Adding Databases and Application Servers to your Developer Studio Environment

To develop and test a distributed application, it is a best practice to add a database and an application server to your development environment.
Our application requires a database and we see here that the Customer Business Entity class requires the Customer table in the database. This code will not compile without errors unless the project has a connection to the database.

First, let's focus on adding a database. Typically, you make a copy of the database so that changes that you may make to it during development will not affect other developers. To get started, we run proenv which opens a window in the OpenEdge\wrk directory.

As you can see OpenEdge\wrk contains files related to OpenEdge server processes.

We use prodb to copy a database into this folder. In this case, we copy the sample database that comes with OpenEdge named sports2000.

[proserve sports2000 –S 9999]
Because we want multiple connections to this database, we must start it as a database server process. To do this we use the proserve command where we specify the name of the database and the port number.

The database server will remain running until the system shuts down or you stop it using the proshut utility.

Now that we have a database server running, we can connect to it from the server project in our workspace. To do this, we configure the database connection as a workspace preference. To begin, we open the wizard for configuring a new database connection.

We name the connection and select the database. We must specify the hostname of localhost and the same port number that was used to start the database server.

A best practice is to test the connection. Testing the connection requires an AVM. We select a project, and then click OK to run the test.

Once the connection test succeeds, we continue with the configuration. An SQL connection is required by Developer Studio to run some of the utility plugins associated with accessing a database, so we must also configure an SQL connection. In the wizard, we accept the defaults.

Then we test the SQL connection.

On the last screen of the database connection wizard, notice that the box is checked to auto-start the database server. You should leave this box checked so that when Developer Studio starts, it will automatically start the database server for you.

As you can see, the sports2000 database connection is defined for the workspace. Next, we need to associate the database connection with our Server project.

We open the properties for the Server project and select the database connection.

[GetCustomerName.p]
Next, we must recompile our application code so that the database tables used by the server code are recognized. As you can see, database table names and fields are now displayed in orange.

Next, let's add an application server to our development environment. We will use the classic AppServer so we can make use of its debugging capabilities. First, we must ensure that we have a connection to the Admin Server. We open the workspace preference area and navigate to the Server area for OpenEdge Explorer.

Here we see the default connection to the Admin Server for OpenEdge Explorer. If we click Test Connection and the test fails, then we must enter the correct admin password and retest.

Now our connection is successful, so we click **create servers and finish,** and then click **OK.**

We are now ready to manage our server processes from Developer Studio. To make the servers view appear, we select it in our current perspective.

To view the server processes if they are not visble, you must select the Servers view.

The servers view shows the OpenEdge server processes available in our development environment.

We will use the classic AppServer named **asbroker1** as our application server. We double-click this server to open the Server Editor.

We want to configure how this application server starts and where it will find our server code. We click **Open launch configuration**.

First, in the Startup tab, we specify where the application server will log its messages. A best practice is to use the OpenEdge\wrk directory.

Next, we add the server project directories to the PROPATH for the application server.

Finally, we want the application server to open a connection to the database server, so we select our already-configured database connection.

We save our runtime configuration and start the application server. We wait a moment for the server to start.

Now let's test the application. We see that it successfully retrieves the data from the database using the application server.
We have now shown you how you can add an OpenEdge database and an application server to your development environment.

To learn about developing OpenEdge ABL applications, take the course, Developing a Progress OpenEdge ABL Application.