf)|*.pdf";
5:     dialogAddFile.InitialDirectory = Application.StartupPath;
6:     dialogAddFile.Title = "Select PDF File/s";
7:     dialogAddFile.DefaultExt = "pdf";

```

```
8: }
```

1.1.2.3.3.12 FrmMain.lbFiles_KeyDown Method

This function is used to delete files in listbox which are chosen If you press the Delete button in your keyboard The selected files will be deleted from the list

C#

```
private void lbFiles_KeyDown(object sender, KeyEventArgs e);
```

Parameters

Parameters	Description
object sender	The sender info (For example Main Form)
KeyEventArgs e	Event Arguments

Body Source

```
1: private void lbFiles_KeyDown(object sender, KeyEventArgs e)
2: {
3:     if (e.KeyValue == Keys.Delete.GetHashCode())
4:         DeleteFilesFromListBox();
5: }
```

1.1.2.3.3.13 FrmMain.menuItemDelete_Click Method

This is used to delete selected items in listbox When menu item Delete is clicked and opened dialog

C#

```
private void menuItemDelete_Click(object sender, EventArgs e);
```

Parameters

Parameters	Description
object sender	The sender info (For example Main Form)
EventArgs e	Event Arguments

Body Source

```
1: private void menuItemDelete_Click(object sender, EventArgs e)
2: {
3:     if (lbFiles.Items.Count > 0)
4:     {
5:         if (lbFiles.SelectedItems.Count > 0)
6:         {
7:             var deleteDialog = new DialogResult();
8:             deleteDialog = MessageBox.Show("You are about to delete " +
lbFiles.SelectedItems.Count + " files in listbox. Are you sure?", AppTitle,
MessageBoxButtons.YesNo, MessageBoxIcon.Warning);
9:             if (deleteDialog == DialogResult.Yes)
10:                 DeleteFilesFromListBox();
11:         }
12:         else
13:             MessageBox.Show("There is no selected files in list.", AppTitle,
MessageBoxButtons.OK, MessageBoxIcon.Error);
14:     }
15:     else
16:         MessageBox.Show("There is no files in list.", AppTitle, MessageBoxButtons.OK,
MessageBoxIcon.Error);
17: }
```

1.1.2.3.3.14 FrmMain.menuItemOrderAscending_Click Method

This method is used to order elements in listbox ascending

C#

```
private void menuItemOrderAscending_Click(object sender, EventArgs e);
```

Parameters

Parameters	Description
object sender	The sender info (For example Main Form)
EventArgs e	Event Arguments

Body Source

```
1: private void menuItemOrderAscending_Click(object sender, EventArgs e)
2: {
3:     if (lbFiles.Items.Count > 0)
4:         lbFiles = SortItems(lbFiles, true);
5:     else
6:         MessageBox.Show("There is no files in list.", AppTitle, MessageBoxButtons.OK,
7:             MessageBoxIcon.Error);
8: }
```

1.1.2.3.3.15 FrmMain.menuItemOrderDescending_Click Method

This method is used to order elements in listbox ascending

C#

```
private void menuItemOrderDescending_Click(object sender, EventArgs e);
```

Parameters

Parameters	Description
object sender	The sender info (For example Main Form)
EventArgs e	Event Arguments

Body Source

```
1: private void menuItemOrderDescending_Click(object sender, EventArgs e)
2: {
3:     if (lbFiles.Items.Count > 0)
4:         lbFiles = SortItems(lbFiles, false);
5:     else
6:         MessageBox.Show("There is no files in list.", AppTitle, MessageBoxButtons.OK,
7:             MessageBoxIcon.Error);
8: }
```

1.1.2.3.3.16 FrmMain.SortItems Method

This method is used to order items in listbox And return them with the given order as parametre

C#

```
private static ListBox SortItems(ListBox lb, bool ascending);
```

Parameters

Parameters	Description
Listbox lb	Listbox info
bool ascending	Order Type is Ascending Or Not

Body Source

```
1: private static ListBox SortItems(ListBox lb, bool ascending)
2: {
3:     List<object> items;
4:     items = lb.Items.OfType<object>().ToList();
5:     lb.Items.Clear();
6:     if (ascending)
7:         lb.Items.AddRange(items.OrderBy(i => i).ToArray());
8: }
```

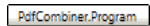
```

8:         else
9:             lb.Items.AddRange(items.OrderByDescending(i => i).ToArray());
10:        return lb;
11:    }

```

1.1.2.4 Program Class

Class Hierarchy



C#

```
public static class Program;
```

File

Program.cs ( see page 33)

Description

This is class PdfCombiner.Program.

Program Methods

	Name	Description
	Main ( see page 21)	The main entry point for the application.

1.1.2.4.1 Program Methods

1.1.2.4.1.1 Program.Main Method

The main entry point for the application.

C#

```

[STAThread]
private static void Main();

```

Body Source

```





1: [STAThread]
2: static void Main()
3: {
4:     Application.EnableVisualStyles();
5:     Application.SetCompatibleTextRenderingDefault(false);
6:     Application.Run(new FrmMain());
7: }

```

1.2 Files

The following table lists files in this documentation.

Files

Name	Description
AssemblyInfo.cs ( see page 22)	This is file AssemblyInfo.cs.
Exceptions.cs ( see page 22)	This is file Exceptions.cs.
FrmMain.cs ( see page 23)	This is file FrmMain.cs.
FrmMain.Designer.cs ( see page 30)	This is file FrmMain.Designer.cs.

PdfCombiner.csproj (see page 33)	This is file PdfCombiner.csproj.
PdfCombiner.sln (see page 33)	This is file PdfCombiner.sln.
Program.cs (see page 33)	This is file Program.cs.
Resources.Designer.cs (see page 34)	This code was generated by a tool. Runtime Version:4.0.30319.42000 Changes to this file may cause incorrect behavior and will be lost if the code is regenerated.
Settings.Designer.cs (see page 35)	This code was generated by a tool. Runtime Version:4.0.30319.42000 Changes to this file may cause incorrect behavior and will be lost if the code is regenerated.

1.2.1 AssemblyInfo.cs

This is file AssemblyInfo.cs.

Body Source

```

1: ?using System.Reflection;
2: using System.Runtime.CompilerServices;
3: using System.Runtime.InteropServices;
4:
5: // General Information about an assembly is controlled through the following
6: // set of attributes. Change these attribute values to modify the information
7: // associated with an assembly.
8: [assembly: AssemblyTitle("PdfCombiner")]
9: [assembly: AssemblyDescription("")]
10: [assembly: AssemblyConfiguration("")]
11: [assembly: AssemblyCompany("")]
12: [assembly: AssemblyProduct("PdfCombiner")]
13: [assembly: AssemblyCopyright("Copyright © 2022")]
14: [assembly: AssemblyTrademark("")]
15: [assembly: AssemblyCulture("")]
16:
17: // Setting ComVisible to false makes the types in this assembly not visible
18: // to COM components. If you need to access a type in this assembly from
19: // COM, set the ComVisible attribute to true on that type.
20: [assembly: ComVisible(false)]
21:
22: // The following GUID is for the ID of the typelib if this project is exposed to COM
23: [assembly: Guid("48e34ee8-dfaa-48db-acfd-70573cee9c00")]
24:
25: // Version information for an assembly consists of the following four values:
26: //
27: //      Major Version
28: //      Minor Version
29: //      Build Number
30: //      Revision
31: //
32: // You can specify all the values or you can default the Build and Revision Numbers
33: // by using the '*' as shown below:
34: // [assembly: AssemblyVersion("1.0.*")]
35: [assembly: AssemblyVersion("1.0.0.0")]
36: [assembly: AssemblyFileVersion("1.0.0.0")]

```

1.2.2 Exceptions.cs

This is file Exceptions.cs.

Body Source

```
1: ?using System;
2:
3: namespace PdfCombiner
4: {
5:     /// <summary>
6:     /// This function is used to take exceptions which can be thrown in
7:     /// PDF Combiner App
8:     /// </summary>
9:     [Serializable]
10:    public abstract class BaseException : Exception
11:    {
12:        protected BaseException()
13:        {
14:        }
15:
16:        protected BaseException(string message) : base(message)
17:        {
18:        }
19:
20:        protected BaseException(string message, Exception innerException) :
21:        base(message, innerException)
22:        {
23:        }
24:
25:        public static class ExceptionExtensions
26:        {
27:            /// <summary>
28:            /// This method is used to take all messages with stack trace info in exception
29:            /// </summary>
30:            /// <param name="exception">Exception</param>
31:            /// <returns>Detailed Error Message</returns>
32:            public static string GetAllMessages(this Exception exception)
33:            {
34:                var messages = exception.Message;
35:                if (exception.InnerException != null)
36:                    messages += " " + exception.InnerException.GetAllMessages();
37:                if (!string.IsNullOrEmpty(exception.StackTrace))
38:                    messages += " " + exception.StackTrace;
39:                if (!string.IsNullOrEmpty(exception.Source))
40:                    messages += " " + exception.Source;
41:                return messages;
42:            }
43:        }
44:    }
```

Namespaces

Name	Description
PdfCombiner (see page 1)	This is namespace PdfCombiner.

1.2.3 FrmMain.cs

This is file FrmMain.cs.

Body Source

```
1: ?using iTextSharp.text;
2: using PdfSharp.Pdf;
3: using PdfSharp.Pdf.IO;
4: using System;
5: using System.Collections.Generic;
6: using System.IO;
```

```

7: using System.Linq;
8: using System.Windows.Forms;
9:
10: namespace PdfCombiner
11: {
12:     public partial class FrmMain : Form
13:     {
14:         public const string AppTitle = "PDF Combiner";
15:
16:         public FrmMain()
17:         {
18:             InitializeComponent();
19:         }
20:
21:         /// <summary>
22:         /// This function is used to terminate application
23:         /// </summary>
24:         /// <param name="sender">The sender info (For example Main Form)</param>
25:         /// <param name="e">Event Arguments</param>
26:         private void FrmMain_FormClosed(object sender, FormClosedEventArgs e)
27:         {
28:             MessageBox.Show("Thanks for using this app", AppTitle,
29:             MessageBoxButtons.OK, MessageBoxIcon.Information);
30:             Application.Exit();
31:         }
32:
33:         /// <summary>
34:         /// This function is used to assign menu to listbox
35:         /// When form is shown
36:         /// </summary>
37:         /// <param name="sender">The sender info (For example Main Form)</param>
38:         /// <param name="e">Event Arguments</param>
39:         private void FrmMain_Shown(object sender, EventArgs e)
40:         {
41:             lbFiles.ContextMenuStrip = menuStrip;
42:         }
43:
44:         /// <summary>
45:         /// It is used to add single or multiple PDF files to combine
46:         /// When you choose file or files
47:         /// The listbox in the form will be filled with the name of the files
48:         /// Which you choose to combine
49:         /// </summary>
50:         /// <param name="sender">The sender info (For example Main Form)</param>
51:         /// <param name="e">Event Arguments</param>
52:         private void btnAddFile_Click(object sender, EventArgs e)
53:         {
54:             using (var dialogAddFile = new OpenFileDialog())
55:             {
56:                 InitializeFileDialog(dialogAddFile);
57:                 var result = dialogAddFile.ShowDialog();
58:                 var addedFileCount = 0;
59:                 if (result == DialogResult.OK && dialogAddFile.FileNames != null)
60:                 {
61:                     foreach (var file in dialogAddFile.FileNames)
62:                     {
63:                         if (!lbFiles.Items.Contains(file))
64:                         {
65:                             lbFiles.Items.Add(file);
66:                             addedFileCount++;
67:                         }
68:                     }
69:                     MessageBox.Show(addedFileCount + " file/s successfully added to
70: the list", AppTitle, MessageBoxButtons.OK, MessageBoxIcon.Information);
71:                 }
72:             }
73:         }
74:
75:         /// <summary>

```

```

74:         /// This function is used to set options of file dialog like
75:         /// Filter, start path etc.
76:         /// </summary>
77:         /// <param name="dialogAddFile">File Dialog</param>
78:         private static void InitializeFileDialog(OpenFileDialog dialogAddFile)
79:         {
80:             dialogAddFile.Multiselect = true;
81:             dialogAddFile.Filter = "pdf files (*.pdf)|*.pdf";
82:             dialogAddFile.InitialDirectory = Application.StartupPath;
83:             dialogAddFile.Title = "Select PDF File/s";
84:             dialogAddFile.DefaultExt = "pdf";
85:         }
86:
87:         /// <summary>
88:         /// This function is used to clear the file list in listbox
89:         /// </summary>
90:         /// <param name="sender">The sender info (For example Main Form)</param>
91:         /// <param name="e">Event Arguments</param>
92:         private void btnClearList_Click(object sender, EventArgs e)
93:         {
94:             var fileCount = lbFiles.Items.Count;
95:             lbFiles.Items.Clear();
96:             MessageBox.Show(fileCount + " file/s successfully deleted from the list",
AppTitle, MessageBoxButtons.OK, MessageBoxIcon.Information);
97:         }
98:
99:         /// <summary>
100:        /// This function is used to add PDF files in a folder which you choose in
101:        /// Folder Dialog recursively
102:        /// When you use it the added files will be seen in listbox
103:        /// </summary>
104:        /// <param name="sender">The sender info (For example Main Form)</param>
105:        /// <param name="e">Event Arguments</param>
106:        private void btnAddFolder_Click(object sender, EventArgs e)
107:        {
108:            using (var dialogAddFolder = new FolderBrowserDialog())
109:            {
110:                var result = dialogAddFolder.ShowDialog();
111:                if (result == DialogResult.OK &&
!string.IsNullOrEmpty(dialogAddFolder.SelectedPath))
112:                {
113:                    var fileNames = Directory.GetFiles(dialogAddFolder.SelectedPath,
"*.pdf", SearchOption.AllDirectories);
114:                    var addedFileCount = 0;
115:                    foreach (var file in fileNames)
116:                    {
117:                        if (!lbFiles.Items.Contains(file))
118:                        {
119:                            lbFiles.Items.Add(file);
120:                            addedFileCount++;
121:                        }
122:                    }
123:                    MessageBox.Show(addedFileCount + " file/s successfully added to
the list", AppTitle, MessageBoxButtons.OK, MessageBoxIcon.Information);
124:                }
125:            }
126:        }
127:
128:        /// <summary>
129:        /// This function is used to delete files in listbox which are chosen
130:        /// If you press the Delete button in your keyboard
131:        /// The selected files will be deleted from the list
132:        /// </summary>
133:        /// <param name="sender">The sender info (For example Main Form)</param>
134:        /// <param name="e">Event Arguments</param>
135:        private void lbFiles_KeyDown(object sender, KeyEventArgs e)
136:        {
137:            if (e.KeyValue == Keys.Delete.GetHashCode())
138:                DeleteFilesFromListBox();

```

```

139:         }
140:
141:         /// <summary>
142:         /// This function is used to take list of PDF files in listbox,
143:         /// Combine them with a single file in location which you choose in folder
144:         /// And you can see the progress in progress bar
145:         /// It uses PdfSharp Nuget Package
146:         /// </summary>
147:         /// <param name="sender">The sender info (For example Main Form)</param>
148:         /// <param name="e">Event Arguments</param>
149:         private void btnCombinePdfSharp_Click(object sender, EventArgs e)
150:         {
151:             try
152:             {
153:                 if (lbFiles.Items.Count < 2)
154:                 {
155:                     MessageBox.Show("There must be at least 2 PDF files to combine",
156: AppTitle, MessageBoxButtons.OK, MessageBoxIcon.Error);
157:                 }
158:                 else
159:                 {
160:                     using (var dialogExport = new FolderBrowserDialog())
161:                     {
162:                         var result = dialogExport.ShowDialog();
163:                         if (result == DialogResult.OK &&
164: !string.IsNullOrEmpty(dialogExport.SelectedPath))
165:                         {
166:                             var outputFileName = dialogExport.SelectedPath + "/" +
167: Guid.NewGuid().ToString() + ".pdf";
168:                             var outputFile = new PdfDocument();
169:                             outputFile.Options.CompressContentStreams = true;
170:                             outputFile.Options.EnableCcittCompressionForBilevelImages
171: = true;
172:                             outputFile.Options.FlateEncodeMode =
173: PdfFlateEncodeMode.BestCompression;
174:                             var fileCount = lbFiles.Items.Count;
175:                             var combinedFiles = 0;
176:
177:                             for (int i = 0; i < lbFiles.Items.Count; i++)
178:                             {
179:                                 try
180:                                 {
181:                                     var inputDocument =
182: PdfReader.Open(lbFiles.Items[i].ToString(), PdfDocumentOpenMode.Import);
183:                                     var count = inputDocument.PageCount;
184:                                     for (int idx = 0; idx < count; idx++)
185:                                     {
186:                                         var page = inputDocument.Pages[idx];
187:                                         outputFile.AddPage(page);
188:                                     }
189:                                     inputDocument.Close();
190:                                     combinedFiles++;
191:                                 }
192:                                 catch (Exception)
193:                                 {
194:                                     fileCount--;
195:                                 }
196:
197:                                 pbFiles.Value = (combinedFiles * 100 / fileCount);
198:                                 if (pbFiles.Value > pbFiles.Maximum)
199:                                     pbFiles.Value = pbFiles.Maximum;
200:                             }
201:                             outputFile.Save(outputFileName);
202:                             outputFile.Close();
203:
204:                             MessageBox.Show(fileCount + " PDF files successfully
205: combined in " + outputFileName, AppTitle, MessageBoxButtons.OK, MessageBoxIcon.Information);
206:

```

```

200:                 pbFiles.Value = pbFiles.Minimum;
201:             }
202:         }
203:     }
204: }
205:     catch (Exception ex)
206:     {
207:         MessageBox.Show("There is an error while combining PDF files in list.
Details: " + ex.GetAllMessages(), AppTitle, MessageBoxButtons.OK, MessageBoxIcon.Error);
208:         pbFiles.Value = pbFiles.Minimum;
209:     }
210: }
211:
212:     /// <summary>
213:     /// This function is used to take list of PDF files in listbox,
214:     /// Combine them with a single file in location which you choose in folder
dialog
215:     /// And you can see the progress in progress bar
216:     /// It uses iTextSharp Nuget Package
217:     /// </summary>
218:     /// <param name="sender">The sender info (For example Main Form)</param>
219:     /// <param name="e">Event Arguments</param>
220:     private void btnCombineITextSharp_Click(object sender, EventArgs e)
221:     {
222:         try
223:         {
224:             if (lbFiles.Items.Count < 2)
225:             {
226:                 MessageBox.Show("There must be at least 2 PDF files to combine",
AppTitle, MessageBoxButtons.OK, MessageBoxIcon.Error);
227:             }
228:             else
229:             {
230:                 using (var dialogExport = new FolderBrowserDialog())
231:                 {
232:                     var result = dialogExport.ShowDialog();
233:                     if (result == DialogResult.OK &&
!string.IsNullOrEmpty(dialogExport.SelectedPath))
234:                     {
235:                         var outputFileName = dialogExport.SelectedPath + "/" +
Guid.NewGuid().ToString() + ".pdf";
236:                         var fileCount = lbFiles.Items.Count;
237:                         var combinedFiles = 0;
238:
239:                         var outputFile = new Document();
240:                         using (FileStream outputStream = new
FileStream(outputFileName, FileMode.Create))
241:                         {
242:                             var pdfWriter = new
iTextSharp.text.pdf.PdfCopy(outputFile, outputStream);
243:                             if (pdfWriter == null)
244:                             {
245:                                 return;
246:                             }
247:                             pdfWriter.SetFullCompression();
248:                             outputFile.Open();
249:                             for (int i = 0; i < lbFiles.Items.Count; i++)
250:                             {
251:                                 try
252:                                 {
253:                                     var pdfReader = new
iTextSharp.text.pdf.PdfReader(lbFiles.Items[i].ToString());
254:                                     pdfReader.ConsolidateNamedDestinations();
255:                                     for (int j = 1; j <= pdfReader.NumberOfPages;
j++)
256:                                     {
257:                                         var page =
pdfWriter.GetImportedPage(pdfReader, j);
258:                                         pdfWriter.AddPage(page);

```

```

259:         }
260:         pdfReader.Close();
261:         combinedFiles++;
262:     }
263:     catch (Exception)
264:     {
265:         fileCount--;
266:     }
267:
268:     pbFiles.Value = (combinedFiles * 100 / fileCount);
269:     if (pbFiles.Value > pbFiles.Maximum)
270:         pbFiles.Value = pbFiles.Maximum;
271:     }
272:     pdfWriter.Close();
273:     outputFile.Close();
274: }
275:
276:     MessageBox.Show(fileCount + " PDF files successfully
combined in " + outputFileName, AppTitle, MessageBoxButtons.OK, MessageBoxIcon.Information);
277:
278:     pbFiles.Value = pbFiles.Minimum;
279: }
280: }
281: }
282: }
283:     catch (Exception ex)
284:     {
285:         MessageBox.Show("There is an error while combining PDF files in list.
Details: " + ex.GetAllMessages(), AppTitle, MessageBoxButtons.OK, MessageBoxIcon.Error);
286:         pbFiles.Value = pbFiles.Minimum;
287:     }
288: }
289:
290:     /// <summary>
291:     /// This is used to delete selected items in listbox
292:     /// When menu item Delete is clicked and opened dialog
293:     /// </summary>
294:     /// <param name="sender">The sender info (For example Main Form)</param>
295:     /// <param name="e">Event Arguments</param>
296:     private void menuItemDelete_Click(object sender, EventArgs e)
297:     {
298:         if (lbFiles.Items.Count > 0)
299:         {
300:             if (lbFiles.SelectedItems.Count > 0)
301:             {
302:                 var deleteDialog = new DialogResult();
303:                 deleteDialog = MessageBox.Show("You are about to delete " +
lbFiles.SelectedItems.Count + " files in listbox. Are you sure?", AppTitle,
MessageBoxButtons.YesNo, MessageBoxIcon.Warning);
304:                 if (deleteDialog == DialogResult.Yes)
305:                     DeleteFilesFromListBox();
306:             }
307:             else
308:                 MessageBox.Show("There is no selected files in list.", AppTitle,
MessageBoxButtons.OK, MessageBoxIcon.Error);
309:         }
310:         else
311:             MessageBox.Show("There is no files in list.", AppTitle,
MessageBoxButtons.OK, MessageBoxIcon.Error);
312:     }
313:
314:     /// <summary>
315:     /// This method is used to order elements in listbox ascending
316:     /// </summary>
317:     /// <param name="sender">The sender info (For example Main Form)</param>
318:     /// <param name="e">Event Arguments</param>
319:     private void menuItemOrderAscending_Click(object sender, EventArgs e)
320:     {
321:         if (lbFiles.Items.Count > 0)

```

```

322:         lbFiles = SortItems(lbFiles, true);
323:     else
324:         MessageBox.Show("There is no files in list.", AppTitle,
325:             MessageBoxButtons.OK, MessageBoxIcon.Error);
326:     }
327:     /// <summary>
328:     /// This method is used to order elements in listbox ascending
329:     /// </summary>
330:     /// <param name="sender">The sender info (For example Main Form)</param>
331:     /// <param name="e">Event Arguments</param>
332:     private void menuItemOrderDescending_Click(object sender, EventArgs e)
333:     {
334:         if (lbFiles.Items.Count > 0)
335:             lbFiles = SortItems(lbFiles, false);
336:         else
337:             MessageBox.Show("There is no files in list.", AppTitle,
338:                 MessageBoxButtons.OK, MessageBoxIcon.Error);
339:     }
340:     /// <summary>
341:     /// This method is used to Delete selected files from ListBox
342:     /// This is used different ways in this app
343:     /// So we created this as a method
344:     /// </summary>
345:     private void DeleteFilesFromListBox()
346:     {
347:         var allItems = new List<string>();
348:         var removedItems = new List<string>();
349:         var fileCount = lbFiles.SelectedItems.Count;
350:         for (int i = 0; i < lbFiles.SelectedItems.Count; i++)
351:         {
352:             removedItems.Add(lbFiles.SelectedItems[i].ToString());
353:         }
354:         foreach (var item in lbFiles.Items)
355:         {
356:             if (!removedItems.Any(j => j == item.ToString()))
357:                 allItems.Add(item.ToString());
358:         }
359:         lbFiles.Items.Clear();
360:         foreach (var item in allItems)
361:         {
362:             lbFiles.Items.Add(item);
363:         }
364:         MessageBox.Show(fileCount + " file/s are deleted from the list.",
365:             AppTitle, MessageBoxButtons.OK, MessageBoxIcon.Information);
366:     }
367:     /// <summary>
368:     /// This method is used to order items in listbox
369:     /// And return them with the given order as parametre
370:     /// </summary>
371:     /// <param name="lb">ListBox info</param>
372:     /// <param name="ascending">Order Type is Ascending Or Not</param>
373:     private static ListBox SortItems(ListBox lb, bool ascending)
374:     {
375:         List<object> items;
376:         items = lb.Items.OfType<object>().ToList();
377:         lb.Items.Clear();
378:         if (ascending)
379:             lb.Items.AddRange(items.OrderBy(i => i).ToArray());
380:         else
381:             lb.Items.AddRange(items.OrderByDescending(i => i).ToArray());
382:         return lb;
383:     }
384: }
385: }
386: }

```

Namespaces

Name	Description
PdfCombiner (see page 1)	This is namespace PdfCombiner.

1.2.4 FrmMain.Designer.cs

This is file FrmMain.Designer.cs.

Body Source

```

1: ?namespace PdfCombiner
2: {
3:     partial class FrmMain
4:     {
5:         /// <summary>
6:         /// Required designer variable.
7:         /// </summary>
8:         private System.ComponentModel.IContainer components = null;
9:
10:        /// <summary>
11:        /// Clean up any resources being used.
12:        /// </summary>
13:        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
14:        protected override void Dispose(bool disposing)
15:        {
16:            if (disposing && (components != null))
17:            {
18:                components.Dispose();
19:            }
20:            base.Dispose(disposing);
21:        }
22:
23:        #region Windows Form Designer generated code
24:
25:        /// <summary>
26:        /// Required method for Designer support - do not modify
27:        /// the contents of this method with the code editor.
28:        /// </summary>
29:        private void InitializeComponent()
30:        {
31:            this.components = new System.ComponentModel.Container();
32:            System.ComponentModel.ComponentResourceManager resources = new
System.ComponentModel.ComponentResourceManager(typeof(FrmMain));
33:            this.btnCombinePdfSharp = new System.Windows.Forms.Button();
34:            this.btnAddFolder = new System.Windows.Forms.Button();
35:            this.btnAddFile = new System.Windows.Forms.Button();
36:            this.btnClearList = new System.Windows.Forms.Button();
37:            this.lbFiles = new System.Windows.Forms.ListBox();
38:            this.lblDetails = new System.Windows.Forms.Label();
39:            this.pbFiles = new System.Windows.Forms.ProgressBar();
40:            this.btnCombineITextSharp = new System.Windows.Forms.Button();
41:            this.menuStrip = new
System.Windows.Forms.ContextMenuStrip(this.components);
42:            this.menuItemDelete = new System.Windows.Forms.ToolStripMenuItem();
43:            this.menuItemOrderAscending = new System.Windows.Forms.ToolStripMenuItem();
44:            this.menuItemOrderDescending = new
System.Windows.Forms.ToolStripMenuItem();
45:            this.menuStrip.SuspendLayout();
46:            this.SuspendLayout();
47:            //
48:            // btnCombinePdfSharp
49:            //

```



```

50:         this.btnCombinePdfSharp.Location = new System.Drawing.Point(487, 12);
51:         this.btnCombinePdfSharp.Name = "btnCombinePdfSharp";
52:         this.btnCombinePdfSharp.Size = new System.Drawing.Size(169, 23);
53:         this.btnCombinePdfSharp.TabIndex = 0;
54:         this.btnCombinePdfSharp.Text = "Combine Files (PdfSharp)";
55:         this.btnCombinePdfSharp.UseVisualStyleBackColor = true;
56:         this.btnCombinePdfSharp.Click += new
System.EventHandler(this.btnCombinePdfSharp_Click);
57:         //
58:         // btnAddFolder
59:         //
60:         this.btnAddFolder.Location = new System.Drawing.Point(12, 38);
61:         this.btnAddFolder.Name = "btnAddFolder";
62:         this.btnAddFolder.Size = new System.Drawing.Size(156, 23);
63:         this.btnAddFolder.TabIndex = 2;
64:         this.btnAddFolder.Text = "Add Folder";
65:         this.btnAddFolder.UseVisualStyleBackColor = true;
66:         this.btnAddFolder.Click += new
System.EventHandler(this.btnAddFolder_Click);
67:         //
68:         // btnAddFile
69:         //
70:         this.btnAddFile.Location = new System.Drawing.Point(12, 12);
71:         this.btnAddFile.Name = "btnAddFile";
72:         this.btnAddFile.Size = new System.Drawing.Size(156, 23);
73:         this.btnAddFile.TabIndex = 3;
74:         this.btnAddFile.Text = "Add Files";
75:         this.btnAddFile.UseVisualStyleBackColor = true;
76:         this.btnAddFile.Click += new System.EventHandler(this.btnAddFile_Click);
77:         //
78:         // btnClearList
79:         //
80:         this.btnClearList.Location = new System.Drawing.Point(174, 12);
81:         this.btnClearList.Name = "btnClearList";
82:         this.btnClearList.Size = new System.Drawing.Size(90, 49);
83:         this.btnClearList.TabIndex = 4;
84:         this.btnClearList.Text = "Clear File List";
85:         this.btnClearList.UseVisualStyleBackColor = true;
86:         this.btnClearList.Click += new
System.EventHandler(this.btnClearList_Click);
87:         //
88:         // lbFiles
89:         //
90:         this.lbFiles.FormattingEnabled = true;
91:         this.lbFiles.Location = new System.Drawing.Point(12, 88);
92:         this.lbFiles.Name = "lbFiles";
93:         this.lbFiles.SelectionMode =
System.Windows.Forms.SelectionMode.MultiExtended;
94:         this.lbFiles.Size = new System.Drawing.Size(644, 186);
95:         this.lbFiles.TabIndex = 5;
96:         this.lbFiles.KeyDown += new
System.Windows.Forms.KeyEventHandler(this.lbFiles_KeyDown);
97:         //
98:         // lblDetails
99:         //
100:        this.lblDetails.AutoSize = true;
101:        this.lblDetails.Location = new System.Drawing.Point(12, 65);
102:        this.lblDetails.Name = "lblDetails";
103:        this.lblDetails.Size = new System.Drawing.Size(0, 13);
104:        this.lblDetails.TabIndex = 6;
105:        //
106:        // pbFiles
107:        //
108:        this.pbFiles.Location = new System.Drawing.Point(270, 12);
109:        this.pbFiles.Name = "pbFiles";
110:        this.pbFiles.Size = new System.Drawing.Size(211, 49);
111:        this.pbFiles.TabIndex = 7;
112:        //
113:        // btnCombineITextSharp

```

```

114:         //
115:         this.btnCombineITextSharp.Location = new System.Drawing.Point(487, 38);
116:         this.btnCombineITextSharp.Name = "btnCombineITextSharp";
117:         this.btnCombineITextSharp.Size = new System.Drawing.Size(169, 23);
118:         this.btnCombineITextSharp.TabIndex = 9;
119:         this.btnCombineITextSharp.Text = "Combine Files (iTextSharp)";
120:         this.btnCombineITextSharp.UseVisualStyleBackColor = true;
121:         this.btnCombineITextSharp.Click += new
System.EventHandler(this.btnCombineITextSharp_Click);
122:         //
123:         // menuStrip
124:         //
125:         this.menuStrip.Items.AddRange(new System.Windows.Forms.ToolStripItem[] {
126:         this.menuItemDelete,
127:         this.menuItemOrderAscending,
128:         this.menuItemOrderDescending});
129:         this.menuStrip.Name = "menuStrip";
130:         this.menuStrip.Size = new System.Drawing.Size(178, 70);
131:         //
132:         // menuItemDelete
133:         //
134:         this.menuItemDelete.Image =
((System.Drawing.Image)(resources.GetObject("menuItemDelete.Image")));
135:         this.menuItemDelete.Name = "menuItemDelete";
136:         this.menuItemDelete.Size = new System.Drawing.Size(177, 22);
137:         this.menuItemDelete.Text = "Delete";
138:         this.menuItemDelete.Click += new
System.EventHandler(this.menuItemDelete_Click);
139:         //
140:         // menuItemOrderAscending
141:         //
142:         this.menuItemOrderAscending.Image =
((System.Drawing.Image)(resources.GetObject("menuItemOrderAscending.Image")));
143:         this.menuItemOrderAscending.Name = "menuItemOrderAscending";
144:         this.menuItemOrderAscending.Size = new System.Drawing.Size(177, 22);
145:         this.menuItemOrderAscending.Text = "Order (Ascending)";
146:         this.menuItemOrderAscending.Click += new
System.EventHandler(this.menuItemOrderAscending_Click);
147:         //
148:         // menuItemOrderDescending
149:         //
150:         this.menuItemOrderDescending.Image =
((System.Drawing.Image)(resources.GetObject("menuItemOrderDescending.Image")));
151:         this.menuItemOrderDescending.Name = "menuItemOrderDescending";
152:         this.menuItemOrderDescending.Size = new System.Drawing.Size(177, 22);
153:         this.menuItemOrderDescending.Text = "Order (Descending)";
154:         this.menuItemOrderDescending.Click += new
System.EventHandler(this.menuItemOrderDescending_Click);
155:         //
156:         // FrmMain
157:         //
158:         this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
159:         this.AutoScaleMode = System.Windows.Forms.AutoScaleModeMode.Font;
160:         this.ClientSize = new System.Drawing.Size(668, 287);
161:         this.Controls.Add(this.btnCombineITextSharp);
162:         this.Controls.Add(this.pbFiles);
163:         this.Controls.Add(this.lblDetails);
164:         this.Controls.Add(this.lbFiles);
165:         this.Controls.Add(this.btnClearList);
166:         this.Controls.Add(this.btnAddFile);
167:         this.Controls.Add(this.btnAddFolder);
168:         this.Controls.Add(this.btnCombinePdfSharp);
169:         this.Name = "FrmMain";
170:         this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
171:         this.Text = "PDF Combiner";
172:         this.FormClosed += new
System.Windows.Forms.FormClosedEventHandler(this.FrmMain_FormClosed);
173:         this.Shown += new System.EventHandler(this.FrmMain_Shown);
174:         this.menuStrip.ResumeLayout(false);

```

```
175:         this.ResumeLayout(false);
176:         this.PerformLayout();
177:
178:     }
179:
180:     #endregion
181:
182:     private System.Windows.Forms.Button btnCombinePdfSharp;
183:     private System.Windows.Forms.Button btnAddFolder;
184:     private System.Windows.Forms.Button btnAddFile;
185:     private System.Windows.Forms.Button btnClearList;
186:     private System.Windows.Forms.ListBox lbFiles;
187:     private System.Windows.Forms.Label lblDetails;
188:     private System.Windows.Forms.ProgressBar pbFiles;
189:     private System.Windows.Forms.Button btnCombineITextSharp;
190:     private System.Windows.Forms.ContextMenuStrip menuStrip;
191:     private System.Windows.Forms.ToolStripMenuItem menuItemDelete;
192:     private System.Windows.Forms.ToolStripMenuItem menuItemOrderAscending;
193:     private System.Windows.Forms.ToolStripMenuItem menuItemOrderDescending;
194: }
195: }
196:
```

Namespaces

Name	Description
PdfCombiner (see page 1)	This is namespace PdfCombiner.

1.2.5 PdfCombiner.csproj

This is file PdfCombiner.csproj.

1.2.6 PdfCombiner.sln

This is file PdfCombiner.sln.

1.2.7 Program.cs

This is file Program.cs.

Body Source

```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Windows.Forms;
5:
6: namespace PdfCombiner
7: {
8:     static class Program
9:     {
10:         /// <summary>
11:         /// The main entry point for the application.
12:         /// </summary>
13:         [STAThread]
14:         static void Main()
```

```
15:         {
16:             Application.EnableVisualStyles();
17:             Application.SetCompatibleTextRenderingDefault(false);
18:             Application.Run(new FrmMain());
19:         }
20:     }
21: }
```

Namespaces

Name	Description
PdfCombiner (see page 1)	This is namespace PdfCombiner.

1.2.8 Resources.Designer.cs

This code was generated by a tool. Runtime Version:4.0.30319.42000

Changes to this file may cause incorrect behavior and will be lost if the code is regenerated.

Body Source

```
1:  ?//-----
2:  // <auto-generated>
3:  //     This code was generated by a tool.
4:  //     Runtime Version:4.0.30319.42000
5:  //
6:  //     Changes to this file may cause incorrect behavior and will be lost if
7:  //     the code is regenerated.
8:  // </auto-generated>
9:  //-----
10:
11: namespace PdfCombiner.Properties {
12:     using System;
13:
14:
15:     /// <summary>
16:     ///     A strongly-typed resource class, for looking up localized strings, etc.
17:     /// </summary>
18:     // This class was auto-generated by the StronglyTypedResourceBuilder
19:     // class via a tool like ResGen or Visual Studio.
20:     // To add or remove a member, edit your .ResX file then rerun ResGen
21:     // with the /str option, or rebuild your VS project.
22:
23: [global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Resources.Tools.StronglyTypedResourceBuilder",
24: "15.0.0.0")]
25:     [global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
26:     [global::System.Runtime.CompilerServices.CompilerGeneratedAttribute()]
27:     internal class Resources {
28:
29:         private static global::System.Resources.ResourceManager resourceMan;
30:
31:         private static global::System.Globalization.CultureInfo resourceCulture;
32:
33: [global::System.Diagnostics.CodeAnalysis.SuppressMessageAttribute("Microsoft.Performance",
34: "CA1811:AvoidUncalledPrivateCode")]
35:         internal Resources() {
36:
37:             /// <summary>
38:             ///     Returns the cached ResourceManager instance used by this class.
39:             /// </summary>
40:
41: [global::System.ComponentModel.EditorBrowsableAttribute(global::System.ComponentModel.Editor
```

```
BrowsableState.Advanced)]
39:         internal static global::System.Resources.ResourceManager ResourceManager {
40:             get {
41:                 if (object.ReferenceEquals(resourceMan, null)) {
42:                     global::System.Resources.ResourceManager temp = new
global::System.Resources.ResourceManager("PdfCombiner.Properties.Resources",
typeof(Resources).Assembly);
43:                     resourceMan = temp;
44:                 }
45:                 return resourceMan;
46:             }
47:         }
48:
49:         /// <summary>
50:         ///     Overrides the current thread's CurrentUICulture property for all
51:         ///     resource lookups using this strongly typed resource class.
52:         /// </summary>
53:
54:         [global::System.ComponentModel.EditorBrowsableAttribute(global::System.ComponentModel.EditorBrowsableState.Advanced)]
55:         internal static global::System.Globalization.CultureInfo Culture {
56:             get {
57:                 return resourceCulture;
58:             }
59:             set {
60:                 resourceCulture = value;
61:             }
62:         }
63:     }
```

Namespaces

Name	Description
PdfCombiner (see page 1)	This is namespace PdfCombiner.

1.2.9 Settings.Designer.cs

This code was generated by a tool. Runtime Version:4.0.30319.42000

Changes to this file may cause incorrect behavior and will be lost if the code is regenerated.

Body Source

```
1: ?//-----
2: // <auto-generated>
3: //     This code was generated by a tool.
4: //     Runtime Version:4.0.30319.42000
5: //
6: //     Changes to this file may cause incorrect behavior and will be lost if
7: //     the code is regenerated.
8: // </auto-generated>
9: //-----
10:
11: namespace PdfCombiner.Properties
12: {
13:
14:
15:     [global::System.Runtime.CompilerServices.CompilerGeneratedAttribute()]
16:
17:     [global::System.CodeDom.Compiler.GeneratedCodeAttribute("Microsoft.VisualStudio.Editors.SettingsDesigner.SettingsSingleFileGenerator",
"11.0.0.0")]
18:     internal sealed partial class Settings :
global::System.Configuration.ApplicationSettingsBase
```

```
18:     {
19:
20:         private static Settings defaultInstance =
21:         ((Settings)(global::System.Configuration.ApplicationSettingsBase.Synchronized(new
22:         Settings())));
23:
24:         public static Settings Default
25:         {
26:             get
27:             {
28:                 return defaultInstance;
29:             }
30: }
```

Namespaces

Name	Description
PdfCombiner (see page 1)	This is namespace PdfCombiner.

Index

A

AssemblyInfo.cs 22

B

BaseException class 4

about BaseException class 4

BaseException 5

E

ExceptionExtensions class 5

about ExceptionExtensions class 5

GetAllMessages 6

Exceptions.cs 22

F

Files 21

FrmMain class 6

about FrmMain class 6

AppTitle 8

btnAddFile 8

btnAddFile_Click 10

btnAddFolder 8

btnAddFolder_Click 11

btnClearList 8

btnClearList_Click 11

btnCombineITextSharp 9

btnCombineITextSharp_Click 12

btnCombinePdfSharp 9

btnCombinePdfSharp_Click 13

components 9

DeleteFilesFromListBox 14

Dispose 15

FrmMain 8

FrmMain_FormClosed 15

FrmMain_Shown 15

InitializeComponent 16

InitializeFileDialog 18

lbFiles 9

lbFiles_KeyDown 19

lblDetails 9

menuItemDelete 9

menuItemDelete_Click 19

menuItemOrderAscending 9

menuItemOrderAscending_Click 19

menuItemOrderDescending 10

menuItemOrderDescending_Click 20

menuStrip 10

pbFiles 10

SortItems 20

FrmMain.cs 23

FrmMain.Designer.cs 30

P

PdfCombiner 1

PdfCombiner namespace 1

Classes 4

PdfCombiner.BaseException 4

PdfCombiner.BaseException.BaseException 5

PdfCombiner.csproj 33

PdfCombiner.ExceptionExtensions 5

PdfCombiner.ExceptionExtensions.GetAllMessages 6

PdfCombiner.FrmMain 6

PdfCombiner.FrmMain.AppTitle 8

PdfCombiner.FrmMain.btnAddFile 8

PdfCombiner.FrmMain.btnAddFile_Click 10

PdfCombiner.FrmMain.btnAddFolder 8

PdfCombiner.FrmMain.btnAddFolder_Click 11

PdfCombiner.FrmMain.btnClearList 8

PdfCombiner.FrmMain.btnClearList_Click 11

PdfCombiner.FrmMain.btnCombineITextSharp 9

PdfCombiner.FrmMain.btnCombineITextSharp_Click 12

PdfCombiner.FrmMain.btnCombinePdfSharp 9

PdfCombiner.FrmMain.btnCombinePdfSharp_Click 13

PdfCombiner.FrmMain.components 9

PdfCombiner.FrmMain.DeleteFilesFromListBox 14

PdfCombiner.FrmMain.Dispose 15

PdfCombiner.FrmMain.FrmMain 8

PdfCombiner.FrmMain.FrmMain_FormClosed 15
 PdfCombiner.FrmMain.FrmMain_Shown 15
 PdfCombiner.FrmMain.InitializeComponent 16
 PdfCombiner.FrmMain.InitializeFileDialog 18
 PdfCombiner.FrmMain.lbFiles 9
 PdfCombiner.FrmMain.lbFiles_KeyDown 19
 PdfCombiner.FrmMain.lblDetails 9
 PdfCombiner.FrmMain.menuItemDelete 9
 PdfCombiner.FrmMain.menuItemDelete_Click 19
 PdfCombiner.FrmMain.menuItemOrderAscending 9
 PdfCombiner.FrmMain.menuItemOrderAscending_Click 19
 PdfCombiner.FrmMain.menuItemOrderDescending 10
 PdfCombiner.FrmMain.menuItemOrderDescending_Click 20
 PdfCombiner.FrmMain.menuStrip 10
 PdfCombiner.FrmMain.pbFiles 10
 PdfCombiner.FrmMain.SortItems 20
 PdfCombiner.Program 21
 PdfCombiner.Program.Main 21
 PdfCombiner.Properties 1
 PdfCombiner.Properties namespace 1
 Classes 1
 PdfCombiner.Properties.Resources 1
 PdfCombiner.Properties.Resources.Culture 3
 PdfCombiner.Properties.Resources.resourceCulture 2
 PdfCombiner.Properties.Resources.resourceMan 3
 PdfCombiner.Properties.Resources.ResourceManager 3
 PdfCombiner.Properties.Resources.Resources 2
 PdfCombiner.Properties.Settings 3
 PdfCombiner.Properties.Settings.Default 4
 PdfCombiner.Properties.Settings.defaultInstance 4
 PdfCombiner.sln 33
 Program class 21
 about Program class 21
 Main 21
 Program.cs 33

S

Settings class 3
 about Settings class 3
 Default 4
 defaultInstance 4
 Settings.Designer.cs 35

R

Resources class 1
 about Resources class 1
 Culture 3