Accessing Zen v15 from C# on Windows Using ADO.NET

A White Paper From

GOLDSTAR SOFTWARE
www.GoldstarSoftware.com

For more information, see our web site at
http://www.goldstarsoftware.com

Accessing Zen v15 from C# on Windows Using ADO.NET

Last Updated: August 2022

The Actian Zen database engine (formerly known as Actian PSQL) supports a wide variety of application programming interfaces (APIs) to access the data. Some of these interfaces leverage a lower-level interface, commonly known as the Btrieve API to provide the needed performance and flexibility. However, for rapid development of applications, it is hard to beat the productivity of the more advanced interfaces like ADO.NET.

This paper defines the minimum of steps required to build a simple ADO.NET application using the Actian Zen v15 database environment. The example is a bit contrived – we use a single button in the GUI dialog box to open a single SQL table and display its records in a simple grid. However, this is mainly done to keep the sample code easy to understand and to concentrate on the core requirements.

Important Pre-requisites

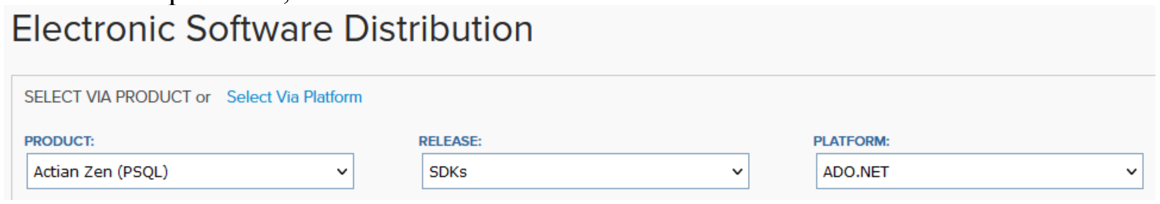
The following pre-requisites should be in place before you start.

1) You should have a Windows operating system properly installed.
2) You should have Microsoft Visual Studio properly installed. Screenshots within this paper are from VS2019, but the requirements should be similar for both newer and older versions. If you need help with this, please see Microsoft's documentation.
3) You should have Actian Zen v15 installed and running. If you are working on a stand-alone development machine, you can install either the Workgroup Engine or the Server Engine. (Both will work just fine with a 30-day trial.) If you are working within a shared environment, you should have a Zen v15 Workgroup Engine or Server Engine installed where the database files are located, and the Zen v15 Client installed on your development computer. See our install documentation at http://www.goldstarsoftware.com/ineedzen15.asp for complete instructions if needed.

Download and Install the ADO.NET SDK

Actian has a formal SDK download for the ADO.NET components, which you can find on their web site following these directions.

1) Go to https://esd.actian.com/product/Zen_PSQL in a web browser.
2) In the boxes provided, select **SDKs** and **ADO.NET**.



Electronic Software Distribution

SELECT VIA PRODUCT or Select Via Platform

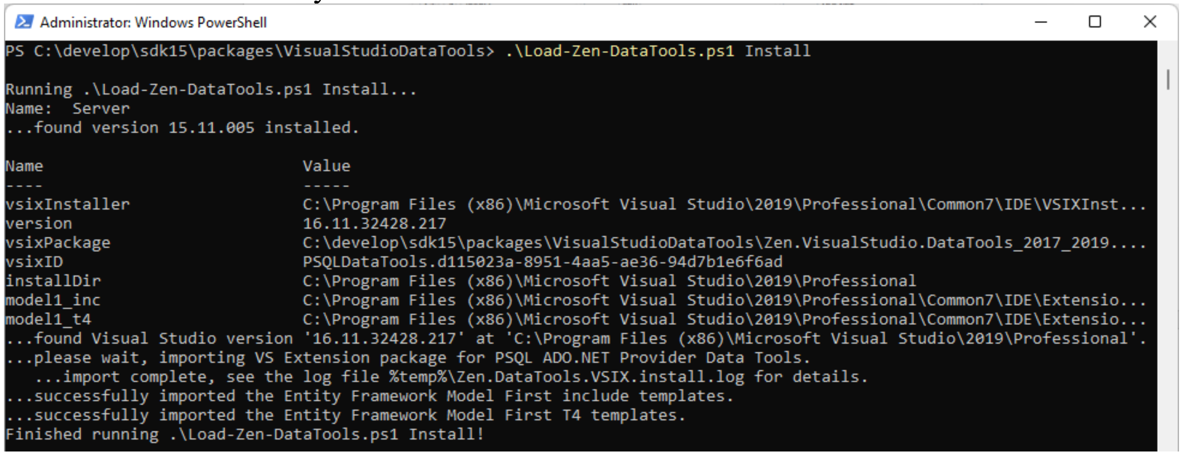
PRODUCT: Actian Zen (PSQL)

RELEASE: SDKs

PLATFORM: ADO.NET

3) Scroll down and open up the link for **ADO.NET**, then click on the **DOWNLOAD** button to download the needed SDK component.(Note that there are multiple versions – for both ADO.NET 4.4 and 4.5, as well as different downloads for .Net Framework and .Net Standard, so be sure to grab the correct version for your environment.)
4) After it downloads, double-click the EXE file to launch the installer.

5) Change the **Unzip to Folder** to your project base folder (**C:\Develop\** in my example**)** and click **Unzip**.
6) You will now have the ADO.NET components and sample code in a folder on your workstation. If you are using the .Net Framework, you will want to install the components into the Visual Studio envioronment as documented in the README file, which should look like this if it worked correctly:

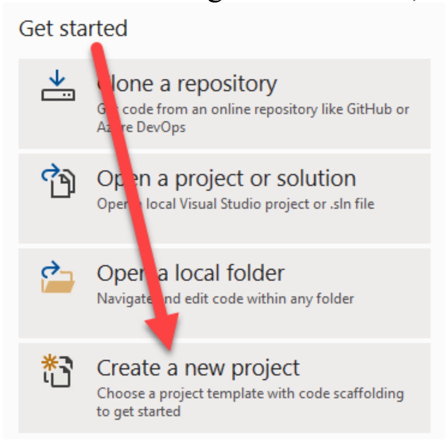


```
PS C:\develop\sdk15\packages\VisualStudioDataTools> .\Load-Zen-DataTools.ps1 Install

Running .\Load-Zen-DataTools.ps1 Install...
Name:  Server
...found version 15.11.005 installed.

Name                           Value
----                           -----
vsixInstaller                  C:\Program Files (x86)\Microsoft Visual Studio\2019\Professional\Common7\IDE\VSIXInst...
version                        16.11.32428.217
vsixPackage                    C:\develop\sdk15\packages\VisualStudioDataTools\Zen.VisualStudio.DataTools_2017_2019....
vsixID                         PSQLDataTools.d115023a-8951-4aa5-ae36-94d7b1e6f6ad
installDir                     C:\Program Files (x86)\Microsoft Visual Studio\2019\Professional
model1_inc                     C:\Program Files (x86)\Microsoft Visual Studio\2019\Professional\Common7\IDE\Extensio...
model1_t4                      C:\Program Files (x86)\Microsoft Visual Studio\2019\Professional\Common7\IDE\Extensio...
...found Visual Studio version '16.11.32428.217' at 'C:\Program Files (x86)\Microsoft Visual Studio\2019\Professional'.
...please wait, importing VS Extension package for PSQL ADO.NET Provider Data Tools.
   ...import complete, see the log file %temp%\Zen.DataTools.VSIX.install.log for details.
...successfully imported the Entity Framework Model First include templates.
...successfully imported the Entity Framework Model First T4 templates.
Finished running .\Load-Zen-DataTools.ps1 Install!
```

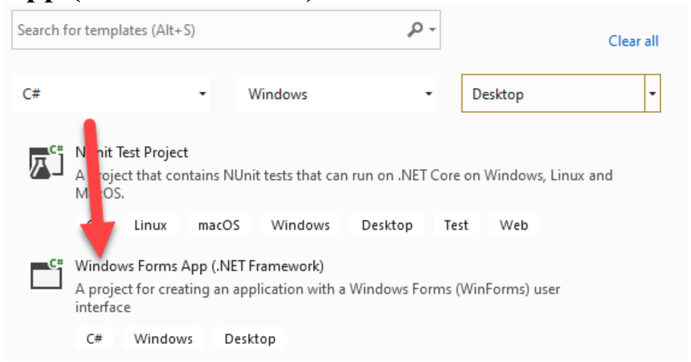
Create a New Visual Studio Project

We need to create a new project within Visual Studio with basic starter code.

1) Start Visual Studio.
2) From the Getting Started screen, select **Create a new project**.


Get started
Clone a repository
Get code from an online repository like GitHub or Azure DevOps
Open a project or solution
Open a local Visual Studio project or .sln file
Open a local folder
Navigate and edit code within any folder
Create a new project
Choose a project template with code scaffolding to get started

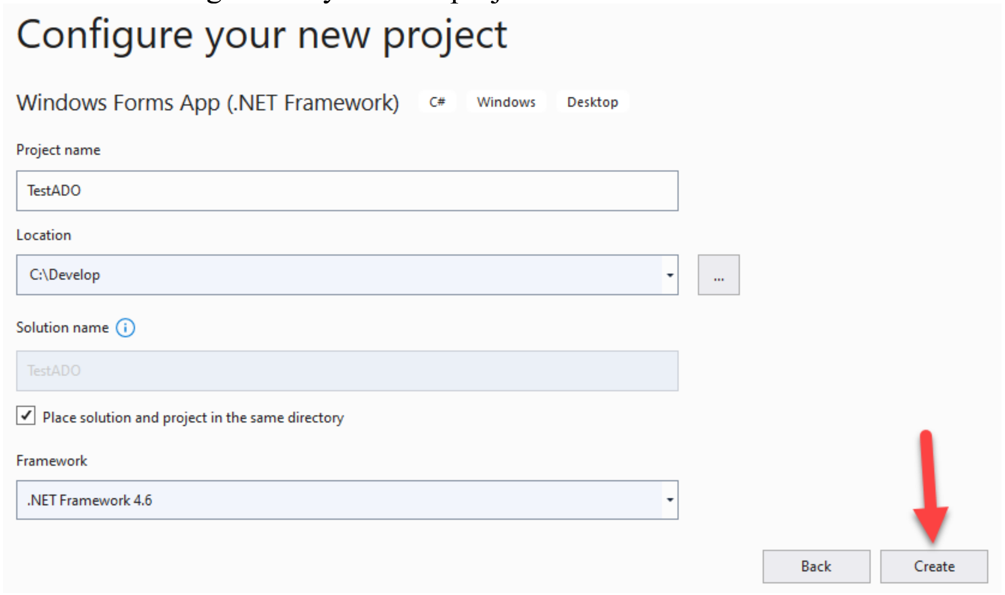
3) In the template search, select C#, Windows, and Desktop, and then select the **Windows Forms App (.Net Framework)** and click **Next**.



NUnit Test Project
A project that contains NUnit tests that can run on .NET Core on Windows, Linux and MacOS.
Linux macOS Windows Desktop Test Web

Windows Forms App (.NET Framework)
A project for creating an application with a Windows Forms (WinForms) user interface
C# Windows Desktop

4) From the next screen, enter your project name, location, and other required information and click **Create** to generate your new project with some starter code:



Configure your new project

Windows Forms App (.NET Framework) C# Windows Desktop

Project name: TestADO

Location: C:\Develop

Solution name: TestADO

☑ Place solution and project in the same directory

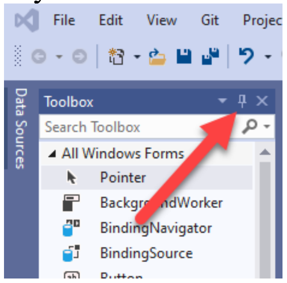
Framework: .NET Framework 4.6

5) If you wish, you can build this program and it will run (though it won't do anything yet).

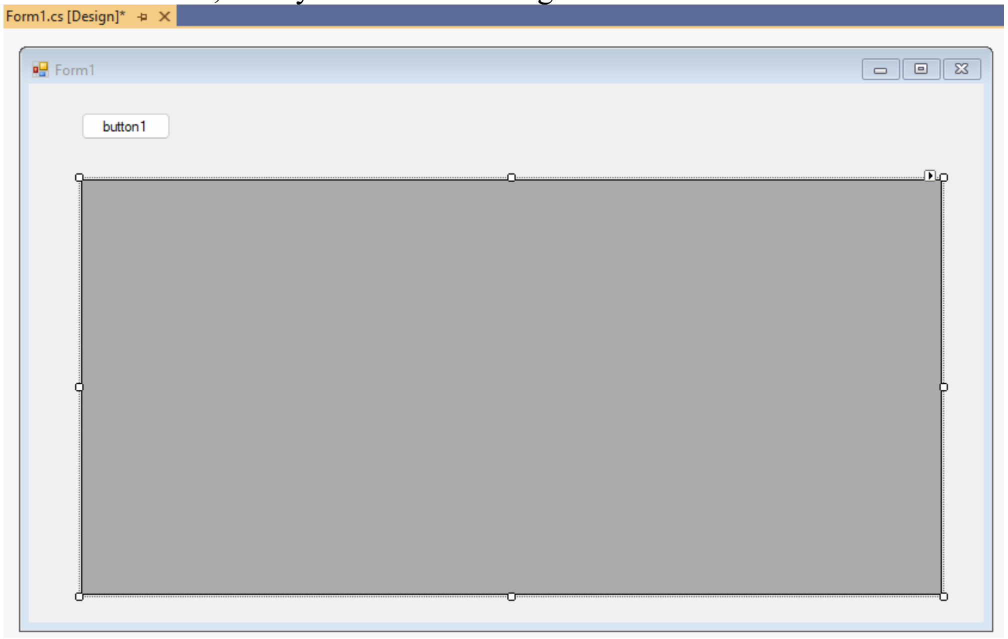
Add the Visual Elements to the Form

We are going to drop two visual elements onto the form.

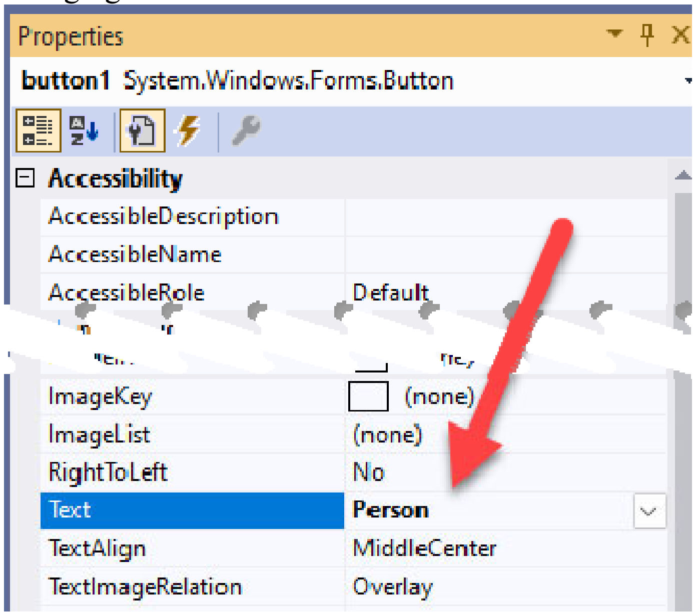
1) If your Toolbox is not pinned to the screen, consider pinning it now with this button:



2) Drag & drop one **Button** object and one **DataGridView** object onto your new form. Align and resize as desired, until you have something like this:



3) To make it more "pretty", click on the **button1** and change its Text in the Properties window. In our case, we are going to create a query that reads data from the Person table, so we are changing the text to "**Person**":

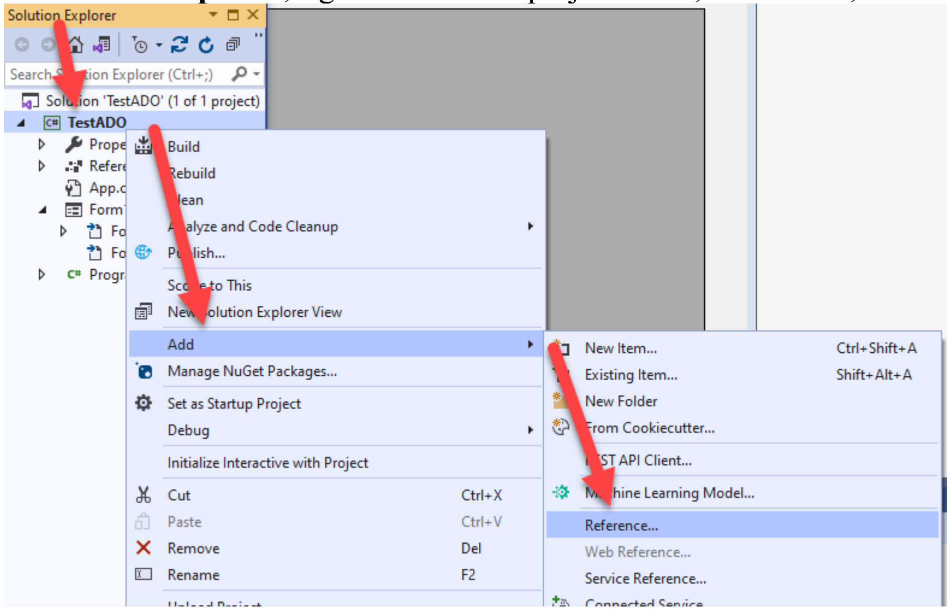


AccessibleDescription	
AccessibleName	
AccessibleRole	Default
ImageKey	(none)
ImageList	(none)
RightToLeft	No
Text	Person
TextAlign	MiddleCenter
TextImageRelation	Overlay

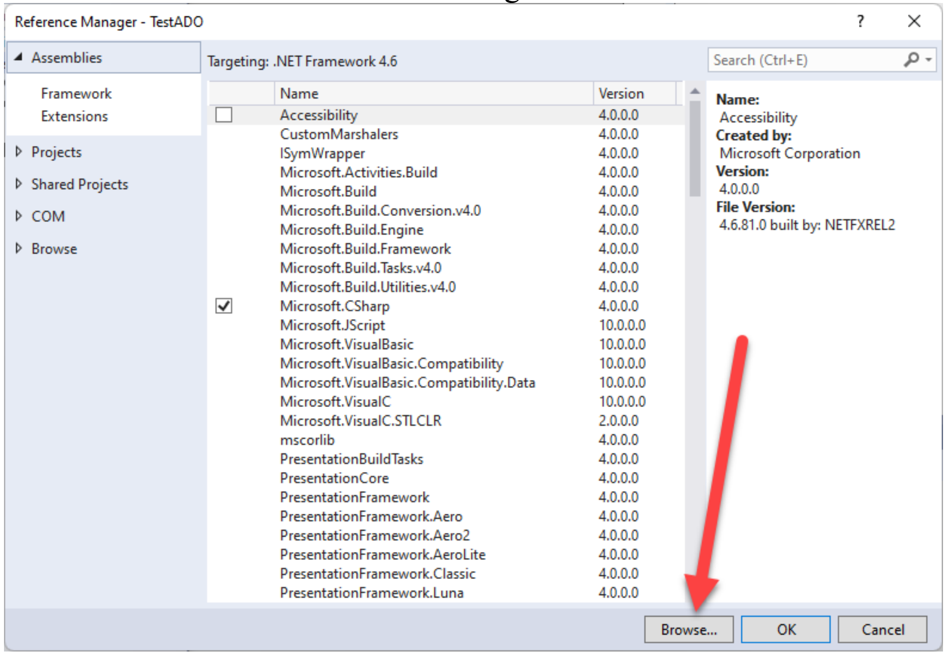
Add the ADO.Net Provider to the Project

We are next going to add the needed provider to the environment. Note that for this to work, you do NOT need to have the SDK installed – the proper provider is already available as part of your Zen v15 installation.

1) In **Solution Explorer**, right-click on the project name, select **Add**, then **Reference…**:



2) Click the Browse button in the lower right:

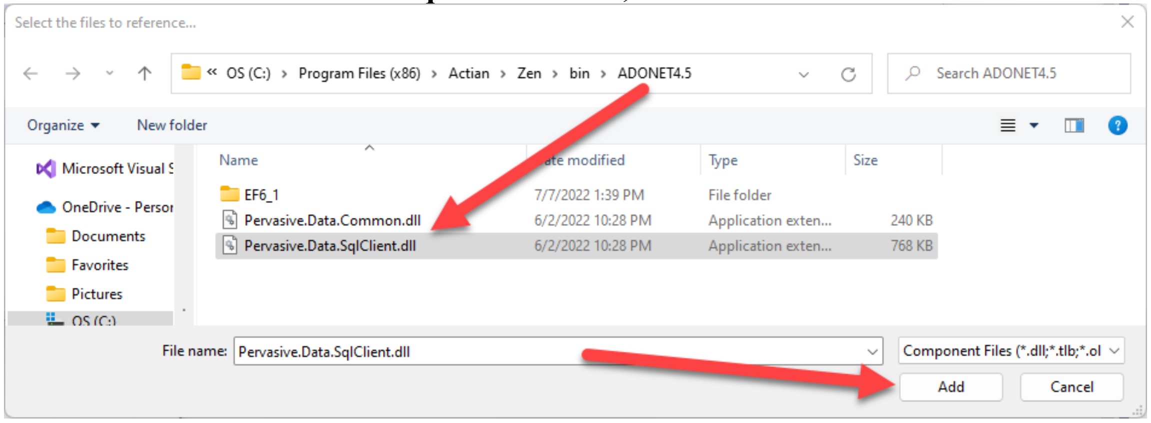


Targeting: .NET Framework 4.6

	Name	Version
☐	Accessibility	4.0.0.0
	CustomMarshalers	4.0.0.0
	ISymWrapper	4.0.0.0
	Microsoft.Activities.Build	4.0.0.0
	Microsoft.Build	4.0.0.0
	Microsoft.Build.Conversion.v4.0	4.0.0.0
	Microsoft.Build.Engine	4.0.0.0
	Microsoft.Build.Framework	4.0.0.0
	Microsoft.Build.Tasks.v4.0	4.0.0.0
	Microsoft.Build.Utilities.v4.0	4.0.0.0
✓	Microsoft.CSharp	4.0.0.0
	Microsoft.JScript	10.0.0.0
	Microsoft.VisualBasic	10.0.0.0
	Microsoft.VisualBasic.Compatibility	10.0.0.0
	Microsoft.VisualBasic.Compatibility.Data	10.0.0.0
	Microsoft.VisualC	10.0.0.0
	Microsoft.VisualC.STLCLR	2.0.0.0
	mscorlib	4.0.0.0
	PresentationBuildTasks	4.0.0.0
	PresentationCore	4.0.0.0
	PresentationFramework	4.0.0.0
	PresentationFramework.Aero	4.0.0.0
	PresentationFramework.Aero2	4.0.0.0
	PresentationFramework.AeroLite	4.0.0.0
	PresentationFramework.Classic	4.0.0.0
	PresentationFramework.Luna	4.0.0.0

Name:
Accessibility
Created by:
Microsoft Corporation
Version:
4.0.0.0
File Version:
4.6.81.0 built by: NETFXREL2

3) Navigate to the directory containing the ADO.NET Provider, as shown in the dialog box below, and select the **Pervasive.Data.SqlClient.dll** file, then click **Add**:


Select the files to reference...
« OS (C:) › Program Files (x86) › Actian › Zen › bin › ADONET4.5
Search ADONET4.5
Organize
New folder
Microsoft Visual S
OneDrive - Persor
Documents
Favorites
Pictures
OS (C:)
Component Files (*.dll;*.tlb;*.ol
Add
Cancel

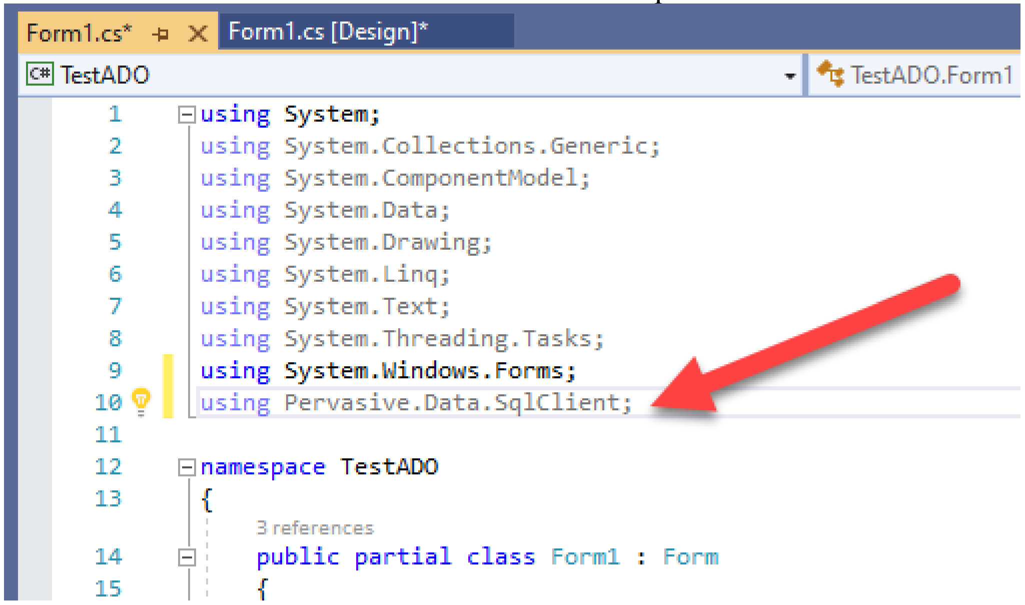
Name	Date modified	Type	Size
EF6_1	7/7/2022 1:39 PM	File folder	
Pervasive.Data.Common.dll	6/2/2022 10:28 PM	Application exten...	240 KB
Pervasive.Data.SqlClient.dll	6/2/2022 10:28 PM	Application exten...	768 KB

File name: Pervasive.Data.SqlClient.dll

Specify the Button Fucntionality

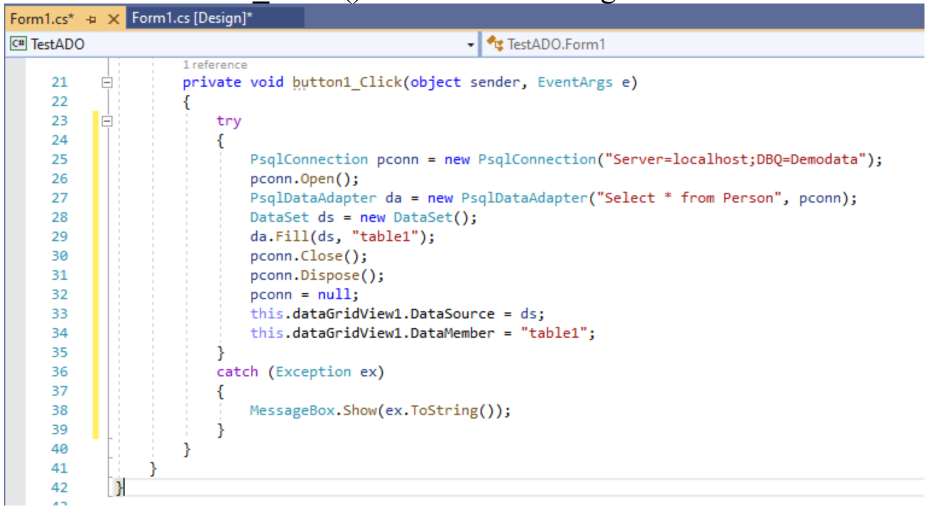
We are next going to add functionality to the button.

1) Double-click on your **Person** button to open the code window, and add the "using" line as shown here in Line 10 to allow it to work with the provider:



```
1   using System;
2   using System.Collections.Generic;
3   using System.ComponentModel;
4   using System.Data;
5   using System.Drawing;
6   using System.Linq;
7   using System.Text;
8   using System.Threading.Tasks;
9   using System.Windows.Forms;
10  using Pervasive.Data.SqlClient;
11
12  namespace TestADO
13  {
14      public partial class Form1 : Form
15      {
```

2) Locate the **button1_Click**() function and change the code to match the following:



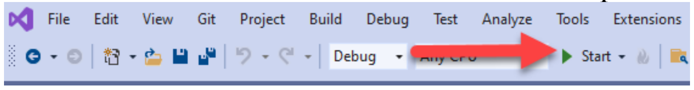
```
21          private void button1_Click(object sender, EventArgs e)
22          {
23              try
24              {
25                  PsqlConnection pconn = new PsqlConnection("Server=localhost;DBQ=Demodata");
26                  pconn.Open();
27                  PsqlDataAdapter da = new PsqlDataAdapter("Select * from Person", pconn);
28                  DataSet ds = new DataSet();
29                  da.Fill(ds, "table1");
30                  pconn.Close();
31                  pconn.Dispose();
32                  pconn = null;
33                  this.dataGridView1.DataSource = ds;
34                  this.dataGridView1.DataMember = "table1";
35              }
36              catch (Exception ex)
37              {
38                  MessageBox.Show(ex.ToString());
39              }
40          }
41      }
42  }
```

(See Appendix A if you want to copy/paste the text instead of typing it.)

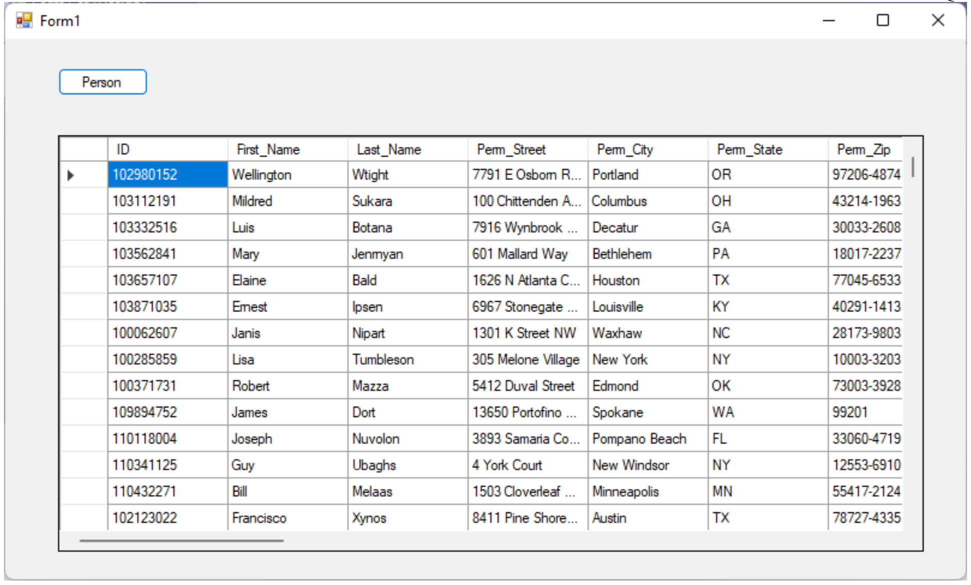
Build and Test the Application

Your application is now complete! To test it:

1) Click on the Start button on the toolbar to compile and launch the app:



2) Click on the Person button to load the DataGrid with the data from the SQL table called Person:



ID	First_Name	Last_Name	Perm_Street	Perm_City	Perm_State	Perm_Zip
102980152	Wellington	Wtight	7791 E Osbom R...	Portland	OR	97206-4874
103112191	Mildred	Sukara	100 Chittenden A...	Columbus	OH	43214-1963
103332516	Luis	Botana	7916 Wynbrook ...	Decatur	GA	30033-2608
103562841	Mary	Jenmyan	601 Mallard Way	Bethlehem	PA	18017-2237
103657107	Elaine	Bald	1626 N Atlanta C...	Houston	TX	77045-6533
103871035	Emest	Ipsen	6967 Stonegate ...	Louisville	KY	40291-1413
100062607	Janis	Nipart	1301 K Street NW	Waxhaw	NC	28173-9803
100285859	Lisa	Tumbleson	305 Melone Village	New York	NY	10003-3203
100371731	Robert	Mazza	5412 Duval Street	Edmond	OK	73003-3928
109894752	James	Dort	13650 Portofino ...	Spokane	WA	99201
110118004	Joseph	Nuvolon	3893 Samaria Co...	Pompano Beach	FL	33060-4719
110341125	Guy	Ubaghs	4 York Court	New Windsor	NY	12553-6910
110432271	Bill	Melaas	1503 Cloverleaf ...	Minneapolis	MN	55417-2124
102123022	Francisco	Xynos	8411 Pine Shore...	Austin	TX	78727-4335

From there, the ADO.NET world is at your beck and call!

A Few Notes About the Source Code

Now that we have built the code and it is working, let's review a few key components of that source code:

- Line 25 instantiates a new PsqlConnection object called **pconn**, which defines the SQL connection to be used. In this example, we are connecting to the database engine on the local machine (i.e. localhost) and opening up the DEMODATA database. You can change the connection string to connect to any database on any accessible Zen database server by simply changing this text. If your database is secured, you may need to add the UID and PWD options. See the ODBC Connection String documentation in the on-line help for a complete list of connection string options.

- Line 26 opens the connection. Any good application would use a separate *try* block to capture errors here in case the server is down or the database is otherwise unavailable.
- Line 27 instantiates a new PsqlDataAdapter object called **da**, connects it to the existing PsqlConnection object **pconn**, and assigns it a static SQL query as shown. To increase flexibility, you may wish to perform these operations separately.
- Line 28 instantiates a new DataSet object called **ds**.
- Line 29 handles the database magic here – it calls the PsqlDataAdapter object to fill the given data set object with data, and then assign that to the **"table1"** table name which will be assigned to the dataGridView1 object later on.
- Lines 30-32 close the database connection and dispose of the connection. You may not want to do this in your application until it is ready to close completely – but like we learned in kindergarten, you should always clean up your mess before going home.
- Lines 33-34 take the DataSet object that we created and populated in lines 28-29 and assign it to the DataGridView object on the form. [Note that if you were playing around and have created multiple DataGridView objects, you may need to change the name of this object to match the one currently showing in the Form Design.]

Expanding Your Knowledge

Want to know if you understand what is going on? Try these modifications:

1) Add a second button to the form called Faculty, and query from the Faculty table instead of the Person table. You will need to populate the button2_Click function.
2) Add a third button that displays each Faculty member's name and Salary, sorted by Last Name. If you are not familiar with the DEMODATA database, we'll give you the SQL query here:
   ```
   Select Person.First_Name, Person.Last_Name, Faculty.Salary from Person INNER JOIN
   Faculty on (Person.id = Faculty.Id) ORDER BY 2
   ```
3) Change your application to connect to your own database and your own tables. This should only require changing the connection string and the SQL statements (lines 25 and 27).

From here, you're only limited by your imagination!

Finding More Help

If you have other problems getting this to work, you can contact Actian directly through their web forums at https://communities.actian.com/s/ for more help. If you need some additional hand-holding, Goldstar Software may be able to assist you as well through our Developer Jump Start Program. You can contact us at 1-708-647-7665or via the web at http://www.goldstarsoftware.com.

Appendix A: Sample Form1.cs Code

The following application makes use of the ADO.NET API and can be used as instructional and sample code.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Pervasive.Data.SqlClient;

namespace TestADO
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            try
            {
                PsqlConnection pconn = new PsqlConnection("Server=localhost;DBQ=Demodata");
                pconn.Open();
                PsqlDataAdapter da = new PsqlDataAdapter("Select * from Person", pconn);
                DataSet ds = new DataSet();
                da.Fill(ds, "table1");
                pconn.Close();
                pconn.Dispose();
                pconn = null;
                this.dataGridView1.DataSource = ds;
                this.dataGridView1.DataMember = "table1";
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.ToString());
            }
        }
    }
}
```