ADO.NET

# What is ADO.NET?

ADO.NET is not a different technology. In simple terms, we can think of ADO.NET, **as a set of classes (Framework), that can be used to interact with data sources like Databases and XML files**. This data can, then be consumed in any.NET application. **ADO stands for Microsoft ActiveX Data Objects**.

The following are, a few of the different types of.NET applications that use ADO.NET to connect to a database, execute commands, and retrieve data.

* ASP.NET Web Applications
* Windows Applications
* Console Applications

## Dot Net Data Providers:

* Data Provider for SQL Server-System.Data.SqlClient
* Data Provider for Oracle - System.Data.OracleClient
* Data Provider for OLEDB-System.Data.OleDb
* Data Provider for ODBC-System.Data.Odbc

All the required classes are residing inside the Namespaces.

## Steps

1. Need to established an **SQLConnection** **con**.
2. Need an **SQLCommand** **cmd**, with the connection object **con**.
3. Need to open the connection
4. Need **ExecuteReader** thatwill execute the **SQLCommand** and retrieve the data. This will return the data as a SQL data format. So, we may need an **SQLDataReader** object get the data in the for of datatable.
5. Do the work/ create Dataset/ etc.
6. Close Connection
7. Dispose the connection

using System.Data.SqlClient;

SqlConnection con = new SqlConnection("data source=.; database=MySQLDB; integrated security=SSPI");

SqlCommand cmd = new SqlCommand("Select \* from tmp\_costing\_data", con);

con.Open();

SqlDataReader rdr=cmd.ExecuteReader();

GridView1.DataSource = rdr;

GridView1.DataBind();

con.Close();

con.Dispose();

using System.Data.OracleClient;

OracleConnection con = new OracleConnection("Data Source=TMCTEST;User ID=tmclive;pwd=tmclive;integrated security = no");

OracleCommand cmd = new OracleCommand("Select \* from tmp\_costing\_data", con);

con.Open();

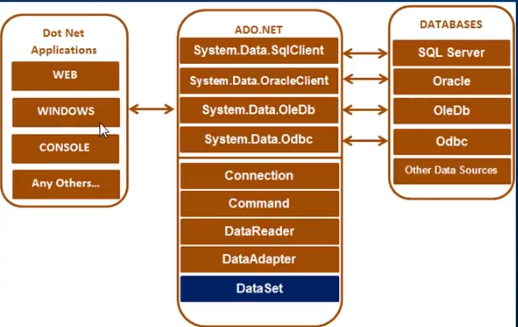
OracleDataReader rdr = cmd.ExecuteReader();

GridView1.DataSource = rdr;

GridView1.DataBind();

con.Close();

con.Dispose();



# SQLConnection/OracleConnection objects

SqlConnection() or OracleConnection() have two constructors. The first one is ParameterLess and the second one is with one parameter with connection string. So, either of this can be used as the following way :

string connectionString = "Data Source=TMCTEST;User ID=tmclive;pwd=tmclive;integrated security = no";

OracleConnection conOr = new OracleConnection();

conOr.ConnectionString = connectionString;

or

SqlConnection con = new SqlConnection("data source=.;database=MySQLDB;integrated security=SSPI");

Once a is SqlConnection opened, it should be closed, as soon as the data get fetched. If there is an exception while fetching data from

**OracleDataReader** rdr = cmd.ExecuteReader();

then it will be opened. The good practice is to open the SqlConnection as late as possible and close as soon as possible. If did not closed, then the scalability of the program will be badly affected.

So, the good practice is, try- catch- finally

SqlConnection con = new SqlConnection("data source=.;database=MySQLDB;integrated security=SSPI");

SqlCommand cmd = new SqlCommand("Select TMP\_MR\_NUM Patient\_Number,TMP\_PAT\_NUM MR\_NUMBER,TMP\_EFF\_DATE DateSQL from tmp\_costing\_data order by tmp\_mr\_num asc", con);

try

{

con.Open();

SqlDataReader rdr = cmd.ExecuteReader();

GridView1.DataSource = rdr;

GridView1.DataBind();

}

catch

{

}

finally

{

con.Close();

}

Or

string connectionString = "Data Source=TMCTEST;User ID=tmclive;pwd=tmclive;integrated security = no";

using (OracleConnection conOr = new OracleConnection(connectionString))

{

OracleCommand cmdOr = new OracleCommand("Select TMP\_MR\_NUM Patient\_Number,TMP\_PAT\_NUM MR\_NUMBER,TMP\_EFF\_DATE DateOracle from tmp\_costing\_data order by tmp\_mr\_num desc", conOr);

conOr.Open();

OracleDataReader rdrOr = cmdOr.ExecuteReader();

GridView2.DataSource = rdrOr;

GridView2.DataBind();

}

1. Connections should be opened as late as possible, and should be closed as early as possible.

2. Connections should be closed in the finally block, or using, the using statement. No need to explicitly mention

con.Close();

# Connection Strings and Web.config

In Web.config inside configuration define connectionStrings and then give a name, the connection string and the provider name

<configuration>

<connectionStrings>

<add name="DBCSSql" connectionString="data source=.;database=MySQLDB;integrated security=SSPI" providerName="System.Data.SqlClient" />

<add name="DBCSOracle" connectionString="Data Source=TMCTEST;User ID=tmclive;pwd=tmclive;integrated security = no" providerName="System.Data.OracleClient" />

</connectionStrings>

Then in aspx.cs

protected void Page\_Load(object sender, EventArgs e)

{

string ConnSQL= ConfigurationManager.ConnectionStrings["DBCSSql"].ConnectionString;

SqlConnection con = new SqlConnection(ConnSQL);

SqlCommand cmd = new SqlCommand("Select TMP\_MR\_NUM Patient\_Number,TMP\_PAT\_NUM MR\_NUMBER,TMP\_EFF\_DATE DateSQL from tmp\_costing\_data order by tmp\_mr\_num asc", con);

try

{

con.Open();

SqlDataReader rdr = cmd.ExecuteReader();

GridView1.DataSource = rdr;

GridView1.DataBind();

}

catch

{

}

finally

{

con.Close();

}

using (OracleConnection conOr = new OracleConnection(ConfigurationManager.ConnectionStrings["DBCSOracle"].ConnectionString))

{

OracleCommand cmdOr = new OracleCommand("Select TMP\_MR\_NUM Patient\_Number,TMP\_PAT\_NUM MR\_NUMBER,TMP\_EFF\_DATE DateOracle from tmp\_costing\_data order by tmp\_mr\_num desc", conOr);

conOr.Open();

OracleDataReader rdrOr = cmdOr.ExecuteReader();

GridView2.DataSource = rdrOr;

GridView2.DataBind();

}

}

For windows application, instead web,config, we may define in app.config.

# SQLCommand/OracleCommand

SqlCommand class is used to prepare an SQL statement or Stored Procedure that we want to execute on a SQL Server database.

The most commonly used methods of the SqlCommand class

1. **ExecuteReader** - Use when the T-SQL statement returns more than a single value. For example, if the query returns rows of data.

string connectionString= ConfigurationManager.ConnectionStrings["DBCSOracle"].ConnectionString;

using (OracleConnection con = new OracleConnection(connectionString))

{

OracleCommand command = new OracleCommand("Select TMP\_MR\_NUM Patient\_Number,TMP\_PAT\_NUM MR\_NUMBER,TMP\_EFF\_DATE DateOracle from tmp\_costing\_data order by tmp\_mr\_num asc", con);

con.Open();

OracleDataReader rdr = command.ExecuteReader();

GridView1.DataSource = rdr;

GridView1.DataBind();

}

//or

string connectionStringSql = ConfigurationManager.ConnectionStrings["DBCSSql"].ConnectionString;

using (SqlConnection conSql = new SqlConnection(connectionStringSql))

{

SqlCommand commandSql = new SqlCommand();

commandSql.CommandText = "Select TMP\_MR\_NUM Patient\_Number,TMP\_PAT\_NUM MR\_NUMBER,TMP\_EFF\_DATE DateSQL from tmp\_costing\_data order by tmp\_mr\_num desc";

commandSql.Connection = conSql;

conSql.Open();

GridView2.DataSource = commandSql.ExecuteReader();

GridView2.DataBind();

}

1. **ExecuteNonQuery** - Use when you want to perform an Insert, Update or Delete operation.
2. **ExecuteScalar** - Use when the query returns a single(scalar) value. For example, queries that return the total number of rows in a table. In case of dataset, it will return only the first column of the first row. Else will be ignored.

string connectionString = ConfigurationManager.ConnectionStrings["DBCSOracle"].ConnectionString;

using (OracleConnection con = new OracleConnection(connectionString))

{

OracleCommand command = new OracleCommand("Select count(1) from tmp\_costing\_data t order by tmp\_mr\_num asc", con);

con.Open();

lblOracle.Text ="Oracle Count:" + command.ExecuteScalar().ToString();

}

//or

string connectionStringSql = ConfigurationManager.ConnectionStrings["DBCSSql"].ConnectionString;

using (SqlConnection conSql = new SqlConnection(connectionStringSql))

{

SqlCommand commandSql = new SqlCommand();

commandSql.CommandText = "Select count(tmp\_pat\_num) from tmp\_costing\_data t ";

commandSql.Connection = conSql;

conSql.Open();

int rowCount = (int)commandSql.ExecuteScalar();

lblSql.Text = "Sql Count:" + rowCount;

}