

www.bytescout.com

How to extract CSV from PDF and fill database (sql server) with PDF extractor SDK in C# using ByteScout Premium Suite

Step-by-step tutorial on how to extract CSV from PDF and fill database (sql server) with PDF extractor SDK in C#

We made thousands of pre-made source code pieces for easy implementation in your own programming projects. ByteScout Premium Suite is the bundle that includes twelve SDK products from ByteScout including tools and components for PDF, barcodes, spreadsheets, screen video recording. It can extract CSV from PDF and fill database (sql server) with PDF extractor SDK in C#.

The SDK samples given below describe how to quickly make your application do extract CSV from PDF and fill database (sql server) with PDF extractor SDK in C# with the help of ByteScout Premium Suite. IF you want to implement the functionality, just copy and paste this code for C# below into your code editor with your app, compile and run your application. If you want to use these C# sample examples in one or many applications then they can be used easily.

ByteScout Premium Suite free trial version is available on our website. C# and other programming languages are supported.

FOR MORE INFORMATION AND FREE TRIAL:

[Download Free Trial SDK \(on-premise version\)](#)

[Read more about ByteScout Premium Suite](#)

[Explore API Documentation](#)

[Get Free Training for ByteScout Premium Suite](#)

[Get Free API key for Web API](#)

[visit \[www.ByteScout.com\]\(http://www.ByteScout.com\)](#)

Source Code Files:

Program.cs

```
using Bytescout.PDFExtractor;
using System;
using System.Data;
using System.Data.SqlClient;

namespace ExtractCsvAndFillDatabase
{
    class Program
    {
        static void Main(string[] args)
        {
            try
            {
                // Step-1: Get Datatable
                var oDataTable = GetDataTableFromDocument("sample.pdf");

                // PLEASE NOTE: Please replace with your connection string, You need to
                // You can find that table from Scripts.sql file
                string connectionString = @"Data Source=REPLACE_WITH_YOUR_DATA_SOURCE;";

                // Step-2: Insert into database
                InsertIntoSqlServerDatabase(oDataTable, connectionString);

                // Step-3: Fetch from database and display results
                DisplayDatabaseResults(connectionString);
            }
            catch (Exception ex)
            {
                Console.WriteLine(ex.Message);
            }

            Console.WriteLine("Press enter key to exit...");
            Console.ReadLine();
        }

        /// <summary>
        /// Inserts into Sql Server database
        /// </summary>
        /// <param name="oDataTable"></param>
        private static void InsertIntoSqlServerDatabase(DataTable oDataTable, string connectionString)
        {
            using (SqlConnection con = new SqlConnection(connectionString))
            {
                // Open connection
                con.Open();

                // Sql query to insert data
                string cmdInsert = "Insert into PersonData (id, first_name, last_name,";

                foreach (DataRow itemRow in oDataTable.Rows)
                {
                    // Prepare sql command
                    SqlCommand cmd = new SqlCommand(cmdInsert, con);
                    cmd.CommandType = CommandType.Text;
                    cmd.Parameters.AddWithValue("@id", itemRow["id"]);
                    cmd.Parameters.AddWithValue("@first_name", itemRow["first_name"]);
                    cmd.Parameters.AddWithValue("@last_name", itemRow["last_name"]);
                    cmd.Parameters.AddWithValue("@email", itemRow["email"]);
                    cmd.Parameters.AddWithValue("@phone", itemRow["phone"]);
                    cmd.Parameters.AddWithValue("@city", itemRow["city"]);
                    cmd.Parameters.AddWithValue("@state", itemRow["state"]);
                    cmd.Parameters.AddWithValue("@zip", itemRow["zip"]);
                    cmd.ExecuteNonQuery();
                }
            }
        }
    }
}
```

```
        cmd.Parameters.Add(new SqlParameter("@id", Convert.ToString(itemRow["id"])));
        cmd.Parameters.Add(new SqlParameter("@first_name", Convert.ToString(itemRow["first_name"])));
        cmd.Parameters.Add(new SqlParameter("@last_name", Convert.ToString(itemRow["last_name"])));
        cmd.Parameters.Add(new SqlParameter("@email", Convert.ToString(itemRow["email"])));
        cmd.Parameters.Add(new SqlParameter("@gender", Convert.ToString(itemRow["gender"])));
        cmd.Parameters.Add(new SqlParameter("@ip_address", Convert.ToString(itemRow["ip_address"])));

        // Execute sql command
        cmd.ExecuteNonQuery();
    }

    // Close connection
    con.Close();
}

/// <summary>
/// Displays inserted database results
/// </summary>
private static void DisplayDatabaseResults(string connectionString)
{
    // Person data holder
    DataTable personDataTable = new DataTable();

    using (SqlConnection con = new SqlConnection(connectionString))
    {
        // Sql query to fetch data
        string cmdInsert = "SELECT id, first_name, last_name, email, gender, ip_address FROM Person";

        // Prepare sql command
        SqlCommand cmd = new SqlCommand(cmdInsert, con);
        cmd.CommandType = CommandType.Text;

        // Prepare DataAdapter
        SqlDataAdapter dataAdapter = new SqlDataAdapter(cmd);

        // Fill person dataTable
        dataAdapter.Fill(personDataTable);
    }

    // Display all person data if any
    if (personDataTable != null && personDataTable.Rows.Count > 0)
    {
        // Print all columns
        foreach ( DataColumn column in personDataTable.Columns)
        {
            Console.Write("{0} | ", column.ColumnName);
        }
        Console.WriteLine();

        // Print all data
        foreach ( DataRow dataRow in personDataTable.Rows)
        {
            foreach ( DataColumn column in personDataTable.Columns)
            {
                Console.Write("{0} | ", dataRow[column.ColumnName]);
            }
            Console.WriteLine();
        }
    }
}
```

```

        else
    {
        Console.WriteLine("No data retrieved..");
    }
}

/// <summary>
/// Get DataTable from Document
/// </summary>
private static DataTable GetDataTableFromDocument(string fileName)
{
    DataTable oDataTable = null;

    // Initialise table detector
    using (TableDetector tableDetector = new TableDetector("demo", "demo"))
    {
        using (CSVExtractor CSVExtractor = new CSVExtractor("demo", "demo"))
        {
            // Set table detection mode to "bordered tables" - best for tables
            tableDetector.ColumnDetectionMode = ColumnDetectionMode.BorderedTable;

            // We should define what kind of tables we should detect.
            // So we set min required number of columns to 2 ...
            tableDetector.DetectionMinNumberOfColumns = 2;
            // ... and we set min required number of rows to 2
            tableDetector.DetectionMinNumberOfRows = 2;

            // Load PDF document
            tableDetector.LoadDocumentFromFile(fileName);
            CSVExtractor.LoadDocumentFromFile(fileName);

            // Get page count
            int pageCount = tableDetector.GetPageCount();

            if (tableDetector.FindTable(0))
            {
                // Set extraction area for CSV extractor to rectangle received
                CSVExtractor.SetExtractionArea(tableDetector.FoundTableLocation);

                // Generate CSV data
                var allCsvData = CSVExtractor.GetCSV();

                // Generate Datatable
                oDataTable = GetDataTableFromCSV(allCsvData);
            }
        }
    }

    return oDataTable;
}

/// <summary>
/// Get Datatable from CSV
/// </summary>
private static DataTable GetDataTableFromCSV(string allCsvData)
{
    var oRetDataTable = new DataTable();
    oRetDataTable.Columns.Add("id");
    oRetDataTable.Columns.Add("first_name");
}

```

```
oRetDataTable.Columns.Add("last_name");
oRetDataTable.Columns.Add("email");
oRetDataTable.Columns.Add("gender");
oRetDataTable.Columns.Add("ip_address");

var rows = allCsvData.Split('\n');

// Ignore first column line
for (int iRow = 1; iRow < rows.Length; iRow++)
{
    // Get all column data
    var columns = rows[iRow].Split(',', ',');

    if (columns.Length >= 5)
    {
        // Prepare new row
        var oRow = oRetDataTable.NewRow();
        oRow["id"] = columns[0];
        oRow["first_name"] = columns[1];
        oRow["last_name"] = columns[2];
        oRow["email"] = columns[3];
        oRow["gender"] = columns[4];
        oRow["ip_address"] = columns[5];

        // Add row back to datatable
        oRetDataTable.Rows.Add(oRow);
    }
}

// Return DataTable
return oRetDataTable;
}
}
```

VIDEO

<https://www.youtube.com/watch?v=NEwNs2b9YN8>

ON-PREMISE OFFLINE SDK

[60 Day Free Trial](#) or [Visit ByteScout Premium Suite Home Page](#)
[Explore ByteScout Premium Suite Documentation](#)
[Explore Samples](#)

[Sign Up for ByteScout Premium Suite Online Training](#)

ON-DEMAND REST WEB API

[Get Your API Key](#)

[Explore Web API Docs](#)

[Explore Web API Samples](#)

[visit www.ByteScout.com](#)

[visit www.PDF.co](#)

[www.bytescout.com](#)