Prerequisites

Note:

The code for the examples has been simplified to show a specific area of interest and does not include proper error checking. As such, the example code is not ready for production purposes and should be seen as an illustration for a specific concept only. It is up to you, the developer, to catch exceptions and handle error situations appropriately. Your application should react gracefully to those conditions and make no assumptions about undocumented behavior.

Visual Studio and .NET

- Visual Studio 2010 (SP1 recommended) or Visual C# Express 2010, Visual Studio 2013 or Visual Studio 2015
- .NET Framework 4.0 (in Windows 7, installed by default)

The example projects were built in C#4.0 with Visual Studio 2010 (SP1). Using Visual Basic.NET or other languages targeting the CLR 4.0 is possible but not supported at this time.

Referencing the JCamNet assembly from a project

The **JCamNet** assembly is delivered as a DLL named **JoeScan.JCamNet.dll** in the **lib** subfolder of the **JCamNet** distribution. At the same location you will find **JoeScan.JCamNet.xml**, which aides Visual Studio in providing IntelliSense help, such as parameter info and function names.

Adding a Reference

You need to add a reference to the **JCamNet** assembly to your project. Here's how:

- In Visual Studio, right-click on your project in the Solution Explorer, and select **Add Reference...**,
- Switch to the "Browse" tab,
- Navigate to the directory where **JoeScan.JCamNet.dll** is stored, and click **OK**.

The result should look like this:

blocked URL

You project will now list JoeScan.JCamNet under its References, and all publicly accessible symbols in the API are available to you. If the **JoeScan.JCamNet.xml** file is at the same file location (you don't need to reference it), IntelliSense in Visual Studio will automatically show the help when using symbols from the **JCamNet** API. In the example projects, this has already been done for you.

The JoeScan.JCamNet namespace

Including the line

```
using JoeScan.JCamNet;
```

at the top of your source file will import the namespace and allow you to use the symbols from the assembly with their short names (e.g., Scanner instead of JoeScan.JCamNet.Scanner). In the example code, this has already been done for you.

The param.dat file

Scan heads are configured by sending a text string of parameters to them. Almost all applications keep this data in a text file, and per convention it is typically named param.dat. For the meaning and format of the parameters, please see the documentation at <link>. The parameters file used to configure the heads needs to be accessible from your application, so it can be read and its contents sent to the scanner. You can either keep this file in a fixed location on your hard drive and refer to it by an absolute path, or keep it co-located with your application. In the example projects, the latter approach was taken. To do the same in your projects:

- In Visual Studio, right click your project in the Solution Explorer, and select **Add**, thenExisting Item,
- In the file dialog opening, change the file type from "Visual C# Files (...)" to "All Files (*.*)",
- Navigate to and pick the "param.dat" file you want to use, and click **Add**.

The file "param.dat" is now part of your project, however, Visual Studio still needs to be told what to do with the file when the project is built. In our case, we want the file to be copied to the output directory where your application (exe) is located. To do so:

- Right-click **param.dat** and select **Properties.**
- In the Property Editor that opens, set the property **Copy to Output Directory** to **Copy always**.

blocked URL

Remember that any changes you make to the parameter file in the JSDiag application will not be reflected in your local copy. Therefore it is advised that you deploy your application in such a way that JSDiag and your application run from the same directory and share the same param.dat.