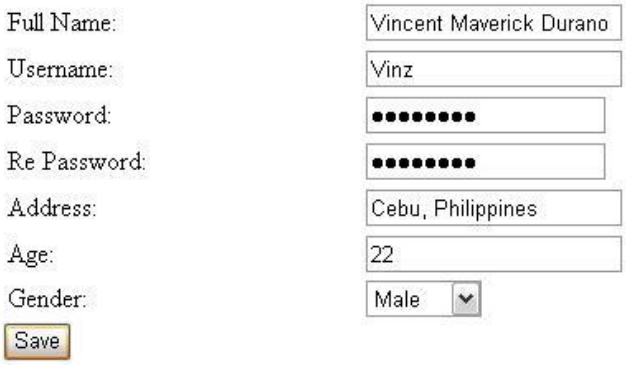
CREATING A SIMPLE REGISTRATION FORM IN ASP.NET (PROGRAMMING)

This example shows how to create a very simple registration form in **ASP.NET** WebForms.



Simple Form in ASP.NET

STEP 1: CREATING THE DATABASE

The following are the basic steps on how to create a simple database in the Sql Server:

Launch **Sql Server Management Studio Express** and then connect;

Expand the **Databases** folder from the Sql Server object explorer;

Right click on the Databases folder and select "**New Database**";

From the pop up window, input the database name you like and click add;

Expand the **Database** folder that you have just added;

Right click on the **Tables** folder and select "**New Table**";

Then add the following fields below:

Column Name	Data Type	Allow Nulls
Id	int	☐
Name	varchar(50)	☑
UserName	varchar(50)	☑
Password	varchar(50)	☑
Gender	char(10)	☑
Age	int	☑
Address	varchar(50)	☑
		☐

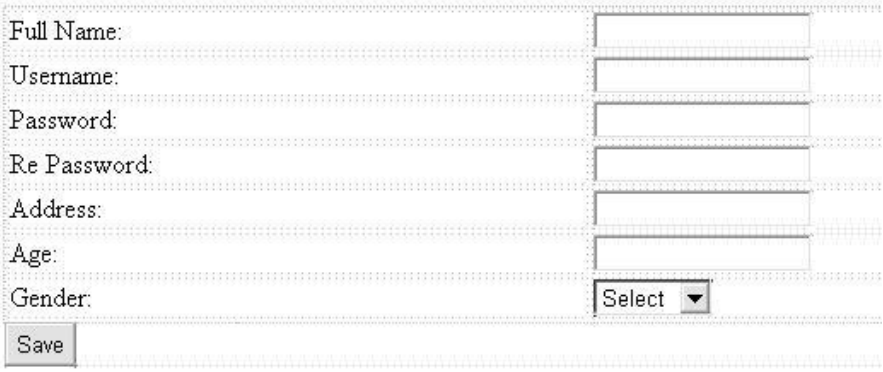
Simple Form in ASP.NET (Database)

Note: in this demo, I set the **Id** to auto increment so that the id will be automatically generated for every new added row. To do this select the Column name "**Id**" and in the column properties set the "**Identity Specification**" to yes.

Then after adding all the necessary fields, name your **Table** the way you like. Note that in this demo I name it "**tblRegistration**".

STEP 2: SETTING UP THE UI

For the simplicity of this demo, I set up the UI like below in the **WebForm**:



Simple form registration in ASP.NET (User Interface)

ASPX:

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<title>Sample Registration Page</title>
<style type="text/css">
.style1
{
width: 100%;
}
</style>
</head>
<body>
<form id="form1" runat="server">
<div>
<table>
<tr>
<td>Full Name:</td>
<td>
<asp:TextBox ID="TxtName runat="server"></asp:TextBox>
</td>
</tr>
```

```
<tr>
<td>Username:</td>
<td>
<asp:TextBox ID="TxtUserName" runat="server"></asp:TextBox>
</td>
</tr>
<tr>
<td>Password:</td>
<td>
<asp:TextBox ID="TxtPassword" runat="server"
TextMode="Password"></asp:TextBox>
</td>
</tr>
<tr>
<td>Re Password:</td>
<td>
<asp:TextBox ID="TxtRePassword" runat="server"
TextMode="Password"></asp:TextBox>
</td>
</tr>
<tr>
<td>Address:</td>
<td>
<asp:TextBox ID="TxtAddress" runat="server"></asp:TextBox>
</td>
</tr>
<tr>
<td>Age:</td>
<td>
<asp:TextBox ID="TxtAge" runat="server"></asp:TextBox>
</td>
</tr>
<tr>
<td>Gender:</td>
<td>
<asp:DropDownList ID="DropDownList1" runat="server"
AppendDataBoundItems="true">
<asp:ListItem Value="-1">Select</asp:ListItem>
<asp:ListItem>Male</asp:ListItem>
```

```
<asp:ListItem>Female</asp:ListItem>

</asp:DropDownList>

</td>

</tr>

</table>

</div>

<asp:Button ID="Button1" runat="server" Text="Save"

onclick="Button1_Click" />

</form>

</body>

</html>
```

As you can see, the UI is very simple. Now let's set up the connection string.

STEP 3: SETTING UP THE CONNECTION STRING

In your web.config file set up the connection string there as shown below:

```
<connectionStrings>

<add name="MyConsString" connectionString="Data Source=WPHVD185022-900;

Initial Catalog=MyDatabase;

Integrated Security=SSPI;"

providerName="System.Data.SqlClient" />

</connectionStrings>
```

Note: MyConsString is the name of the Connection String that we can use as a reference in our codes for setting the connection string later.

STEP 4: CALLING UP THE CONNECTIONSTRING IN OUR CODES

Here's the method for calling the connection string that was set up in the **web.config** file.

```
public string GetConnectionString(){

//sets the connection string from your web config file "ConnString" is the name of your Connection
String

return

System.Configuration.ConfigurationManager.ConnectionStrings["MyConsString"].ConnectionString;
}
```

STEP 5: WRITING THE METHOD FOR INSERTING THE DATA FROM THE REGISTRATION PAGE TO THE DATABASE

In this demo, we are using the ADO.NET objects for manipulating the data from the page to the database. If you are not familiar with ADO.NET then I would suggest you to refer the following link below to get started:

ADO.NET Tutorial

Here's the code block for inserting the data to the database.

```
private void ExecuteInsert(string name, string username, string password, string gender, string age,
string address)

{

SqlConnection conn = new SqlConnection(GetConnectionString());

string sql = "INSERT INTO tblRegistration (Name, UserName, Password, Gender, Age, Address) VALUES "

+ " (@Name,@UserName,@Password,@Gender,@Age,@Address)";

try

{

conn.Open();

SqlCommand cmd = new SqlCommand(sql, conn);

SqlParameter[] param = new SqlParameter[6];

//param[0] = new SqlParameter("@id", SqlDbType.Int, 20);

param[0] = new SqlParameter("@Name", SqlDbType.VarChar, 50);

param[1] = new SqlParameter("@UserName", SqlDbType.VarChar, 50);

param[2] = new SqlParameter("@Password", SqlDbType.VarChar, 50);

param[3] = new SqlParameter("@Gender", SqlDbType.Char, 10);

param[4] = new SqlParameter("@Age", SqlDbType.Int, 100);

param[5] = new SqlParameter("@Address", SqlDbType.VarChar, 50);

param[0].Value = name;

param[1].Value = username;

param[2].Value = password;

param[3].Value = gender;

param[4].Value = age;

param[5].Value = address;

for (int i = 0; i < param.Length; i++)

{

cmd.Parameters.Add(param[i]);

}

cmd.CommandType = CommandType.Text;

cmd.ExecuteNonQuery();

}
```

```
catch (System.Data.SqlClient.SqlException ex)
{
string msg = "Insert Error:";
msg += ex.Message;
throw new Exception(msg);
}
finally
{
conn.Close();
}
```

STEP 6: CALLING THE METHOD EXECUTEINSERT()

You can call the method above at Button_Click event for saving the data to the database. Here's the code block below:

```
protected void Button1_Click(object sender, EventArgs e){
if (TxtPassword.Text == TxtRePassword.Text)
{
//call the method to execute insert to the database
ExecuteInsert(TxtName.Text,
TxtUserName.Text,
TxtPassword.Text,
DropDownList1.SelectedItem.Text,
TxtAge.Text, TxtAddress.Text);
Response.Write("Record was successfully added!");
ClearControls(Page);
}
else
{
Response.Write("Password did not match");
TxtPassword.Focus();
}
}
```

As you can see from the above code block, we check the value of the **TxtPassword** and **TxtRePassword** to see if match. If it match then call the method **ExecuteInsert** else display the error message stating that the "Password did not match".

You also noticed that we call the method **ClearControls** for clearing the Text fields in the page. See the code block below for the **ClearControls** method:

```
public static void ClearControls(Control Parent){
if (Parent is TextBox)
{ (Parent as TextBox).Text = string.Empty; }
else
{
foreach (Control c in Parent.Controls)
ClearControls(c);
}
}
```

That's it! Hope you will find this example useful!