

FormFieldReader.aspx

```
1 <%@ Page language="c#" Inherits="Web_CSharp.MergerExamples.FormFieldReader" CodeFile="FormFieldReader.aspx.cs" %>
2
```

FormFieldReader.aspx.cs

```
1  using System;
2  using ceTe.DynamicPDF;
3  using ceTe.DynamicPDF.Merger;
4  using ceTe.DynamicPDF.Merger.Forms;
5  using ceTe.DynamicPDF.PageElements;
6
7  namespace Web_CSharp.MergerExamples
8  {
9      public partial class FormFieldReader : System.Web.UI.Page
10     {
11         protected void Page_Load(object sender, System.EventArgs e)
12         {
13             // PDF to read form fields from
14             PdfDocument pdfDocument = new PdfDocument( MapPath("../PDFs/fw9AcroForm_14_filled.pdf") );
15
16             // Create Table to display the form fields and values
17             Table2 table = new Table2( 0, 0, 512, 676, ceTe.DynamicPDF.Font.Helvetica, 12 );
18             table.Border.Width = 1;
19             table.Border.Color = RgbColor.Black;
20             table.RepeatColumnHeaderCount = 1;
21             table.RepeatRowHeaderCount = 1;
22             BuildTable( table );
23             CreateList( table, pdfDocument.Form.Fields );
24
25             Document document = new Document();
26             AddTableToPage( document, table );
27
28             Table2 overflowRowTable = table.GetOverflowRows();
29             while( overflowRowTable != null )
30             {
31                 AddTableToPage( document, overflowRowTable );
32                 overflowRowTable = overflowRowTable.GetOverflowRows();
33             }
34             document.DrawToWeb( "FormFieldReader.pdf" );
35         }
36
37         private void BuildTable( Table2 table )
38         {
39             CreateColumns( table );
40             CreateRowHeadings( table );
41         }
42
43         private void CreateList( Table2 table, ceTe.DynamicPDF.Merger.Forms.PdfFormFieldList fieldList )
44         {
45             for( int i = 0; i < fieldList.Count; i++ )
46             {
47                 Row2 row = table.Rows.Add( 20 );
48                 row.Cells.Add( fieldList[i].FullName );
49                 row.Cells.Add( GetFieldType(fieldList[i]) );
50                 row.Cells.Add( fieldList[i].GetValue() );
```

FormFieldReader.aspx.cs

```
51             if( fieldList[i].HasChildFields == true )
52             {
53                 CreateList( table, fieldList[i].ChildFields );
54             }
55         }
56     }
57
58     private string GetFieldType(PdfFormField pdfFormField)
59     {
60         if(pdfFormField is PdfTextField)
61         {
62             return "Text Field";
63         }
64         if(pdfFormField is PdfButtonField)
65         {
66             return "Button Field";
67         }
68         if(pdfFormField is PdfChoiceField)
69         {
70             return "Choice Field";
71         }
72         if(pdfFormField is PdfSignatureField)
73         {
74             return "Signature Field";
75         }
76         return "Container Field";
77     }
78
79     private void AddTableToPage( Document document, Table2 table )
80     {
81         Page page = new Page( PageSize.Letter );
82         if( table != null )
83             page.Elements.Add( table );
84         document.Pages.Add( page );
85     }
86
87     private void CreateRowHeadings( Table2 table )
88     {
89         Row2 row = table.Rows.Add( 40, Font.TimesBold, 12, RgbColor.Black, RgbColor.LightGrey );
90         row.CellDefault.Align = TextAlign.Center;
91         row.CellDefault.VAlign = VAlign.Top;
92         row.Cells.Add( "FormField Name" );
93         row.Cells.Add( "FormField Type" );
94         row.Cells.Add( "FormField Value" );
95     }
96
97     private void CreateColumns( Table2 table )
98     {
99         table.Columns.Add( 150 );
100        table.Columns.Add( 150 );
```

FormFieldReader.aspx.cs

```
101         table.Columns.Add( 212 );
102     }
103
104     #region Web Form Designer generated code
105     override protected void OnInit(EventArgs e)
106     {
107         //
108         // CODEGEN: This call is required by the ASP.NET Web Form Designer.
109         //
110         InitializeComponent();
111         base.OnInit(e);
112     }
113
114     /// <summary>
115     /// Required method for Designer support - do not modify
116     /// the contents of this method with the code editor.
117     /// </summary>
118     private void InitializeComponent()
119     {
120     }
121     #endregion
122 }
123 }
```