First Look: ADO.NET and InterSystems IRIS

Version 2018.1
2018-01-31

InterSystems Corporation 1 Memorial Drive Cambridge MA 02142 www.intersystems.com

First Look: ADO.NET and InterSystems IRIS
InterSystems Version 2018.1 2018-01-31

For Support questions about any InterSystems products, contact:

InterSystems Worldwide Response Center (WRC)
Tel: +1-617-621-0700
Tel: +44 (0) 844 854 2917
Email: support@InterSystems.com

Table of Contents

First Look: ADO.NET and InterSystems IRIS

This First Look guide explains how to connect to InterSystems IRIS™ via the InterSystems ADO.NET Managed Provider. Once you have completed this guide, you will have configured Visual Studio to use the InterSystems.Data.SqlClient.dll assembly, established an ADO.NET connection to InterSystems IRIS, run several SQL queries from your .NET application, and confirmed the effects of these queries in the InterSystems IRIS System Management Portal.

To give you a taste of the ADO.NET Managed Provider without bogging you down in details, we've kept this exploration simple. These activities are designed to only use the default settings and features, so that you can acquaint yourself with the fundamentals of the feature without having to deal with details that are off-topic or overly complicated. When you bring ADO.NET to your production systems, there may be things you will need to do differently. Be sure not to confuse this exploration of ADO.NET with the real thing! The sources provided at the end of this document will give you a good idea of what is involved in using ADO.NET in production.

For more documentation on ADO.NET, see *Using .NET and the ADO.NET Managed Provider with InterSystems IRIS*.

1 Why ADO.NET Is Important

ADO.NET is a database technology from the Microsoft .NET Framework that provides access to data sources. It is used to establish database connectivity and provides a standard, reliable way for .NET Framework programmers to connect to many types of data sources or perform operations on them with SQL. Connecting to InterSystems IRIS via the ADO.NET Managed Provider is simple, especially if you've used ADO.NET before. Establishing an ADO.NET connection to InterSystems IRIS from a .NET application allows you to run SQL commands against InterSystems IRIS databases from your .NET application.

If you're new to InterSystems IRIS but familiar with .NET and SQL, you can use your existing expertise right away to help you become familiar with the database platform. You can test ADO.NET connections and SQL commands in a development environment with just a few lines of code.

2 ADO.NET and InterSystems IRIS

InterSystems IRIS is a fully compliant implementation of the ADO.NET specification. The InterSystems ADO.NET Managed Provider allows you to access InterSystems IRIS as an ADO.NET-compliant database and provides easy relational access to data. It processes ADO.NET function calls from applications and submits SQL requests to InterSystems IRIS. It then returns results to the calling application — in this case, your .NET application.

Connecting to InterSystems IRIS via ADO.NET is a very straightforward process.

In order to use InterSystems IRIS ADO.NET capability, you must first configure Visual Studio to use the InterSystems.Data.SqlClient.dll assembly. After confirming a few settings, use our sample code to establish an ADO.NET connection to InterSystems IRIS and to execute SQL queries. Note that the InterSystems.Data.SqlClient.dll assembly is implemented using .NET managed code throughout, making it easy to deploy within a .NET environment. It is thread-safe and can be used within multithreaded .NET applications.

3 Exploring ADO.NET

We have developed a brief demo that shows you how to work with ADO.NET and InterSystems IRIS.

3.1 Before you Begin

To run the demo, you'll need a single Windows 10 machine with a running, licensed instance of InterSystems IRIS and version 4.5 of the Microsoft .NET Framework. The InterSystems.Data.SqlClient.dll assembly is installed with InterSystems IRIS and requires no special preparation. In this demo, we will use Visual Studio Community 2017.

For instructions on how to install and license a development instance of InterSystems IRIS, see Quick Start: InterSystems IRIS Installation.

3.2 Things to Note About the Sample

The connection string syntax for the InterSystems ADO.NET Managed Provider is:

```
Server=host_IP; Port=SuperServerPort; Namespace=namespace; Password=password; User ID=user;
```

Note the following:

- *SuperServerPort* — The SuperServer port is distinct from the Web server port for InterSystems IRIS, and is set at installation time. To find the superserver port number, open the Management Portal for InterSystems IRIS and navigate to **System Administration** > **Configuration** > **System Configuration** > **Memory and Startup**.
- *namespace* — You must connect to a specific existing namespace in your InterSystems IRIS instance. In this demo, we connect to the **USER** namespace.

3.3 Configuring Visual Studio

In the Visual Studio main menu, create a new Project by selecting **File** > **New** > **Project**. In the resulting dialog, click the **Visual C#** option, and choose **Console App (.NET Framework)**. For the **Name** field, enter ADONET. This should create a new console application using the .NET Framework.

Next, in the Visual Studio main menu, select **Project** > **ADONET Properties**. Under **Target framework**, select **.NET Framework 4.5**.

3.3.1 Adding the Assembly Reference

To add an InterSystems.Data.SqlClient.dll assembly reference to a project:

1. From the Visual Studio main menu, select **Project** > **Add Reference...**
2. In the resulting window, click **Browse...**.
3. Browse to the subdirectory *<install-dir>*\dev\dotnet\bin\v4.5, where <*install-dir*> is the installation directory for your instance of InterSystems IRIS.
4. Select InterSystems.Data.SqlClient.dll and click **Add**.
5. In the Visual Studio Solution Explorer, the InterSystems.Data.SqlClient.dll assembly should now be listed under **References**.

3.4 Connecting via ADO.NET

At this point, you are ready to connect to InterSystems IRIS from your .NET application. Remove the default template code and paste the following code into Visual Studio:

```
using System;
using InterSystems.Data.CacheClient;

namespace ADONET
{
    class Program
    {
        static void Main(string[] args)
        {
            CacheConnection CacheConnect = new CacheConnection();
            CacheConnect.ConnectionString = "Server = localhost; "
              + "Port = [YOUR PORT HERE]; " + "Namespace = USER; "
              + "Password = SYS; " + "User ID = _SYSTEM;";


            CacheConnect.Open();


            String queryString = "CREATE TABLE People(ID int, FirstName varchar(255), LastName
varchar(255))";
            String queryString2 = "INSERT INTO People VALUES (1, 'John', 'Smith')";
            String queryString3 = "INSERT INTO People VALUES (2, 'Jane', 'Doe')";
            String queryString4 = "SELECT * FROM People";


            CacheCommand cmd = new CacheCommand(queryString, CacheConnect);
            CacheCommand cmd2 = new CacheCommand(queryString2, CacheConnect);
            CacheCommand cmd3 = new CacheCommand(queryString3, CacheConnect);
            CacheCommand cmd4 = new CacheCommand(queryString4, CacheConnect);


            //ExecuteNonQuery() is used for CREATE, INSERT, UPDATE, and DELETE
            cmd.ExecuteNonQuery();
            cmd2.ExecuteNonQuery();
            cmd3.ExecuteNonQuery();

            //ExecuteReader() is used for SELECT
            CacheDataReader Reader = cmd4.ExecuteReader();

            Console.WriteLine("Printing out contents of SELECT query: ");
            while (Reader.Read())
            {

                Console.WriteLine(Reader.GetValue(0).ToString() + ", " + Reader.GetValue(1).ToString()
 + ", " + Reader.GetValue(2).ToString());

            }

            Reader.Close();
            cmd.Dispose();
            cmd2.Dispose();
            cmd3.Dispose();
            cmd4.Dispose();
            CacheConnect.Close();

            Console.WriteLine("Press any key to continue...");
            Console.ReadKey();

        }
    }
}
```

Make sure to edit your connection string to have accurate `Port`, `User ID`, and `Password` values.

Run the code by clicking the **Start** button, or by pressing F5.

If the connection and queries have completed successfully, you should see a console window containing the results of the SELECT query.

3.5 Confirming the Changes in the System Management Portal

Next, you will want to confirm the results of the queries in the System Management Portal. First, ensure that you are in the USER namespace. Navigate to the **SQL** page (**System Explorer** > **SQL**). Click the **Execute Query** tab and paste in the following SQL query:

```
SELECT
ID, FirstName, LastName
FROM SQLUser.People
```

Click **Execute**.

The page should display the contents of the People table created in the sample code.

4 For More Information about ADO.NET

For more information on ADO.NET, SQL, and InterSystems IRIS, see:

Using .NET and the ADO.NET Managed Provider with InterSystems IRIS

Using InterSystems IRIS ADO.NET Managed Provider Classes

Using InterSystems SQL

ADO.NET Overview