**In this video session we will learn about**  
**1.** Storing connection strings  in a configuration file. For example, web.config for an asp.net web application and app.config for windows application  
**2.** Reading the connection strings from web.config and app.config files.  
**3.** Disadvantages of storing connection strings in application code.  
**4.** Advantages of storing connection string in configuration files - web.config and app.config.   
  
   
  
[**In Part 2**](http://csharp-video-tutorials.blogspot.com/2012/10/sqlconnection-in-adonet-part-2.html), of this video series, we have discussed about the **SqlConnection**object. **SqlConnection** object uses connection string. The connection strings were hard coded in the examples that we worked with in Part 2.   
  
**There are 2 issues with hard coding the connection strings in application code**  
**1.** For some reason, if we want to point our application to a different database server, we will have to change the application code. If you change application code, the application requires a re-build and a re-deployment which is a time waster.  
**2.** All the pages that has the connection string hard coded needs to change. This adds to the maintenance overhead and is also error prone.  
  
**In real time, we may point our applications from time to time, from Development database to testing database to UAT database.**  
  
**Because of these issues, the best practice is to store the connection in the configuration file**, from which all the pages can read and use it. This way we have only one place to change, and we don't have to re-build and re-deploy our application. This saves a lot of time.  
  
**In an asp.net web application, the configuration strings can be stored in web.config file**, as shown below. Give a meaningful name to your connection string. Since we are working with sql server, the provider name is **System.Data.SqlClient.**  
**<connectionStrings>**  
**<add name="DatabaseConnectionString"**  
**connectionString="data source=.; database=Sample\_Test\_DB; Integrated Security=SSPI"**  
**providerName="System.Data.SqlClient" />**  
**</connectionStrings>**   
  
   
  
**How to read the connection string from web.config file?**  
Use the **ConnectionStrings**property of the **ConfigurationManager** class to retrieve the connection string value from **web.config**. **ConfigurationManager** class is present in **System.Configuration** namespace.  
protected void Page\_Load(object sender, EventArgs e)  
{  
    string ConnectionString = ConfigurationManager.co .ConnectionStrings["DatabaseConnectionString"].ConnectionString;  
    using (SqlConnection connection = new SqlConnection( ConnectionString ))  
    {  
        SqlCommand cmd = new SqlCommand("Select \* from tblProductInventory", connection);  
        connection.Open();  
        GridView1.DataSource = cmd.ExecuteReader();  
        GridView1.DataBind();  
    }  
}  
  
**The configuration file in a windows application is App.config.** Storing connection strings in **App.config** is similar to **web.config**. The same **ConfigurationManager** class can be used to read connection string from **App.config** file. The example below, shows how to read connection strings from App.config file, and bind the data to a DataGridview control in a windows application.  
private void Form1\_Load(object sender, EventArgs e)  
{  
    string ConnectionString = ConfigurationManager.ConnectionStrings["DatabaseConnectionString"].ConnectionString;  
    using (SqlConnection connection = new SqlConnection( ConnectionString ))  
    {  
        SqlCommand cmd = new SqlCommand("Select \* from tblProductInventory", connection);  
        connection.Open();  
        BindingSource source = new BindingSource();  
        source.DataSource = cmd.ExecuteReader();  
        dataGridView1.DataSource = source;  
    }  
}